Distribution Agreement

Dylan Connor April 4th, 2012

Numerical Approximation of the Black-Scholes Equations: A Practical Experience

by

Dylan Connor

Alessandro Veneziani
Adviser

Applied Mathematics

Alessandro Veneziani

Adviser

Emily Hamilton

Committee Member

Clifton Green

Committee Member

2012

Numerical Approximation of the Black-Scholes Equations: A Practical Experience

By

Dylan Connor

Alessandro Veneziani

Adviser

An abstract of
a thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Business Administration with Honors

Applied Mathematics

2012

Abstract

Numerical Approximation of the Black-Scholes Equations: A Practical Experience
By Dylan Connor

Black and Scholes equations for pricing of derivatives are an interesting and up-to-date topic of research, where both backgrounds in math and finance are fundamentals. In this work we aim at experiencing the mathematical approach and the numerical approximation of this differential problem. We will assume these equations to be a reliable model and will work on their numerical approximation with a mixed finite elements(FE)/finite differences (FD) approach. More precisely, we will use the mathematically well-established Finite Element Method (FEM) for the underlying price dependence, while time dependence will be discretized with the Finite Difference Method (FDM). The ultimate purpose of the work is to earn sensitivity when using numerical tools in financial mathematics, with particular attention to the critical analysis of the results. This includes an extensive error analysis and appropriate visualization of the results. This will allow us to comment on the practicality of numerical mathematics used to solve the Black-Scholes equation.

Numerical Approximation of the Black-Scholes Equations: A Practical Experience

By

Dylan Connor

Alessandro Veneziani

Adviser

A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Business Administration with Honors

Applied Mathematics

2012

Table of Contents

Table of Figures

Chapter 1

Introduction

1.1 Purpose of the Present Work

Black and Scholes equations for pricing of derivatives are an interesting and up-to-date topic of research, where both backgrounds in math and finance are fundamentals. This subject matter has been heavily data mined in recent years. In this work we aim at experiencing the mathematical approach and the numerical approximation of this differential problem. We do not intend to question the assumptions behind the Black-Scholes model in order to bring insight on the validity of the existent partial differential equation. Instead, we assume these equations to be a reliable model and we will work on their numerical approximation with a mixed finite elements(FE)/finite differences (FD) approach. More precisely, we will use the mathematically well established Finite Element Method (FEM) for the underlying price dependence, while time dependence will be discretized with the Finite Difference Method (FDM). The ultimate purpose of the work is to earn sensitivity when using numerical tools in financial mathematics, with particular attention to the critical analysis of the results. This includes an extensive error analysis and appropriate visualization of the results. This will allow us to comment on the practicality of numerical mathematics used to solve the Black-Scholes equation.

1.2 Theoretical and Practical Application

The Black & Scholes problem consists of a differential equation depending on $d+1$ independent variables, where d is the number of independent underlying and the additional dependence is on time t. For an extensive introduction to the problem, we refer to [9], Chap 2.9. The equation is of second order and needs to be completed by suitable initial and boundary conditions. We will extensively discuss the model in the next chapter.

Apart from special cases that we will discuss later on, the analytical solution of the problem cannot be computed. The best we can do is to prove that there exists a unique solution, depending continuously on the data. This is called *well posedness* of the problem according to Hadamard (see e.g. [3, 9]). The proof requires using concepts of functional analysis and in particular a result called *Lax-Milgram Lemma*, which are out of the scope of the present work. The relevant point, here, is that in general we cannot solve explicitly the equations and this prevents a quantitative analysis. Fortunately, it is possible to approximate the differential equation with a system of algebraic equations, see e.g. [5, 2]. This can be accomplished in several ways and virtually for all the situations of interest (e.g. European, American or Asiatic options). In this way, the difficulties intrinsic to the differential problem are downscaled to a large system of algebraic (and in our case linear) equations, that can be easily managed by numerical software (see e.g. [6]).

Among the other methods, the FEM belongs to the class of Galerkin approaches. They rely upon the idea of approximating the solution with a piecewise polynomial interpolating function with "optimal" approximating properties. The theory of FEM is well developed and analyzed in several books (see e.g. [8]). In particular, the versatility of the method in switching from one to more independent variables makes them the choice for several applications. Nevertheless, in some contexts, like for time discretization, an easier FD approach is often preferred, which promptly descends from the classical theory of numerical differentiation [6].

The conversion of differential to algebraic problems is clearly not for free. It introduces an unavoidable numerical error that needs to be assessed. This error will depend specifically on the problem at hand, the different options and the discretization parameters (e.g. the number of nodes used for the construction of the piecewise polynomial functions or of the time steps used for getting to the final time of the computation).

In this work we will specifically consider the numerical simulation of European options, where an analytical solution is actually available. The extensive comparison of the numerical solution with the analytical one to assess the error dynamics and control is the core of the present work. This error analysis could then be the foundation for more interpreting more complex models where there is no analytic solution.

1.2.1 Software

In this paper we will use basically two codes:

1. `Fem1D` is a simple finite element software working in Matlab, developed with a graphical user interface (GUI) by F. Saleri and A. Veneziani. The code is equipped with proper methods for the error computation and it is for this reason particularly suitable for our error analysis.

2. `FreeFem++` is a C++ based software for the easy implementation of the FEM in 2D problems, developed by F. Hecht and O. Pironneau [1].

1.3 Error Analysis

The process of crafting numerical results from the given financial problem is quite long. Given the starting point of the problem at hand we must convert this into a mathematical model. In this process modelling error will arise. Next the mathematical model must be numerically approximated. In this step of forming an algebraic representation, discretization errors are introduced. Finally, this numerical approximation needs to be solved in some way. The step from approximation to results is affected by rounding errors. See 1.1. With this in mind error analysis is a key component of any numerical method. In this paper we will focus on discetization error. In considering discretization error there are two main components. Space discretization represented by h. In this problem it is actually price of the underlying rather than space discretization. However, "space discretization" will be used in this paper to describe the process rather than "price discretization". Time discretization is represented by Δt. The smaller each of these parameters become, the closer the algebraic representation will approximate the analytic problem. That is, as $h \to 0$ space error should be under control given the theory of FEM and as $\Delta t \to 0$ the time error should be under control given the FDM theory. Additionally, we introduce additional error due to the truncation of an infinite domain. When the axis $x \epsilon \mathbb{R}$ is truncated in $[-L, L]$, we introduce an error and the parameter L qualifies the entirety of the error. That is, the larger we make L the smaller this error becomes. Since we have an exact analytic solution to the 1D problem, this will be a strong starting point for the implications and sensitivity of each of these parameters and their corresponding error.

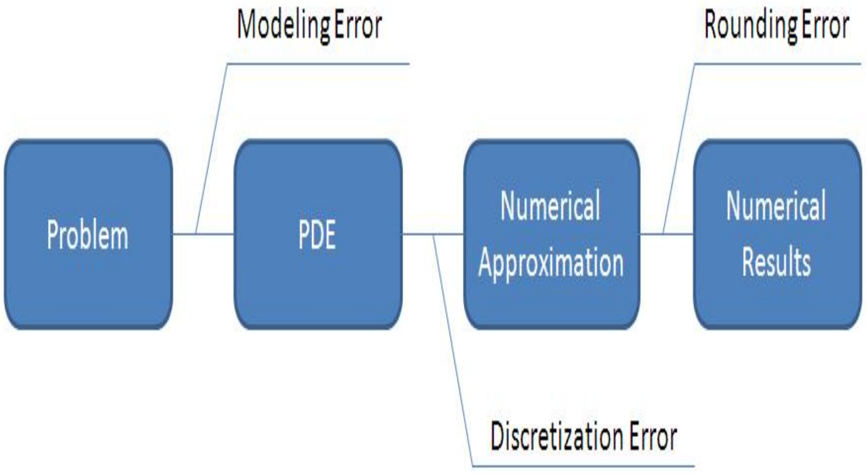


Figure 1.1: Discretization Error Schematic.

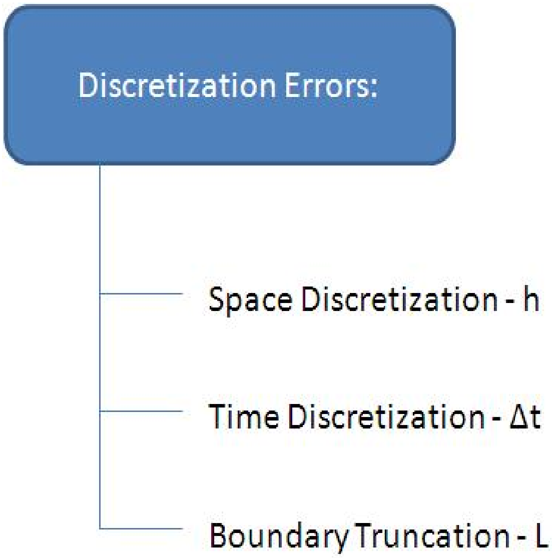


Figure 1.2: Discretization Error Break Down.

1.4 Data Visualization

Error analysis will be a key goal of this paper, but data visualization is also important. Data visualization is an important element in understanding the items at hand as well as bringing meaningful conclusions to a broader audience. This is especially true when the heart of the information lies in 50,000 plus entry matrix; it is extremely hard to look at a large matrix and intuitively understand that entries are right or wrong. The post processing of the solution in both FEM1D and FreeFEM++ provide visual synthesis of the data output. However, since the original B & S problem requires manipulation in order to pose the "physical" (financial) space to the mathematical (solvable) space, visualization will require some additional steps to unwind this and present the solution back in the intuitive financial space. The necessity of this data manipulation will be discussed in more detail later. In 1D this requires some additional steps in FEM1D; in 2D this process is done by FreeFem++. Also, we aim to visualize the error analysis so large areas of error will stand out. Visualization could be a key element in interpreting the data in dimensions higher than 1D. By doing an error analysis and data visualization in 1D, we will consider the main elements of error that exist within this scenario and hypothesize that similar anomalies may stand out when moving to two dimensions and more.

Chapter 2

Understanding Black-Scholes

2.1 Basic Terminology

As the purpose of this paper is not entirely related to the derivation or assumptions necessary for Black-Scholes we will shortly present the fundamental components that came together to spawn this equation. Black-Scholes equation was derived back in the early 70's by Fischer Black and Myron Scholes [12]. The equation solves the value of a stock option. It is a partial differential equation that relies on simplifying assumptions of financial structures and the mathematical consequences of a perfectly hedged portfolio strategy. Let us quickly recap the terminology necessary for a basic understanding of Black-Scholes and what will be used in this paper.

An *option* is a derivative that one can buy (sell); it gives the buyer the choice to buy (sell) a stock for a specific price, at a future date. The option to buy the stock is called a *call option*, where the option to sell is called a *put option.* The price that the underlying can be bought at is called the *exercise* or *strike* price and the date by which the transaction must occur is called the *date of maturity*. In addition, one can buy or sell (short) each option. In the case of selling, one would be paid to give someone the option to buy or sell from them at a future date at the exercise price. The payoff diagrams below help in understanding each option. In addition, there are *American options* and *European options*. American options can be exercised any time up until the date of maturity where European options can only be exercised on the date of maturity. The value of an option, V, is a combination of the options *intrinsic value* and its *time value*. Intrinsic value is how much an option is in-the-money, that is, $max(S - E, 0)$ or $max(E - S, 0)$ depending on if one has a call or put option. The time value relates to the likelihood that the option will become in-the-money and by how much by the time of expiration.

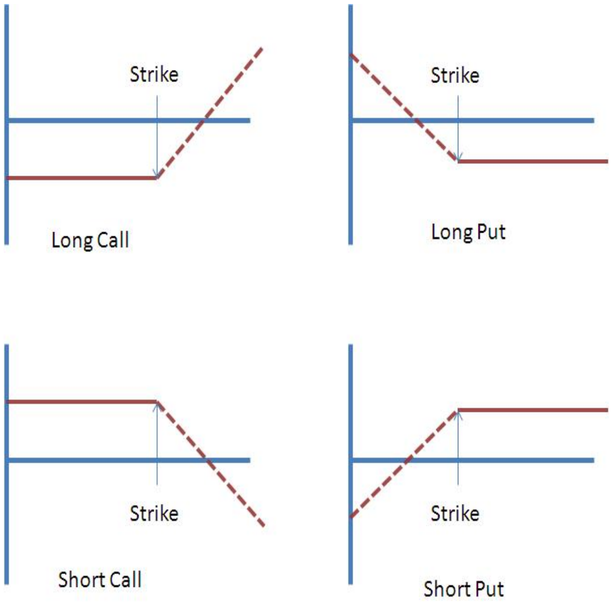


Figure 2.1: Pay-off diagram.

2.2 The 1D and 2D Black-Scholes Equations

The partial differential equation known as the Black-Scholes equation reads:

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0 \tag{2.1}$$

Here V is a function of S and t, and is the value of the derivative option. S is the price of the stock and t is time. Here σ is the volatility, that is, a measure of how much the stock is changing and how fast. The variable r is the risk free rate, that is, what return on investment one should expect without bearing any risk. Similarly, the two dimension Black-Scholes equation reads:

$$\frac{\partial V}{\partial t} + \frac{(\sigma_1 S_1)^2}{2}\frac{\partial^2 V}{\partial S_1^2} + \frac{(\sigma_2 S_2)^2}{2}\frac{\partial^2 V}{\partial S_2^2} + \rho S_1 S_2 \frac{\partial^2 V}{\partial S_1 \partial S_2} + rS_1\frac{\partial V}{\partial S_1} + rS_2\frac{\partial V}{\partial S_2} - rV = 0 \tag{2.2}$$

This equation refers [7] to two stocks with the same exercise price. Here ρ is the correlation coefficient and is an indicator of how much the two stocks move in relation to each other. For example, if the correlation is one, the two stocks should both grow together or decline together equivalently. If the correlation is negative one then the two stocks should act in reverse of one another with respect to growth.

2.3 Building the Model: The Lognormal Density

The following assumptions and their mathematical interpretations are key in understanding the fact that S follows the lognormal law. The first assumption is that all new information is instantaneously reacted to by the market. This relates to some kind of market efficiency, closest to semi-strong form efficiency, where all public information is included in the share price. This instantaneous reaction is necessary to interpret the problem as continuous. The second assumption is that price is memoryless. In this sense, the current price holds all information inherently in that number. This assumption ultimately frames the problem such that changes in S, the underlying, can be interpreted as a random walk, Brownian motion with drift. Brownian motion is conceptually derived from the process of random particles drifting into each other in a liquid, but the model has vast applications in math, economics, and physics [13]. Thus S can be represented as a combination of its drift over time, denoted by μ, and its Brownian motion. This leads to the equation:

$$\frac{dS}{S} = \mu dt + \sigma dB \tag{2.3}$$

Since this is a stochastic problem, regular (deterministic) rules of differentiation do not apply. Therefore we must consider Itô's formula to solve. The derivation is explained in [9]. The equation is as follows. In an Itô's diffusion problem we have $X = X(t, B)$ with $B = B(t)$. While B is simply a function of t, the properties present within the Brownian motion present a probability density component that must be treated differently, thus we cannot consider simply $X = X(t)$. We consider:

$$dX = a(X, t)dt + \sigma(X, t)dB \tag{2.4}$$

Then for any twice differentiable $F = F(x, t)$, Itô's differentiation yields

$$dF = (F_t + aF_x + \frac{1}{2}\sigma^2 F_{xx})dt + \sigma F_x dB \tag{2.5}$$

By taking the above equations and with some manipulation we ultimately get the equation:

$$log(S(t)) = \log(S_0) + (\mu - \frac{1}{2}\sigma^2)t + \sigma B(t) \tag{2.6}$$

This is then used to show that S has a lognormal density, that is, since S is a stochastic variable, it is associated to a probability density function. A random variable $Y = log(S)$ has a normal distribution with mean $log(S_0) + (\mu - \frac{1}{2}\sigma^2)t$, variance $\sigma^2 t$, and a probability density of:

$$f(y) = \frac{1}{\sqrt{2\pi\sigma^2 t}} exp \left\{ -\frac{(y - log(S_0) - (\mu - \frac{1}{2}\sigma^2)t)^2}{2\sigma^2 t} \right\} \tag{2.7}$$

and S has the lognormal density:

$$p(s) = \frac{1}{s} f(log(s)) = \frac{1}{s\sqrt{2\pi\sigma^2 t}} exp \left\{ -\frac{(log(s) - log(S_0) - (\mu - \frac{1}{2}\sigma^2)t)^2}{2\sigma^2 t} \right\} \tag{2.8}$$

2.4 Building the Model: The Equation

Let us consider further assumptions in order to build (2.1). We must assume that

1. volatility is constant and known

2. there are no transaction costs or dividends
3. there is full access to any amount of the underlying asset
4. a riskless interest rate exists and is known
5. the market is arbitrage free

Most of these assumptions are necessary mostly for reducing noise that would be hard to capture in a partial differential equation. However, the final assumption will be a key component in building the equation. Arbitrage free means that there is no opportunity for instantaneous riskless profit. While the other assumptions help by eliminating possible noise, this assumption can actually be quantified by understanding the math behind a hedged portfolio. The idea is to find the value of the option V at a given time; since we know the intrinsic value of V we want to find the time value of V in order to appropriately price the option. We start by considering Itô's lemma. We can construct a portfolio that will eliminate the risk through hedging. Thus this riskless portfolio investment should grow at the risk free rate which we can compare to the value solved by Itô's lemma. By starting with Itô's lemma we see:

$$dV = (V_t + \mu S V_S + \frac{1}{2}\sigma^2 S^2 V_{SS})dt + \sigma S V_S dB \tag{2.9}$$

The second term, that is $\sigma S V_s dB$, is the term that holds all the risk as it is the portion representing the random walk. Thus we want to eliminate this with a portfolio. Thus, we want:

$$\Pi = V - SU \tag{2.10}$$

Where $-U$ is the quantity we want of the underlying. Therefore changes in Π can be modelled as:

$$d\Pi = dV - UdS \tag{2.11}$$

By substituting and rearanging considering (2.3) and (2.9) we see:

$$d\Pi = \left[V_t + \mu S V_S + \frac{1}{2}\sigma^2 S^2 V_{SS} - \mu S U\right] dt + \sigma S(V_S - U)dB \tag{2.12}$$

It can be shown that by choosing $U = V_s$, that is, we take U to be equal to V_s at t, we can get rid of the stochastic portion. In doing so we are left with the riskless equation:

$$d\Pi = \left(V_t + \frac{1}{2}\sigma^2 S^2 V_{SS}\right) dt \tag{2.13}$$

Now let us compare this with:

$$d\Pi = r\Pi dt \tag{2.14}$$

By doing so the arbitrage free assumption ultimately holds:

$$d\Pi = (V_t + \frac{1}{2}\sigma^2 S^2 V_{SS})dt = r\Pi dt \tag{2.15}$$

Using a substitution, with consideration to:

$$\Pi = V - SU = V - V_S S \tag{2.16}$$

we arrive at the Black-Scholes equation:

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0 \tag{2.17}$$

2.5 Building the Model: The Boundary and "Final" Conditions

Boundary and initial conditions are a key component of solving a PDE. With B&S we have a final condition rather than an initial condition. The general form of a PDE of this order, for example the heat equation, has a negative diffusion term with an initial condition. Since the B&S equation has a positive space derivative term and a final condition it is still well posed as we will be able to use a variable change to alter this.

Boundary conditions and the final condition are going to be dependent upon whether solving a put option or a call option. For a put, when the stock price is zero, the put is worth the present value of the exercise price. As S approaches infinity the holder would not exercise the option and thus it has a value of zero. At the date of maturity, time T, the option has no time value, so the value should just be its intrinsic value; that is the value of the put should be the exercise price minus the stock price, or zero if this is negative. Therefore we have:

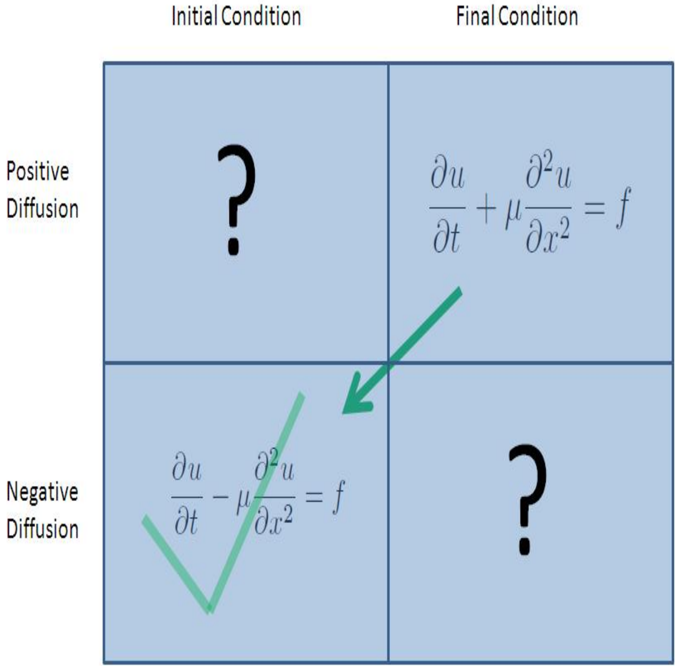

Initial Condition
Final Condition
Positive Diffusion
Negative Diffusion
?
$\frac{\partial u}{\partial t} + \mu \frac{\partial^2 u}{\partial x^2} = f$
$\frac{\partial u}{\partial t} - \mu \frac{\partial^2 u}{\partial x^2} = f$
?

Figure 2.2: Well Posed Conditions

$$\begin{cases} P(0,t) = Ee^{-r(T-t)} \\ P(S_1,t) = 0 \quad S_1 \to +\infty \\ P(S,T) = \max(E-S,0) \end{cases} \tag{2.18}$$

Similarly, with a call option we know that if the stock price is zero, then it will not be exercised and is very unlikely to become in the money so it has a value of zero. As S approaches infinity, the value of the call should approach infinity. At the final time T, the call should have the value of the stock price minus the exercise price, or zero if this is negative. Thus we have:

$$\begin{cases} C(0,t) = 0 \\ C(S_1,t) - (S - Ee^{-r(T-t)}) \to 0 \quad S_1 \to 0 \\ C(S,T) = \max(S-E,0) \end{cases} \tag{2.19}$$

For the reasons stated above, we must now use a variable change to make the problem well posed. In order to solve with a change of variable we take:

$$\begin{cases} x = \log \frac{S}{E} \\ \tau = \frac{1}{2}\sigma^2(T-t) \\ w(x,\tau) = \frac{V}{E}\left[(Ee^x, T - \frac{2\tau}{\sigma^2}\right] \end{cases} \tag{2.20}$$

When we now equate the V time and space derivatives to the w time and space derivative we have:

$$\begin{cases} V_t = -\frac{1}{2}\sigma^2 E w_\tau \\ V_S = \frac{E}{S} w_x \\ V_{SS} = -\frac{E}{S^2} w_x + \frac{S^2}{w_{xx}} \end{cases} \tag{2.21}$$

By substituting these into the Black-Scholes equation stated above and some mathematical rearranging and simplifying we find that:

$$w_\tau = w_{xx} + (k-1)w_x - kw \tag{2.22}$$

where $k = \frac{2r}{\sigma^2}$. We can then set:

$$w(x,\tau) = e^{-\frac{k-1}{2}x-\frac{1}{4}(k+1)^2\tau} v(x,\tau) \tag{2.23}$$

where it can be shown that

$$v(x,\tau) = e^{\frac{1}{2}(k+1)x-\frac{1}{4}(k+1)^2\tau} N(d_+) - e^{\frac{1}{2}(k-1)x+\frac{1}{4}(k-1)^2\tau} N(d_-) \tag{2.24}$$

with

$$N(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-\dfrac{1}{2}y^2}\, dy \tag{2.25}$$

and

$$d_\pm = \frac{x}{\sqrt{2\tau}} + \frac{1}{2}(k \pm 1)\sqrt{2\tau} \tag{2.26}$$

The well posed problem relating to w is what we will solve with FEM1D.

Chapter 3

Numerical Discretization

3.1 Overview

While FEM and FDM are quite complex methods and a complete introduction is beyond the purpose of this work, this paper will focus on a broad understanding as well as the strengths and weaknesses of the technique. FEM and FDM are numerical methods used to solve partial differential equations via space and time discretization. The finite difference method entails approximating a derivative and using a *collocation* method to produce a solution on the specified nodes. Interpolation is necessary to solve the value in between nodes. On the other hand the finite element method looks at an approximate set, or more precisely, functional space where we look for a solution. Where the posed problem has solution $u \in V$, FEM considers $u_h \in V_h$ where $V_h \subseteq V$. u_h is not in general the closest function of V_h to the solution u but it averages with the same velocity. This is captured in the following equation, where

u_h^* represents the optimal solution [8]:

$$||u - u_h|| \leq C^{\infty} ||u - u_h^*|| \tag{3.1}$$

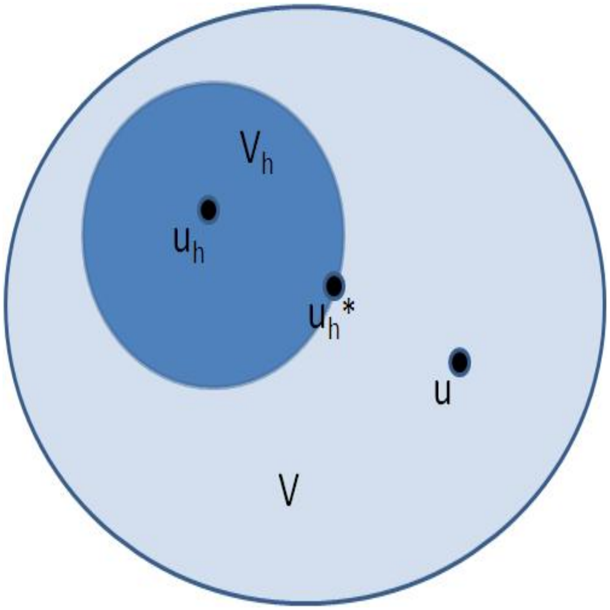


Figure 3.1: V_h Versus V Space.

3.2 Strengths of FEM

FEM is used in many different areas of life, from understanding flows of waters to product stability and weakness testing, but there are certain components of a posed problem that make it a good candidate for these methods. The rigorousness, adaptive capabilities and the user's ability to mold specific domains makes it a strong tool for problems with complex and changing domains. Finally, since the user sets the mesh, it is a great method when there are varying needs of precision. The user can put focus on areas where extreme accuracy is needed and waste less processing power on known or unnecessary areas of the problem. Clearly, the finer the mesh the higher the computational cost [11].

3.3 Weak Form Solution

Although familiar, the differential (strong) formulation of a partial differential equation is often unsuitable for describing real problems. This is due to the the concept that the objective relies upon continuity in the problem, which we do not always have (for example, a traffic jam problem)[10]. A more natural formulation is therefore the so called variational one. A complete understanding of this is based on functional analysis concepts, beyond the purpose of this work. We refer to [9] for a complete description. Let u be the solution to our problem:

$$\frac{\partial u}{\partial t} - \mu \frac{\partial^2 u}{\partial x^2} = f \tag{3.2}$$

$$\begin{cases} u(a,t) = u(b,t) = 0 \\ u(x,0) = u_0(x) \end{cases} \tag{3.3}$$

Let u belong to a set or, more precisely, to a functional space V. The variational formulation can be formally obtained by equation (3.2) with a weighted average approach and after an integration by parts. We say that we look for $u \in V$ such that

$$\int_a^b \frac{\partial u}{\partial t} v + \mu \int_a^b \frac{\partial u}{\partial x} \frac{\partial v}{\partial x} = \int_a^b f v \qquad \forall v \in V \tag{3.4}$$

This is the basis for well posedness theory and the numerical discretization with the finite element method. For the well posedness, a classical result of functional analysis called Lax-Milgram Lemma guarantees that for a problem in the form of equation (3.4) the solution exists, is unique and depends continuously on the data, under usual assumptions on the initial condition function. The same result can be extended to a more general case:

$$\frac{\partial u}{\partial t} - \frac{\partial \mu(x) \frac{\partial u}{\partial x}}{\partial x} + \beta(x) \frac{\partial u}{\partial x} + \sigma(x) u = f(x) \tag{3.5}$$

with

$$\begin{cases} u(a,t) = \phi_a(t) \\ u(b,t) = \phi_b(t) \\ u(x,0) = u_0(x) \end{cases} \tag{3.6}$$

Here as explained earlier, we need an initial condition; exactly what we get when we revert time in the B&S problem with the negative sign in front of the term

$$\frac{\partial \mu(x) \frac{\partial u}{\partial x}}{\partial x} \qquad (\mu > \mu_0 > 0) \tag{3.7}$$

3.4 Discretization

With stability and existence assured by the weak formulation the next step is to build a mesh. The key purpose of this is to replace an infinite dimensional problem with a finite dimensional problem. In the FEM, V_h, is a subspace of V of piecewise polynomial functions. The discretization then comes from the choice of how and where to triangulate this domain. The simplest mesh might entail equispaced nodes, changing this spacing in order to increase or decrease the number of nodes but this need not be the case. As explained earlier, the adaptability of the mesh is a key strength of FEM/FDM. Depending on the boundaries as well as other components inherent in the problem, we may choose to build a mesh that has heavy focus in certain areas. To be more specific, we know that as $h \to 0$, the error $u - u_h \to 0$ but the smaller h is, the higher the computational cost. To best utilize this trade-off, we can make h relatively small in areas of the domain where high accuracy is most desired. For example, with Black-Scholes, often higher accuracy is desired around the exercise price because this is the region that many options will be trading.

3.4.1 1D Discretization

Let us consider the 1D discretization. If we take the model problem:

$$\frac{\partial u}{\partial t} - \mu \frac{\partial^2 u}{\partial x^2} = f \tag{3.8}$$

We replace the variable u representing the solution to the problem, with u_h a piecewise polynomial. Let us introduce a mesh. h is a representative dimension of the size of the elements. We can consider u_h as a sum of Lagrange piecewise polynomials.

$$u_h = \sum U_j(t)\phi_j(x) \tag{3.9}$$

Here $\phi_j(x)$ are the Lagrange piecewise polynomials, meaning $\phi_j = 1$ at node x_j and 0 on all other nodes, and is piecewise linear around the node x_j. Now we will look at the weak formulation:

$$\int_a^b \frac{\partial u_h}{\partial t}\phi_i + \int_a^b \frac{\partial u_h}{\partial x}\frac{\partial \phi_i}{\partial x} = \int_a^b f\phi_i \tag{3.10}$$

and thus

$$\sum_j (\int_a^b \frac{dU_j(t)}{dt}\phi_j\phi_i + \int_a^b U_j \frac{\partial \phi_j}{\partial x}\frac{\partial \phi_i}{\partial x} = \int_a^b f\phi_i \tag{3.11}$$

Now we can define the matrix M with the entries

$$M_{ij} = \int_a^b \phi_j\phi_i \tag{3.12}$$

K, the matrix with the entries

$$K_{ij} = \int_a^b \frac{\partial \phi_j}{\partial x}\frac{\partial \phi_i}{\partial x} \tag{3.13}$$

and vector b with entries

$$b_i = \int_a^b f\phi_i \tag{3.14}$$

Then we are left with the ordinary differential equation system:

$$M\frac{dU}{dt} + KU = b. \tag{3.15}$$

Under the theory of FDM we now perform a time discretization with a θ method. As stated above the FDM approximates the derivative and solves considering the following general equation:

$$\lim_{\Delta x \to 0} \frac{y^{n+1} - y^n}{\Delta x} = f(y^{n+1}, x^{n+1}) \tag{3.16}$$

The theta method considers the following:

$$\begin{cases} (A) \quad \dfrac{y^{n+1} - y^n}{\Delta t} = f(t^{n+1}, x^{n+1}) \\ (B) \quad \dfrac{y^{n+1} - y^n}{\Delta t} = f(t^n, x^n) \end{cases} \tag{3.17}$$

Then we solve the following equation in every time loop:

$$\theta(A) + (1 - \theta)(B) \tag{3.18}$$

More specifically, we must solve:

$$M\frac{U^{n+1} - U^n}{\Delta t} + \theta K U^{k+1} + (1 - \theta)KU^n = \theta b^{n+1} + (1 - \theta)b^n \tag{3.19}$$

Note that U^n for $n = 0$ is given by the initial condition, and is computed for $n > 0$. At each time step we resort to the linear system:

$$(\frac{1}{\Delta t}M + \theta K)U^{n+1} = \theta b^{n+1} + (1 - \theta)b^n - (1 - \theta)KU^n \tag{3.20}$$

By solving this linear system, we have our approximation.

For 1D problems as well as mostly for 2D and 3D, we have a strong theory relating error to θ. That is we can consider the inequality:

$$max_t(\int_a^b (u-u_h)^2 dx)^{\frac{1}{2}} \leq C(\Delta t^p + h^q) \tag{3.21}$$

and literature states that [8]:

1. $p = 1$ for $\theta \neq \frac{1}{2}$ and $p = 2$ for $\theta = \frac{1}{2}$
2. $q = 1$ for linear finite elements and $q = 2$ for quadratic finite elements

Equation (3.20) is a new linear system to be solved at each time step.

To summarize, the process of approximating a differential problem with FEM features 4 steps. Processing, that is, the mesh generation. Assembly, which is the computation of the matrices. Linear system solution where the system of equations is actually solved. Finally, post processing which entails error analysis and visualization. Assembly and linear system solution are the steps that are repeated through a time advancing loop.

This structure for approximation will be similar for the 2D and 3D problems. However, in these cases the mesh generation is much more complicated. It is worth noting that a proper mesh is crucial for an accurate numerical simulation. Mesh adaptive strategies try to find the “best” node localization on the basis of error estimation performed the processing itself.

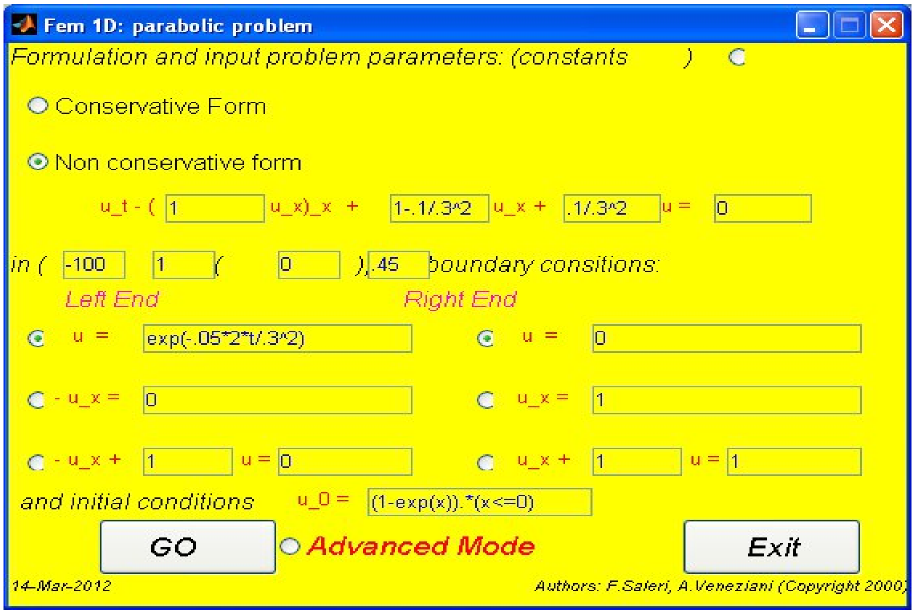
Chapter 4

One Dimensional Experimentation

4.1 Understanding FEM1D

The first task was to understand the components of FEM1D and how to represent the desired problem within the given framework. Since it is a parabolic problem this was the starting point. The parabolic problem was set up according to this screenshot below:

For this test case, we shall consider equation (2.1) for a put with exercise value of \$35, a risk-free rate of .05 and a σ of .3. In solving for w, we must consider equations (2.23) and (2.20); that is, consider the implications on the boundary and initial conditions as well as the parameters. The coefficients were taken as follows. We remember that $k = \frac{2r}{\sigma^2}$, thus in this case, $k = frac.1.3^2$. The boundary of the inherent problem is the entire real axis in x space. For the purposes of a FEM solution, we must discretize the problem and truncate the infinite boundaries. In



Formulation and input problem parameters: (constants) ○

○ Conservative Form

● Non conservative form

u_t - (1 u_x)_x + 1-.1/.3^2 u_x + .1/.3^2 u = 0

in (-100 1 (0),.45 boundary consitions:

Left End

● u = exp(-.05*2*t/.3^2)

○ - u_x = 0

○ - u_x + 1 u = 0

Right End

● u = 0

○ u_x = 1

○ u_x + 1 u = 1

and initial conditions u_0 = (1-exp(x)).*(x<=0)

○ Advanced Mode

14-Mar-2012

Authors: F.Saleri, A.Veneziani (Copyright 2000)

Figure 4.1: FEM1D Screen 1

this discretization, $-\infty$ was approximated by -100 and $+\infty$ was approximated by 1. Although S ranges from zero to infinity, the variable x ranges from negative infinity to infinity, so a high negative number was used to approximate negative infinity accurately enough without requiring too much processing power. -100 would equate to $1.302 * 10^{-42}$ in S space, which would hopefully sufficiently control the truncation error. On the other side of the boundary, it was important to use a number that went reasonably far past the exercise of 35. $S_1 = 35e^1 = 95.1399$. The time boundary was chosen such that $T = 10$ thus $(0, .45)$ was chosen for τ as $\frac{.45*2}{.3^2} = 10$. The right side boundary condition is zero since a put will approach zero as x approaches infinity. The left side boundary condition is $exp(\frac{-.01t}{.3^2})$. This comes from equation (2.18), considering the change of variable. This produces:

$$P_w(-\infty, \tau) = e^{\frac{-2r\tau}{\sigma^2}} \tag{4.1}$$

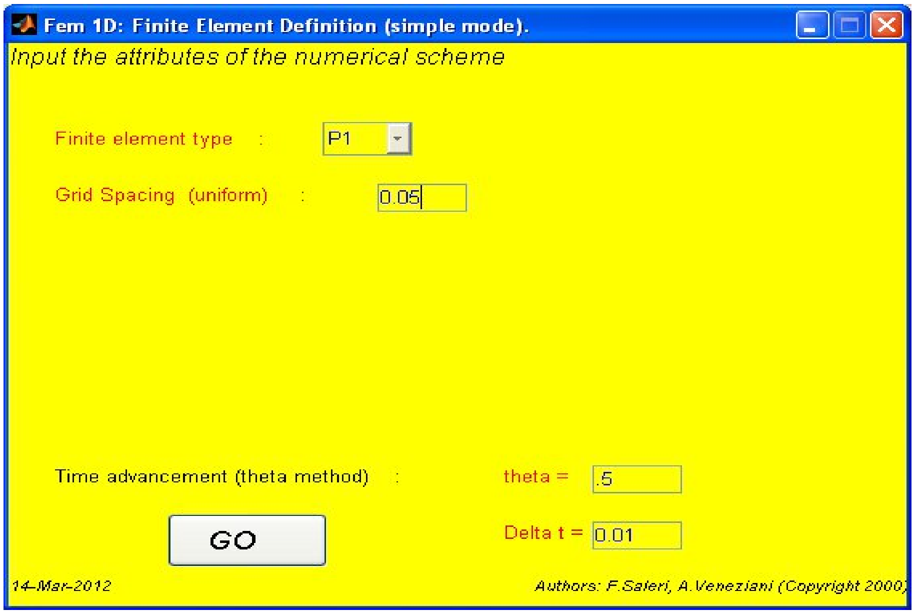
The initial condition is altered as well for the change of variables. Considering equation (2.18) with the change of variable we see:

$$P(x, 0) = \max(E - Ee^x, 0) \tag{4.2}$$

and

$$P_w(x, 0) = \max(1 - e^x, 0) \tag{4.3}$$

The next screen looks like the following 4.2:



Fem 1D: Finite Element Definition (simple mode).

Input the attributes of the numerical scheme

Finite element type : P1

Grid Spacing (uniform) : 0.05

Time advancement (theta method) : theta = .5

Delta t = 0.01

14-Mar-2012

Authors: F.Saleri, A.Veneziani (Copyright 2000)

Figure 4.2: FEM1D Screen 2

This screen was key in interpreting different components of the error because it is used to determine h, Δt and the order of the finite elements represented by P. Under advanced settings, we can be more creative and try to make the mesh more adaptive, but for the purposes of this experiment this was unnecessary. Our test utilizes P1, meaning to solve via piecewise linear continuous finite elements. Grid spacing is h; for this test .05 was used. Δt was taken as .01 in this example. Finally, for θ, .5 was used over 1 as it is more accurate as discussed in above. The output of FEM1D refers to the auxiliary variables which need to be converted to the "real" variables in S and t space. We do so with the code in Appendix A.3. By completing this conversion we see 4.3, 4.4

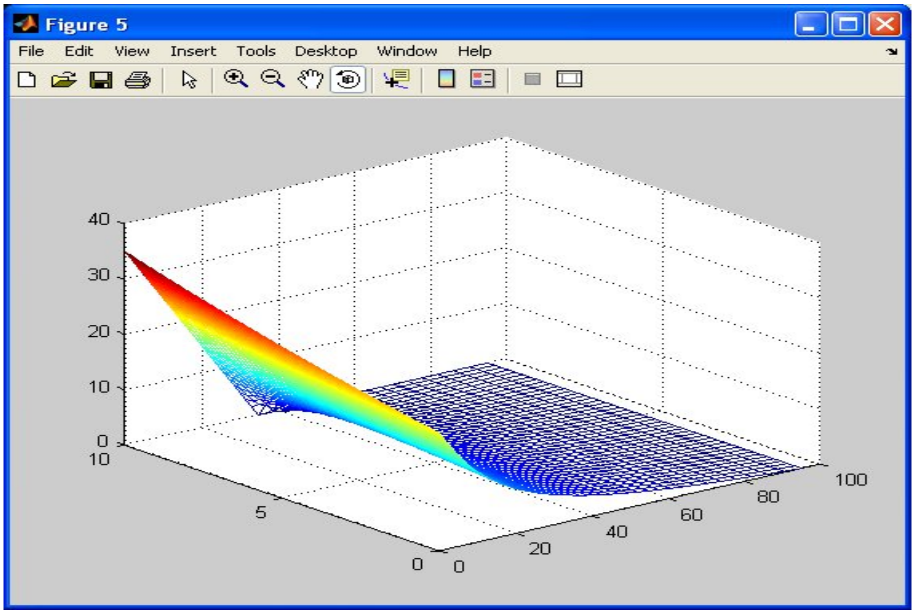


Figure 4.3: Numeric Solution, View 1

Though we have yet to conduct a rigorous analysis on the error, it is plain to see that the general structure is correct. For example, the final condition was clearly

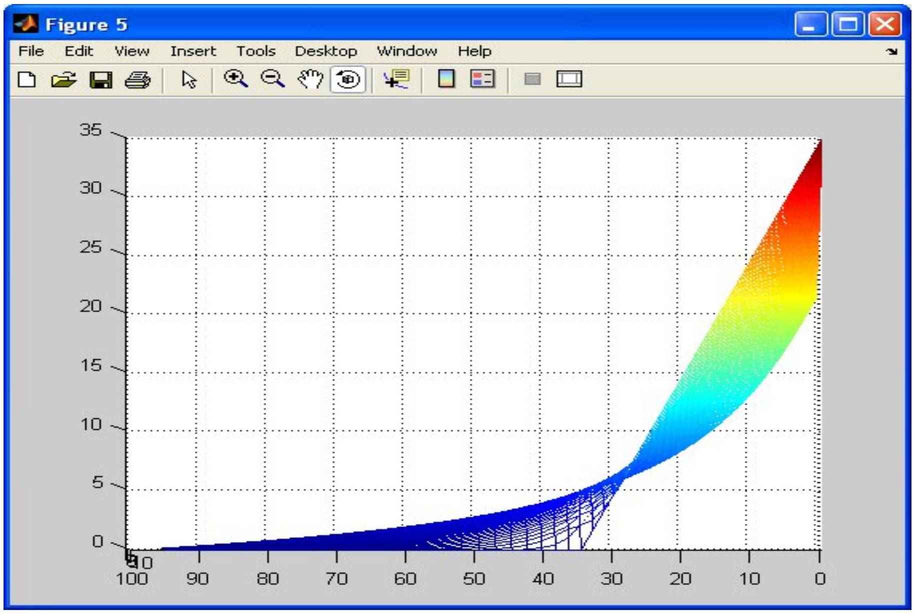


Figure 4.4: Numeric Solution, View 2

captured as presented by the linear segment seen at time 10 in the second image. The curvature also looks good; the slope as time approaches 10 represents the fact that there is less time for drastic change. Also, in the first image the present value boundary condition can be seen clearly.

4.2 Error Analysis

The FEM1D output would then be compared to an analytical solution solved explicitly in a MATLAB subroutine on the nodes that FEM1D solved on. The .m file would produce a matrix with the same dimensions as the FEM1D matrix. In this example, the size of V was 46x2021. The subroutine can be seen in Appendix A.1. It solves for w as well as the Put and Call, given inputs of a risk free rate, volatility, and the size dimensions of the matrix. It does so by running through equation (2.23) in MATLAB notation. By running the subroutine and graphing this example we see 4.5

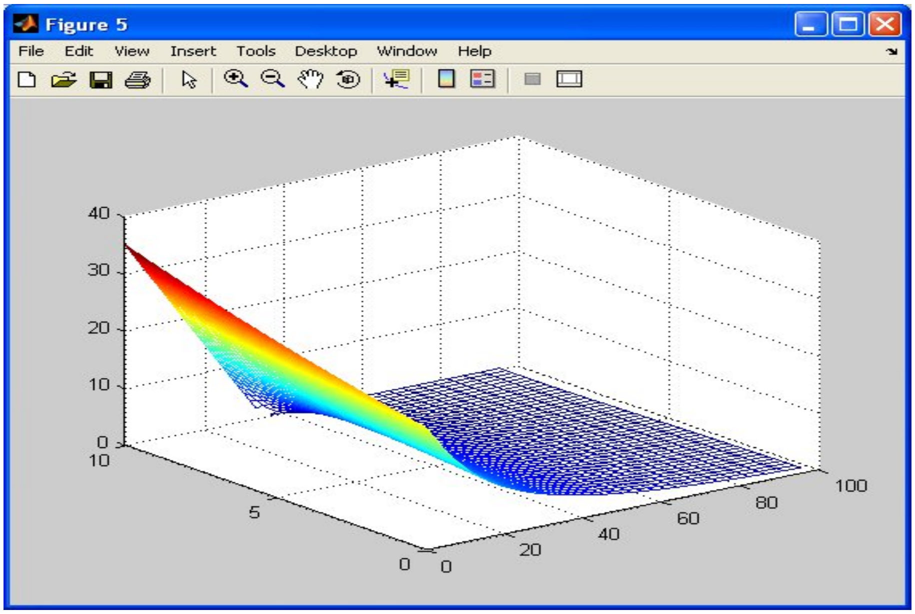


Figure 4.5: Analytic Solution

This is quite similar to the solution solved by FEM1D. To get a better sense of how close the two we can measure the difference of the two. Doing so we see 4.6

We must analyze this picture with an eye towards the expected elements of error. We see that for a portion of the mesh there is nearly zero visible error. This is clearly good and validates the inputs and the output of FEM1D. However, two major visible sources of error arise. One relates to the boundary that is closest to time zero and nearing the S boundary. This error was expected, as we have truncated the infinite domain to a finite boundary point. The reason the error increases as time approaches zero is because the equation used a final condition. That is, we have an exact solution

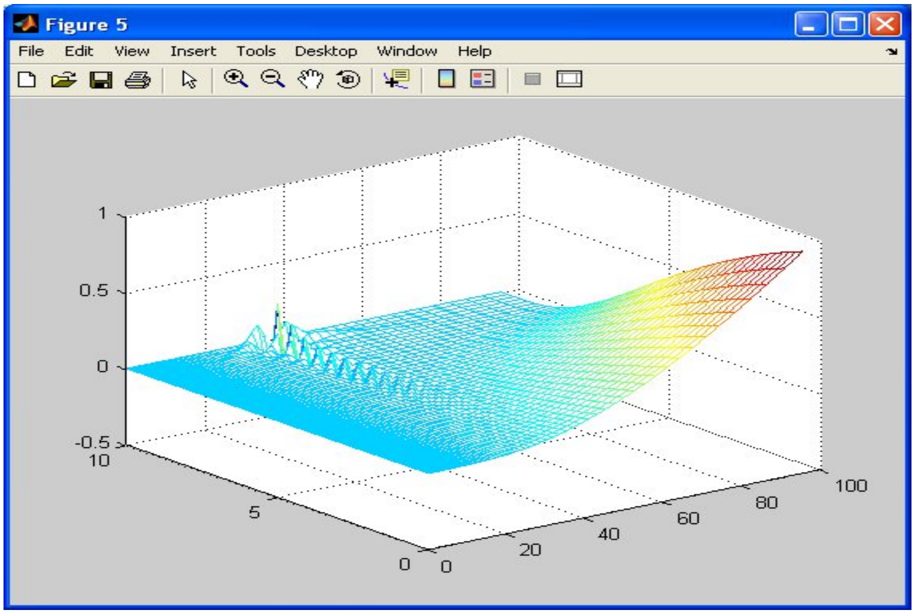


Figure 4.6: Difference of Analytic and Numeric Solutions

at time 10, so there should be no error there. The strength of this initial condition is keeping error down in the area near time 10. However, as it gets away from the certainty of this final condition while also moving towards the truncated boundary point, error arises. The other anomaly visible in the graph is the set of spikes that arises at $S = 35$, as time approaches the end boundary. This is actually not intuitive at all but is an explained occurrence in numerical analysis. It relates to the use of .5 for theta [4].

While the visualization is a strong tool for comparison it fails to emphasize the errors relating to Δt and h . It was important to find the best way to encapsulate the simulation's error in a concise numerical fashion. The most appropriate way for measuring the error for a parabolic problem is either

1. the max in time of the space quadratic average of the error or

2. the quadratic average of the error and derivative error.

1.

$$max_t(\int_a^b (u-u_h)^2 dx)^{\frac{1}{2}}$$

2.

$$(\int_0^t (\int_a^b (u-u_h)^2) + (\frac{\partial u}{\partial x} - \frac{\partial u_h}{\partial x})^2 dx)^{\frac{1}{2}}$$

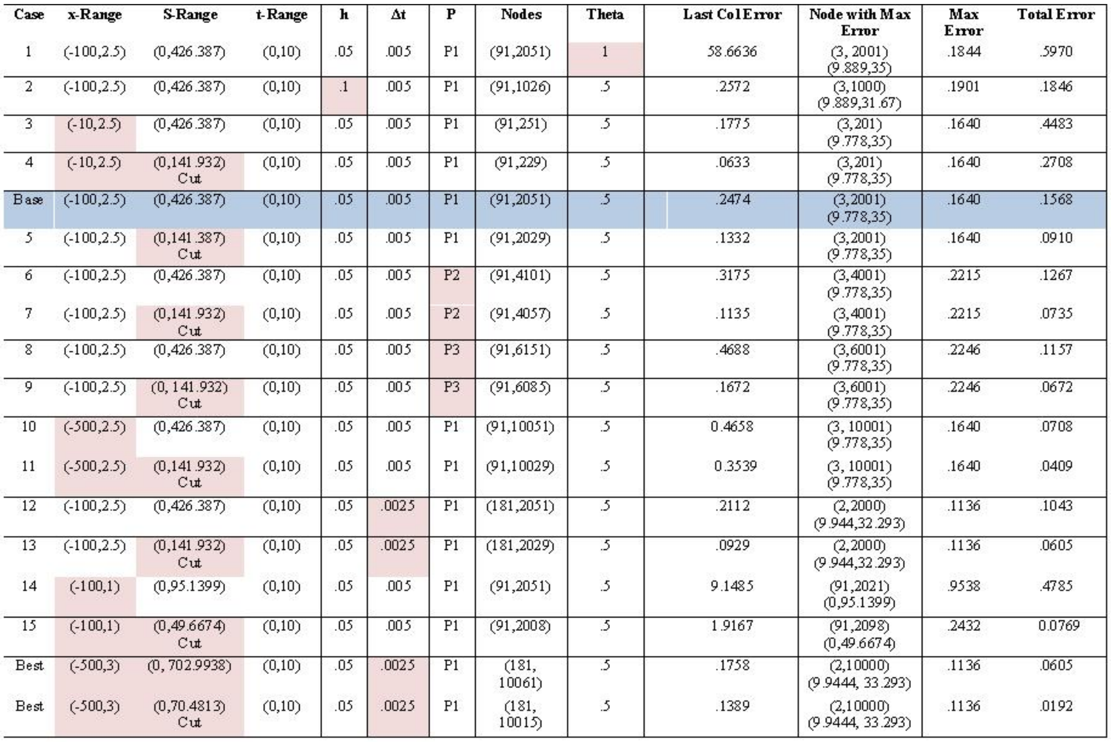
The first of these methods is denoted as error in $L^\infty(L^2)$ norm, the latter is in $L^2(H^1)$. Ultimately, the method used as the main consideration for error analysis was a numerical approximation of (1) used for simplicity. The subroutine can be found in Appendix A.2 . We approximate the integral with a sum of the squared errors at each time step. This step generates new errors by approximating an integral with a finite sum. We can be assured that this error is under control as the mesh becomes finer ($h \to 0$). This will not however evaluate the error introduced through the boundary truncation. In addition to this the error of the last column was an important metric to consider in order to get an idea of the error caused by the boundary. Finally, the node with the max error and the value of that error were noted.

With this example the error from method one is .4785. The absolute error in the last column is 9.3069. The largest error occurs in the node $(46, 2021)$ and is .9538. Though these calculations hold some merit alone, more experimentation would be necessary in order to compare results and get a handle on what elements were truly creating the error.

4.3 Other Test Cases

After repeated trials the following chart was created to compare results 4.7. The elements of the chart recount all of the inputs and outputs discussed above. The "Case" column is added simply for referencing purposes. The x-Range refers to the range in x space, as entered in FEM1D. The S range converts the x-Range into S space according to $x = ln(S/E)$. t-Range refers to the range in t space. Grid spacing refers to h, again an input in FEM1D. Δt refers to the space between each t node. The "P" column refers to the method of solving. That is, whether the posed problem should be solved via piecewise linear, or with degree 2 or 3. The "Nodes" column recounts the size of V. As discussed, θ is the smoothing factor. The right side consists of the outputs. "Last Col Error" is the measurement of the absolute error in the final column, representing the range of time steps at the S boundary. The next column recounts which element in the matrix has the largest error. The column after states the value of that error. Finally, the last column is the error calculated through the subroutine.

The "base" case in the chart above was the starting point for comparison. Each other case was an alteration on this initial set in order to analyze how different inputs affect the output. This would be the basis for analysis in considering the different components of time discretization, space discretization and boundary error. The alteration made to the base case is highlighted in red for convenience.



Case	x-Range	S-Range	t-Range	h	Δt	P	Nodes	Theta	Last Col Error	Node with Max Error	Max Error	Total Error
1	(-100,2.5)	(0,426.387)	(0,10)	.05	.005	P1	(91,2051)	1	58.6636	(3, 2001) (9.889,35)	.1844	.5970
2	(-100,2.5)	(0,426.387)	(0,10)	.1	.005	P1	(91,1026)	.5	.2572	(3,1000) (9.889,31.67)	.1901	.1846
3	(-10,2.5)	(0,426.387)	(0,10)	.05	.005	P1	(91,251)	.5	.1775	(3,201) (9.778,35)	.1640	.4483
4	(-10,2.5)	(0,141.932) Cut	(0,10)	.05	.005	P1	(91,229)	.5	.0633	(3,201) (9.778,35)	.1640	.2708
Base	(-100,2.5)	(0,426.387)	(0,10)	.05	.005	P1	(91,2051)	.5	.2474	(3,2001) (9.778,35)	.1640	.1568
5	(-100,2.5)	(0,141.387) Cut	(0,10)	.05	.005	P1	(91,2029)	.5	.1332	(3,2001) (9.778,35)	.1640	.0910
6	(-100,2.5)	(0,426.387)	(0,10)	.05	.005	P2	(91,4101)	.5	.3175	(3,4001) (9.778,35)	.2215	.1267
7	(-100,2.5)	(0,141.932) Cut	(0,10)	.05	.005	P2	(91,4057)	.5	.1135	(3,4001) (9.778,35)	.2215	.0735
8	(-100,2.5)	(0,426.387)	(0,10)	.05	.005	P3	(91,6151)	.5	.4688	(3,6001) (9.778,35)	.2246	.1157
9	(-100,2.5)	(0, 141.932) Cut	(0,10)	.05	.005	P3	(91,6085)	.5	.1672	(3,6001) (9.778,35)	.2246	.0672
10	(-500,2.5)	(0,426.387)	(0,10)	.05	.005	P1	(91,10051)	.5	0.4658	(3, 10001) (9.778,35)	.1640	.0708
11	(-500,2.5)	(0,141.932) Cut	(0,10)	.05	.005	P1	(91,10029)	.5	0.3539	(3, 10001) (9.778,35)	.1640	.0409
12	(-100,2.5)	(0,426.387)	(0,10)	.05	.0025	P1	(181,2051)	.5	.2112	(2,2000) (9.944,32.293)	.1136	.1043
13	(-100,2.5)	(0,141.932) Cut	(0,10)	.05	.0025	P1	(181,2029)	.5	.0929	(2,2000) (9.944,32.293)	.1136	.0605
14	(-100,1)	(0,95.1399)	(0,10)	.05	.005	P1	(91,2051)	.5	9.1485	(91,2021) (0,95.1399)	.9538	.4785
15	(-100,1)	(0,49.6674) Cut	(0,10)	.05	.005	P1	(91,2008)	.5	1.9167	(91,2098) (0,49.6674)	.2432	0.0769
Best	(-500,3)	(0, 702.9938)	(0,10)	.05	.0025	P1	(181, 10061)	.5	.1758	(2,10000) (9.9444, 33.293)	.1136	.0605
Best	(-500,3)	(0,70.4813) Cut	(0,10)	.05	.0025	P1	(181, 10015)	.5	.1389	(2,10000) (9.9444, 33.293)	.1136	.0192

Figure 4.7: Error Table

4.3.1 Analyzing the Results

The results showed how certain elements of error were stronger than others. Additionally, the results help clarify what elements are controllable and what are not. The boundary error stuck out as the most prominent source of error so this was the first observed metric. To address this, a method of exceeding the desired S by a large portion, then truncating the scope of the analyzed area to a segment that relates to the desired portion of S. Consider this 4.8:

The results of this strategy are shown by the word “Cut” in the “S-Range” column. Case 5 was the test that only altered this metric. A max x-Range of 2.5,

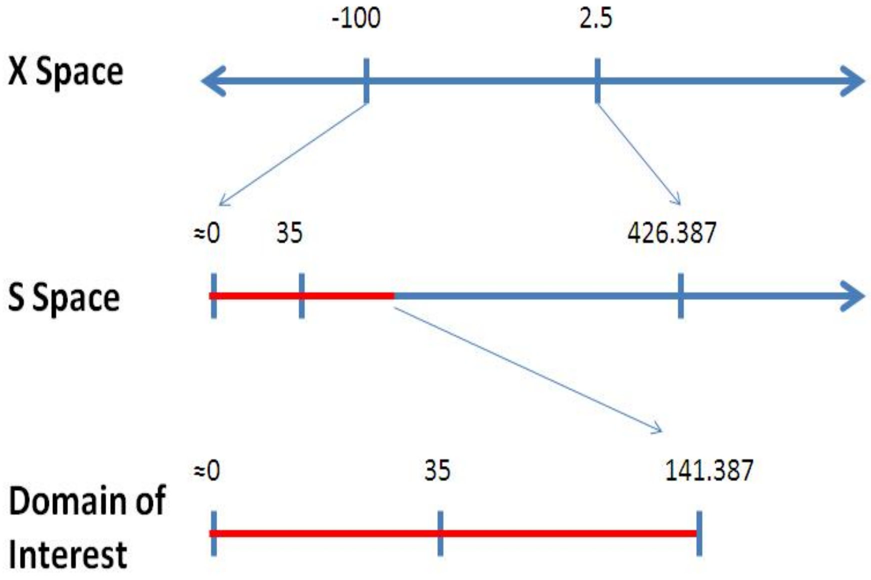


Figure 4.8: Domain of Interest

will bring the S-Range up to 426.387. This is far beyond any practical portion; no one is going to be looking at options with an exercise price of $35 when the stock price is $426.387. This high number alone helps to reduce truncation error. The larger that number becomes, the closer it is to its intended positive infinity, thus less error will arise. Additionally, altering the scope helps to eliminate this error which is located around the boundary. Since we only need to examine the solution in the range around $35 we can take a cut between 35 and 426.387 that will limit the damage of the boundary error. In case 5, this cut was taken at 141.387. The new error in the last column is .1332, and the total error calculation decreases down to .0910 from .1568. This tactic comes at the cost of computation; by extending the interval we create a larger linear system which adds computational complexity. A large portion of the error is attributable to the boundary and with some simple manipulation this error can be strongly contained. Looking through the chart, the cases that are cut tend to reduce total error down to about three fifths. Additionally, the errors in the last columns are still relatively high, so there would be more room to take advantage of this method.

Another source of this truncation boundary error arises from the replacement of negative infinity with a large negative number. To test the effect of this the left boundary in cases 10 and 11 has been increased to -500. This had the strongest effect of all of the different tests, reducing total error to .0708, and when cut going down to .0409. This was surprising; intuitively we may expect the other boundary to have a much more profound effect on error because the error it produces is more easily observed.

Case 12 deals with time discretization. In this experiment the number of nodes was doubled by halfing Δt. This method decreased the error by .0525. This cut is again a sizable decrease and suggests time discretization error is a material source of error. Interestingly, this is the only test that the error spike did not have a max at $S = 35$ but instead at $S = 32.293$.

The negative effects of decreasing the space discretization were captured by case 2. In this method, h was doubled and it had a material but not drastic impact. The total error rose up to .1846, a little less than .03. A similar jump in error was seen in the max error node from the $35 spike anomaly. The error in the last column was not significantly impacted.

Beyond these main tests some alternate methods were considered and analyzed. The results of utilizing finite elements of degree 2 (P2) and 3 (P3) were reported.

As can be seen in cases 6 through 9, this had varying results. Among the different methods, this was least successful in decreasing total error, and actually increased the spike anomalies that exist on the $35 line. As for the time advancing scheme, we tried different values of θ but $\theta = \frac{1}{2}$ turned out to be the best option.

Ultimately, it was clear that error was arising from time and space discretization as well as boundary error. The "best" case column combines all of the elements to try to decrease error as much as possible. The x-Range was taken as (-500,3), which was cut by 56 space nodes. Δt was taken as .0025. With these inputs, we see a total error of .0192; this is a large improvement over the base case value. Figure 4.9 shows the error graph of this best case scenario. This best case scenario sharply cuts into all of the components of error analyzed. Additionally, the large gap between the cut and non-cut max in time of the space quadratic average calculation implies that still a large portion of the error is qualified by our boundary truncation. This is a byproduct of the analytical solution comparison. Often in a more practical case boundary data on a finite domain is known. Therefore we need to be less concerned with this component and find comfort in the fact that the space and time discretization are very much under control.

The two main restrictions to reducing this error are processing power and the spike anomaly that occurs. In regards to the spikes, it is hard to tell exactly what causes them, and efforts to reduce them have been fairly unsuccessful. Perhaps they are heavily influenced by time discretization error as the tests on this metric were the only ones to have a reasonable impact on the spikes. Beyond this, the technical capabilities of the machine in use can be a limiting factor. This experimentation

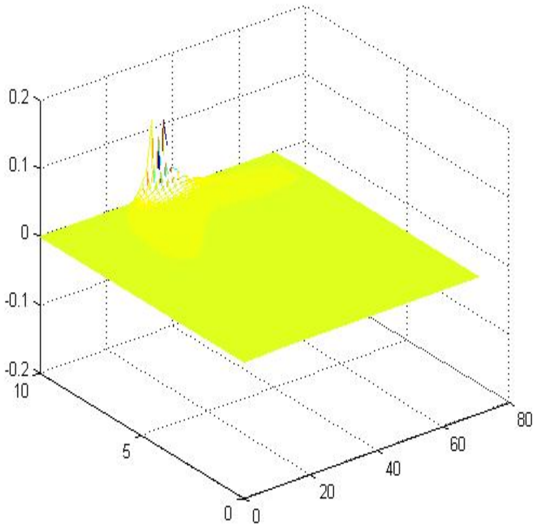


Figure 4.9: Best Case Error

was performed on a Windows XP Professional, with an Intel(R) Core(TM)2 Duo CPU, P8400 @2.26GHz (2 CPUs). The best case scenario presented above took around an hour to solve with FEM1D, and increasing the space discretization often resulted in crashing the program. This represents a hurdle for practicality. Especially because this is a simple one dimensional example, it is foreseeable that the necessary processing power for extremely accurate results could be impractical when speed is necessary and more complex problems are posed. Some of this processing power could be cut down by smarter meshing, but this too have their limitations.

Chapter 5

FreeFem++ Experimentation

5.1 Coding

While the traditional 2D Black-Scholes problem is posed in the form of equation (2.2), the coding and solving of the equation with FEM requires the weak formulation of the problem. Therefore the equation, whose weak formulation can be seen in the code in Appendix A.4, reads:

$$\begin{aligned} &\frac{\partial u}{\partial t} - \frac{\partial}{\partial x}\left(\frac{x{\sigma_x}^2}{2}\frac{\partial u}{\partial x}\right) - \frac{\partial}{\partial y}\left(\frac{y{\sigma_y}^2}{2}\frac{\partial u}{\partial y}\right) - \\ &\frac{\partial}{\partial x}\left(\frac{\rho\sigma_x\sigma_y xy}{2}\frac{\partial u}{\partial y}\right) - \frac{\partial}{\partial y}\left(\frac{\rho\sigma_x\sigma_y xy}{2}\frac{\partial u}{\partial x}\right) + ruv = 0 \end{aligned} \tag{5.1}$$

As explained in chapter 3, this formulation requires multiplying by a test function v and integrating. This weak formulation of the problem is now certainly well posed; we know the solution exists, is unique and depends continuously on the data.

5.2 1D Augmentation

Equation (2.2) states the 2D Black-Scholes formula. A 2D Black-Scholes code comes with the download of FreeFem++ and a more advanced code can be found in the manual for FreeFem++ [7]. This was the starting point for analysis in 2D. Some alterations have been made to the code to change user interaction and visualization. The altered code can be seen in the Appendix. With an understanding of the 1D situation, it was a valuable exercise to see the FreeFem++ interpretation of the 1D problem. To do this we can simply multiply the y component by 0, and take a $\rho = 0$ (we remember that this coefficient is the correlation between the two stocks). Thus the second stock price will have no effect on the solution. In doing so we see:

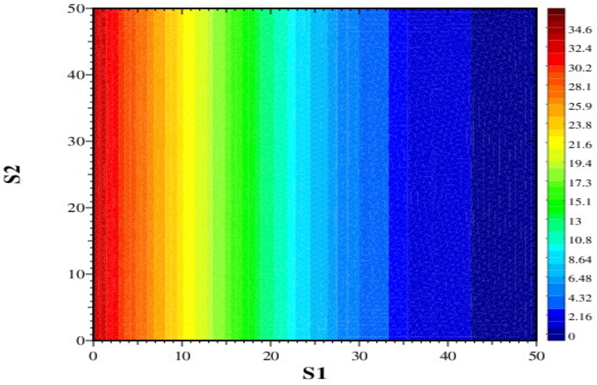


Figure 5.1: FreeFem++ 1D Error: Value

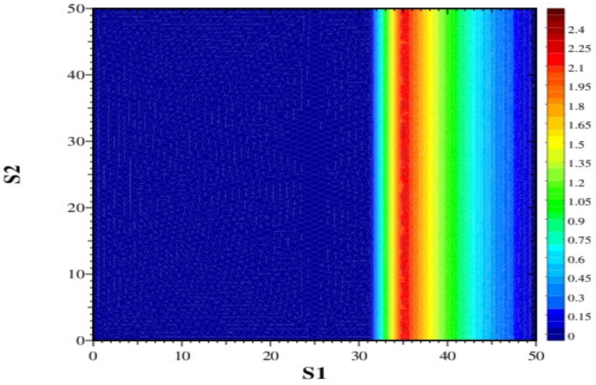


Figure 5.2: FreeFem++ 1D Error: Time Value

This solution is as expected. The y axis has no relevance on the solution since we are treating stock two as if it does not exist. We can then compare the values of option at each time step given different values along the x axis. Comparing each time step we see the line of $V = 0$ pushing closer and closer to the $S = 35$ with this line ending at time T on $S = 35$. We see small blips of error along the final iso value color. This is likely due to the fineness of the mesh. If we severely decrease the number of nodes in mesh down to 10 on each border, we see:

As can be seen, the majority of the error arises around this final iso value. Similar to the earlier observations, this error decreases as time approaches T. For example, one of the final time images shows 5.5:

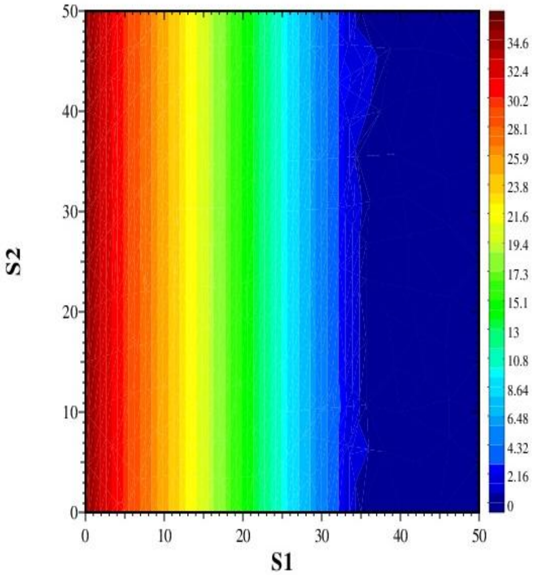


Figure 5.3: Coarse Mesh 1D, Beginning

5.3 2D Analysis

Considering a correlation of of $\rho = .3$ and all other constants taken the same, the image that results from the the 2D simulation is 5.6.

We see the image curve inward most in the range where S_1 and S_2 approach 35. This portion relates the time value of the two options; both options are near the border of being in or out of the money so the curve relates this likelihood of either or both getting in the money.

Comparing again in the 2D case with a smaller mesh we see similar results with the error tending heavily towards the boundary. We also notice that with a problem such as this, we can control error with less processing power with a more adaptive mesh. Here we start with 20 nodes on each side and the program works through an

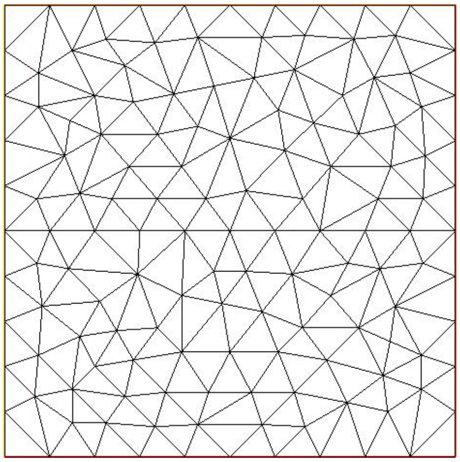


Figure 5.4: 10x10 Mesh

adaptive process to focus the mesh on key areas 5.8:

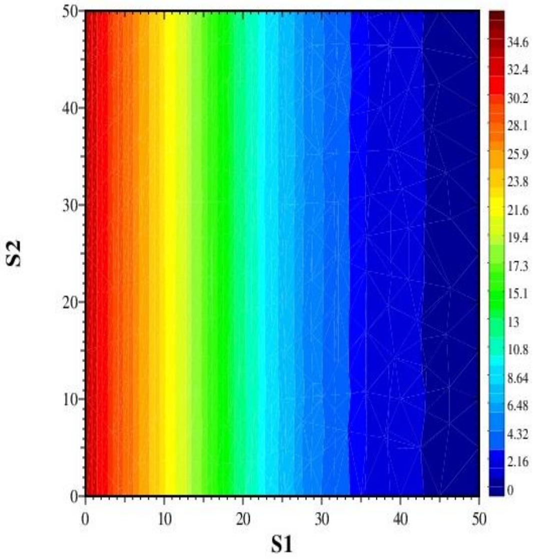


Figure 5.5: Coarse Mesh 1D, End

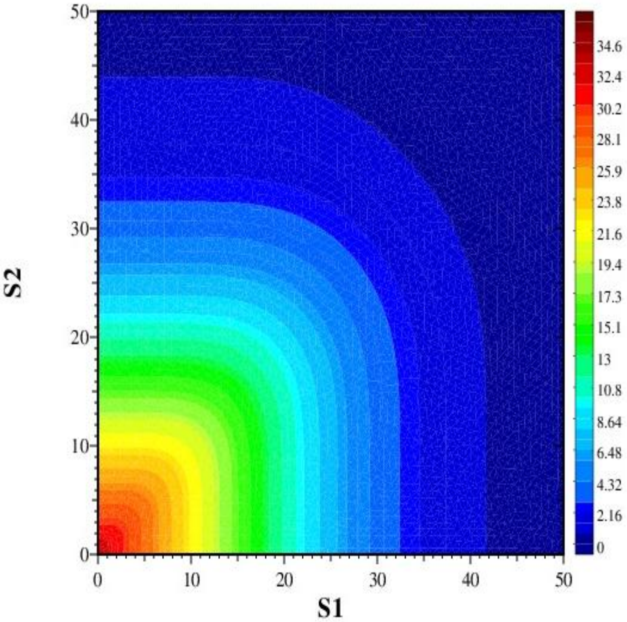


Figure 5.6: 60x60 2D Image End, Value

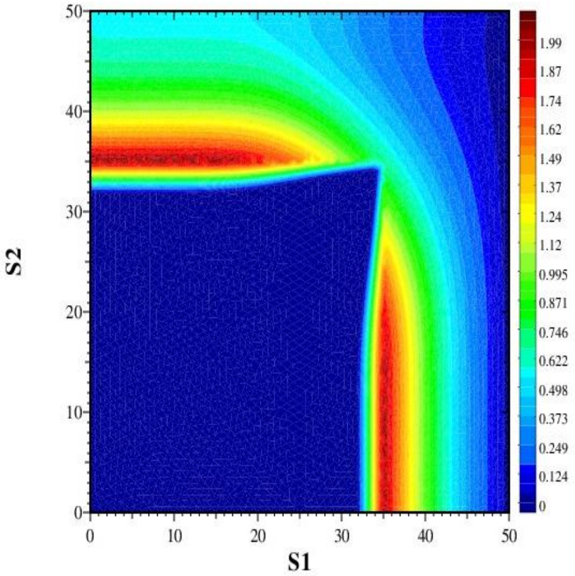


Figure 5.7: 60x60 2D Image End, Time Value

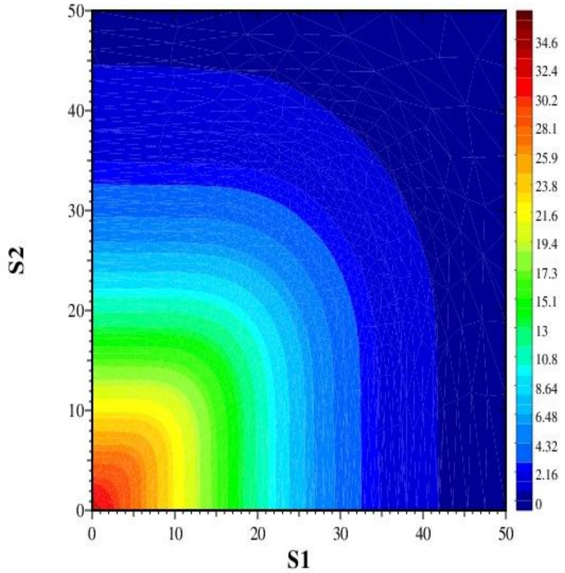


Figure 5.8: 20x20 Adapt Mesh, 2D Image End

Chapter 6

Conclusion

6.1 Findings

Through the 1D error analysis of a FEM solution to the Black-Scholes problem we have discovered components that were hypothesized as well as pieces that were less expected. The most visible and abundant error is *boundary* error. This error arises entirely from the truncation of the infinite boundaries. However, as mentioned, this error is simply a byproduct of the analytical solution we are comparing with. In a "practical case", we know boundary data on a finite domain, so the error components to consider are the discretization error segments, which appear to be under control in the experiments above. Space and time discretization played very material roles in the error and were largely controllable, at the cost of processing power. Though more experimentation would need to be done, a FEM/FDM approach seems like a strong tool in analysis of the Black-Scholes equation at least for European options.

For American options, as discussed, strong assumptions need to be made, and that is not the concern of this paper. Putting together options in more complex ways, FEM/FDM could still be a strong tool; however, the power accuracy trade off will become more crucial. As the problems become more complex with additional elements, it will ramp up the necessary technical power. To combat this mesh adaptation is key; there are large segments of stock prices that are not particularly relevant because they would never trade practically, so we can limit the mesh in these areas focusing more on the crucial points nearer to the exercise price.

The methods outlined in this paper can be easily followed if one wishes to repeat the methods used in this research. The images of FEM1D provide clarity on this segment and subroutines as well as MATLAB steps are laid out in the appendices.

6.2 Going Forward

There are numerous branches to consider in moving forward with this study. More analysis with the two dimensional case would be the first step. FreeFEM++ is a powerful tool that can be utilized more fully in ways not touched upon in this study. A next step could be to follow in the footsteps of the 1D experimentation in the 2D case. Additionally, we could consider more the methods necessary to analyze American options. American options are generally more prevalent than European and more interesting because of their nature. The literature suggests heavy limitations on the analysis of American options with these methods, but this realm could be explored further. Perhaps we could repeat Dempster's work [2] and apply the concepts of the research of this paper to see if there were ways to better interpret results. Another consideration moving forward would be to step out of the theoretical spectrum and consider real world values. In this way we could observe how the implications of this study hold outside of the theoretic construct. Finally, as an additionally step it would be interesting to try a similar code in FreeFEM3D and observe the results with three assets.

Appendix A

Subrountines

A.1 1D Exact Solution Subroutine

```
\%set variables and size/boundaries of matrix

r=.05;sigma=.3;T=10;

n= 2021;

x=[-100:101/(n-1):1];

t=[0:0.005:T*sigma^2/2];

k=2*r/sigma^2;
```

```
\%solve for w

[mx,nx]=size(x);

[mt,nt]=size(t);

for i=1:nx

for j=1:nt

dp(i,j)=x(i)/sqrt(2*t(j))+0.5*(k+1)*sqrt(2*t(j));

dm(i,j)=x(i)/sqrt(2*t(j))+0.5*(k-1)*sqrt(2*t(j));

Nm=0.5*(erfc(0)+erf(dm(i,j)/sqrt(2)));

Np=0.5*(erfc(0)+erf(dp(i,j)/sqrt(2)));

v(i,j)=exp(0.5*(k+1)*x(i)+0.25*(k+1)\^{ }2*t(j))*Np-
exp(0.5*(k-1)*x(i)+0.25*(k-1)\^{ }2*t(j))*Nm;

w(i,j)=exp(-(k-1)/2*x(i)-(k+1)\^{ }2/4*t(j))*v(i,j);
```

```
end;

end;



E=35;

s=E*exp(x);

tt=T-2*t/sigma\^{ }2;

\%solve for Put and Call

[w1, w2]= size(w);

Call=zeros(w1,w2);

Put=zeros(w1,w2);

for i=1:w1
for j=1:w2

dp(i,j)=x(i)/sqrt(2*t(j))+0.5*(k+1)*sqrt(2*t(j));

dm(i,j)=x(i)/sqrt(2*t(j))+0.5*(k-1)*sqrt(2*t(j));

Nm=0.5*(erfc(0)+erf(dm(i,j)/sqrt(2)));

Np=0.5*(erfc(0)+erf(dp(i,j)/sqrt(2)));

Call(i,j)= -E*exp(-r*(T-tt(j)))*Nm+ s(i)*Np;

Put(i,j)= E*exp(-r*(T-tt(j)))-s(i)+Call(i,j);

end;

end;
```

A.2 Error Solver 1D Subroutine

```
[a,b]=size(Err);

buf=zeros(a,1);
```

```
length=s(b);

for k=1:a

buf(k)=(Err(k,:)*Err(k,:)')*length/b;

end;

total=sqrt(max(buf))
```

A.3 FEM1D Conversion

```
load Results;
E=35;
V=E*uh;
t=10-2*tempo/.3^2;
S=E*exp(coord);
mesh(S,t,V)
```

A.4 FreeFem++ 2D BS Code

```
int s=10; // y-scale
int m=60;
int L=50;
int LL=50;


border aa(t=0,L){x=t;y=0;};    //length 50 borders
border bb(t=0,LL){x=L;y=t;};
border cc(t=L,0){x=t ;y=LL;};
border dd(t=LL,0){x = 0; y = t;};
mesh th = buildmesh(aa(m)+bb(m)+cc(m)+dd(m));
//mesh with 60 points on the borders
fespace Vh(th,P1);             // continuous piecewise linear mesh
real sigmax=0.3;               // volatility x
real sigmay=0.3;               // volatility y
real rho=0.0;                 // correlation
real r=0.05;                  // risk free
real K=35;                    // strike price
real dt=0.01;                 // timestep
real eps=0.3;
func f = max(K-max(x,0*y),0.);   // put options,  max(strike-value, 0)
Vh u=f,v,w;
func beta = 1; // (w<=f-eps)*eps + (w>=f) +
(w<f)*(w>f-eps)*(eps+(w-f+eps)/eps)*eps);
plot(u,wait=1,value=1);
//th = adaptmesh(th,u,abserror=1,nbjacoby=2,
//err=0.004, nbvx=5000, omega=1.8,ratio=1.8, nbsmooth=3,

//splitpbedge=1, maxsubdiv=5,rescaling=1 );

u=u;

plot(u,wait=1,value=1);

Vh xveloc = -x*r+x*sigmax^2+x*rho*sigmax*sigmay/2;

Vh yveloc = -y*r+y*sigmay^2+y*rho*sigmax*sigmay/2;

int j=0;

int n;

real t=0;

problem eq1(u,v,init=j,solver=LU) = int2d(th)(

u*v*(r+1/dt/beta)

+ dx(u)*dx(v)*(x*sigmax)^2/2.

+ dy(u)*dy(v)*(y*sigmay)^2/2.

+ dy(u)*dx(v)*rho*sigmax*sigmay*x*y/2.

+ dx(u)*dy(v)*rho*sigmax*sigmay*x*y/2. )

+ int2d(th)( -v*convect([xveloc,yveloc],dt,w)/dt/beta)

+ on(bb,u=f*exp(-r*t));

//+ on(Vh,u>f*exp(-r*t));

;

int ww=1;

for ( n=0; n*dt <= 1.0; n++)

{

t=t+dt;
```

```
cout <<" iteration " << n << " j=" << j << endl;
w=u;
eq1;
v=f;
//plot(v,wait=ww,value=1,fill=1);
v=max(u-f,0.);
plot(th,wait=ww,value=1);
plot(u,wait=ww,value=1,fill=1);
//plot(v,wait=ww,value=1,fill=1);
//u = max(u,f);
//plot(u,wait=ww,value=1,fill=1);
//ww=0;
/*if(j>10) { cout << " adaptmesh " << endl;
th = adaptmesh(th,u,verbosity=1,abserror=1,nbjacoby=2,
err=0.001, nbvx=5000, omega=1.8, ratio=1.8, nbsmooth=3,
splitpbedge=1, maxsubdiv=5,rescaling=1) ;
j=-1;
xveloc = -x*r+x*sigmax^2+x*rho*sigmax*sigmay/2;
yveloc = -y*r+y*sigmay^2+y*rho*sigmax*sigmay/2;
u=u;
ww=1;
};
*/
```

```
j=j+1;

cout << " j = " << j << endl;

};

v = u-f; //max(u-f,0.);

plot(v,wait=1,value=1);

plot(u,wait=1,value=1);

plot(th,wait=1);
```

Bibliography

[1] freefem, December, 2010. [Online; accessed 10-November-2011].

[2] MAH Dempster and JP Hutton. Fast numerical valuation of american, exotic and complex options. *Applied Mathematical Finance*, 4(1):1–20, 1997.

[3] S.J. Farlow. *Partial differential equations for scientists and engineers.* Dover Pubns, 1993.

[4] L. Formaggia and A. Veneziani. *Solving Numerical PDEs: problems, models exercises.* Springer, 2011.

[5] R. Geske and K. Shastri. Valuation by approximation: a comparison of alternative option valuation techniques. *Journal of Financial and Quantitative Analysis*, 20(01):45–71, 1985.

[6] I. Gladwell, J. Nagy, and W.E. Ferguson Jr. Introduction to scientific computing with matlab. http://www.mathcs.emory.edu/ nagy/.

[7] F. Hecht, O. Pironneau, A. Le Hyaric, and K. Ohtsuka. Freefem++ manual, 2005.

[8] A. Quarteroni and A. Valli. *Numerical approximation of partial differential equations*, volume 23. Springer Verlag, 2008.

[9] S. Salsa. *Partial differential equations in action: from modelling to theory.* Springer Verlag, 2008.

[10] RJ Le Veque. *Numerical methods for conservation laws.* Birkhauser, 1990.

[11] Wikipedia. Finite element method, 2012. [Online; accessed 10-March-2012].

[12] Wikipedia. Black–scholes, April 2012. [Online; accessed 5-January-2012].

[13] Wikipedia. Brownian motion, March 2012. [Online; accessed 5-March-2012].