Spring 5-2019

Comparative Error Analysis of the Black-Scholes Equation

Chuan Chen
University of Southern Mississippi

The University of Southern Mississippi

Comparative Error Analysis of the Black-Scholes Equation

by

Chuan Chen

A Thesis
Submitted to the Honors College of
The University of Southern Mississippi
in Partial Fulfillment
of Honors Requirement

May 2019

Approved by

James V. Lambers, Ph.D., Thesis Adviser
Associate Professor of Mathematics

Bernd Schroeder, Ph.D., Director
School of Mathematics and Natural Sciences

Ellen Weinauer, Ph.D., Dean
Honors College

Abstract

Finance is a rapidly growing area in our banking world today. With this ever-increasing development come more complex derivative products than simple buy-and-sell trades. Financial derivatives such as futures and options have been developed stemming from the traditional stock, bond, currency, and commodity markets. Consequently, the need for more sophisticated mathematical modeling is also rising.

The Black-Scholes equation is a partial differential equation that determines the price of a financial option under the Black-Scholes model. The idea behind the equation is that there is a perfect and risk-free way for one to hedge the options by buying and selling the underlying asset in just the right way. This hedge implies that there is a unique and right price for the option, as returned by the Black-Scholes formula. Traditionally, the Black-Scholes equation is solved by first being reduced to a simple heat diffusion equation through exponentially scaling and changing the variables. This conversion ensures a simpler, faster, and more practical numerical scheme. However, there are several drawbacks with this method. Accuracy is often compromised and could be very unevenly distributed across the domain due to the variables being exponentially scaled. Transformation is also very limited and inflexible. For my research project, my adviser and I have proposed to solve the Black-Scholes equation directly using finite-difference schemes. By doing so, we can ensure robustness and accuracy. Error analysis of these two approaches was conducted. An assessment of the accuracy, efficiency, and robustness of each method is reported.

Key Words: Black-Scholes Equation, finite-difference methods, local truncation error

Acknowledgments

I want to express my sincerest gratitude to my research adviser, Dr. James Lambers, for all the help, encouragement, and inspiration he has given me during my academic and research journey. I could not have imagined a better mentor for my undergraduate study.

I would also like to thank my Honors thesis committee, Dr. Haiyan Tian, Dr. Huiqing Zhu, and Dr. Karen Kohl, for their helpful comments and suggestions.

Finally, I would like to say thank you to my parents, Suanrong Chen and Jianxiang Chen, for all of their love and support.

Table of Contents

List of Illustrations

Chapter 1

Introduction

Since the creation of the first stock exchange in 1604, various kinds of financial markets and derivative products have been developed and evolved. The most prominent ones are the stock markets, bond markets, currency markets, commodity markets, and the futures and options markets. This project focuses on the futures and options markets, which are derived from the traditional stock markets. Stocks, also known as equities or shares, are sold by companies to investors to raise money. The investors, or shareholders, then own part of the company and take part of its profit as dividends of so much per share. The stock represents the financial potential of the company and reflects the value of future dividends predicted by investors. This value is quantified by the price at which the stock is bought and sold. As markets expand and become more intricate, more evolved products than simple buy and sell have been developed. These are known as financial derivatives, and they offer investors more diverse ways and opportunities to venture. The futures and options markets trade one of the most common derivatives.

The simplest financial product in the futures and options market is the European call option. It is a contract between the writer, who draws up the contract, and the holder, who purchases the contract. It gives the holder the right to purchase a prescribed asset (the underlying asset) at a prescribed amount (exercise price) at a prescribed time in the future (expiry date). For the holder, this contract is not an obligation, so he may choose to not exercise it on the expiry day. However, the writer of the contract is obligated to sell if the holder chooses to purchase the asset. The holder of a call option would then want the asset price to rise. The value of this contract, which is paid at the time of opening, will be dependent on the financial potential of the prescribed asset and the prospective loss the writer will experience from selling the asset at the exercise price [1]. The European put option is similar except that it has the opposite payoff properties. Instead of the right

to purchase, the holder has the right to sell the asset at a prescribed amount on a specific date. The holder would then want the asset price to decrease as much as possible. A more complicated option that is traded commonly is the American option. This option allows the trade to be exercised at any time before the expiry. The modeling for the American option is, therefore, more interesting because it can be interpreted as a free boundary problem, which is a partial differential equation in an unknown domain. Not only is the option pricing a concern, but also the best time for it to be exercised. Two main concerns arise from the options market. First, how much the holder would be willing to pay for this right, and second, how can the writer reduce the risk accompanied by his obligation.

Dealing with options gives rise to the gearing effect where changes in the asset price are exaggerated. This can be seen in the following example. On 1 Jan 2019, one share of company X is valued at \$130, and a call option for one share of company X is drafted that gives the holder the option to buy this share at \$130 exactly one year later on 1 Jan 2020. Let us suppose that only two possible events will occur on the expiry date.

1. The price of the asset rises to \$150.

   In this scenario, the holder of the option can purchase the share at \$130 and immediately sell it for \$150 elsewhere, making an instant \$20 profit.

   $$\$150 - \$130 = \$20$$

2. The price of the asset falls to \$110.

   In this scenario, the holder would simply not exercise his rights because it would be illogical to purchase the share at \$130 when it can be bought at \$110 elsewhere.

Assuming that these two scenarios have equal probabilities of taking place, then, the expected profit to be made by the holder is

$$\frac{1}{2} \times \$20 + \frac{1}{2} \times \$0 = \$10$$

Thus, it seems reasonable to price this option of having a \$10 value. Let us suppose that the holder, as expected, paid \$10 for this option, and that the first scenario takes place. Then, the holder has made a net profit computed as follows:

$$
\begin{array}{ll}
\text{profit on exercise} = & \$20 \\
\text{cost of option} & = -\$10 \\
\hline
\text{net profit} & = \$10
\end{array}
$$

The net profit of \$10 gives the holder an 100% profit. On the other hand, if the second scenario takes place, the holder would also experience an 100% loss in a similar fashion. If, instead, the holder chooses to purchase the share directly for \$130 on 1 Jan 2019, then, the drop of \$20 in asset price would only cause him to lose about 15% of the initial investment.

In reality, valuing an option is not as simple as this, and many other variables such as interests and volatility have to be taken into account, but this straightforward example shows how the gain and loss of the investment can be exaggerated by dealing with options. It is then of vital concern to us to model the option prices as accurately as possible because we are dealing with a more rewarding but much riskier financial product. This thesis aims to investigate the shortcomings of existing methods for modeling option values and proposes a different approach for obtaining more accurate results.

Chapter 2

Literature Review

2.1. The Black-Scholes Model

One of the primary concepts in the theory of financial derivative pricing and hedging is arbitrage. This can be understood in layman's terms as "there is no free lunch." In financial terms, arbitrage is the concept that the opportunities to make an instantaneous risk-free profit cannot exist for a length of time significant enough before prices move to stop them [2]. Almost all finance theory assumes the existence of risk-free investments that give a guaranteed return. Good examples of such investments include government bonds and deposits in a reliable bank. The greatest risk-free return that one can make on a portfolio of assets is the same as the interest earned if the same amount of money were deposited in a bank. The keyword here is risk-free. By investing in equities, one can probably make more profit than depositing in the bank, but this cannot be certain. A greater return must come with greater risk.

Another premise of option pricing theory is that asset prices must move randomly because of the efficient market hypothesis [2]. This hypothesis basically states two things. First, the past history is fully reflected in the present price, which does not contain any additional information; second, markets react instantly to any new information about an asset. Thus, the modeling of asset values is about modeling the arrival of new information which influences the price. Before understanding asset pricing models, it has to be noted that absolute change in the asset price is not a valuable quantity: a fluctuation of 2 dollars is much more notable when the asset price is 10 dollars than when it is 100 dollars. Thus, with each asset price, instead of the absolute change, a return is associated. A return is defined to be the shift in the price divided by the original value. Since this is a relative value, it is a clearer indicator of change.

To model the return of a change in asset price, a small time interval of $\mathrm{d}t$, during which the asset price S changes to $S + \mathrm{d}S$ is considered ($\mathrm{d}\cdot$ is used to denote an infinitesimal change in any quantity). The most common model for a return, $\frac{\mathrm{d}S}{S}$, is separated into two components. One is a deterministic and expected return similar to the return on money invested in a risk-free bank. It is expressed as $\mu \mathrm{d}t$, where μ is the average rate of growth of the asset value. In a simplistic model, this rate of growth is assumed to be fixed. In more complicated models, this rate of growth is variable and can be a function of asset value and time. The second part of the $\frac{\mathrm{d}S}{S}$ model represents the random shift in the asset price in response to external and unpredictable effects, such as sudden news. It is expressed using a random sample from a normal distribution with mean zero and adds the term $\sigma\ \mathrm{d}X$ to the model. Here, σ is a quantity termed volatility, which measures the standard deviation of the returns, and $\mathrm{d}X$ is a random variable formed from a normal distribution. Combining these two equations together, a stochastic differential equation that models $\frac{\mathrm{d}S}{S}$ is obtained:

$$\frac{\mathrm{d}S}{S} = \sigma\ \mathrm{d}X + \mu\ \mathrm{d}t. \tag{2.1}$$

In real life, asset values are priced not continuously but at discrete time intervals. Thus, there exists a reasonable lower bound for the time-step $\mathrm{d}t$ of the model above. However, when used in practice to price options, this generates an unmanageably large amount of data. Therefore, it is more practical to set a continuous time limit of $\mathrm{d}t \to 0$ and solve the resulting differential equation. This model is a nice example of a random walk, which is a random process that describes a path which consists of a series of random steps. It fits real time series data very well and is also the starting point for more sophisticated models such as the Black-Scholes model.

The Black-Scholes model is a mathematical model developed by Fischer Black and Myron Scholes in the late 1960s. This model received the 1997 Nobel Memorial Prize in Economic Sciences and led to an upsurge in options trading around the world [3]. It uses the following basic assumptions:

- Asset prices follow the lognormal random walk $\frac{\mathrm{d}S}{S} = \sigma\ \mathrm{d}X + \mu\ \mathrm{d}t$.

- Risk-free interest rates and asset volatility are known functions over time.
- There are no transaction costs associated with trading and hedging a portfolio.
- The underlying asset does not pay dividends during the lifetime of the option.
- There are no arbitrage possibilities. All risk-free investments must gain the same return.
- Trading of the underlying asset can occur continuously.
- Short selling is allowed, and the assets are divisible.

The model's assumptions have been altered and generalized in many different directions since its development to suit different pricing needs. There are a plethora of variants of the model currently used in derivative pricing. However, the original Black-Scholes model still serves as the foundation of all option pricing models.

This model gives a partial differential equation, known as the Black-Scholes equation, that calculates the hypothetical value of an option over time. The fundamental notion behind the equation is that there is a way for one to perfectly hedge the option by buying and selling the underlying assets to eliminate risk (which makes the dynamically hedged portfolio earn the risk-free interest rate) [4]. This hedge, in turn, implies only one unique price for the option. Before introducing the partial differential equation, we first define the variables as follows:

- V is the value of an option. When a distinction is essential, C and P are used to denoted the value of a call and put option, respectively.
- t is the time.
- S is the current value of the underlying asset.
- σ is the volatility of the underlying asset.
- E is the exercise price.

- T is the time of expiry.
- r is the risk-free interest rate.

To the random walk for return (2.1), Itô's lemma can be applied, which relates a small change in the function of a random variable to a small change in the variable itself [5]. This identity gives the differential of a time-dependent function of a random walk and generates a more sophisticated model.

$$\mathrm{d}V = \sigma S \frac{\partial V}{\partial S}\mathrm{d}X + \left(\mu S \frac{\partial V}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + \frac{\partial V}{\partial t}\right)\mathrm{d}t \tag{2.2}$$

Now, suppose a portfolio is constructed with an undetermined fixed number $-\Delta$ of underlying assets and one option. The value of this portfolio would be

$$\Pi = V - \Delta S. \tag{2.3}$$

The change in the portfolio value in one time-step would be

$$\mathrm{d}\Pi = \mathrm{d}V - \Delta \mathrm{d}S. \tag{2.4}$$

Combining equations (2.1), (2.2), and (2.4) together, we obtain

$$\mathrm{d}\Pi = \sigma S\left(\frac{\partial V}{\partial S} - \Delta\right)\mathrm{d}X + \left(\mu S \frac{\partial V}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + \frac{\partial V}{\partial t} - \mu \Delta S\right)\mathrm{d}t. \tag{2.5}$$

To eliminate the random component, Δ is chosen to be

$$\Delta = \frac{\partial V}{\partial S} \tag{2.6}$$

at the start of the time-step $\mathrm{d}t$. It is a measure of the relationship between the changes in the option and the changes in the underlying asset. This results in the deterministic change in the portfolio value:

$$\mathrm{d}\Pi = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right)\mathrm{d}t. \tag{2.7}$$

Because of the assumption that there are no arbitrage possibilities and no transaction costs, the return on an amount Π invested in risk-free assets would be the same as a risk-free bank

deposit with a fixed interest rate r. The return on Π would then see a growth of $r\Pi\,\mathrm{d}t$ in a time t. Thus, we obtain

$$r\Pi\,\mathrm{d}t = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right)\mathrm{d}t. \tag{2.8}$$

After substituting equation (2.3) and (2.6) into (2.8), dividing by $\mathrm{d}t$ throughout, and rearranging, we obtain

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0.$$

This is the Black-Scholes equation. Under the assumptions stated previously, all derivative securities whose price depend solely on the current value of S and t must satisfy this equation. Since the value of a put option is solved in the exact way as a call option with the same solutions, just in the opposite direction, we are going to restrict our attention to the call option to avoid the repetition of information. This gives us the model

$$\frac{\partial C}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} + rS\frac{\partial C}{\partial S} - rC = 0. \tag{2.9}$$

The Black-Scholes equation is a partial differential equation and, by its nature, can be solved using numerical methods. However, it is a rather lengthy and cluttered system. Also, the equation steps backward in time with the final data given at $t = T$. To work around these issues, numerical analysts have been converting the Black-Scholes equation to the heat equation [7]:

$$\frac{\partial u}{\partial \tau} = \frac{\partial^2 u}{\partial x^2} \tag{2.10}$$

Conversion is achieved through the following variable transformations:

$$S = Ee^x,\ t = T - \tau/\frac{1}{2}\sigma^2,\ C = Ev(x,\tau), \tag{2.11}$$

where v is:

$$v = e^{-\frac{1}{2}(k-1)x - \frac{1}{4}(k+1)^2\tau}u(x,\tau),\ k = r/\frac{1}{2}\sigma^2 \tag{2.12}$$

The heat or diffusion equation is a partial differential equation that has been studied for almost two centuries as a model of the flow of heat in a continuous medium. It is one of the most successful and broadly applied mathematical models, and a substantial body of theory on its properties and solution is accessible [6]. As can be seen, it is a much simpler system

than the original Black-Scholes equation. Thus, the numerical scheme for solving the heat equation will be more straightforward, giving it the potential to be faster and more adaptable. This is the reason that computer algebra programs such as Maple solve the Black-Scholes equation using this conversion method in their built-in functions [7].

2.2. Finite-difference Methods

Finite-difference methods are methods used to obtain numerical solutions to partial differential equations. They are powerful techniques that can generate accurate solutions to many partial differential equations arising in different scientific fields, including financial sciences. The underlying idea behind these methods is to replace the partial derivatives in the equations by approximations produced by the Taylor expansions of functions near the point of interest [8]. For the heat equation, we can generate a Taylor series evaluated at the point $u(x, \tau + \delta\tau)$ centered at (x, τ) :

$$u(x, \tau + \delta\tau) \approx u(x, \tau) + \frac{\partial u}{\partial \tau}\delta\tau + \mathcal{O}(\delta\tau^2) \tag{2.13}$$

After rearranging, we obtain an approximation of the partial derivative

$$\frac{\partial u}{\partial \tau} \approx \frac{u(x, \tau + \delta\tau) - u(x, \tau)}{\delta\tau} + \mathcal{O}(\delta\tau) \tag{2.14}$$

When we regard $\delta\tau$ as an infinitesimal but nonzero value, we can use the approximation above to estimate the value of our partial derivative. This is called a finite-difference approximation. Equation (2.14) is a particular type of finite-difference formula called the forward difference because the differencing is in the forward τ direction.

Similarly, we can generate a Taylor series for $u(x, \tau - \delta\tau)$ and use that to obtain a backward difference approximation for the partial derivative

$$\frac{\partial u}{\partial \tau} \approx \frac{u(x, \tau) - u(x, \tau - \delta\tau)}{\delta\tau} + \mathcal{O}(\delta\tau) \tag{2.15}$$

The backward difference is used to generate an implicit finite-difference schemes for the partial differential equation. Both the forward and back difference approximations give a first order rate of convergence.

As it turns out, there is another finite-difference method that can be derived to give a faster convergence rate of $\mathcal{O}\left((\delta\tau)^2\right)$. We can obtain the centered difference approximation of $\frac{\partial u}{\partial \tau}$ by combining the Taylor series of both $u(x,\tau-\delta\tau)$ and $u(x,\tau+\delta\tau)$

$$u(x,\tau+\delta\tau) \approx u(x,\tau) + \frac{\partial u}{\partial \tau}\delta\tau + \mathcal{O}(\delta\tau)$$

$$u(x,\tau-\delta\tau) \approx u(x,\tau) - \frac{\partial u}{\partial \tau}\delta\tau + \mathcal{O}(\delta\tau)$$

and subtract them to get the partial derivative

$$\frac{\partial u}{\partial \tau} \approx \frac{u(x,\tau+\delta\tau) - u(x,\tau-\delta\tau)}{2\delta\tau} + \mathcal{O}\left((\delta\tau)^2\right) \tag{2.16}$$

Figure 2.1 shows a geometric interpretation of the three types of finite-difference approximations. It can be easily seen that the centered difference give more accurate approximations than either the forward or the backward difference.

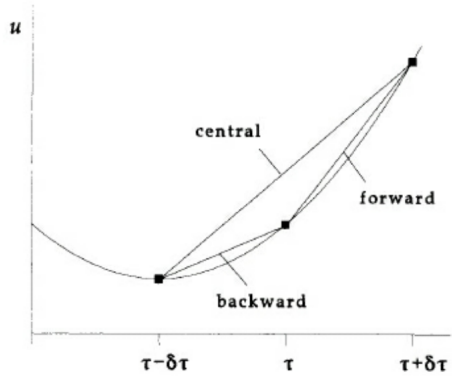

Figure 2.1: Forward, backward, and centered difference approximations [7].

When approximating second order partial derivatives, such as the $\frac{\partial^2 u}{\partial x^2}$ term from the heat equation (2.10), a symmetric finite-difference approximation can be obtained by applying the backward difference to the forward difference approximations of the first derivative.
Forward difference of the first derivative:

$$\frac{\partial u}{\partial \tau} \approx \frac{u(x,\tau+\delta\tau) - u(x,\tau)}{\delta\tau} + \mathcal{O}\left((\delta\tau)^2\right)$$

Backward difference of the second derivative:

$$\frac{\partial}{\partial \tau}\left(\frac{\partial u}{\partial \tau}\right) \approx \frac{\frac{u(x,\tau+\delta\tau)-u(x,\tau)}{\delta\tau} - \frac{u(x,\tau)-u(x,\tau-\delta\tau)}{\delta\tau}}{\delta\tau} + \mathcal{O}\left((\delta\tau)^2\right)$$

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u(x+\delta x,\tau)-2u(x,\tau)+u(x-\delta x,\tau)}{(\delta x)^2}+\mathcal{O}\left((\delta x)^2\right) \tag{2.17}$$

In fact, this difference approximation can also be obtained by applying the forward difference approximation to the backward difference of the first derivative, or by applying the centered difference approximation to the centered difference of the first derivative. Although there exist other methods for approximation, the symmetric centered-difference is preferred because the symmetry preserves the reflectional symmetry of the second-order partial derivative. The centered difference also makes the approximation more accurate than other methods.

Using different approximation methods, we can generate different numerical schemes to solve our partial differential equation. By using a forward difference for $\frac{\partial u}{\partial \tau}$ (2.14) and a symmetric difference for the second derivative $\frac{\partial^2 u}{\partial x^2}$ (2.17), after rearranging, we obtain the explicit method

$$\frac{u_n^{m+1}-u_n^m}{\delta\tau}=\frac{u_{n+1}^m-2u_n^m+u_{n-1}^m}{(\delta x)^2},$$

where $u_n^m = u(n\delta x, m\delta\tau)$. After rearranging our terms based on time indices, we obtain

$$u_n^{m+1}=\alpha u_{n+1}^m+(1-2\alpha)u_n^m+\alpha u_{n-1}^m$$

$$\alpha=\frac{\delta\tau}{(\delta x)^2}.$$

As we can see, we can explicitly calculate the value at time-step $m+1$ by using the values at time-step m. This makes the explicit method a very easy and straightforward computation. However, this numerical scheme has a stability limitation. We can rewrite our heat equation discretizing only in space to obtain a system of ordinary differential equations (ODE)

$$u_t = Au$$

where

$$A=QDQ^T=\frac{1}{(\delta x^2)}\begin{bmatrix}-2 & 1 & & & \\ 1 & -2 & 1 & & \\ & & \ddots & & \\ & & 1 & -2 & 1 \\ & & & 1 & -2\end{bmatrix},\ u^m=\begin{bmatrix}u_1\\ \vdots \\ u_N\end{bmatrix},$$

with D being a diagonal matrix of eigenvalues and the columns of Q being eigenvectors. We can transform our ODE into

$$v_t = Dv$$

where

$$v = Q^T u.$$

Then, applying the explicit scheme, we have

$$\begin{aligned} v_n^{m+1} &= v_n^m + \delta\tau\lambda_n v_n^m \\ &= (1+\delta\tau\lambda_n)v_n^m \end{aligned}$$

where λ_n is an eigenvalue of A and $-\frac{4}{(\delta x)^2} \leq \lambda_n \leq 0$. Because the heat solution decays over time, for stability, we must have

$$\begin{aligned} |1+\delta\tau\lambda_n| &\leq 1 \\ -1 \leq 1+\delta\tau\lambda_n &\leq 1 \\ -2 \leq -\delta\tau\frac{4}{(\delta x^2)} &\leq 0 \\ \frac{1}{2} \geq \frac{\delta\tau}{(\delta x)^2}&. \end{aligned}$$

Thus, when α is greater than one half, this scheme is unstable, and our rounding error grows in magnitude at each iteration of the process. This puts a huge limit on the size of the time-steps we can take. For instance, if we want to increase accuracy by using twice as much spatial steps, we must then take 4 times as many time steps to keep our scheme stable. This means that we will spend a significant amount of time on computation more than maybe what is needed.

To overcome the stability limitation, we can instead use the backward difference to approximate $\frac{\partial u}{\partial \tau}$. This generates the implicit method

$$\frac{u_n^{m+1} - u_n^m}{\delta\tau} = \frac{u_{n+1}^{m+1} - 2u_n^{m+1} + u_{n-1}^{m+1}}{(\delta x)^2}.$$

After rearranging, we obtain

$$\alpha u_{n-1}^{m+1} + (1+2\alpha)u_n^{m+1} - \alpha u_{n+1}^{m+1} = u_n^m$$

$$\alpha = \frac{\delta\tau}{(\delta x)^2}.$$

Even though we have to solve the values at time-step $m+1$ implicitly from values at time-step m, which means that we have to set up a linear system to find the solution, the implicit method is overall a more efficient scheme because now it allows us to take larger time-steps. Thus, the overall computation time would be less. The implicit method do not have the stability limitation because we have the transformation

$$\begin{aligned}(1-\delta\tau\lambda_n)v_n^{m+1} &= v_n^m \\ v_n^{m+1} &= \frac{1}{(1-\delta\tau\lambda_n)}v_n^m.\end{aligned}$$

Because λ_n is negative, $\left|\frac{1}{(1-\delta\tau\lambda_n)}\right|$ is always guaranteed to be less than 1, and the scheme is unconditionally stable. Since the explicit and implicit method use forward and backward differences, they both have a first order rate of convergence. As we recall, the centered difference has a second rate of convergence. So naturally, it makes sense for us to try to generate a numerical scheme using that to obtain a faster rate of convergence.

If we use the centered difference (2.16) to approximate $\frac{\partial u}{\partial \tau}$, we obtain the Leap-frog method

$$\frac{u_n^{m+1}-u_n^{m-1}}{2\delta\tau} = \frac{u_{n+1}^m - 2u_n^m + u_{n-1}^m}{(\delta x)^2},$$

and we have the transformation

$$v_n^{m+1} = v_n^{m-1} + 2\delta\tau\lambda_n v_n^m.$$

This equation is basically a 3-term recurrence relation

$$v_n^m = c_1 z_1^m + c_2 z_2^m$$

where c_1 and c_2 are coefficients and $z_{1,2} = \delta\tau\lambda_n \pm \sqrt{(\delta\tau\lambda_n)^2+1}$.
For $z = \delta\tau\lambda_n + \sqrt{(\delta\tau\lambda_n)^2+1} \geq 1$, we will always get growth in the solutions, which means that our scheme is unstable. Thus, centered differences of the form (2.16) are never used in practice because they always lead to unstable numerical schemes [7]. Instead, we use a

slightly varied version of the centered difference of the form

$$\frac{\partial u}{\partial \tau} \approx \frac{u(x,\tau+\frac{1}{2}\delta\tau)-u(x,\tau-\frac{1}{2}\delta\tau)}{\delta\tau}+\mathcal{O}\left((\delta\tau)^2\right) \tag{2.18}$$

where the Taylor expansion takes $\frac{1}{2}$ steps in each direction instead of 1 full step. This leads us to the Crank-Nicolson method.

The Crank-Nicolson scheme can be achieved by taking an average of the implicit and explicit methods, which gives

$$\frac{u_n^{m+1}-u_n^m}{\delta\tau}=\frac{1}{2}\left(\frac{u_{n+1}^m-2u_n^m+u_{n-1}^m}{(\delta x)^2}+\frac{u_{n+1}^{m+1}-2u_n^{m+1}+u_{n-1}^{m+1}}{(\delta x)^2}\right).$$

This is essentially using the centered difference approximation (2.18) derived from the Taylor expansion centered at the half-step $u(x,\tau+\frac{1}{2}\delta\tau)$. We can transform this scheme as

$$v_n^{m+1}=\frac{1+\frac{1}{2}\delta\tau\lambda_n}{1-\frac{1}{2}\delta\tau\lambda_n}v_n^m,$$

and the multiplier is also guaranteed to be ≤ 1 in absolute value. Thus, the Crank-Nicolson scheme is superior to the previous numerical schemes because it does not have the stability limitation, and it has a faster rate of convergence.

Chapter 3

Methodology

As we saw in Chapter 2, once the Black-Scholes equation is reduced to a heat equation (2.10), it is relatively simple to find solutions using finite-difference methods because the heat equation is a considerably simpler and less cluttered equation. Thus, it is much easier to solve the heat equation and then convert the solutions back into the numerical solutions of the Black-Scholes equation through the variable transformations in equation (2.11) than it is to solve the Black-Scholes equation directly. However, there may be several drawbacks of the conversion approach. The heat equation is elegant and straightforward, however, the variable transformations are exponentially scaled and cluttered. The rather uniform error produced by the numerical scheme for the heat equation could be exponentially multiplied and become very unevenly distributed after being transformed back into financial terms, compromising accuracy in the results. There are also many examples, such as the case of multi-factor models, where variable transformation either cannot be performed or would result in a scheme not that much simpler than the original equation. Thus, we cannot expect to count on the conversion to solve the Black-Scholes model for all occasions. To overcome these limitations, we have proposed to solve the Black-Scholes equation directly using finite-difference methods. We hypothesize that the resulting numerical scheme will be more accurate and robust with a more evenly distributed error across the domain.

To thoroughly investigate the differences, advantages, and disadvantages of both approaches, we solve the Black-Scholes equation using each method and apply the same parameters to both resulting schemes to analyze their behavior. We then conduct an error analysis of each approach in which the local truncation errors are derived, plotted, and compared to reveal the error distribution of each scheme.

Before we begin to solve the Black-Scholes partial differential equation, we must first return to one of the assumptions about the model. The value of an option is unique because

otherwise, there would be arbitrage possibilities. However, a partial differential equation on its own has many solutions. Thus, to ensure the uniqueness of the solution, final and boundary conditions must be imposed. From the Black-Scholes equation (2.9), we can solve for the value of a call option $C(S,t)$ with exercise price E and expiry date T.

The final condition, which is applied at $t = T$, comes from the theory of arbitrage. At the final time, the value of a call option is known with certainty to be the payoff:

$$C(S,T) = \max(S-E,0). \tag{3.1}$$

Our boundary conditions are applied at $S = 0$ and at $S \to \infty$. From the random walk model (2.1), we can see that if S is zero, dS is also zero, and the asset value can never change. Thus, the call option is worthless regardless of the time to expiry. Hence, we have

$$C(0,t) = 0 \tag{3.2}$$

as our first boundary condition. For our second boundary condition, we note that as asset price rises to infinity, it becomes ever more probable that the option will be exercised, and the magnitude of the exercise price becomes less significant. Hence, we have our second boundary condition [7]:

$$C(S,t) \sim S - Ee^{-r(T-t)},\ S \to \infty. \tag{3.3}$$

We first solve the Black-Scholes equation using the traditional method of reducing to the heat equation.

3.1. The Heat Equation Approach

As stated in Chapter 2, the Black-Scholes equation can be reduced to a simple heat equation through variable transformations (2.11). We can then solve the much simpler heat partial differential equation and convert our solutions back into financial variables. Because the heat equation steps forward in time, our final condition is converted to an initial condition

$$u(x,0) = \max\left(e^{\frac{1}{2}(k+1)x} - e^{\frac{1}{2}(k-1)x}, 0\right). \tag{3.4}$$

Our first boundary condition at $x = 0$ is transformed into

$$u(0,\tau) = 0, \tag{3.5}$$

and our second boundary condition at $x \to \infty$ becomes

$$u(x,\tau) \sim e^{\frac{1}{4}(k+1)^2\tau}\left(e^{\frac{1}{2}(k+1)\tau} - e^{\frac{1}{2}(k-1)\tau}e^{-k\tau}\right). \tag{3.6}$$

To solve the heat equation:

$$\frac{\partial u}{\partial \tau} = \frac{\partial^2 u}{\partial x^2},$$

we denote time index m and spatial index n, and we average the explicit scheme

$$\frac{u_n^{m+1} - u_n^m}{\delta\tau} \approx \frac{u_{n+1}^m - 2u_n^m + u_{n-1}^m}{(\delta x)^2}$$

with the implicit scheme

$$\frac{u_n^{m+1} - u_n^m}{\delta\tau} \approx \frac{u_{n+1}^{m+1} - 2u_n^{m+1} + u_{n-1}^{m+1}}{(\delta x)^2}$$

to obtain:

$$\frac{u_n^{m+1} - u_n^m}{\delta\tau} \approx \frac{1}{2}\left(\frac{u_{n+1}^m - 2u_n^m + u_{n-1}^m}{(\delta x)^2} + \frac{u_{n+1}^{m+1} - 2u_n^{m+1} + u_{n-1}^{m+1}}{(\delta x)^2}\right).$$

After rearranging, we arrive at the Crank-Nicolson scheme:

$$u_n^{m+1} - \frac{1}{2}\alpha(u_{n-1}^{m+1} - 2u_n^{m+1} + u_{n+1}^{m+1}) = u_n^m + \frac{1}{2}\alpha(u_{n-1}^m - 2u_n^m + u_{n+1}^m) \tag{3.7}$$

where $\alpha = \frac{\delta\tau}{(\delta x)^2}$.

Since the heat equation steps forward in time, everything on the right-hand side of equation (3.7) can be calculated explicitly. Solving this system is then reduced to first calculating

$$Z_n^m = (1-\alpha)u_n^m + \frac{1}{2}\alpha(u_{n-1}^m + u_{n+1}^m) \tag{3.8}$$

and then implicitly solving

$$(1+\alpha)u_n^{m+1} - \frac{1}{2}\alpha(u_{n-1}^{m+1} + u_{n+1}^{m+1}) = Z_n^m. \tag{3.9}$$

We can rewrite equation (3.9) as a linear system

$$Cu^{m+1} = b^m \tag{3.10}$$

where the matrix C is

$$C = \begin{bmatrix} 1+\alpha & -\frac{1}{2}\alpha & 0 & \cdots & 0 \\ -\frac{1}{2}\alpha & 1+\alpha & -\frac{1}{2}\alpha & & \vdots \\ 0 & -\frac{1}{2}\alpha & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & -\frac{1}{2}\alpha \\ 0 & 0 & & -\frac{1}{2}\alpha & 1+\alpha \end{bmatrix} \tag{3.11}$$

and vectors u^{m+1} and b^m are

$$u^{m+1} = \begin{bmatrix} u^{m+1}_{N^-+1} \\ \vdots \\ u^{m+1}_n \\ \vdots \\ u^{m+1}_{N^+-1} \end{bmatrix}, \; b^m = \begin{bmatrix} Z^m_{N^-+1} \\ \vdots \\ Z^m_n \\ \vdots \\ Z^m_{N^+-1} \end{bmatrix} + \frac{1}{2}\alpha \begin{bmatrix} u^{m+1}_{N^-} \\ 0 \\ \vdots \\ 0 \\ u^{m+1}_{N^+} \end{bmatrix} \tag{3.12}$$

where N^- and N^+ denote the lower and upper spatial boundary terms respectively. After establishing the linear system and imposing the initial and boundary conditions, the solution u is obtained by solving

$$u^{m+1} = (C)^{-1} b^m = C \backslash b^m. \tag{3.13}$$

The solution is then transformed back into call option values via equation (2.11) and (2.12).

The MATLAB algorithm (`crankheatdirecttrans.m`) was coded which solves the Black-Scholes equation using this approach.

3.2. The Direct Approach

To solve the Black-Scholes equation

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0 \tag{3.14}$$

directly, we derive the following finite-difference approximations:

$$\frac{\partial V}{\partial t} = \frac{V^{m+1}_n - V^m_n}{\delta t} \tag{3.15}$$

$$\frac{\partial V}{\partial S} = \frac{1}{2}\left(\frac{V_{n+1}^{m} - V_{n-1}^{m}}{2\delta S} + \frac{V_{n+1}^{m+1} - V_{n-1}^{m+1}}{2\delta S}\right). \tag{3.16}$$

$$\frac{\partial^2 V}{(\partial S)^2} = \frac{1}{2}\left(\frac{V_{n+1}^{m} - 2V_{n}^{m} + V_{n-1}^{m}}{(\delta S)^2} + \frac{V_{n+1}^{m+1} - 2V_{n}^{m+1} + V_{n-1}^{m+1}}{(\delta S)^2}\right) \tag{3.17}$$

Substituting (3.15), (3.16), and (3.17) back into the equation and rearranging the terms, we obtain the Crank-Nicolson scheme for the Black-Scholes equation:

$$\begin{aligned}\left(1 - \frac{\delta t}{2}r - \frac{1}{2}\alpha\sigma^2 S^2\right)V_n^{m+1} + \left(\frac{1}{4}\frac{\delta t}{\delta S}rS + \frac{1}{4}\alpha\sigma^2 S^2\right)V_{n+1}^{m+1} + \left(-\frac{1}{4}\frac{\delta t}{\delta S}rS + \frac{1}{4}\alpha\sigma^2 S^2\right)V_{n-1}^{m+1} \\ = \left(1 + \frac{\delta t}{2}r + \frac{1}{2}\alpha\sigma^2 S^2\right)V_n^{m} + \left(-\frac{1}{4}\frac{\delta t}{\delta S}rS - \frac{1}{4}\alpha\sigma^2 S^2\right)V_{n+1}^{m} + \left(\frac{1}{4}\frac{\delta t}{\delta S}rS - \frac{1}{4}\alpha\sigma^2 S^2\right)V_{n-1}^{m}\end{aligned} \tag{3.18}$$

where $\alpha = \frac{\delta t}{(\delta S)^2}$.

Since the Black-Scholes equation steps backward in time with given final condition, the left-hand side of this scheme can be calculated explicitly. Thus, solving this system is then reduced to first calculating

$$Z_n^{m+1} = \left(1 - \frac{\delta t}{2}r - \frac{1}{2}\alpha\sigma^2 S^2\right)V_n^{m+1} + \left(\frac{1}{4}\frac{\delta t}{\delta S}rS + \frac{1}{4}\alpha\sigma^2 S^2\right)V_{n+1}^{m+1} + \left(-\frac{1}{4}\frac{\delta t}{\delta S}rS + \frac{1}{4}\alpha\sigma^2 S^2\right)V_{n-1}^{m+1}$$

and then implicitly solving

$$\left(1 + \frac{\delta t}{2}r + \frac{1}{2}\alpha\sigma^2 S^2\right)V_n^{m} + \left(-\frac{1}{4}\frac{\delta t}{\delta S}rS - \frac{1}{4}\alpha\sigma^2 S^2\right)V_{n+1}^{m} + \left(\frac{1}{4}\frac{\delta t}{\delta S}rS - \frac{1}{4}\alpha\sigma^2 S^2\right)V_{n-1}^{m} = Z_n^{m}. \tag{3.19}$$

Equation (3.19) can then be rewritten as a linear system

$$BV^m = d^{m+1} \tag{3.20}$$

where the matrix B is

$$B = \begin{bmatrix} \left(1 + \frac{\delta t}{2}r + \frac{1}{2}\alpha\sigma^2 S^2\right) & \left(-\frac{1}{4}\frac{\delta t}{\delta S}rS - \frac{1}{4}\alpha\sigma^2 S^2\right) & 0 & \cdots & 0 \\ \left(\frac{1}{4}\frac{\delta t}{\delta S}rS - \frac{1}{4}\alpha\sigma^2 S^2\right) & \ddots & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \left(-\frac{1}{4}\frac{\delta t}{\delta S}rS - \frac{1}{4}\alpha\sigma^2 S^2\right) \\ 0 & 0 & & \left(\frac{1}{4}\frac{\delta t}{\delta S}rS - \frac{1}{4}\alpha\sigma^2 S^2\right) & \left(1 + \frac{\delta t}{2}r + \frac{1}{2}\alpha\sigma^2 S^2\right) \end{bmatrix} \tag{3.21}$$

and vectors V^m and d^{m+1} are

$$V^m = \begin{bmatrix} V^m_{N^-+1} \\ \vdots \\ V^m_n \\ \vdots \\ V^m_{N^+-1} \end{bmatrix}, \; d^{m+1} = \begin{bmatrix} Z^{m+1}_{N^-+1} \\ \vdots \\ Z^{m+1}_n \\ \vdots \\ Z^{m+1}_{N^+-1} \end{bmatrix} - \left(\frac{1}{4}\frac{\delta t}{\delta S} rS - \frac{1}{4}\alpha\sigma^2 S^2 \right) \begin{bmatrix} V^m_{N^-} \\ 0 \\ \vdots \\ 0 \\ V^m_{N^+} \end{bmatrix} \tag{3.22}$$

where N^- and N^+ denotes the lower and upper spacial boundary terms respectively.

After setting up the linear system, the value of the option can be obtained directly by solving

$$V^m = (B)^{-1} d^{m+1} = B \backslash d^{m+1} \tag{3.23}$$

with the final and boundary conditions given by (3.1), (3.2), and (3.3). The MATLAB algorithm (`crankblackh.m`) was coded which solves the Black-Scholes equation using this approach.

3.3. Error Analysis

Now that we have obtained the numerical schemes for both approaches, we can seek the local truncation error (LTE) and further analyze the convergence behavior of each method. The local truncation error of a numerical method is an estimate of the error generated from a single iteration of that method [9]. Because it is the residual from each iteration, we can calculate its value by approximating both sides of the numerical scheme using Taylor series and subtract them from each other to obtain the residual.

From our Crank-Nicolson scheme for the heat equation

$$u^{m+1}_n - \frac{1}{2}\alpha(u^{m+1}_{n-1} - 2u^{m+1}_n + u^{m+1}_{n+1}) = u^m_n + \frac{1}{2}\alpha(u^m_{n-1} - 2u^m_n + u^m_{n+1}), \tag{3.24}$$

we Taylor expand both sides around center $u(x_n, \tau_m)$ to the fourth degree. After rearrangements simplification, we obtain the error

$$\text{LTE for heat} = \frac{1}{2}(2u_{xx} + \frac{1}{6}u_{xxxx}(\delta x)^2 + u_{xx\tau}\delta\tau + \frac{1}{2}u_{xx\tau\tau}(\delta\tau)^2) - u_\tau - \frac{1}{2}u_{\tau\tau}\delta\tau.$$

Substituting $u_\tau = u_{xx}$, we simplify to

$$\text{LTE for heat} = (\delta x)^2 \cdot \frac{1}{12}u_{\tau\tau} + (\delta\tau)^2 \cdot \frac{1}{4}u_{\tau\tau\tau}.$$

Although this error formula looks simple and brief, it is not of much use to us. Because the solutions from the heat equations were transformed back into financial variables, the error terms must be transformed as well for it to disclose information about the final solution.

From the variable transformation equations (2.11) and (2.12), we obtain that

$$\delta x = \frac{1}{S} \cdot \delta S \tag{3.25}$$

and

$$\delta \tau = -\frac{1}{2}\sigma^2 \delta t. \tag{3.26}$$

From the transformation

$$u = \frac{1}{E} \cdot e^{\frac{1}{2}(k-1)x + \frac{1}{4}(k+1)^2\tau} \cdot V,$$

we also obtain

$$u_\tau = \frac{1}{4E}(k+1)^2 e^q \cdot V - \frac{2}{\sigma^2 E} e^q \cdot V_t \tag{3.27}$$

$$u_{\tau\tau} = \frac{1}{16E}(k+1)^4 e^1 \cdot V - \frac{1}{\sigma^2 E}(k+1)^2 e^q \cdot V_t + \frac{4}{\sigma^4 E} e^1 V_{tt} \tag{3.28}$$

$$u_{\tau\tau\tau} = \frac{1}{64E}(k+1)^6 e^q \cdot V - \frac{3}{8\sigma^2 E}(k+1)^4 e^1 \cdot V_t + \frac{3}{\sigma^4 E}(k+1)^2 e^q \cdot V_{tt} - \frac{8}{\sigma^6 E} e^q \cdot V_{ttt}, \tag{3.29}$$

where $e^q = e^{\frac{1}{2}(k-1)x + \frac{1}{4}(k+1)^2\tau}$. Substituting these term back into the LTE equation, we obtain

$$\begin{aligned} \text{LTE for heat} = {} & \frac{1}{12S^2}(\delta S)^2 e^q \left(\frac{1}{16E}(k+1)^4 V - \frac{1}{\sigma^2 E}(k+1)^2 V_t + \frac{4}{\sigma^4 E} V_{tt} \right) \\ & + \frac{1}{16}\sigma^4 (\delta t)^2 e^q \left(\frac{1}{64E}(k+1)^6 V - \frac{3}{8\sigma^2 E}(k+1)^4 V_t + \frac{3}{\sigma^4 E}(k+1)^2 V_{tt} - \frac{8}{\sigma^6 E} V_{ttt} \right). \end{aligned} \tag{3.30}$$

This is a very messy and cluttered error formula, and it is exponentially scaled.

From our Crank-Nicolson scheme for the Black-Scholes equation

$$\begin{aligned} & \left(1 - \frac{\partial t}{2} r - \frac{1}{2}\alpha\sigma^2 S^2\right) V_n^{m+1} + \left(\frac{1}{4}\frac{\partial t}{\partial S} rS + \frac{1}{4}\alpha\sigma^2 S^2\right) V_{n+1}^{m+1} + \left(-\frac{1}{4}\frac{\partial t}{\partial S} rS + \frac{1}{4}\alpha\sigma^2 S^2\right) V_{n-1}^{m+1} \\ & = \left(1 + \frac{\partial t}{2} r + \frac{1}{2}\alpha\sigma^2 S^2\right) V_n^m + \left(-\frac{1}{4}\frac{\partial t}{\partial S} rS - \frac{1}{4}\alpha\sigma^2 S^2\right) V_{n+1}^m + \left(\frac{1}{4}\frac{\partial t}{\partial S} rS - \frac{1}{4}\alpha\sigma^2 S^2\right) V_{n-1}^m, \end{aligned} \tag{3.31}$$

we Taylor expand both sides around center $V(S_n, t_m)$ to the third degree. After simplification and rearrangements, we obtain

$$\begin{aligned} \text{LTE for direct} = (\delta t)^2 \left(-\frac{1}{4} r V_{tt} + \frac{1}{4} r S V_{stt} + \frac{1}{6} V_{ttt} \right) \\ + (\delta s)^2 \left(\frac{1}{6} r S V_{SSS} \right) + (\delta t)^3 \left(-\frac{1}{12} r V_{ttt} \right). \end{aligned} \tag{3.32}$$

As can be seen, the error formula for the direct approach is much simpler and contains no exponential component.

Chapter 4

Results

To analyze the different behavior of each approach, we apply the following parameters to both schemes:

$$S = \begin{bmatrix} 0.05 \\ 0.10 \\ \vdots \\ 15.95 \\ 16 \end{bmatrix}, \delta S = 0.05, \delta t = 0.01, E = 10, T = 1, r = 0.1, \sigma = 3.$$

Solutions from both schemes were generated by the corresponding MATLAB algorithms developed in the previous chapter. Solutions obtained from the heat equation scheme are shown in Figure 4.1, and solutions from the direct approach are shown in Figure 4.2.

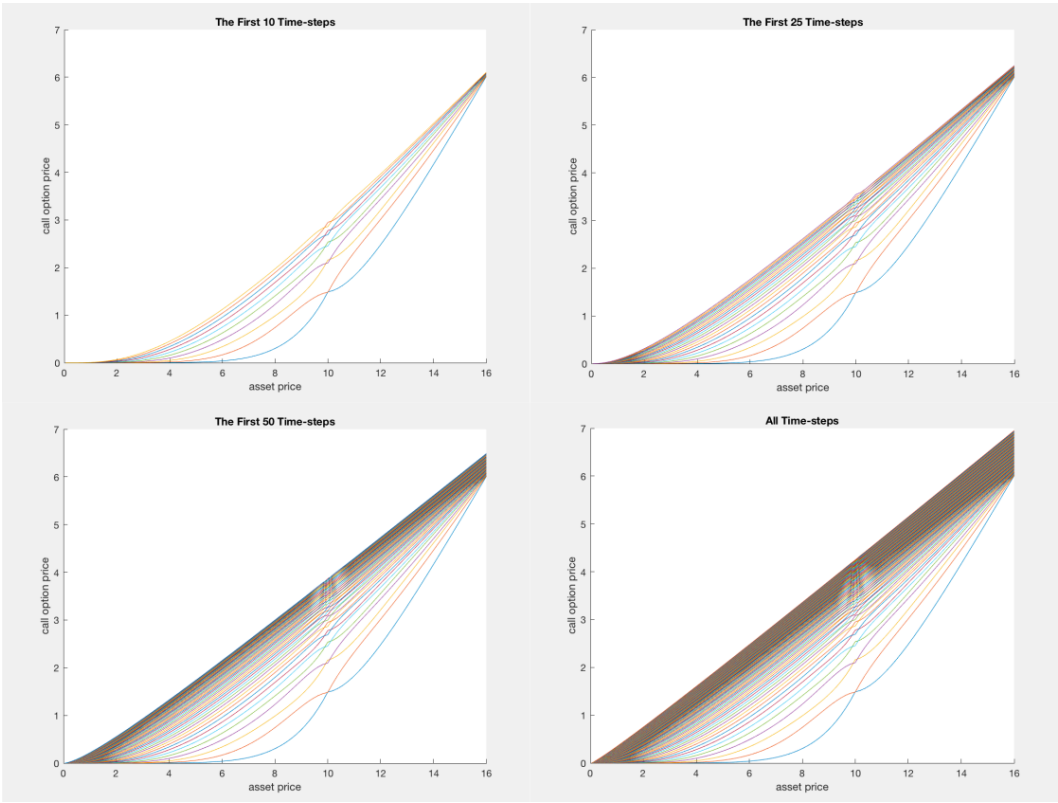

Figure 4.1: Progression of solutions from the heat equation approach.

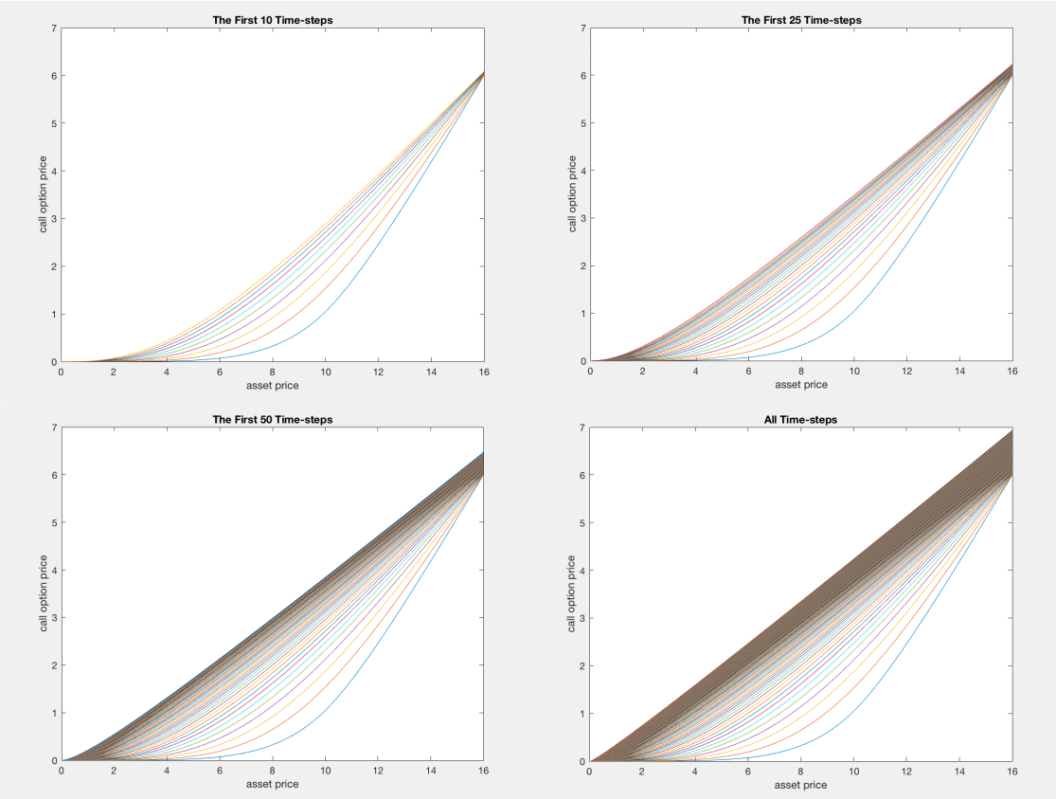

Figure 4.2: Progression of solutions from the direct approach.

It can be observed that both schemes converge to the same result at a similar rate. However, solutions from the heat equation scheme contain a kink at around $S = 10$, which represents a discontinuity in the higher order derivatives. The solutions from the direct approach appear smooth and contain no such discontinuities.

To better understand the error distribution of both approaches, the local truncation errors in the last ten steps from both schemes were analyzed and plotted. For the direct scheme, as shown in Figure 4.3, the local truncation error stays within the range of 10^{-5}. For the heat equation, as shown in Figure 4.4, LTE ranges from 10^{-9} to 10^{-3}. Although, overall, the heat equation scheme yields a smaller average error, we can see that the error is very unevenly distributed. The spike in the error around $S = 10$ also corresponds to the discontinuity we saw in the solutions before. When we take the infinity norm of both errors, the heat equation scheme will yield an error two orders of magnitude higher than the direct scheme.

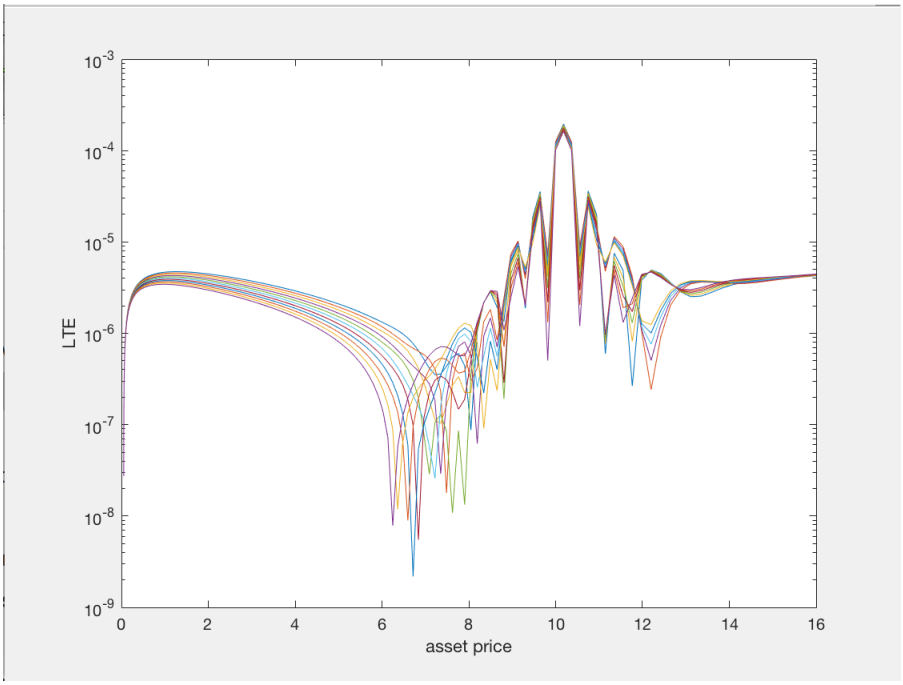

Figure 4.3: LTE from the heat equation approach.

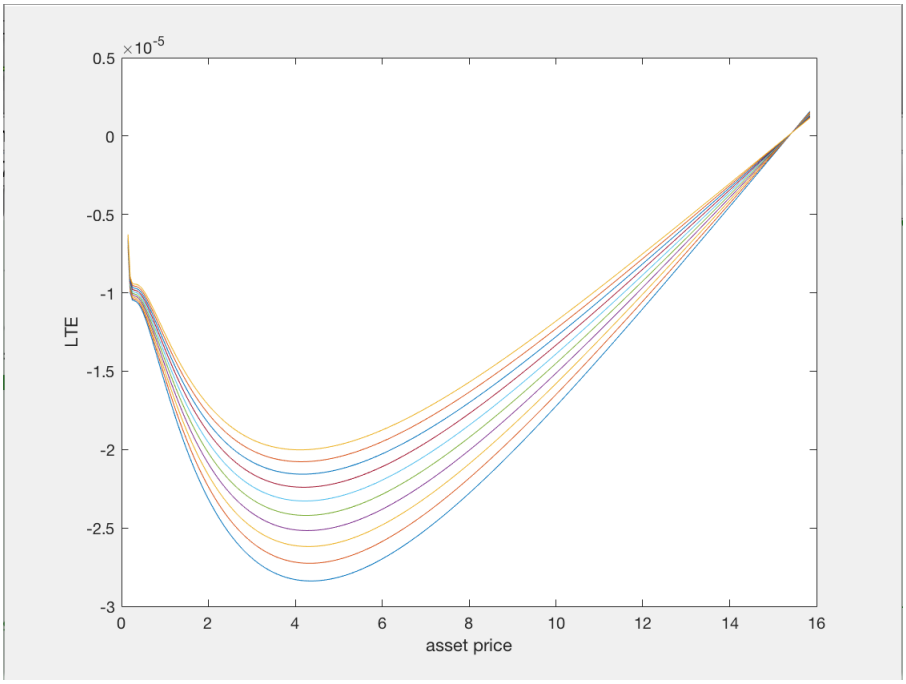

Figure 4.4: LTE from the direct approach.

Chapter 5

Conclusion

The heat equation is a very familiar model with a considerable amount of theory on its properties and solutions available to us. However, just because a method works does not mean that it is the best method. Our hypothesis that the heat equation sacrifices accuracy through the process of transformation is confirmed by the error analysis. Having a more evenly distributed error, the direct approach to solving the Black-Scholes equation gives a much more accurate and robust scheme.

There are still a lot that can be done for this project. In the future, we hope to start experimenting our scheme with varying volatility and interest and analyze its behavior. We hope to investigate further into the cause of the discontinuity in the heat equation scheme. There are also many other higher-order finite difference methods we can experiment with to directly solve the Black-Scholes model that might improve the accuracy and efficiency of our current scheme.

Appendix A

Code

A.1. Heat scheme (`crankheatdirecttrans.m`)

```
function [b,U] = crankheatdirecttrans(S,E,dt,ti,T,r,sigma)
% using crank-nicolson to solve the transformed heat equation ...
    with direct back-slash
% then heat variables are transformed back into financial ...
    variables and are plotted
% S = vector of asset value
% E = exercise price
% dt = change in time for black scholes
% T = end time
% ti = initial time
% r = interest rate
% sigma = volatility% t = vector of all time values in variable t
t = (T:-dt:ti)'; % we stepped backwards in time for black-scholes
tstep = round((T-ti)/dt); % total number of time steps
x = log(S/E); % this is a vector
N = length(x);  % total # of points
num = N-2 ; % # of interior points
dx = (x(end) - x(1))/(num+1);
x = x(1):dx:x(end); % make x equally spaced
tau = zeros(length(t),1);
for i = 1:length(t)
    tau(i) = (T - t(i))*(1/2)*sigma^2;
    % tau is now a vector of all transformed time value
end
dtau = (tau(end)-tau(1))/tstep; % new dtau transformed for tau
a = dtau/(dx*dx);
atwo = a/2;
k = r/(0.5*sigma^2);
u = zeros(N,1);
% initial condition at (time)tau = 0
for i = 1:N
    taui = 0;
    u(i,1) = payoffcall(x(i),taui,k);
end
U = zeros(N,tstep+1);
V = zeros(N,tstep+1);
% (rows,columns)
% prelocate space needed
% columns corresponds to each time step
% rows corresponds to each N
```

```
U(:,1) = u; %inital condition
v = exp( (-1/2)*(k-1)*x - (1/4)*(k+1)^2*tau(1) ).*u'; %transform ...
    value back to V
C = E*v;
V(:,1)=C; % store value in V
b = zeros(N,1);
f1 = figure('Name','heat values');
f2 = figure('Name','transformed-back values');
for i = 2:tstep+1 % step forward in time (column)
    % uptdate end points
    for n = 2 : N-1 % (row) change only the inside points of b
        b(n,1) = (1-a)*u(n,1)+ atwo*(u(n+1,1)+u(n-1,1));
    end
    u(1) = uminfcall(x(1),tau(i),k);
    u(end) = upinfcall(x(end), tau(i),k);
    b(2,1) = b(2,1)+atwo*u(1);
    b(N-1,1) = b(N-1,1)+atwo*u(end);
    e = ones(N,1);
    A = a * spdiags ([e -2*e e], [-1, 0, 1], N-2, N-2);
    I = speye(N-2,N-2);
    u(2:end-1,1)= (I - (1/2)*A) \ b(2:N-1);
    U(:,i) = u; % store results in corresponding time-step column
    v = exp( (-1/2)*(k-1)*x - (1/4)*(k+1)^2*tau(i) ).*u'; % ...
        transform value back to v
    C = E*v; % transform value back to the value of the option
    V(:,i)=C; % store it in the matrix
    S = E*exp(x); %change space values back to in terms of S ...
        based on the equally spaced x-vector
    %since x at this point is equally spaced, S is not because it ...
        is now exponentially spaced
    hold on
    figure(f1);
    plot(x,u)
    hold on
    figure(f2);
    plot(S,C)
    pause(0.1)
end
hold off

LTE=zeros(N-1,tstep-3);% prelocate space needed, with 2 layers of ...
    the time edge trimmed and 1 side of the space edge trimmed)
% columns corresponds to each time step
% rows corresponds to each N space step
f3 =figure('Name','heat LTE');
figure(f3);
for m = tstep-13:1:tstep-3
% we are really only interested in the last ten steps the scheme ...
    takes since they are supposed to be the most accurate
% since as more time steps are taken, results converge more to ...
    the correct values
    for n = 1:N-1
        % we have to leave the last N out because it cannot be a ...
            center, only a stenil
```

```
85          % as there is no N+1 that can give it a ds value
86          ds=S(n+1)-S(n);
87          % since our S vector is expoenential scaled from an ...
                equally spaced x vector, it is not equally spaced anymore
88          % and ds is not constant but changes with each spacial step
89          % thus, unlike the LTE for B-S, we have a changing ds ...
                based on the space step we are on
90          LTE(n,m)=heatLTE(V,dt,ds,n,m+2,S,r,sigma,E,T,k,t);
91      end
92      semilogy(S(2:N),abs(LTE(:,m)));
93      % because plotting it on a linear scale creates a huge spike ...
            in the middle
94      % makeing the changes in the smaller values difficult to observe
95      % we change it to a log scale so bigger values are brought ...
            closer to the smaller values
96      % and we can observe the change in the smaller value regions ...
            better
97      % this makes the y-axis logarithmically scaled with a base 10
98      % since you can't log(-#) we have to put an absolute value ...
            around our y-values
99      pause(0.1);
100     hold on
101 end
102 hold off
```

`crankheatdirecttrans.m` is used to generate and plot the solutions and error from the Heat equation approach.

A.2. Heat payoff (`payoffcall.m`)

```
1 function g = payoffcall(x,tau,k)
2 g = exp(0.25*((k+1)^2)*tau) * ...
      max(exp(0.5*(k+1)*x)-exp(0.5*(k-1)*x),0);
3 return
```

`payoffcall.m` is used to generate the initial condition for the heat equation.

A.3. Heat lower boundary (`uminfcall.m`)

```
1 function um = uminfcall(x,tau,k)
2 um = 0;
3 return
```

`uminfcall.m` is used to generate the lower boundary for the heat equation.

A.4. Heat upper boundary (`upinfcall.m`)

```
function up = upinfcall(x,tau,k)
up = exp(0.25*((k+1)^2)*tau ) * (exp(0.5*(k+1)*x) - ...
    exp(0.5*(k-1)*x)*exp(-k*tau));
return
```

`upinfcall.m` is used to generate the upper boundary for the heat equation.

A.5. Heat LTE (`heatLTE.m`)

```
function er=heatLTE(V,dt,ds,n,m,S,r,sigma,E,T,k,t)
Vt=(V(n,m+1)-V(n,m-1))/(2*dt);
Vtt=(V(n,m+1)-2*V(n,m)+V(n,m-1))/(dt)^2;
Vttt=(V(n,m+2)-2*(V(n,m+1)-V(n,m-1))-V(n,m-2))/(2*(dt)^3);
eq=(S(n))^(r/sigma^2-1/2)*E^(1/2-r/sigma^2)*...
    exp((2*r+sigma)^2/(8*sigma^2)*(T-t(m)));
er=1/S(n)^2*ds^2*(1/12)*(V(n,m)*1/(16*E)*(k+1)^4*eq-Vt*...
    1/(sigma^2*E)*(k+1)^2*eq+Vtt*4/(sigma^4*E)*eq)+1/4*sigma^4*...
    dt^2*1/4*(V(n,m)*1/(64*E)*(k+1)^6*eq-Vt*3/(8*sigma^2*E)*...
    (k+1)^4*eq+Vtt*3/(sigma^4*E)*(k+1)^2*eq-Vttt*8/(sigma^6*E)*eq);
return
```

`heatLTE.m` is used to calculate the local truncation error generated by each iteration of solving the heat equation.

A.6. Direct scheme (`crankblackh.m`)

```
function V = crankblackh(dt,ti,T,r,sigma,S,E,theta)
% dt = change in time
% ti = initial time
% T = end time
% r = interest
% sigma = volatility
% S = vector of values of asset varying spatially
% E = exersice price fixed
% V = value of the option, depends on S and t
N = length(S); % total number of points in the S vector
num = N-2; % num = # of interior points
ds = (S(end) - S(1)) / (num+1); % ds is constant as our S vector ...
    is equally spaced
alpha = dt/(ds*ds);
```

```
tstep = round((T-ti)/dt); % total number of time steps
t = ti:dt:T; % vector of all time steps
Vmplus = zeros(N,1); % prelocate space
% value of the option at the end of time = T
% initial contition
for i = 1:N
    Vmplus(i) = max(S(i)-E,0); % spatially varying
end
Vmplus(1) = 0; % lower boundary condition
Vmplus(end) = S(end) - E*exp(-r*(T-T)); % upper boundary condition
V = zeros(N,tstep+1); % prelocate space needed
% columns corresponds to each time step
% rows corresponds to each N space step
f1 = figure('Name','against S');
f2 = figure('Name','against t');
figure(f1);
for i = tstep+1:-1:2 % we fill-in the matrix backwards because we ...
   are stepping backwards in time
    V(:,i) = Vmplus; % fill-in the column (that time-step) with ...
       the last computed Vmplus value
    Zmplus = explicith(N,dt,r,alpha,sigma,S,ds,Vmplus,theta);
    Vm = implicith(N,dt,r,alpha,sigma,S,ds,Zmplus,Vmplus,1-theta);
    Vmplus(2:N-1) = Vm; % update Vmplus vector values, leaving ...
       boundary conditions
    Vmplus(1) = 0; % boundary condition
    Vmplus(end) = S(end) - E*exp(-r*(T-t(i))); % boundary condition
    plot(S,Vmplus);
    pause(0.1);
    hold on;
end
hold off;
V(:,1) = Vmplus; % fill in the last column in time-stepping ...
   (first column in matrix) with the last Vmplus
figure(f2);
for n = 1:N
    row = V(n,:);
    plot(t,row);
    pause(0.1);
    hold on;
end
hold off


% caculate LTE for this method
f3 = figure('Name','B-S LTE');
LTE = zeros(N-4,tstep-3); % prelocate space needed, with 2 layers ...
   of the edge trimmed to allow space for stencils
% columns corresponds to each time step
% rows corresponds to each N space step
figure(f3);
for m=10:-1:1
    % we are really only interested in the last ten steps the ...
       scheme takes since they are supposed to be the most accurate
```

```
61      % in this scheme, since we are stepping backwards in time, ...
            our last ten steps were the first ten time points
62      for n = 1:(N-4)
63          % corresponding to S(3:N-2)
64          % we leave two spaces at the begining and end to allow ...
                for stencil points around the point of focus in the center
65          LTE(n,m)=BSLTE(V,n+2,m+2,dt,ds,S,r);
66      end
67      plot(S(3:N-3),LTE(1:n-1,m));
68      % we trimmed out the last point in our vector because it ...
            spikes up
69      % and makes the changes in the smaller values un-observable
70      pause(0.1);
71      hold on
72  end
73  hold off
```

`crankblackh.m` is is used to generate and plot the solutions and error from the direct approach.

A.7. Explicit solve (`explicith.m`)

```
1   function Zmplus = explicith(N,dt,r,alpha,sigma,S,ds,Vmplus,h)
2   mainA = zeros(N-2,1);
3   superA = zeros(N-2,1);
4   subA = zeros(N-2,1);
5   for i = 1:(N-2)
6       mainA(i) = 1 - dt*r*h - alpha*sigma^2*S(i+1)^2*h;
7       subA(i) = -1/2*dt/ds*r*S(i+2)*h + 1/2*alpha*sigma^2*S(i+2)^2*h;
8   end
9   for i = 2:(N-2)
10      superA(i) = 1/2*dt/ds*r*S(i)*h + 1/2*alpha*sigma^2*S(i)^2*h;
11  end
12  A =spdiags ([subA mainA superA], [-1, 0, 1], N-2, N-2);
13  % coefficient matrix, boundary terms excluded
14  % Vmplus = ones(N,1) * max(max(S)-E,0);
15  % spdiags takes elements of super-diagonals from the lower part ...
        of the specified vector
16  % takes elements of the sub-diag from the upper part of the ...
        specified vector
17  Zmplus = A*Vmplus(2:N-1); %boundary terms excluded
18  % add excluded boundary terms
19  Zmplus(1) = Zmplus(1) + (-1/2*dt/ds*r*S(2)*h + ...
        1/2*alpha*sigma^2*S(2)^2*h)*Vmplus(1);
20  Zmplus(end) = Zmplus(end) + (1/2*dt/ds*r*S(end-1)*h + ...
        1/2*alpha*sigma^2*S(end-1)^2*h)*Vmplus(end);
```

`explicith.m` is used to calculate right-hand side of the (3.18).

A.8. Implicit solve (`implicith.m`)

```
function Vm = implicith(N,dt,r,alpha,sigma,S,ds,Zmplus,Vmplus,h)
mainB = zeros(N-2,1);
superB = zeros(N-3,1);
subB = zeros(N-3,1);
for i = 1:(N-2)
    mainB(i) = 1 + dt*r*h + alpha*sigma^2*S(i+1)^2*h;
    subB(i) = 1/2*dt/ds*r*S(i+2)*h - 1/2*alpha*sigma^2*S(i+2)^2*h;
end
for i = 2:(N-2)
    superB(i) = -1/2*dt/ds*r*S(i)*h - 1/2*alpha*sigma^2*S(i)^2*h;
end
B =spdiags ([subB mainB superB], [-1, 0, 1], N-2, N-2); % ...
    coefficient matrix boundary terms excluded
% B*V = Z
Zmplus(1) = Zmplus(1) - (1/2*dt/ds*r*S(2)*h - ...
    1/2*alpha*sigma^2*S(2)^2*h)*Vmplus(1);
Zmplus(end) = Zmplus(end) - (-1/2*dt/ds*r*S(end-1)*h - ...
    1/2*alpha*sigma^2*S(end-1)^2*h)*Vmplus(end);
% move known boundary terms to Zmplus's side
Vm = B\Zmplus;
% Vm = inv(B)*Zmplus
```

`implicith.m` is used to implicitly solve the left-hand side of (3.18).

A.9. Direct LTE (`BSLTE.m`)

```
function e=BSLTE(V,n,m,dt,ds,S,r)
Vtt=(V(n,m+1)-2*V(n,m)+V(n,m-1))/(dt)^2;
Vttt=(V(n,m+2)-2*(V(n,m+1)-V(n,m-1))-V(n,m-2))/(2*(dt)^3);
Vsss=(V(n+2,m)-2*(V(n+1,m)-V(n-1,m))-V(n-2,m))/(2*(ds)^3);
Vstt=(V(n+1,m+1)-V(n-1,m+1)-2*(V(n+1,m)-V(n-1,m))+V(n+1,m-1) - ...
    V(n-1,m-1))/(2*ds*dt^2);
e=(dt)^2*((-1/4)*r*Vtt+(1/4)*r*S(n)*Vstt+(1/6)*Vttt) + ...
    (ds)^2*((1/6)*r*S(n)*Vsss)+(dt)^3*((-1/12)*r*Vttt);
```

`BSLTE.m` is used to calculate the local truncation error generated by each iteration of solving the Black-Scholes equation directly.

A.10. Parameters (`EUdefine_variables.m`)

```
S = 0.05:0.05:16;
% S = value of asset varying spatially
E = 10;
% E = exersice price fixed
dt = 0.01;
% dt = change in time
ti = 0;
% ti = initial time
T = 1;
% T = end time
r = 0.1;
% r = interest
sigma = 3;
% sigma = volatility
theta = 0;
% control

crankheatdirecttrans(S,E,dt,ti,T,r,sigma);
crankblackh(dt,ti,T,r,sigma,S,E,theta);
```

`EUdefine_variables.m` is used to input parameters and compare the solutions from the two approaches side by side.

Bibliography

[1] John Hull, *Introduction to Futures and Options Markets.*, Prentice-Hall Of India Pv, 2009.

[2] Burton Malkiel, *The Efficient Market Hypothesis and Its Critics*, Journal of Economic Perspectives **17** (2003), 59-82, DOI 10.1257/089533003321164958.

[3] Donald MacKenzie, *An Engine, Not a Camera: How Financial Models Shape Markets*, MIT Press, Cambridge, MA, 2008.

[4] Robert Jarrow, *In Honor of the Nobel Laureates Robert C. Merton and Myron S. Scholes: A Partial Differential Equation That Changed the World*, Journal of Economic Perspectives **13** (1999), 229-248, DOI 10.1257/jep.13.4.229.

[5] Albert Wang, *Generalized Ito's Formula and Additive Functionals of Brownian Motion*, Probability Theory and Stochastic Processes **41** (1977), 153-159, DOI https://doi.org/10.1007/BF00538419.

[6] D. V. Widder, *The Heat Equation*, Academic Press, Cambridge, MA, 1993.

[7] Paul Wilmott, Sam Howison, and Jeff Dewynne, *The Mathematics of Financial Derivatives: a Student Introduction*, Cambridge University Press, New York, NY, 2010.

[8] Iserles Arieh, *A First Course in the Numerical Analysis of Differential Equations*, Cambridge University Press, New York, NY, 2009.

[9] James Lambers and Amber Sumner, *Explorations in Numerical Analysis*, World Scientific Publishing, 2019.