# Pricing Stochastic Volatility Models Using Random Grids

**Rishay Rajkumar**

A dissertation submitted to the Faculty of Commerce, University of Cape Town, in partial fulfilment of the requirements for the degree of Master of Philosophy.

January 5, 2022

*MPhil in Mathematical Finance,*
*University of Cape Town.*

# Declaration

I declare that this dissertation is my own, unaided work. It is being submitted for the Degree of Master of Philosophy in the University of the Cape Town. It has not been submitted before for any degree or examination in any other University.

January 5, 2022

# Abstract

Assets can be priced using a variety of numerical methods. In some instances, a particular numerical method may be more appropriate than others. If one method is used to calibrate the model to market conditions, but another method is used to price the asset, the results obtained may be inconsistent. This dissertation addresses the fundamental problem of this bias that is introduced when calibrating and pricing options using inconsistent methods. The random grids approach, developed by Andreasen and Huge (2011), is a pricing method that guarantees discrete consistency between calibration, finite difference solution and Markov-chain Monte-Carlo simulation based on the random grids approach. This dissertation provides a review and implementation of this random grids approach for pricing under the Heston model as well as the stochastic local volatility model. Consistent results are obtained for a call option under the various pricing methods using similar parameters as those used in the random grids paper. More specifically, when using a Heston model, consistent prices are obtained for the characteristic function pricing method, the backward finite difference method, the forward finite difference method as well as the Markov-chain Monte-Carlo method based on the random grids approach. Similarly, consistent prices are obtained under the stochastic local volatility model for the backward finite difference method, the forward finite difference method and the Markov-chain Monte-Carlo method based on the random grids approach.

# Acknowledgements

Firstly, I would like to thank my supervisor, Thomas McWalter, for the time and effort he has put in to help me with this dissertation. With 2020 marking the start of the Covid-19 global pandemic, this year has been unlike any other and I could not have done this without Tom's guidance. Secondly, I would like to thank my parents, Asha and Victor Rajkumar for supporting me through this difficult year. Their unconditional support has encouraged me to complete this dissertation to the best of my ability.

# Contents

# List of Figures

# List of Tables

# 1 Introduction

A common goal in mathematical finance is to model share price movements over time. Deterministic models are inadequate due to the fact that the share price is observed to move randomly through time. Therefore, stochastic processes are used to model the share price. Geometric Brownian motion is the most common stochastic process used to model the share price, whereby the share price at time $t$, denoted $S(t)$, is modelled using a random component as well as a deterministic component. This is described by the stochastic differential equation

$$dS(t) = \mu S(t)dt + \sigma S(t)dW(t),$$

where $\mu$ is the constant drift, $\sigma$ is the constant volatility and $W(t)$ is the standard Brownian motion process.

Since European options depend on the underlying share price, the share price model is instrumental in determining a fair option price. A computational advantage of pricing options using a geometric Brownian motion process is that a closed-form solution for the option price is available. This closed-form price was developed by Black and Scholes (1973). Another advantage of the Black-Scholes model, as Derman and Kani (1994) describe, is that the option can be hedged by taking a position in the underlying stock. The purpose of hedging an option is to minimise the loss incurred in the event that the option payoff is unfavourable.

Although the Black and Scholes (1973) model was universally respected for a long period of time, other models have been developed to represent existing market conditions more accurately. One characteristic commonly changed in other models is the fact that the Black and Scholes (1973) model uses a constant volatility. Although a constant volatility was applicable when the Black and Scholes (1973) model was created, recent market trends show that a constant volatility is not sufficient to model current market conditions. A consequence of this, as Andersen and Brotherton-Ratcliffe (1998) explain, is that it is impossible to calibrate the Black-Scholes model to the current market across all strikes.

Calibration is the process of varying the model parameters until they are able to fit the observed market values accurately. When calibrating a model, the volatility

that produces the correct option price under the Black-Scholes model is known as the implied volatility. Therefore, different share price models were developed to address the fact that calibration yields a constant implied volatility under the Black-Scholes model, whereas, in practice, the implied volatilities vary when different strikes are used. The implied volatility of an option can therefore be thought of as a function of the strike rate. If one had to plot a function of the implied volatility over different strikes, the outcome often takes the shape of a "volatility smile", where the minimum or maximum implied volatility is at the at-the-money strike. Another common shape is the volatility "skew", in which the implied volatility is an increasing or decreasing function of the strike. It is clear that a constant volatility model, such as the Black-Scholes model, would be unable to exhibit such qualities.

Some share price models have attempted to improve on the Black-Scholes model. A useful adjustment is the use of a volatility function as opposed to the constant volatility in the Black-Scholes model. An important share price model to consider is the local volatility model of Dupire (1994), which allows the volatility to be a function of the share price as well as time. A benefit of using local volatility models is that the implied volatility can be calibrated exactly to market conditions. Note that the Black-Scholes model can be described as a simplified local volatility model, with a constant volatility function. However, Gatheral (2011) states that in comparison to the Black-Scholes model, the local volatility model is more easily calibrated to fit existing market conditions and that the local volatility model is better able to match the implied volatility of the option over a range of strikes. Schied and Stadje (2007) state that a potential downside to using local volatility models is that the model is less accurate at delta-hedging path-dependent options. Examples of path-dependent options include American options, look-back options and barrier options. In order to obtain a share price model that is able to fit the implied volatility of the option at all times (and therefore be able to price path-dependent options more accurately), the volatility process must also be stochastic. Popular stochastic volatility models, such as the Heston (1993) model, allow the volatility to vary randomly over time. A problem with stochastic volatility models is that calibration often provides inaccurate implied volatilities. This is solved by using a model that adds a time-dependent local volatility component to the stochastic volatility model. The resulting model is known as a stochastic local volatility model which has all the benefits of a stochastic volatility model, with the added feature of calibration exactly fitting the implied volatility. The stochastic local volatility model is the focus of this dissertation.

Once share price dynamics have been chosen (and hence the share price at each time is determined), the goal is to use those dynamics to compute the option price

at a particular time. Since European options can only be exercised at maturity, only the share price at maturity is considered. As discussed above, if the stock price is modelled as a geometric Brownian motion process, then a closed-form price for the option exists, thanks to Black and Scholes (1973). However, if no closed-form solution is available, such as the case for the stochastic local volatility model, the standard procedure would be to use numerical methods to price the option.

There are a number of different methods that can be used to numerically price an option. The pricing methods explored in this dissertation include finite difference approximations, characteristic function prices and Markov-chain Monte-Carlo estimates. Each method can be remarkably accurate if it is made to be robust enough — for example, if enough Monte-Carlo samples are used, if the finite-difference grid has a large number of points or if the characteristic function pricing method has a large number of quadrature steps. There are cases, however, where some methods may be more appropriate than others. Andreasen and Huge (2011) point out that when pricing a call option under the Heston model, calibration is typically done using a Fourier transform. However, pricing other options requires backward finite difference pricing using the Craig and Sneyd (1988) method or the Andersen (2007) method based on, for example, a Monte-Carlo simulation.

This means that the different tasks of calibration and pricing are handled by different discrete numerical methods. If the model is calibrated using one discrete technique and priced using another, the results may be biased and therefore, inconsistent. This introduces the risk of model inconsistency, which is an increasing concern for institutions involved in the pricing and hedging of options. When assuming an underlying model, pricing method consistency refers to using a different pricing method but still obtaining similar values. Essentially, two methods are consistent if they are bias-free in the sense that they both converge to the same value in the limits. When this condition is not met, the pricing methods are said to be inconsistent. This type of inconsistency is the fundamental problem that this dissertation addresses.

Some option pricing models may be particularly useful if they allow for accurate hedging strategies. A hedging strategy is a decision to take an offsetting position in the underlying asset to reduce investment risk. In order to accurately hedge an option, the correct hedge value (amount invested in the underlying asset) must be chosen. Clearly, there are many causes for a model being unable to provide accurate hedge values, such as if the model is a poor fit for the observed market, if the data used for calibration is not useful, or if inaccurate parameter values are used. These causes are, however, not the main focus of this dissertation. This dissertation addresses the problem of inaccurate hedging values due to model inconsistency

between pricing methods and calibration. Green and Figlewski (1999) explain that the inability to hedge market positions accurately provides a significant downside risk for financial institutions. For this reason, the bias introduced by inconsistent pricing methods is important in the field of mathematical finance.

One way of making these methods consistent is by letting the number of Monte-Carlo samples, the size of the finite difference grid and the number of quadrature steps for the characteristic pricing function tend to infinity. The problem with this approach is that it is not computationally tractable. Therefore, there is a need for a pricing method that allows for discrete consistency between calibration, finite difference pricing as well as Monte-Carlo pricing. One possible method for producing discrete results that are consistent with each other is by using the random grids method by Andreasen and Huge (2011).

Andreasen and Huge (2011) claim that their random grids approach achieves full discrete consistency between calibration, finite difference solution and Markov-chain Monte-Carlo simulation. This means that under a specific share price dynamic, if the model is calibrated using one numerical method and priced using another, the results will still be consistent. More specifically, if the stochastic local volatility model is used and the most appropriate calibration method is a finite difference method, but the most appropriate pricing method is a Monte-Carlo method, then the random grids approach ensures that the results obtained are fully consistent with each other. The ingenuity of the random grids method lies in how the different methods are defined. Instead of using a traditional approach, Andreasen and Huge (2011) define each pricing method slightly differently to ensure that model consistency is achieved.

The random grids approach is derived from the relationship between the forward finite difference (FFD) and the backward finite difference (BFD) equations. The BFD method is the most common finite difference method used in option pricing. However, Andreasen and Huge (2011) have described a method of linking the BFD method to the FFD method by using the adjoint operators. The adjoint operators ensure that the FFD method is fully consistent with the BFD method. Therefore, the pricing and calibration results from the BFD method are fully consistent with the results from the FFD method when the adjoint operators are used. This is useful because a Markov-chain Monte-Carlo method is developed from the FFD method. Since the Markov-chain Monte-Carlo approach is derived from the FFD method, Markov-chain Monte-Carlo results are consistent with results obtained from the FFD method. Furthermore, since the FFD method is consistent with the BFD method, the Markov-chain Monte-Carlo method is consistent with the BFD approach too. Therefore, by the way that this Markov-chain Monte-Carlo method

is constructed, consistent results are guaranteed, even if the model is calibrated using a finite difference method and priced using the Markov-chain Monte-Carlo method.

The term ”random grids” is derived from the fact that finite difference methods rely on a grid of discretised state values and the fact that the FFD method prices options using a probability density. Since the Markov-chain Monte-Carlo method is derived from the FFD method, it uses a grid of discretised state values as well as a probability density to generate random numbers. Therefore, the term ”random grids” describes this Markov-chain Monte-Carlo method. This dissertation reviews and implements the random grids approach in order to achieve results similar to those obtained by Andreasen and Huge (2011).

The dissertation begins by developing the necessary background for the random grids approach. This involves expressing the stochastic local volatility model that is used in this review as well as the call option that is evaluated. Furthermore, a description of the various pricing methods is provided. This includes a derivation and description of the BFD method, the FFD method as well as the Markov-chain Monte-Carlo approach that is based on the grids. Accompanying these derivations is a brief description of the characteristic function pricing method that is used to verify random grids results. In addition, a discussion on the assumptions made by Andreasen and Huge (2011) is provided.

The following section provides a description of how to implement each pricing method. Emphasis is placed on developing the grids used in the forward and backward finite difference methods, since these are the same grids used in the Markov-chain Monte-Carlo method. For the FFD approach, emphasis is placed on using the adjoint operators to produce the probability distributions, since these are also used by the Markov-chain Monte-Carlo approach. The penultimate part of this section provides insight into implementing the random grids method by using parameters relevant to a different scenario. The reason that this is important, is that although Andreasen and Huge (2011) state what parameters and data-sets they used to obtain their results, it is useful to adapt the method for a different market, which will require a different set of parameters. This review provides insight on how to decide on the parameter values so that the random grids method can be universally applied. Finally, a discussion on using more efficient algorithms to implement the random grids approach is provided.

Once the implementation is described, the results that have been obtained are presented. Along with the stochastic local volatility model, results are obtained for a Heston model. This shows that the random grids method works for share price dynamics other than the stochastic local volatility model. The results highlight that

under a given share price model, consistent prices are obtained for each pricing method and, therefore, that the use of random grids addresses the bias introduced by model inconsistency. In addition, the implied volatilities are stated. The implied volatilities show that the stochastic local volatility and Heston models are able to express implied volatility curves as a function of the strike. Subsequent to the display of results, an analysis is provided, expressing all findings and interpretations of the results that have been obtained.

# 2 Framework for Random Grids

This section explores the framework that is necessary to define the random grids approach. This involves specifying the stochastic processes used to propagate the share price from the present time ($t_0$) to the maturity date ($T$). In addition, the basic concepts for the finite difference and Monte-Carlo pricing methods are described.

## 2.1 Share Price Process

Recall from the introduction that there are various ways to model the share price. The random grids method by Andreasen and Huge (2011) assumes that the forward share price process[1] $S$ follows a stochastic local volatility model, specified as

$$dS(t) = \sqrt{z(t)}\sigma(t, S(t))dW(t), \tag{2.1}$$

where $\sigma(t, S(t))$ is the volatility function and where $z$ is a stochastic mean reverting process satisfying

$$dz(t) = \theta(1 - z(t))dt + \epsilon z(t)^{\gamma} dZ(t), \tag{2.2}$$

where $\epsilon$, $\theta$ and $\gamma$ are constants and where $W(t)$ and $Z(t)$ are independent Brownian motion processes. The constant $\theta$ is the rate that the process $z(t)$ reverts to its long-run mean of 1, and $\epsilon$ and $\gamma$ are volatility scaling parameters. This model allows both the share price process as well as the volatility process to be stochastic. Observe that the share price process (2.1) is assumed to be driftless and that the forward measure is being used. Therefore, $S$ refers to the forward share price process. Under these driftless conditions, bond prices are also assumed to be uncorrelated with the stock price and volatility. In addition, Andreasen and Huge (2011) assume that the asset process and volatility process have a zero correlation, resulting in

$$dW(t)dZ(t) = 0. \tag{2.3}$$

[1] Note that other assets may also be modelled by this forward process (2.1), such as forward rates.

To help with some of the explanations, a simplified set of notation is used. Suppose that the share price is denoted as $S = x$, and that the volatility is denoted as $z = y$. Since the random grids method is a discrete pricing method that relies on finite difference grids, discrete values for $x$, $y$ and $t$ are used. The discrete values that $x$ may take on in the finite difference grid are contained in the vector $x_{\text{range}} = \{x_0, x_1, ..., x_{m-1}\}$, the discrete values that $y$ may take on are contained in the vector $y_{\text{range}} = \{y_0, y_1, ..., y_{n-1}\}$ and the discrete values that $t$ may take on are contained in the vector $t_{\text{range}} = \{t_0, t_1, ..., t_{\tau-1} = T\}$, where $n, m, \tau \in \mathbb{N}$. An arbitrary element from each vector is expressed as $x_i$, $y_j$ and $t_h$. In order to express the value that the share price and volatility take on at a discrete point in time $t_h$, the notation $x(t_h)$ and $y(t_h)$ is used.

This completes the specification of the share price process. Since the share is the underlying asset for the option contract, the option instrument is specified hereafter.

## 2.2 Option Instrument

An option is a contract that provides the holder with the right but not the obligation to exercise the option at a specified strike price at a specified time in future. In this review, a standard European call option is considered. The call option payoff depends on the share price at maturity, which is determined by the stochastic local volatility model described in (2.1). Therefore if $V(t, x, y)$ represents the value of the option with a share price of $x$ and volatility of $y$ at time $t$, then at maturity (time $T$), the call option payoff is expressed as

$$V(T, x, y) = \max(x - K, 0), \tag{2.4}$$

where $K$ is the forward strike rate. Since the option instrument is a European call, $V$ only depends on the value of $x$ at time $T$. However, other options, such as a volatility swap, may depend on the values of $y$ at a particular point in time.

It is important to note that various numerical methods can be used to price a call option at time $t_0$. Each method attempts to price the option by forming realisations of the share price at maturity. The subsections hereafter describe the various pricing methods that are used in this dissertation.

## 2.3 Framework for Finite Difference Methods

The finite difference method for the stochastic local volatility model is a discrete numerical method for solving for $V$ across all possible combinations of $x_i$, $y_j$ and $t_h$ on the grid.

Since this dissertation deals with the share price process being a stochastic local volatility model, the finite difference method must accommodate stochastic local volatility dynamics. Observe that the stochastic local volatility model has two stochastic processes — the share price process as well as the volatility process. Considering (2.1) and (2.2), it is clear that the share price process depends on the volatility process. Thus, when the volatility process varies, a different share price is attributed to that level of volatility at each time. Therefore, a three-dimensional finite difference grid is formulated to accommodate these dynamics. At time $t_h$, the resulting grid will have a point that represents the value for the option price $V(t_h, x_i, y_j)$ for a given combination of share price and volatility $(x_i, y_j)$.

The goal of the three-dimensional finite difference approach, using a stochastic local volatility model, is to consider a time $(t_h)$ where information is known for all $(x_i, y_j)$ combinations and use it to obtain the option values for all $(x_i, y_j)$ combinations at a different time $(t_{\tilde{h}})$. If information known for all $(x_i, y_j)$ combinations at an earlier time is used to obtain the option price at a later time, then an FFD method is used. Conversely, if information known for all $(x_i, y_j)$ combinations at a later time is used to obtain the option price at an earlier time, then a BFD method is used. The FFD method involves solving a set of equations forward in time, while the BFD method involves solving a different set of equations backwards in time. The set of equations that are solved are known as the pricing partial differential equations (pricing PDEs). More specifically, the equation involved when starting at time $T$ and moving backwards in time to $t_0$, is known as the backward pricing PDE. Alternatively, if an initial condition is given at time $t_0$, the forward pricing PDE is solved at each time step to obtain a price at maturity.

For the BFD method, the information that is known at maturity (time $T$) for all $(x_i, y_j)$ combinations is the option price. The reason that this is known is that the share price process is defined by the stochastic local volatility model (2.1), which is able to produce a share price path at all times. Therefore, using the share price at maturity, the option payoff at maturity can be computed for each $(x_i, y_j)$ combination using (2.4).

For the FFD method, it is the probability density that is known at time $t_0$ for all $(x_i, y_j)$ combinations. Therefore, the probability density is evolved forward in time using the forward pricing PDE to obtain a probability density for all $(x_i, y_j)$ combinations at maturity. The probability density at maturity is used to obtain an expected option payoff at maturity. Thereafter, risk-neutral pricing is used to obtain the time $t_0$ option price. The details for the forward and backward pricing PDEs are described hereafter.

### 2.3.1 Backward Pricing PDE for Option Price

Consider the call option payoff at time $T$, described by (2.4). The discounted expected payoff ($V(t, x, y)$) is the solution to the Feynman Kac theorem applied to a PDE of a certain form, with a terminal condition described by (2.4). This PDE is known as the backward pricing PDE. Under the stochastic local volatility model, the backward pricing PDE is defined as

$$0 = \frac{\partial V}{\partial t} + D_x V + D_y V, \tag{2.5}$$

$$\text{where } D_x = \frac{1}{2}\sigma^2 y \frac{\partial^2}{\partial_x^2}$$
$$\text{and } D_y = \theta(1-y)\frac{\partial}{\partial y} + \frac{1}{2}\epsilon^2 y^{2\gamma}\frac{\partial^2}{\partial y^2}.$$

The PDE above is defined using continuous variables $x$, $y$ and $t$. Therefore the backward pricing PDE expresses some relationship about the instantaneous change of the option price over an infinitely small change in the share price and volatility. However, the finite difference method is a discrete pricing method; therefore, the continuous nature of the backward pricing PDE is incompatible. As a result, some discrete approximations have to be made to this PDE. The discretised equation expresses the change in the value of the option over an entire step in time, share price and volatility ($t_h$, $x_i$, $y_j$), as opposed to being expressed by their continuous counterparts. This discrete approximation leads to the BFD equations, which are provided later in this chapter. It is, in fact, these BFD equations that are solved to progress the option price at each discrete time period. However, before discussing these, it is important to understand how the forward pricing PDE is formed.

### 2.3.2 Forward Pricing PDE for Density

The FFD process differs from the BFD process in two important ways. The first difference is the direction in which time progresses, and the second difference is the quantity that is being evolved. The BFD method evolves the option price backwards in time, while the FFD method evolves the joint probability density of $x$ and $y$ forward in time. The joint probability density of $x$ and $y$ at time $t$ is defined as

$$p(t, x, y) = \mathbb{Q}[S(t) \in dx, z(t) \in dy],$$

where $\mathbb{Q}$ is the risk-neutral probability measure.

Suppose that the current market share price ($x(t_0)$) and volatility ($y(t_0)$) are known. In that case, the share price at time $t_0$ takes on the value of $x(t_0)$, with

a probability of 1 and the volatility process at time $t_0$ takes on the value of $y(t_0)$, with a probability of 1. Furthermore, there is a zero probability that the share price and volatility take on any other value at time $t_0$. Therefore, the formal expression for the initial condition of the forward pricing PDE at time $t_0$ is

$$p(t_0, x, y) = \mathbb{I}_{\{x=x(t_0)\}}\mathbb{I}_{\{y=y(t_0)\}}.$$

The forward pricing PDE evolves this joint probability density forward in time until a joint probability density at maturity is obtained. The forward pricing PDE (also known as the Kolmogorov Forward Equation) that is therefore needed to be solved for $p$, forward in time, is defined as

$$0 = -\frac{\partial p}{\partial t} + D_x^* p + D_y^* p, \tag{2.6}$$

where the adjoint operators are defined as

$$D_x^* p = \frac{1}{2}\frac{\partial^2}{\partial_x^2}[\sigma^2 y p] \text{ and}$$
$$D_y^* p = -\frac{\partial}{\partial y}[\theta(1-y)p] + \frac{1}{2}\frac{\partial^2}{\partial y^2}[\epsilon^2 y^{2\gamma} p].$$

In a similar case as the backward pricing PDE, the forward pricing PDE expresses an equation in terms of the instantaneous change of the probability density over infinitely small changes in the share price and volatility. However, instead of discretising the forward PDE, Andreasen and Huge (2011) suggest an alternative formulation of the FFD equations. The FFD equations stated by Andreasen and Huge (2011) are derived from the BFD equations in conjunction with the adjoint operators. This formulation ensures consistency between the FFD and BFD equations. It is, in fact, these FFD equations that are solved discretely to obtain the probability density forward in time. The BFD and FFD equations are provided below.

### 2.3.3 Backwards Finite Difference Equations

The BFD method uses the option price at maturity (time $T$) to solve for the option price backwards in time until an option price at time $t_0$ is obtained. Since the option depends on the underlying share price, which depends on the volatility, the BFD equation must be solved at each discrete time for each discrete combination of $x_i$ and $y_j$. As explained earlier, the backward pricing PDE needs to be discretised to obtain the BFD equation. The expanded form of the backward PDE in (2.5) is expressed as

$$0 = \frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 y\frac{\partial^2 V}{\partial x^2} + \theta(1-y)\frac{\partial V}{\partial y} + \frac{1}{2}\epsilon^2 y^{2\gamma}\frac{\partial^2 V}{\partial y^2}. \tag{2.7}$$

The discretisation involves using discrete grid values $x_i \in x_{\text{range}}$, $y_j \in y_{\text{range}}$ and $t_h \in t_{\text{range}}$ instead of the continuous counterparts. Discrete approximations for the partial derivatives in (2.7) are obtained from a Taylor series expansion, which provides the forward and central difference approximations. The partial derivative with respect to time is approximated by the forward difference approximation, and the partial derivatives with respect to the spatial variables ($x$ and $y$) are approximated using central difference approximations. Therefore, the discrete approximations made to (2.7) are

$$\begin{aligned}
\frac{\partial V(t_h,x_i,y_j)}{\partial t} &\approx \frac{V(t_{h+1},x_i,y_j)-V(t_h,x_i,y_j)}{\Delta t},\\
\frac{\partial^2 V(t_h,x_i,y_j)}{\partial x^2} &\approx \frac{V(t_h,x_{i-1},y_j)-2V(t_h,x_i,y_j)+V(t_h,x_{i+1},y_j)}{\Delta x^2} = \delta_{xx}V(t_h,x_i,y_j),\\
\frac{\partial V(t_h,x_i,y_j)}{\partial y} &\approx \mathbb{I}_{\{y_j<1\}}\frac{V(t_h,x_i,y_{j+1})-V(t_h,x_i,y_j)}{\Delta y}\\
&\quad + \mathbb{I}_{\{y_j>1\}}\frac{V(t_h,x_i,y_j)-V(t_h,x_i,y_{j-1})}{\Delta y} = \delta_y V(t_h,x_i,y_j) \text{ and}\\
\frac{\partial^2 V(t_h,x_i,y_j)}{\partial y^2} &\approx \frac{V(t_h,x_i,y_{j-1})-2V(t_h,x_i,y_j)+V(t,x_i,y_{j+1})}{\Delta y^2} = \delta_{yy}V(t_h,x_i,y_j),
\end{aligned}$$

where $\Delta t = t_{h+1}-t_h$, $\Delta x = x_{i+1}-x_i$ and $\Delta y = y_{j+1}-y_j$. The resulting discrete expression for (2.7) is

$$\begin{aligned}
0 = \frac{V(t_{h+1},x_i,y_j)-V(t_h,x_i,y_j)}{\Delta t} &+ \frac{1}{2}\sigma^2 y_j \delta_{xx}V(t_h,x_i,y_j)\\
&+ \theta(1-y_j)\delta_y V(t_h,x_i,y_j) + \frac{1}{2}y_j^{2\gamma}\epsilon^2\delta_{yy}V(t_h,x_i,y_j).
\end{aligned} \tag{2.8}$$

At time $t_{h+1}$, the BFD equation uses $V(t_{h+1},x_i,y_j)$ to solve for the option price ($V(t_h,x_i,y_j)$) at the previous time period ($t_h$) for all combinations of $x_i$ and $y_j$. To simplify the notation, the terms corresponding to the discrete approximations with respect to $x$ and $y$ are expressed as

$$\begin{aligned}
\bar{D}_{x,y_j} &= \frac{1}{2}y_j\sigma^2\delta_{xx} \text{ and}\\
\bar{D}_{y,x_i} &= \theta(1-y_j)\delta_y + \frac{1}{2}y_j^{2\gamma}\epsilon^2\delta_{yy}.
\end{aligned}$$

Observe that the system in (2.8) has two spatial variables ($x$ and $y$) associated with the time variable. Although it is sufficiently stable to solve such a system using a fully implicit scheme, the computation can be complex. Suppose that there are $m$ variables associated with $x$ and $n$ variables associated with $y$. In that case, there are $m \times n$ unknowns at each time point. Solving these $m \times n$ unknowns may be possible by using $m \times n$ simultaneous equations, which involves forming an $m \times n$ by $m \times n$ matrix. However, this matrix would not necessarily be tridiagonal,

and therefore, a standard matrix inversion algorithm such as the Thomas algorithm cannot be used to solve the equations. An alternative way to solve a system with two spatial variables ($x$ and $y$) at a particular point in time is to use a method known as an alternating direction implicit (ADI) scheme.

An ADI scheme involves splitting the portion of the equation with respect to each variable into two separate components. Therefore, the process would entail solving backwards for $V$ at the half time step with respect to $x$ and subsequently solving for $V$ at the remaining half time step with respect to $y$. This method is computationally more efficient than a fully implicit method. Therefore, instead of solving (2.8), the BFD equations that correspond to the backward pricing PDE (2.5), are described by the ADI scheme

$$\begin{aligned}(1-\Delta t\bar{D}_{x,y_j})V(t_{h+1/2},\cdot,y_j) &= V(t_{h+1},\cdot,y_j) \quad \forall y_j,\\ (1-\Delta t\bar{D}_{y,x_i})V(t_h,x_i,\cdot)^\top &= V(t_{h+1/2},x_i,\cdot)^\top \quad \forall x_i,\end{aligned}$$

where $V(t_{h+1},\cdot,y_j)$ is an $m\times 1$ column vector for the option price at time $t_{h+1}$ across all $x_i \in x_{\text{range}}$, with volatility $y_j \in y_{\text{range}}$ and where $V(t_h,x_i,\cdot)^\top$ is the transpose of the $1\times n$ row vector (i.e. an $n\times 1$ column vector) for the option price at time $t_h$ across all $y_j \in y_{\text{range}}$, with the share price at $x_i \in x_{\text{range}}$.

In the ADI scheme above, $(1-\Delta t\bar{D}_{x,y_j})$ and $(1-\Delta t\bar{D}_{y,x_i})$ are tridiagonal matrices. Furthermore, the boundary conditions are set such that $\delta_{xx}V(t_h,x_i,y_j)=0$ and $\delta_{yy}V(t_h,x_i,y_j)=0$. If $A_{x,y_j}\equiv(1-\Delta t\bar{D}_{x,y_j})$ and $A_{y,x_i}\equiv(1-\Delta t\bar{D}_{y,x_i})$, then the ADI scheme for the BFD equation is expressed as

$$\begin{aligned}A_{x,y_j}V(t_{h+1/2},\cdot,y_j) &= V(t_{h+1},\cdot,y_j) \quad \forall y_j,\\ A_{y,x_i}V(t_h,x_i,\cdot)^\top &= V(t_{h+1/2},x_i,\cdot)^\top \quad \forall x_i.\end{aligned}$$

The tridiagonal matrix $A_{x,y_j}$ is defined as

$$A_{x,y_j}=\begin{bmatrix} 1 & 0 & & & & & \\ -a_1 & 1+2a_1 & -a_1 & & & & \\ & -a_2 & 1+2a_2 & -a_2 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -a_{m-2} & 1+2a_{m-2} & -a_{m-2} & \\ & & & & 0 & 1 & \end{bmatrix}, \tag{2.9}$$

with $a_i=\frac{1}{2}\frac{\Delta t}{\Delta x^2}y_j\sigma(t,x_i)^2$ and $x_i=x_0+i\Delta x$. Furthermore, the tridiagonal matrix

$A_{y,x_i}$ is defined as

$$A_{y,x_i} = \begin{bmatrix} 1+\bar{c}_0 & -\bar{c}_0 & & & & \\ -\bar{a}_1 & 1+\bar{b}_1 & -\bar{c}_1 & & & \\ & -\bar{a}_2 & 1+\bar{b}_2 & -\bar{c}_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -\bar{a}_{n-2} & 1+\bar{b}_{n-2} & -\bar{c}_{n-2} \\ & & & & -\bar{a}_{n-1} & 1+\bar{a}_{n-1} \end{bmatrix}, \tag{2.10}$$

where

$$\begin{aligned} \bar{a}_j &= \frac{1}{2}\frac{\Delta t}{\Delta y^2}\epsilon^2 y_j^{2\gamma} - \frac{\Delta t}{\Delta y}\theta(1-y_j)\mathbb{I}_{\{y_j>1\}}, \\ \bar{c}_j &= \frac{1}{2}\frac{\Delta t}{\Delta y^2}\epsilon^2 y_j^{2\gamma} + \frac{\Delta t}{\Delta y}\theta(1-y_j)\mathbb{I}_{\{y_j<1\}}, \\ \bar{b}_j &= \bar{a}_j + \bar{c}_j, \\ \bar{c}_0 &= \frac{\Delta t}{\Delta y}\theta(1-y_0), \\ \bar{a}_{n-1} &= -\frac{\Delta t}{\Delta y}\theta(1-y_{n-1}) \text{ and} \\ y_j &= y_0 + j\Delta y. \end{aligned}$$

Since each $A_{y,x_i}$ matrix is independent of $x_i$, all $A_{y,x_i}$ matrices are identical and are denoted as $A_y$. In that case, instead of solving the second step of the ADI scheme for $V(t_h, x_i, \cdot)$ for all $x_i$, the entire $V(t_h, \cdot, \cdot)$ matrix can be solved for using $A_y$. Therefore, the ADI scheme for the BFD equation is expressed as

$$A_{x,y_j} V(t_{h+1/2}, \cdot, y_j) = V(t_{h+1}, \cdot, y_j) \quad \forall y_j, \tag{2.11}$$

$$A_y V(t_h, \cdot, \cdot)^\top = V(t_{h+1/2}, \cdot, \cdot)^\top. \tag{2.12}$$

Given that the BFD method works backwards in time, the initial value $V(T, \cdot, \cdot)$ is given at the maturity of the the call option. Thereafter, the BFD equations (2.11) and (2.12) are solved backward in time until the option value $V(t_0, \cdot, \cdot)$ is obtained for each $(x_i, y_j)$ combination. The final step is to extract the value from $V(t_0, \cdot, \cdot)$ that corresponds to the share price $(x(t_0))$ and volatility $(y(t_0))$ at time $t_0$.

### 2.3.4 Forward Finite Difference Equations

Instead of deriving the FFD equations the traditional way (as was done for the BFD equations), Andreasen and Huge (2011) use the BFD equations along with the adjoint operators to form an elegant formulation of the FFD method. The benefit

of expressing the FFD equation this way is that discrete consistency with the BFD equation is guaranteed.

Andreasen and Huge (2011) choose the forward ADI scheme such that

$$A_y^\top p(t_{h+1/2},\cdot,\cdot)^\top = p(t_h,\cdot,\cdot)^\top, \tag{2.13}$$

$$A_{x,y_j}^\top p(t_{h+1},\cdot,y_j) = p(t_{h+1/2},\cdot,y_j) \quad \forall y_j \tag{2.14}$$

$$\text{and} \quad p(t_0,x_i,y_j) = \mathbb{I}_{\{x_i=x(t_0)\}}\mathbb{I}_{\{y_j=y(t_0)\}} \quad \forall x_i,y_j,$$

where $A_y^\top$ and $A_{x,y_j}^\top$ are the adjoint operators, which are just the transpose of the relevant $A_y$ and $A_{x,y_j}$ matrices.

Note that $A_y$ is an $n \times n$ matrix, $A_{x,y_j}$ is an $m \times m$ matrix and $p(t_h,\cdot,\cdot)$ and $V(t_h,\cdot,\cdot)$ are $m \times n$ matrices. If $A$ and $B$ are two $m \times n$ matrices, then an important result is that $\text{trace}(AB^\top) = \sum_i \sum_j A_{i,j}B_{i,j}$. In addition, $\text{trace}(AB^\top) = \text{trace}(A^\top B)$. Using these properties of the trace operator, as well as the definitions for the BFD ((2.11) and (2.12)) and FFD ((2.13) and (2.14)) equations, it is clear that[2]

$$\begin{aligned}
\sum_i \sum_j p(t_h,x_i,y_j)V(t_h,x_i,y_j) &= \text{trace}(p(t_h,\cdot,\cdot)V(t_h,\cdot,\cdot)^\top) \\
&= \text{trace}(p(t_h,\cdot,\cdot)A_y^{-1}V(t_{h+1/2,\cdot,\cdot})^\top) && \text{from (2.12)} \\
&= \text{trace}(p(t_{h+1/2},\cdot,\cdot)A_yA_y^{-1}V(t_{h+1/2,\cdot,\cdot})^\top) && \text{from (2.13)} \\
&= \text{trace}(p(t_{h+1/2},\cdot,\cdot)V(t_{h+1/2,\cdot,\cdot})^\top) \\
&= \text{trace}(p(t_{h+1/2},\cdot,\cdot)^\top V(t_{h+1/2,\cdot,\cdot})) \\
&= \sum_j p(t_{h+1/2},\cdot,y_j)^\top V(t_{h+1/2},\cdot,y_j) \\
&= \sum_j p(t_{h+1/2},\cdot,y_j)^\top A_{x,y_j}^{-1}V(t_{h+1},\cdot,y_j) && \text{from (2.11)} \\
&= \sum_j p(t_{h+1},\cdot,y_j)^\top A_{x,y_j}A_{x,y_j}^{-1}V(t_{h+1},\cdot,y_j) && \text{from (2.14)} \\
&= \sum_j p(t_{h+1},\cdot,y_j)^\top V(t_{h+1},\cdot,y_j) \\
&= \text{trace}(p(t_{h+1},\cdot,\cdot)^\top V(t_{h+1},\cdot,\cdot)) \\
&= \sum_i \sum_j p(t_{h+1},x_i,y_j)V(t_{h+1,x_i,y_j}).
\end{aligned}$$

[2] Special acknowledgement to my supervisor Thomas McWalter for providing this proof.

Thus, by induction, for European options, it is clear that

$$\begin{aligned} V(t_0) &= \sum_i \sum_j p(t_0, x_i, y_j) V(t_0, x_i, y_j) \\ &= \sum_i \sum_j p(T, x_i, y_j) V(T, x_i, y_j), \end{aligned}$$

which is consistent with the definition of the risk-neutral expectation, assuming that all $p$ values are non-negative and sum to one, i.e., $\sum_i \sum_j p(t_h, x_i, y_j) = 1$ for all $t_h$.

Since the initial condition is described as $p(t_0, x_i, y_j) = \mathbb{I}_{\{x_i = x(t_0)\}} \mathbb{I}_{\{y_j = y(t_0)\}}$, it is clear that $p(t_0, \cdot, \cdot)$ has unit mass at time $t_0$. Due to the way that the $A_{x,y_j}$ and $A_y$ matrices are constructed, it is clear that $A_{x,y_j}.1 = 1$ and $A_y.1 = 1$, where 1 is a vector of ones. Therefore, $A_{x,y_j}^{-1}.1 = 1$ and $A_y^{-1}.1 = 1$ as well. Since the FFD method progresses $p$ at each time step by pre-multiplying by $(A_{x,y_j}^{-1})^\top$ and $(A_y^{-1})^\top$, the $p(t_h, \cdot, \cdot)$ matrices always sum to one. Therefore, the $A_{x,y_j}$ and $A_y$ matrices are constructed such that $p(t_h, \cdot, \cdot)$ preserves its mass of one at all times $t_h$.

The fact that the $a_i$ components in $A_{x,y_j}$ are non-negative, along with results from Nabben (1999) show that $A_{x,y_j}^{-1} \geq 0$. Similarly, since the $\bar{a_j}$, $\bar{b_j}$ and $\bar{c_j}$ components of $A_y$ are non-negative, $A_y^{-1} \geq 0$ as well. Therefore, since $p(t_0, \cdot, \cdot)$ starts off being non-negative, and is progressed in time by pre-multiplying by the relevant $(A_{x,y_j}^{-1})^\top$ and $(A_y^{-1})^\top$ matrices, the $p(t_h, \cdot, \cdot)$ matrices are non-negative at all times. Therefore, the $p(t_h, \cdot, \cdot)$ matrices satisfy the conditions to be probability density matrices.

Having proved the remaining conditions, it is clear that the ADI scheme, given by (2.13) and (2.14), evolves the probability density matrix ($p$) in a way that is consistent with the backward ADI scheme.

Instead of computing new $A_{x,y_j}$ and $A_y$ matrices along with their boundary conditions, the use of the adjoint operators allows for an elegant solution for the FFD equations. Andreasen and Huge (2011) point out that direct discretisation of (2.6) results in a different formulation for the FFD equations; however, the advantage of the method developed by Andreasen and Huge (2011) is that it is fully consistent with the BFD method.

Once the probability density matrix $p(T, \cdot, \cdot)$ at maturity is obtained, the risk-neutral valuation approach is used to price the instrument. The risk-neutral pricing method computes the expected option payoff at maturity under the risk-neutral measure (formed by summing across the product of all possible option payoffs and

probabilities). The option price at time $t_0$ is therefore calculated as

$$\begin{aligned} V(t_0) &= \mathbb{E}[V(T,\cdot,\cdot)] \\ &= \sum_{x_i}\sum_{y_j}\left[V(T,x_i,y_j)p(T,x_i,y_j)\right]. \end{aligned}$$

## 2.4 Framework for Monte-Carlo Methods

Monte-Carlo simulations are one of the most popular numerical methods used for option pricing. Recall that a risk-neutral valuation requires the discounted expected payoff of the option at maturity. The Monte-Carlo method is one way of estimating the integral required to compute the expected value of the option payoff. Glasserman (2013) explains that although there are various methods for estimating this integral, the Monte-Carlo method is especially useful for higher-dimensional problems. The convergence of the Monte-Carlo method is of order $\frac{1}{\sqrt{\text{sample size}}}$, regardless of the number of dimensions in the integration problem. It is for this reason that the Monte-Carlo method is such a popular option pricing method. It is useful to first discuss a general Monte-Carlo approach to pricing an option, and thereafter, the Markov-chain Monte-Carlo method based on random grids will be discussed.

A general Monte-Carlo simulation to price options aims to generate share price paths according to a desired distribution. Since the share price process is stochastic, the Monte-Carlo method uses random number generation to simulate where the share price may go at a future point in time. Once the random numbers are used to generate a share price at maturity, the call option payoff at maturity is computed using (2.4). This is enough to compute the option payoff for a single sample; however, to make the estimate more accurate, many samples need to be used. Note that each sample represents one possible evolution of the share price. The goal is to have as many possible share price paths to produce a large number of option payoffs. The option is priced using risk-neutral valuation, which requires the expected option payoff. The expected payoff of the option is estimated as the sample mean and by the law of large numbers, as the sample size increases, the sample mean tends to the expected value.

Just as the goal in the general Monte-Carlo method was to generate the share price at each time until maturity, the goal of the Markov-chain Monte-Carlo method based on random grids is to generate values for the share price ($x_i$) and the volatility ($y_j$) at each time point ($t_h$) until maturity. However, the Markov-chain Monte-Carlo method based on random grids generates values consistent with the BFD and FFD approaches. Therefore, to make the Markov-chain Monte-Carlo method con-

sistent with the finite-difference methods, it uses the same $A_{x,y_j}$ and $A_y$ matrices obtained from the finite difference method. Since the Markov-chain Monte-Carlo method uses these $A_{x,y_j}$ and $A_y$ matrices to simulate a probability density forwards in time, the random grids methods can be compared to the FFD pricing method.

Since the probabilities are derived from operators in a finite difference grid, each probability value corresponds to a point on the finite difference grid. Each point on the finite difference grid can be thought of as a state $(x_i, y_j)$, and the transition probabilities are the probabilities of moving from one state to the next. This is the reason for the so-called name "random grid". It is worth mentioning that the use of the $A_{x,y_j}$ and $A_y$ matrices ensures that the distribution of the share prices and volatilities simulated, follows the stochastic local volatility model, defined in (2.1) and (2.2).

A formal explanation for this random grids approach is provided hereafter.

### 2.4.1 Conditional Transition Probability Matrices

The conditional transition probability matrices are obtained by taking the inverse of the $A_y$ and $A_{x,y_j}$ matrices, resulting in

$$(A^{-1}_{x,y_j})_{nm} = \mathbb{Q}[x(t_{h+1}) = x_m | x(t_h) = x(t_{h+1/2}) = x_n)] \text{ and} \tag{2.15}$$

$$(A^{-1}_{y})_{kl} = \mathbb{Q}[y(t_{h+1}) = y(t_{h+1/2}) = y_l | y(t_h) = y_k], \tag{2.16}$$

where $x_m$ and $x_n$ are values within the $x_{\text{range}}$ vector, where $y_k$ and $y_l$ are values within the $y_{\text{range}}$ vector and where $\mathbb{Q}$ is the risk-neutral probability measure.

Nabben (1999) highlights that the values in these matrices behave as conditional probabilities should, in that they sum to one and are always positive. Furthermore, these matrix values are constructed such that the generated share price and volatility values follow stochastic local volatility dynamics. However, to fully realise this, the cumulative conditional probabilities are used.

The cumulative distribution function for each $A^{-1}_{x,y_j}$ and $A^{-1}_{y}$ matrix ($Q_{x,y_j}$ and $Q_y$ respectively) is obtained by cumulatively summing across their rows. Each element of the required cumulative distribution function is computed as

$$(Q_{x,y_j})_{nm} = \sum_{r \leq m} (A^{-1}_{x,y_j})_{nr} = \mathbb{Q}[x(t_{h+1}) \leq x_m | x(t_h) = x_n] \text{ and} \tag{2.17}$$

$$(Q_{y})_{kl} = \sum_{r \leq l} (A^{-1}_{y})_{kr} = \mathbb{Q}[y(t_{h+1}) \leq y_l | y(t_h) = y_k]. \tag{2.18}$$

It is these cumulative probabilities that are used to generate the share price and volatility states.

### 2.4.2 Simulation Algorithm

The task is to use the set of cumulative probabilities to generate a value or state for $x$ and $y$ at each time step. At time $t_0$, the current share price and volatility values are set to today's share price $x(t_0)$ and today's volatility $y(t_0)$. Thereafter, the share price and volatility at each time step until maturity are generated, resulting in a path for the share price over time and a path for the volatility over time.

The following description explains the method for using the cumulative probability distribution matrix at one time point to simulate a value for $x$ and $y$ at the next time step.

Suppose that at time $t_{h-1}$, the $x$-value is $x_a$ and the $y$-value is $y_b$. Therefore, at time $t_h$, the relevant tridiagonal matrices are $A_{x,y_b}$ and $A_{y,x_a} = A_y$. Using these matrices, the corresponding probability distribution matrices, $A^{-1}_{x,y_b}$ and $A^{-1}_{y,x_a} = A^{-1}_y$, are accumulated to form their corresponding cumulative probability matrices, $Q_{x,y_b}$ and $Q_y$.

The simulation can therefore be described as follows:
0. Suppose $x(t_{h-1}) = x_a$ and $y(t_{h-1}) = y_b$.
1. Draw uniform random numbers $\tilde{U_x} \sim U(0,1)$ and $\tilde{U_y} \sim U(0,1)$.
2. Set $a_{\text{new}}$ = index of the first value of $Q_{x,y_b}(a,.)$ for which $Q_{x,y_b}(a,.) > \tilde{U_x}$.
3. Set $b_{\text{new}}$ = index of the first value of $Q_y(b,.)$ for which $Q_y(b,.) > \tilde{U_y}$.
4. $x(t_h) = x_{a_{\text{new}}}$ and $y(t_h) = y_{b_{\text{new}}}$.
5. Set $a = a_{\text{new}}$, set $b = b_{\text{new}}$.

Repeating the algorithm for each time step ensures that an $x$-value and $y$-value are generated at each time until maturity (time $T$). However, this generates only one path for $x$ and $y$. To obtain an accurate solution, many paths should be simulated so that many option payoffs are obtained at maturity. Following the general Monte-Carlo method, the sample mean is used to estimate the expected payoff of the option.

## 2.5 Characteristic Function Pricing

Another method that can be used for pricing is the characteristic function pricing method. It is important to note that this method is not directly related to the random grids approach. However, it is a useful method for testing that the option prices obtained using the random grids method are correct. This method is used to verify the price obtained for a call option under the Heston model.

A general Heston model is described as

$$dS(t) = \mu S(t)dt + \sqrt{v(t)}S(t)dW(t),$$
$$dv(t) = \kappa(\alpha - v(t))dt + \beta\sqrt{v(t)}dZ(t),$$

where $S(t)$ is the share price process and $v(t)$ is the volatility process. Note that $\kappa$, $\alpha$ and $\beta$ are constants, while $W(t)$ and $Z(t)$ are independent Brownian Motion processes.

A universal way of introducing the characteristic function would be by stating its definition. The necessary definition and pricing formula, as expressed by McWalter (2020), are provided below.

Suppose that $X$ is a random variable, then the characteristic function of $X$ is defined as

$$\phi_x(u) = \mathbb{E}[e^{iuX}].$$

Although a lot can be said about characteristic functions, one important thing to note is that each distribution has its own characteristic function. Once a characteristic function is known, the call option price can easily be determined.

For a share price distribution, such as the Heston model, it is useful to consider the random variable $S_T$, which represents the share price at time $T$. Therefore, the characteristic function for pricing under the Heston model described above, is expressed as

$$\begin{aligned}\phi_{s_T}(u) &= \mathbb{E}[e^{iuS_T}] \\ &= \exp(C + Dv(t_0) + iu\log(S(t_0))).\end{aligned}$$

In the expression above, the values for $C$ and $D$ are expressed as

$$C = rTiu + \alpha\kappa(Tx_- - \frac{1}{a}\log(\frac{1-ge^{-Td}}{1-g})) \text{ and}$$
$$D = \frac{1-e^{-Td}}{1-ge^{-Td}}x_-,$$

where

$$
\begin{aligned}
a &= \frac{\beta^2}{2}, \\
b &= \kappa - \rho\beta iu, \\
c &= -\frac{u^2 + iu}{2}, \\
d &= \sqrt{b^2 - 4ac}, \\
x\pm &= \frac{b \pm d}{2a}, \\
g &= \frac{x_-}{x_+},
\end{aligned}
$$

and where $r$ is the risk-free rate and $\rho$ is the correlation between the share price and volatility process.

Once the characteristic function is computed, the intermediary step to computing the call option price is to compute the values for $P_1$ and $P_2$. Values for $P_1$ and $P_2$ are computed as

$$
\begin{aligned}
P_1 &= 0.5 + 1/\pi \sum_{n=1}^{N} \mathrm{Re}\left[\frac{e^{-iu_n\kappa}\phi_{s_T}(u_n - i)}{iu_n\phi_{s_T}(-i)}\right]\Delta u, \\
P_2 &= 0.5 + 1/\pi \sum_{n=1}^{N} \mathrm{Re}\left[\frac{e^{-iu_n\kappa}\phi_{s_T}(u_n)}{iu_n}\right]\Delta u,
\end{aligned}
$$

where N represents the number of quadrature steps used, $u_n = (n - \frac{1}{2})\Delta u$ and $\Delta u = u_{\max}/N$. Finally, the expression for the price of the call option is give as

$$
C(K) = S_0 P_1 - K e^{-rT} P_2.
$$

Since the random grids approach is defined in terms of the stochastic local volatility model (2.1) and (2.2), parameters for $\sigma(t, S(t))$, $\theta$, $\epsilon$ and $\gamma$ are chosen. However, under certain conditions, the Heston parameters for the characteristic pricing function can be expressed in terms of the random grids parameters that are chosen. Therefore, the parameters used in the characteristic function pricing method, under the Heston model described above, are calculated as $\beta = \sigma(t, S(t))\epsilon$, $\kappa = \theta$, $\alpha = \sigma(t, S(t))^2$ and $v(t_0) = z(t_0)\sigma(t_0, S(t_0))$, as long as $\gamma = 0.5$ and $\mu = 0$.

## 2.6 Consequences of Assumptions Made

Andreasen and Huge (2011) made several assumptions to simplify their calculations. This section discusses the impact of these assumptions on the random grids pricing method.

### 2.6.1 Risk-Neutral Measure

The risk-neutral measure ($\mathbb{Q}$) is merely a way of assigning probabilities to a pricing event (such as the event of payoff when the option is exercised) in such a way that the probabilities are not influenced by the investor's risk preference. $\mathbb{Q}$ ensures that the discounted expected share price does indeed yield the time $t_0$ share price ($S(t_0)$) — i.e., $\mathbb{Q}$ ensure that no arbitrage exists.

The risk-neutral measure is also known as an equivalent martingale measure to the real-world probability measure ($\mathbb{P}$). Although $\mathbb{P}$ and $\mathbb{Q}$ assign different probabilities to the same events, it is important to note that the same collection of events are possible under both measures (i.e. $\mathbb{P}$ and $\mathbb{Q}$ have the same null-sets).

### 2.6.2 Forward Rates

As seen in (2.1), the stochastic process has no drift term. The fact that Andreasen and Huge (2011) assume zero rates implies that they are working under a drift-less process. As a result, the forward strike and forward share price are used to obtain the option prices. This requires an additional assumption to be made. The assumption is that the bond price is completely uncorrelated to the stock price and volatility. This may not be the most realistic assumption. However, it does not detract from the purpose of the random grids method, which is to obtain discrete price consistency across the various pricing methods.

### 2.6.3 Use of the ADI Schemes

Andreasen and Huge (2011) acknowledge that the ADI scheme is not the most accurate method, with an accuracy of order $O(\delta t + \delta x^2 + \delta y)$. In section 2.3.3, it is discussed that the ADI scheme is computationally more efficient than using a fully implicit scheme since the ADI scheme allows for the matrix equations to be solved using a standard Thomas algorithm.

### 2.6.4 Zero Correlation Between Stock Price and Volatility

It is very seldom that there would be a zero correlation between the stock price and the volatility process. However, assuming (2.3) simplifies calculations considerably. According to Andreasen and Huge (2011), the Markov-chain Monte-Carlo method does not work directly when a different correlation is applied. The random grids method can be extended to accommodate for this. However, that method is not discussed in this dissertation.

# 3 Implementation

This section provides a detailed description on how to implement the random grids approach. The description details pricing a call option under an $m \times n \times \tau$ grid. The definition for $x_{\text{range}}$, $y_{\text{range}}$ and $t_{\text{range}}$ are identical to those described in Section 2.1. When implementing the random grids approach, it is useful to consider $x_{\text{range}}$, $y_{\text{range}}$ and $t_{\text{range}}$ as column vectors. Furthermore, suppose that the current share price is $x(t_0) = x_a$ and the current volatility is $y(t_0) = y_b$.

Since $V$ represents the three-dimensional option price matrix, the form that it takes on at a particular time $t_h$ is

$$V(t_h, \cdot, \cdot) = \begin{bmatrix} V(t_h, x_0, y_0) & V(t_h, x_0, y_1) & \dots & V(t_h, x_0, y_{n-1}) \\ V(t_h, x_1, y_0) & V(t_h, x_1, y_1) & \dots & V(t_h, x_1, y_{n-1}) \\ \vdots & & & \\ V(t_h, x_{m-1}, y_0) & V(t_h, x_{m-1}, y_1) & \dots & V(t_h, x_{m-1}, y_{n-1}) \end{bmatrix}.$$

Therefore, $V(t_h, x_i, \cdot)$ represents a row vector containing the option price at time $t_h$ for each $y_j$ and at share price $x_i$. Alternatively, $V(t_h, \cdot, y_j)$ represents a column vector for the option price at time $t_h$ for each $x_i$, with volatility $y_j$ .

The implementation methods mentioned below describe the pricing of a call option using the BFD pricing method, the FFD pricing method as well as the Markov-chain Monte-Carlo method that is consistent with these finite difference methods.

## 3.1 Pricing Using the Backward Finite Difference Method

As discussed in Section 2.3, the goal of the BFD approach is to iteratively use the known option value at time $t_{h+1}$ ($V(t_{h+1}, \cdot, \cdot)$) to solve the BFD equations ((2.11) and (2.12)) and obtain the option value at the previous time step ($V(t_h, \cdot, \cdot)$).

In order to solve (2.11) and (2.12), the relevant $A_{x,y_j}$ and $A_{y,x_i}$ matrices must be computed. At each time ($t_h$), there are $n$ $A_{x,y_j}$ matrices ($A_{x,y_0}$, $A_{x,y_1}$, ..., $A_{x,y_{n-1}}$), where each $A_{x,y_j}$ matrix is defined by (2.9). Similarly, at each time ($t_h$), there are $m$ $A_{y,x_i}$ matrices ($A_{y,x_0}$, $A_{y,x_1}$, ..., $A_{y,x_{m-1}}$), where each $A_{y,x_i}$ matrix is defined by (2.10). Observe that the equation to generate an $A_{y,x_i}$ matrix, does not depend on

the $x_i$, nor the $t_h$ chosen, so each $A_{y,x_i} = A_y$ matrix is identical and time independent. Computing and storing (either by means of sparse matrices or by storing the diagonals as separate vectors) each $A_{x,y_j}$ matrix as well as the $A_y$ matrix may prove to be more computationally efficient than generating the relevant matrix each time it is needed. This is especially true if the volatility function is time independent (such as the Heston model with $\sigma(t_h, x_i) = 0.3x_i$ used in this dissertation). The reason for this is that if the volatility function is time independent, then the $A_{x,y_j}$ matrix is also time independent. Furthermore, since each $A_y$ matrix is identical (and therefore also time independent), the same $A_{x,y_j}$ and $A_y$ matrix is used to generate a particular grid point $V(t_h, x_i, y_j)$ at each time $t_h$. Unfortunately, such a convenience is not extended to the stochastic local volatility model with a volatility function that is dependent on both time and the share price. In that case, each $A_{x,y_j}$ matrix is time dependent, therefore a new $A_{x,y_j}$ matrix is re-calculated at each time $t_h$.

For the BFD method, the option payoff at maturity time $T = t_{\tau-1}$ is known for all grid points. Therefore the column vectors ($V(t_{\tau-1,\cdot,y_j})$) that are required for (2.11) are easily obtained. At maturity, the call option payoff for $y_0$ across all $x_i$ is the vector

$$V(t_{\tau-1,\cdot,y_0}) = \max(x_{\text{range}} - K, 0),$$

where $x_{\text{range}}$ is a column vector containing all discrete $x_i$. Since the call option payoff does not depend on $y_j$, the column vectors for $V(t_{\tau-1}, \cdot, y_1)$, $V(t_{\tau-1}, \cdot, y_2)$, ..., $V(t_{\tau-1}, \cdot, y_{n-1})$ are identical to $V(t_{\tau-1}, \cdot, y_0)$. Once the above quantities have been computed, the ADI equations can be solved.

The first step of the ADI scheme involves solving (2.11) backwards in time for the half time step option price. Since $A_{x,y_j}$ and $V(t_{\tau-1}, \cdot, y_j)$ have been computed for all $y_j$, (2.11) can be solved half a time step backward in time for $V(t_{\tau-2+1/2}, \cdot, y_j)$ for all $y_j$. Therefore, at maturity time $t_{\tau-1}$, the first step of the ADI scheme involves solving

$$A_{x,y_j} V(t_{\tau-2+1/2}, \cdot, y_j) = V(t_{\tau-1,\cdot,y_j})$$

for all $y_j$. The result will be $n$ column vectors (each with $m$ entries) for the option price, which can be used to form a single $m \times n$ matrix representing the option price at time $t_{\tau-2+1/2}$ for each $(x_i, y_j)$ combination ($V(t_{\tau-2+1/2}, \cdot, \cdot)$).

The second step of the ADI scheme uses the solution from (2.11) to solve (2.12) backward in time by another half time step to obtain the option price at the full time step $t_{\tau-2}$. This involves using the transpose of half time step row vectors (which are the column vectors $V(t_{\tau-2+1/2}, x_i, \cdot)^\top$) to solve for $V(t_{\tau-2}, x_i, \cdot)^\top$ for each $x_i$. Since

each $A_{y,x_i}$ matrix is the same, the process can be simplified. Instead of using each row vector to solve the equation for each $x_i$ ($m$ times), the entire $V(t_{\tau-2+1/2},\cdot,\cdot)^\top$ matrix can be used in (2.12) to solve for the entire $V(t_{\tau-1},\cdot,\cdot)^\top$ matrix. This is described by the matrix equation

$$A_y V(t_{\tau-2},\cdot,\cdot)^\top = V(t_{\tau-2+1/2},\cdot,\cdot)^\top,$$

where $A_y = A_{y,x_0} = A_{y,x_1} = ... = A_{y,x_{m-1}}$. Once the solution for $V(t_{\tau-2},\cdot,\cdot)^\top$ has been found, the transpose matrix ($V(t_{\tau-2},\cdot,\cdot)$) can be set as the initial condition to solve the ADI scheme at the next time step — i.e. solve (2.11) for $V(t_{\tau-3+1/2},\cdot,y_j)$ for each $y_j$, after which (2.12) would be used to solve for $V(t_{\tau-3},\cdot,\cdot)$. This process is to continue until $V(t_0,\cdot,\cdot)$ is obtained, which is an $m \times n$ matrix containing the option price for each value of $x_i$ and $y_j$ at time $t_0$.

Finally, the call option price at time $t_0$ is obtained by choosing the value in the $V(t_0,\cdot,\cdot)$ matrix that corresponds to today's share price and today's volatility. Therefore, if $x(t_0) = x_a$ and $y(t_0) = y_b$, then $V(t_0, x_a, y_b)$ represents the call option price at time $t_0$.

## 3.2 Pricing Using the Forward Finite Difference Method

While the BFD method evolved the option price matrix from maturity, backward in time to $t_0$, the FFD method evolves the probability density matrix forward in time to compute the expected payoff at maturity. A more detailed explanation of solving for the option price using the FFD equations is provided hereafter.

As stated in Section 2.3.4, the FFD equations use the adjoint operators, which are simply the transpose of the $A_{x,y_j}$ and $A_{y,x_i}$ matrices. Thus, similar to the BFD, at each time $t_h$ there are $n$ $A^\top_{x,y_j}$ matrices ($A^\top_{x,y_0}$, $A^\top_{x,y_1}$, ..., $A^\top_{x,y_{n-1}}$). Similarly, at each time $t_h$, there are $m$ $A^\top_{y,x_i}$ matrices ($A^\top_{y,x_0}$, $A^\top_{y,x_1}$, ..., $A^\top_{y,x_{m-1}}$). Note that since each $A_{y,x_i}$ matrix is identical, each $A^\top_{y,x_i}$ matrix is also identical and is denoted as $A^\top_y$.

As discussed in Section 2.3.2, the initial condition for the FFD is the joint probability density function for $x_i$ and $y_j$ at time $t_0$. Assuming that the time $t_0$ share price and volatility are known with certainty, the initial condition is set by the Dirac delta function. Thus if $x(t_0) = x_a$ and $y(t_0) = y_b$, then the initial condition for the forward finite difference is expressed as

$$p(t_0, x_i, y_j) = \mathbb{I}_{\{x_i = x_a\}}\mathbb{I}_{\{y_j = y_b\}}.$$

Since $p(t_0, x_i, y_j)$ is known for all $x_i$ and $y_j$, the $p(t_0,\cdot,\cdot)$ matrix is formed as a matrix of zeros, except for the one in the $a_{th}$ row and $b_{th}$ column.

Once the initial probability density matrix has been identified, the next step is to solve the FFD equations for the probability density at time $t_1$. As was the case

for the BFD equation, the FFD equation takes on the form of an ADI scheme where (2.13) is solved for the half time step density $p(t_{0+1/2}, \cdot, \cdot)$ and thereafter, (2.14) is solved for the density $p(t_1, \cdot, y_j)$ for each $y_j$.

As expressed by (2.13), since $p(t_{0+1/2}, \cdot, \cdot)^\top$ is an $n \times m$ matrix, the first step of the ADI scheme is to solve

$$A_y^\top p(t_{0+1/2}, \cdot, \cdot)^\top = p(t_0, \cdot, \cdot)^\top.$$

Since the $A_y^\top$ matrix is independent of $x_i$, the entire $p(t_{0+1/2}, \cdot, \cdot)^\top$ matrix is solved for (instead of solving for $p(t_{0+1/2}, x_i, \cdot)$ for each $x_i$ ($m$ times)). This completes the first step of the ADI scheme.

The second step of the ADI scheme, expressed by (2.14), uses the $m \times 1$ column vectors $p(t_{0+1/2}, \cdot, y_j)$ for each $y_j$, obtained from the first step of the ADI scheme, to solve for the density at the full time step ($p(t_1, \cdot, y_j)$) for each $y_j$. Therefore, the second step of the ADI scheme is completed by solving

$$A_{x,y_j}^\top p(t_1, \cdot, y_j) = p(t_{0+1/2}, \cdot, y_j),$$

for each $y_j$ ($n$ times). Joining the solution vectors together will form a single $m \times n$ probability matrix $p(t_1, \cdot, \cdot)$ for all $(x_i, y_j)$ combinations at time $t_1$.

Once the solution for $p(t_1, \cdot, \cdot)$ has been found, this matrix can be set as the initial condition to solve the ADI scheme at the next time step, i.e. solve (2.13) for $p(t_{1+1/2}, \cdot, \cdot)^\top$, after which (2.14) would be solved for $p(t_2, \cdot, \cdot)$. This process is to continue until the probability density matrix $p(T, \cdot, \cdot)$ is obtained, which contains the transition probabilities for each combination of $x_i$ and $y_j$ at maturity.

Once the probability density matrix has been formed at maturity, this matrix is used to compute the expected payoff of the call option at maturity. The call option payoff at time $T$ (payoff$(T, \cdot, \cdot)$) is described by an $m \times n$ matrix, where each matrix entry is set as

$$\text{payoff}(T, x_i, y_j) = \max(x_i - K, 0).$$

The expected payoff at time $T$ is obtained by performing element-wise multiplication of $p(T, \cdot, \cdot)$ with payoff$(T, \cdot, \cdot)$, and thereafter taking the sum of the sum along the rows, as shown in Section 2.3.4.

## 3.3 Markov-Chain Monte-Carlo Approach

The goal of the Markov-chain Monte-Carlo method is to simulate paths for the share price process ($x(t_h)$) and volatility process ($y(t_h)$) and thereafter, use these

paths to price the call option. As discussed in Section 2.4, the Markov-chain Monte-Carlo approach is constructed to be consistent with the finite difference approaches. As a result, the simulation for the stock price and volatility uses the finite difference operators to construct conditional transition probabilities, which are evolved forwards in time. The implementation of constructing these probabilities and simulating share price and volatility paths is described below.

### 3.3.1 Computing the Cumulative Transition Probability Matrices

The first step to simulate an $x(t_h)$ and $y(t_h)$ path is to compute the conditional transition probabilities. The transition probability matrices are formed by taking the inverse of the relevant $A_{x,y_j}$ and $A_y$ operators from the finite difference approach, as specified by (2.15) and (2.16).

Note that a more efficient algorithm for computing this inverse is discussed in the Appendix A. Furthermore, note that storing each $A^{-1}_{x,y_j}$ (if the volatility function is time-independent) and $A_y^{-1}$ matrix will improve efficiency by not needing to recalculate the inverse each time it is needed.

Suppose that the initial $x$-value is $x(t_0) = x_a$ and the initial $y$-value is $y(t_0) = y_b$. In that case, to generate an $x$ and $y$ value at the first time step $t_1$, only $A^{-1}_{x,y_b}$ and $A^{-1}_{y,x_a} = A_y^{-1}$ are needed. The cumulative transition probability matrices ((2.17) and (2.18)) are computed by accumulating (or summing) the probabilities found in the transition probability matrices $A^{-1}_{x,y_b}$ and $A_y^{-1}$ along their rows. The resulting matrices that are formed are $Q_{x,y_b}$ and $Q_y$. These cumulative transition probability matrices are used to simulate the $x_i$ and $y_j$ values at the next time step.

### 3.3.2 Simulating Share Price and Volatility Paths

Since the share price process ($x(t)$) depends on the volatility process ($y(t)$), it is necessary to generate a discrete path for both $x_i$ and $y_j$. The goal of the simulation process is to use the current value of $x_i$ and $y_j$ at time $t_0$, to simulate a value for $x_i$ and $y_j$ at time $t_1$. Since the cumulative transition probability matrices have been computed, the simulation algorithm described in Section 2.4.3 can be applied to generate an $x_i$ value at time $t_1$, conditional on $x_i$ taking on the value of $x(t_0)$ at time $t_0$ and conditional on $y_j$ taking on the value of $y(t_0)$ at time $t_0$. In addition, the simulation algorithm in Section 2.4.3 should also be used to generate a $y_j$ value at time $t_1$, conditional on $y_j$ taking on $y(t_0)$ and conditional on $x_i$ taking on $x(t_0)$ at time $t_0$.

Therefore, the algorithm successfully obtains $x(t_1)$ and $y(t_1)$. The process for generating the next $x_i$ value in the path will be identical to the first time step, with

the obvious exception that now the $A_{x,y_j}$ matrix chosen will be corresponding to the $y_j$ value generated at $y(t_1)$. Since the $A_y$ matrix is independent of $x_i$, it remains unchanged at each step. Since a call option depends on the value of the stock price at maturity, it is essential to continue simulating $x_i$ values until a path has been generated up until time $T$. Subsequently, a $y_j$ path will also have to be produced until time $T$.

Applying these steps correctly, leads to a complete share price path from $t_0$ until maturity. However, this is just one possible path, and in order to get a more accurate representation, a number of share price paths need to be generated. For instance, if the number of samples is set to $N$, then the algorithm will be performed $N$ times to have $N$ sample paths for $x$ and $y$. The process for finally computing the Markov-chain Monte-Carlo price is described below.

### 3.3.3 Markov-Chain Monte-Carlo Option Price

Once a share price path is simulated, the call option payoff at maturity, expressed as $V(T) = \max(x(T) - K, 0)$, is computed. This should be computed for each path so that $N$ payoffs at maturity are obtained. The option price at time $t_0$ is then computed using risk-neutral valuation. This involves computing the expected payoff of the call option at maturity. Since each path is equally likely to occur, an estimate for the expected payoff is just the mean of all $N$ payoffs.

This concludes the Markov-chain Monte-Carlo approach based on random grids. If it has been implemented correctly, then by construction, the results should be consistent with those obtained from the finite difference methods.

## 3.4 Determining the Model Parameters

The method described above highlights a general way to implement the random grids approach. What it does not explain, however, is how to determine some of the input parameters for yourself so that the random grids method can be adapted to suit your own needs. This section aims to give some insight on how to structure the random grids inputs when none are provided.

In the general implementation, the size of the grid was $m \times n \times \tau$. Once again, $x_{\text{range}}$, $y_{\text{range}}$ and $t_{\text{range}}$ take on the same values as those described in Section 2.1. Furthermore, $x(t_0)$ is set to $x_a$ and $y(t_0)$ is set to $y_b$.

When attempting to replicate the results obtained by Andreasen and Huge (2011), the grid was specified as being of size $200 \times 50 \times 25$. The values that the parameters would take on was $x_i \in [0, 4]$, $y_j \in [1, 5]$ and $t_h \in [0, 5]$. Furthermore, $x(t_0)$ was set to 1 and $y(t_0)$ was set to 1.

Because Andreasen and Huge (2011) provided the necessary parameters, implementing their random grids approach was relatively straightforward. The question that arises, however, is how to implement this random grids approach under different conditions. Clearly, the same range of $x$ values would not be appropriate if the share price today was significantly different (for example, if it was as large as \$800), and clearly, the same volatility range would not be appropriate if today's volatility was significantly different (for example if it was as small as 0.001). The following description aims to provide some insight into designing your own parameters to suit your unique needs.

The most attainable piece of information is clearly the time $t_0$ stock price. This can be obtained by using a popular data provider such as Reuters or Bloomberg. Therefore the input parameter for $x(t_0)$ is obtained easily. The input parameter for $y(t_0)$ can be estimated using a variety of methods. One method is to assume a share price and option pricing model and solve for the implied volatility. Alternatively, the volatility can be observed over the last few days, and a weighted average volatility can be estimated for the current time $t_0$. Another piece of information that is relatively straightforward is the start and endpoint for the time range grid. The starting point will be time $t_0$, and the ending point will be the maturity date of the option being priced. From this point on, it is up to interpretation and, at times, seems to be more of an art than a science. Nonetheless, a few general guidelines can be followed to yield accurate results.

The quantities that still need to be defined are the grid points for $x_{\text{range}}$ and $y_{\text{range}}$. When determining the start and endpoints for the share price grid, the key lies in knowing the underlying share price distribution. Note that, for example, the Heston model, as well as the stochastic local volatility model, are right-skewed distributions. It would therefore be reasonable to set the $x$ grid values such that the strike is roughly on the $25^{th}$ percentile, with a quarter of the $x_{\text{range}}$ values lying to the left of the strike and three-quarters of the $x_{\text{range}}$ lying to the right of the strike. This would ensure that most of the payoffs computed are indeed measurable by the grid. Note that if any other share price model was used, these numbers would need to be adapted accordingly to match that distribution. Unfortunately, this method is not as intuitive when undertaking the same procedure with the volatility process. The volatility grid values need to be chosen at the implementer's digression. Another decision that needs to be made is the size of these grids. Clearly, more steps in each dimension means more accuracy; however, that needs to be balanced with computational time. This decision is up to the implementer to decide; however, it is recommended to use a larger grid than those implemented by Andreasen and Huge (2011).

In summary, one can easily obtain information on the initial share price and the initial volatility. In addition, it is straightforward to estimate the start and endpoints for the time grid and the share price grid. Unfortunately, there is no easy way to decide on the volatility grid points, and the size of the grids should be large enough to ensure accuracy but still maintain computational tractability.

## 3.5 Efficient Algorithms

The random grids method uses numerous algorithms to achieve its purpose. This section discusses ways to improve the efficiency of the matrix inversion algorithm and the matrix solving algorithm used in the random grids scheme.

### 3.5.1 Efficient Algorithm for Computing Matrix Inverses

Recall that the inverse matrices $A_{x,y_j}^{-1}$ and $A_y^{-1}$ represent the transition probability matrices for generating a path for $x$ and $y$ values. A typical matrix inversion formula involves an inversion algorithm applied to an LU decomposition. For computing random grids, a more efficient algorithm for computing a matrix inverse is given by Andreasen and Huge (2011). According to the authors, the advantage of using their inversion method is that, unlike other inversion methods, it does not suffer from numerical overflow. The full algorithm with implementation details can be found in the Appendix A.

### 3.5.2 Efficient Algorithm for Solving Matrix Equations

Most compilers have a designated way of solving a matrix equation. To solve the BFD equations (2.11) and (2.12) and the FFD equations (2.13) and (2.14), common built-in functions may not be viable as the matrix sizes get larger. Instead, the entire problem can be solved using vectors alone, reducing each tridiagonal matrix into three vectors for each diagonal. From there, the Thomas algorithm, or alternatively the tridiag() algorithm from Press *et al.* (1988) can be used to efficiently solve the finite difference equations.

In most cases, the Thomas algorithm is sufficient to solve matrix equations. In fact, the Thomas algorithm is set as the default solver in many compilers. Despite this, Andreasen and Huge (2011) have implemented the Press *et al.* (1988) tridiag algorithm because it has a similar structure and numerical complexity to their matrix inversion algorithm mentioned above.

# 4 Results

This section details the results obtained when pricing a standard call option under the various pricing methods discussed earlier. The option is first priced under a Heston model, where the volatility function is time-independent. Thereafter, the process is repeated under the stochastic local volatility model, which requires a volatility function that depends on time. In both cases, the parameters used are similar to those used by Andreasen and Huge (2011).

## 4.1 Heston Model

As in the paper by Andreasen and Huge (2011), the grid size is set to $200 \times 50 \times 25$ ($m = 200$, $n = 50$ and $\tau = 25$). In addition, the following Heston parameters are used: $\sigma(t_h, x_i) = 0.3x_i$, $\theta = 1$, $\gamma = 0.5$, $\epsilon = 3$ and $T = 5$. Furthermore, $x_i \in [0, 4]$ with 200 equidistant steps, $y_j \in [1, 5]$ with 50 equidistant steps and $t_h \in [0, 5]$ over 25 equidistant steps. The time $t_0$ value of the share price and volatility are $x(t_0) = 1$ and $y(t_0) = 1$. Note that the strike is quoted as a percentage of the share price at time $t_0$.

The results are obtained by pricing the call option using different numerical pricing methods. The following methods are used: the characteristic function pricing method, the BFD method, the FFD method and the Markov-chain Monte-Carlo pricing method with a sample size of 100,000.

### 4.1.1 Option Price Under Varying Strikes

The following table provides the option prices obtained by implementing each pricing method. To investigate the option prices as the strike changes, three strikes are chosen: 50%, 100% and 200% of the current share price $x(t_0)$. In addition, the standard deviation for the Markov-chain Monte-Carlo method is displayed.

| Call Option Price under a 200x50x25 Grid | | | |
|---|---|---|---|
| Pricing Method | $K$ = 50% | $K$ = 100% | $K$ = 200% |
| BFD | 0.53573 | 0.26125 | 0.07111 |
| FFD | 0.53573 | 0.26125 | 0.07111 |
| Monte Carlo | 0.53538 | 0.25919 | 0.06869 |
| (Std Deviation) | (0.00217) | (0.00176) | (0.00121) |
| Heston Implied volatility for Call Option under a 200x50x25 Grid | | | |
| Pricing Method | $K$ = 50% | $K$ = 100% | $K$ = 200% |
| BFD | 0.29991 | 0.29830 | 0.29939 |
| FFD | 0.29991 | 0.29830 | 0.29939 |
| Monte Carlo | 0.29888 | 0.29586 | 0.29591 |

**Tab. 4.1:** Heston call option price and implied volatility under a 200x50x25 grid.

Observing the results in Table 4.1, it is important to note that the call option price for the FFD and BFD are identical in each case. When $K = 100\%$, both the FFD and BFD price under the Heston model is 0.26125. This indicates that the forward finite difference and backward finite difference methods are consistent with each other. This comes as no surprise since the methods are essentially the same, except working in reverse order, with the BFD solving for the option price backwards in time and the FFD solving for the probability density forwards in time, with the option price computed using risk-neutral pricing methods. Therefore, the use of the adjoint operators in the FFD method successfully reverses time to produce results that are consistent with the BFD method.

The results obtained for the Monte-Carlo method are more interesting. While the outcome price of 0.25919 is very close to the price obtained from FFD and BFD, it is not exact. This is as expected due to the Monte-Carlo error that arises from random number generation. The standard deviation provides an indication of the size of the Monte-Carlo error for the given number of samples. Observe that with a standard deviation of 0.00176, the Monte-Carlo price of 0.25919 is within three standard deviations of the finite difference price of 0.26125. As the sample size increases, the Markov-chain Monte-Carlo price will tend to the BFD and FFD prices. Therefore, the Markov-chain Monte-Carlo method obtained prices that are consistent with the BFD and FFD prices.

### 4.1.2 Implied Volatility as a Function of Strikes

The second portion of Table 4.1 demonstrates the implied volatilities obtained for each method, using varying strikes. Since the BFD and FFD prices are identical, it is not surprising that the implied volatilities obtained for both methods are identical. When the strike is set to 100% of $x(t_0)$, the implied volatility for both the FFD and BFD methods are 0.29830. Since the Monte-Carlo prices are slightly different to the finite difference prices, the Monte-Carlo implied volatilities are not identical to those obtained under the finite difference methods. However, the implied volatility of 0.29586 will tend to the finite difference implied volatility as the sample size tends to infinity.

In an attempt to further investigate how the implied volatility varies as the strike changes, the implied volatility curve in Figure 4.1 is obtained. Note that the parameters used are identical to those that obtained the results in Table 4.1 above.

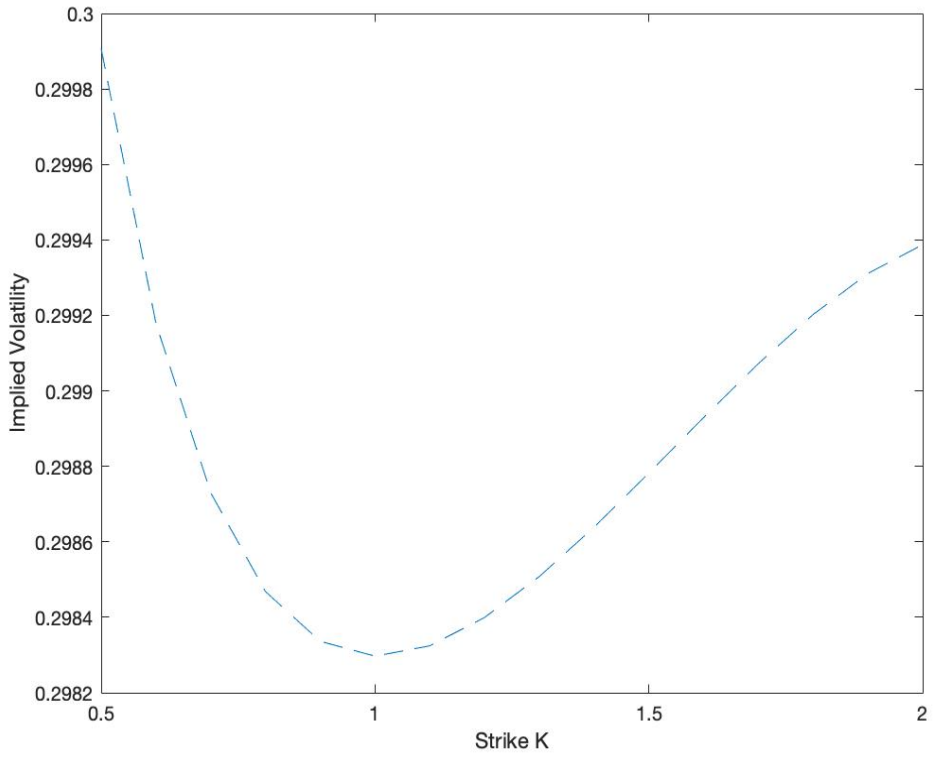


**Fig. 4.1:** Implied volatility curve under the Heston model.

Earlier in this dissertation, the concept of an implied volatility being a function of the strike was discussed. This result is shown in Figure 4.1, which depicts a

different implied volatility under different strikes. The shape clearly resembles a volatility "smile", with the minimum being the at-the-money strike. As the strike moves further out-the-money or further in-the-money, the implied volatility increases. This result shows the benefit of using a Heston model instead of the geometric Brownian motion under which the Black-Scholes implied volatilities are computed. Recall that the Black-Scholes model uses a constant volatility parameter and therefore cannot express implied volatility as a function of the strike.

### 4.1.3 Option Price under a Larger Finite Difference Grid

To investigate the accuracy of the random grids method, results obtained from characteristic function pricing are used (since no closed-form price exists for the Heston model). The parameters for $\sigma(t_h, x_i)$, $\theta$, $\gamma$ and $\epsilon$ are identical to the original grid, however, to get comparable results between the random grids methods and the characteristic function pricing method, a larger grid of $200 \times 100 \times 500$ ($m = 200$, $n = 200$ and $\tau = 500$) is used. In addition, the maturity is reduced to $T = 0.5$, a larger range of $x_i$ values are used and the same range of $y_j$ values are used ($x_i \in [0, 400]$ and $y_j \in [1, 5]$). Initial values for $x(t_0)$ are set to 100 and $y(t_0)$ remains at 1. The characteristic function pricing parameters that are used are $N = 500$ quadrature steps, $u_{\max} = 30$, $v(t_0) = 0.3^2$, $\kappa = 1$, $\alpha = 0.3^2$, $\beta = 0.3 \times 3$, $\rho = 0$ and $r = 0$.

| Call Option Price under a 200x100x500 Grid | | | |
|---|---|---|---|
| Pricing Method | $K$ = 50% | $K$ = 100% | $K$ = 200% |
| Characteristic | 50.05855 | 7.51628 | 0.11710 |
| BFD | 50.05208 | 7.51627 | 0.10402 |
| FFD | 50.05208 | 7.51627 | 0.10402 |
| Monte Carlo | 50.07376 | 7.53854 | 0.09794 |
| (Std Deviation) | (0.06792) | (0.04798) | (0.00785) |
| Heston Implied volatility for Call Option under a 200x100x500 Grid | | | |
| Pricing Method | $K$ = 50% | $K$ = 100% | $K$ = 200% |
| Characteristic | 0.41150 | 0.26684 | 0.41150 |
| BFD | 0.40568 | 0.26684 | 0.40561 |
| FFD | 0.40567 | 0.26684 | 0.40561 |
| Monte Carlo | 0.42362 | 0.26654 | 0.40363 |

**Tab. 4.2:** Heston call option price and implied volatility under a 200x100x500 grid.

In a similar way to the smaller finite difference grid, Table 4.2 shows that identical results are obtained for the BFD and FFD price and implied volatility across all

strikes. In addition, the Markov-chain Monte-Carlo prices are similar to the finite difference prices and will tend to the finite difference price as the sample size tends to infinity. Similarly, the characteristic function prices obtained are similar to the finite difference prices, but are not exact. The characteristic function price will tend to the finite difference price as the number of quadrature steps tends to infinity.

In Table 4.2, the characteristic function under a strike of 100% yielded a price of 7.51628. This is incredibly close to the price obtained from the FFD and BFD under the same parameters, but will improve as more quadrature steps are used. Considering the standard deviation of 0.04798, the Markov-chain Monte-Carlo price of 7.53854 is also consistent with the finite difference price — although this will improve as more samples are used. The fact that all three methods used to define the random grids approach obtains similar results to the characteristic function price indicates that the price generated by the Monte-Carlo random grids method is accurate.

Therefore it is apparent that under the Heston model, using the random grids approach to price options does indeed yield consistent results across the various pricing methods.

### 4.1.4 Effect of Sample Size on Monte Carlo Error

The Monte-Carlo estimate is computed for a range of sample sizes to investigate its convergence pattern. These estimates are obtained using the original grid size and parameters specified at the start of Section 4.1 with a strike of $K = 50\%$ of $x(t_0)$.

Figure 4.2, which depicts the call option price over a range of sample sizes, demonstrates that the accuracy of this Monte-Carlo estimate improves as the number of samples increases. This is demonstrated by the three standard deviation bound decreasing as the sample size increases. Since the standard deviation represents the error for prices obtained, figure 4.2 indicates that the error between the Monte-Carlo price and finite difference price decreases as the sample size increases.

In theory, it is possible to improve the accuracy of these Monte-Carlo solutions by using variance reduction techniques. Some variance reduction techniques attempt to generate more samples at less computational expense (such as the use of antithetic variables), while other techniques address the spread of the random numbers generated (such as the use of Quasi-random numbers instead of the default pseudo-random numbers used by most compilers).

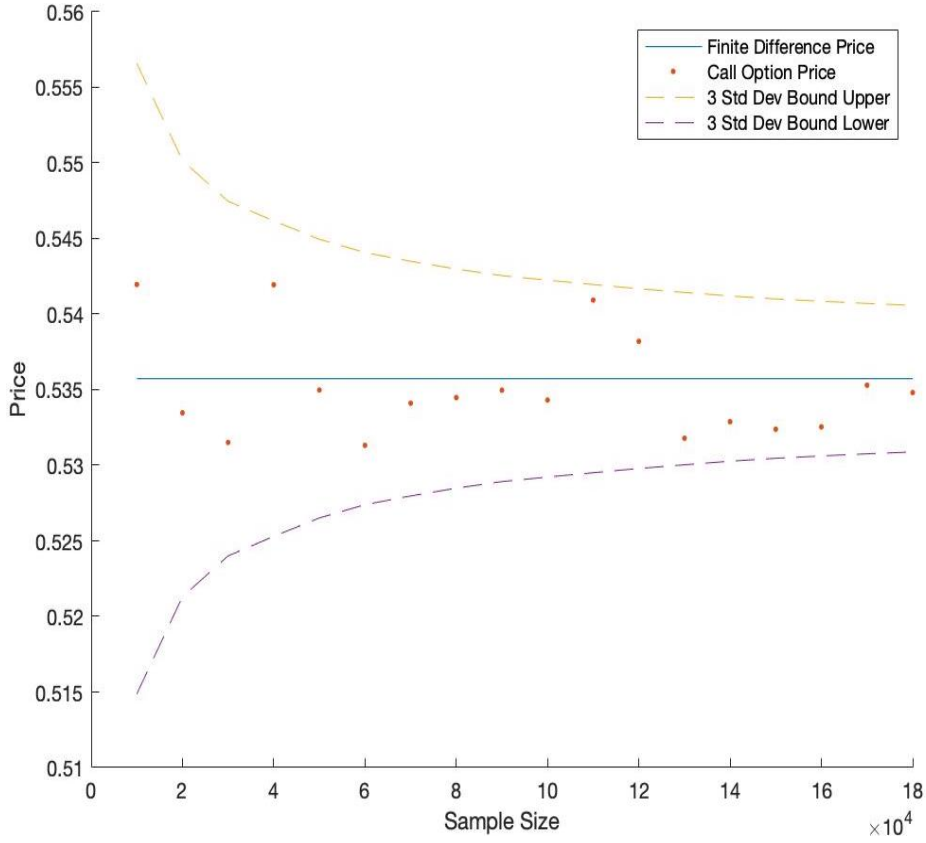


**Fig. 4.2:** Monte-Carlo estimates over a range of sample sizes.

## 4.2 Stochastic Local Volatility Model

Option prices are obtained using the random grids method with stochastic local volatility parameters. Under the stochastic local volatility model, the volatility function depends on both time and the share price. In this case, the volatility function $\sigma(t_h, x_i(t_h)) = 0.3x_i(t_h)(1 + t_h)^{-0.1}$ is used. The remaining parameters are set to $\theta = 1$, $\gamma = 0.5$, $\epsilon = 3$, $T = 5$, $x_i \in [0, 4]$ over 200 steps, $y_j \in [1, 5]$ over 50 steps and $t_h \in [0, 5]$ over 25 steps. In addition, $x(t_0) = 1$, $y(t_0) = 1$ and a sample size of 50,000 is used.

### 4.2.1 Option Price Under Varying Strikes

Table 4.3 below represents the call option price under varying levels of the strike for each pricing method. This is done while assuming that the share price follows a stochastic local volatility process.

The stochastic local volatility results in Table 4.3 show a similar pattern to those obtained under the Heston model. The BFD and FFD prices are identical for each

| Call option price under a 200x50x25 Grid. | | | |
|---|---|---|---|
| Pricing Method | $K$ = 50% | $K$ = 100% | $K$ = 200% |
| BFD | 0.52500 | 0.23345 | 0.04992 |
| FFD | 0.52500 | 0.23345 | 0.04992 |
| Monte Carlo | 0.52878 | 0.23270 | 0.04722 |
| (Std Deviation) | (0.00499) | (0.00396) | (0.00198) |
| Implied volatility for a call option under a 200x50x25 Grid. | | | |
| Pricing Method | $K$ = 50% | $K$ = 100% | $K$ = 200% |
| BFD | 0.26735 | 0.26554 | 0.26722 |
| FFD | 0.26735 | 0.26554 | 0.26722 |
| Monte Carlo | 0.27931 | 0.26468 | 0.26278 |

**Tab. 4.3:** Stochastic local volatility call option price and implied volatility under a 200x50x25 grid.

strike. Furthermore, the Markov-chain Monte Carlo prices obtained are within three standard deviations of the finite difference prices. For example, under a strike of $K = 50\%$, the BFD and FFD prices are identical at 0.52500. This price is reasonably consistent with the Monte-Carlo price of 0.52878 and a standard deviation of 0.00499.

As a result, it is clear that the BFD, FFD and Monte-Carlo prices are consistent with each other, with Monte-Carlo prices once again yielding Monte-Carlo error. Therefore, the random grids approach under the stochastic local volatility model is shown to produce prices that are consistent with finite difference prices.

### 4.2.2 Implied Volatility as a Function of Strikes

Since the BFD and FFD prices are identical, it is no surprise that their corresponding implied volatilities are also identical. Although the implied volatility under the Markov-chain Monte Carlo method is not identical, it is relatively close and will improve as the standard deviation decreases with more samples.

To further investigate how the implied volatility varies as the strike changes, the following volatility curve is obtained. Note that the parameters used are identical to those used in Table 4.3 above.

Under the stochastic local volatility model, Figure 4.3 shows that the implied volatility is a function of the strike.

The shape of the function is similar to that obtained from the Heston model, with the minimum implied volatility obtained at the at-the-money strike, the implied volatility of 0.26735 obtained at the strike of 50% and the implied volatility of

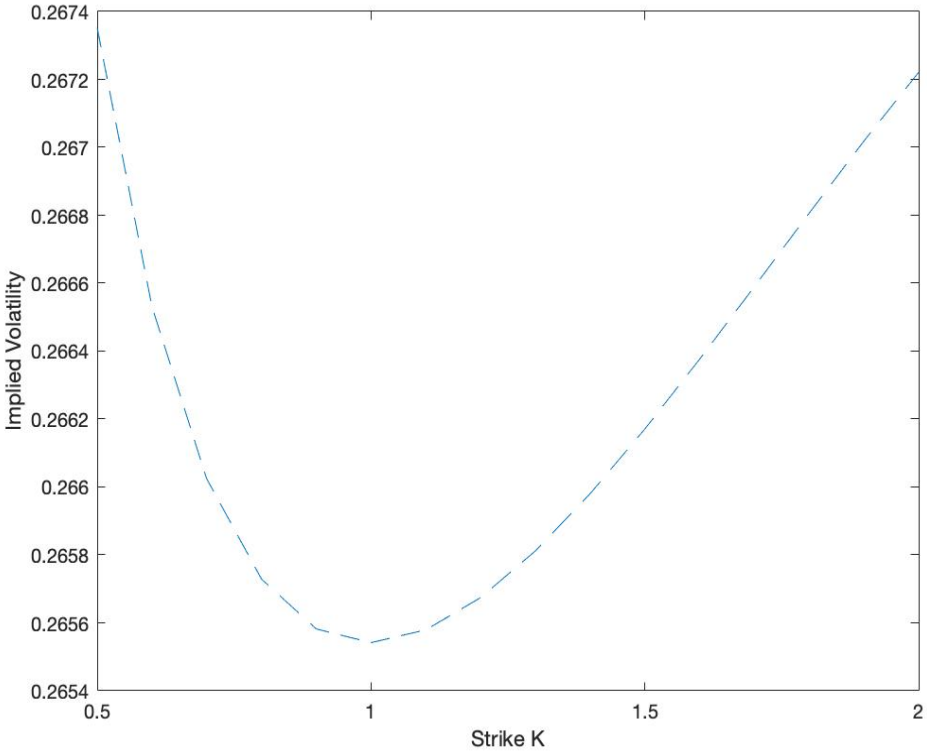


**Fig. 4.3:** Implied volatility curve under the stochastic local volatility model.

0.26722 obtained at the strike of 200%.

# 5 Conclusion

This dissertation highlighted the various aspects of share price modelling and took the Heston model and stochastic local volatility model as examples of such a model.

The main problem that this dissertation addressed is the fact that a particular option pricing method may be more appropriate for calibration, while another method may be more appropriate for pricing. More specifically, the problem arises when the parameters obtained from calibration under one method are used to price the instrument under another method. The bias that arises is in the form of model inconsistency, whereby the prices obtained are inaccurate because the calibration method and pricing method are not consistent with each other. To solve this problem, Andreasen and Huge (2011) developed a method called the random grids approach, which is designed to produce consistent results across the various pricing methods. This random grids method was implemented in this dissertation to determine if it did indeed generate consistent prices across the various methods.

Under the Heston model, results showed that the characteristic function, FFD, BFD and Markov-chain Monte-Carlo pricing methods were able to produce consistent prices for a call option. Furthermore, under the Heston model, the results showed that the implied volatilities were also consistent with each other and, as a function of the strike, displayed a clear volatility smile.

Under the stochastic local volatility model, the results were once again consistent across BFD, FFD and Markov-chain Monte-Carlo pricing methods. Furthermore, the implied volatility was shown to be a function of the strike, with the curve displaying another volatility smile.

In addition, this dissertation showed that although the prices are indeed consistent, there does exist Monte-Carlo error. As expected, the size of this error reduces as the sample size increases.

Although more realistic assumptions could have been used, none affect the viability of the method as a whole, except for the assumption of a zero correlation between share price and volatility processes. Even though the random grids approach was only applied to price a call option, the same methodology can be applied to price various other instruments by simply adapting the corresponding

payoff functions.

Overall, the random grids approach presented by Andreasen and Huge (2011) has been shown to accurately price derivative instruments, with the results obtained from each method being fully consistent.

# Bibliography


Andersen, L. B. (2007). Efficient simulation of the Heston stochastic volatility model, *Available at SSRN 946405* .

Andersen, L. and Brotherton-Ratcliffe, R. (1998). The equity option volatility smile: an implicit finite-difference approach, *Journal of Computational Finance* **1**(2): 5–37.

Andreasen, J. and Huge, B. (2011). Random grids, *Risk* **24**(7): 62.

Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities, *Journal of Political Economy* **81**(3): 637–654.

Craig, I. J. and Sneyd, A. D. (1988). An alternating-direction implicit scheme for parabolic equations with mixed derivatives, *Computers & Mathematics with Applications* **16**(4): 341–350.

Derman, E. and Kani, I. (1994). Riding on a smile, *Risk* **7**(2): 32–39.

Dupire, B. (1994). Pricing with a smile, *Risk* **7**(1): 18–20.

Gatheral, J. (2011). *The volatility surface: a practitioner's guide*, Vol. 357, John Wiley & Sons.

Glasserman, P. (2013). *Monte Carlo methods in financial engineering*, Vol. 53, Springer Science & Business Media.

Green, T. C. and Figlewski, S. (1999). Market risk and model risk for a financial institution writing options, *The Journal of Finance* **54**(4): 1465–1499.

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

McWalter, T. (2020). Characteristic function pricing, stochastic volatility & the fast Fourier transform, *Numerical Methods in Finance II Notes* .

Nabben, R. (1999). Decay rates of the inverse of nonsymmetric tridiagonal and band matrices, *SIAM Journal on Matrix Analysis and Applications* **20**(3): 820–837.

Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P. (1988). *Numerical recipes in C*, Cambridge university press Cambridge.

Schied, A. and Stadje, M. (2007). Robustness of delta hedging for path-dependent options in local volatility models, *Journal of Applied Probability* **44**(4): 865–879.

# A Appendix

## A.1 Decomposition of Tridiagonal Matrix to Compute Tridiagonal Inverse

### A.1.1 Decomposing $A_{x,y_k}$ and $A_y$

Recall that $A_{x,y_k}$ and $A_y$ are tridiagonal matrices. In the decomposition algorithm presented below, vector $a$ represents the lower diagonal, vector $b$ represents the main diagonal and vector $c$ represents the upper diagonal. It is important to ensure that the $a$ vector is padded with a zero at the beginning and that the $c$ vector is padded with a zero at the end so that vectors $a$, $b$ and $c$ are all the same length. This ensures that vectors $\tilde{x}$, $\tilde{y}$ and $\tilde{d}$ are also the same length.

The Algorithm Presented by Andreasen and Huge (2011) to decompose the tridiagonal matrix into vectors $\tilde{x}, \tilde{y}, \tilde{d}$ is presented here:

Suppose that the tridiagonal matrix has a lower diagonal $a$, main diagonal $b$ and upper diagonal $c$.

%Forward process:
$x(0) = 1$;
$x(1) = 1$;
for $i = 2 : n - 1$
$x(i) = x(i-1) - (a(i-1)/b(i-1)) * (c(i-2)/b(i-2)) * x(i-2)$;
end

%Backward Process
$y(n-1) = 1/(x(n-1) - (a(n-1)/b(n-1)) * (c(n-2)/b(n-2))x(n-2))$;
$y(n-2) = y(n-1)$;
for $i = n - 3 : 0$
$y(i) = y(i+1) - (a(i+2)/b(i+2)) * (c(i+1)/b(i+1)) * y(i+2)$;
end

% Forward Process for $d$
$d(0) = x(0) * y(0)/b(0)$;
for $i = 1 : n$
$d(i) = -(a(i)/b(i)) * (y(i)/y(i-1)) * d(i-1) + x(i) * y(i)/b(i)$;
end

Once the decomposition is complete, the following vectors will have been computed from $A_{x,y_k}$:
$\tilde{x}_x = [\tilde{x}_{x_0}, \tilde{x}_{x_1}, ...\tilde{x}_{x_{m-1}}]$,
$\tilde{y}_x = [\tilde{y}_{x_0}, \tilde{y}_{x_1}, ...\tilde{y}_{x_{m-1}}]$ and
$\tilde{d}_x = [\tilde{d}_{x_0}, \tilde{d}_{x_1}, ...\tilde{d}_{x_{m-1}}]$.

Similarly, from $A_y$ the decomposition vectors would be:
$\tilde{x}_y = [\tilde{x}_{y_0}, \tilde{x}_{y_1}, ...\tilde{x}_{y_{n-1}}]$,
$\tilde{y}_y = [\tilde{y}_{y_0}, \tilde{y}_{y_1}, ...\tilde{y}_{y_{n-1}}]$ and
$\tilde{d}_y = [\tilde{d}_{y_0}, \tilde{d}_{y_1}, ...\tilde{d}_{y_{n-1}}]$

### A.1.2 Using $\tilde{x}$, $\tilde{y}$ and $\tilde{d}$ to Compute the Transition Probability Matrix

The transition probability matrix for $x$ and $y$ is nothing more than the inverses $A^{-1}_{x,y_k}$ and $A^{-1}_y$. The components for these transition probability matrices are computed efficiently by using the $\tilde{x}$, $\tilde{y}$ and $\tilde{d}$ vectors as follows:

$$\sum_{j \leq i} A^{-1}_{ij} = \tilde{d}_i$$

$$A^{-1}_{ii} = \tilde{x}_i \tilde{y}_i / A_{ii},$$

$$A^{-1}_{ij} = A^{-1}_{i,j+1} . \frac{\tilde{x}_j}{\tilde{x}_{j+1}} . \frac{-A_{j+1,j}}{A_{jj}}, \text{ for } j < i \text{ and}$$

$$A^{-1}_{ij} = A^{-1}_{i,j-1} . \frac{\tilde{y}_j}{\tilde{y}_{j-1}} . \frac{-A_{j-1,j}}{A_{jj}}, \text{ for } j > i,$$

where $A^{-1}_{ij}$ represents the value that corresponds to the $i^{th}$ row and $j^{th}$ column of the $A^{-1}_{x,y_k}$ or $A^{-1}_y$ matrix, $\tilde{d}_i$ represents the $i^{th}$ entry of $\tilde{d}_x$ or $\tilde{d}_y$, $\tilde{x}_i$ represents the $i^{th}$ entry of $\tilde{x}_x$ or $\tilde{x}_y$ and $\tilde{y}_j$ represents the $j^{th}$ element of $\tilde{y}_x$ or $\tilde{y}_y$.