Master Thesis

Pricing of American options with discrete dividends using a
PDE and a volatility surface while calculating derivatives
with automatic differentiation

David Hjelmberg & Björn Lagerström

LIU - IEI - TEK - A - - 14/02087—SE
Pricing of American options with discrete dividends using a PDE and a volatility surface while calculating derivatives with automatic differentiation

Production Economics, Department of Management and Engineering, Linköping University

David Hjelmberg & Björn Lagerström

LIU - IEI - TEK - A - - 14/02087—SE

Master Thesis: 30 hp
Level: A
Supervisor: Jörgen Blomvall,
Production Economics, Department of Management and Engineering,
Linköping University
Examiner: Fredrik Berntsson,
Department of Mathematics, Linköping University
Linköping: September 2014
Abstract

In this master thesis we have examined the possibility of pricing multiple American options, on an underlying asset with discrete dividends, with a finite difference method. We have found a good and stable way to price one American option by solving the BSM PDE backwards, while also calculating the Greeks of the option with automatic differentiation. The list of Greeks for an option is quite extensive since we have been using a local volatility surface.

We have also tried to find a way to price several American options simultaneously by solving a forward PDE. Unfortunately, we haven’t found any previous work that we could use with our local volatility surface, while still keeping down the computational time. The closest we got was to calculate the value of a compound option in a forward mode, but in order to use this to value an American option, we needed to go through an iterative process which calculated a forward or backward European PDE in every step.

Keywords: American options, BSM PDE, discrete dividends, forward PDE, local volatility surface, automatic differentiation.
Acknowledgements

We would like to thank our supervisor Jörgen Blomvall and examiner Fredrik Berntsson for all their help during the progress of this master thesis.
Nomenclature

Most of the reoccurring symbols and abbreviations are described here.

Symbols

\( S_t \)  The value of the underlying asset at time \( t \).
\( S^*(t) \)  The EEB at time \( t \).
\( V \)  The price of a derivative contingent on \( S_t \).
\( c \)  The price of a European call option.
\( p \)  The price of a European put option.
\( C \)  The price of an American call option.
\( P \)  The price of an American put option.
\( T \)  The time of maturity of an option.
\( K \)  The strike of an option.
\( D \)  Dividend amount.
\( D_t \)  Dividend time.
\( r \)  The risk-free rate.
\( \sigma \)  The volatility of the underlying asset.
\( \sigma_{i,j} \)  The local volatility of node \((i,j)\) in the local volatility surface.
\( f_{i,j} \)  The value of an option in node \((i,j)\) in the finite difference grid.
\( \Delta t_i \)  Difference in \( t \) between the nodes \( f_{i,j} \) and \( f_{i+1,j} \).
\( \Delta Z_j \)  Difference in \( Z \) between the nodes \( f_{i,j} \) and \( f_{i,j+1} \).
\( M \)  Number of nodes in the \( Z \)-direction.
\( N \)  Number of nodes in the \( t \)-direction.
\( \theta \)  The percentage of the implicit difference method to use in combination with the explicit difference method.
\( \alpha \)  The percentage of a discrete dividend that the underlying asset drops by on the ex-dividend date.
\( \phi \)  The maximum percentage of the current value of the underlying asset that can be used as dividend.

Abbreviations

BSM  Black-Scholes-Merton
EEB  Early Exercise Boundary
PDE  Partial Differential Equation
PIDE  Partial Integro-Differential Equation

Hjelmberg, Lagerström, 2014.
# Contents

1 Introduction .......................................................... 1
   1.1 Purpose .......................................................... 1
   1.2 Delimitations .................................................... 1
   1.3 Topics covered ................................................... 1

2 Method .................................................................. 3

3 Theoretical Background ............................................ 5
   3.1 Options ............................................................. 5
   3.2 BSM PDE ............................................................ 5
   3.3 Finite Difference Methods ...................................... 6
      3.3.1 Implicit ......................................................... 7
      3.3.2 Explicit ......................................................... 8
      3.3.3 Crank-Nicolson ................................................. 8
      3.3.4 Combining the Implicit and Explicit Solutions .......... 9
   3.4 Volatility Surface .................................................. 9
   3.5 Gradient and Hessian ............................................. 10

4 Literature Study ....................................................... 13
   4.1 Early Exercise Premium .......................................... 13
      4.1.1 Early Exercise Boundary ..................................... 14
      4.1.2 Finding the Early Exercise Boundary ...................... 15
      4.1.3 Change of Variables ......................................... 16
   4.2 Discrete Dividends .................................................. 16
      4.2.1 Dividends Effect on the Assets Value ..................... 17
      4.2.2 Dividends Bigger than the Assets Value ................. 17
   4.3 Replicate an American Option .................................... 18
      4.3.1 Analytical Solution for Compound Options ................ 18
      4.4 Automatic Differentiation ..................................... 18

5 Forward PDE .......................................................... 21
   5.1 Dupire Equation .................................................... 21
   5.2 Partial Integro-Differential Equation ........................... 22
   5.3 Compound Options ................................................ 22

Hjelmberg, Lagerström, 2014. xi
# Contents

6 Implementation 25

6.1 The Grid 25

6.2 Pricing of American Options 26

6.3 Finding the EEB 27

6.4 Gradient and Hessian of Option Prices 28

6.5 Our Programs 29

6.5.1 Option Prices 30

6.5.2 Gradients 30

6.5.3 Hessians 31

6.6 Other Programs 31

6.6.1 Implementation with a Change of Variable 31

6.6.2 Compound Option 32

7 Results 33

7.1 Validation 34

7.1.1 European Prices 34

7.1.2 Comparison with DerivaGem 36

7.1.3 Replicate an American Call Option 36

7.2 Option Prices And Premiums 37

7.3 Convergence 41

7.3.1 Different Thetas 41

7.3.2 Critical Ratios 47

7.3.3 Options Almost at Maturity 48

7.4 Dividends 49

7.4.1 The Early Exercise Boundary 52

7.5 Derivatives 53

7.6 Runtimes 61

8 Discussion 63

8.1 The Theta Method 63

8.2 Derivatives 63

8.3 The Phi Variable 64

8.4 Change of Variables for the American Put Option 64

9 Conclusions 67

9.1 Summarized Results 67

9.2 How to Enhance or Continue This Master Thesis 67

9.2.1 Forward PDE for American Options 68

9.2.2 Automatic Differentiation 68

9.2.3 Second Order Derivatives for Nodes Exercised Early 68

9.2.4 Methods Using the Boundary Condition of the EEB 68

9.2.5 Increased Accuracy for the Finite Differences 68

Bibliography 69

A Derivation of the Theta Method 73

A.1 Implicit 73

A.2 Explicit 74

A.3 Combining Implicit and Explicit Solution 74

A.4 Boundary nodes 75
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.4.1 Call option</td>
<td>75</td>
</tr>
<tr>
<td>A.4.2 Put option</td>
<td>75</td>
</tr>
<tr>
<td><strong>B Interpolation</strong></td>
<td>77</td>
</tr>
<tr>
<td>B.1 Linear Interpolation</td>
<td>77</td>
</tr>
<tr>
<td>B.2 Four Point Interpolation</td>
<td>77</td>
</tr>
<tr>
<td>B.3 Bilinear Interpolation</td>
<td>77</td>
</tr>
<tr>
<td><strong>C Numerical Derivatives</strong></td>
<td>79</td>
</tr>
<tr>
<td>C.1 First Order Derivatives</td>
<td>79</td>
</tr>
<tr>
<td>C.2 Second Order Derivatives</td>
<td>79</td>
</tr>
<tr>
<td><strong>D AD Variables</strong></td>
<td>81</td>
</tr>
<tr>
<td>D.1 Gradient</td>
<td>82</td>
</tr>
<tr>
<td>D.2 Hessian</td>
<td>83</td>
</tr>
<tr>
<td>D.3 Boundary nodes</td>
<td>85</td>
</tr>
<tr>
<td>D.3.1 Call Option</td>
<td>85</td>
</tr>
<tr>
<td>D.3.2 Put Option</td>
<td>85</td>
</tr>
</tbody>
</table>
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The gradient</td>
<td>11</td>
</tr>
<tr>
<td>3.2</td>
<td>The Hessian</td>
<td>11</td>
</tr>
<tr>
<td>6.1</td>
<td>Flowchart of the procedure for finding the EEB.</td>
<td>28</td>
</tr>
<tr>
<td>7.1</td>
<td>Call prices from our solver (blue) and BSM calculations (red) to the left and the differences (black) to the right.</td>
<td>35</td>
</tr>
<tr>
<td>7.2</td>
<td>Put prices from our solver (blue) and BSM calculations (red) to the left and the differences (black) to the right.</td>
<td>35</td>
</tr>
<tr>
<td>7.3</td>
<td>Prices from our solver (blue) and the replicated valuation (red).</td>
<td>37</td>
</tr>
<tr>
<td>7.4</td>
<td>Call option prices for different strikes and different times to maturity.</td>
<td>38</td>
</tr>
<tr>
<td>7.5</td>
<td>Put option prices for different strikes and different times to maturity.</td>
<td>38</td>
</tr>
<tr>
<td>7.6</td>
<td>European call option prices for different strikes and different times to maturity.</td>
<td>39</td>
</tr>
<tr>
<td>7.7</td>
<td>European put option prices for different strikes and different times to maturity.</td>
<td>39</td>
</tr>
<tr>
<td>7.8</td>
<td>The early exercise premium for a call option.</td>
<td>40</td>
</tr>
<tr>
<td>7.9</td>
<td>The early exercise premium for a put option.</td>
<td>40</td>
</tr>
<tr>
<td>7.10</td>
<td>Call price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) without dividends.</td>
<td>41</td>
</tr>
<tr>
<td>7.11</td>
<td>Put price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) without dividends.</td>
<td>42</td>
</tr>
<tr>
<td>7.12</td>
<td>Call price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) with dividends.</td>
<td>42</td>
</tr>
<tr>
<td>7.13</td>
<td>Put price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) with dividends.</td>
<td>43</td>
</tr>
<tr>
<td>7.14</td>
<td>Call price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) without dividends.</td>
<td>44</td>
</tr>
<tr>
<td>7.15</td>
<td>Put price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) without dividends.</td>
<td>44</td>
</tr>
<tr>
<td>7.16</td>
<td>Call price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) with dividends.</td>
<td>45</td>
</tr>
<tr>
<td>7.17</td>
<td>Put price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) with dividends.</td>
<td>45</td>
</tr>
<tr>
<td>7.18</td>
<td>Price convergence for $\theta = 0$ (magenta), 0.5 (red) and 1 (black) with dividends, with $\Delta t$ (cyan) and $\Delta Z$ (green).</td>
<td>46</td>
</tr>
</tbody>
</table>

Hjelmberg, Lagerström, 2014.
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.19</td>
<td>Convergence ratio between $\Delta t$ and $\Delta Z$ for put options</td>
<td>47</td>
</tr>
<tr>
<td>7.20</td>
<td>Convergence ratio between $\Delta t$ and $\Delta Z$ for put options with different volatility</td>
<td>48</td>
</tr>
<tr>
<td>7.21</td>
<td>Prices for options close to maturity for $\theta = 0$ (magenta), 0.5 (red) and 1 (black) without dividends</td>
<td>49</td>
</tr>
<tr>
<td>7.22</td>
<td>Different option prices for different values of $\alpha$</td>
<td>50</td>
</tr>
<tr>
<td>7.23</td>
<td>Different option prices for different values of $\phi$</td>
<td>50</td>
</tr>
<tr>
<td>7.24</td>
<td>Different option prices for different sizes of dividends</td>
<td>51</td>
</tr>
<tr>
<td>7.25</td>
<td>Different EEBs for different changes in variables</td>
<td>51</td>
</tr>
<tr>
<td>7.26</td>
<td>Different EEBs for constant (blue) and random (red) volatility and risk-free rate</td>
<td>52</td>
</tr>
<tr>
<td>7.27</td>
<td>Different EEBs for one small dividend (black), one larger dividend (red) and several small dividends (magenta), compared to no dividends (blue)</td>
<td>53</td>
</tr>
<tr>
<td>7.28</td>
<td>Price and first order derivatives for call options with different strikes with our calculations (blue) and DerivaGem’s (magenta) to the left and the comparison with numerical derivatives to the right</td>
<td>56</td>
</tr>
<tr>
<td>7.29</td>
<td>Price and first order derivatives for put options with different strikes with our calculations (blue) and DerivaGem’s (magenta) to the left and the comparison with numerical derivatives to the right</td>
<td>57</td>
</tr>
<tr>
<td>7.30</td>
<td>Second order derivatives for call options for different strikes with our calculations (blue) and numerical derivatives (red) with an extra Gamma from DerivaGem (magenta)</td>
<td>58</td>
</tr>
<tr>
<td>7.31</td>
<td>Second order derivatives for put options for different strikes with our calculations (blue) and numerical derivatives (red) with an extra Gamma from DerivaGem (magenta)</td>
<td>59</td>
</tr>
<tr>
<td>7.32</td>
<td>Second order derivatives for call options, without dividends, for different strikes with our calculations (blue) and numerical derivatives (red) with an extra Gamma from DerivaGem (magenta)</td>
<td>60</td>
</tr>
</tbody>
</table>
List of Tables

7.1 Four standard options we use when presenting results. . . . . . . 33
7.2 European call option without dividend. . . . . . . . . . . . . . 34
7.3 European put option without dividend. . . . . . . . . . . . . . 34
7.4 European call option with dividend at $T/2$. . . . . . . . . . . . . 34
7.5 European put option with dividend at $T/2$. . . . . . . . . . . . . 35
7.6 Value and derivatives for a call option. . . . . . . . . . . . . . . 36
7.7 Value and derivatives for a put option. . . . . . . . . . . . . . . 36
7.8 Values on an American call option. . . . . . . . . . . . . . . . 36
7.9 Derivatives for Option 4 as a call option. . . . . . . . . . . . . 54
7.10 Derivatives for Option 4 as a put option. . . . . . . . . . . . . 54
7.11 Runtimes of call option programs. . . . . . . . . . . . . . . . 61
7.12 Runtimes of put option programs. . . . . . . . . . . . . . . . 61
Chapter 1

Introduction

The most common option on a specific stock is the American option [Hull 2011]. If one could correctly price these kind of options with a local volatility surface, the local volatility surface for a specific stock could be extracted if the market prices of options on that stock are known. By constructing a program that prices these options, local volatility surfaces of a big range of stocks could be extracted and used for pricing new options on these stocks. In this thesis, we have constructed such a pricing program.

Our work is meant to be used in research at Production Economics within the Department of Management and Engineering at Linköping University. They want to extract the correct local volatility surface for a stock given market prices of corresponding American options.

1.1 Purpose

The purpose is, given known discrete dividends (both time and amount) and a local volatility surface, to use a finite difference method to price and calculate derivatives for American options. We will also investigate the possibility to price several American options simultaneously.

1.2 Delimitations

We will throughout this master thesis assume that the options are on a dividend paying stock, but as long as the underlying asset follows a geometric Brownian motion and the user has access to the local volatility surface for the underlying asset, the model will work. [Black and Scholes 1973]

1.3 Topics covered

Besides this introduction chapter, there are eight additional chapters and four appendices. Main topics dealt with are:

Chapter 2, Method: A brief description of how we have searched for articles to get proofs and theories needed to solve the different problems associated with the purpose of this thesis.
Chapter 3 Theoretical Background: We show the theory the reader should be familiar with in order to understand this thesis, i.e. financial options, solving partial differential equations with a finite difference method, volatility surfaces and the derivatives of an option.

Chapter 4 Literature Study: We present our literature study that has been made in order to solve the various problems associated with the purpose of our master thesis (e.g. how to implement the early exercise premium, how to cope with discrete dividends and how to implement automatic differentiation).

Chapter 5 Forward PDE: This chapter is for interested readers and explains why we were not able to implement a forward PDE for American options. It can be overlooked if one is just interested in our implemented work.

Chapter 6 Implementation: Describes how we have implemented the theory from the background and the literature study and made it into several programs, that prices an option or prices an option and calculates its gradient or prices an option and calculates the gradient and Hessian of the option.

Chapter 7 Results: Here we show how different variables affect the option price, how quickly we can calculate the option price and its gradient and Hessian and we show that the calculated price and derivatives are reasonable. We furthermore show how the price(s) converge or diverge as we change the size of our grid and we also provide an example of an option price surface and its corresponding American premium. Since we do not have a local volatility surface we do not provide any comparison with actual American options available on the market in this Chapter.

Chapter 8 Discussion: We discuss the results that need a more thorough analysis than what has been shown in the Results chapter, e.g. convergence, derivatives and other dependences of different variables.

Chapter 9 Conclusions: Summarizes the thesis and finally draws conclusions of our work. We also discuss how this work could be continued or enhanced.

Appendix A Derivation of the Theta Method: An appendix showing the derivation of the finite difference method that we have used.

Appendix B Interpolation: Different interpolation schemes that has been used in our program or that one could use instead of the ones we have used.

Appendix C Numerical Derivatives: An introduction to numerical derivatives that we have used to ensure our results.

Appendix D AD Variables: Our derived variables that have been used when hard-coding the derivatives, in each time-step, for our quickest AD solution.
Chapter 2

Method

When conducting this thesis, we have mainly used Google Scholar and the website of the library at Linköping University to find professional articles, published in financial or mathematical journals, and other scientific papers, e.g. Doctoral dissertations. We have also used published books within the field. In order to verify the sources we have checked the publishers and that the mathematical proofs are reasonable. There is a possibility that our study has not yielded all relevant research because there could be articles that are not accessible through Google Scholar or the library’s website.
Chapter 3

Theoretical Background

In this chapter we explain the basics behind Black-Scholes-Merton partial differential equation, how to solve it using a grid and finally describe a volatility surface. These are all basics that we need to understand before we can identify and solve the different problems associated with the purpose of our master thesis.

3.1 Options

An option is an agreement between a seller (writer) and a buyer (holder) that lets the buyer of the option choose to exercise the option at a future date for a specified price. In the financial world there are two basic options i.e. the call and the put. The call option lets the holder of the option buy a specific asset for a specified price at a future date from the writer of the option. The put option lets the holder sell a specific asset for a specified price at a future date to the writer. (Hull 2011)

Besides defining an option as just a put or call option, there are many other ways to describe an option. Two of the most common definitions are the European and American options. A European option can only be exercised at the expiration date, i.e. the maturity of the option. An American option lets the buyer of the option exercise the option at any time up until the expiration date. Combining these gives us four different options, i.e. European call (with the price usually referred to as \( c \)), European put (\( p \)), American call (\( C \)) and American put (\( P \)). The most common option on a specific stock is the American option. (Hull 2011)

3.2 BSM PDE

Black and Scholes (1973) and Merton (1973) defined the classical Black-Scholes-Merton (BSM) partial differential equation (PDE) \( \text{(3.1)} \) in 1973. It is the basis for many pricing models where the solution for the European call option is probably the most known. However, since we are going to solve the classic PDE with a finite difference method, we do not want to use the classic solutions, but
instead we want to use the original formula

\[ \frac{\partial V}{\partial t} + rS \frac{\partial V}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} = rV \]  

(3.1)

where \( S \) is the value of the underlying asset, \( V \) is the price of a derivative contingent on \( S \) hence the variable must be some function of \( S \) and \( t \), \( \sigma \) is the volatility of the underlying asset, \( r \) is the risk-free interest rate and \( t \) is the time. \( \text{Black and Scholes (1973, p. 640)} \) dictates that the following assumptions must be fulfilled in order for the formula (3.1) to be valid:

In deriving our formula for the value of an option in terms of the price of the stock, we will assume "ideal conditions" in the market for the stock and for the option:

a) The short-term interest rate is known and is constant through time.

b) The stock price follows a random walk in continuous time with a variance rate proportional to the square of the stock price. Thus the distribution of possible stock prices at the end of any finite interval is lognormal. The variance rate of the return on the stock is constant.

c) The stock pays no dividends or other distributions.

d) The option is "European", that is, it can only be exercised at maturity.

e) There are no transaction costs in buying or selling the stock or the option.

f) It is possible to borrow any fraction of the price of a security to buy it or to hold it, at the short-term interest rate.

g) There are no penalties to short selling. A seller who does not own a security will simply accept the price of the security from a buyer, and will agree to settle with the buyer on some future date by paying him an amount equal to the price of the security on that date.

Since then, some of these limitations have been relaxed and we will show how we can disregard requirement \( a, c \) and \( d \).

### 3.3 Finite Difference Methods

One way to solve the BSM PDE is to use a finite difference method. This is done by creating a grid with the value of the underlying asset on one axis and the time on the other axis. The value of the underlying asset is specified at \( M \) different values with continually higher values. The time axis is divided in a similar way with \( N \) different moments in time. The option value in each node is then referred to as \( f_{i,j} \) where \( 0 \leq i \leq N \) and \( 0 \leq j \leq M \). Note that the node \( f_{i,j} \) corresponding to \((S, t) = (S_0, t_0)\) contains the option value and is called \( V \) in (3.1). The difference in the value of two adjacent nodes on the grid is explained by a difference in \( t (\Delta t) \) or \( S (\Delta S) \). Many articles solving the BSM PDE with a finite difference method assumes that \( \Delta t \) and \( \Delta S \) are the same all over the grid, but as \( \text{Benbow (2005)} \) showed, this is not necessary. \( \text{Hull (2011)} \)
When creating the grid we have to make sure that the time axis is long enough to include the time of valuation and the time of maturity of the option. Similarly we need to make sure that the axis for the value of the underlying assets is long enough for the value of a put option to go from 0 to \(K\), where \(K\) is the strike price of the option. When pricing a call option the axis needs to be long enough for the value of the option to be as low as 0 and allow for high enough values of the underlying asset to make small changes in the underlying asset correspond to equally small changes in the price of the option. (Hull, 2011)

When the calculations are done, the node corresponding to the current value of the underlying asset will have the correct price for the option we are pricing (Hull, 2011). If we do not have an exact node, Haug (1998) shows that we can simply do a linear interpolation between the closest nodes at the current time i.e. if the nodes \(f_{0,k}\) and \(f_{0,k+1}\) are closest to the current value of the underlying asset, the exact value of the option is \(S_0 - S_k \Delta S_{k+1} f_{0,k+1} + S_k \Delta S_k f_{0,k}\) where \(\Delta S_k = S_{k+1} - S_k\). For other interpolation methods see Appendix B.

We then define the boundaries, depending on the kind of option we want to value. If we define the highest value of the underlying asset as \(S_{\text{max}}\) and the lowest as \(S_{\text{min}}\), we can define all nodes on the corresponding borders. For a put option the value for all the nodes corresponding to \(S_{\text{min}}\) is \(K - S_{\text{min}}\) and the nodes corresponding to \(S_{\text{max}}\) is 0. For a call option the nodes corresponding to \(S_{\text{min}}\) is set to 0 and the nodes corresponding to \(S_{\text{max}}\) is set to \(S_{\text{max}} - K\). We then set the values on the line with nodes where \(t = T\), where \(T\) is the expiration date of the option, as \(\max(S - K, 0)\) (for a call) or \(\max(K - S, 0)\) (for a put). (Hull, 2011)

Another way to set up the boundaries is to set the derivative for a call option that is far in the money to 1, and to -1 for a put option, i.e \(\partial V / \partial S = 1\) for a call option and \(\partial V / \partial S = -1\) for a put option that is far in the money and leave the other boundaries as stated above. (Andricopoulos, 2002)

In order to properly price the other nodes there are a couple of different ways to do this, e.g. implicitly, explicitly or a combination of these. Both the implicit and the explicit way is described thoroughly in Hull (2011). Both start at the column before the time of maturity and work their way backwards to \(t = t_0\).

### 3.3.1 Implicit

When calculating the value of the nodes in the grid with the implicit (or fully-implicit (Wilmott et al. 1995)) approach we start by noting that in any node in the grid, \(\partial V / \partial S\) can be approximated to \(f_{i+1,j} - f_{i,j} / \Delta S\) or \(f_{i,j} - f_{i-1,j} / \Delta S\). Hull (2011) uses an average of these to get a more accurate approximation, i.e. \(\partial V / \partial S = f_{i+1,j} - f_{i,j-1} / 2\Delta S\). Furthermore, \(\partial^2 V / \partial S^2\) can be approximated to \(f_{i+1,j} - f_{i,j} / \Delta S\) and \(\partial^2 V / \partial S^2\) to \(f_{i+1,j} + f_{i,j-1} - 2f_{i,j} / \Delta S^2\). Using these in (3.1) Hull (2011) gets

\[
\frac{f_{i+1,j} - f_{i,j}}{\Delta t} + r_j \Delta S f_{i,j+1} - f_{i,j-1} - \frac{\Delta t}{2\Delta S} r_f f_{i,j}^2 + \frac{1}{2} \sigma^2 \Delta S^2 \frac{f_{i,j+1} + f_{i,j-1} - 2f_{i,j}}{\Delta S^2} = r f_{i,j}
\]  

for \(j = 1, 2, ..., M - 1\) and \(i = 0, 1, ..., N - 1\) with the same \(\Delta S\) in the entire grid.
By fixing $i$ and $j$ we get three nodes corresponding to the same value on the time axis and one node on the next higher value on the time axis. Since the entire line of $f_{N,j}$ is known as well as $f_{i,0}$ and $f_{i,M}$, we get $M - 1$ equations with a total of $M - 1$ unknown variables. By solving these equations we get the values for the entire line prior to the time of maturity. We can then repeat this process for each line backwards until we get to the current time, at which time we are done. (Hull, 2011)

### 3.3.2 Explicit

It is also possible to calculate the values in each node in an explicit manner. Hull (2011) does this by noting that if we assume that the values of $\frac{\partial V}{\partial S}$ and $\frac{\partial^2 V}{\partial S^2}$ at a node $(i,j)$ in the grid are the same as at the node $(i + 1,j)$ we get $\frac{\partial V}{\partial S} = \frac{f_{i+1,j+1} - f_{i+1,j-1}}{2\Delta S}$ and $\frac{\partial^2 V}{\partial S^2} = \frac{f_{i+1,j+1} + f_{i+1,j-1} - 2f_{i+1,j}}{\Delta S^2}$. Using this in (3.1) Hull (2011) gets

$$\frac{f_{i+1,j} - f_{i,j}}{\Delta t} + r_j \Delta S f_{i+1,j+1} - f_{i+1,j-1} - \frac{1}{2} \sigma^2 j^2 \Delta S^2 f_{i+1,j+1} + f_{i+1,j-1} - 2f_{i+1,j} = rf_{i,j}$$

(3.3)

for $j = 1, 2, ..., M - 1$ and $i = 0, 1, ..., N - 1$.

For a fixed set of $i$ and $j$ we get a similar expression as in the implicit version, except that we now have the unknown value of one node and the known values of three nodes. In this way we get a much quicker calculation since we only have one equation for each node and we do not need to calculate the values of all the nodes at the same time. (Hull, 2011)

Wilmott et al. (1995) shows that even though the explicit solution is quicker than the implicit version, it is unstable and does not give correct prices for options, unless $\frac{\Delta t}{\Delta S^2} < 0.5$. One way to increase the calculation speed while still keeping the equation stable is to combine the implicit and explicit versions (Wilmott et al., 1995).

### 3.3.3 Crank-Nicolson

Another way to calculate the values for the nodes in the grid is to use a combination of the implicit and the explicit finite difference method. Crank and Nicolson (1947) was the first to do this and the method has been used by many other since then (e.g. Andersen and Brotherton-Ratcliffe (1998), Benbow (2005) and Hull (2011)) and is referred to as the Crank-Nicolson method.

However, there is no need to combine the implicit and explicit solutions equally. Benbow (2005) shows that one can simply use a variable $\theta \in [0, 1]$ to decide how much of each part to use, which White (2013, p. 7) calls the “Theta Method”. By setting $\theta = 0$ we get the explicit version described above, if $\theta = 1$ we get the implicit version and if we mix the two solutions equally by setting $\theta = 1/2$ we get the Crank-Nicolson solution. When using the Crank-Nicolson solution, Wilmott et al. (1995) shows that the error in the calculations goes from $O(\Delta t)$ to $O(\Delta t^2)$. However, the system is only unconditionally stable if $\theta \geq 1/2$ (White, 2013).
3.3.4 Combining the Implicit and Explicit Solutions

In order to enhance the computation of the value of the option we can use $Z = \ln S$ instead of $S$ as the variable for the underlying asset (Hull, 2011) and thus get

$$\frac{\partial V}{\partial t} + \left(r - \frac{\sigma^2}{2}\right) \frac{\partial V}{\partial Z} + \frac{1}{2} \sigma^2 \frac{\partial^2 V}{\partial Z^2} = rV. \quad (3.4)$$

When using $Z$ as the underlying variable and solving the PDE with the Theta Method while not requiring $\Delta Z$ and $\Delta \tau$ to be constant in the entire grid, we get (for full equations see Appendix A)

$$\theta \alpha_j f_{i,j-1} + (\theta \beta_j + 1 - \theta) f_{i,j} + \theta \gamma_j f_{i,j+1} = (1 - \theta) \alpha_j^* f_{i+1,j-1} + (1 - \theta) \beta_j^* f_{i+1,j} + (1 - \theta) \gamma_j^* f_{i+1,j+1} \quad (3.5)$$

where

$$\alpha_j = \frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left(r - \frac{\sigma^2}{2}\right) - \frac{\Delta t_i}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \sigma^2$$

$$\beta_j = 1 + r \Delta t_i \frac{\Delta t_i}{\Delta Z_j \Delta Z_{j-1}} \sigma^2$$

$$\gamma_j = -\frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left(r - \frac{\sigma^2}{2}\right) - \frac{\Delta t_i}{\Delta Z_j(\Delta Z_j + \Delta Z_{j-1})} \sigma^2$$

$$\alpha_j^* = \frac{1}{1 + r \Delta t_i} \left(1 - \frac{\Delta t_i}{\Delta Z_j \Delta Z_{j-1}} \sigma^2\right)$$

$$\beta_j^* = \frac{1}{1 + r \Delta t_i} \left(1 - \frac{\Delta t_i}{\Delta Z_j \Delta Z_{j-1}} \sigma^2\right)$$

$$\gamma_j^* = \frac{1}{1 + r \Delta t_i} \left(\frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left(r - \frac{\sigma^2}{2}\right) + \frac{\Delta t_i}{\Delta Z_j(\Delta Z_j + \Delta Z_{j-1})} \sigma^2\right).$$

If we use the boundaries $\frac{\partial V}{\partial S} = 1$ for a call option and $\frac{\partial V}{\partial S} = -1$ for a put option we get $\frac{\partial V}{\partial Z} = \frac{\partial V}{\partial S} = S$ for a call option and $\frac{\partial V}{\partial Z} = \frac{\partial V}{\partial S} = -S$ for a put option. Using these in the PDE for $\ln Z$ means that we use

$$f_{i,N} = \frac{f_{i+1,N} + rS_{\text{max}} \Delta t_i}{1 + r \Delta t_i} \quad (3.6)$$

for the border nodes at $S_{\text{max}}$ for a call option and

$$f_{i,0} = \frac{f_{i+1,0} - rS_{\text{min}} \Delta t_i}{1 + r \Delta t_i} \quad (3.7)$$

for the border nodes at $S_{\text{min}}$ for a put option.

3.4 Volatility Surface

In the original BSM equation, the volatility is assumed to be constant during the entire valuation. A lot of work over the years have noticed the flaw in this and when creating the implied volatility from market values for options, a volatility smile is observed. By relaxing the volatility in one dimension (time) the smile can be created and used when valuing options. If we also let the
volatility depend on another factor we get a volatility surface which depends on two dimensions. This second dimension is a function of the underlying asset and it can be of many different versions (e.g. $S$, $\ln S$, $S/K$). Because of the computational benefits (Hull, 2011) it is preferred to choose $\ln S$ on this axis. (Andersen and Brotherton-Ratcliffe, 1998)

Two common volatility surfaces are the implied volatility surface and the local volatility surface. The implied BSM volatility surface is a grid of volatilities where the volatility in node $(i, j)$ is the volatility for an option with strike $K = S_j$ and time to maturity $T = t_i$. This volatility can easily be withdrawn from any pricing scheme by investigating which volatility gives the correct value of the option, according to the currently traded option on the market, assuming that the option is actively being traded. A local volatility surface consists of a grid of instantaneous volatilities where the combination of the volatilities between $t_0$ and $T$ is the implied volatility of an option with maturity at time $T$. (Dempster and Richards, 2000)

3.5 Gradient and Hessian

We want to calculate the gradient and Hessian of the price with automatic differentiation (introduced later in Section 4.4). Before we start we need to define the variables that the option price depends on so that we can evaluate the gradient and Hessian properly. When calculating the price of an option, the price always depends on the value ($S$) and the volatility ($\sigma$) of the underlying asset, the risk-free rate ($r$), the strike ($K$), the time of maturity ($T$) and time ($t$). When pricing an option, the date of maturity and the strike are fixed, so there exists four first-level derivatives (i.e. Delta, Vega, Rho and Theta). The second-order derivatives (e.g. Gamma and Vomma) are then just combinations of these giving an addition of 10 unique derivatives, which gives a total of 14 different Greeks. (Hull, 2011)

However, since we want to use a volatility surface we do not have one derivative (i.e. one Vega) of the option for the entire volatility surface. We therefore need to define every small independent change in the volatility surface as its own derivative in order to get the entire gradient and Hessian. The easiest way to define orthogonal directions in a multi-dimensional surface for us is to just define a small change in the volatility of a node as a unique independent direction and then do the same for all nodes. We have the exact same situation for our risk-free rate if we do not want this to be constant during the entire life of the option we are pricing. The risk-free rate only depends on time ($t$) so this does not add as many derivatives as the volatility surface.

If we have a volatility surface of 900 data points (30 times 30) this means that we get a gradient that consists of 932 variables (1 from $S_0$, 1 from $t$, 30 from the risk-free rate and 900 from the volatility surface). This is not a problem and should be easily implemented and calculated. Please observe that if we create a grid for the finite difference method that is limited by $0 \leq i \leq N$ and $0 \leq j \leq M$ the volatility surface only need to span $0 \leq i \leq N-1 = n$ and $1 \leq j \leq M-1 = m$ and the risk-free rate only need to span $0 \leq i \leq N-1 = n$. The reason for this is that the borders of the grid is defined by the boundaries of the PDE and thus does not use their local volatilities, or local risk-free rate, even if we use (3.6) and (3.7). Hence the gradient will consist of 843 variables,
when a grid of size 30 times 30 is used for the finite difference method, which is shown in Figure 3.1.

\[
\begin{pmatrix}
\frac{\partial V}{\partial S} & \frac{\partial V}{\partial t} & \ldots & \frac{\partial V}{\partial \sigma_1} & \ldots & \frac{\partial V}{\partial \sigma_m}
\end{pmatrix}
\]

Figure 3.1: The gradient.

The Hessian, on the other hand, is rather troublesome to calculate if we have \(2 + 29 + 29 \times 28 = 843\) (or its equivalent) independent variables for the option price. This would lead to calculating 355 746 unique values of the Hessian matrix. One way to handle this is to define one specific direction for the risk-free rate and one for the volatility surface. When calculating Rho and Vega we then only have the changes of the option price due to changes along these specific directions. This means that we will only get a Hessian of four times four variables, as seen in Figure 3.2.

\[
\begin{pmatrix}
\frac{\partial^2 V}{\partial S^2} & \frac{\partial^2 V}{\partial S \partial t} & \frac{\partial^2 V}{\partial S \partial r} & \frac{\partial^2 V}{\partial S \partial \sigma} \\
\frac{\partial^2 V}{\partial t \partial S} & \frac{\partial^2 V}{\partial t^2} & \frac{\partial^2 V}{\partial t \partial r} & \frac{\partial^2 V}{\partial t \partial \sigma} \\
\frac{\partial^2 V}{\partial r \partial S} & \frac{\partial^2 V}{\partial r \partial t} & \frac{\partial^2 V}{\partial r^2} & \frac{\partial^2 V}{\partial r \partial \sigma} \\
\frac{\partial^2 V}{\partial \sigma \partial S} & \frac{\partial^2 V}{\partial \sigma \partial t} & \frac{\partial^2 V}{\partial \sigma \partial r} & \frac{\partial^2 V}{\partial \sigma^2}
\end{pmatrix}
\]

Figure 3.2: The Hessian.
Chapter 4

Literature Study

There are some problems that need to be addressed and solved before we can calculate the prices of our options. We will start by addressing the premium concerning the pre-mature exercise of American options. Afterwards we will focus on how discrete dividends from the underlying asset affect the price of the option. At the end of this chapter we will provide an introduction to automatic differentiation.

4.1 Early Exercise Premium

American options are unlike their European counterparts possible to exercise at any time between their issue and their maturity. This means that they could be worth more than their European equivalence. This difference in prices are called the Early Exercise Premium.

Merton (1973) showed that if a stock does not pay any dividends the price of an American call and a European call is the same. Since we are dealing with options on assets with discrete dividends this is not the case. Merton (1973) also showed that the value of an American put usually is higher than the value of a European put, even without dividends. Most of the information below concerns American put options, since the only time that it is beneficial to exercise an American call option early is just before a discrete dividend. This will be discussed more in Section 4.2.

When pricing American options there are several different approaches. One can either calculate the European value of the option and then add the premium for the ability to exercise early, or try to calculate the American value directly. The BSM PDE for American options changes from the European version (3.1) such that it is no longer an equality but an inequality (Kwok 2008), i.e.

\[
\frac{\partial V}{\partial t} + rS\frac{\partial V}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} \geq rV.\tag{4.1}
\]

If we then add the premium \((E_P)\) on the left hand side, we again get an equality (Carr et al. 1992), i.e.

\[
\frac{\partial V}{\partial t} + rS\frac{\partial V}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + E_P = rV.\tag{4.2}
\]
The difference between these approaches is that we either have to solve the inequality of the first equation, or find the value of the premium in the second equation. These might seem as the same problem, but there are different solutions throughout the span of academic articles evaluating American options.

4.1.1 Early Exercise Boundary

When valuing an American put option with a lattice method or an finite difference method, Hull (2011) calculates the price in each node in the European way and then if the value of the option is less than $K - S_t$, where $S_t$ is the value of the stock in the node, the value is increased to $K - S_t$. This is done because if the value of the option is less than $K - S_t$, the option is not worth to keep, thus it will be exercised early. When calculating an option with an implicit finite difference method or a combination of the implicit and explicit methods (e.g. Crank-Nicolson or the Theta Method) one should not change the value of a node after the calculations. If you do this the calculations of the other nodes is no longer correct since we solve a system of equations to get the value of all the nodes at time $t$ at the same time.

However, the BSM PDE (3.1) is true for American put options as long as $S(t) > S^*(t)$, where $S^*(t)$ is the Early Exercise Boundary (EEB). The EEB is the line between nodes that should be exercised early and those that shouldn’t. At the nodes above the EEB for a put, the value of the option is higher than the effect of exercising the option early. This means that it is disadvantageous to exercise the option early. On the EEB the value of the option and the pay-off for exercising the option is the same. When decreasing the value of the underlying asset further and the value of the put option increases we need to make sure that the $\delta = \frac{\partial f}{\partial s} \geq -1$ because that means that as we decrease the value of the asset we might reach a point when it is no longer beneficial to exercise the option. However, this is not a problem since the $|\delta|$ of an option can never be higher than 1 and the $\delta$ of the option is actually -1 below the EEB for a put option. (Sevcovic, 2001) (Chiarella and Ziveyi, 2013) (Rodrigo, 2014)

This means that the EEB is a smooth border across our surface for a put option, as long as the underlying asset do not pay any dividends. However, this is not the case for a call option. We have almost the exact same definition, but since it is only worth to exercise an American option the moment before the value of an underlying asset drops because of a dividend, we do not have a continuous border across the surface. In the moment before a dividend, when it might be beneficial to exercise the option, we do however have the similar case of an EEB where the nodes above and on the EEB represent positions where the option should be exercised early. The $\delta$ of the option above the EEB is of course 1 for a call option. (Rodrigo, 2014)

There has been extensive research on the Early Exercise Boundary (e.g. Ju (1998), Sevcovic (2001), Broadie and Detemple (2004), Detemple (2006), Kwok (2008), Chiarella and Ziveyi (2013) and Rodrigo (2014)) because, once the EEB is calculated it is very easy to price an American option, but finding the entire EEB is very time-consuming and rather inefficient (Ju, 1998). We furthermore want to use a finite difference method to calculate the price so the solution provided by Ju is not so convenient. Whereas the work by Rodrigo (2014) makes the assumption that $\int_{S^*(t)}^{\infty} p_a(x,t)dx \approx \int_{S^*(t)}^{\infty} p_e(x,t)dx$ which is a too big approximation for us since $p_a(S^*(t),t) \neq p_e(S^*(t),t)$, where $p_a$ and $p_e$ is the
4.1. Early Exercise Premium

value of an American and a European put respectively.

Furthermore, the standard BSM PDE is not true when $0 < S(t) < S^*(t)$ (for a put option) so one should try to avoid using the nodes outside the borders. For an American put option we would therefore like to calculate the EEB in each time step and use this as our bottom border instead of $S_{min}$, when calculating the value of a put option. Since the EEB is constantly non-decreasing as $t$ increases (Ehrhardt and Mickens 2008), there will be times when we need to value more nodes than we had in the last step. Furthermore, $S^*(T) = K$ will always be the starting point for us, since we do not have a continuous dividend yield (Carr et al. 1992) (Sevcovic 2001).

Since a call option is only worth to exercise early immediately before a dividend we will calculate the value of the nodes up until the dividend and then set up new boundaries (as described in Section 4.2) and start with a new PDE when calculating the value of an American call option.

4.1.2 Finding the Early Exercise Boundary

To calculate the values in the grid correctly we need to find the EEB in the best way possible. Hull (2011) suggests that one should not care about the EEB and just calculate each node in a European fashion and then compare it with the value of exercising it early and increase the value if it is beneficiary to exercise it early.

Another way of finding the EEB is as shown by Sevcovic (2001). He shows that the EEB for an American call option with continuous dividend ($D$) can be calculated as

$$S^*(\tau) = \frac{rK}{D} + \frac{\sigma^2}{2D} \frac{\partial \Pi}{\partial x}(0, \tau), \quad S^*(0) = \frac{rK}{D},$$

where

$$\tau = T - t, \quad x = \ln \left( \frac{S^*(\tau)}{S} \right), \quad \Pi(x, \tau) = V(S, T - \tau) - S \frac{\partial V}{\partial S}(S, T - \tau).$$

Using the same notation and process, the EEB for an American put option on an underlying asset paying a discrete dividend ($D$) can be calculated as

$$S^*(\tau) = \frac{K}{2} + \frac{\sigma^2}{4r} \frac{\partial \Pi}{\partial x}(0, \tau), \quad S^*(0) = E.$$

Unfortunately, Sevcovic (2001) then continues to calculate the nonlinear integral equation for $S^*(\tau)$ by applying the Fourier sine and cosine integral transforms. Because of the advanced calculations required to do this we have not implemented this method.

When finding the EEB, Ehrhardt and Mickens (2008) showed that the EEB is constantly non-decreasing for larger $t$. Kwok (2008) however, shows that this is only true after the last dividend before maturity. When a discrete dividend occurs, the EEB usually goes directly to 0. When continuing to calculate the values in the grid towards $t_0$ the EEB will soon increase again, back to the level where it would have been if there was no dividend and then follow this level until we get a new dividend, at which time the procedure will repeat again. The time before the EEB increases depends on the risk-free rate, the volatility of the underlying asset and the level of the dividend. If we have a large enough dividend at a time close enough to $t_0$ the EEB will of course never resurface but will stay below our lowest node.
4.1.3 Change of Variables

A common approach when pricing options by solving a PDE is to do some change of variable. Widdicks (2002) does this by defining that $\hat{S}(t) = S(t) - S^*(t)$. This change of variable means that we get a new PDE to solve and that the grid in each time-step goes from $S^*(t)$ to $S_{\text{max}}(t)$ where $S_{\text{max}}(t)$ will be a fixed distance from $S^*(t)$. Widdicks (2002) shows that the PDE that needs to be solved is

$$\frac{\sigma^2}{2}(\hat{S} + S^*)^2 \frac{\partial^2 V}{\partial \hat{S}^2} + r(\hat{S} + S^*) \frac{\partial V}{\partial t} + \frac{\partial V}{\partial \hat{S}} \frac{dS^*(t)}{dt} = rV$$

(4.4)

instead of the ordinary BSM PDE (3.1). Due to the change from $S$ to $\hat{S}$ we will have an unknown grid before the calculations in each step so we will always have to interpolate from our volatility surface. This is no problem when calculating the value of an option from a predetermined volatility surface, but because of the purpose of this program it is not beneficiary to do this or any other change of variables which means that we have to interpolate the volatility from the volatility surface.

The problem with the PDE provided by Widdicks (2002) (4.1.3) is that there is no known value for $S_f$ except at the options maturity, where the EEB is the same as the strike price of the put option. He solves this by adding a small error to each unknown value, in every time step, i.e.

$$f_{m+1}^{i,j} = f_m^{i,j} + \frac{\partial f_i^{j}}{\partial S}$$

and

$$S_{m+1}^f = S_m^f + \frac{\partial S_f}{\partial S}. $$

He starts by defining the initial value of the unknown variables as the known variables in the time-step before, i.e. $f_0^{i,j} = f_{i+1,j}$ and $S_0^f = S_{f+1}$ and then sets up the PDE as a system of equations, using the boundary conditions that $f_{i,M} = 0$, $f_{i,0} = K - S$ and $\frac{\partial f_{i,0}}{\partial S} = -1$ so that he gets $M + 2$ equations for the $M + 2$ unknown $\partial$-variables. When approximating $\frac{\partial f_{i,0}}{\partial S}$ with a finite difference he uses a one-sided difference (i.e. $3f_{i,0} - 4f_{i,1} + f_{i,2} = 2\Delta \hat{S}$) so that he does not need to use any nodes below the EEB (since the nodes on the EEB are the bottom nodes).

After solving the system of equations, Widdicks (2002) then adds the $\partial$-variables to the values currently being calculated and unless all $\partial$-variables are below a certain distance from zero (e.g. $|\partial f_{i,j}| < 10^{-8}$), the calculation is repeated with the new values for the unknown variables at the current time-step until the tolerance level has been met.

4.2 Discrete Dividends

There are many different ways to deal with discrete, non-continuous and non-constant dividends when pricing American options. However, Frishling (2002) clearly states that one should only use a numerical method (e.g. lattices or finite difference method) when pricing options on assets with discrete dividends. Furthermore Haug (1998) lists several different ways of calculating the price by changing the volatility or reducing the price of the asset at time $t_0$ or increasing the strike with a discounted dividend. Unfortunately, all of these methods have been proven to give inadequate results (Haug et al., 2003). Vellekoop and Nieuwenhuis (2006) says that one can use the ordinary BSM equation (which includes a dividend yield) if one simply recalculate the dividends as percentages of the stock price at the dividend dates.
4.2. Discrete Dividends

Hull and White (1990) on the other hand show that one can simply switch which nodes to use when calculating the value for the current nodes. Normally nodes $f_{i,j-1}$, $f_{i,j}$ and $f_{i,j+1}$ are calculated from the nodes $f_{i+1,j-1}$, $f_{i+1,j}$ and $f_{i+1,j+1}$. Hull and White (1990) state that one can instead use nodes $f_{i+1,k-1}$, $f_{i+1,k}$ and $f_{i+1,k+1}$ ($0 \leq k \leq M$) to calculate the value of the nodes $f_{i,j-1}$, $f_{i,j}$ and $f_{i,j+1}$ ($k \neq j$). They do, however, only show this when using an explicit finite difference method, but it should work just as well for an implicit method or for a combination of the two. When using this approach, one must either make sure that each dividend can be described as a sum of one or more $\Delta S$’s used next to each other in the grid or use some interpolation scheme to calculate the values of $f_{i+1,k-1}$, $f_{i+1,k}$ and $f_{i+1,k+1}$ ($k \in \mathbb{R}^+$). See Appendix B for different interpolation schemes. This is practically the same thing as calculating the values just after the dividend, shifting them according to the dividend and then setting up the new values as the boundary for a new PDE that starts just before the dividend as suggested by Andricopoulos (2002).

4.2.1 Dividends Effect on the Assets Value

It is common to expect the value of the underlying asset to drop by the same amount as the dividend. However, history has shown that this is not the actual case. Roll (1977) and Geske (1979) introduces $\alpha$ as a variable to solve this problem. When an asset with value $S_t$ at time $D_t$ gives dividend $D$ the value of the asset will decrease to $S_t - \alpha D$. When using $\alpha$, it is important that one uses a correct value. Geske (1979) states that if one has estimated $\alpha$ incorrectly, when calculating the implied volatility from an option, it might be incorrect. Roll (1977) states that when implementing $\alpha$ it should include the marginal shareholder tax rate (e.g. setting $\alpha = 0.7$ in Sweden). One could of course set $\alpha$ to 1 and instead state the dividend(s) as the value that the underlying asset will decrease when the dividend(s) occurs.

4.2.2 Dividends Bigger than the Assets Value

Since our grids lowest value of the underlying asset ($S_{\text{min}}$) usually goes as low as 0 and our dividends are discrete and above zero this may cause some problems when pricing options. If we do not address the problem we can create a situation where the value of the underlying asset is lower than the current dividend, thus creating a negative value of the asset, directly after the dividend, which is of course impossible. Haug et al. (2003) discusses this matter thoroughly and defines two opposing solutions, i.e. the liquidator or the survivor. When using the liquidator solution you assume that a company whose value is lower than its dividend always pays out its entire value and thus forces the company into default. If you are using the survivor approach you instead assume that if a company’s value is lower than its dividend it will not pay any dividend at all.

You can of course use any of these, including any combination of the above. However, if you use the liquidator policy the dividend will be a continuous function which is beneficial for calculations, even though a survivor policy is more common in the real world (Butyak and Guo 2011).
4.3 Replicate an American Option

Roll (1977) and Geske (1979) both published articles on how an American option can be replicated with other options. They used a portfolio consisting of two European call options and one compound option to replicate an American call option on an underlying asset paying one known dividend. They also present extended methods for options on stocks paying more than one dividend during the life of the option. Later Whaley (1981) pointed out some minor imperfections and proposed corrections for them. The criticism performed was directed at the intermediate payment.

The value of an American call option, with strike $K$ and maturity $T_2$, of a stock with a certain dividend payment $D$ at time $T_1$ is the sum of

- a similar European call option
- plus a European call option with strike $\tilde{S}_{T_1 - \epsilon}$ and maturity $T_1 - \epsilon$ (where $\epsilon = 0^+$)
- minus a European compound call option, with strike $\tilde{S}_{T_1 - \epsilon} + \alpha D - K$ and maturity $T_1 - \epsilon$, on a European call with strike $K$ maturity $T_2$

where $\tilde{S}_t$ is defined as the solution to $c(\tilde{S}_t, T - t; K) = X$, where the value of the function $c(\cdot)$ is the price of a European call option. $\tilde{S}_{T_1 - \epsilon}$ is the solution to $X = \tilde{S}_{T_1 - \epsilon} + \alpha D - K$.

4.3.1 Analytical Solution for Compound Options

A compound option is an option on an option and for that reason it has two maturity dates. At the first maturity $T_1$ the holder can buy a new option using the strike $K_1$, which has the maturity $T_2$ and strike $K_2$. The holder of a call-on-a-call ($\text{c}_c$) compound option will only exercise the first option if the second option is worth more than $K_1$, i.e. $c(\tilde{S}_{T_1}, T_2 - T_1; K_2) > K_1$. (Kwok, 2008)

Under the lognormal assumption of the underlying asset price process, the price formula for a call-on-a-call is given by (Kwok, 2008)

$$c_c(S, t) = SN_2(a_1, b_1; \rho) - K_2e^{-r(T_2 - t)}N_2(a_2, b_2; \rho) - K_1e^{-r(T_1 - t)}N(a_2) \quad (4.5)$$

where $N_2$ is the bivariate standard normal distribution and

$$a_1 = \ln \frac{S}{\tilde{S}_{T_1}} + (r + \frac{\sigma^2}{2})(T_1 - t), \quad a_2 = a_1 - \sigma \sqrt{T_1 - t}$$
$$b_1 = \ln \frac{S}{K_2} + (r + \frac{\sigma^2}{2})(T_2 - t), \quad b_2 = b_1 - \sigma \sqrt{T_2 - t}$$
$$\rho = \sqrt{\frac{T_1 - t}{T_2 - t}}.$$

4.4 Automatic Differentiation

This section will describe the basics of automatic differentiation (AD), which is also known as algorithmic differentiation, computational differentiation and
differentiation of algorithms. AD is a technique to numerically evaluate the derivative of a function with just the function itself specified which works whenever the chain rule holds. Symbolic differentiation (as is easily made by hand in calculus) and divided differences (as presented in previous chapters, e.g. Section 3.3.1) are two well-known other approaches to receive the derivative. The latter, divided differences, is just an approximation often used in numerical analysis. Symbolic differentiation on the other hand performs an exact calculation (not taking computational round-off in consideration), but it requires for the formula to be either specified or generated from an advance algorithm. AD requires less memory and CPU time but still yields an exact answer. An additional advantage of AD is when the gradient and/or Hessian of a function with a substantial number of variables are wanted. [Rall and Corliss, 1996]

AD can be implemented in various ways, but generally source transformation, operator overloading or a combination of the two is used. Source transformation is used to add the capability of AD into an already existing code. It generates new code for the evaluation of derivatives from the current code. The concept with operator overloading on the other hand is that it overloads the basic mathematical operators in order for them to perform some extra calculations, when calculating the function value, so that the wanted derivatives are calculated as well. Some languages do not permit overloading, but this will not be a problem for the solution presented in this master thesis since we will be using MATLAB [MathWorks, 2014]. There are many finished software packages available, both free and purchasable, that can be helpful when implementing AD either by source transformation or operator overloading. Currently there are six AD packages available for MATLAB, these are: ADiGator, ADiMat, ADMAT/ADMIT, INTLAB, MAD and TomSys [autodiff.org, 2014]. [Rall and Corliss, 1996]

It is of course possible to not use any of these methods and instead hard-code calculations to obtain the wanted derivatives. The advantage by doing this is the possibility of using the characteristics and connections associated with a specific calculation scheme. [Rall and Corliss, 1996] clearly explain the possibility to differentiate the function in either forward or reverse mode. Forward and reverse mode are also referred to as tangent linear and adjoint mode respectively in articles and literature. They show that if \( f(x, y) = (xy + \sin x + 4)(3y^2 + 6) \) then \( \nabla f = [(y + \cos x)(3y^2 + 6), 6y(xy + \sin x + 4) + x(3y^2 + 6)] \) with both modes. The forward mode is quite easy to understand and applies the rules of differentiation in sequence with the function value calculations. The reverse mode uses another way to apply the chain rule. First, the values are evaluated and then the dependent (output) variables are differentiated before the intermediate (inner) and independent (input) variables in reverse order of their evaluation. [Rall and Corliss, 1996] claim that the best performance is achieved by mixing the two modes.

Furthermore, [Rall and Corliss, 1996] give some guidelines of when to use which mode and the computational costs. They define \( m \) as the number of independent variables, \( q \) as the number of dependent variables and \( n \) as the total numbers of intermediate variables. The forward mode is generally the mode of choice if \( m < cq \), where \( c \) is some factor greater than 1, at the cost of \( mn \). The reverse mode is generally favoured if \( m \gg q \) with the cost \( \propto n \). [Homescu, 2011] states that the reverse mode is favoured when computing a
large number of Greeks and Neidinger (2010) states that it is more efficient for a large number of independent variables. The conclusion of this tells us that the reverse mode is probably the fastest mode of choice for an implementation such as ours.

Finally, it is stated that forward mode uses less memory since an output value and corresponding gradient can be overwritten following the next step in the calculations. The reverse mode occupies more memory space but is generally faster. For example, a number of output variables may share an intermediate variable and the code can be optimized to take advantage of this fact and reduce the computational time. The present availability of large and fast storage makes the speed of execution preferred and thus it seems that the reverse mode is more suitable. (Rall and Corliss 1996)
Chapter 5

Forward PDE

A majority of all option pricing methods start from the maturity and works backwards in time to provide the price. Since our thesis will be used to extract local volatility surfaces from options prices, it would be very efficient if we could price several options at the same time to reduce run-time. This is possible if a method performing calculations forward in time is used. It is well documented how to do this for European options and we have investigated the possibility of doing this even for American style options. This chapter can be overlooked if one is just interested in our implemented work.

5.1 Dupire Equation

A breakthrough occurred in the beginning of 1994 when [Dupire (1994) and Derman and Kani (1994)] developed the concept of local volatility.  [Derman and Kani (1994)] contributed with discrete time binomial tree models that are consistent with the volatility smile effect.  [Dupire (1994)] published a continuous time theory showing that, under deterministic carrying costs and a diffusion process for the underlying price, no arbitrage implies that European option prices satisfy a certain PDE which differ from the ordinary BSM PDE (3.1). This new PDE is solved forward in time since the independent variables are the options strike ($K$) and time of maturity ($T$), instead of the value of the underlying asset ($S$) and time ($t$). If European option prices can be observed, then the underlying’s local volatility surface can be determined by solving this forward PDE, now commonly called the Dupire equation.  [Hirsa and Neftci 2013]

White (2013) specifies the forward PDE as

$$\frac{\partial C}{\partial T} - \frac{1}{2} \sigma^2(K,T)K^2 \frac{\partial^2 C}{\partial K^2} + (r_T - q_T)K \frac{\partial C}{\partial K} + q_T C = 0 \quad (5.1)$$

where $C$ is the price of the option as a function of $K$ and $T$ which are specified above, $r_T$ is the term-structure of the risk-free rate from $t_0$ to $T$, $q_T$ is the continuous dividend and $\sigma$ is the volatility of the underlying asset. The starting condition is $C(K,0) = \max((-1)^{\text{put}}(S_0 - K), 0)$ where $\text{put}$ is 1 if we price a put option and 0 if we price a call option.

When evaluating options with a correct local volatility surface one can simply pick the prices from the nodes within the grid, after the grid is fully calculated,
corresponding to the specified combinations of strikes and maturities. Since the 
price of the underlying asset and current time are fixed variables in the \((K, T)\) 
space, there exists a solution for every combination of \(S\) and \(t\), e.g. \((S_0, t_0)\) 
corresponds to option prices traded today at the current value of the underlying 
asset.

White \cite{White2013} furthermore defines the PDE if one changes variables so that 
\(x = \ln(K/S_0)\) as

\[
\frac{\partial C}{\partial T} - \frac{1}{2} \sigma^2(x, T) \frac{\partial^2 C}{\partial x^2} + (rT - qT + \frac{1}{2} \sigma^2(x, T)) \frac{\partial C}{\partial K} + qT C = 0.
\]

The starting condition changes correspondingly to 
\(C(x, 0) = S_0 \max((-1)e^x, 0)\).

Unfortunately it does not exist a similar forward equation for American style 
options.

5.2 Partial Integro-Differential Equation

Another way of implementing forward equations is to allow the price to follow a 
jump diffusion process, e.g. a Lévy process. The existence of a jump component 
causes an addition of an integral term to the partial derivatives in both the 
backward and the forward equation. This is done for American options by Carr 
and Hirsa \cite{CarrHirsa2003} who assume that (log) prices jump in order to capture the 
smile, instead of what we want to assume, that the instantaneous volatility is a 
function of stock price and time.

Carr and Hirsa \cite{CarrHirsa2003} presents the forward partial integro-differential equa-
tion (PIDE) solution

\[
-\frac{\partial \Pi(s, t; K, T)}{\partial T} + \frac{\sigma^2 K^2}{2} \frac{\partial^2 \Pi(s, t; K, T)}{\partial K^2} - (r - q)K \frac{\partial \Pi(s, t; K, T)}{\partial K} - q\Pi(s, t; K, T) + \int_{-\infty}^{\infty} \left[ \Pi(s, t; Ke^{-x}, T) - \Pi(s, t; K, T) - \frac{\partial}{\partial K} \Pi(s, t; K, T)Ke^{-x} - 1 \right] e^x \nu(x) dx
\]

\[
= 0 \quad (5.3)
\]

for American options. For a full explanation and declaration of variables, we refer to their article.

5.3 Compound Options

We have now showed why we cannot price American options with a forward equation. But what if we could use forward equations for compound options? Then we could price American options in a forward way using the theory in Section 4.3.

Twenty years after those publications Buraschi and Dumas \cite{BuraschiDumas2001} derived a forward solution for the valuation of compound option prices, for general
diffusion processes with deterministic volatility, with no intermediate payment. They instead require that the option must fall above some hurdle value in order for the option to continue in existence. The forward calculations are made in the spirit of the Dupire equation for European options. Buraschi and Dumas (2001) show great improvements in speed calculating compound option prices with the forward representation, but their result is based on that they already know the value \( S_t \) (introduced in Section 4.3). In order to find \( S_t \), an iterative procedure that calculates a forward or backward European PDE in each step has to be done, for every American option. The reason for this, as introduced in Section 5.1, is because the forward equations are calculated in the \((K,T)\) space but the calculations require us to find the value \( S_t \) of the variable \( S \) which is fixed in this space.

One additional problem is that this is just a way to price American call options. Carr and Chesney (1996) have derived a relationship between the values and exercise boundaries of American call and put options. The solution is a developed version of the classic Put-Call Parity and price options with the same moneyness. They define the moneyness of an American call as the log price relative of the underlying to the strike. The moneyness of the put is analogously defined as the log price relative of the strike to the underlying. However, they impose a symmetry condition to the volatility structure which does not coincide with our choice of using a local volatility surface.

In conclusion, it is possible to price American call options fast (at least on underlying assets with one dividend payment) by calculating European and compound options with the forward representation when the value \( S_t \) is known. Since it is expensive in a computational point of view to find \( S_t \), the gain of the forward representation disappears. Finally, the American put option prices can not be obtained with this method. Although the last reason is not required for extracting local volatility surfaces, it is part of our thesis.
Chapter 6

Implementation

The theory and methods presented in Chapter 3 and 4 was implemented into several programs that prices and calculates derivatives for an American option. This chapter explains how this was made and implemented in MATLAB.

6.1 The Grid

When setting up the grid and the boundaries used for solving our PDE \[3.5\] we are using \( Z = \ln(\frac{S}{S_0}) \) and \( t \) as axes for the grid. The reason for scaling the value-axis is to get as high precision around the current value of the underlying asset as possible. We could of course scale it with \( \frac{1}{S} \) or \( \frac{2}{S+K} \) instead but we found that \( \frac{1}{S_0} \) gave good enough answers. The time-axis could also be changed to \( \tau = T - t \) or a scaling of some kind. It is also possible to use a combination of the two, since we want the possibility of equally high precision during the entire life of the option and we do not want to alter the grid once we start, there is no reason to change the direction or scaling of the time-axis. We have used \( Z \) as the vertical axis and \( t \) as the horizontal axis, but this is not a requirement.

We are also assuming that there exists a volatility surface for the underlying asset of the option that we want to price. We furthermore assume that the size of the volatility surface is large enough to provide the volatility for all nodes in the grid used to solve a PDE, described in Section 3.3. In this case we do not need the volatility of the border nodes so the volatility surface only need to be specified for the \((M - 2) \times (N - 1)\) nodes in the grid excluding all boundary nodes except those corresponding to \( t = t_0 \). The risk-free rate is assumed to be deterministic and known at the start of the calculations and is required to be predefined by the user.

Using these definitions in \[3.5\] gives us

\[
\begin{align*}
\theta \alpha_j f_{i, j-1} + (\theta \beta_j + 1 - \theta) f_{i, j} + \theta \gamma_j f_{i, j+1} &= \frac{1}{2} (\theta \beta_j^2 f_{i, j} + \theta \gamma_j^2 f_{i, j+1}) \\
(1 - \theta) \alpha_j^* f_{i+1, j-1} + ((1 - \theta) \beta_j^* + \theta) f_{i+1, j} + (1 - \theta) \gamma_j^* f_{i+1, j+1} &= \frac{1}{2} (1 - \theta) \beta_j^* f_{i+1, j} + (1 - \theta) \gamma_j^* f_{i+1, j+1}
\end{align*}
\]

(6.1)

where \( \alpha_j = \frac{\Delta t_i}{\Delta Z_j} \left( r_i - \frac{\sigma_{i,j}^2}{2} \right) - \frac{\Delta t_i}{\Delta Z_{j-1}} \left( r_i - \frac{\sigma_{i,j}^2}{2} \right) - \frac{\Delta t_i}{\Delta Z_{j-1}} \left( r_i - \frac{\sigma_{i,j}^2}{2} \right) \).
\[ \beta_j = 1 + r_i \Delta t_i + \frac{\Delta t_i}{\Delta Z_j \Delta Z_{j-1}} \sigma_{i,j}^2 \]
\[ \gamma_j = -\frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r_i - \frac{\sigma_{i,j}^2}{2} \right) - \frac{\Delta t_i}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \sigma_{i,j}^2 \]
\[ \alpha_j^* = \frac{1}{1 + r_i \Delta t_i} \left( -\frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r_i - \frac{\sigma_{i,j}^2}{2} \right) + \frac{\Delta t_i}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \sigma_{i,j}^2 \right) \]
\[ \beta_j^* = \frac{1}{1 + r_i \Delta t_i} \left( 1 - \frac{\Delta t_i}{\Delta Z_j \Delta Z_{j-1}} \sigma_{i,j}^2 \right) \]
\[ \gamma_j^* = \frac{1}{1 + r_i \Delta t_i} \left( \frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r_i - \frac{\sigma_{i,j}^2}{2} \right) + \frac{\Delta t_i}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \sigma_{i,j}^2 \right) \]

For the calculations in the border nodes (3.6) and (3.7) changes to
\[ f_{i,N} = \frac{f_{i+1,N} + r_i Z_{\text{max}} \Delta t_i}{1 + r_i \Delta t_i} \]
for the border nodes at \( S_{\text{max}} \) for a call option and
\[ f_{i,0} = \frac{f_{i+1,0} - r_i Z_{\text{min}} \Delta t_i}{1 + r_i \Delta t_i} \]
for the border nodes at \( S_{\text{min}} \) for a put option.

### 6.2 Pricing of American Options

When calculating the price of an American option in our model we start by defining the borders as described in Section 3.3 and then, if it is a put option, find the EEB. Once the EEB has been found the nodes below the EEB are set to \( K - S \), the node just above the EEB is calculated explicitly and then all the nodes above the EEB will be calculated with the Theta Method as described in Section 3.3.4. If we are pricing a call option there exists no EEB so all the nodes are priced with the Theta Method. Before we continue with the next time-step, we check if a dividend is reached. If this is the case we are moving each node up in the grid corresponding to the reduction of the stock price because of the dividend. If the user has failed to define the grid size so that each dividend occurs at the exact time of a node column (which is rather difficult due to the calculation limitations in MATLAB) we are checking if we have passed the time of a dividend for a call option or if we will pass the time of a dividend in the next time-step for a put option. The reason for this difference is because a put option is usually not worth to exercise immediately before a dividend whereas a call option is only worth to exercise immediately before a dividend.

For the nodes corresponding to very low levels of the value of the underlying asset, we are using a variable \( \phi \in [0, 1] \) that the user will have to specify when running the program. \( \phi \) is the maximum percentage of the current asset price that the price will be allowed to drop. By setting \( \phi \) to 0 or 1, the user can choose to use the survivor or the liquidator tactics as described in Section 4.2.2. If we are pricing a call option we are checking each node after every dividend to see if any of the nodes should be exercised early. If this is the case the value of the node(s) is changed to \( S - K \).
6.3 Finding the EEB

As we described in Section 4.1.1 there are several different ways to calculate the EEB, but most of them are rather cumbersome. One easy way when doing an explicit finite difference calculation is to perform the calculations as if the option was European and then change the nodes that should be exercised early. Hull (2011) also suggests that this could be done for implicit calculations as well, but as stated before, this is not an optimal solution. However, if we make sure that our calculations converge for an explicit solution as well and not just for implicit or Crank-Nicolson calculations, we should be able to price some nodes explicitly and still get a correct value for the option. We will in Chapter 7 show that we do not have the standard explicit convergence requirement when doing this, but there is still a risk that the option price fail to converge if we are reckless. In order to have as good of a limit as possible we will price as few nodes in each time-step as possible in a fully explicit manner, i.e. one node, and to be specific, the node just above the EEB.

To find the EEB we start by assuming that the EEB is at the same level as at the earlier time-step. At the first time-step we start at the level of the strike. Initially the EEB is non-increasing, but as soon as we have encountered a dividend the EEB might be both increasing and decreasing. In order to have just one method of finding the EEB we look for both increasing and decreasing EEB in each time-step. As a matter of fact, when having a randomized local volatility surface, the EEB could be both decreasing and increasing right from the start, as we will show in Chapter 7.4.1. Nevertheless, we do this by calculating the value of the node just above the EEB explicitly. If this calculation shows that the option is worth less than the value of exercising the option at this moment we know that we are currently below the EEB. We then set a variable (or a flag) called \texttt{EEBdirection} to $+1$, increase the EEB and check if that node should be exercised early as well. We continue doing this until we find a node that should not be exercised early and then we know that the EEB is somewhere between the last two calculated nodes. If the first calculation shows that the node just above the EEB of the last time-step should not be exercised early we know that the EEB is decreasing. We then set the \texttt{EEBdirection} variable to $-1$ and decrease the EEB until we reach a node that should be exercised early and then we know that the EEB is between these last two nodes. See Figure 6.1 for the corresponding flowchart. In the flowchart the EEB position corresponds to the node just above the EEB.

Since the EEB usually drops to zero after a dividend we check if the node corresponding to $S_{\text{min}}$ should be exercised early after each dividend. If this is not the case we change the EEB to 0 and otherwise we will leave it at the previous level and let the procedure above find the EEB.
6.4 Gradient and Hessian of Option Prices

When implementing automatic differentiation in order to obtain the gradient and Hessian (described in Section 3.5) of the calculated option price, we started by implementing a object-oriented method described by Neidinger (2010). He has defined a MATLAB class of value-and-derivative objects, called valder objects, and then overloaded the definitions of standard operations and functions (see Section 4.4 for introduction to operator overloading) in order to handle these objects. Neidinger (2010) share a m-file where the class definition and corresponding operations are already written. The reason why we chose to use operator overloading over source transformation was because we wanted control over the code and did not want automatically generated code. For further explanations on how we handle the valder object and a complete description of all the programs see Section 6.5.

We quickly found that when solving a system of equations and/or when using vector multiplications the valder object is not a powerful tool to use. Neidinger (2010) also says that using valder objects can be very inefficient, especially when calculating the Hessian. Therefore, we have also made a hard-coded solution, which was both easier and quicker to use, that calculates the gradient and Hessian of the option price. When doing this we have used the product rule on...
Our Programs

We have created fourteen different programs for calculating the option price for American options and two programs for calculating the option price for European options. These are two different versions for calculating the option price for a call option (PDESolverCall1 and PDESolverCall2), three different versions for calculating the option price and gradient for a call option (PDESolverCallADGrad1, PDESolverCallADGrad2 and PDESolverCallADGrad3) and two
different versions for calculating the option price, gradient and Hessian of a call option (PDESolverCallADHess1 and PDESolverCallADHess2). The exact same set of programs also exist for put options. The differences between all these programs will be described below.

6.5.1 Option Prices

The six programs that only calculates the option price of an option are three programs for call options and three for put options. Of these there is one each for European options (PDESolverCallEuro and PDESolverPutEuro). These are quite straightforward and calculates the prices from the nodes at time $T$ until time $t_0$ where the option price is interpolated from the nodes corresponding to the $S$-values surrounding $S_0$. The other programs work in similar ways, except that they allow for early exercise (as described in Section 4.2 for call options and in Section 6.3 for put options). The difference between the programs is that the first versions (PDESolverPut1 and PDESolverCall1) calculates all $\alpha$, $\beta$, $\gamma$, $\alpha^*$, $\beta^*$ and $\gamma^*$ variables for the entire life-span of the option first and then starts the iterative process that prices the nodes in each time step. The tri-diagonal equation system that needs to be solved in each iteration is created as sparse matrix and then solved by MATLAB’s internal equation solver. The second versions (PDESolverCall2 and PDESolverPut2) only calculates some help variables that are the same for each time-step before starting the iterative process. In each time-step the exact value for $\alpha$, $\beta$, $\gamma$, $\alpha^*$, $\beta^*$ and $\gamma^*$ are then calculated and the system of equations is solved by our own equation solver. The solver takes the three vectors $\alpha$, $\beta$ and $\gamma$ instead of the tri-diagonal matrix and then solves the equations system, knowing that we have a tri-diagonal matrix, even though we never actually create it.

6.5.2 Gradients

The six programs that calculates the option price and the full gradient of an option are three programs for call options and three for put options. A problem when using objects with overloaded functions is that we cannot use vector calculations (which is one of the great benefits with MATLAB). We have therefore created two different versions for this approach. We first use the basic valder object (as described in Section 6.4). Since all calculations are made with vectors we have to use loops (or some other iterative process) to work our way through each variable, both when defining the variable and when doing any calculations with it. This solution has been implemented in our first versions of the programs (PDESolverCallADGrad1 and PDESolverPutADGrad1).

One way to increase the calculating speed of this is to change the valder object so that we can have vectors of variables inside the object, instead of just one variable. This means that we have a vector for the object’s values and a matrix for the object’s derivatives (since we have more than one derivative for each value). We can then do vector calculations for the objects, but if we want to use only part of the vector we need to create a new object from parts of the old vector, which is rather cumbersome. This solution has been implemented in our second versions (PDESolverCallADGrad2 and PDESolverPutADGrad2).

In the third versions of our program (PDESolverCallADGrad3 and PDESolverPutADGrad3) we have instead hard-coded a solution to (6.2). This creates
a much bigger program, but it reduces the runtime significantly.

Another way to increase computational speed of version one and two would be to alter the valder object even further so that it could do complete vector operations, but since we only use these versions to check that we get the correct answers from our hard-coded versions, we have not seen any reason to do this.

### 6.5.3 Hessians

To calculate the Hessian for our option we have created four programs. They all require the user to define directions for the risk-free rate and the volatility surface for which the derivatives will be calculated. The first versions of the Hessian program *(PDESolverCallADHess1 and PDESolverPutADHess1)* uses a combination of the first and second version of the gradient programs. We use the alternative version of the valder object that we use in the second version, but we use the iterative loop code from the first version. The reason for this is that when using objects with overloaded operators for calculating derivatives we create an object consisting of two objects instead of a value and a vector or a vector and a matrix. The first object is then a value and a vector, while the second object is a vector and a matrix. The second objects vector is the same as the first object’s vector as these are the first order derivative of the value.

The other versions *(PDESolverCallADHess2 and PDESolverPutADHess2)* that calculates the Hessian (as well as the gradient and option price, of course) uses the hard-coded solution to (6.3), analogously to the solution for the gradient. This is a much quicker program than any of the others since we only have four first order and ten second order derivatives to calculate, due to the requirement that the user specifies which directions the derivative should be calculated for. On the other hand these programs are much bigger than the first versions (actually more than three times as big).

### 6.6 Other Programs

Beside the sixteen programs described above, we have also created some other programs used to affirm our calculations. These programs are briefly described below.

#### 6.6.1 Implementation with a Change of Variable

In Section 4.1.3 we discussed Widdicks (2002) approach with a change of variable *(\( \hat{S}(t) = S(t) - S^*(t) \)). We have implemented this solution but have been unable to replicate his results. The problem has been that when defining that \( \frac{\partial}{\partial S} = -1 \) on the EEB, we force the EEB to be much higher than it actually is. When we then decrease the value of \( \Delta S \) for the nodes just above the EEB we can decrease the level of the EEB, and for very small values for \( \Delta S \) we get really low levels for the EEB. This should of course not be the case, but no matter where the EEB ends up, way always get too low values for the options we are pricing. When implementing this program we have used \( S \) and \( t \) on the axes and we have only tried it for underlying assets without dividend payments.
6.6.2 Compound Option

To be able to validate that our programs pricing American call options are accurate, we made a simple program that uses the theory in Section 4.3. Since this was just made in order to validate the result, we did not implement a solution for the compound option that used a local volatility surface. Instead we implemented the analytical solution, with constant volatility and risk-free rate, for the call-on-a-call option that we needed.
Chapter 7

Results

During all of our simulations we have used one or several of the options below, sometimes with some smaller alternations, which will then be mentioned specifically in each respective chapter. The first and last option has a random risk-free rate and volatility surface with upper and lower limits stated in the table. The random variables are created by MATLAB’s standard function that generates numbers from a uniform distribution of the entire interval.

<table>
<thead>
<tr>
<th>Option</th>
<th>Set 1</th>
<th>Set 2</th>
<th>Set 3</th>
<th>Set 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_0$</td>
<td>90</td>
<td>50</td>
<td>90</td>
<td>115</td>
</tr>
<tr>
<td>$K$</td>
<td>100</td>
<td>50</td>
<td>100</td>
<td>108</td>
</tr>
<tr>
<td>$r$</td>
<td>[1.5%, 3%]</td>
<td>2%</td>
<td>2.25%</td>
<td>[2%, 4%]</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>[15%, 40%]</td>
<td>30%</td>
<td>25%</td>
<td>[30%, 40%]</td>
</tr>
<tr>
<td>$T$</td>
<td>1.5</td>
<td>0.9</td>
<td>1.5</td>
<td>0.5</td>
</tr>
<tr>
<td>$\theta$</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.7</td>
<td>0.7</td>
<td>0.7</td>
<td>0.7</td>
</tr>
<tr>
<td>$\phi$</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>$M$</td>
<td>1000</td>
<td>300</td>
<td>1000</td>
<td>100</td>
</tr>
<tr>
<td>$N$</td>
<td>1000</td>
<td>300</td>
<td>1000</td>
<td>100</td>
</tr>
<tr>
<td>$D_1$</td>
<td>5</td>
<td>2.5</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>$D_2$</td>
<td>4</td>
<td>-</td>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>$D_{t1}$</td>
<td>0.25</td>
<td>0.45</td>
<td>0.25</td>
<td>0.4</td>
</tr>
<tr>
<td>$D_{t2}$</td>
<td>1.25</td>
<td>-</td>
<td>1.25</td>
<td>-</td>
</tr>
<tr>
<td>$Z_{min}$</td>
<td>-5</td>
<td>-3</td>
<td>-5</td>
<td>-3</td>
</tr>
<tr>
<td>$Z_{max}$</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>$\Delta Z$</td>
<td>0.01</td>
<td>0.02</td>
<td>0.01</td>
<td>0.06</td>
</tr>
<tr>
<td>$\Delta t$</td>
<td>0.0015</td>
<td>0.003</td>
<td>0.0015</td>
<td>0.005</td>
</tr>
</tbody>
</table>

Table 7.1: Four standard options we use when presenting results.

We have chosen these options to get good diversity for our tests, while still keeping some values, e.g. $\alpha$, constant to properly mimic the real world. These
options can be used as both American and European versions but unless specified specifically we will refer to American options for the remainder of this chapter.

7.1 Validation

In this section we compare our programs with other methods of pricing options. Our solvers are then compared with each method and we discuss the eventual differences. The differences stated in every table are defined as the absolute value of how much our PDE solver differ from the compared value, measured in percent.

7.1.1 European Prices

First we compare our solvers for European options with the classical BSM solutions. Because the European options are unable to be exercised early, they do not have any early exercise premium and therefore are rather simple to price.

No Dividends

Table 7.2 and Table 7.3 shows valuation of Option 2 with no dividend for a call and a put option respectively.

<table>
<thead>
<tr>
<th>BSM</th>
<th>PDE Solver</th>
<th>Difference (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.06458980</td>
<td>6.06105121</td>
<td>0.058348</td>
</tr>
</tbody>
</table>

Table 7.2: European call option without dividend.

<table>
<thead>
<tr>
<th>BSM</th>
<th>PDE Solver</th>
<th>Difference (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.17264142</td>
<td>5.16913685</td>
<td>0.067752</td>
</tr>
</tbody>
</table>

Table 7.3: European put option without dividend.

One Dividend

The usual way of taking care of dividends with both the BSM formula and binomial trees is either to discount the dividend to $t_0$ and subtract that value from $S_0$ (low estimate) or compound the dividend to $T$ and add that value to $K$ (high estimate). Table 7.4 and Table 7.5 shows such valuations with the BSM formula for Option 2, but with $\alpha = 1$ together with the mean of these estimates.

<table>
<thead>
<tr>
<th>Decrease $S_0$</th>
<th>Increase $K$</th>
<th>Mean</th>
<th>PDE Solver</th>
<th>Difference (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.71054051</td>
<td>4.98914722</td>
<td>4.84984387</td>
<td>4.84937276</td>
<td>0.009713</td>
</tr>
</tbody>
</table>

Table 7.4: European call option with dividend at $T/2$. 
7.1. Validation

<table>
<thead>
<tr>
<th>Decrease $S_0$</th>
<th>Increase K</th>
<th>Mean</th>
<th>PDE Solver</th>
<th>Difference (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.29619307</td>
<td>6.57479979</td>
<td>6.43549643</td>
<td>6.43590938</td>
<td>0.006417</td>
</tr>
</tbody>
</table>

Table 7.5: European put option with dividend at $T/2$.

We found these exact results fascinating so we made more calculations for this theory. In the calculations presented in Figure 7.1 and Figure 7.2 the dividend has been moved from $D_t = t^0$ to $D_t = T^-$, with $T^-$ being the time just before maturity. Instead of using the mean value, the price was calculated as

$$V = \frac{T - D_t}{T} V_{DecreaseS} + \frac{D_t}{T} V_{IncreaseK}.$$ 

The differences in the figures are the absolute value differences of the option prices.

Figure 7.1: Call prices from our solver (blue) and BSM calculations (red) to the left and the differences (black) to the right.

Figure 7.2: Put prices from our solver (blue) and BSM calculations (red) to the left and the differences (black) to the right.
7.1.2 Comparison with DerivaGem

The software DerivaGem, included in the book by Hull (2011), can compute option prices and some derivatives with several different basic methods. Using binomial trees with 500 time-steps in DerivaGem and Option 2 without the dividend gives the results presented in Table 7.6 and Table 7.7. Since DerivaGem only provides some derivatives these are the only ones we can compare. A more thorough analysis of the derivatives can be found in Section 7.5.

<table>
<thead>
<tr>
<th>Price/Greek</th>
<th>DerivaGem</th>
<th>PDE Solver</th>
<th>Difference (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>6.06178144</td>
<td>6.06105121</td>
<td>0.012046</td>
</tr>
<tr>
<td>Delta</td>
<td>0.58138772</td>
<td>0.58138976</td>
<td>0.000351</td>
</tr>
<tr>
<td>Gamma</td>
<td>0.02748940</td>
<td>0.02747054</td>
<td>0.068608</td>
</tr>
<tr>
<td>Theta</td>
<td>-3.55299342</td>
<td>-3.54994504</td>
<td>0.085798</td>
</tr>
<tr>
<td>Rho</td>
<td>20.89027495</td>
<td>20.70522833</td>
<td>0.865245</td>
</tr>
<tr>
<td>Vega</td>
<td>18.51354426</td>
<td>18.53840052</td>
<td>0.134260</td>
</tr>
</tbody>
</table>

Table 7.6: Value and derivatives for a call option.

<table>
<thead>
<tr>
<th>Price/Greek</th>
<th>DerivaGem</th>
<th>PDE Solver</th>
<th>Difference (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>5.24639897</td>
<td>5.24556970</td>
<td>0.015806</td>
</tr>
<tr>
<td>Delta</td>
<td>-0.42739609</td>
<td>-0.42731927</td>
<td>0.017974</td>
</tr>
<tr>
<td>Gamma</td>
<td>0.02848078</td>
<td>0.02845833</td>
<td>0.078825</td>
</tr>
<tr>
<td>Theta</td>
<td>-2.67202131</td>
<td>-2.66864100</td>
<td>0.126508</td>
</tr>
<tr>
<td>Rho</td>
<td>-18.28016916</td>
<td>-18.88425940</td>
<td>3.304621</td>
</tr>
<tr>
<td>Vega</td>
<td>18.51179732</td>
<td>18.55442532</td>
<td>0.230167</td>
</tr>
</tbody>
</table>

Table 7.7: Value and derivatives for a put option.

7.1.3 Replicate an American Call Option

In Table 7.8 the option prices for the replicated American option and our solver, using Option 2, are presented. Because we want to highlight the dependence of the size of the dividend, we present the differences in price between our solver and the replicated American options in Figure 7.3. The figure clearly shows that the replicated American options are mispriced for larger dividends, but gives satisfactory results for the original dividend size of Option 2. Further test have shown, that when the dividend becomes larger in relation to the limit $S_{T_1-\epsilon}$, the mispricing becomes even worse.

<table>
<thead>
<tr>
<th>Replicated</th>
<th>PDE Solver</th>
<th>Difference (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.34048023</td>
<td>5.30956497</td>
<td>0.578886</td>
</tr>
</tbody>
</table>

Table 7.8: Values on an American call option.
7.2 Option Prices And Premiums

One of the purposes of this master thesis is to price several options at the same time. As we described in Chapter 5 we are unable to do this by just solving one PDE. We can however run our program several times and thus create an option price surface. When doing this we have used Option 1 so that we have variable risk-free rate and volatility. We have priced options with a strike from 2 to 200 and time to maturity from 0.0015 to 1.5 so that we have a minimum of 10 time-steps up to the maximum of 1000 time-steps as specified in Option 1. In Figure 7.4 we have call option prices and in Figure 7.5 we have put option prices and please observe that we have time to maturity on one axis and the strike on the other axis.

As we can see there are no visible jumps for the call option prices around the dividends. This is because the call options that would be reduced in price due to the dividends are instead exercised early so that we do not loose any value because of the dividend. For the put option on the other hand, we can clearly see a jump at the first dividend and a small jump at the second dividend for options with strikes just above $S_0$. The reason for this is of course that the options who matures before the dividends do not benefit from the drop in the underlying asset.

We have also examined the premium for the American options, compared to their European counterpart (as described in Section 4.1). We have done this by calculating the prices for the European counterparts of the options above. These option price surfaces can be seen in Figure 7.6 (call option) and 7.7 (put option). As we can see in Figure 7.8 (call option) and 7.9 (put option) we have an increase of the premium for options in the money with significant jumps at each dividend. We can also see that the premium for the call option reaches some sort of a platform for very low strikes where the premium is very close to the discounted value of the dividend(s).
Figure 7.4: Call option prices for different strikes and different times to maturity.

Figure 7.5: Put option prices for different strikes and different times to maturity.
Figure 7.6: European call option prices for different strikes and different times to maturity.

Figure 7.7: European put option prices for different strikes and different times to maturity.
Figure 7.8: The early exercise premium for a call option.

Figure 7.9: The early exercise premium for a put option.
7.3 Convergence

An important aspect when pricing options with the Theta Method is of course the value of $\theta$ and the convergence or divergence of the option price. We have examined this thoroughly and will present the results on the following pages. We will in the first section show how the convergence changes with different values on $\theta$ and then make a more careful convergence analysis for $\theta = 0.5$ in the following section. Finally, we will investigate the dependency of $\theta$ for valuations of options close to maturity.

7.3.1 Different Thetas

When changing $\theta$ we can see how the option price converges and/or diverges as we change $\Delta Z$ or $\Delta t$. The figures on the following pages have been made with Option 3, with the difference that the time axis has been divided into different numbers of time-steps (altered $\Delta t$) in Figure 7.10, 7.11, 7.12 and 7.13 and variable length of the $Z$ axis (altered $\Delta Z$) in Figure 7.14, 7.15, 7.16 and 7.17. In Figure 7.10, 7.11, 7.14 and 7.15 there has been no dividend, but in the other figures we have used the ordinary dividends for Option 3.

Figure 7.10: Call price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) without dividends.

We can clearly see, in Figure 7.10, that without dividends the price for the call option converges rather quickly as soon as we have $\theta \geq 0.5$, but for lower $\theta$ we need a larger grid, and for the fully explicit solution we almost need a grid that is as wide as it is high. For the put option, in Figure 7.11 we need a minimum of 500-600 time-steps ($N$) to ensure convergence, when we do not
Figure 7.11: Put price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) without dividends.

Figure 7.12: Call price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) with dividends.
have any dividends, unless we have a fully explicit solution, in which case we need about the same amount of time-steps as for the fully explicit solution for the call option. We also observe that the option price never goes below 10 for the put option, which means that the EEB is above $S_0$ and the option should be exercised early, which is of course incorrect.

If we include the dividends for Option 3, we see that the option price never quite converges. This is because the percentage of nodes on each side of the dividends changes as we add more and more nodes. However, for the call option in Figure 7.12 we can see that the prices start to converge quickly, except for the fully explicit solution that requires a minimum of around 900 time-steps before it starts to converge close to the other nodes. For the put option we have a similar case except that we can see a clear mispricing for all values of $\theta$ in Figure 7.13, unless we have a minimum of around 600 time-steps and even more for $\theta$ below 0.5. Due to the dividends the EEB is below the value of $S_{min}$ at $t_0$ so we have no case of exercising the option immediately.

When we instead change the number of nodes on the Z-axis ($M$) and keep the number of nodes on the time-axis constant, we can see that all values of $\theta$ for the call option, in Figure 7.14, follows each other initially. If we have a $\theta$ below 0.5 the calculation starts to diverge as $\Delta Z$ gets smaller and for a fully explicit solution it diverges when $\Delta Z \approx 0.0015$. For a put option we have a similar case except that all values of $\theta$ diverge as we approach $\Delta Z = 0.001$, as shown in Figure 7.15. The reason for this is because we have an explicit calculation for the EEB and when this calculation diverges we soon reach a position when all nodes should be exercised as early as possible, due to miscalculations of the
Chapter 7. Results

Figure 7.14: Call price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) without dividends.

Figure 7.15: Put price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) without dividends.
7.3. Convergence

Figure 7.16: Call price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) with dividends.

Figure 7.17: Put price convergence for $\theta = 0$ (magenta), 0.25 (blue), 0.5 (red), 0.75 (green) and 1 (black) with dividends.
If we include dividends there is almost no difference for either the call or put option, as can be seen in Figure 7.16 and Figure 7.17. The call option starts to diverge when $\Delta Z$ gets too small if $\theta$ is below 0.5, and all values of $\theta$ leads to miscalculations for the put option and as for the changes in $\Delta t$ the EEB is below $S_0$ due to the dividends so there is no case of immediate exercise of the option. We do however see an increase of the option value for $\theta \geq 0.5$ and as we decrease $\Delta Z$ even further these option prices starts to oscillate even more.

We can clearly see from all the diagrams that we can only guarantee convergence for call options if we have $\theta \geq 0.5$. On the other hand, in this circumstance we can guarantee convergence for all values of $\Delta t$ and $\Delta Z$. For lower values of $\theta$ we need to make sure that we have a ration between $\Delta t$ and $\Delta Z$ that allow for convergence. For a put option we always have a minimum of one fully explicit calculation in each time step so in this case we can never be guaranteed convergence without knowledge of $\Delta t$ and $\Delta Z$ (see Section 7.3.2 for more on this topic).

![Figure 7.18: Price convergence for $\theta = 0$ (magenta), 0.5 (red) and 1 (black) with dividends, with $\Delta t$ (cyan) and $\Delta Z$ (green).](image)

Nevertheless, we can define clearly for both call and put options, that the error is $O(\Delta t, \Delta Z)$. In Section 3.3.3 we observed that the error should go from $O(\Delta t)$ to $O(\Delta t^2)$ when using $\theta = 0.5$ instead of $\theta = 0$ or 1, but as shown in Figure 7.18, the option prices for all values of $\theta$ are rather close to each other and the ratio between $\Delta t$ and $\Delta Z$ is much more important. In the diagram we go from a grid of 10 times 25 nodes to a grid of 2000 times 5000 nodes where the larger amount of nodes are on the time-axis, to ensure that we get a low
7.3. Convergence

enough value for $\Delta t$. We have the diagram for a call option at the top and the put option at the bottom in the figure below.

7.3.2 Critical Ratios

As was shown in Section 7.3.1 the put option diverges for large $\Delta t$ or small $\Delta Z$ even if $\theta \geq 0.5$. We have therefore examined the special case when $\theta = 0.5$ and found the area where the price converges and when the sizes of $\Delta t$ and $\Delta Z$ makes the EEB increase even when we have no dividends. If the underlying has dividends we showed in Section 7.3.1 that the calculations diverge at the same time but in different ways, which does not matter since we do not want them to diverge at all. The convergence requirement for an explicit finite difference method was discussed in Section 3.3.2 and in order for the price to converge the requirement is that $\frac{\Delta t}{\Delta Z} < 0.5$.

![Figure 7.19: Convergence ratio between $\Delta t$ and $\Delta Z$ for put options.](image)

We do not have the same convergence requirement for our Crank-Nicolson since we only make one explicit calculation in each time-step (that we keep), but we can still use the requirement to see that we need to keep $\Delta t$ low and/or $\Delta Z$ high in order to get convergence of the option price. This means that we need to be in the area above the line in the top diagram and below the line in the bottom diagram in Figure 7.19. These diagrams have been created from Option 3 without dividends and with the same grid size the whole time. This
means that we have changed the length of the option when changing $\Delta t$. The
diagrams are meant to give the reader a feel for the ratio that is required in
order to get convergence for the put option price rather than to give the exact
convergence ratio between $\Delta Z$ and $\Delta t$ for all type of options.

The diagrams was generated by approaching the critical limit with small
changes in the vertical direction for a specific value on the horizontal axis. This
means that the limit was approached from above in the upper diagram and from
beneath in the lower diagram.

Furthermore, we have noticed that when changing $S_0$, $K$ and $r$, the con-
vergence ratio does not change much. The differences in the convergence ratio
are not correlated to the changes in these three variables, but rather oscillating
around the first result with a magnitude of approximately $10^{-4}$. Changes in
the (constant) local volatility surface on the other hand clearly influences the
convergence ratio, as shown in Figure 7.20. When in the area above each line
associated with a certain volatility, the option price converges. The volatility
surface has been changed $\pm 5\%$.

Furthermore, we have noticed that when changing $S_0$, $K$ and $r$, the con-
vergence ratio does not change much. The differences in the convergence ratio
are not correlated to the changes in these three variables, but rather oscillating
around the first result with a magnitude of approximately $10^{-4}$. Changes in
the (constant) local volatility surface on the other hand clearly influences the
convergence ratio, as shown in Figure 7.20. When in the area above each line
associated with a certain volatility, the option price converges. The volatility
surface has been changed $\pm 5\%$.

Figure 7.20: Convergence ratio between $\Delta t$ and $\Delta Z$ for put options with differ-
ent volatility.

### 7.3.3 Options Almost at Maturity

When pricing options that have a very short time until expiry we want to make
sure that our program works and that we do not receive different prices because
of different values of $\theta$. When keeping the grid size constant and just changing
the time to maturity, we have found that even though the price gets lower as
we get closer to maturity, the methods also converge quickly even if we have
a rather big grid size. As seen in Figure 7.21 there are no differences between
the different values of $\theta$. In the diagram we have used Option 3, except that we
have removed the dividends and changed time to maturity so that it goes from
0.0025 years to 0.5 years. The call option is at the top and the put option at
Figure 7.21: Prices for options close to maturity for $\theta = 0$ (magenta), 0.5 (red) and 1 (black) without dividends.

the bottom in the figure. The red line for $\theta = 0.5$ is the last line drawn and the lines are so close that only one line seem to be drawn.

7.4 Dividends

As we have specified earlier, correct calculations for dividends is very crucial when pricing options on stocks paying discrete dividends. We have used Option 3 in order to obtain smooth EEB vectors in Figure 7.25. For more results regarding the EEB for put options, please check Section 7.4.1. When allowing discrete dividends to affect the price of our options, we have 3 different variables that can be altered, i.e. $\alpha$, $\phi$ and the dividends themselves.

$\alpha$ defines how much each dividend reduces the value of the underlying asset. Setting $\alpha = 0$ means that we have no loss in value at all and $\alpha = 1$ means that we decrease the value of the underlying asset by the same amount as the size of the dividend. In Figure 7.22 we have allowed $\alpha$ to go from 0.4 to 1.0 and we can clearly see that we have a linear effect on the option price due to changes in $\alpha$.

$\phi$ is the maximum ratio of the underlying asset that is allowed to be paid as dividend. If we set $\phi = 0$ we do not allow any dividend to be paid at all and for $\phi = 1$ we are using the liquidator strategy (as described in Section 4.2.2). We have used this limit for all levels of the underlying asset and not only when the dividend is higher than the current value of the underlying asset. In Figure 7.23 we have allowed $\phi$ to go from 0 to 1. As we can see $\phi$ have almost no effect on
the option price unless we have very small values of $\phi$, in which case we reduce the level of the dividend for values of the underlying asset close to $S_0$ and $K$ as well.

We can of course also change the level and timing of the dividend(s). As Option 3 have two dividends we have chosen not to change the time of them, but only their sizes. We have allowed the first dividend to go from 2 to 8 and the second to go from 1 to 7. As we can see in Figure 7.24 we have a linear effect on the option price for dividends close to the time of valuation and a more curvature effect from dividends further in the future. We get a lower value for a call option, the bigger the dividend is, while the opposite is true for a put option, as should be expected.

We can also examine how these variables affect the EEB for a put option. As we can see in Figure 7.25 the lower $\alpha$ we have the sooner the EEB will resurface, which means that we have more situations when it is worth to exercise the option early. $\phi$ on the other hand hardly affect the EEB at all unless we go as low as $\phi = 0$ in which case the EEB looks as if we had no dividend at all. If we change the size of the dividend we get a quicker rise of the EEB for smaller dividends. We do not see this effect for changes in the first dividend, because the second dividend is large enough to prevent the EEB from rising at all before the first dividend is reached. For all values we have used blue as the lowest value for the
7.4. Dividends

Figure 7.24: Different option prices for different sizes of dividends.

Figure 7.25: Different EEBs for different changes in variables.
variable and with a constant increase of the variables we use the colours red, magenta, green and finally black for the highest value for each variable. Each variable range in the same region as earlier in this section, with the lower and upper limits in blue and black.

7.4.1 The Early Exercise Boundary

As has been explained earlier (see Section 6.3), we find the early exercise boundary by calculating node values explicitly and then only keeping the node just above the EEB. Since the EEB is so crucial to the calculation of the American put, we have created several diagrams showing how it changes during our calculations. We start off by calculating the EEB for Option 1 (without dividends) and then repeat the process with constant volatility and risk-free rate. If we set the volatility surface to be constant with the average volatility from the volatility surface of Option 1 and do the same for the risk-free rate we can see that we get a smooth EEB diagram, while the EEB level for random volatility and risk-free rate is a little bit more varied, as seen in Figure 7.26.

![Figure 7.26: Different EEBs for constant (blue) and random (red) volatility and risk-free rate.](image)

If we then include a small dividend (0.5) at time \( t = 1.4 \) which is rather close to the time of maturity we can clearly see in the left diagram in Figure 7.27 that the EEB will resurface earlier than if we have a larger dividend (1.5). We can then include more small dividends (0.5 again) to see how the EEB goes to zero at each dividend but then quickly rises back to its earlier level, in the right diagram of the same figure.

When pricing American call options we only have the case of early exercise
just before each dividend. That means that we get a spike at each dividend and
the level of the dividend definitely affects the length of this spike, but since no
other nodes are exercised early, it is hard to see the full effect by looking at the
EEB. The effect is much more clearly seen when examining the early exercise
premium of the call option as was shown in Section 7.2.

7.5 Derivatives

As was defined in Section 6.5 we have created several programs for calculating
the derivatives. These all give the exact same results down to machine preci-
sion, in Table 7.9 (call option) and Table 7.10 (put option) the derivatives are
presented with five decimals. These derivatives has been calculated for Option
4 which has a rather small grid with one dividend and random local volatility
surface and risk-free rate. To be able to compare Rho and Vega for our gradient
programs to those from our Hessian programs we have summarized all Rhos and
Vegas from the gradient programs and used a parallel shift of 1 as the direction
for the derivatives of the risk-free rate and the volatility surface.

In Figure 7.28 and 7.29 we can see that all first order derivatives estimate
all small changes to any of the depending variables. We have used Option 2
for these calculations and allowed the strike to go from 0 to 100 (on the x-axis). In
Figure 7.28 we have the call option and in Figure 7.29 we have the put option.
From top to bottom we have the option price, Delta, Theta, Rho and Vega (with
their respective values on the y-axes). To compare our calculated derivatives we
have withdrawn the derivatives from DerivaGem that have been calculated with
a 500 step binomial tree. As can be seen, these are usually very close to the our
derivatives. We have also made numerical derivatives by changing one of the
dependent variables by a small amount (i.e. ±1 for \( S_0 \), ±0.03 for \( t_0 \), ±0.005 for
\( r \) and ±0.01 for \( \sigma \)) and then divided the difference by the total change. For a full
introduction of numerical derivatives see Appendix C. The difference between
these numerical derivatives and our calculated derivatives have been plotted in
red on the right side with \( \tau = \frac{x_{\text{num}} - x_{\text{our}}}{x_{\text{num}}} \), where \( x_{\text{num}} \) is a numerical derivative
and \( x_{\text{our}} \) is our calculated derivative.
### Call Program

<table>
<thead>
<tr>
<th>Option Price</th>
<th>Grad1</th>
<th>Grad2</th>
<th>Grad3</th>
<th>Hess1</th>
<th>Hess2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delta</td>
<td>0.66932</td>
<td>0.66932</td>
<td>0.66932</td>
<td>0.66932</td>
<td>0.66932</td>
</tr>
<tr>
<td>Gamma</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.01409</td>
<td>0.01409</td>
</tr>
<tr>
<td>Charm</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-0.33384</td>
<td>-0.33384</td>
</tr>
<tr>
<td>(\frac{\partial V}{\partial \sigma})</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.61862</td>
<td>0.61862</td>
</tr>
<tr>
<td>Vanna</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-0.23379</td>
<td>-0.23379</td>
</tr>
<tr>
<td>(\frac{\partial^2 V}{\partial S \partial r})</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-127.52887</td>
<td>-127.52887</td>
</tr>
<tr>
<td>Veta</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-33.25814</td>
<td>-33.25814</td>
</tr>
<tr>
<td>(\frac{\partial^2 V}{\partial r^2})</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>17.75179</td>
<td>17.75179</td>
</tr>
<tr>
<td>Vomma</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>6.01515</td>
<td>6.01515</td>
</tr>
</tbody>
</table>

Table 7.9: Derivatives for Option 4 as a call option.

### Put Program

<table>
<thead>
<tr>
<th>Option Price</th>
<th>Grad1</th>
<th>Grad2</th>
<th>Grad3</th>
<th>Hess1</th>
<th>Hess2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Option Price</td>
<td>8.99622</td>
<td>8.99622</td>
<td>8.99622</td>
<td>8.99622</td>
<td>8.99622</td>
</tr>
<tr>
<td>Delta</td>
<td>-0.39016</td>
<td>-0.39016</td>
<td>-0.39016</td>
<td>-0.39016</td>
<td>-0.39016</td>
</tr>
<tr>
<td>Theta</td>
<td>-12.02115</td>
<td>-12.02115</td>
<td>-12.02115</td>
<td>-12.02115</td>
<td>-12.02115</td>
</tr>
<tr>
<td>Vega</td>
<td>31.06150</td>
<td>31.06150</td>
<td>31.06150</td>
<td>31.06150</td>
<td>31.06150</td>
</tr>
<tr>
<td>Gamma</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.01405</td>
<td>0.01405</td>
</tr>
<tr>
<td>Charm</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-0.39459</td>
<td>-0.39459</td>
</tr>
<tr>
<td>(\frac{\partial V}{\partial \sigma})</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.65792</td>
<td>0.65792</td>
</tr>
<tr>
<td>Vanna</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-0.03734</td>
<td>-0.03734</td>
</tr>
<tr>
<td>(\frac{\partial^2 V}{\partial S \partial r})</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-125.11800</td>
<td>-125.11800</td>
</tr>
<tr>
<td>Veta</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-30.84043</td>
<td>-30.84043</td>
</tr>
<tr>
<td>(\frac{\partial^2 V}{\partial r^2})</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>55.32802</td>
<td>55.32802</td>
</tr>
<tr>
<td>Vomma</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>42.73837</td>
<td>42.73837</td>
</tr>
</tbody>
</table>

Table 7.10: Derivatives for Option 4 as a put option.
The ten second order derivatives are plotted in Figure 7.30 for the call options and in Figure 7.31 for the put options. These are, from upper left corner to the right and then down, Gamma, Charm, \( \frac{\partial^2 V}{\partial r^2} \), Vanna, \( \frac{\partial^2 V}{\partial r \partial \sigma} \), Veta, \( \frac{\partial^2 V}{\partial \sigma^2} \), and lastly Vomma, with the same axes as for the first order derivatives.

For the call option we can see that the derivatives calculated by our program for \( \frac{\partial^2 V}{\partial r^2} \), \( \frac{\partial^2 V}{\partial r \partial \sigma} \) and Vomma are not quite the same as the result of numerical derivatives. The same is true for the put option and the reason for all these miscalculations are probably due to the early exercise effect on the options. On the other hand one could argue that our small changes to the dependant variables are to big (e.g. 25% increase or decrease for the risk-free rate), we have chosen to use these changes anyway since the derivatives for all other calculations are still almost spot on.

Our implementation assumes that all derivatives of a node that is exercised early are zero. For some nodes that are close to not being exercised early the dependencies on the risk-free rate and the local volatility are not zero since a change of the risk-free rate or the volatility might mean that the node should not be exercised early. We have unfortunately been unable to find any literature covering the internal derivatives of nodes exercised early for an American option, so we are unable to set the derivatives of these nodes to anything except 0. We can however remove the dividend and thus get a call option that should not be exercised early. We can in Figure 7.32 see that we have no differences between the numerical derivatives and ours if we do not have a reason nor the option to exercise the option early. From DerivaGem we have only been able to withdraw Gamma, so therefore there are no other second order derivative diagrams with three plotted lines.
Figure 7.28: Price and first order derivatives for call options with different strikes with our calculations (blue) and DerivaGem’s (magenta) to the left and the comparison with numerical derivatives to the right.
Figure 7.29: Price and first order derivatives for put options with different strikes with our calculations (blue) and DerivaGem’s (magenta) to the left and the comparison with numerical derivatives to the right.
Figure 7.30: Second order derivatives for call options for different strikes with our calculations (blue) and numerical derivatives (red) with an extra Gamma from DerivaGem (magenta).
Figure 7.31: Second order derivatives for put options for different strikes with our calculations (blue) and numerical derivatives (red) with an extra Gamma from DerivaGem (magenta).
Figure 7.32: Second order derivatives for call options, without dividends, for different strikes with our calculations (blue) and numerical derivatives (red) with an extra Gamma from DerivaGem (magenta).
7.6 Runtimes

To give the user of our solvers an idea of each solver’s runtime we present Table 7.11 and Table 7.12, where we have measured the runtime for the different programs that we have implemented. The runtimes are presented in seconds.

<table>
<thead>
<tr>
<th>Grid size</th>
<th>100x100</th>
<th>300x300</th>
<th>1000x100</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pricing</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call1</td>
<td>0.066</td>
<td>0.207</td>
<td>1.081</td>
</tr>
<tr>
<td>Call2</td>
<td>0.019</td>
<td>0.049</td>
<td>0.403</td>
</tr>
<tr>
<td><strong>Full gradient</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CallADGrad1</td>
<td>60.514</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CallADGrad2</td>
<td>28.555</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CallADGrad3</td>
<td>18.610</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Hessian</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CallADHess1</td>
<td>182.888</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CallADHess2</td>
<td>0.112</td>
<td>0.648</td>
<td>5.482</td>
</tr>
</tbody>
</table>

Table 7.11: Runtimes of call option programs.

<table>
<thead>
<tr>
<th>Grid size</th>
<th>100x100</th>
<th>300x300</th>
<th>1000x100</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pricing</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Put1</td>
<td>0.060</td>
<td>0.210</td>
<td>1.073</td>
</tr>
<tr>
<td>Put2</td>
<td>0.016</td>
<td>0.056</td>
<td>0.422</td>
</tr>
<tr>
<td><strong>Full gradient</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PutADGrad1</td>
<td>55.423</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PutADGrad2</td>
<td>29.266</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PutADGrad3</td>
<td>17.383</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Hessian</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PutADHess1</td>
<td>157.265</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PutADHess2</td>
<td>0.115</td>
<td>0.594</td>
<td>5.019</td>
</tr>
</tbody>
</table>

Table 7.12: Runtimes of put option programs.

1 On a late 2013 MacBook Pro with 2.4 GHz Intel Core i5 processor.
Chapter 8

Discussion

In Chapter 7 we showed our results. Some of these results do, however, require some extra discussion as these results are not as straightforward as the others. These are mainly our results regarding different $\theta$, derivatives and the effect from $\phi$. We also end by discussing Widdicks (2002) solution by using $\hat{S}(t) = S(t) - S^*(t)$.

8.1 The Theta Method

In Section 7.3.1 we showed that the option price converges for all values of $\theta$ as long as we have a good enough ratio between $\Delta t$ and $\Delta Z$. We could however see that $\theta = 0.5$ converges the most quickly and $\theta = 0$ requires the largest grid before it converges to the correct value. Nevertheless, if one picks a large enough grid it does not matter which value of $\theta$ one chooses as all the choices lead to the same option price, if the ratio between $\Delta t$ and $\Delta Z$ is good enough for convergence. A larger grid will of course lead to an increased computational time. This is all true for both call and put options, on underlying asset with or without dividends.

8.2 Derivatives

As we showed in the Section 7.5 we have been able to correctly determine all first order derivatives for both call and put options, on an underlying asset with an arbitrary amount of dividends. We have furthermore been able to determine all second order derivatives for all call and put options that depends on $S$ or $t$ in some way. What we have not been able to price correctly are the second order derivatives $\frac{\partial^2 V}{\partial r^2}$, $\frac{\partial^2 V}{\partial r \partial \sigma}$ and Vomma. The reason for this, as we stated in Section 7.5 and Figure 7.32, is that we are unable to price the second order derivatives correctly when we exercise a node early.

The reason why only $\frac{\partial^2 V}{\partial r^2}$, $\frac{\partial^2 V}{\partial r \partial \sigma}$ and Vomma are valued incorrectly is that any derivative of $\frac{\partial}{\partial t}$ is essentially a derivative of $\frac{\partial}{\partial t_0}$ and $\frac{\partial}{\partial S}$ is of $\frac{\partial}{\partial S_0}$. This means that we do not get any contribution to these derivatives except in the last time-step (for $t_0$) or after the last time-step (for $S_0$), when we do the interpolation to get the option price for the current value of the underlying asset.
As we discussed earlier we set all derivatives to 0 when we exercise a node prematurely. If we could instead find a way to set these derivatives correctly we would probably end up with a much better estimation of these derivatives. The problem is that when a node is exercised early its price changes and this change is a non-linear change. This value would not necessarily be higher for a certain change of the local volatility or the local risk-free rate, but it would definitely be higher if we changed the volatility or the risk-free rate enough so that it would no longer be beneficial to exercise the option early. It would, nevertheless, never be lower since the lowest value an American option can have at any time for any value of the underlying asset is the value to exercise the option early. This means that the first order derivative for these nodes are non-continuous. When we are close to the EEB these derivatives are probably still zero (since we are getting good results for all our first order derivatives) but as we calculate the second order derivatives we need to incorporate this discontinuity of the first order derivative in order to value it correctly. Unfortunately, we have not found any way to do this correctly.

8.3 The Phi Variable

In Section 4.2.2 we showed that when a dividend is higher than the current value of the underlying asset (which is common for numerical pricing schemes on assets with discrete dividends) one has to decide whether to use the liquidator or the survivor approach. In Section 6.2 we added the variable \( \phi \) to cope with this and allow for the user to specify any combination of the two approaches. As we found out in Section 7.4 there is no difference to the option price, no matter which value for \( \phi \) we pick, unless we pick a value so low that any ordinary dividend will be reduced, because of \( \phi \), even for values of the underlying asset close to \( S_0 \) or \( K \). This should not be the case as the liquidator or survivor approach should only be implemented when the value of the underlying asset is below the size of the dividend. However, we have implemented the use of \( \phi \) so that it affects dividends for all levels of the underlying asset to ensure that we always have a continuous impact from the dividend, no matter what level of \( \phi \) the user picks. Our conclusion is nevertheless the same as the one by Haug et al. (2003). It does not matter if one chooses the liquidator or the survivor approach, the price of the option will not be affected.

8.4 Change of Variables for the American Put Option

In Section 4.1.3 we discussed how one could calculate the value of an American put option by changing the variables, and more specifically we discussed the work by Widdicks (2002). We have implemented his solution, but have been unable to get results that are anywhere near any other that we have got from any of the different pricing models we used. What we can see is that the EEB changes a lot as we change the distance between the nodes just above the boundary. Besides the problem that the EEB are dependent on the distance between nodes, we get too low prices for the option to be realistic. When comparing the result with our own programs and calculating the value numerically from the nodes
just around the border (one below and one above), we can see that we do not have a derivative of -1 across the EEB. The derivative is closer to -0.5 than -1, but it varies of course for different options and for different times during the life of the option.

It seems to us as if the requirement that $\frac{\partial V}{\partial S} = -1$ on the EEB requires too small distances of the nodes just above the EEB to be able to work properly. Widdicks (2002) claims that he gets good results from his approximation, but we are unable to reproduce those results and therefore cannot conclude that it works. As far as we have found the solution provided by Hull (2011) is much more reliable and does not provide any major errors, even though one has to use nodes outside the area where the PDE is valid, when pricing all nodes inside the area. We discussed this briefly in Section 4.1.1 but we have found that although the nodes outside of the boundary do not necessarily follow the requirements of the PDE, they are still close enough to generate good results.
Chapter 9

Conclusions

We have created several programs that gives good results. We will in this final chapter discuss these results further and give some suggestions on how our work can be further developed.

9.1 Summarized Results

We have constructed several programs that price call or put options for both American and European options correctly and efficiently. We can furthermore price these options correctly when the underlying asset is paying discrete dividends, if these are known beforehand. We have shown that we get convergence for all values of $\theta$, as long as we have a good enough ratio between $\Delta Z$ and $\Delta t$ and that our grid is large enough. We have made sure that all programs work good for long as well as for short options, and that there is no risk that we get any mispricings for options with a short time to maturity, no matter what value for $\theta$ we use. We have also clearly showed that our hard-coded AD solution is much quicker than our operator overloaded solutions. We have lastly been able to value all first order derivatives correctly, which have been ensured by comparison with two other models, as well as most of the second order derivatives.

What we have not been able to do is to price American options with a forward PDE. We have discussed that the American option on an underlying asset paying one dividend can be replicated with a collection of options with a forward representation, but it will not price an American option correctly if the discrete dividend is too large. We have furthermore, been unable to price all second order derivatives correctly, due to our inability to set the second order derivatives for nodes close the EEB correctly.

9.2 How to Enhance or Continue This Master Thesis

There are some areas that we have left untouched or that we have felt that we did not have enough time to fully examine. We would here like to give some
examples for others what could be done if one wanted to dwell deeper in any of the areas concerning this master thesis.

9.2.1 Forward PDE for American Options

When we did our literature study we were unable to find any articles or other scientific papers that prices American options with a forward PDE. The best article we found on the subject was the article by White (2013). It was just above one year after its publication when we began our studies and hopefully some new advancements have been, or will be, made in this area in the coming future. The possibility of pricing American options with a forward PDE would do a lot to decrease the time necessary to price several options which would then lead to a much quicker process when determining a local volatility surface.

9.2.2 Automatic Differentiation

We have used the valder object provided by Neidinger (2010) for overloaded operations. As we described in Section 6.4 this was a very slow solution and our hard-coded programs were much quicker than any of the programs that used any of the implemented versions of the valder object. However, we did not alter the valder object so that it could do vector operations. If one would be able to incorporate this into the valder object it would probably mean a very quick solution that still has a short code for our pricing program, even though a hard-coded solution usually is the quickest.

9.2.3 Second Order Derivatives for Nodes Exercised Early

We discussed thoroughly in Section 8.2 that we have not been able to price the second order derivatives properly for nodes that are exercised early. It would probably have been a benefit for us if we would have had any valuation schemes that took care of this problem for us.

9.2.4 Methods Using the Boundary Condition of the EEB

As we stated in Section 8.4, we failed to incorporate the solution by Widdicks (2002) and were not able to price the options correctly with his method. We furthermore concluded that the finite difference of our method was not -1 across the EEB, which was one of the key elements in Widdicks’ solution. It would be interesting to search for other methods that incorporates this boundary condition for the border and compare their results with ours and Widdicks’.

9.2.5 Increased Accuracy for the Finite Differences

When setting up the grid, as stated in Section 3.3 we used the common way to do this, as described by Hull (2011). But as we describe in Appendix C it is possible to get a higher accuracy by using more than three nodes to define \( \frac{\partial V}{\partial S} \) and \( \frac{\partial^2 V}{\partial S^2} \). It would be quite interesting to examine this further and see if a quicker convergence of the option price could be found.
Bibliography


Appendix A

Derivation of the Theta Method

In this appendix we show the derivation of the combination of implicit and explicit finite difference method, that starts from Hull’s (2011) definition and ends with the equation given in Section 3.3.4. We also derive the specific equation for the upper (or lower) border nodes for a call (or put) option in the money.

Remember the PDE (3.4), using $Z = \ln S$ as the underlying variable, defined as

$$
\frac{\partial V}{\partial t} + \left( r - \frac{\sigma^2}{2} \right) \frac{\partial V}{\partial Z} + \frac{1}{2} \sigma^2 \frac{\partial^2 V}{\partial Z^2} = rV.
$$

A.1 Implicit

For the implicit solution Hull (2011) furthermore defines

$$
\frac{\partial V}{\partial t} = f_{i+1,j} - f_{i,j} \Delta t,
$$

$$
\frac{\partial V}{\partial Z} = \frac{f_{i,j+1} - f_{i,j-1}}{2 \Delta Z},
$$

$$
\frac{\partial^2 V}{\partial Z^2} = \frac{f_{i,j+1} - 2f_{i,j} + f_{i,j-1}}{\Delta Z^2}.
$$

However, we do not want to keep $\Delta t$ and $\Delta Z$ constant throughout the grid so we want to use $\Delta t_i$ and $\Delta Z_j$ instead where $\Delta t_i$ is the time difference between node $f_{i,j}$ and $f_{i+1,j}$ and $\Delta Z_j$ is the difference in the underlying asset between the nodes $f_{i,j}$ and $f_{i,j+1}$. This means that we have to use

$$
\frac{\partial V}{\partial t} = \frac{f_{i+1,j} - f_{i,j}}{\Delta t_i} \tag{A.1}
$$

$$
\frac{\partial V}{\partial Z} = \frac{f_{i,j+1} - f_{i,j-1}}{\Delta Z_j + \Delta Z_{j-1}} \tag{A.2}
$$

$$
\frac{\partial^2 V}{\partial Z^2} = \left( \frac{f_{i,j+1} - f_{i,j}}{\Delta Z_j} - \frac{f_{i,j} - f_{i,j-1}}{\Delta Z_{j-1}} \right) / \left( \frac{\Delta Z_j}{2} + \frac{\Delta Z_{j-1}}{2} \right) \tag{A.3}
$$

instead.

When putting (A.1), (A.2) and (A.3) in (3.4) we get

$$
f_{i,j+1} + \left( r - \frac{\sigma^2}{2} \right) \frac{f_{i,j+1} - f_{i,j-1}}{\Delta Z_j + \Delta Z_{j-1}} + \sigma^2 \left( \frac{f_{i,j+1} - f_{i,j}}{\Delta Z_j} - \frac{f_{i,j} - f_{i,j-1}}{\Delta Z_{j-1}} \right) / (\Delta Z_j + \Delta Z_{j-1}) = r f_{i,j} \tag{A.4}
$$

Hjelmberg, Lagerström, 2014.
or
\[
\alpha_j f_{i,j-1} + \beta_j f_{i,j} + \gamma_j f_{i,j+1} = f_{i+1,j} \tag{A.5}
\]

where
\[
\alpha_j = \frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r - \frac{\sigma^2}{2} \right) - \frac{\Delta t_i}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \sigma^2
\]
\[
\beta_j = 1 + r \Delta t_i + \frac{\Delta t_i}{\Delta Z_j \Delta Z_{j-1} \sigma^2}
\]
\[
\gamma_j = -\frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r - \frac{\sigma^2}{2} \right) - \frac{\Delta t_i}{\Delta Z_j(\Delta Z_j + \Delta Z_{j-1})} \sigma^2.
\]

### A.2 Explicit

For the explicit solution, Hull (2011) defines \( \frac{\partial V}{\partial Z} = \frac{f_{i+1,j} - f_{i,j}}{2\Delta Z^2} \) and \( \frac{\partial^2 V}{\partial Z^2} = \frac{f_{i+1,j+1} - 2f_{i+1,j} + f_{i+1,j-1}}{\Delta Z_j + \Delta Z_{j-1}} \). Repeating the steps as for the implicit solution gives us
\[
\frac{\partial V}{\partial t} = \frac{f_{i+1,j} - f_{i,j}}{\Delta t_i}
\]
\[
\frac{\partial^2 V}{\partial Z^2} = \left( \frac{f_{i+1,j+1} - f_{i+1,j}}{\Delta Z_j} - \frac{f_{i+1,j} - f_{i+1,j-1}}{\Delta Z_{j-1}} \right) / \left( \frac{\Delta Z_j}{2} + \frac{\Delta Z_{j-1}}{2} \right)
\]
that we combine into
\[
\frac{f_{i+1,j} - f_{i,j}}{\Delta t_i} + \left( r - \frac{\sigma^2}{2} \right) \frac{f_{i+1,j+1} - f_{i+1,j-1}}{\Delta Z_j + \Delta Z_{j-1}} + \sigma^2 \left( \frac{f_{i+1,j+1} - f_{i+1,j}}{\Delta Z_j} - \frac{f_{i+1,j} - f_{i+1,j-1}}{\Delta Z_{j-1}} \right) / (\Delta Z_j + \Delta Z_{j-1}) = r f_{i,j} \tag{A.9}
\]
or
\[
\alpha_j^* f_{i+1,j-1} + \beta_j^* f_{i+1,j} + \gamma_j^* f_{i+1,j+1} = f_{i,j} \tag{A.10}
\]

where
\[
\alpha_j^* = \frac{1}{1 + r \Delta t_i} \left( -\frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r - \frac{\sigma^2}{2} \right) + \frac{\Delta t_i}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \sigma^2 \right)
\]
\[
\beta_j^* = \frac{1}{1 + r \Delta t_i} \left( 1 - \frac{\Delta t_i}{\Delta Z_j \Delta Z_{j-1}} \sigma^2 \right)
\]
\[
\gamma_j^* = \frac{1}{1 + r \Delta t_i} \left( \frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r - \frac{\sigma^2}{2} \right) + \frac{\Delta t_i}{\Delta Z_j(\Delta Z_j + \Delta Z_{j-1})} \sigma^2 \right).
\]

### A.3 Combining Implicit and Explicit Solution

Using \( \theta \) in the same way as Benbow (2005) and combining (A.5) and (A.10) gives
\[
\theta \alpha_j f_{i,j-1} + (\theta \beta_j + (1 - \theta)) f_{i,j} + \theta \gamma_j f_{i,j+1} = (1 - \theta) \alpha_j^* f_{i+1,j-1} + ((1 - \theta) \beta_j^* + \theta) f_{i+1,j} + (1 - \theta) \gamma_j^* f_{i+1,j+1} \tag{A.11}
\]
which is equal to (3.5).
A.4 Boundary nodes

Since we want to use $\frac{\partial V}{\partial S} = 1$ for a call option and $\frac{\partial V}{\partial S} = -1$ for a put option as the upper respectively the lower boundary, we need to derive the finite difference equation for these boundaries.

### A.4.1 Call option

For a call option the upper boundary is $\frac{\partial V}{\partial S} = 1$. With $Z = \ln \frac{S}{S_0}$ we get $\frac{\partial Z}{\partial S} = \frac{\partial}{\partial S} \ln \frac{S}{S_0} = \frac{S_0}{S S_0} = \frac{1}{S}$. If we use this when deriving $\frac{\partial V}{\partial S}$ we see that $\frac{\partial V}{\partial Z} = \frac{\partial}{\partial Z} S = S_0 e^Z = S_0 e^{\ln \frac{S}{S_0}} = S_0 \frac{S}{S_0} = S$. Using these equations in (3.4) we get

$$f_{i+1,N} - f_{i,N} \Delta t + \left( r - \frac{\sigma^2}{2} \right) S_{\text{max}} - \frac{1}{2} \sigma^2 S_{\text{max}}^2 = r f_{i,N}.$$  

This can then be reorganized so that we get

$$f_{i,N} = \frac{f_{i+1,N} + r S_{\text{max}} \Delta t}{1 + r \Delta t}.$$  

(A.12)

### A.4.2 Put option

For a put option the lower boundary is $\frac{\partial V}{\partial S} = -1$. In the same way as for the call option we can derive that $\frac{\partial V}{\partial Z} = -S$. Since $S = S_0 e^Z$, $\frac{\partial^2 V}{\partial Z^2} = \frac{\partial}{\partial Z} S = -S_0 e^Z = -S_0 \frac{S}{S_0} = -S$. Using these equations in (3.4) we get

$$f_{i+1,0} - f_{i,0} \Delta t = \left( r - \frac{\sigma^2}{2} \right) S_{\text{min}} - \frac{1}{2} \sigma^2 S_{\text{min}}^2 = r f_{i,0}.$$  

This can then be reorganized so that we get

$$f_{i,0} = \frac{f_{i+1,0} - r S_{\text{min}} \Delta t}{1 + r \Delta t}.$$  

(A.13)
Appendix B

Interpolation

There exists many different interpolation schemes and the most common is of course the standard linear interpolation. We will here present these and other (more advanced) schemes for interpolation that will be used in this master thesis.

B.1 Linear Interpolation

Given two points \((x_1, x_2)\) with a distance \(\Delta x\) between them with known values \((X_1, X_2)\), we want to find the unknown value \((Y)\) of a point \((y)\) located somewhere between the points. With a linear interpolation the value is simply calculated as

\[
Y = X_1 + (X_2 - X_1) \frac{y - x_1}{\Delta x}. \tag{B.1}
\]

By using more than two points one can estimate the derivatives of the value function and thus get better values for \(Y\).

B.2 Four Point Interpolation

Andricopoulos (2002) shows an easy way to get a good measure of \(Y\) by using 4 different points \((x_1, x_2, x_3, x_4)\). Andricopoulos starts by calculating the average distances \((A_1, A_2, A_3, A_4)\) to \(y\) from the four points and then combining them to get the value for \(y\) as

\[
Y = A_1 X_1 + A_2 X_2 + A_3 X_3 + A_4 X_4 \tag{B.2}
\]

where

\[
A_1 = \frac{y - x_3}{x_1 - x_3} \frac{y - x_4}{x_1 - x_4}, \quad A_2 = \frac{y - x_1}{x_2 - x_1} \frac{y - x_3}{x_2 - x_3}, \quad A_3 = \frac{y - x_1}{x_4 - x_1} \frac{y - x_2}{x_4 - x_2}, \quad A_4 = \frac{y - x_2}{x_3 - x_2} \frac{y - x_4}{x_3 - x_4}.
\]

B.3 Bilinear Interpolation

A bilinear interpolation is the combination of two ordinary linear interpolations when you want to find the value \((Y)\) of a point \((y_{i,j})\) located in the middle of
four other points \((x_{11}, x_{12}, x_{21}\) and the upper right corner \(x_{22}\)) with known values \((X_{11}, X_{12}, X_{21}\) and \(X_{22}\)). Two adjacent nodes in the square formation are spaced \(\Delta x\). This interpolation scheme is given by

\[
Y = X_1 + (X_2 - X_1) \frac{y_j - x_1}{\Delta x}
\]  \hspace{1cm} (B.3)

where

\[
X_1 = X_{11} + (X_{12} - X_{11}) \frac{y_i - x_{11}}{\Delta x} \quad \text{and} \quad X_2 = X_{21} + (X_{22} - X_{21}) \frac{y_i - x_{21}}{\Delta x}.
\]
Appendix C

Numerical Derivatives

When calculating derivatives, one easy way is to calculate the numerical derivatives. This is usually the case when one has access to the input and output variables but not full knowledge of the function involved, or if the expression describing the function is much to cumbersome to derive.

C.1 First Order Derivatives

For first order derivatives we assume that we have an unknown function $f$, with the result $y$ and a dependent variable $x$. We can then increase $x$ by a small amount, $\Delta x$, and then calculate $y$ again and get $y_+$. If we instead decrease $x$ by the same small amount we get $y_-$. By subtracting $y_-$ from $y_+$ and dividing the difference by $2\Delta x$ we get the first order derivative of $y$.

If we are unable to decrease or increase $x$ for any reason we can do a one-sided numerical derivative instead. We then change $x$ in the possible direction, take the $y$ for the highest value of $x$ that we have used, subtract the other $y$ and divide the difference by $\Delta x$.

To increase the accuracy one can do larger (or smaller) changes of $x$ and combine all of these values of $y$. The important thing is then to use to correct coefficients for each $y$. Since we do not have the same accuracy for a one-sided difference as for a two-sided difference, it is rather common (e.g. Widdicks (2002)) to use $x+\Delta x$ to get $y_+$ and $x+2\Delta x$ to get $y_{++}$. In this case the derivative is calculated as $-\frac{y_{++}+y_+-y_+}{2\Delta x}$. Since this is a good enough approximation for us, we do not include any other higher levels of accuracy.

C.2 Second Order Derivatives

For second order derivatives we can have the dependence of one or two variables. If we start with dependence on one variable we can use the one-sided difference to get the derivative at one side of $y$. This is in fact the two-sided derivative for $y_+$ if our dependent variable for $y_+$ is $x + \frac{\Delta x}{2}$. If we then calculate the one-sided difference on the other side of $y$, we get the derivative for $y_- if our dependent variable is $x - \frac{\Delta x}{2}$. Finally the difference between these two one-sided differences are divided by $\Delta x$ to obtain the second order derivative for $y$ when our dependent variable is $x$.  

Hjelmberg, Lagerström, 2014.
For the second order derivatives with dependence on two variables (\(x_1\) and \(x_2\)) we need to calculate four different values of \(y\). Since we can increase and decrease both \(x_1\) and \(x_2\), we get four values if we calculate \(y\) for all combinations of \(x_1\) and \(x_2\). What is basically done is a variant of bilinear interpolation. The derivative of \(y_+\) is calculated by keeping \(x_1 + \Delta x_1\) fixed and checking the dependence for \(y_+\) on \(x_2\) as if it was a first order derivative. We then repeat this for \(y_-\) when using \(x_1 - \Delta x_1\) as the fixed value instead. We can then calculate the second order derivative of \(y\) by taking \(\frac{y_+ - y_-}{2\Delta x_1}\).

Both these calculations can of course be done with higher accuracy but there has not been any reason for doing this during the entire progress of this master thesis.
Appendix D

AD Variables

When calculating the derivatives of the option price we will need the derivatives of $\alpha$, $\beta$, $\gamma$, $\alpha^*$, $\beta^*$ and $\gamma^*$ as well as the derivative of the nodes one time step ahead of our current calculations. When $t = T$, all the derivatives of the nodes will be zero as explained in Section 6.4. We have derived all the variables below and sorted them for easy reading. When calculating the derivatives of $\frac{\partial}{\partial r}$ and $\frac{\partial}{\partial \sigma}$ (or any combination with any of these for the second order derivatives) we need to incorporate the specific derivatives of each node as well. These are represented by $r'$ and $\sigma'$ in the equations below. For the gradient these variables are one for the specific node(s) that are currently being priced and zero for all other variables. For the Hessian the variables depends on the direction of the risk-free rate and volatility surface that the user specifies.

The variables that we have derived are

$$\tilde{\alpha}_j = \theta \alpha_j = \theta \left( \frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r - \frac{\sigma^2}{2} \right) - \frac{\Delta t_i}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \left( r - \frac{\sigma^2}{2} \right) \right)$$

$$\tilde{\beta}_j = 1 - \theta + \theta \beta_j = 1 - \theta + \theta \left( 1 + r \Delta t_i + \frac{\Delta t_i}{\Delta Z_j \Delta Z_{j-1}} \sigma^2 \right)$$

$$\tilde{\gamma}_j = \theta \gamma_j = \theta \left( - \frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r - \frac{\sigma^2}{2} \right) - \frac{\Delta t_i}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \left( r - \frac{\sigma^2}{2} \right) \right)$$

$$\tilde{\alpha}^*_j = (1-\theta)\alpha^*_j = \frac{1 - \theta}{1 + r \Delta t_i} \left( - \frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r - \frac{\sigma^2}{2} \right) + \frac{\Delta t_i}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \left( r - \frac{\sigma^2}{2} \right) \right)$$

$$\tilde{\beta}^*_j = \theta + (1-\theta)\beta^*_j = \theta + \frac{1 - \theta}{1 + r \Delta t_i} \left( 1 - \frac{\Delta t_i}{\Delta Z_j \Delta Z_{j-1}} \sigma^2 \right)$$

$$\tilde{\gamma}^*_j = (1-\theta)\gamma^*_j = \frac{1 - \theta}{1 + r \Delta t_i} \left( \frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} \left( r - \frac{\sigma^2}{2} \right) + \frac{\Delta t_i}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \left( r - \frac{\sigma^2}{2} \right) \right)$$

from (3.5), where we have multiplied them with the corresponding value of $\theta$ since those are the variables that we will use in the program. When calculating the dependencies on $t$ we need to derive $\Delta t_i$. The definition $\Delta t_i = t_{i+1} - t_i$ is used to get $\frac{\partial \Delta t_i}{\partial t_j} = -1$ which has been used below. As stated in Section 6.4 this variable is only used in the last time-step.

Hjelmberg, Lagerström, 2014. 81
D.1 Gradient

These are the variables that will be used to calculate the gradient of the option price in each node.

**Alpha**

\[
\frac{\partial \tilde{\alpha}_j}{\partial r} = \theta \Delta t_i \Delta Z_{j-1} r' \\
\frac{\partial \tilde{\alpha}_j}{\partial t} = \theta \left( -\frac{r}{\Delta Z_j + \Delta Z_{j-1}} + \frac{1}{2} \frac{\Delta^2}{\Delta Z_j + \Delta Z_{j-1}} \sigma^2 \right) \\
\frac{\partial \tilde{\alpha}_j}{\partial \sigma} = -\theta \frac{2 + \Delta Z_{j-1}}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \Delta t_i \sigma r'
\]

**Beta**

\[
\frac{\partial \tilde{\beta}_j}{\partial r} = \theta \Delta t_i r' \\
\frac{\partial \tilde{\beta}_j}{\partial t} = -\theta \left( r + \frac{1}{\Delta Z_j + \Delta Z_{j-1}} \sigma^2 \right) \\
\frac{\partial \tilde{\beta}_j}{\partial \sigma} = \theta \frac{2 - \Delta Z_j}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \Delta t_i \sigma r'
\]

**Gamma**

\[
\frac{\partial \tilde{\gamma}_j}{\partial r} = -\theta \frac{\Delta t_i}{\Delta Z_j + \Delta Z_{j-1}} r' \\
\frac{\partial \tilde{\gamma}_j}{\partial t} = \theta \left( \frac{r}{\Delta Z_j + \Delta Z_{j-1}} - \frac{1}{2} \frac{1}{\Delta Z_j + \Delta Z_{j-1}} \sigma^2 \right) \\
\frac{\partial \tilde{\gamma}_j}{\partial \sigma} = -\theta \frac{2 - \Delta Z_j}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \Delta t_i \sigma r'
\]

**Alpha***

\[
\frac{\partial \tilde{\alpha}^*_j}{\partial r} = \frac{1 - \theta}{(1 + r \Delta t_i)^2} \left( \Delta t_i - \frac{1}{\Delta Z_j + \Delta Z_{j-1}} \Delta t_i \sigma^2 \right) r' \\
\frac{\partial \tilde{\alpha}^*_j}{\partial t} = \frac{1 - \theta}{(1 + r \Delta t_i)^2} \left( \frac{r}{\Delta Z_j + \Delta Z_{j-1}} - \frac{1}{2} \frac{1}{\Delta Z_j + \Delta Z_{j-1}} \sigma^2 \right) \\
\frac{\partial \tilde{\alpha}^*_j}{\partial \sigma} = \frac{1 - \theta}{1 + r \Delta t_i} \frac{2 + \Delta Z_{j-1}}{\Delta Z_{j-1}(\Delta Z_j + \Delta Z_{j-1})} \Delta t_i \sigma r'
\]

**Beta***

\[
\frac{\partial \tilde{\beta}^*_j}{\partial r} = -\frac{1 - \theta}{(1 + r \Delta t_i)^2} \left( \Delta t_i - \frac{1}{\Delta Z_j + \Delta Z_{j-1}} \Delta t_i \sigma^2 \right) r' \\
\frac{\partial \tilde{\beta}^*_j}{\partial t} = \frac{1 - \theta}{(1 + r \Delta t_i)^2} \left( r + \frac{1}{\Delta Z_j + \Delta Z_{j-1}} \sigma^2 \right) \\
\frac{\partial \tilde{\beta}^*_j}{\partial \sigma} = -\frac{1 - \theta}{1 + r \Delta t_i} \Delta Z_j \Delta t_i \sigma r'
\]
D.2. Hessian

These are the variables that will be used to calculate the Hessian of the option price in each node.

**Alpha**

\[
\frac{\partial^2 \hat{\alpha}_j}{\partial r^2} = -\theta \frac{\Delta Z_j}{\Delta Z_{j-1}} \Delta t_i (\sigma')^2 \\
\frac{\partial^2 \hat{\alpha}_j}{\partial r \partial t} = \frac{\partial^2 \hat{\alpha}_j}{\partial t \partial r} = \frac{\partial^2 \hat{\alpha}_j}{\partial t^2} = \frac{\partial^2 \hat{\alpha}_j}{\partial \sigma^2} = \frac{\partial^2 \hat{\alpha}_j}{\partial \sigma \partial r} = 0
\]

**Beta**

\[
\frac{\partial^2 \hat{\beta}_j}{\partial \sigma^2} = \theta \frac{2 \Delta Z_j}{\Delta Z_{j-1}} \Delta t_i (\sigma')^2 \\
\frac{\partial^2 \hat{\beta}_j}{\partial r \partial t} = \frac{\partial^2 \hat{\beta}_j}{\partial t \partial r} = \theta r' \\
\frac{\partial^2 \hat{\beta}_j}{\partial t \partial \sigma} = \frac{\partial^2 \hat{\beta}_j}{\partial \sigma \partial t} = \theta \frac{2 \Delta Z_j}{\Delta Z_{j-1}} \sigma (\sigma') \\
\frac{\partial^2 \hat{\beta}_j}{\partial r \partial t} = \frac{\partial^2 \hat{\beta}_j}{\partial t^2} = \frac{\partial^2 \hat{\beta}_j}{\partial t \partial \sigma} = \frac{\partial^2 \hat{\beta}_j}{\partial \sigma \partial t} = 0
\]

**Gamma**

\[
\frac{\partial^2 \hat{\gamma}_j}{\partial \sigma^2} = -\theta \frac{2 - \Delta Z_j}{\Delta Z_{j-1}} \Delta t_i (\sigma')^2 \\
\frac{\partial^2 \hat{\gamma}_j}{\partial r^2} = \frac{\partial^2 \hat{\gamma}_j}{\partial t^2} = \frac{\partial^2 \hat{\gamma}_j}{\partial r \partial \sigma} = \frac{\partial^2 \hat{\gamma}_j}{\partial \sigma \partial r} = 0
\]
Appendix D. AD Variables

\[ \frac{\partial^2 \tilde{\alpha}^*_i}{\partial t^2} = \frac{(1-\theta)}{(1+r\Delta t)^3} \left( \frac{\Delta Z_j}{\Delta Z_{j-1}} \right)^{2+2\Delta Z_j} \Delta t_i (\sigma')^2 \right) r' \]

\[ \frac{\partial^2 \tilde{\alpha}^*_j}{\partial t^2} = \frac{1-\theta}{1+r\Delta t} \left( \frac{\Delta Z_j}{\Delta Z_{j-1}} + \Delta Z_j \Delta t_i (\sigma')^2 \right) \]

\[ \frac{\partial^2 \tilde{\alpha}^*_j}{\partial \sigma \partial t} - \frac{\partial \tilde{\alpha}^*_j}{\partial \sigma \partial t} = \frac{(1-\theta)}{(1+r\Delta t)^3} \left( \frac{1-r\Delta t_j}{\Delta Z_j + \Delta Z_{j-1}} + \frac{2+\Delta Z_j}{\Delta Z_j + \Delta Z_{j-1}} \Delta t_i (\sigma')^2 \right) \]

\[ \frac{\partial^2 \tilde{\alpha}^*_j}{\partial \sigma \partial t} = \frac{1-\theta}{1+r\Delta t} \left( \frac{\Delta Z_j}{\Delta Z_{j-1}} \Delta t_i (\sigma')^2 \right) \]

\[ \frac{\partial^2 \tilde{\alpha}^*_j}{\partial \sigma \partial t} = \frac{1-\theta}{1+r\Delta t} \left( \frac{\Delta Z_j}{\Delta Z_{j-1}} \Delta t_i (\sigma')^2 \right) \]

\[ \frac{\partial^2 \tilde{\beta}^*_j}{\partial t^2} = \frac{(1-\theta)}{(1+r\Delta t)^3} \left( \frac{2}{\Delta Z_j + \Delta Z_{j-1}} \Delta t_i (\sigma')^2 \right) \]

\[ \frac{\partial^2 \tilde{\beta}^*_j}{\partial t^2} = \frac{(1-\theta)}{(1+r\Delta t)^3} \left( \frac{2}{\Delta Z_j + \Delta Z_{j-1}} \Delta t_i (\sigma')^2 \right) \]

\[ \frac{\partial^2 \tilde{\beta}^*_j}{\partial \sigma \partial t} = \frac{(1-\theta)}{(1+r\Delta t)^3} \left( \frac{1-r\Delta t_j}{\Delta Z_j + \Delta Z_{j-1}} \Delta t_i (\sigma')^2 \right) \]

\[ \frac{\partial^2 \tilde{\beta}^*_j}{\partial \sigma \partial t} = \frac{(1-\theta)}{(1+r\Delta t)^3} \left( \frac{1-r\Delta t_j}{\Delta Z_j + \Delta Z_{j-1}} \Delta t_i (\sigma')^2 \right) \]

\[ \frac{\partial^2 \tilde{\gamma}^*_j}{\partial t^2} = \frac{(1-\theta)}{(1+r\Delta t)^3} \left( \frac{2}{\Delta Z_j + \Delta Z_{j-1}} \Delta t_i (\sigma')^2 \right) \]

\[ \frac{\partial^2 \tilde{\gamma}^*_j}{\partial \sigma \partial t} = \frac{(1-\theta)}{(1+r\Delta t)^3} \left( \frac{1-r\Delta t_j}{\Delta Z_j + \Delta Z_{j-1}} \Delta t_i (\sigma')^2 \right) \]

\[ \frac{\partial^2 \tilde{\gamma}^*_j}{\partial \sigma \partial t} = \frac{(1-\theta)}{(1+r\Delta t)^3} \left( \frac{1-r\Delta t_j}{\Delta Z_j + \Delta Z_{j-1}} \Delta t_i (\sigma')^2 \right) \]
D.3 Boundary nodes

Since we are using \( \frac{\partial V}{\partial S} = 1 \) for a call option and \( \frac{\partial V}{\partial S} = -1 \) for a put option as the upper respectively the lower boundary, we need to derive the derivatives for these nodes separately.

D.3.1 Call Option

For a call option we have the upper boundary \( \frac{\partial V}{\partial S} = 1 \), which gives \( f_{i,N} = f_{i+1,N} + S_{\text{max}} r_i \Delta t_i \) as was defined in (3.6). If we derive this equation we get

\[
\begin{align*}
\frac{\partial f_{i,N}}{\partial r} &= \frac{\partial f_{i+1,N}}{\partial r} + s_{\text{max}} - f_{i+1,N} \Delta t_i r_i' \\
\frac{\partial f_{i,N}}{\partial t} &= 1 + r_i \Delta t_i + f_{i+1,N} - S_{\text{max}} (1 + r_i \Delta t_i)^2 r_i' \\
\frac{\partial^2 f_{i,N}}{\partial r^2} &= \frac{\partial^2 f_{i+1,N}}{\partial r^2} - 2 \frac{\partial f_{i+1,N}}{\partial r} (1 + r_i \Delta t_i)^2 \Delta t_i r_i' + 2 f_{i+1,N} - S_{\text{max}} (1 + r_i \Delta t_i)^3 (\Delta t_i r_i')^2 \\
\frac{\partial^2 f_{i,N}}{\partial t^2} &= \frac{\partial^2 f_{i+1,N}}{\partial t^2} + 2 (1 + r_i \Delta t_i)^2 r_i + 2 f_{i+1,N} - S_{\text{max}} (1 + r_i \Delta t_i)^3 r_i^2 \\
\frac{\partial^2 f_{i,N}}{\partial r \partial t} &= \frac{\partial^2 f_{i+1,N}}{\partial r \partial t} + \frac{\partial f_{i+1,N}}{\partial r} (1 + r_i \Delta t_i)^2 r_i' + (f_{i+1,N} - S_{\text{max}}) (1 - r_i \Delta t_i) r_i' \\
\frac{\partial f_{i,N}}{\partial \sigma} &= 0 \\
\end{align*}
\]

for the border nodes at \( S_{\text{max}} \) for a call option.

D.3.2 Put Option

For a put option we have the lower boundary \( \frac{\partial V}{\partial S} = -1 \), which gives \( f_{i,0} = f_{i+1,0} - S_{\text{min}} r_i \Delta t_i \) as was defined in (3.7). If we derive this equation we get

\[
\begin{align*}
\frac{\partial f_{i,0}}{\partial r} &= \frac{\partial f_{i+1,0}}{\partial r} - s_{\text{min}} + f_{i+1,0} (1 + r_i \Delta t_i)^2 \Delta r_i r_i' \\
\frac{\partial f_{i,0}}{\partial t} &= 1 + r_i \Delta t_i + S_{\text{min}} + f_{i+1,0} (1 + r_i \Delta t_i)^2 r_i' \\
\frac{\partial^2 f_{i,0}}{\partial r^2} &= \frac{\partial^2 f_{i+1,0}}{\partial r^2} - 2 \frac{\partial f_{i+1,0}}{\partial r} (1 + r_i \Delta t_i)^2 \Delta t_i r_i' + 2 f_{i+1,0} + s_{\text{min}} (\Delta t_i r_i')^2 \\
\frac{\partial^2 f_{i,0}}{\partial t^2} &= \frac{\partial^2 f_{i+1,0}}{\partial t^2} + 2 (1 + r_i \Delta t_i)^2 r_i + 2 f_{i+1,0} + S_{\text{min}} (1 + r_i \Delta t_i)^3 r_i^2 \\
\frac{\partial^2 f_{i,0}}{\partial r \partial t} &= \frac{\partial^2 f_{i+1,0}}{\partial r \partial t} + \frac{\partial f_{i+1,0}}{\partial r} (1 + r_i \Delta t_i)^2 r_i' + (f_{i+1,0} + S_{\text{min}}) (1 - r_i \Delta t_i) r_i' \\
\frac{\partial f_{i,0}}{\partial \sigma} &= 0 \\
\end{align*}
\]
\( \frac{\partial f_{i,0}}{\partial \sigma} = 0 \)
for the border nodes at \( S_{\text{min}} \) for a put option.
COPYRIGHT

The publishers will keep this document online on the Internet - or its possible replacement - for a period of 25 years from the date of publication barring exceptional circumstances. The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility. According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement. For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

UPPHOVSRÄTT

Detta dokument hålls tillgängligt på Internet - eller dess framtida ersättare - under 25 år från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår. Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphöva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det läsningar av teknisk och administrativ art. Upphovsmannens ideella rätt innehåller rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart. För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/