UPPSALA
UNIVERSITET

Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology 206

Accurate Finite Difference Methods for Option Pricing

JONAS PERSSON

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2006

ISSN 1651-6214
ISBN 91-554-6627-3
urn:nbn:se:uu:diva-7097

Dissertation presented at Uppsala University to be publicly examined in Sal 2446 Polacksbacken, Hus 2, Uppsala, Friday, September 29, 2006 at 10:15 for the degree of Doctor of Philosophy. The examination will be conducted in English.

Abstract
Persson, J. 2006. Accurate Finite Difference Methods for Option Pricing. Acta Universitatis Upsaliensis. *Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology* 206. 70 pp. Uppsala. ISBN 91-554-6627-3.

Stock options are priced numerically using space- and time-adaptive finite difference methods. European options on one and several underlying assets are considered. These are priced with adaptive numerical algorithms including a second order method and a more accurate method. For American options we use the adaptive technique to price options on one stock with and without stochastic volatility. In all these methods emphasis is put on the control of errors to fulfill predefined tolerance levels. The adaptive second order method is compared to an alternative discretization technique using radial basis functions. This method is not adaptive but shows potential in option pricing for one and several underlying assets. A finite difference method and a Monte Carlo method are applied to a new financial contract called Turbo warrant. A comparison of these two methods shows that for the case considered the finite difference method is superior.

Keywords: Finite differences, Option pricing, Adaptive methods

Jonas Persson, Department of Information Technology, Box 337, Uppsala University, SE-75105 Uppsala, Sweden

ISSN 1651-6214
ISBN 91-554-6627-3
urn:nbn:se:uu:diva-7097 (http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-7097)

Think different,
think finite differences.
From Sloganizer.net

List of Papers

This thesis is based on the following papers, which are referred to in the text by their Roman numerals.

I J. Persson and L. von Sydow (2003). Pricing European Multi-asset Options Using a Space-time Adaptive FD-method. To appear in *Computing and Visualization in Science*.

II P. Lötstedt, J. Persson, L. von Sydow and J. Tysk (2006). Space-Time Adaptive Finite Difference Method for European Multi-Asset Options. Submitted to *Computers and Mathematics with Applications*.

III G. Linde, J.Persson and L. von Sydow (2006). A Highly Accurate Adaptive Finite Difference Solver for the Black-Scholes Equation. Submitted to *International Journal of Computer Mathematics*

IV U. Pettersson, E. Larsson, G. Marcusson and J. Persson (2006). Improved Radial Basis Function Methods for Multi-dimensional Option Pricing. Submitted to *Journal of Computational and Applied Mathematics*

V J. Eriksson and J. Persson (2006). Pricing Turbo Warrants. Submitted to *Journal of Economic Dynamics and Control*.

VI J. Persson (2006). Pricing American Options Using a Space-time Adaptive Finite Difference Method. Submitted to *Mathematics and Computers in Simulation*

Contents

1. Summary in Swedish

Noggranna finita differensmetoder för prissättning av optioner

Det här är en kort svensk sammanfattning av avhandlingen **Accurate finite difference methods for option pricing** i ämnet numerisk analys. Avhandlingen har den svenska titeln **Noggranna finita differensmetoder för prissättning av optioner** och handlar om hur man med en dators hjälp snabbt och effektivt kan prissätta s.k. aktie*optioner*. I detta kapitel ska vi beskriva och ge en kort introduktion till vad aktieoptioner är och vad det innebär att prissätta dessa med hjälp av en dator. Dessutom kommer vi att kort beskriva den adaptiva metod vi har använt för att med så lite datorkraft som möjligt räkna fram ett resultat med stor noggrannhet. Men låt oss börja med att förklara vad en option är.

1.1 Aktieoptioner

En aktie kan enkelt sägas vara en andel i ett företag. Aktiens värde beror (bland annat) på företagets ekonomiska resultat, men också på hur andra *tror* att företaget kommer att prestera, vad marknaden tror om företaget o.s.v. Det handlar om förväntningar och följer man A-ekonomi på TV så har man garanterat hört uttalanden som "Ericsson sjönk idag på börsen efter en positiv kvartalsrapport som dock var sämre än vad marknaden hade förväntat sig", d.v.s det är inte bara företagets resultat som har betydelse för aktiepriset. Då och då blir vi också påminda om att även helt andra faktorer, som t.ex. en naturkatastrof, kan påverka värdet av en aktie.

Ibland köper man aktier för att man vill äga en del i ett före-

tag och få bestämmanderätt över företaget men vanligare är nog att man köper och säljer aktier enbart för att tjäna pengar på det. Annars skulle nog inte omsättningen på aktiemarknaden vara i närheten av så stor som den är idag.

Vad är en option?

En option är ett finansiellt kontrakt som ger ägaren av kontraktet rätten att köpa eller sälja en aktie under speciella villkor. I kontraktet brukar det stå vilken aktie det är frågan om, till vilket pris man har rätt att köpa/sälja, s.k. *lösenpris*, och när man får köpa eller sälja, det som brukar kallas för *lösendatum*. Det finns också mer komplicerade kontrakt men här kommer vi att hålla oss till dessa enkla.

När vill man ha en option?

Låt oss illustrera med ett exempel med en bagare. Tänk dig ett bageri som bakar och säljer stora mängder bröd året om. Bagaren vet att om skörden av vete blir dålig ett år så kommer priset på mjöl att stiga kraftigt. Problemet för bagaren är att om hon inte kan höja priset på sina produkter som hon säljer lika mycket som prisökningen på mjöl, då finns det risk att vinsten sjunker. Ett högt pris på mjöl betyder att bagarens vinst kommer att sjunka. En option på att köpa mjöl skulle kunna fungera så här: Optionen ger bagaren rätt att köpa en viss mängd mjöl till ett fast pris vid en av henne bestämd tidpunkt i framtiden. Det skulle ge henne tryggheten att kunna köpa mjöl till ett rimligt pris om veteskörden skulle bli dålig och mjölpriset stiger. Denna trygghet, denna rätt att köpa till ett visst fast pris har naturligtvis ett värde. Bagaren får betala en liten premie, ungefär som en försäkringspremie, för denna rätt. Om priset på mjöl inte stiger utan istället sjunker under det pris som hon enligt kontraktet har rätt att köpa för, då använder hon naturligtvis inte sin rätt. Förlusten är begränsad till storleken på den premie hon har betalat. Detta är ett exempel på hur man kan använda en option som ett slags försäkring. Den forskning som presenteras i denna avhandling handlar inte om optioner på att köpa mjöl utan aktier men samma resonemang gäller, även om de som handlar med optioner allt som oftast faktiskt inte använder optionen till att köpa eller sälja aktien utan bara som ett handelsinstrument i sig.

1.2 Numerisk lösning av differentialekvationer

Många fenomen i naturen och inom tekniken kan beskrivas med hjälp av *differentialekvationer*. Differentialekvationer kan t.ex. beskriva hur vågor sprider sig på en vattenyta (vågekvationen) eller hur värmen sprider sig från en kamin i ett rum (värmeledningsekvationen). Båda dessa exempel finns det fysikaliska modeller för och genom att hitta approximativa (numeriska) lösningar till differentialekvationerna kan man förutsäga hur vågorna och värmen utbreder sig i dessa fall. En liknande differentialekvation kan också användas för att beräkna priset på en option trots att de bakomliggande modellerna ser väldigt olika ut. De som först visade en differentialekvation som kan användas för prissättning av optioner var Black och Scholes [3]. Detta var år 1973. Samma år arbetade Merton med liknande idéer och år 1997 fick han tillsammans med Scholes "Sveriges Riksbanks pris i ekonomisk vetenskap till Alfred Nobels minne", det som ibland kallas Nobelpriset i ekonomi. Black hade tyvärr gått bort några år tidigare och kunde inte dela priset med dem.

Liksom våg- och värmeledningsekvationen kan Black&Scholes differentialekvation ibland lösas analytiskt, dvs det finns en formel som man kan uttrycka lösningen med. Men så är dock inte alltid fallet och i de fall det *inte* är möjligt kan man ta till s.k. *numeriska metoder*. Det finns en uppsjö av dessa, och att välja rätt metod för rätt problem och tillfälle är en vetenskap i sig. Här kommer den metod som till största delen använts i denna avhandling att beskrivas, den s.k. *finita differensmetoden*. Man ska komma ihåg att numeriska metoder inte kan lösa dessa differentialekvationer exakt utan bara approximativt, d.v.s. ungefärligt. Det betyder att förutom det fel man gör när man modellerar verkligheten med en differentialekvation så kommer man också att göra ett fel i sina beräkningar. När man använder finita differensmetoden så delar man in det område där man ska göra sina beräkningar, *beräkningsområdet*, i ett antal *nätpunkter* och väljer att studera lösningen till differentialekvationen bara i dessa nätpunkter. Att konstruera en approximation av differentialekvationen i dessa punkter kallas att *diskretisera* problemet och det finns många andra metoder än finita differensmetoden för att göra just detta delsteg. Denna diskretisering ger upphov till ett stort antal *algebraiska*

ekvationer. Vilka algebraiska ekvationer man får och deras egenskaper beror på vilken diskretisering man har gjort, men det ska vi inte gå in på mer här. För att hitta lösningen till dessa ekvationer måste ofta stora linjära *ekvationssystem* lösas. Detta är en sak som en dator är mycket lämpad för eftersom det är en stor mängd (ofta många miljoner) enkla och systematiska räkneoperationer som behöver utföras, något datorer är väldigt bra på. Det fel man gör i diskretiseringen av ekvationen kallas för *diskretiseringsfel* och det är viktigt att det minskar om man använder fler och fler nätpunkter. Enkelt förklarat kan man säga att ju fortare dikretiseringsfelet minskar desto bättre är metoden. De värden man har i nätpunkterna kommer vi här att kalla för den numeriska lösningen till differentialekvationen.

1.3 En adaptiv finit differensmetod

I stora delar av forskningen har vi arbetat med *adaptiva metoder* där man innan man utför beräkningarna får ange hur mycket diskretiseringfel man vill ha och sedan ska den adaptiva metoden placera nätpunkter i beräkningsområdet på ett sådant sätt att diskretiseringsfelet blir av den storlek som man har angett. Kort kan man beskriva den adaptiva metoden så här:

Steg 1 Gör en enkel diskretisering med få nätpunkter och beräkna den numeriska lösningen i dessa nätpunkter.

Steg 2 Använd den numeriska lösningen och uppskatta hur mycket diskretiseringsfel man har gjort i varje nätpunkt.

Steg 3 Skapa en nytt adaptivt nät sådant att när det används ska diskretiseringsfelet bli så litet som man har angett.

Steg 4 Gör den adaptiva diskretiseringen och beräkna den numeriska lösningen i de nya nätpunkterna.

I Steg 2 ovan uppskattas det diskretiseringsfel som gjorts i Steg 1. Detta kan man göra genom att jämföra två numeriska lösningar med varandra, en beräknad med ett visst antal nätpunkter och en annan beräknad med bara hälften så många nätpunkter. Tack var att man på förhand vet ungefär hur diskretiseringsfelet ser ut kan man sedan

med dessa två numeriska lösningar uppskatta diskretiseringsfelet i den numeriska lösning man beräknade i Steg 1. När man har den uppskattningen kan man gå vidare och göra en ny indelning i nätpunkter som ger det diskretiseringsfel man vill ha.

Att kunna bestämma innan man utför en numerisk beräkning hur mycket fel man är beredd att göra har flera förtjänster. Om man vill ha en viss storlek på felet men inte har något sätt att avgöra på förhand vilken diskretisering som ger just det felet så måste man prova sig fram, skapa en diskretisering, utföra beräkningarna, mäta hur mycket diskretiseringsfel det blev och sedan skapa en ny diskretisering och prova igen. Vill man sedan inte ha en jämn fördelning av nätpunkterna i sin diskretisering måste man samtidigt lösa problemen "Hur många nätpunkter behövs?" och "Var skall dom placeras?". Detta är tidskrävande och inte så effektivt. Den adaptiva metoden kan hjälpa till i sådana situationer. Dessutom använder den datorns minne effektivitet eftersom den åstadkommer ett givet fel med en bra placering av nätpunkterna.

1.4 Avslutning

Avslutningsvis bör påpekas att avhandlingen egentligen inte alls handlar om de faktiska optionspriser som beräknas. Syftet med avhandlingen är att studera och utveckla olika numeriska metoder för att prissätta aktieoptioner. När man utvecklar nya numeriska metoder är det ofta fördelaktigt att studera enkla modellproblem, ibland till och med sådana där man känner till en exakt lösning. Att ha en exakt lösning att jämföra sin numeriska metod med är ett ypperligt redskap när man studerar den numeriska metodens egenskaper. Genom att jämföra med en exakt lösning kan man snabbt få ett besked om kvalitén på den numeriska lösningen, något som annars kan vara lika svårt som att faktiskt lösa ett mer komplicerat problem. Exempel på olika typer av optioner som avhandlingen behandlar är: Europeiska optioner på en eller flera underliggande aktier, Amerikanska optioner med och utan stokastisk *volatilitet* (stokastisk volatilitet innebär i princip att ett mer slumpmässigt beteende antas oss aktien) och Turbo warranter. Det sistnämnda är en speciell sorts aktieoption vars egenskaper förändras

om aktiens värde når en viss nivå.

Förutom finita differensmetoder har även s.k. *Monte Carlo* metoder och *Radiella basfunktioner* använts för att beräkna optionspriser. Den förstnämnda är en metod där man slumpar fram tänkta slumpmässiga aktiebanor och sedan tar ett medelvärde av alla dessa. Den senare är en alternativ diskretiseringsmetod som har stor potential när det gäller optionsprisberäkningar. Den är dock ganska ny och mer forskning om dess egenskaper och möjligheter behöver utföras i framtiden.

2. Introduction

2.1 Different types of options

Options are financial contracts that gives the holder certain rights. As a holder you buy the rights stipulated in the contract. It can be the right to buy or sell something, but it can also be the right to exchange one thing for another. There are many different kinds of contracts on many different kinds of underlying assets. Perhaps one of the most common contracts are the stock options, contracts that allow you to buy or sell a stock. Two of the simplest type of such contracts that are traded are:

- *European options* - Gives the holder the right to buy/sell the underlying asset at a certain time (usually called time of maturity) for a certain price (usually called strike-price).
- *American options* - Gives the holder the right to buy/sell the underlying asset at any time before the time of maturity for a certain price.

Options that give the right to buy something are called *call* options and those that give the right to sell something are called *put* options. The American option gives the holder more freedom when to use the option and is hence often a little bit more expensive than a corresponding European option. There are numerous of other more exotic options on the market like Asian, Parisian, Bermudan and so on. Here we will briefly introduce also the *barrier option* and the *Turbo warrant* since some basic knowledge about these contracts will be needed later on.

Barrier options are like regular options with the feature that a special action occurs when the stock hits a barrier. An example is the European barrier call option. Assume that we have a European option with strike-price K and a barrier $b \leq K$ and that the value of the underlying asset S at time $t < T$ is $s > b$. If the value of the stock at any time prior to T hits the barrier b, the option expires worthless immediately. If the stock does not reach the barrier the value at time T is $\max(S(T) - K, 0)$. This type of contract is called a *down-and-out*

barrier option. Then there are of course a similar *up-and-out* barrier option and options that come to life when the barrier is hit, these are known as *down-and-in* and *up-and-in*. We will not elaborate more on this kind of contracts here, one can read more on the subject in [2] for mathematical formulations and in e.g. [51] and [58] for the numerical treatment. In a more general case the barrier can be time-dependent.

In early 2005 the French bank Societé Generale (SG) issued a contract which they named Turbo warrant. The name had been used earlier in Germany for a regular barrier option of down-and-out style with the barrier at the strike-price. The special features with the Turbo warrant from SG is that it is a barrier down-and-out option that has the barrier in-the-money (this means that $b > K$) and that it pays a rebate to the holder if the barrier is hit. The rebate can be considered as a new contract, a call option on the minimum process of the stock. This contract was mathematically priced with geometric Brownian motion as underlying stochastic process by Eriksson in [11]. In Paper V we numerically priced such contracts and allowed for more general underlying processes. The Turbo warrants can be priced as a combination of options, the barrier down-and-out and barrier down-and-in call described earlier.

2.2 Buy the stock or an option on the stock?

Why would anyone want to buy a right to buy an asset (like a stock) instead of actually buying the asset? There are several answers to that question. Options are in most cases used only for the purpose of speculation and the intention of the holder is most often not to actually buy or sell the stock. Options on stocks are used as instruments for the possibility to make more money than dealing with the stocks directly. Why is that? Well, let us give an explanatory example:

Assume that a stock you are interested in has the price 100 on the market and that you think the price is about to rise. If you buy the stock and it rises to 110 then you can sell your stock and the gain is 10 (ignoring trading costs such as courtage). This is depicted in Figure 2.1. However, if you instead buy an option for 10 that gives you the

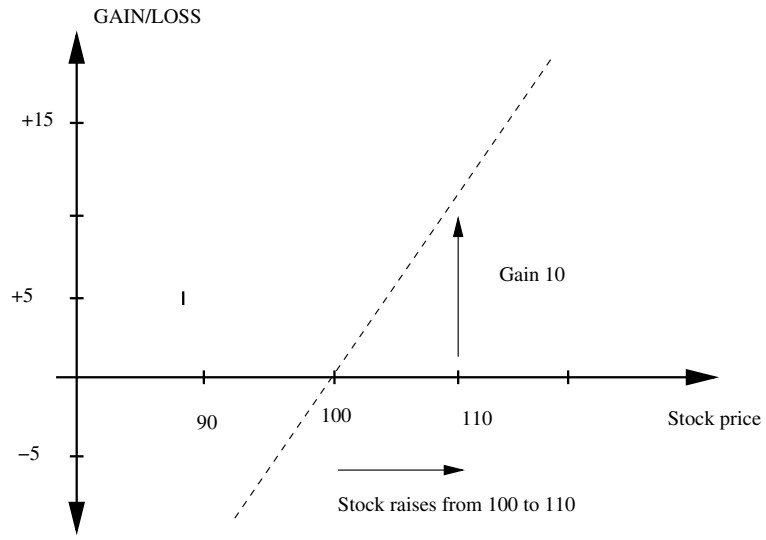

Figure 2.1: Gain when buying the stock.

right to buy a stock for 100 the situation is different. If the stock doesn't rise above 110 you will not make any money. Your maximum loss is 10 since that is what you lose if your option expires worthless. If you had bought the stock you would still have something of value when it falls of course, but you would have had to invest 100 instead of only 10. The interesting thing though is when the stock goes above 110, say to 115. If you hold the stock you would have gained 15 which is 15% of what you started with, but if you hold the option you can use it and gain 5 which is 50% of what you started with. Using options instead of stocks is a way to expose oneself to higher risk (you can lose all of your investment) but at the same time have a chance to gain more. This can be seen in Figure 2.2. Another use of options is to use it as an insurance. In such a case the holder actually intends to use the option to buy or sell the physical product. This is very common on the commodity market.

2.3 Background and our work

The financial market has been an ever expanding economical field over the last few years. The value of the financial assets traded on the stock

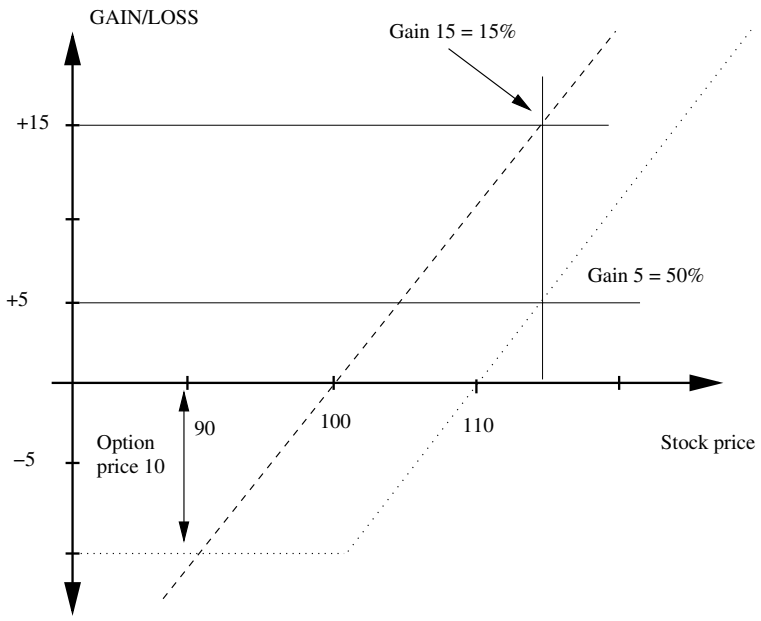

Figure 2.2: Gain when buying the option.

markets all over the world has reached astronomical amounts. Even though the financial industry has been around for more than 30 years it got a huge boost with the Nobel-prize awarded work by Black and Scholes in 1973 [3] who were the first to show a differential equation that could be used to give a "fair" price to an option. This was a break through on the financial scene since it allowed for consistent option pricing. The same year Merton worked on similar problems and he shared the prize with Scholes in 1997. Black had unfortunately past away a few years earlier.

Since then the financial theory has evolved into more complex models for options and many new and more complicated financial contracts have been developed. For the simplest case of a European call option in the Black&Scholes model Black and Scholes also derived an analytical formula for the price but for more complicated contracts and more involved models there are no such formulas. In such cases numerical means of pricing the contracts are necessary. This is the

research field we are working in, we work on the numerical pricing of options. Mainly with the numerical solution of PDEs using finite differences, see e.g. [53] or [51], but also with other methods. The finite difference method will be introduced in Sec. 4.3.1.

An example of a more complicated contract is when the option has several underlying assets. With better computers these contracts have been more and more popular the last few years. A call option on e.g. the mean value of two stocks give you the right to buy a combination of the involved stocks for a certain price. Monte Carlo methods, see Sec. 4.1, are often used to price this kind of contract. They have rather slow convergence properties and when high accuracy is of interest for problems with few underlying assets finite differences in the PDE setting often outperforms them. In the PDE setting, pricing these contracts requires the solution of a multi-dimensional PDE. This is a very computationally demanding problem and it is often described as having "a curse of dimensionality". The reason is that with a standard discretization the numerical method typically requires N^d grid-points to discretize a d-dimensional problem. It is not hard to see that a large number of grid-points will be needed for just a few underlying assets. For this reason Monte Carlo methods are still the only alternative for many dimensions, despite their slow convergence. For moderate sized problems the sparse grid technique, see Sec. 4.3.2, is a new alternative.

We have studied an adaptive approach to reduce the number of grid-points in this problem. The basic and simple idea is to place few grid-points adaptively to make the computational problem smaller while keeping a high accuracy in the solution. Using the adaptive method it is possible to control the discretization errors which is also very important. In paper I-III we do this for European options and in paper VI we consider American options. In paper I we studied local discretization errors and in paper II we approached the global error in the solution. The ideas from paper I were combined with highly accurate discretizations to reduce the number of grid-points even further in paper III. In paper IV we used radial basis functions (RBFs), see [13], as approximants to price an option on several underlying assets. The RBF method was compared with the adaptive finite difference method used in paper I. In paper V we price the new contract

Turbo warrant that was introduced in 2005. This more complicated option is a combination of a regular barrier option on the stock and a barrier option on the minimum process of the stock. Two numerical methods, a finite difference method and a Monte Carlo method, are compared regarding efficiency. Paper VI is a study of the adaptive FD method used in previous papers applied to American options with and without stochastic volatility.

3. Option pricing

In this section we will introduce some basic financial mathematics. We will use the original Black-Scholes model to keep the presentation simple. For a real-world application a more advanced model is probably necessary.

3.1 European options

In their award-winning work [3] Black and Scholes derived a boundary value partial differential equation (PDE) for the value $F(t,s)$ of an option on a stock. This value $F(t,s)$ solves the Black&Scholes PDE

$$\begin{aligned} \frac{\partial F(t,s)}{\partial t} + rs\frac{\partial F(t,s)}{\partial s} + \frac{1}{2}s^2\sigma^2(t,s)\frac{\partial^2 F(t,s)}{\partial s^2} - rF(t,s) &= 0, \\ F(T,s) &= \Phi(s), \end{aligned} \tag{3.1}$$

in $[0\ T] \times \mathbb{R}^+$. Here r is the interest rate, σ is the volatility of the underlying asset S and $\Phi(s) = \max(s-K,0)$ is the contract function. According to the Feynman-Kac theorem (see e.g. [2]) the solution $F(t,s)$ to such a PDE can be represented as an expected value

$$F(t,s) = e^{-r(T-t)}E_{t,s}[\Phi(S(T))], \tag{3.2}$$

where the underlying stock $S(t)$ follows the dynamics

$$dS(u) = rS(u)du + S(u)\sigma(u,S(u))dW(u), \tag{3.3}$$

$$S(t) = s. \tag{3.4}$$

Here W is a Wiener process (see e.g. [2] for a definition) and $E_{t,s}$ in Eq. (3.2) denotes the expected value where S starts in s at time t, see Eq. (3.4). For the purpose of option pricing we thus should assume that the underlying stock follows this dynamics even if we in reality do not the expect the value of the stock to grow with the interest rate r.

For the special case of a European call option with contract function $\Phi(s) = \max(s-K,0)$ and constant volatility and interest rate there is a formula for the value $F(t,s)$ known as the "Black–Scholes formula",

$$\begin{aligned} F(t,s) &= s\mathcal{N}(d_1(t,s)) - Ke^{-r(T-t)}\mathcal{N}(d_2(t,s)), \\ d_1(t,s) &= \frac{ln(s/K)+(r+\frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}}, \\ d_2(t,s) &= d_1(t,s) - \sigma\sqrt{T-t}. \end{aligned} \tag{3.5}$$

Here $\mathcal{N}(x)$ is the cumulative normal distribution function. This formula will in some cases be used as a reference solution in the numerical experiments in this thesis where we use both the PDE Eq. (3.1) and the stochastic representation Eq. (3.2) to price options numerically.

3.2 The transformed PDE

Note that Eq. (3.1) is a final value problem. Most people use the transformation $\bar{t} = T - t$ to get an initial value problem in $\bar{t}$ of more standard type. It is used in e.g. [51]. For ease of notation we will also in transformed time denote time by t. The PDE (3.1) is transformed to

$$\begin{aligned} \frac{\partial F(t,s)}{\partial t} &= rs\frac{\partial F(t,s)}{\partial s} + \frac{1}{2}s^2\sigma^2(t,s)\frac{\partial^2 F(t,s)}{\partial s^2} - rF(t,s) := \mathcal{L}F, \\ F(0,s) &= \Phi(s). \end{aligned} \tag{3.6}$$

Note that we here also define what will be referred to as the Black&Scholes operator $\mathcal{L}$. The multi-dimensional version for several underlying assets is stated in Sec. 6.1, but we will here limit ourselves to the one-dimensional case. If transformations in time are used by most authors the transformation in space on the other hand is not straight forward to choose. Popular transformations include the log-transform $x = log(s)$ that gives a PDE with constant coefficients, see e.g. [5], and the transformation to the heat equation, see e.g. [53, 36]. Some people argue for the log-transform and some recommend it only for theoretical use [30]. It creates an infinite domain from a half-infinite domain which introduces another far-field boundary. However, it removes the potentially large variable coefficients in the PDE which

could improve some numerical schemes. What transformation to use does not seem to be so important though, Achdou and Pironneau [1] answers the question whether the log transformation should be used or not with: "*... it does not matter, what is important is the grid.*". They state that what is important is to place grid-points close to the singularity of the pay-off function. This is precisely what the adaptive methods presented in paper I-III and VI do.

We have in our work always used the reversed time transformation and in some cases also transformations in space and/or time.

3.3 American options

Pricing American contracts is in some ways very similar to pricing European contracts but in others very different. The most apparent difference is the early exercise constraint for the American option. While the European contracts only can be exercised at the pre-defined time T (see Sec. 2) the American contracts can be exercised at any time $t \leq T$. This means that the price of the American option will always be higher than what you get by exercising the option. This give rise to what is usually known as an internal *free boundary*. We will here give an introduction to the American option pricing problem in the PDE setting.

With the Black-Scholes operator $\mathscr{L}$ defined above in Eq. 3.6 we can formulate the American option pricing problem as a linear time dependent complementarity problem (LCP), here for one underlying asset and constant volatility.

$$\begin{cases} (\frac{\partial}{\partial t} - \mathscr{L})F \geq 0, \\ F \geq \Phi, \\ (F-\Phi)(\frac{\partial}{\partial t} - \mathscr{L})F = 0. \end{cases} \tag{3.7}$$

Here the function Φ is the contract function (for the American put) defined as $\Phi(s) = \max(K-s, 0)$. Note that these constraints mean that in one region the price obeys the Black-Scholes equation and in the other

region the contract function holds. The two regions are often referred to as the continuation region and the stopping region and refers to that you either continue to hold the option or decide to stop and cash it in. These regions as well as the free boundary, denoted by b^*, is displayed in Fig. 3.1. For the simplest case of constant parameters and no dividends it is possible to show [2] that the European and the American option price coincides for the American call option.

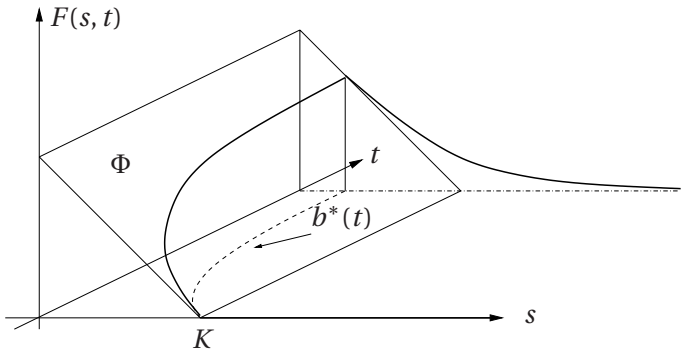

Figure 3.1: The free boundary $b^*(t)$ in the s-t-plane. In the continuation region $s > b^*(t)$, in the stopping region $s \leq b^*(t)$.

4. Pricing European options

As mentioned in Sec. 3.1 we can sometimes price a financial contract using analytic formulas. However, for complicated contracts and in more general settings the analytical formulas are seldom there to help. In such cases the use of a numerical method is not only useful but necessary. In this section we will give an introduction to some numerical methods that are used to price financial contracts. Despite our previous comment that for simple contracts there are analytical formulas we will here use the European call option on one underlying stock as an example. Having a basic understanding of this simple example will be useful in understanding the more advanced methods studied in this thesis.

4.1 Monte Carlo methods

The Monte Carlo method is a stochastic method that finds the option value by approximating the expected value in Eq. (3.2). Here we will give an introductory example of how to price a simple European call option in the Black&Scholes model using the Monte Carlo method. Then we will give examples of how the efficiency of the method can be improved.

We model the behavior of the asset with the dynamics defined in Eq. (3.3) and want to know the price $F(t,s)$ of the option today at time $t = 0$ given that the value of the stock today is s. For a European call option the contract function is $\Phi = \max(S(T) - K, 0)$ where $S(T)$ is the value of the stock on the expiry date. In this setting the solution to Eq. (3.3) is

$$S(T) = S(0)\exp\left(\left[r - \frac{1}{2}\sigma^2\right]T + \sigma W(T)\right),$$

where $S(0) = s$ is todays stock price and the value of the Wiener process $W(T)$ is a random variable that is normally distributed with mean 0 and

variance T. Using a standard normal random variable Z with mean 0 and variance 1 we can replace $W(T)$ with $\sqrt{T}Z$ that will have the correct distribution and we can calculate $S(T)$ as

$$S(T) = S(0)\exp\left([r - \frac{1}{2}\sigma^2]T + \sigma\sqrt{T}Z\right). \tag{4.1}$$

Now we need two things, an algorithm for generating the random numbers Z and a way to compute the expected value. Generating random numbers is an important part of the algorithm and care should be given to it. Many Z might be needed and the speed of the number generator is very important. See e.g. the book [15] by Glasserman for more information about the Monte Carlo method and number generators. The expected value is approximated by taking the mean of N realizations of $S(T)$ as given in Eq. (4.1) above,

$$E(\Phi(S(T))) \approx \frac{1}{N}\sum_{\omega=1}^{N}\Phi(S_\omega(T)) = C_N,$$

where $S_\omega(T)$ is the value computed with the random number Z_ω. The estimate C_N is strongly consistent which means that C_N will go towards the true value with probability 1 as $N \to \infty$. The convergence of the error of this method is $\mathcal{O}(1/\sqrt{N})$. For large N it is possible to provide a confidence interval for the error of the estimate. For details on this see [15].

In this simple example the option is not dependent on the entire path of the stock which allows for the shortcut to directly compute the value $S(T)$ at time T. For many options this is not possible since the value depends on the whole trajectory of the stock from time zero. In such a case one must approximate the trajectory with a numerical procedure. The simplest method for doing this is the Euler scheme. Dividing the time interval $[0\ T]$ into intervals of length Δt we can simulate $S(T)$ in Eq. (4.1) by

$$S(t + \Delta t) = S(t) + rS(t)\Delta t + \sigma S(t)\sqrt{\Delta t}Z,$$

with Z as before. Each such trajectory will give us one value of $S(T)$ and then we can compute an approximation to the expected value in the same way as was described earlier. The Euler scheme has strong

convergence of order $1/2$.

One way of improving the efficiency of the Monte Carlo method is to use some kind of variance reduction technique. Some examples of variance reduction techniques are: Control variates, Antithetic variables, Stratification and Importance sampling. These are discussed in detail in [15] and [8]. In [41] the authors study variance techniques for pricing baskets of several underlying assets. Another way to improve the Monte Carlo method is to use so called quasi random numbers, numbers that are not random at all. For an introduction to such methods see e.g. [15] and [8]. One way to create the normally distributed numbers needed is to generate uniformly distributed numbers (using some method) in $[0\ 1]$ and then use a transform to get random numbers with normal distribution. Numbers in $[0\ 1]$ can be taken from a quasi-random sequence. The idea with quasi-random numbers is that they will "fill" the interval $[0\ 1]$ in a predictable way. The generated numbers are distributed in such a way that they are prevented from being too close to each other. Examples of quasi-random number generators are the Halton-, Faure- and Sobol-sequences.

Pricing options of American type is a viable but fairly complicated using the Monte Carlo method, at least compared to pricing European options. This is because one must typically first solve an optimal stopping problem to find the optimal exercise rule and then compute the expected discounted pay-off using this rule. However, there are examples of Monte Carlo methods for American options in e.g. [15] and [46].

4.2 Lattice methods

Lattice methods were introduced by Cox, Ross and Rubinstein in 1979 in [10] and mimic a discrete random walk of the underlying stock. These methods include numerical algorithms such as bi-/trinomial tree methods. We will here give an outline of how to price a European option using a binomial tree method. This introduction follows the one given in [53].

Let us start with the assumption that we live in a risk-neutral world in which the underlying asset follows a log-normally distributed random path and we want to price a standard European call option. The time interval [0 T] is divided into M intervals of length $\Delta t = M/T$. Assume now that at time $m\Delta t$ the value of the stock is S_m. From this value we now assume that it can either make a jump up of size u with probability p or make a jump down of size d with probability $1-p$, see Fig. 4.1. For the sizes of the jumps an often used condition is that $u = 1/d$. Note that we here, for simplification of the presentation, assume that the sizes and probabilities of the jumps are constant over time. We could allow them to be time-dependent. Going from time zero to time T like this will form a lattice (or tree).

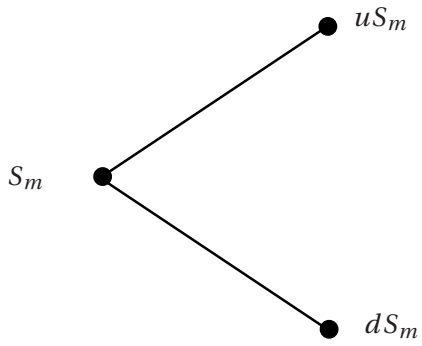

Figure 4.1: Going from node S_m to the next nodes.

With this setting it is possible to start from the known stock value at time zero and compute stock values and jump probabilities for all nodes in the tree. The value of the option is known at time T, it is the contract function evaluated in the nodes. When this step has been completed we can use these known option values at time $T = m\Delta t$ and start computing the option values one step backwards in time, at $(m-1)\Delta t$, using the jump probabilities that was computed when the tree was formed. This is repeated all the way back to time zero and will give us the value of the option.

Lattice methods are popular among practitioners because of their intuitive way of computing the price. Some of the work continuing the pioneering work of Cox, Ross and Rubinstein will be mentioned here. Heston and Zhou studied the convergence of multinomial models in [22]. Tian showed in [52] an improved convergence for a binomial model with a "tilt" parameter that skews the binomial tree . In [14] Gerbessiotis implemented a parallel trinomial tree based algorithm for the computations needed for pricing e.g. an American option.

4.3 PDE methods

Here we will briefly mention some of the discretization techniques to price European options with PDEs that have been used by authors over the years. All of them are of great interest since they can all be used as building blocks for pricing American options. Being able to numerically price the European option efficiently is often a good start at developing methods for the more complicated American option. The discretization techniques mentioned here can be combined with techniques presented in Sec. 5 for American options.

4.3.1 Finite difference methods

Let us here introduce the standard second order centered finite difference discretization of the Black&Scholes operator $\mathscr{L}$ together with a discretization of the time derivative. See Sec. 3.2 for the PDE. Since the domain in space is unbounded we truncate it at s_{max} for the numerical computations. Then we divide $[0\ s_{max}]$ into $M+1$ equally spaced gridpoints $s_i, i = 0,1,\ldots,M$. The space-step is $h = s_{max}/M$, see Fig. 4.2.

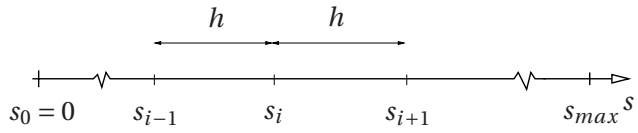

Figure 4.2: Space discretization.

The first and second derivatives in space at the point (t, s_i) can be approximated with finite differences. Introducing the notation $F_i = F(s_i)$ we get

$$\frac{\partial F(t,s_i)}{\partial s} = \frac{F_{i+1}(t) - F_{i-1}(t,)}{2h} + \mathcal{O}(h^2), \tag{4.2}$$

$$\frac{\partial^2 F(t,s_i)}{\partial s^2} = \frac{F_{i+1}(t) - 2F_i(t) + F_{i-1}(t)}{h^2} + \mathcal{O}(h^2), \tag{4.3}$$

where $\mathcal{O}(h^2)$ denotes the discretization error of order 2. To solve the PDE numerically we will also need boundary conditions. This is something that is very important for the numerical solution but here we will not go into any details. Let us assume that the solution in the s-direction is nearly linear near the boundaries so that we can extrapolate the values F_0 and F_M from the values next to the boundary. This implies that

$$F_0 = 2F_1 - F_2, \tag{4.4}$$

$$F_M = 2F_{M-1} - F_{M-2}. \tag{4.5}$$

Next we transform our PDE to a system of ordinary differential equations with the unknowns $F_i, i = 0, 1, \ldots, M$. Letting $\mathbf{F} = [F_0\, F_1 \ldots F_{M-1}\, F_M]^T$ and ignoring the truncation errors we end up with the system

$$\frac{d\mathbf{F}}{dt} = A\mathbf{F} \tag{4.6}$$

where A is the so called finite difference matrix illustrated below. The matrix A will be a structured and very sparse matrix. Including the boundary conditions above and collecting the terms we find that A will be

$$A = \begin{pmatrix} 1 & -2 & 1 & 0 & \ldots & 0 & 0 \\ \beta_1 & -\gamma_1 & \alpha_1 & 0 & \ldots & 0 & 0 \\ 0 & \beta_2 & -\gamma_2 & \alpha_2 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & 0 & 0 & \beta_{M-1} & -\gamma_{M-1} & \alpha_{M-1} \\ 0 & 0 & 0 & 0 & 1 & -2 & 1 \end{pmatrix}$$

with

$$\alpha_i = \frac{1}{2}\left(\frac{\sigma^2}{h^2} + \frac{r}{h}\right), \quad \beta_i = \frac{1}{2}\left(\frac{\sigma^2}{h^2} - \frac{r}{h}\right), \quad \gamma_i = \frac{\sigma^2}{h^2} + r.$$

Dividing also time in equally spaced points between 0 and T and denoting them by t_n we can use the practical notation $\mathbf{F}_i^n = F_i(t_n)$ and denote a time-step by $k = t_n - t_{n-1}$. Then the simplest one-step method is the explicit Euler method

$$\mathbf{F}^n = (I + kA)\mathbf{F}^{n-1},$$

for advancing the solution one time-step. An alternative one-step method is the implicit Euler backward

$$(I - kA)\mathbf{F}^n = \mathbf{F}^{n-1},$$

which requires the solution of a linear sparse system of equations to find $\mathbf{F}^n$. The initial condition $\mathbf{F}^0$ is given by the contract function. Many other time discretizations can be used. In this thesis we have mainly used the implicit *Backward differentiation formula* of order 2 (BDF2) [20]

$$\begin{aligned}
&\alpha_0^{n+1}\mathbf{F}^{n+1} = k_{n+1}A(\mathbf{F}^{n+1}) - \alpha_1^{n+1}\mathbf{F}^n - \alpha_2^{n+1}\mathbf{F}^{n-1} \\
&\alpha_0^{n+1} = (1+2\theta^{n+1})/(1+\theta^{n+1}), \\
&\alpha_1^{n+1} = -(1+\theta^{n+1}), \\
&\alpha_2^{n+1} = (\theta^{n+1})^2/(1+\theta^{n+1}),
\end{aligned} \tag{4.7}$$

which is an unconditionally stable method. Note that $\theta^{n+1} = \frac{k_{n+1}}{k_n}$, where k_n is the time-step between t_{n-1} and t_n.

Using finite differences for the discretization is a common method that have been used by many authors. For an introduction to these methods we can recommend the books [47, 51, 53]. They all introduce the concept of finite differences for option pricing and give the basic knowledge needed for a simple implementation of the method. For more advanced readers we suggest [50] with more and [19] with much more analysis of finite difference methods in general.

4.3.2 Sparse Grids

The sparse grids method has been known under different names such as 'Hyperbolic crosspoints', 'splitting extrapolation' or as boolean sum of grids. The first reference was probably Smolyak [48]. In 1991 Zenger [56] introduced sparse grids in finite element calculations to reduce the number of degrees of freedom. Zenger derived a higher-dimensional

multiscale basis as a tensor product construction of a one-dimensional multiscale basis. A standard discretization method typically (e.g. finite differences) needs $\mathcal{O}(N^d)$ grid-points to discretize a d-dimensional hypercube with N grid-points in each dimension but the sparse grid method due to its construction only uses $\mathcal{O}(N(\log(N))^d)$ with N points along each coordinate axis at the boundary. This makes the sparse grids method an interesting candidate for high dimensional problems. Not long after Zengers paper was published Griebel [17, 18] introduced a combination technique that implement the sparse grid idea of combining the numerical solution on several simple grids to form an accurate solution that can be computed in an efficient way. Since the first ground-breaking articles many authors have used this technique successfully, see e.g. [32] for advection problems and the thesis [45] on option pricing. For a survey of the fundamentals and applications of sparse grids the reader is referred to [7].

4.3.3 Radial Basis Functions

Radial basis functions (RBFs) are global approximants that can be used to approximate the solution to the Black&Scholes PDE. There are several types of global basis functions to choose from and using infinitely smooth functions the method has spectral order of accuracy. For this reason it is a good candidate for an efficient discretization method for options on several underlying assets. The RBF method is mesh-free since the only thing needed to compute the approximations is the relative distances from a node-point to all other node-points. This is a big advantage for obvious reasons but also a drawback since the linear systems that must be solved using an implicit time-stepping algorithm are dense. Because of the spectral accuracy though one can hope that few node-points are needed. For more details on this method, see [43].

RBFs for option pricing have been studied by several others including Fasshauer et al. in [13] for multi-asset European and American options. Hon [23] and Hon and Mao [24] also use RBFs for American options.

4.3.4 Adaptive methods

Adaptive methods aim at controlling the error in the solution in one way or the other. We have used both space- and time-adaptive finite difference methods in several of the papers presented in this thesis. Here we will give some examples of others that have used adaptive methods for numerical option pricing. However, papers about adaptive methods are not so frequent in option pricing literature. Adaptive time-stepping have been proposed by some authors but not so many use space-adaptive methods. Examples though can be found in Achdou and Pironneau [1] and Pironneau and Hecht in [44] who use a space-adaptive finite element method for the discretization of the Black&Scholes PDE. Another example using space- and time-adaptivity is Vande Wouwer et al. [54] who move a fixed number of grid-points around in the grid for every time-step. In [47] Seydel propose a technique similar to Richardson extrapolation for finding the number of grid-points that is needed to solve the problem to a given accuracy. This could be considered as an adaptive method even though it only aims at finding the number of equally spaced grid-points that is needed.

Another approach used by some authors can be exemplified by Clarke and Parrott [9] who use a $\sinh^{-1}(x)$ transform of the space coordinate to create a grid-stretching that places grid-points close to the strike-price. This is not an adaptive method and can thus not control the error. However, it aims at reducing the error by placing grid-points in a good way.

Authors who have used adaptive time-stepping techniques for option pricing include Clarke and Parrott [9] who use a local error indicator e of the form

$$e = \frac{\left\| \frac{\partial \tilde{F}}{\partial t} - \frac{\mathbf{F}^n - \mathbf{F}^{n+1}}{k} \right\|_2}{\|\mathbf{F}^n\|_2},$$

where $\frac{\partial \tilde{F}}{\partial t}$ is a fourth order accurate estimate of the time derivative $\frac{\partial F}{\partial t}$ and $\mathbf{F}^n$ and $\mathbf{F}^{n+1}$ are approximations of option values at consecutive time-steps. The time-step is then adjusted if e is not in a certain error tolerance interval defined by ε. In the conclusions they state that their adaptive time-stepping method improved the efficiency for long-dated

options. Zvan, Vetzal and Forsyth also propose an automatic time-step selector in [58]. Their method selects time-steps as

$$k^{n+1} = 2\varepsilon / \sqrt{\left\| \frac{\partial^2 F^n}{\partial (t^n)^2} \right\|_\infty}.$$

In their conclusions they state that their automatic time-stepping algorithm can be used to avoid the trial-and-error of non-adaptive methods. But, as Parrott and Clarke they say that it reduces the computational times only for options with long times to maturity.

5. Pricing American options

In the following sections we will give a short overview of some of the numerical techniques that have been used over the last few years to price American options in the PDE setting. See Sec. 3 for the theory. These techniques include a non-linear penalty formulation, front-tracking algorithms, operator splitting and discretized LCP formulations. It is also possible to use methods like binomial tree algorithms and Monte Carlo algorithms that have been discussed for European options in Sec. 4. However, here we limit ourselves to methods for the PDE setting.

5.1 The Brennan-Schwartz algorithm

One of the first methods proposed for pricing the American put option was the so called Brennan–Schwartz algorithm from 1977 by Brennan and Schwartz [4]. The authors propose an algorithm that solves the linear complementarity problem by forcing the solution to stay above Φ for all x and t. They discretize the Black-Scholes PDE using finite differences and end up with a tridiagonal finite difference matrix for the space operator. They truncate the computational domain at the highest stock price considered and use the boundary condition $\frac{\partial F}{\partial s} = 0$ there. Then they do an LU-decomposition of the tridiagonal matrix to form upper and lower bidiagonal matrices U and L and solve using backward/forward substitution. Then in each time-step, i.e. for each discrete LCP, they do a projection in the back-substitution step such that the solution is always greater than Φ. The convergence of this method was studied in [28]. Brennan and Schwartz also consider discrete dividends and give a way of including this in their a algorithm.

5.2 Penalty formulation

The idea with a penalty formulation is to add a penalty term to Eq. (3.6) that ensures that the early exercise constraint is fulfilled. At the same time it must not destroy the solution far away from the free boundary, i.e. the penalty term should be small when not needed. Nielsen et al. use a penalty formulation in [38] for pricing the American put option. They define a barrier function $q(s)$ as $q(s) = K - s$ and the penalty term by

$$\frac{\epsilon C}{F + \epsilon - q}, \tag{5.1}$$

where $0 < \epsilon << 1$ is a parameter and C is a positive constant. Here one should note two things, first that the penalty term is small when $F >> q$ and thus the Black–Scholes equation is approximately satisfied there and second that when $F \to q$ the penalty term approaches C which makes sure that the early exercise constraint is not violated. The authors present this method for several underlying assets and combines it with explicit, semi-implicit and fully implicit finite difference methods. They show that the method fulfills some basic properties of American option prices and at the end they propose the semi-implicit method as the method of choice.

A similar approach is used by Zvan et al. in [57]. However, they add a penalty term to the discrete equations while Nielsen et al. add a penalty term to the continuous equations. Zvan et al. view the problem as a non-linear PDE and develop a general method which they claim can be used together with *black-box off-the-shelf* software for nonlinear PDEs. This has the advantage that advanced discretization methods like flux limiters can be incorporated. Other types of constraints, such as time-dependent barriers, also fit into their framework. The authors also give examples for the American put option with stochastic volatility. They discretize space using a combination of a standard Galerkin finite element method and a finite volume method. The nonlinear discretized equations are solved with approximative Newton iterations.

5.3 Front-tracking/Front-fixing methods

An alternative approach to pricing the American put option is to explicitly keep track of the free-boundary at all times during the computations. The front-tracking formulation for this problem was studied by Meyer and Van der Hoek in [37] and by Wu and Kwok in [55]. Several other authors have studied the same problem including Pantazopoulos, Houstis and Kortesis in [40] and Nielsen, Skavhaug and Tveito in [39]. We will here in short describe the ideas used in these papers to price the American put option.

In the front-tracking algorithm presented in [40] the authors use a finite difference method with an expanding computational domain. At the beginning the space domain $[K\ S_{\max}]$ is discretized, where $S_{\max}$ is chosen sufficiently large. At the first time-step the free boundary will be at $s = K$ and then move to the left towards $s = 0$. For this reason the algorithm must expand the domain to incorporate the new location of the free boundary (and thus add one or more grid-points). For an approximate location of the free boundary, the authors derive a nonlinear equation in p, where p is the fraction of a space-step from the last left-most point to the location of the free boundary that must be solved in each time-step. When p is larger than the space-step h a new point is added to the grid. The authors claim that this method works very well and is competitive in one space dimension. They also point out that due to the "difficulty to describe and track the boundary geometry" the method becomes difficult in more than one space dimension.

Nielsen, Skavhaug and Tveito takes a different approach in [39]. Instead of changing the computational domain they define a time-dependent transformation of variables that removes the free boundary and instead leave them with a nonlinear PDE on a fixed domain. Defining the free boundary as $\bar{s}(t)$ and using the change of variables $x = s/\bar{s}(t)$ and $p(x,t) = F(s,t) = F(x\bar{s}(t),t)$ they derive a new nonlinear PDE in $p(x,t)$. They use finite differences to solve the problem and give both an implicit and an explicit formulation for the time-stepping. The implicit solver requires Newton iterations at each time-step.

5.4 Projected Successive Overrelaxation

Discretizing the LCP, Eq. (3.7), with e.g. finite differences and using an implicit time-stepping method (which is in many ways tractable) the solution of a sequence of (discrete) LCPs is required, one for each time-step,

$$\begin{cases} A\mathbf{F}^{n+1} \geq b, \\ \mathbf{F}^{n+1} \geq \Phi, \\ (\mathbf{F}^{n+1} - \Phi)^T (A\mathbf{F}^{n+1} - b) = 0, \\ \mathbf{F}^0 = \Phi \quad \text{(initial condition)}. \end{cases} \tag{5.2}$$

Here b is a combination of old known solution values and the boundary conditions. The precise form of b depends on the time discretization method and how boundary conditions are implemented.

Projected Successive Overrelaxation (PSOR) is a modification of the standard SOR algorithm [16] and is a popular way of solving the discrete Eqs. (5.2). See e.g. [51, 47] for examples of how to use this technique. The projection comes from the use of the max(·) function in each step that causes the solution to be positive. The draw-back with the PSOR algorithm is that in the continuation region where it is optimal to hold on to the option the PSOR reduces to the unaccelerated SOR method for sparse linear systems.

5.5 Operator splitting

The operator splitting technique for solving the LCP problem will here be explained in short. The technique was introduced in a series of papers by Ikonen and Toivanen, see e.g. [25, 26]. Let us assume that we have the LCP problem formulated as in (3.7). The operator splitting is then based on a reformulation of the problem using an auxiliary variable $\lambda = \lambda(x,t)$.

$$\left(\frac{\partial}{\partial t} - \mathscr{L}\right) F(x,t) = \lambda, \tag{5.3}$$

$$(F(x,t) - \Phi(x))\,\lambda = 0, \tag{5.4}$$

$$F(x,t) \geq \Phi(x), \tag{5.5}$$

$$\lambda \geq 0. \tag{5.6}$$

Ikonen and Toivanen propose and analyze an operator splitting for the time-stepping schemes Euler Backward, BDF2 and a Runge-Kutta method, all with constant time-steps. Here we will show the BDF2 scheme, and allow for variable time-steps.

The time-stepping scheme BDF2 with variable time-steps for a discrete vector **F** is given in Eq. (4.7). The proposed operator splitting method for this scheme with variable step-lengths is

$$\alpha_0^{n+1}\tilde{\mathbf{F}}^{n+1} = k_{n+1}A\tilde{\mathbf{F}}^{n+1} - \alpha_1^{n+1}\mathbf{F}^n - \alpha_2^{n+1}\mathbf{F}^{n-1} + k_{n+1}\lambda^n \tag{5.7}$$

$$\mathbf{F}^{n+1} - \tilde{\mathbf{F}}^{n+1} - \frac{k_{n+1}}{\alpha_0^{n+1}}(\lambda^{n+1} - \lambda^n) = 0, \tag{5.8}$$

$$(\lambda^{n+1})^T(\mathbf{F}^{n+1} - \Phi) = 0,\ \mathbf{F}^{n+1} \geq \Phi,\ \lambda^{n+1} \geq 0. \tag{5.9}$$

First one intermediate step (5.7) is calculated, then the equations (5.8) and (5.9) are used to make sure that the solution at time t_{n+1} fulfills the constraints. This is a fast operation since it can be done point-wise.

In [25] Ikonen and Toivanen show that for the constant volatility case the operator splitting method with simple projections for the LCP is faster than the PSOR algorithm (see Sec. 5.4). The convergence of PSOR also relies on a method parameter ω which need to be tuned for optimal performance, which is not the case with operator splitting making it easier to use. In [26] Ikonen and Toivanen implement their splitting technique also for American options with stochastic volatility, in this case Heston's model [21].

In the more recent paper [27] Ikonen and Toivanen introduce another kind of operator splitting technique called 'componentwise operator splitting' for the American option under stochastic volatility. The method is based on the LCP formulation and by choosing a special space discretization the problem decomposes into several LCPs of tridiagonal structure which can be solved very fast using the Brennan-Schwartz [4] algorithm. The accuracy of the method is increased by the use of Strang symmetrization [49].

6. Our contributions

Here is an indication of what my contribution to each paper is.

Paper I I did all of the implementation and testing and much of the report writing. The space adaptive method was developed with help from my advisors. The time adaptive method is a version of what is used by Söderberg et al. in [34].

Paper II I did all of the implementation and testing and some of the report writing.

Paper III This started as a Master thesis project by Linde. I was coadvisor and Linde used my code and the ideas developed in paper I. Adaptive time-stepping similar to what is used in paper I was added by me and I also performed all experiments with adaptive time-stepping. I did much of the report writing.

Paper IV Started as a Master thesis project by G. Marcusson. I was coadvisor and Marcusson used some of my code when developing the RBF method. I compared the results from the RBF method with results from a version of the method used in paper I.

Paper V I developed the numerical finite difference method for this problem and helped improve efficiency of the implementation of the MC method. Most experiments were performed together by the authors. Report writing was divided between the authors.

Paper VI I did all of the work; theory, implementation, testing and report writing.

The notation in Sec. 6.1 to 6.6 follows, in almost all cases, the notation in each paper respectively. This means that e.g. the value of an option might be denoted by F in one section and by P in the next.

6.1 Paper I

Title: Pricing European Multi-asset Options Using a Space-time Adaptive FD-method

Paper I concerns the numerical pricing of European stock options on several underlying assets. However, the method and technique used is more general and can be used to price also other financial derivatives. We make an attempt at addressing the "curse of dimensionality", a problem that has concerned many researchers not only in the financial area but also in e.g. biology and quantum chemistry. The problem is that many fast and accurate methods for one-dimensional problems are very expensive to use when the number of dimensions increases. Option pricing in the Black&Scholes model faces such problems since each underlying asset adds a dimension in the PDE. For d underlying assets the Black&Scholes equation transformed to forward time is

$$\frac{\partial F}{\partial t} = \sum_{i=1}^{d} r s_i \frac{\partial F}{\partial s_i} + \frac{1}{2} \sum_{i,j=1}^{d} \left[\sigma\sigma^*\right]_{ij} s_i s_j \frac{\partial^2 F}{\partial s_i \partial s_j} - rF := \mathscr{L}F, \qquad (6.1)$$

$$F(s,0) = \Phi(s).$$

where Φ is the contract function, σ is the volatility matrix for the d assets, r is the interest rate, s_i is the value of stock i and F is the price of the option.

The main idea in Paper I is to solve Eq. (6.1) and control the local discretization errors so that they are smaller than some constant ε. This is achieved by first solving the problem very quickly on a coarse grid in both space and time. Then we estimate the local discretization errors on these grids and create new grids with the desired properties. Finally, we solve the problem again on these new grids. This means that we will solve the problem twice but in the second solve we know how large the discretization errors are in the final solution. Moreover, the first solution is obtained very quickly due to the coarse grids used. The local discretization error is estimated using a technique similar to Richardson extrapolation. For one space-dimension we can reason like this: Let A_h be a discrete approximation of $\mathscr{L}$ using step-length h and Au be the exact operator $\mathscr{L}u$ evaluated in the grid-points. Then for any

smooth solution $u(s)$ it holds that

$$A_h u_h = Au + \tau_h,$$

where u_h is a vector of unknowns. We also assume that τ_h can be approximated by the leading term

$$\tau_h = h^p \eta(s) + \mathcal{O}(h^{p+1}),$$

where our second order method has $p = 2$. Now we define δ_h and δ_{2h} as

$$\delta_h = A_h u_h = Au + \tau_h, \tag{6.2}$$

$$\delta_{2h} = A_{2h} u_{2h} = Au + \tau_{2h}. \tag{6.3}$$

The local discretization error on the fine grid with step-lengths h can then be estimated by

$$\tau_h = \frac{1}{2^p - 1}(\delta_{2h} - \delta_h).$$

Using a coarse mesh with step-lengths $\bar{h}$ we then get an estimate $\tau_{\bar{h}}$ that can be used to calculate a discrete approximation of $\eta(s)$ in the grid-points. In each grid-point we then compute a value for the new step-lengths as

$$h(x_j) = \bar{h}(x_j) \left(\frac{\epsilon_h}{|\tau_{\bar{h}}(x_j)|} \right)^{1/p},$$

where x_j are grid-points in the coarse grid. Using linear interpolation we can then find the values of the new step-lengths for the adaptive grid everywhere. Solving again we obtain a solution with a local discretization error smaller than some constant $\epsilon_h > 0$. The idea works similarly in several dimensions. The error in time is estimated by comparing the solutions from an implicit and an explicit time-stepping scheme. Knowledge about the discretization error of each method allows us to estimate the leading term of the discretization error in the implicit BDF2 scheme. During Step 1 in the algorithm we estimate this discretization error on the coarse time grid and store the result as $\tau_{\bar{k}}$, where $\bar{k}$ are the time-step used. Similarly as in space we can then calculate the new grid in time as

$$k(t^n) = \bar{k}(t^n) \left(\frac{\epsilon_t}{|\tau_{\bar{k}}(t^n)|} \right)^{1/2},$$

where t^n is the time-level where we estimate the discretization error $\tau_{\bar{k}}(t^n)$. For global time-stepping we calculate $\tau_{\bar{k}}(t^n)$ as the maximum value over all grid-points in space. Again we can then use linear interpolation to find the sequence of time-steps needed to achieve ε_t local discretization error. The discretization in space and time leads to large sparse systems of equations to solve in each time-step. We have used the iterative method restarted `GMRES`, see [16] for the algorithm, and an incomplete LU factorization as preconditioner. The system matrix is factorized only once in the beginning of the time-stepping, even though the time-steps are changing. This saves computational work and is faster than recomputing the factorization for each time-step.

In this paper we show that we can control the local discretization errors so that the user can choose the level of accuracy by a parameter ε rather than adjusting space and time-steps manually. The method can also be used to limit the number of space-steps in each dimension and allow the program to choose the discretization so that the local discretization error is as low as possible using a fixed number of points. This can be used if controlling the memory usage is more important than accuracy.

Our experiments show that using our method we do not break the "curse of dimensionality" but we reduce the number of grid-points needed to achieve a certain accuracy by a factor 2^{-d} compared to a standard second order finite difference method. Options on up to four underlying assets have been considered with this adaptive method.

6.2 Paper II

Title: Space-time Adaptive Finite Difference Method for European Multi-asset Options

In this paper we aim at controlling the global error in the solution when pricing the multi-asset European call option. By *a priori - a posteriori* error estimates we estimate a functional of the error $E(0,s)$

multiplied by a function $g(s)$ so that the value is smaller than some constant $\epsilon > 0$,

$$|\int_D g(s)E(0,s)\,ds| \le \epsilon. \tag{6.4}$$

The model problem is the same as in paper I. Transforming the problem to a PDE in forward time for the price P and the underlying stock x_i we get

$$\begin{aligned} P_t - r\sum_{i=1}^{d} x_i P_i - \sum_{i,j=1}^{d} a_{ij}x_i x_j P_{ij} + rP &= 0, \\ P(0,x) &= \Psi(x). \end{aligned} \tag{6.5}$$

Here the coefficients for the mixed and second derivatives are denoted by a_{ij} and $\Psi(x)$ is the contract function for the option.

The main idea is to apply a technique used in error control for finite element methods, see e.g. [12], where one estimates the global error using the discretization error and the adjoint solution. If we denote the solution error by $E = \tilde{P} - P$ where $\tilde{P}$ is a smooth reconstruction of the numerical solution P_h^n and P is the solution to the regular Black–Scholes equation then E approximately solves the PDE

$$\begin{aligned} E_t - r\sum_{i=1}^{d} x_i E_i - \sum_{i,j=1}^{d} a_{ij}x_i x_j E_{ij} + rE = E_t - \mathscr{L}E = \tau, \\ E(0,x) = 0, x \in D, \qquad E(t,x) = 0, x \in \partial D, \end{aligned} \tag{6.6}$$

where the right-hand-side τ is the local discretization error. Note here that the local discretization error is $\tau = \tau_h + \tau_k$, i.e. the sum of the space- and time-discretization errors. The goal is to use an *a posteriori* estimate of τ at t^n and then estimate how much the discretization error can grow for $t > t^n$. For this we need the adjoint solution u to P,

$$u_t + \mathscr{L}^* u = 0, \tag{6.7}$$

$$\begin{aligned} \mathscr{L}^* u = -r\sum_{i=1}^{d} (x_i u)_i + \sum_{i,j=1}^{d} a_{ij}(x_i x_j u)_{ij} - ru, \\ u(T,x) = g(x). \end{aligned} \tag{6.8}$$

Note that the boundary condition for the adjoint problem is that $u = 0$ on ∂D and that it is a final value problem while our transformed

Black&Scholes PDE is an initial value problem.

If we use the equation for the error (6.6) and the equation for the adjoint solution we can obtain the following relation

$$\int_0^T \int_D u\tau \, dx \, dt = \int_D g(x)E(T,x)dx. \tag{6.9}$$

Instead of solving the error equation and compute the integral (6.4) we will divide the space-time domain $D \times [0\ T]$ into L boxes and estimate the local space discretization error at the beginning of each box. The growth of this *a posteriori* estimate is then estimated within each box and a new space discretization is created to control the spatial error. The total error ϵ is "distributed" to each box using the adjoint solution according to

$$\epsilon_\ell = \epsilon / (L||u||_\ell),$$

where $||u||_\ell$ is the maximum of the adjoint solution in box ℓ for $\ell = 1, \ldots, L$. The adjoint problem is solved on a very coarse grid in both space and time. The local discretization error in space is estimated *a priori* by applying the maximum principle to PDEs where the variable is higher order derivatives of the solution. These terms typically look like $-\frac{1}{6}h_k^2 P_{kkk}$ and $-\frac{1}{12}h_k^2 P_{kkkk}$ for the first and second derivatives respectively. Here we have made the assumption that the dominant error terms come from the first and second derivatives in (6.5) and not from the mixed derivatives. The adaptive method will use different adaptive grids in space in each box. Between the boxes the solution is interpolated using cubic interpolation. The time-steps are adjusted step by step using a technique very similar to what was used in [34].

	Estimate of $\int_D gEdx$	$\int_D^* gEdx$	Grid
Ad. grid	0.001088	-0.000083	[81 61 49 45 41 37 33 53]
Equi. grid	0.002229	-0.000178	[81]
Equi. grid	0.001086	-0.000075	[121]

Table 6.1: *The tolerance was set to* 10^{-3} *for the adaptive grid. Estimates of the functional is presented for equidistant grid and adaptive grid.*

The results of our experiments with the method show that it produces estimates of the error at the same level as requested. In Table

6.1 we see for a one-dimensional example that the estimate of the functional is of the correct level and that the number of grid-points in space varies over the boxes. In this case we used 8 boxes and the number of grid-points goes down over time except for the last box. The reason for this is that the last box was larger and thus required more grid-points. We also see that using 81 adaptively placed points, which was the largest number used, but equidistantly distributed, yields an estimate of the integral two times as large as ε. To get the value of the integral down to ε using an equidistant distribution we need as many as 121 points. Computing the true error and a numerical value of the integral we see that we over-estimate the value with about a factor 10, these are the $\int_D^* gEdx$ values in the table. Similar experiments where performed also with two and three underlying assets with similar results. In this experiment we used $g(x) = c\exp(-5(x-1)^2)$ where c is chosen to get $\int_D g(x)dx = 1$. In several dimensions we used a product of such Gaussian functions.

In this paper we also compare a standard second order finite difference method with some different Monte Carlo methods regarding efficiency. The experiments with one underlying asset show that the finite different method is more efficient for a few underlying assets when a small errors is required. This is in agreement with the theory.

6.3 Paper III

Title: A Highly Accurate Adaptive Finite Difference Solver for the Black-Scholes Equation

The work in this paper originates from the Master thesis by Linde [33]. We develop a technique to achieve a $\mathcal{O}(h^6)$ space-discretization for the multi-dimensional Black&Scholes equation (6.1) presented in Sec. 6.1. The new method is combined with a second order accurate time-discretization similar to what was used in paper I. To improve the method even further it has been combined with adaptivity in both space and time using the techniques in paper I.

We consider the Black&Scholes equation transformed to

backward time. Applying higher-order space discretizations to this equation does not give higher accuracy due to the discontinuous first derivative of the initial condition. As a remedy to this problem we have used a grid with smaller steps $\bar{h}$ in a region around the discontinuity. From experiments we know that close to the strike price the error is of order $\mathcal{O}(\bar{h}^2)$ even when sixth order approximations of the derivatives are used. This leads us to the idea of refining the grid around the discontinuity such that

$$\bar{h}^2 = C^2 h^6,$$

where C depends on higher order derivatives of the solution. This means that refining the grid according to

$$R = \frac{h}{\bar{h}} = \frac{1}{Ch^2}$$

near the discontinuity we can get the overall error to be $\mathcal{O}(h^6)$. Here R denotes the refinement of the grid.

The sixth-order approximation requires a seven points wide stencil and experiments show that the fine grid only has to cover an interval of four points on each side of K. This ensures that the finite difference stencil used outside the fine area does not reach over the discontinuity. Grid-points outside the fine area are used as boundary conditions and extra values are interpolated from that grid if not present in the coarse grid. Experiments have also shown that the fine grid in space only have to be used for a very short period to keep the higher accuracy of the method. Let us denote the first time when we no longer use the fine grid by T'. Then for each time-step with $t < T'$ do

1. Compute the solution on the coarse grid.
2. Compute the solution on the fine grid.
3. Update the values on the coarse grid that have a corresponding point on the fine grid.

Go back to 1, unless $t = T'$. Then

4. Compute the solution on the coarse grid only in the interval $t \in (T', T]$.

In several space dimensions the fine grid would ideally be used only just around the discontinuous first derivative of the initial

condition. Such an implementation though, would be much more complicated than our current approach with using a hypercube that encompasses the area where a fine grid is needed. Our way requires more grid-points but is much simpler to handle and easier to implement.

For the time-discretization we use an adaptive version of BDF2 as in Paper I. For the space adaptivity to work properly we must have a certain number of time-steps during the short time with two grids, i.e. during $t \in [0\ T']$. This can in most cases be achieved by using a logarithmic distribution of the time-steps during the first solve, that will give us a good estimate of the first short important period.

Our results show that with this method a discretization with over all error $\mathcal{O}(h^6)$ can be achieved. We compare the accuracy of our highly accurate scheme with and without adaptivity to a second order and a sixth order method. Our experiments show that the developed $\mathcal{O}(h^6)$ method is very accurate. The results also show that the standard sixth order method without the small grid only has an effective rate of convergence of 2. Moreover, the efficiency has been studied in comparison to the adaptive second order methods used in [42]. The two-dimensional experiments show that with our new method much fewer grid-points can be used to achieve the same error as with the second order method. The second order method is still competitive with respect to computational time for some cases even though it uses many more grid-points, probably due to the simpler structure of the finite difference matrix involved.

6.4 Paper IV

Title: Improved Radial Basis Function Methods for Multi-dimensional Option Pricing

In this paper we again address the problem of pricing the European basket option efficiently for several underlying assets, see Sec. 6.1 for the problem formulation and the PDE. However, here we use RBFs as global approximants instead of finite differences. With

RBFs we approximate the solution using continuous smooth functions whose derivatives can be found analytically. In this paper we derive the RBF method for European multi-asset options and compare it with the finite difference method used in Paper I. We also propose what boundary conditions to use, on what boundaries they are needed and how they can be implemented. In this section the value of the underlying asset will be denoted by x and we have scaled the space variables such that the strike-price K is at $\bar{K} = 1$.

Since RBFs are mesh-free and potentially of spectral order (when using infinitely smooth RBFs and the solution is regular enough) they can be an efficient alternative to standard methods. The RBF approximant we use has the form

$$u(x) = \sum_{j=1}^{N} \lambda_j \phi(\varepsilon \| x - x_j \|),$$

where $\phi(\varepsilon R)$ (with εR the scaled distance) denotes the RBF, x_j are the center points and ε is the important shape parameter. The effect of the shape parameter on the accuracy is studied in this paper. It has also been previously studied for more general problems in e.g. [35] and in [6] for other problems.

When solving the Black&Scholes PDE a very important, but often overlooked issue is the boundary conditions. In the finite difference method we use a linearity condition at the boundary and. With the RBF method we use

$$u(x) \to \frac{1}{d} \sum_{i=1}^{d} x_i - \bar{K} e^{-2rt}, \, ||x|| \to \infty, \tag{6.10}$$

at the far-field boundary and $u(0) = 0$ at the origin, at all other boundaries we impose no boundary conditions. Janson and Tysk show in [29] that the problem is well-posed even without boundary conditions as long as the growth at infinity is restricted.

The time-dependent solution of the Black&Scholes PDE is approximated using a linear combination of RBFs centered at the

node-points x_k, $k = 1,\ldots,N$.

$$u(t,x) = \sum_{k=1}^{N} \lambda_k(t)\phi(\varepsilon\|x - x_k\|) = \sum_{k=1}^{N} \lambda_k(t)\phi_k(x). \tag{6.11}$$

The $\lambda_k(t)$ are the unknown coefficients to determine. We have used the method of collocation at the node-points. Where appropriate we use the boundary conditions discussed earlier. Let $u_i(t) = (u(t,x_1),\ldots,u(t,x_{N_i}))^T$ be the values in the interior points and $u_b(t) = (u(t,x_{N_{i+1}}),\ldots,u(t,x_N))^T$ be the values in the boundary points we get

$$\mathscr{L}u_i(t) = \begin{pmatrix} C_{ii} & C_{ib} \end{pmatrix} \begin{pmatrix} u_i(t) \\ u_b(t) \end{pmatrix} \tag{6.12}$$

where the values in C_{ii} and C_{ib} can be computed by letting the space operator act on the basis functions and by using Eq. (6.11). The boundary conditions are implemented by setting $u_b^n = g_b^n$. The values of g_b^n are given by 0 at the origin and (6.10) at the far-field boundary $||x||_1 = C > 0$.

We have used the second order accurate two-step method BDF2 [20] for time-stepping. Including the boundary conditions we get

$$\begin{pmatrix} I - k\beta_0 C_{ii} & -k\beta_0 C_{ib} \\ 0 & I \end{pmatrix} \begin{pmatrix} u_i^n \\ u_b^n \end{pmatrix} = \begin{pmatrix} -\beta_1 u_i^{n-1} - \beta_2 u_i^{n-2} \\ g_b^n, \end{pmatrix}. \tag{6.13}$$

where $\beta_0 = \frac{2}{3}, \beta_1 = -\frac{4}{3}$ and $\beta_2 = \frac{1}{3}$. For all numerical experiments we used the multiquadric RBFs defined by $\phi(\varepsilon R) = \sqrt{1+\varepsilon^2 R^2}$. Here the shape parameter is introduced as a scaling of the distance R. We have computed the error in the solution using two different measures. In what we call the *financial norm* we study the absolute error in the interior only

$$E_f = \max_{x\in\Omega_i} |E(x)|, \quad \Omega_i = \{x_i | \frac{1}{d}\sum_{i=1}^{d} x_i \in [\bar{K}/3 \quad 5\bar{K}/3]\}.$$

The other error norm used is a weighted integral norm defined by

$$E_w = \int_\Omega w(x)|E(x)|d\Omega,$$

where Ω is the whole computational domain and $w(x)$ is a weight function chosen as a product of d Gaussian functions with $\int_\Omega w(x)dx = 1$.

Our experiments show that placing the node-points in a non-uniform way can improve the accuracy by one order of magnitude. In one space-dimension we place more node-points close to the discontinuity of the first derivative (i.e. close to the strike-price) and less node-points in the far-field region according to a certain strategy. In two dimensions we use the same strategy in the diagonal direction from the origin and then out towards the sides.

Choosing an optimal shape parameter ε is an important issue when using RBFs. Unfortunately it is problem dependent, see [31], and it is not easy to *a priori* determine a good choice. One problem is that the RBF matrices become more and more ill-conditioned when ε decreases. In this study we have found that choosing $\varepsilon = 1 + N/20$, where N is the number of node-points, works very well in one dimension for small N. However, changing the shape parameter with the number of node-points gives only algebraic convergence. Spectral accuracy with increasing N is achieved only with fixed ε.

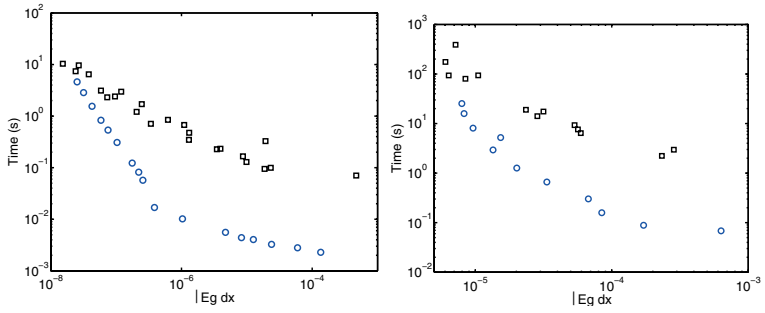

Figure 6.1: Comparison with adaptive FD method, 1D (LEFT) and 2D (RIGHT). RBF results denoted by a circle ('$\circ$') and FD with squares ('$\square$').

Finally we compare the RBF method with the finite difference method that was used in Paper I. The RBF method performs very

well both in one and two space-dimensions. In Figure 6.1 we see the computational time needed to achieve a certain financial error. The RBF method is up to 40 times faster than the FD method, it is especially more competitive for lower accuracies. The strength of the adaptive FD method though is that we can control the local discretization errors. A tolerance level can be given by the user and discretizations to meet these demands are created automatically. There is no need to guess the number of space and time-steps needed for a certain tolerance.

6.5 Paper V

Title: Pricing Turbo Warrants

Paper V is different from the other papers in the sense that it does not aim at developing new numerical methods, but rather to solve a new option pricing problem using existing methods. In this paper we study the fairly new contract issued by the French company Société Generale in early spring 2005. We price numerically the *Turbo warrant* call (but we have methods also for the put) and compare a PDE based finite difference method with a Monte Carlo method. The Turbo warrant is a barrier knock-out option with the barrier *in-the-money* that pays a small rebate to the holder if the barrier is hit. When the original contract ends at the barrier a new one starts. The new contract is essentially also a call option but on the minimum of the stock-process. The new contract only lasts for 3 hours and the amount paid to the holder is the difference between the lowest recorded stock price and the strike-price during this 3 hour period. The SG Turbo warrant call considered here can be summarized as:

A turbo warrant call pays $(S_T - K)^+$ *at maturity* T *if a prescribed barrier* $b \geq K$ *has not been hit by* S_t *at any time* $t < T$. *If* S_t *hits the barrier the contract is immediately terminated and a new contract starts. The new contract is a call option on the minimum process of* S_t, *with the same strike-price, maturing* δ *time-units later.*

In the pricing we have assumed that the underlying stock S_t

at time t follows the process

$$dS_t = rS_t dt + S_t \sigma(S_t, t) dW_t.$$

The contract is priced as a sum of two contracts, one knock-out option, denoted by $F(s,t)$, maturing at time T and one knock-in option, denoted by $G(s,t)$, maturing δ after knock-in. The knock-out part is given by

$$F(s,t) = \mathbf{E}_{s,t}\left[e^{-r(T-t)}(S_T - K)^+ \mathbf{1}_{\{\tau_b > T\}}\right]. \tag{6.14}$$

Here $\tau_b = \inf\{t \geq 0 : S_t \leq b\}$ denotes the first hitting time. The knock-in part is given by

$$G(s,t) = \mathbf{E}_{s,t}\left[e^{-r(\tau_b+\delta-t)}(m_{\tau_b+\delta} - K)^+ \mathbf{1}_{\{\tau_b \leq T\}}\right], \tag{6.15}$$

where $m_t = \min_{0\leq u\leq t} S_u$ is the running minimum of S_t.

Both F and G satisfy the Black&Scholes equation

$$\mathcal{M}\psi = \frac{1}{2}s^2\sigma^2(s,t)\frac{\partial^2\psi}{\partial s^2} + rs\frac{\partial\psi}{\partial s} - r\psi + \frac{\partial\psi}{\partial t}. \tag{6.16}$$

For the first problem we solve for $\psi = F(s,t)$ to find the value of the barrier knock-out part. This is a standard problem and can be solved as a European contract with the value zero at the barrier. To price the knock-in part we divide the problem into two parts. The knock-in contract has a value $G(s,t)$ if it is initiated, i.e. *after* the stock hits the barrier, but it also has a value $H(s,m,t)$ *before* the stock hits the barrier since there is a chance it will become valuable. Solving Eq. (6.16) numerically for $F(s,t), G(s,t)$ and $H(s,m,t)$ involves the discretization of Black&Scholes equation for different domains and with different boundary conditions. The formulation of the problem for finding F is a standard contract and will not be stated here, the formulations needed to find $G(s,t)$ and $H(s,m,t)$ are given by

$$\begin{cases} \mathcal{M}G(s,t) = 0 \text{ on } (b,\infty)\times[0,T), \\ G(b,t) = g(b,t) \text{ for } t \in [0,T], \\ G(s,T) = 0 \text{ for } s > b, \end{cases} \tag{6.17}$$

and

$$\begin{cases} \mathscr{M} H^u(s,m,t) = 0 \text{ on } \Omega_u \\ H^u(s,K,t) = 0 \text{ for } t \in [u, u+\delta), \\ H^u(s,m,u+\delta) = (m-K)^+ \text{ for } s \geq m, K \leq m \leq b, \\ \frac{\partial H^u}{\partial m}(m,m,t) = 0 \text{ for } t \in [u, u+\delta), K \leq m < b. \end{cases} \tag{6.18}$$

Note that the $H(s,m,t)$-problem must be solved before the $G(s,t)$-problem since we need the value $H(b,b,t)$ according to Proposition 1 below.

Proposition 1 *The boundary values $g(b,t)$ are given by $g(b,t) = H^t(b,b,t)$ where $H^u(s,m,t)$ solves Equation (6.18).*

The domain is discretized in space and time using central second order finite differences in space and the second order accurate time-stepping algorithm BDF2 [20] in time. The $H(s,m,t)$ problem may look two-dimensional but since the s and m variables only couple at the boundary $s = m$ it is essentially only a set of one-dimensional problems. The computational domain is depicted in Figure 6.2.

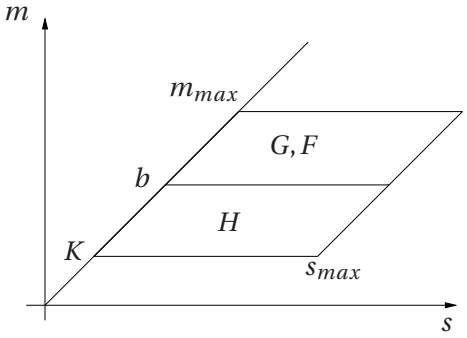

Figure 6.2: The computational domain for the Turbo warrant call.

The boundary condition $\frac{\partial H}{\partial m} = 0$ was implemented by approximating the derivative using a finite difference approximation in the m-direction.

Using the Monte Carlo method for standard contracts it is not always necessary to simulate the whole trajectory of the underlying

stock. However, due to the path-dependent nature of the Turbo warrant one must in general simulate the whole trajectories. The value of the Turbo warrant call is given by $V(s,t) = F(s,t) + G(s,t)$ with F and G defined in Eqs. (6.14) and (6.15). $F(s,t)$ can be approximated using the standard Monte Carlo approach

$$F \approx e^{-r(T-t)} \frac{1}{N} \sum_{S \in \mathcal{Z}} (S_T - K)^+,$$

where $\mathcal{Z}$ are those trajectories that do not hit the barrier. The value of G can be approximated by continuing to simulate the trajectories that do hit the barrier for another δ time-units after they hit. For these trajectories we record the minimum process m_t and approximate the expected value as above but as a call option on m_t as described before.

Our numerical experiments show that the finite difference method is up to 50 times faster to achieve a certain level of accuracy for the error. In this experiment we set an error level using confidence intervals for the Monte Carlo method and then reproduced the same error with the finite difference method. The results are shown in Figure 6.3 (LEFT). For the finite difference method we have compared the numerical solution with the exact solution derived by Eriksson in [11]. In Figure 6.3 (RIGHT) we have studied the exact errors for both finite

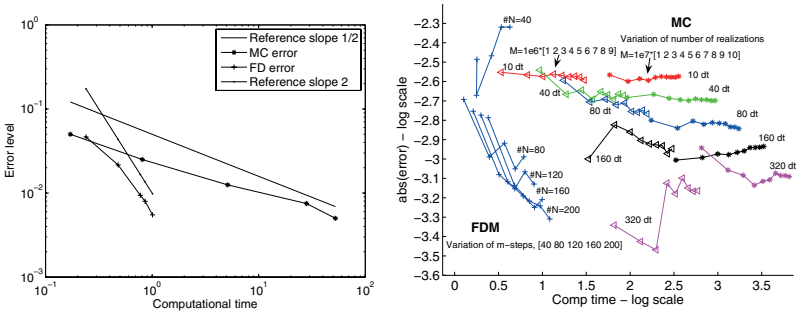

Figure 6.3: LEFT: Errors plotted for the finite difference and the Monte Carlo method when pricing the Turbo warrant call. RIGHT: Absolute error for the MC and the finite difference method. FDM: N is number of grid-points in space, denoted by '+'. MC: M =1 to 9 million realizations denoted by '$\triangleleft$' and M =10 to 50 millions by '$*$'.

difference and Monte Carlo methods and we again see that the finite difference method is much faster and more accurate than the Monte Carlo method.

6.6 Paper VI

Title: Pricing American Options Using a Space-time Adaptive FD-method

Paper VI continues the work from the paper I and II with space- and time-adaptive finite difference methods from European options to American options. The early exercise feature in American options is what causes most of the numerical difficulties compared to pricing European options. One such difficulty is that the solution looses regularity at the internal free boundary.

In this paper we consider the American put option on one underlying stock both with constant and stochastic volatility in the setting of linear complementarity problems (LCP). The PDE inequalities that must be solved to price these contracts has for constant volatility the same operator as in Eq. (3.7) and the equations for stochastic volatility are described below.

In the case of stochastic volatility we have used Heston's model [21] where the stochastic differential equations

$$dx_t = \mu x_t dt + \sqrt{y_t} x_t dW_1, \tag{6.19}$$

$$dy_t = \alpha(\beta - y_t)dt + \gamma\sqrt{y_t} dW_2, \tag{6.20}$$

are used to model the stock and the variance. Here x_t is the stochastic stock process, y_t is a variance process and μ is the positive drift of the stock. The parameter γ is the volatility of the variance process and α, $\beta > 0$ are parameters for the mean-reversion of the variance process. The Brownian motions W_1 and W_2 are correlated with a factor $\rho \in [-1, 1]$. Using these two stochastic differential equations a generalized PDE

$$\frac{\partial F}{\partial t} = rx\frac{\partial F}{\partial x} + \frac{1}{2}yx^2\frac{\partial^2 F}{\partial x^2} + \rho\gamma xy\frac{\partial^2 F}{\partial x \partial y} + \{\alpha(\beta - y) - \theta\gamma\sqrt{y}\}\frac{\partial F}{\partial y} - rF := LF \tag{6.21}$$

can be derived where F is the price of the American option. Here r is the interest rate and θ is the so called market price of risk which we will assume is zero. The PDE is here stated in forward time. The price will still solve the system of PDE inequalities defined in Eq. 3.7 but with the differential operator from Eq. (6.21).

The discretization of the PDEs and the adaptivity in space is similar to what was used in paper I-III but modifications have been made to the method to take into account the internal free boundary. Here we have focused on the time adaptivity since we numerically find that when we use a second order accurate method we only get first order convergence in time near the free boundary. The first method tested is similar to what was used in paper II and will be denoted x1a, the second is denoted by x1b and is like the first but is modified close to the free boundary. The modification is a removal of the estimate of the local discretization error close to the free boundary. The third method uses a local time-stepping procedure close to the free boundary. This method will be denoted x2 where x later on will be replaced by C for constant or by S for stochastic volatility. For both cases the method of operator splitting [25], introduced by Ikonen and Toivanen, has been used to solve the LCPs.

In Fig. 6.4 we show where local time-steps are used for the methods C2 and S2. The local time-stepping uses Euler backward and

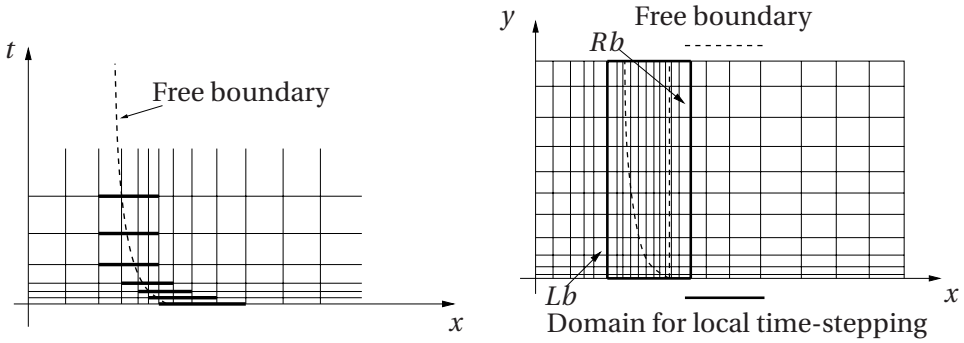

Figure 6.4: LEFT: Bold lines indicate where local time-steps are used in C2. RIGHT: The area where local time-steps are used for all time-steps in S2 includes the free boundary at all times.

Euler forward to estimate the local discretization error (in the same way as with BDF2, see paper I Sec. 6.1) and then chooses the number of steps needed to reach the pre-described tolerance ε. In S2 the number of maximum local steps have been restricted.

Using known results about the behavior of the solution we study mathematically the continuity of the option price near the free boundary. The "smooth fit" condition states that the first derivative in space is continuous over the free boundary and some simple calculations show that the same thing holds for the first time derivative. Some further calculations and the use of the PDE show that the second derivatives in space and time are not continuous over the free boundary.

Summarizing the results from the numerical experiments we find that in the case of constant volatility all the methods produce errors of about the same level for a given ε but C1b and C2 are faster than C1a. Both C1b and C2 chooses time-steps in a smooth way but with C1b the local discretization errors are not of the required size ε as can be expected. Fig 6.5 show the measured errors with these three methods for some ε. For stochastic volatility both S1b and S2 produce

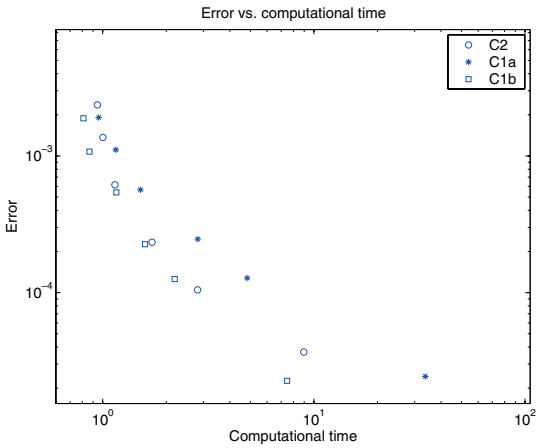

Figure 6.5: Error with C1a, C1b and C2 plotted vs. computational time using the same space discretization.

smoothly varying time-steps but the estimated local discretization

errors in time using S1b are not of the required size. In this case S1a and S1b produces solutions with errors that are of about the same size while S2 produce solutions with larger errors. This is probably due to the restriction in the number of allowed time-steps in the local time-stepping algorithm.

7. Concluding remarks

This thesis concerns numerical option pricing using finite differences. Adaptive finite difference methods have been developed that can be used to solve option pricing problems with control of either the local discretization error or the error in the solution. The methods have been tested and proven to work well for European options on one and several underlying assets. The implementation is straightforward and requires no special data structure. In general a lot fewer grid-points are needed to achieve the same error with an adaptive method as with a non-adaptive method. Being able to predefine an error tolerance is perhaps the biggest advantage. For American options both constant and stochastic volatility with and without local time-stepping have been considered. The numerical algorithms developed keep the local discretization errors at a predescribed level. No "trial and error" is necessary for finding suitable space- and time-steps.

Also radial basis functions as approximants have been considered for option pricing. An introductory study shows that it is a potentially very good method to solve multi-dimensional option pricing problems. Since it is mesh-free no specific data structures are needed and its spectral accuracy can keep the number of unknowns to a low level. The drawback is that the linear systems of equations that must be solved when using an implicit time-stepping algorithm are dense. This makes it memory demanding when the number linear equations is large.

A new contract, the Turbo warrant, has been solved both with a finite difference method and with a Monte Carlo algorithm. Numerical experiments show that the finite difference method can be very competitive and in the experimental setup studied outperformed the Monte Carlo method by a factor 50.

8. Acknowledgment

This thesis was funded by FMB, the Graduate School in Mathematics and Computing. Thank you for giving me the opportunity to put this thesis together and for allowing me to travel to very interesting conferences in beautiful locations. The funds for traveling have really been appreciated. I also wish to thank all the nice people at TDB and Syscon for all the nice coffee breaks I have had, always at 10.00 in the morning and at 15.00 in the afternoon. *Thank you.*

I also would like to send some thanks to my friends at the Math department. You helped me through many of the courses and answered many of my mathematical questions. *Thank you.*

However, there would never have been a thesis without my advisors, Lina von Sydow, Per Lötstedt and Johan Tysk who helped me through it all. *Thank you.*

Last but not least my thanks goes to Åsa who supported me the whole time. *Thank you.*

This thesis was funded by FMB

Bibliography

[1] Y. Achdou and O. Pironneau. *Computational Methods for Option Pricing*. SIAM, Philadelphia, 2005.

[2] T. Björk. *Arbitrage Theory in Continuous Time*. Oxford University Press, New-York, 1998.

[3] F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81:637–659, 1973.

[4] M. Brennan and E. Schwartz. Finite difference methods and jump processes arising in the pricing of contingent claims: A synthesis. *Journal of Financial and Quantitative Analysis*, 13:461–474, 1977.

[5] G. W. Buetow and J. S. Sochaki. The trade-offs between alternative finite difference techniques used to price derivative securities. *Applied Mathematics and Computation*, 115:177–190, 2000.

[6] Martin Buhmann and Nira Dyn. Spectral convergence of multiquadric interpolation. *Proceedings of the Edinburgh Mathematical Society. Series II*, 36(2):319–333, 1993.

[7] H. J. Bungartz and M. Griebel. Sparse grids. Acta Numerica, 13:147–269, 2004.

[8] R. E. Caflisch. Monte Carlo methods and quasi-Monte Carlo methods. *Acta Numerica*, pages 1–49, 1998.

[9] N. Clarke and K. Parrott. Multigrid for American option pricing with stochastic volatility. *Applied Mathematical Finance*, 6:177–195, 1999.

[10] J. C. Cox, S. A. Ross, and M. Rubinstein. Option pricing: A simplified approach. *Journal of Financial Economics*, 7:229–263, 1979.

[11] J. Eriksson. Explicit pricing formulas for turbo warrants. Accepted in *RISK* magazine.

[12] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. Introduction to adaptive methods for dierential equations. *Acta Numerica*, 4:105–158, 1995.

[13] Gregory E. Fasshauer, Abdul Q. M. Khaliq, and David A. Voss. Using meshfree approximation for multi-asset American option problems. *Journal of Chinese Institute Engineers*, 27:563–571, 2004.

[14] A. V. Gerbessiotis. Trinomial-tree based parallel option price valuations. *Parallel Algorithms and Applications*, 18(4):181–196, 2003.

[15] P. Glasserman. *Monte Carlo Methods in Financial Engineering*. Springer Verlag, New York, 2004.

[16] G.H. Golub and C.F. Van Loan. *Matrix Computations*. The John Hopkins University Press, 3rd edition, Baltimore, 1996.

[17] M. Griebel. The combination technique for the sparse grid solution of pdes on multiprocessor machines. *Parallel Processing Letters*, 2:61–70, 1992.

[18] M. Griebel, M. Schneider, and C. Zenger. A combination technique for the solution of sparse grid problems. *Iterative Methods in Linear Algebra, IMACS*, pages 263–281, 1992.

[19] B. Gustafsson, H.-O. Kreiss, and J. Oliger. *Time Dependent Problems and Difference Methods*. Wiley, Pure and Applied Mathematics, 1995.

[20] Hairer, E., Nørsett, S. P., and G. Wanner. *Solving ordinary differential equations. I - Nonstiff problems*. Springer-Verlag, Berlin, second edition, 1993.

[21] S. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies*, 6:327–343, 1993.

[22] S. Heston and G. Zhou. On the rate of convergence of discrete-time contingent claims. *Mathematical Finance*, 10(1):53–75, 2000.

[23] Y. C. Hon. A quasi-radial basis functions method for American options pricing. *Computers and Mathematics with Applications*, 43(3-5):513–524, 2002.

[24] Yiu-Chung Hon and Xian-Zhong Mao. A radial basis function method for solving options pricing model. *Journal of Financial Engineering*, 8:1–24, 1999.

[25] S. Ikonen and J. Toivanen. Operator splitting methods for pricing American options. *Applied Mathematical Letters*, 17:809–814, 2004.

[26] S. Ikonen and J. Toivanen. Operator splitting methods for pricing American options with stochastic volatility. report, Department of Mathematical Information Technology, 2004. University of Jyväskylä, Finland.

[27] S. Ikonen and J. Toivanen. Componentwise splitting methods for pricing American options under stochastic volatility. report, Department of Mathematical Information Technology, 2005. University of Jyväskylä, Finland.

[28] P. Jaillet, D. Lamberton, and B. Lapeyre. Variational inequalities and the pricing of American options. *Acta Applicande Mathematicae*, 21:263–289, 1990.

[29] S. Janson and J. Tysk. Feynman–Kac formulas for Black–Scholes type operators. *Bulletin of the London Mathematical Society*, 38:269–282, 2006.

[30] R. Kangro and R. Nicolaides. Far field boundary conditions for Black–Scholes equations. *SIAM Journal of Numerical Analysis*, 38(4):1357–1368, 2000.

[31] E. Larsson and B. Fornberg. Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions. *Computers & Mathematics and Applications*, 5:103–130, 2005.

[32] B. Lastdrager, B. Koren, and J. Verwer. The sparse-grid combination technique applied to time-dependent advection problems. *Applied Numerical Mathematics*, 38:377–401, 2001.

[33] G. Linde. High-order adaptive space-discretizations for the Black-Scholes equation. Master Thesis at Uppsala University Department of Information Technology., 2005.

[34] P. Lötstedt, S. Söderberg, A. Ramage, and L. Hemmingsson-Frändén. Implicit solution of hyperbolic equations with space-time adaptivity. *BIT*, 42:128–153, 2002.

[35] W. R. Madych. Miscellaneous error bounds for multiquadric and related interpolators. *Computers & Mathematics with Applications*, 24(12):121–138, 1992.

[36] B. J. McCartin and S. M. Labadie. Accurate and efficient pricing of vanilla stock options via the crandall-douglas scheme. *Applied Mathematics and Computation*, 143:39–60, 2003.

[37] G.H. Meyer and J. van der Hoek. The valuation of American options with the method of lines. Technical report, Georgia Institute of Technology, 1995.

[38] B. F. Nielsen, O. Skavhaug, and A. Tveito. Penalty methods for the numerical solution of American multi-asset option problems. Preprint 2000-289, Department of Informatics, University of Oslo, 2000.

[39] B. F. Nielsen, O. Skavhaug, and A. Tveito. Penalty and front-fixing methods for the numerical solution of american option problems. *Journal of Computational Finance*, 5(4):69–97, 2002.

[40] K.N. Pantazopoulos, E.N. Houstis, and S. Kortesis. Front-tracking finite difference methods for the valuation of American options. *Computational Economics*, 12:255–273, 1998.

[41] P. Pellizzari. Efficient Monte Carlo pricing of European options using mean value control variates. *Decisions in Economics and Finance*, 24:107–126, 2001.

[42] J. Persson and L. von Sydow. Pricing European multi-asset options using a space-time adaptive FD-method. Technical Report 2003-059, Dept. of Information Technology, Uppsala University, Available at `http://www.it.uu.se/research/reports/`, 2003. Accepted in *Computing and Visualization in Science*.

[43] U. Pettersson, E. Larsson, G. Marcusson, and J. Persson. Improved radial basis function methods for multi-dimensional option pricing. Technical report 2006-028, Dept. of Information Technology, Uppsala University, Uppsala, Sweden, 2006. Submitted.

[44] O. Pironneau and F. Hecht. Mesh adaption for the Black& Scholes equations. *East-West Journal of Numerical Mathematics*, 8:25–35, 2000.

[45] C. Reisinger. *Numerische Methoden für hochdimensionale parabolische Gleichungen am Beispiel von Optionspreisaufgaben*. PhD thesis, Universität Heidelberg, 2004.

[46] L. C. G. Rogers. Monte Carlo valuation of American options. *Mathematical Finance*, 12(3):271–286, 2002.

[47] R. Seydel. *Tools for Computational Finance*. Springer Verlag, Berlin, 2003.

[48] S. A. Smolyak. *Quadrature and interpolation formulas for tensor products of certain classes of functions*. Doklady Akademii Nauk SSSR, 1963.

[49] G. Strang. On the construction and comparison of difference schemes. *SIAM Journal of Numerical Analysis*, 5:506–517, 1968.

[50] J. C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations*. Chapman & Hall, London, 1989.

[51] D. Tavella and C. Randall. *Pricing Financial Instruments - The Finite Difference Method*. John Wiley & Sons, Inc., 2000.

[52] Y. Tian. A flexible binomial option pricing model. *The Journal of Futures Market*, 19(7):817–843, 1999.

[53] P. Wilmott, J. Dewynne, and S. Howison. *Option Pricing: Mathematical Models and Computation*. Oxford Financia Press, Oxford, 1993.

[54] A. Vande Wouwer, P. Saucez, and W. E. Schiesser. *Adaptive Method of Lines*. Chapman & Hall, Boca Raton, LA, 2001.

[55] L. Wu and Y. Kwok. A front-fixing finite difference method for the valuation of American options. *Journal of Financial Engineering*, pages 83–97, 1997.

[56] C. Zenger. *Sparse grids*. Proc. 6th GAMM seminar in Kiel, 1991.

[57] R. Zvan, P.A. Forsyth, and K.R. Vetzal. Penalty methods for american options with stochastic volatility. *Journal of Computational and Applied Mathematics*, 91:199–218, 1998.

[58] R. Zwan, K.R. Vetzal, and P.A. Forsyth. PDE methods for pricing barrier options. *Journal of Economic Dynamics and Control*, 24:1563–1590, 2000.

Acta Universitatis Upsaliensis

Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology 206

Editor: The Dean of the Faculty of Science and Technology

A doctoral dissertation from the Faculty of Science and Technology, Uppsala University, is usually a summary of a number of papers. A few copies of the complete dissertation are kept at major Swedish research libraries, while the summary alone is distributed internationally through the series Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology. (Prior to January, 2005, the series was published under the title "Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology".)

Distribution: publications.uu.se
urn:nbn:se:uu:diva-7097

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2006