Conservative and Finite Volume Methods for the Pricing Problem

Master Thesis
M.Sc. Computer Simulation in Science

Germán I. Ramírez-Espinoza
Faculty of Mathematics and Natural Science
Bergische Universität Wuppertal

Supervisor: Prof. Dr. Matthias Ehrhardt

Abstract

This master thesis aims to solve some of the issues present in the simulation of the Black-Scholes partial differential equation (PDE) for the pricing problem if the hyperbolic behavior dominates. Hyperbolic behavior in a convection-diffusion equation like the Black-Scholes equation causes standard numerical methods to fail to deliver acceptable approximations. For European options, the hyperbolic behavior appears when the ratio of the risk-free interest rate and the squared volatility – known in fluid dynamics as Péclet number – is high. For Asian options, in addition to present hyperbolic behavior in when the Péclet number is high also present this behavior in other cases: when the spatial variable is approaching zero or when the maturity is small.

Three methods to obtain approximations to the Black-Scholes PDE are studied: the general Exponentially Fitted scheme, a Finite Volume method specially suited to the Black-Scholes equation, and the Kurganov-Tadmor scheme for a general convection-diffusion equation. Emphasis is put in the Kurganov-Tadmor scheme because its flexibility allows the simulation of a great variety of types of options and because its simplicity in comparison to the Finite Volume method. In addition to that, the Kurganov-Tadmor scheme exhibits quadratic convergence whereas the others only linear convergence. To support the claims of flexibility, simplicity and convenience of the Kurganov-Tadmor scheme, extensive experiments and comparisons are presented with different PDEs and even a nonlinear Black-Scholes equation.

Due to the characteristics of the Kurganov-Tadmor discretization, exact solution of the boundary conditions are obtained – when the conditions are itself a PDE – and implemented into the numerical scheme. For the similarity reduction proposed by Wilmott, a put-call parity is developed.

According to the author's knowledge, this is the first time the Kurganov-Tadmor scheme is applied to option pricing problems. In addition to that, a modification to the Wang's finite volume method is proposed as a way to avoid numerical issues present on the original formulation.

Erklärung

Ich versichere, dass meine Masterarbeit selbständig verfasst wurde sowie keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht wurden.

Wuppertal, 15.12.2011

Germán I. Ramírez-Espinoza

Contents

Part I.

Introduction

1. Introduction

The importance of financial options is evident when we take account of the volume of instruments traded on organized markets during different periods of 2011: in September, nearly 360 million options on equities were traded on the U.S. market [OCC11]. If we also include options on indices, futures, etc., we have a total of 400 million contracts and if we sum the contributions of each month starting in January 2011 up to the end of September 2011, then we found that ~ 3.5 billion contracts have been traded.

An option is an instrument in which two parties agree to the possibility to exchange an asset, the underlying, at a predefined price and maturity. Because of the stochastic nature of the price of the underlying asset, the profit or loss (P&L) at maturity is unknown and instead a profile of the P&L is given for a range of prices; this profile is known as the payoff of the option. There are a great variety of options ranging from European options, to American, Asian, Barrier options, Binary options, etc., and many of these instruments are valuated with the pricing formulae developed by Fischer Black, Myron Scholes and Robert Merton. The type of the option refers in many cases to the type of payoff profile of the option but for the European and American option, the type refers to the maturity of it: the maturity of an European option is fixed whereas the American option can be exercised at any time before the maturity. In this sense, there exist Asian options of European and American type, for instance. Despite the simplifications made in the original formulation – namely, no transaction costs, no arbitrage opportunities and constant volatility and risk-free interest rate – the Black-Merton-Scholes mathematical framework is of interest to researchers and practitioners and can be extended to include, for example, stochastic volatility.

The partial differential equation (PDE) proposed by Black, Scholes and Merton is known as the Black-Scholes equation and is a particular case of the more general convection-diffusion equation that also arises in other areas of science like fluid dynamics. Loosely speaking, the convection diffusion equation can be seen as the combination of a first order hyperbolic PDE and the diffusion equation. Due to the hyperbolic term, the solution is a traveling wave transporting the initial condition (IC) and due to the diffusive term the IC is dissipated: a dissipating traveling wave. When the diffusion contribution to the solution is small – i.e. the coefficient is small in comparison to the coefficient of the hyperbolic term – then the solution behaves, to some extent, as a traveling wave only and the convection-diffusion

equation is said to present hyperbolic behavior – also denoted in the literature as convection-dominated.

For purely first order hyperbolic PDEs, it is known [GRS07] that standard methods fail to obtain an acceptable approximation when discontinuities are present in the initial condition and a similar issue is observed on the convection-diffusion equation under a convection-dominated environment and discontinuous initial condition. Some schemes like the Lax-Friedrichs or the Upwind method were proposed to obtain satisfactory approximations for hyperbolic PDEs, but artificial diffusion is introduced by the method which leads to smeared solutions – see [Pul10] for examples with these schemes.

In terms of the Black-Scholes equation, the hyperbolic behavior appears when the squared volatility is small in comparison with the risk-free rate. Other PDEs proposed for the pricing of Asian options, in addition to be convection-dominated when the ratio of the risk-free rate and the squared volatility is high are also convection-dominated when the maturity is small or when the spatial variable is approaching zero.

When solving numerically the Black-Scholes PDE it is useful to transform the time variable to use the payoff function – known terminal condition – as the initial condition of the system. Albeit the payoff of an European option is only non-smooth and the numerical solution for the price is acceptable, artificial oscillations appear near the strike price when the first numerical derivative with respect to the underlying price of this approximation is obtained. These oscillations are worst when higher derivatives are calculated. Having access to the first derivative of the option price is important to measure the sensitivity of the option to movements on the price of the underlying or other parameters like volatility. For example, if the price of the option is denoted as $v(s,t)$ and the price of the underlying as s, then the elasticity of the option price with respect to the asset price is obtained as $\frac{\partial v}{\partial s}\frac{s}{v}$. Higher derivatives of the option price provide also important information about the behavior of the option. These quantities are known in the financial literature as the *Greeks*. Due to the Greeks being relevant for the quantitative analysts, reliable numerical methods are required for the pricing of options which not only provide a good approximation for the price, but also for the derivatives of the price. In addition to the issues encountered when obtaining the Greeks, some options, like the Binary type, define discontinuous payoffs which are difficult to deal with in order to obtain an accurate, reliable approximation to the price.

Three families of options are specially interesting for researchers because they represent benchmark problems to study: European, American and Asian options. For plain-vanilla European option, the PDE has one temporal independent variable and one spatial independent variable – i.e. the underlying price – whereas for an Asian option an additional spatial variable is introduced into the PDE. The Black-

Scholes equation has a closed-form solution for the case of a plain-vanilla European option and for some Asian options but for any other case, numerical methods are needed.

There are two main alternatives to obtain the price of an option: the price can be summarized as the discounted expectation at $t = 0$ of the price of the option at $t = T$, i.e. the payoff

$$v(s, 0) = \exp(-rT)\,\mathbb{E}[v(s, T)],$$

and a good alternative to calculate the expectation is simulating repeatedly the stochastic differential equation that drives the price of the underlying asset. This method is known as Monte Carlo (MC) simulation and the reasoning behind MC methods is to perform large simulations that provides the analyst with a meaningful approximation of the expectation of the price of the option. A large computational effort is needed when using MC methods for the pricing models because the convergence of the method to the solution is slow, but its simplicity represents many advantages when the dimensionality of the problem is high. The second alternative is to use the mathematical model defined by the Black-Scholes equation and solve it numerically. Solving a PDE numerically has the advantage of a well studied area – a vast literature is available – and high efficiency in terms of computational and memory costs when the dimensionality is low: most methods to obtain approximations to solutions of PDEs have a finite number of operations and in addition to that, good convergence properties can be achieved. For finite difference methods a rule of thumb is to consider the dimensionality d of the problem low when $d \leq 3$. This is considered so because, for example, when $d = 3$ and 500 discretization points are required, then the system matrix is composed of $500 \times 500 \times 500 = 125,000,000$ elements and, in general, it is a dense matrix which requires $\sim$ 1GB in memory when using double-precision floating numbers. When $d > 3$ then it is difficult to achieve efficiency with finite difference methods and a Monte Carlo technique shall be used.

In this work, finite difference methods for the convection-diffusion equation are presented. Our main purpose is to propose reliable methods for the Black-Scholes equation with a wide range of parameters, including the convection-dominated case. Conservative methods, a special family of finite difference methods and also denoted as Finite Volume methods in the literature, are presented as a the method of choice to solve convection-dominated problems. Conservative numerical methods arise in the study of conservation laws and its computational modeling.

Two conservative methods are studied: the Kurganov-Tadmor scheme [KT00], a high resolution method for a general convection-diffusion equation which exhibits quadratic convergence and introduces small artificial viscosity in comparison to other methods like Lax-Friedrichs; and the Wang scheme [Wan04] for an European

option in which the flux of the Black-Scholes equation in conservative form is solved analytically and exhibits linear convergence. We also present the Exponentially Fitted scheme first proposed by Il'in [Il'69] and then presented in the context of finance by Duffy [Duf06]. According to the author's knowledge, this is the first time the Kurganov-Tadmor scheme is applied to option pricing problems.

This thesis is structured as follows: Section 2 introduces the basis of options and the general concept of put-call parity along with an expression for the case of an European option. In Section 3 a standard derivation for the Black-Scholes is presented alongside with the full PDE for Asian options and a similarity reduction based on the mentioned full PDE, proposed by Wilmott [WDH94]; a third PDE for Asian options proposed by Rogers-Shi [RS95] is presented in this section. Based on the put-call parity for Asian options, we obtained an expression for the put-call parity for both of the similarity reduction presented. These expressions are useful to obtain boundary conditions. In Section 4 the finite difference method is introduced and the concepts of consistency, stability and convergence are outlined. The derivation of finite differences is based on Taylor expansion of the solution of the PDE. The concepts of explicit and implicit methods are shown and the Exponentially Fitted method, the finite volume method and the Kurganov-Tadmor schemes are presented in a simplified, general manner, following each author's derivation. The Section 5 shows the results of solving numerically the Black-Scholes equation with the methods delineated in Section 4. Simulations with a fully implicit method, the Exponentially Fitted schemes, Wang's finite volume method and Kurganov-Tadmor scheme are shown. Extensive comparisons between the Kurganov-Tadmor scheme and other existing methods are performed. Our Conclusions are presented in Section 6. A prototype Matlab code can be found in Appendix A.

Notation, conventions and simulation times

In finance literature it is common to represent the price of the underlying as S and the price of an option as $V(S,t)$. In this work, mathematical functions are stated in lower case letters whereas numerical approximations to those function are denoted with capital letters. In this sense, the price of an option is denoted as $v(s,t)$ and its approximation as $V_i^n \approx v(s_i,t_n)$ – c.f. Chapter 4 for a detailed explanation of how s_i and t_n are defined. On the other hand, sub-script notation and variable names like r, σ, γ, α, etc, are valid only Chapter-wise.

Simulations on this thesis work were performed on a computer with an Intel i5 M480 processor and 8GB of Ram with a 64-bit Matlab 2011b under Kubuntu Linux version 11.04. Execution times must be interpreted within this context.

2. Options

2.1. Definition

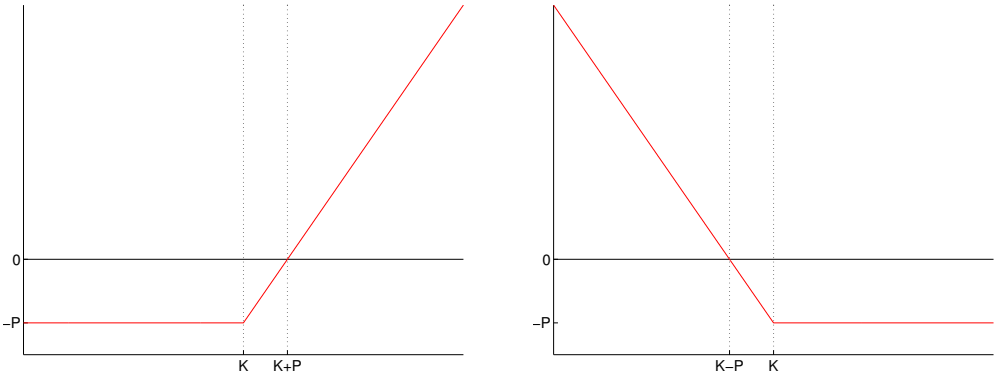
An option is a financial instrument in which two parties agree to exchange an asset at a predefined price or *strike* and date or *maturity*. By paying an up-front quantity – known as the price or *premium* of the option – the holder of the contract has the right, but not the obligation, to buy/sell the asset at maturity. The underlying asset on the contract is typically a stock or a commodity but the possibilities are immense; for instance, it is possible to create an option with a *future* or a *swap* as the underlying – the latter is called *swaption* in the financial literature.

An option in which the holder has the right to buy the underlying is a Call option whereas if the contract gives the holder the right to sell the underlying then it is denominated as a Put option.

This financial instrument could be used to hedge against unexpected conditions in the market but also as a trading strategy. For example, a corn producer could buy a put option in order to protect its production from unfavorable changes on the price of the commodity. On the other hand, a hedge fund manager, based on its beliefs, quantitative analyses or knowledge of the market, could use options to profit from volatility on the prices of certain stock.

The value of a call option from the perspective of the holder at maturity time is shown in Figure 2.1.1a. The price of the option is denoted as P and the strike price as K. The x-axis represents the price of the underlying asset whereas the red line represents the value of the option. If the price of the underlying is less than the strike price, the option is worthless for the holder because it is possible to buy the underlying at a lower price, i.e. market price. When the price of the underlying is greater than $K + P$ then the value of the option increases and the holder of the option profits from the difference $s_T - P - K$ where s_T is the price of the underlying at maturity. In Figure 2.1.1b the payoff of a put option is shown and in this case the holder profits from the difference $K - s_T - P$. In both cases, the risk for the holder is limited to the premium of the option, but the profit is unlimited, theoretically, in the case of a call option and bounded by the strike in the case of a put option.

Both payoffs in Figure 2.1.1 are referred as *long* positions on an option. In financial terminology, *going long* on a financial instrument is used as a synonym



(a) Call option. (b) Put option.

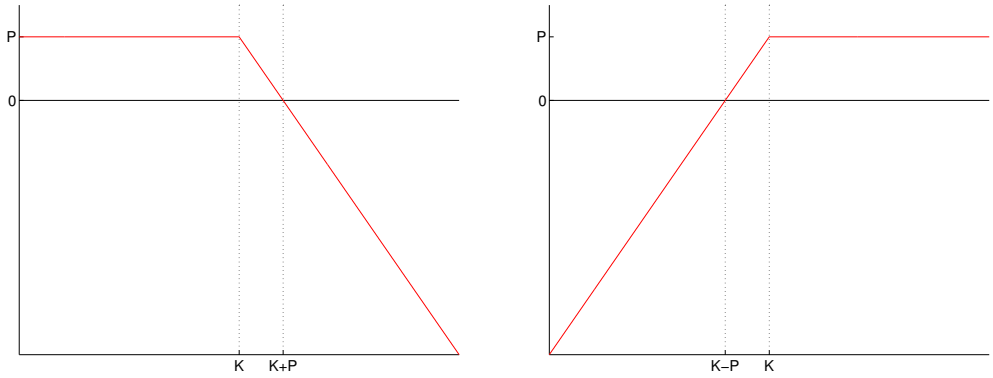
Figure 2.1.1.: Payoffs of a *long* position on an option with strike price K and premium P.

of buying it and therefore benefiting when the price increases. This terminology applies for a great variety of instruments like futures, options, stocks, commodities, swaps, bonds, etc.

In contrast to the long positions, *short* positions on a financial instrument is synonym of selling it. The payoff for a short position on an option is shown in Figure 2.1.2. To sell an asset is a simple concept, however, in finance, it is possible to *go short* on a stock – or some other instruments – without owning it by first borrowing the stock from a broker – usually a bank like JPMorgan or UBS – and then sell it in the market. In addition to that, a *naked short sell* refers to the case when a short position is taken without first having borrowed the financial instrument. Short selling is a risky operation: by going short on a stock, for example, the borrower could incur in big loses. Nevertheless, the ability to take short positions or naked short positions gives the market great flexibility to create instruments for different purposes: creating strategies with long and short positions on options could help further to hedge against undesired movements on prices – c.f. Section 2.2.

By looking at Figure 2.1.2 it is evident the inherent risk when taking a short position on an option, specially for the case of a call in which an unexpected rise on the price of the underlying could cause considerable loses.

What we have described up to now as an option is known in the financial literature as *European* option. Another important type of option is known as *American* option which can be exercised at any time before the maturity, but the payoff is the same as in the case for the European type. A third type are the Asian options that define a payoff dependent on the average of the price of the underlying.



(a) Call option. (b) Put option.

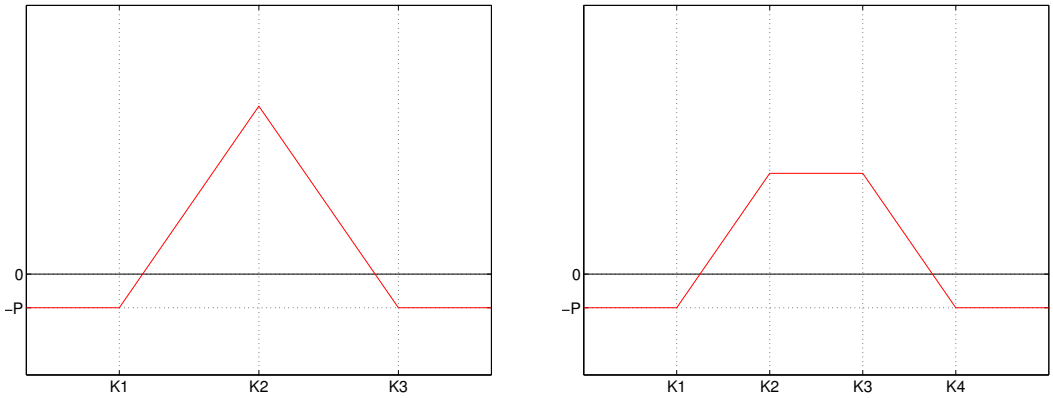
Figure 2.1.2.: Payoffs of a *short* position on an option with strike price K and premium P.

2.2. Strategies with Options

Traders often combine long and short options with different strike price in order to create *option strategies*:

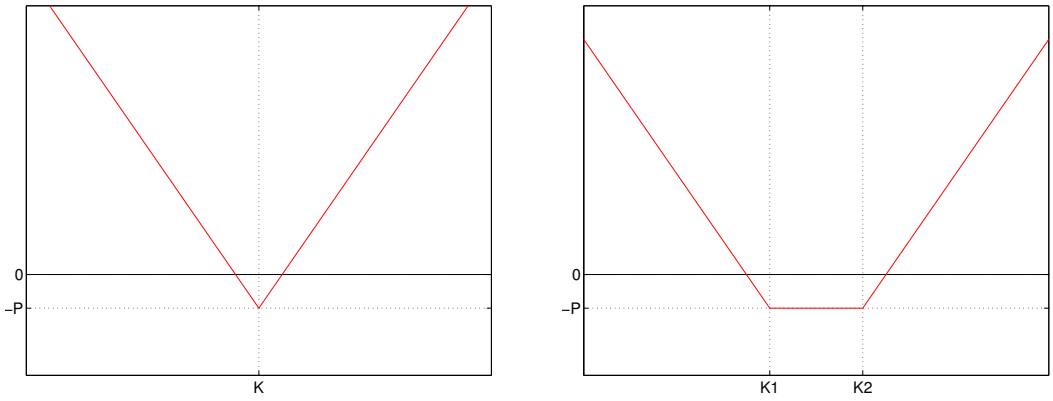
- **Butterfly**: is used when it is expected that the price of the underlying will remain in the vicinity of K_2. The components are: a long call with strike K_1, two short calls with strike K_2 and one long call with strike K_3, with $K_1 < K_2 < K_3$.
- **Condor**: similar to a butterfly but with a wider range which allows to contain slightly higher volatilities than the butterfly. The components of the condor are: long call with strike K_1, short call with strike K_2, short call with strike K_3 and a long call with strike K_4, with $K_1 < K_2 < K_3 < K_4$.
- **Straddle**: this strategy is used when a high volatility is expected on the price of the underlying but no directional information is available, i.e. it is unknown whether it will be an upside or a downside volatility. The components are just a put and a call with the same strike price.
- **Strangle**: a major movement is expected with uncertainty on the direction. The components are a long put with strike K_1 and a long call with a strike price K_2, with $K_1 < K_2$.

These strategies, payoffs shown in Figure 2.2.1, are just some simple cases that exemplify how options can be combined to profit or hedge under different market conditions. More elaborated instruments can be created when options with different strike and different maturity are combined.



(a) Butterfly.

(b) Condor.



(c) Straddle.

(d) Strangle.

Figure 2.2.1.: Different strategies with options.

To obtain a fair price of an option is not an easy task and researchers have worked extensively in this area to provide different methods and models for the *pricing* problem. Assuming an underlying that follows a geometric Brownian motion, the Black-Scholes equation, introduced in Section 3, is between the most prominent models for the pricing of an option.

2.3. Put-Call Parity

For the same maturity, strike price and underlying, a relation between the price of a call and a put option under a frictionless market can be defined. This relationship is known as the *put-call parity* and arises from the fact that with combinations of long/short calls and long/short puts it is possible to create *synthetic* instruments with the same payoff as the real ones. For instance, by combining a long call and a short put on a stock, it is possible to create an instrument with the same payoff as the underlying, i.e. a synthetic stock; it is also possible to create a synthetic long call by creating a portfolio of a long put and holding a stock. A relation between a call and a put in terms of the price of the stock must be fulfilled in order to keep the arbitrage-free market.

For an European option with maturity T and strike K and ignoring the premium of the option, we can create two portfolios to obtain the desired relationship. In the first portfolio a long call and a short put is held with payoff $s-K$. The second portfolio holds a long stock and K bonds with maturity T that pays a unit of currency at T, achieving a payoff $s-K$. By arbitrage arguments, these portfolios must have the same price at time t

$$v_C(s,t) - v_P(s,t) = s_t - Kb(t,T), \tag{2.3.1}$$

where $b(t,T)$ is the price of the bond with maturity T, $v(s,t)$ the price of the option and s_t the price of the stock at t. A constant risk-free interest rate – required by Black-Scholes – defines the price of the bond as

$$b(t,T) = \exp(-r(T-t)),$$

which completes relation 2.3.1.

Besides the obvious theoretical and practical importance of the Put-Call parity, it is also useful in numerical analysis to obtain boundary conditions for pricing schemes: when the boundary conditions are known only for a put or call, the unknown boundary conditions for the other instrument can be easily obtained with equation 2.3.1.

Part II.

The Mathematical Model

3. The Black-Scholes Equation

The Black-Scholes equation is an important mathematical model for the pricing of financial derivatives. The model assumes the following:

1. The price of the underlying follows a geometric Brownian motion (GMB).
2. Arbitrage-free world, i.e. the price for an asset is the same in all markets.
3. The risk-free interest rate is constant.
4. The volatility of the underlying is constant during the period of the contract.
5. The option is of European type.

Given point 1, the model for the price of the underlying, a GMB, is defined as

$$ds_t = \mu s_t dt + \sigma s_t dW_t, \tag{3.0.1}$$

where W_t is a Wiener process, μ is the drift coefficient and σ is the diffusion coefficient, both constant. A bond, considered a risk-less investment, follows the process

$$dB_t = rB_t dt. \tag{3.0.2}$$

Equation (3.0.1) is also known an an Itô process and assuming a function to be $v(s_t, t)$ a $C^{2,1}$ smooth function, we have that $v(s_t, t)$ follows an Itô process with the same Wiener process

$$dv = \left(\mu s \frac{\partial v}{\partial s} + \frac{\partial v}{\partial t} + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 v}{\partial s^2}\right) dt + \sigma s \frac{\partial v}{\partial s} dW_t.$$

By constructing a portfolio with α_t shares of the asset and β_t shares of bond, we have that the value of it is

$$\Pi_t = \alpha_t s_t + \beta_t B_t, \tag{3.0.3}$$

and it is assumed that the portfolio can be rebalanced but no influx or out-flux of money is allowed, i.e.

$$d\Pi_t = \alpha_t ds_t + \beta_t dB_t,$$

and substituting equations (3.0.1) and (3.0.2) we get

$$d\Pi_t = (\alpha_t \mu s_t + \beta_t r B_t)\, dt + \alpha_t \sigma s_t dW_t.$$

Our goal when constructing this portfolio is emulating the price of an option, i.e.

$$\begin{aligned} \Pi_t &= v_t, \\ d\Pi_t &= dv_t, \\ (\alpha_t \mu s_t + \beta_t r B_t)\, dt + \alpha_t \sigma s_t dW_t &= \left(\mu s \frac{\partial v}{\partial s} + \frac{\partial v}{\partial t} + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 v}{\partial s^2}\right) dt + \sigma s \frac{\partial v}{\partial s} dW_t. \end{aligned} \tag{3.0.4}$$

By comparison we obtain

$$\alpha_t = \frac{\partial v}{\partial s}$$

and

$$\mu s \frac{\partial v}{\partial s} + \beta_t r B_t = \mu s \frac{\partial v}{\partial s} + \frac{\partial v}{\partial t} + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 v}{\partial s^2}, \tag{3.0.5}$$

where $s_t = s$. With equation (3.0.3) and (3.0.4) we obtain an expression for B

$$\begin{aligned} s\frac{\partial v}{\partial s} + \beta B &= v, \\ \beta B &= v - s\frac{\partial v}{\partial s}, \end{aligned}$$

or, equivalently

$$r\beta B = rv - rs\frac{\partial v}{\partial s}. \tag{3.0.6}$$

Substituting equation (3.0.6) into (3.0.5) results in the Black-Scholes PDE

$$\frac{\partial v(s,t)}{\partial t} + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 v(s,t)}{\partial s^2} + rs\frac{\partial v(s,t)}{\partial s} - rv(s,t) = 0, \tag{3.0.7}$$

in the interval $s \in [-\infty, \infty]$ and for $t \in [0, T]$. The parameters are listed in Table 3.1.

For a call option, the boundary conditions can be expressed as

$$\begin{aligned} v(0,t) &= 0, \\ v(s,t) &\to s\exp(-d(T-t)) - K\exp(-r(T-t)), \text{ for } s \to \infty, \end{aligned}$$

and the terminal condition is defined as

$$v(s,T) = (s-K)^+. \tag{3.0.8}$$

Parameter	
r:	Continuously compounded, annualized risk-free rate.
σ:	Volatility of the stock price.
s:	Stock price.
K:	Strike price.
T:	Maturity.

Table 3.1.: List of parameters for the Black-Scholes equation.

with the notation $(a)^+ := \max(a, 0)$

For a put option, the boundary conditions are

$$\begin{aligned} v(s,t) &= K\exp(-r(T-t)) - s\exp(-d(T-t)), \text{ for } s \to 0 \\ v(s,t) &= 0, \text{ for } s \to \infty, \end{aligned}$$

and the terminal condition is defined as

$$v(s,T) = (K-s)^+. \tag{3.0.9}$$

The boundary and terminal conditions fully define the problem (3.0.7) for the function $v(s,t)$.

Multidimensional Black-Scholes Equation

The general n-factor model is described by the process

$$\begin{aligned} dS_i &= (\mu_i - \delta_i) S_i dt + \sigma_i S_i dW^{(i)}, \ i = 1, \ldots, n, \\ E\left(dW^{(i)} dW^{(j)}\right) &= \rho_{ij} dt, \ i, j = 1, \ldots, n, \end{aligned}$$

where ρ_{ij} is the correlation between the asset i and asset j, and δ_i denotes the dividend flow paid by the ith asset. The Black-Scholes type PDE of the model is

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sum_{i,j=1}^{n} \rho_{ij}\sigma_i\sigma_j S_i S_j \frac{\partial^2 V}{\partial S_i \partial S_j} + \sum_{i=1}^{n} (r - \delta_i) S_i \frac{\partial V}{\partial S_i} - rV = 0$$

3.1. Asian Options

An Asian option is a specific type of so-called path-dependent options in which the payoff is determined by the average of the price of the underlying instrument. The price of an Asian option is then denoted as $v(s, a, t)$ where $a(t)$ is the average of s.

The concept of Asian options was motivated as a way to further reduce the risk of market manipulation on the price of the underlying asset. For example, an issuer of a plain-vanilla European option with a stock as underlying could be exposed to induced price movements at maturity leading to loses. On the other hand, a company or financial institution looking to hedge certain asset is exposed to steep movements on its price. Although steep movements could not be common on the stock market – on normal market conditions – it is common to experience unexpected changes in the commodity market in a matter of days: gold prices went from $\sim$ 1800 to $\sim$ 1600 USD in just five days during September 2011 [KIT11]. An hypothetical gold mining company using options to hedge its production could be less vulnerable to volatility by using an Asian option instead of an American or European one.

By using the continuous arithmetic average over the interval $[0, t]$, then

$$a(t) := \frac{1}{t}\int_0^t s(\tau)\, d\tau;$$

the equation (3.0.7) must be modified to include the new term. The inclusion of the average $a(t)$ leads to a new dimension. The pricing equation is

$$\frac{\partial v}{\partial t} + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 v}{\partial s^2} + rs\frac{\partial v}{\partial s} - rv + \frac{1}{t}(s-a)\frac{\partial v}{\partial a} = 0, \tag{3.1.1}$$

with the following payoff or terminal condition:

$$\text{for a call} \quad \begin{cases} (a-K)^+ & \text{fixed strike,} \\ (s-a)^+ & \text{floating strike,} \end{cases}$$

$$\text{for a put} \quad \begin{cases} (K-a)^+ & \text{fixed strike,} \\ (a-s)^+ & \text{floating strike.} \end{cases}$$

The boundary condition for (3.1.1) at $s = 0$ is

$$\frac{\partial v}{\partial t} - \frac{a}{t}\frac{\partial v}{\partial a} - rv = 0,$$

whereas for $s \to \infty$ is

$$\frac{\partial v}{\partial t} + \frac{1}{t}(s-a)\frac{\partial v}{\partial a} = 0.$$

From equation (3.1.1) it can be observed that there are no diffusion terms for $a(t)$, i.e. purely hyperbolic behavior is expected in that direction.

The put-call parity for an Asian option takes the form

$$v_C - v_P = s - \frac{s}{rT}\left[1 - \exp\left(-r(T-t)\right)\right] - \exp\left(-r(T-t)\right)\frac{1}{T}\int_0^t s(\tau)\, d\tau. \tag{3.1.2}$$

3.1.1. The Wilmott Similarity Reduction

It is possible to reduce the full PDE for Asian options (3.1.1) to one spatial and one temporal dimension by using a similarity reduction proposed by Wilmott [WDH94].

Let us consider the floating strike payoff of a call option

$$(s-a)^+ = s\left(1 - \frac{1}{st}\int_0^t s(\tau)\,d\tau\right),$$

and by letting

$$x = \frac{1}{s}\int_0^t s(\tau)\,d\tau, \tag{3.1.3}$$

it is possible to define the separation ansatz $v(s,a,t) = s \cdot y(x,t)$. Substituting this ansatz into (3.1.1) we get

$$\frac{\partial y}{\partial t} + \frac{1}{2}\sigma^2 x^2 \frac{\partial^2 y}{\partial x^2} + (1-rx)\frac{\partial y}{\partial x} = 0, \tag{3.1.4}$$

with the terminal condition

$$y(x,T) = \left(1 - \frac{1}{T}x\right)^+. \tag{3.1.5}$$

The boundary conditions for a call are easily obtained by observing the limits of (3.1.4). For example for $x = 0$, the equation (3.1.4) is

$$\frac{\partial y}{\partial t} + \frac{\partial y}{\partial x} = 0, \tag{3.1.6}$$

because, assuming $y(x,t)$ is bounded, it is possible to show that the term

$$x^2 \frac{\partial^2 y}{\partial x^2} \to 0 \text{ for } x \to 0,$$

whereas for the case of $x \to \infty$ it can be seen from the payoff (3.1.5) that the option is not exercised, therefore

$$y = 0. \tag{3.1.7}$$

This PDE for Asian options is advantageous, computationally speaking, because we handle only one spatial and one temporal dimension in contrast to the full expression (3.1.1) which requires two spatial and one temporal dimension leading to considerably higher computational costs and higher memory requirements.

With the ansatz $v(s,a,t) = s \cdot y(x,t)$, the put-call parity takes the form

$$\begin{aligned} y_C - y_P &= 1 - \frac{1}{rT}\left[1 - \exp\left(-r\left(T-t\right)\right)\right] - \exp\left(-r\left(T-t\right)\right)\frac{1}{sT}\int_0^t s\left(\tau\right)d\tau, \\ y_C - y_P &= 1 - \frac{1}{rT}\left[1 - \exp\left(-r\left(T-t\right)\right)\right] - \exp\left(-r\left(T-t\right)\right)\frac{x}{T}, \end{aligned}$$

where the definition of the new independent variable 3.1.3 was used in the last part.

In financial literature the ansatz is denoted as $v(s,a,t) = s \cdot H(R,t)$ and

$$\frac{\partial H}{\partial t} + \frac{1}{2}\sigma^2 R^2 \frac{\partial^2 H}{\partial R^2} + (1 - rR)\frac{\partial H}{\partial R} = 0,$$

however, to avoid confusion with the numerical flux defined in Section 4.8, we chose a different notation.

A drawback of the reduction is that it is only possible reduce the PDE for the case of a floating strike options.

3.1.2. The Rogers-Shi Reduction

An alternative PDE was presented in [RS95] using a scaling property of the GMB.

A new variable is defined as

$$x = \frac{1}{s}\left[K - \int_0^t s\left(\tau\right)\mu\left(d\tau\right)\right],$$

with μ as probability measure with density $\rho(t)$ such that

$$\rho(t) = \begin{cases} \frac{1}{T}, & \text{for a fixed strike option,} \\ \frac{1}{T} - \delta\left(T-\tau\right), & \text{for a floating strike option,} \end{cases}$$

where in the case for a floating strike option, K must be set to zero.

The proposed PDE is

$$\frac{\partial w}{\partial t} + \frac{1}{2}\sigma^2 x^2 \frac{\partial^2 w}{\partial x^2} - \left(\rho(t) + rx\right)\frac{\partial w}{\partial x} = 0$$

with the following terminal condition for a fixed strike call option

$$w(x,T) = \min(0,x) =: (x)^-, \tag{3.1.8}$$

and for a floating strike put option

$$w(x,T) = (1+x)^-. \tag{3.1.9}$$

Boundary conditions are defined depending on the type of payoff. For a fixed strike call we have that

$$w(x,t) = \frac{\exp(r(T-t)) - 1}{r} - x, \text{ for } x < 0,$$

and from the payoff (3.1.8) we obtain the boundary

$$w(x,t) = 0 \text{ for } x \to \infty.$$

On the other hand, for a floating strike put we have that

$$w(x,t) = \frac{\exp(r(T-t)) - 1}{rT} - \exp(r(T-t)) - x, \text{ for } x \ll 0$$

and, again, from the corresponding payoff (3.1.9) it is possible to obtain the other boundary

$$w(x,t) = 0 \text{ for } x > -1.$$

The price of the option is then $s_0 w(K/s_0, 0)$ for the case of a fixed strike option and $s_0 w(0,0)$ for the case of a floating strike, where s_0 is the current price of the underlying.

Part III.

Numerical Aspects

4. Finite Difference Methods

Finite difference methods (FDM) are simple yet powerful techniques to solve PDEs numerically. In FDM the PDE's partial derivatives are replaced by discrete approximations obtained via Taylor expansions. An error is introduced by the truncation of the infinite Taylor series, but the main goal is to maintain this error bounded and low. Consistency, stability and convergence are required for a method to be considered useful.

4.1. Fundamentals of FDM

Without loss of generality, let us consider a function with one spatial variable and one temporal variable $u(x,t)$. Its pointwise approximation is $U_i^n := u(x_i,t_n) + \epsilon$ where ϵ is the truncation error.

The space of integration is divided in discrete points and the Taylor expansion evaluated on these grid points. In this thesis we prefer to divide the space in $N+2$ grid points such that

$$a = x_0 < x_1 < x_2 < \cdots < x_N < x_{N+1} = b,$$

where $x \in [a,b]$, while it is preferred that the temporal variable $t \in [0,T]$ is discretized in M grid points such that

$$0 = t_1 < t_2 < \cdots < t_M = T,$$

but in some cases it will be convenient to have the time variable defined as $t^* = T - t$, therefore

$$T = t_1 > t_2 > \cdots > t_M = 0.$$

For the case of Dirichlet boundary conditions x_0 and x_{N+1} represent known boundary conditions, i.e.

$$\begin{aligned} u(a,t) &= g_a(t), \\ u(b,t) &= g_b(t), \end{aligned}$$

whereas for the von Neumann boundary conditions, the derivative of the function is known

$$\begin{aligned} \frac{\partial u(a,t)}{\partial x} &= \alpha_1(t), \\ \frac{\partial u(b,t)}{\partial x} &= \alpha_2(t). \end{aligned}$$

Depending on the PDE, we may also need to prescribe a initial condition of the form

$$u(x,0) = g_t(x).$$

Finally, we define a step-size for each dimension as the distance between grid points

$$\Delta x = \frac{b-a}{N+1}, \tag{4.1.1}$$

$$\Delta t = \frac{T}{M-1}. \tag{4.1.2}$$

In numerical analysis, it is common to denote the truncation error with the Landau symbols – sometimes know as the big-O notation: let f and g be two functions of the continuous, real variable x defined on a subset of $\mathbb{R}$. If

$$f(x) \leq Cg(x) \text{ as } x \to 0,$$

for some constant C independent of x, then we write $f(x) = \mathcal{O}(g(x))$.

For instance, the infinite Taylor expansion for $u(x+\Delta x, t)$ is

$$u(x+\Delta x,t) = u(x,t) + \Delta x\frac{\partial u(x,t)}{\partial x} + \frac{1}{2}\Delta x^2\frac{\partial^2 u(x,t)}{\partial x^2} + \frac{1}{3!}\Delta x^3\frac{\partial^3 u(x,t)}{\partial x^3} + \cdots,$$

which is computationally intractable. Instead, an approximation is defined up to a certain order

$$u(x+\Delta x,t) = u(x,t) + \Delta x\frac{\partial u(x,t)}{\partial x} + \mathcal{O}\left(\Delta x^2\right).$$

In this sense, the big-O notation defines an upper limit for the terms that are truncated, i.e.

$$\frac{1}{2}\Delta x^2\frac{\partial^2 u(x,t)}{\partial x^2} + \frac{1}{3!}\Delta x^3\frac{\partial^3 u(x,t)}{\partial x^3} + \cdots \leq C\left|\Delta x^2\right|,$$

where C is a positive, small, real number that do not depend on x.

4.1.1. Discrete First Derivative

Again, without loss of generality, the first partial derivative of a function $u(x,t)$ with respect to x is considered. Obtaining the Taylor expansion for the function u at $x+\Delta x$ we get

$$\begin{aligned} u(x+\Delta x,y) &= u(x,t)+\Delta x\frac{\partial u(x,t)}{\partial x}+\frac{1}{2}\Delta x^2\frac{\partial^2 u(x,t)}{\partial x^2}+\frac{1}{3!}\Delta x^3\frac{\partial^3 u(x,t)}{\partial x^3}+\cdots \\ &= u(x,t)+\Delta x\frac{\partial u(x,t)}{\partial x}+\mathcal{O}\left(\Delta x^2\right), \end{aligned} \tag{4.1.3}$$

or, rearranging

$$\frac{\partial u(x,t)}{\partial x}=\frac{u(x+\Delta x,t)-u(x,t)}{\Delta x}+\mathcal{O}(\Delta x). \tag{4.1.4}$$

The expression (4.1.4) is a first-order approximation for the first partial derivative. A second-order approximation is achieved by subtracting

$$u(x-\Delta x,t)=u(x,t)-\Delta x\frac{\partial u(x,t)}{\partial x}+\frac{1}{2}\Delta x^2\frac{\partial^2 u(x,t)}{\partial x^2}-\frac{1}{3!}\Delta x^3\frac{\partial^3 u(x,t)}{\partial x^3}+\cdots \tag{4.1.5}$$

from (4.1.3), yielding

$$u(x+\Delta x,t)-u(x-\Delta x,t)=2\Delta x\frac{\partial u(x,t)}{\partial x}+\mathcal{O}\left(\Delta x^3\right)$$

and then solving for the desired term we get

$$\frac{\partial u(x,t)}{\partial x}=\frac{u(x+\Delta x,t)-u(x-\Delta x,t)}{2\Delta x}+\mathcal{O}\left(\Delta x^2\right). \tag{4.1.6}$$

By dropping higher order terms in (4.1.4) and (4.1.6) a Finite Difference approximation for the first derivative is obtained

$$\frac{\partial u(x_i,t_n)}{\partial x}\approx\frac{U_{i+1}^n-U_i^n}{\Delta x}$$

and

$$\frac{\partial u(x_i,t_n)}{\partial x}\approx\frac{U_{i+1}^n-U_{i-1}^n}{2\Delta x}.$$

Requirements of smoothness for the function $u(x,t)$ to obtain an approximation for the first derivative are $u\in C^2([a,b])$ for the first-order approximation and $u\in C^3([a,b])$ for the second-order approximation.

4.1.2. Discrete Second Derivative

Without loss of generality now we consider the second derivative of $u(x,t)$ with respect to x. We follow a similar technique as the one used for the second-order first derivative. In this case, we sum (4.1.3) and (4.1.5)

$$u(x+\Delta x,t)+u(x-\Delta x,t)=2u+\Delta x^2\frac{\partial^2 u}{\partial x}+\mathcal{O}\left(\Delta x^4\right).$$

The approximation is then

$$\frac{\partial^2 u(x_i,t_n)}{\partial x^2}\approx\frac{U_{i+1}^n-2U_i^n+U_{i-1}^n}{\Delta x^2}, \tag{4.1.7}$$

which is also of second order. Here, the function $u(x,t)$ is assumed to be smooth enough to obtain the approximation (4.1.7), namely $u\in C^4([a,b])$.

4.1.3. Discrete Mixed Derivative

We can obtain the mixed derivative of a function $u(x,t)$ in several ways. For example, with the expression

$$\frac{\partial^2 u(x,t)}{\partial t\partial x}=\frac{\partial}{\partial t}\left(\frac{\partial u(x,t)}{\partial x}\right)$$

and the second-order approximation (4.1.6) for the first derivative

$$\begin{aligned}\frac{\partial^2 u(x_i,t_n)}{\partial t\partial x} &\approx \frac{\partial}{\partial t}\left(\frac{u(x_{i+1},t)-u(x_{i-1},t)}{2\Delta x}\right)|_{t=t_n}\\ &\approx \frac{1}{4\Delta x\Delta t}\left(U_{i+1}^{n+1}-U_{i+1}^{n-1}-U_{i-1}^{n+1}+U_{i-1}^{n-1}\right)\end{aligned}$$

which is of second order in space and time. For more options to discretize mixed derivatives see [GRS07].

4.2. Consistency, Stability and Convergence

A numerical method is considered useful when certain properties are present with respect to the exact solution of the PDE we are studying. In most of the cases we do not have an exact solution to compare with but we can define these properties for a benchmark problem with a known solution and then generalize to other cases.

In this section we work with a general PDE of the form

$$\mathcal{L}u(x)=0,$$

where $\mathcal{L}$ is a differential operator and U_i an approximation to the solution at $u(x_i)$. The *global error* is defined as

$$e(x_i) = |u(x_i) - U_i|,$$

and *the local error* as

$$\tau(x_i) = |\mathcal{L}u(x_i) - \mathcal{L}_{\Delta x}u(x_i)|,$$

where $\mathcal{L}_{\Delta x}$ is a discretized differential operator.

A numerical method can be seen as a series of structured computations that transform the initial condition into the approximation to the solution. A desirable and important property is that the round-off and truncation errors are kept bounded during the series of steps in order to obtain a reasonable solution. This property is called the *stability* of the scheme. In the case of PDEs with constant parameters, the von Neumann stability analysis is an important tool to study the so-called *growth factor* of numerical schemes – c.f. Section 4.2.1. With the von Neumann analysis it is possible to obtain relations in terms of the step sizes in order to fulfill the stability condition. As an example we can see the condition imposed on explicit methods in Section 4.3 to achieve a stable scheme.

A discretized differential operator $\mathcal{L}_{\Delta x}$ is called *consistent* if the local error fulfills

$$\lim_{\Delta x \to 0} \tau(x_i) = 0$$

uniformly in $x \in [a, b]$, or is consistent of order p if

$$\tau(x_i) = \mathcal{O}(\Delta x^p)$$

uniformly in $x \in [a, b]$. The local error describes how well the exact solution satisfies the discretized differential operator.

A numerical method is *convergent* if the global error satisfies

$$\lim_{\Delta x \to 0} \left[\max_{x_i \in [a,b]} e(x_i) \right] = 0$$

or is convergent of order p if

$$\max_{x_i \in [a,b]} e(x_i) = \mathcal{O}(\Delta x^p).$$

The convergence is a rather important property: we can obtain an approximation to the solution and improve it to obtain a desired error tolerance, even for the cases when the analytic solution is unknown. Moreover, the convergence provides information on how fast the error will decrease when Δx is diminished; for instance, if $p = 2$ and we double the number of grid points, then the error will decrease by an order of 4. This property was used by Courant, Friedrichs and Lewy to prove the existence of solutions of PDEs [CFL28].

Theorem. Lax-Richtmyer Equivalence. *A consistent finite difference scheme for a partial differential equation for which the initial value problem is well-posed is convergent if and only if it is stable [Str89].*

For some finite difference schemes, the proof of convergence is much harder than the proof for consistency and stability, but Lax-Richtmyer theorem implies convergence when a scheme fulfills both consistency and stability.

When relying on Lax-Richtmyer theorem to prove a scheme convergent, no information on the order of convergence p is available, but it is possible to obtain an empiric or *computational order of convergence* by observing how the error decreases when the grid is made finer.

4.2.1. The von Neumann Stability analysis

The von Neumann analysis is useful to verify the stability of linear discretization schemes by expressing the approximation in terms of its Fourier modes

$$U_i^n = \sum_{\varphi} c_\varphi^n \exp\left(j\varphi i \Delta s\right),$$

where $j^2 = -1$ is the imaginary unit and φ represents the wave number. For linear PDEs we can restrict our considerations to one Fourier mode

$$U_i^n = c_\varphi^n \exp\left(j\varphi i \Delta s\right). \tag{4.2.1}$$

We are interested in investigating the growth of the error of the approximation from a time step t_1 to a time step t_2 where $t_1 < t_2$. Defining the *growth factor* as

$$G_\varphi = \frac{c_\varphi^{t_2}}{c_\varphi^{t_1}},$$

it follows that the scheme is stable if $|G| \leq 1$.

4.3. Explicit Methods

Let us now consider the initial-boundary value problem (IBVP) defined by the one-dimensional heat equation supplied with Dirichlet boundary conditions

$$\frac{\partial u\left(x,t\right)}{\partial t} = \frac{\partial^2 u\left(x,t\right)}{\partial x^2}, \tag{4.3.1}$$

$$u\left(x,0\right) = u_0\left(x\right), \tag{4.3.2}$$

$$u(a,t) = g_a\left(t\right), \tag{4.3.3}$$

$$u(b,t) = g_b\left(t\right), \tag{4.3.4}$$

where $x \in [a, b]$ and $t \in [0, 1]$.

Following the techniques from Section 4.1 a discrete expression of equation (4.3.1) is achieved by substituting the derivatives with the discrete approximations obtained in the last sections:

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} = \frac{U_{i-1}^n - 2U_i^n + U_{i+1}^n}{\Delta x^2}, \tag{4.3.5}$$

and rearranging the terms, it is possible to express the new approximation U_i^n in terms of known quantities:

$$\begin{aligned} U_i^{n+1} &= U_i^n + \frac{\Delta t}{\Delta x^2}\left(U_{i-1}^n - 2U_i^n + U_{i+1}^n\right), \\ U_i^{n+1} &= rU_{i-1}^n + (1-2r)\,U_i^n + rU_{i+1}^n, \text{ with } r = \frac{\Delta t}{\Delta x^2}. \end{aligned} \tag{4.3.6}$$

The method (4.3.6) is of order two in space and order one in time.

Expressing the explicit method as a system of equations leads to

$$U^{n+1} = A_e U^n,$$

where the system matrix A_e reads

$$A_e = \begin{bmatrix} (1-2r) & r & & \\ r & \ddots & \ddots & \\ & \ddots & & r \\ & & r & (1-2r) \end{bmatrix} \in \mathbb{R}^{N \times N}$$

and $U^n = (U_1^n, U_2^n, \ldots, U_N^n)^\intercal$, where points x_0 and x_{N+1} are excluded because of known boundary conditions. The explicit method is very economical in terms of computational complexity because only a matrix-vector multiplication is required on each time step and in addition to that, the matrix A_e is tridiagonal and therefore the matrix-vector multiplication computational effort is $\mathcal{O}(N)$.

Nonetheless, using the von Neumann analysis we can prove that the explicit method is stable only for certain values of the parabolic mesh ratio r. Replacing the ansatz (4.2.1) into equation (4.3.6) we get

$$\begin{aligned} c_\varphi^{n+1} \exp(j\varphi i \Delta s) = {} & r c_\varphi^n \exp(j\varphi(i-1)\Delta s) + (1-2r)\, c_\varphi^n \exp(j\varphi i \Delta s) \\ & + r c_\varphi^n \exp(j\varphi(i+1)\Delta s), \end{aligned}$$

dividing by $\exp(j\varphi i \Delta s)$ leads to

$$\begin{aligned} c_\varphi^{n+1} &= c_\varphi^n \left[r \exp(-j\varphi\Delta s) + (1-2r) + r \exp(j\varphi\Delta s)\right], \\ &= c_\varphi^n \left[1 - 2r\,(1 - \cos(\varphi\Delta s))\right], \end{aligned}$$

and hence the growth factor is

$$G_\varphi = \frac{c_\varphi^{n+1}}{c_\varphi^n} = 1 - 2r\left(1 - \cos\left(\varphi\Delta s\right)\right).$$

The term $1-\cos\left(\varphi\Delta s\right) \in [0,2]$ and therefore we need $r \leq 1/2$, so that $|G_\varphi| \leq 1$. This is a rather restrictive condition, typical for explicit schemes. It means that $\Delta t \sim \Delta x^2$, which requires a high computational effort for cases when Δx is small.

4.4. Implicit Methods

The discretization of the problem (4.3.1)-(4.3.4) was done with respect to the point U_i^n for the spatial derivative. If instead we use the point U_i^{n+1} then an implicit method is achieved

$$\begin{aligned} \frac{U_i^{n+1} - U_i^n}{\Delta t} &= \frac{U_{i-1}^{n+1} - 2U_i^{n+1} + U_{i+1}^{n+1}}{\Delta x^2} \\ -rU_{i-1}^{n+1} + \left(1+2r\right)U_i^{n+1} - rU_{i+1}^{n+1} &= U_i^n \end{aligned} \tag{4.4.1}$$

It is possible to express the scheme (4.4.1) as a linear system

$$A_i U^{n+1} = U^n$$

where

$$A_i = \begin{bmatrix} (1+2r) & -r & & \\ -r & \ddots & \ddots & \\ & \ddots & & -r \\ & & -r & (1+2r) \end{bmatrix} \in \mathbb{R}^{N\times N}$$

and U^n is defined as in Section 4.3.

Using the von Neumann analysis we can show that the method (4.4.1) is unconditionally stable. Replacing the ansatz (4.2.1) into (4.4.1) we have

$$\begin{aligned} -rc_\varphi^{n+1}\exp\left(j\varphi\left(i-1\right)\Delta s\right) + \left(1+2r\right)c_\varphi^{n+1}\exp\left(j\varphi i\Delta s\right) \\ -rc_\varphi^{n+1}\exp\left(j\varphi\left(i+1\right)\Delta s\right) = c_\varphi^n\exp\left(j\varphi i\Delta s\right), \end{aligned}$$

dividing by $\exp\left(j\varphi i\Delta s\right)$ leads to

$$\begin{aligned} c_\varphi^n &= c_\varphi^{n+1}\left[-r\exp\left(-j\varphi\Delta s\right) + \left(1+2r\right) - r\exp\left(j\varphi\Delta s\right)\right], \\ &= c_\varphi^{n+1}\left[1 + 2r\left(1-\cos\left(\varphi\Delta s\right)\right)\right], \\ &= c_\varphi^{n+1}\left[1 + 4r\sin^2\left(\frac{\varphi\Delta s}{2}\right)\right]. \end{aligned}$$

The growth factor is then

$$G_\varphi = \frac{c_\varphi^{n+1}}{c_\varphi^n} = \frac{1}{1 + 4r \sin^2\left(\frac{\varphi \Delta s}{2}\right)},$$

where it is easy to see that $|G_\varphi| \leq 1$ for all r.

4.5. Crank-Nicolson Methods

The Crank-Nicolson scheme can be seen as an average of the explicit and the implicit method in space and the trapezoidal rule in time. In this sense, the discretization for the PDE (4.3.1)-(4.3.4) is

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} = \frac{1}{2}\left[\frac{U_{i-1}^n - 2U_i^n + U_{i+1}^n}{\Delta x^2}\right] + \frac{1}{2}\left[\frac{U_{i-1}^{n+1} - 2U_i^{n+1} + U_{i+1}^{n+1}}{\Delta x^2}\right]$$

which is order two in time and space. Rearranging the terms, we get

$$-rU_{i-1}^{n+1} + 2\,(1+r)\,U_i^{n+1} - rU_{i+1}^{n+1} = rU_{i-1}^n + 2\,(1-r)\,U_i^n + rU_{i+1}^n$$

or expressed in matrix form

$$A_{cn} U^{n+1} = B_{cn} U^n$$

with

$$A_{cn} = \begin{bmatrix} 2\,(1+r) & -r & & \\ -r & \ddots & \ddots & \\ & \ddots & & -r \\ & & -r & 2\,(1+r) \end{bmatrix} \in \mathbb{R}^{N \times N},$$

$$B_{cn} = \begin{bmatrix} 2\,(1-r) & r & & \\ r & \ddots & \ddots & \\ & \ddots & & r \\ & & r & 2\,(1-r) \end{bmatrix} \in \mathbb{R}^{N \times N}.$$

Let us note that the Crank-Nicolson scheme is just a special case of the more general $\theta-$Method

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} = \theta\left[\frac{U_{i-1}^n - 2U_i^n + U_{i+1}^n}{\Delta x^2}\right] + (1-\theta)\left[\frac{U_{i-1}^{n+1} - 2U_i^{n+1} + U_{i+1}^{n+1}}{\Delta x^2}\right]. \quad (4.5.1)$$

The Crank-Nicolson method is also unconditionally stable.

4.6. Exponentially Fitted Schemes

In this section we consider schemes for the initial-boundary PDE of the form

$$\frac{\partial u}{\partial t} - \varepsilon \Delta u + b \nabla u + cu = f, \tag{4.6.1}$$

where the data are scaled in such a way that $\|f\|_\infty$, $\|b\|_\infty$ and $\|c\|_\infty$ are all $\mathcal{O}(1)$ while $0 < \varepsilon \ll 1$ [GRS07]. Equation (4.6.1) is known as the convection-diffusion equation and is said to be convection-dominated or singularly perturbed because as $\varepsilon \to 0$, classical numerical schemes to obtain u do not converge pointwise to the solution of the related problem when $\varepsilon = 0$ or require an extremely fine mesh.

Let us set $b =$ constant, $c = 0$, $f = 0$ and $u := u(x)$ in equation (4.6.1) such that

$$-\varepsilon \frac{d^2 u}{dx^2} - b \frac{du}{dx} = 0, \tag{4.6.2}$$

with $x \in [0, 1]$, corresponding boundary conditions $u(0) = 0$ and $u(1) = 1$ and exact solution

$$u(x) = \frac{1 - \exp\left(-\frac{bx}{\varepsilon}\right)}{1 - \exp\left(-\frac{b}{\varepsilon}\right)}. \tag{4.6.3}$$

The discretization of equation (4.6.2) is

$$-\varepsilon \frac{U_{i+1} - 2U_i + U_{i-1}}{\Delta x^2} - b \frac{U_{i+1} - U_{i-1}}{2\Delta x} = 0 \tag{4.6.4}$$

with the grid and Δx defined as in Section 4.1. It can be shown [Roo94] that a very stringent condition is required for convergence, namely

$$\Delta x < \frac{2\varepsilon}{b}. \tag{4.6.5}$$

The exact solution for the difference equation (4.6.4) is

$$U_i = \frac{1 - \left(\frac{2\varepsilon - b\Delta x}{2\varepsilon + b\Delta x}\right)^i}{1 - \left(\frac{2\varepsilon - b\Delta x}{2\varepsilon + b\Delta x}\right)^{N+2}},$$

where boundary conditions are fulfilled: $U_0 = 0$ and $U_{N+2} = 1$. It is easily observed that if the condition (4.6.5) is not taken into account, the difference equation will never converge to the exact solution. For instance, if we let $b\Delta x = 2\varepsilon$ then $U_i = 1$ for all i.

If instead of using the second-order approximation for the first derivative, the first-order approximation is used, then the discretization for (4.6.2) is

$$-\varepsilon \frac{U_{i+1} - 2U_i + U_{i-1}}{\Delta x^2} - b \frac{U_{i+1} - U_i}{\Delta x} = 0,$$

with solution

$$U_i = \frac{1 - \left(\frac{\varepsilon}{\varepsilon + b\Delta x}\right)^i}{1 - \left(\frac{\varepsilon}{\varepsilon + b\Delta x}\right)^{N+2}}.$$

Nevertheless, this scheme also shows difficulties approximating the true solution. Let us take $\varepsilon = b\Delta x$ and obtain the value for U_1 which is equivalent to $u(x_0 + \Delta x)$

$$U_1 = \frac{\frac{1}{2}}{1 - \left(\frac{1}{2}\right)^{N+1}} = \frac{1}{2\left(1 - 2^{-N-2}\right)} = \frac{1}{2 - 2^{-1/\Delta x}},$$

whereas the exact solution (4.6.3) is

$$u(x_0 + \Delta x) = \frac{1 - \exp(-1)}{1 - \exp\left(-\frac{1}{\Delta x}\right)}.$$

By letting $\Delta x \to 0$ or, equivalently $N \to \infty$, we encounter further issues:

$$\lim_{\Delta x \to 0} \left(u(x_0 + \Delta x) - U_1\right) = \frac{1}{2} - \exp(-1) \approx 0.13212,$$

i.e. even if Δx is arbitrarily small, the error of the approximation is very big.

The behavior of the numerical approximation of the PDE (4.6.2) is because the solution has a boundary layer at $x = 0$; this is a small region in which the solution changes rapidly if ε is small. This behavior is illustrated in Figure 4.6.1.

In Figure 4.6.2 we plot $u(x, \varepsilon)$ in order to visualize how the boundary layer steepens as $\varepsilon \to 0$, to the point in which the solution is discontinuous at $\varepsilon = 0$. This characteristic of the solution causes major issues when obtaining numerical approximations for the equation (4.6.1) with standard methods.

Il'in proposed in [Il'69] to introduce the so-called *fitting factor* ρ to the difference equation (4.6.4)

$$-\rho\varepsilon \frac{U_{i+1} - 2U_i + U_{i-1}}{\Delta x^2} - b\frac{U_{i+1} - U_{i-1}}{2\Delta x} = 0,$$

and require that the exact solution (4.6.3) also satisfy the difference equation, i.e.

$$\begin{aligned}
\rho\varepsilon \frac{1 - \exp\left(-\frac{b(x+\Delta x)}{\epsilon}\right) - 2\left(1 - \exp\left(-\frac{bx}{\epsilon}\right)\right) + 1 - \exp\left(-\frac{b(x-\Delta x)}{\epsilon}\right)}{\Delta x^2} & \\
+b\frac{1 - \exp\left(-\frac{b(x+\Delta x)}{\epsilon}\right) - 1 + \exp\left(-\frac{b(x-\Delta x)}{\epsilon}\right)}{2\Delta x} &= 0, \\
\rho\varepsilon \frac{-\exp\left(-\frac{b(x+\Delta x)}{\epsilon}\right) + 2\exp\left(-\frac{bx}{\epsilon}\right) - \exp\left(-\frac{b(x-\Delta x)}{\epsilon}\right)}{\Delta x^2} & \\
+b\frac{-\exp\left(-\frac{b(x+\Delta x)}{\epsilon}\right) + \exp\left(-\frac{b(x-\Delta x)}{\epsilon}\right)}{2\Delta x} &= 0;
\end{aligned}$$

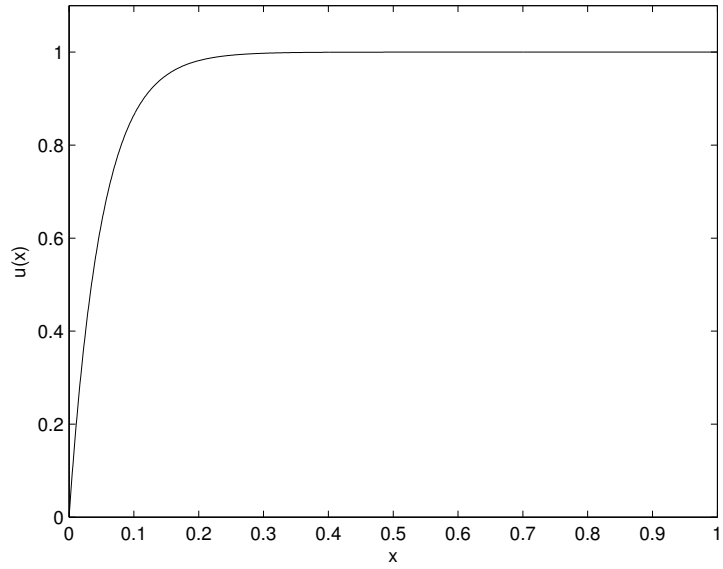


Figure 4.6.1.: Boundary layer at $x = 0$ with $b = 1$ and $\varepsilon = 0.05$.

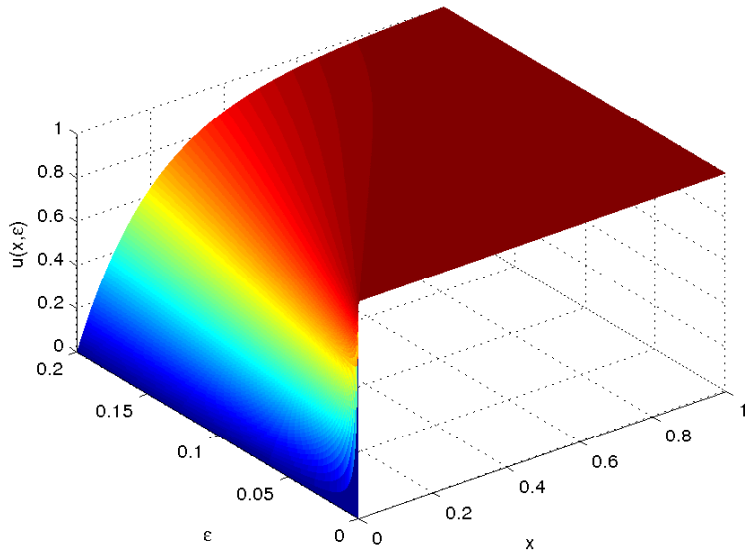


Figure 4.6.2.: Surface for $u(x, \varepsilon)$.

we should note that the term $1-\exp\left(-\frac{b}{\varepsilon}\right)$ was factored out in the last two equations. By dividing by $\exp\left(-\frac{bx}{\varepsilon}\right)$ we get

$$\rho\varepsilon\frac{-\exp\left(-\frac{b\Delta x}{\epsilon}\right)-2-\exp\left(\frac{b\Delta x}{\epsilon}\right)}{\Delta x^2}+b\frac{-\exp\left(-\frac{b\Delta x}{\epsilon}\right)+\exp\left(\frac{b\Delta x}{\epsilon}\right)}{2\Delta x}=0,$$

and defining $\zeta = {}^{b\Delta x}/_{\varepsilon}$

$$\rho\varepsilon\frac{-\exp\left(-\zeta\right)+2-\exp\left(\zeta\right)}{\Delta x^2}+b\frac{\exp\left(\zeta\right)-\exp\left(-\zeta\right)}{2\Delta x}=0;$$

solving for ρ

$$\begin{aligned}\rho\varepsilon\frac{\exp\left(-\zeta\right)-2+\exp\left(\zeta\right)}{\Delta x^2} &= b\frac{\exp\left(\zeta\right)-\exp\left(-\zeta\right)}{2\Delta x}\\ \rho\left(\exp\left(-\zeta\right)-2+\exp\left(\zeta\right)\right) &= \frac{1}{2}\frac{b\Delta x}{\varepsilon}\left(\exp\left(\zeta\right)-\exp\left(-\zeta\right)\right)\\ \rho\cosh\left(\zeta\right) &= \frac{1}{2}\zeta\sinh\left(\zeta\right)+\rho.\end{aligned}$$

With the trigonometric identities

$$\begin{aligned}\sinh\left(2x\right) &= 2\sinh\left(x\right)\cosh\left(x\right),\\ \cosh\left(2x\right) &= 2\sinh^2\left(x\right)+1,\end{aligned}$$

the following expression is obtained

$$\begin{aligned}\rho\left(2\sinh^2\left(\frac{\zeta}{2}\right)+1\right) &= \frac{1}{2}\zeta\left(2\sinh\left(\frac{\zeta}{2}\right)\cosh\left(\frac{\zeta}{2}\right)\right)+\rho\\ \rho\sinh^2\left(\frac{\zeta}{2}\right) &= \frac{1}{2}\zeta\sinh\left(\frac{\zeta}{2}\right)\cosh\left(\frac{\zeta}{2}\right)\\ \rho &= \frac{1}{2}\zeta\coth\left(\frac{1}{2}\zeta\right).\end{aligned}$$

Il'in proved that the fitted schemes are uniformly convergent in the discrete maximum norm, i.e.

$$\max_i\left|u\left(x_i\right)-U_i\right|\leq C\cdot\Delta x,$$

with a constant C that it is independent of ε and Δx.

4.7. Finite Volume Methods

Finite Volume Methods (FVMs) are a special type of FDM derived on the basis of the integral form of the conservation law. One of the advantages of FVMs is that the method is conservative in the sense that it mimics the true solution. Moreover, it is easier to handle problems with irregular geometry with FVMs in comparison to FDMs.

The method defines a volume surrounding each discretization point in the domain of study – these volumes are also called cells – and inside these cells, an approximation of the average value of the unknown is achieved.

Let us consider the discretization of the strong differential form of the general conservation law

$$\frac{\partial u\left(x,t\right)}{\partial t}+\nabla\cdot f\left(u\left(x,t\right)\right)=0, \tag{4.7.1}$$

where $x=\left(x_1,x_2,\ldots,x_d\right)^\intercal$ and $f=\left(f_1,f_2,\ldots,f_d\right)^\intercal$ is the flux of the system, with d as the dimension of the system.

We start by defining an admissible mesh suitable for FVMs on an interval $x\in[a,b]$. A family of equidistant points as $\left(x_i\right)_{i=0,\ldots N+1}$ and a family of midpoints $\left(I_i\right)_{i=1,\ldots,N}$ such that $I_i=\left[x_{i-1/2},x_{i+1/2}\right]$ is defined resulting in a grid

$$x_0=x_{1/2}=a<x_1<x_{3/4}<\cdots<x_{i-1/2}<x_i<x_{i+1/2}<x_N<x_{N+1/2}=x_{N+1}=b, \tag{4.7.2}$$

with Δx defined as in (4.1.1) and $x_{i\pm1/2}=x_i\pm\frac{1}{2}\Delta x$ [EGH00].

Considering (4.7.1) with $d=1$ and integrating it over the rectangle $I_i\times[t,t+\Delta t]$ we have

$$\begin{aligned}\int_t^{t+\Delta t}\int_{I_i}\frac{d}{dt}u\left(x,t\right)dxdt+\int_t^{t+\Delta t}\int_{I_i}\frac{d}{dx}f\left(u\right)dxdt &= 0,\\ \int_{I_i}u\left(x,t+\Delta t\right)dx-\int_{I_i}u\left(x,t\right)dx &= -\int_t^{t+\Delta t}f\left(u\left(x_{i+1/2},t\right)\right)dt\\ &\quad+\int_t^{t+\Delta t}f\left(u\left(x_{i-1/2},t\right)\right)dt.\end{aligned}$$

Dividing the last equation by Δx we obtain

$$\begin{aligned}\frac{1}{\Delta x}\int_{I_i}u\left(x,t+\Delta t\right)dx &= \frac{1}{\Delta x}\int_{I_i}u\left(x,t\right)dx\\ &\quad-\frac{1}{\Delta x}\int_t^{t+\Delta t}f\left(u\left(x_{i+1/2},t\right)\right)dt\\ &\quad+\frac{1}{\Delta x}\int_t^{t+\Delta t}f\left(u\left(x_{i-1/2},t\right)\right)dt\end{aligned}$$

and if we define $\bar{U}_i^n$ as the average value of $u(x,t)$ in the interval I_i at time t_n we obtain

$$\bar{U}_i^{n+1} = \bar{U}_i^{\,n} - \frac{1}{\Delta x}\left[\int_t^{t+\Delta t} f\left(u\left(x_{i+1/2},t\right)\right)dt - \int_t^{t+\Delta t} f\left(u\left(x_{i-1/2},t\right)\right)dt\right]. \quad (4.7.3)$$

Equation (4.7.3) is a relation which provides a mechanism to update the value of cell $\bar{U}_i$ at each time step to obtain the next approximation.

In general, the integral of the flux over time cannot be evaluated analytically, so an approximation to it – also known as numerical flux – is defined as

$$F_{i+1/2}^n \approx \frac{1}{\Delta t}\int_t^{t+\Delta t} f\left(u\left(x_{i+1/2},t\right)\right)dt,$$

and we can now express the fully discrete version of (4.7.3) as

$$\bar{U}_i^{n+1} = \bar{U}_i^n - \frac{\Delta t}{\Delta x}\left[F_{i+1/2}^n - F_{i-1/2}^n\right].$$

A difference method is called *conservative* if it can be written in the form

$$\bar{U}_i^{n+1} = \bar{U}_i^n - \frac{\Delta t}{\Delta x}\left[F\left(\bar{U}_{i-p}^n, \bar{U}_{i-p+1}^n, \ldots, \bar{U}_{j+q}^n\right) - F\left(\bar{U}_{i-p-1}^n, \bar{U}_{i-p}^n, \ldots, \bar{U}_{j+q-1}^n\right)\right] \quad (4.7.4)$$

for some integers $p, q > 0$ [Pul10]. The most important case is for $p = 0,\ q = 1$

$$\bar{U}_i^{n+1} = \bar{U}_i^n - \frac{\Delta t}{\Delta x}\left[F\left(\bar{U}_i^n, \bar{U}_{i+1}^n\right) - F\left(\bar{U}_{i-1}^n, \bar{U}_i^n\right)\right].$$

The term *finite volume method* is used in the scientific literature as synonym of conservative methods.

4.7.1. Discrete Conservation

An important characteristic of conservative schemes is the property of discrete conservation. It can be shown [LeV05] that if

$$\Delta x \sum_{j=J}^{K} U_j^0 = \int_{x_{J-1/2}}^{x_{K+1/2}} u_0(x),$$

where $u_0(x)$ is the initial condition and $J < K$ are arbitrary indices, then it holds

$$\Delta x \sum_{j=J}^{K} U_j^n = \int_{x_{J-1/2}}^{x_{K+1/2}} u(x,t_n)\,dx.$$

Letting $U_k(x,t)$ denote a piecewise function defined by the approximation U_i^n, then we have

$$\int_{x_{J-1/2}}^{x_{K+1/2}} U_k(x,t_n)\,dx = \int_{x_{J-1/2}}^{x_{K+1/2}} u(x,t_n)\,dx,$$

i.e. the integral of the approximation coincides with the integral of the exact solution on the interval $\left[x_{J-1/2}, x_{K+1/2}\right]$.

4.7.2. Finite Volume Methods as Fitted Schemes

Roos [Roo94] proved that fitted schemes can be generated with FVMs. We consider again the equation (4.6.2) with $f \neq 0$. Expressing it in conservative form

$$-\varepsilon\left(\exp\left(-\frac{q}{\varepsilon}\right)u'\right)' = \exp\left(-\frac{q}{\varepsilon}\right)f$$

with $q' = -b$. Integrating over the interval $I_i = \left(x_{i-1/2}, x_{i+1/2}\right)$ yields

$$-\varepsilon\exp\left(-\frac{q}{\varepsilon}\right)\left(u'\left(x_{i+1/2},t\right) - u'\left(x_{i-1/2},t\right)\right) = \int_{I_i}\exp\left(-\frac{q}{\nu}\right)f dx.$$

We assume $b = \text{const.}$ or, equivalently, $q = -b_i x$ on the interval I_i. The derivative u' is replaced by the first-order approximation (4.1.4). The integral on the right-hand side is also approximated numerically to obtain

$$-\varepsilon\exp\left(\frac{b_i x_{i+1/2}}{\varepsilon}\right)\frac{u_{i+1}-u_i}{\Delta x} + \varepsilon\exp\left(\frac{b_i x_{i-1/2}}{\varepsilon}\right)\frac{u_i - u_{i-1}}{\Delta x} = -f_i\frac{\varepsilon}{b_i}\left(\exp\left(\frac{b_i x_{i+1/2}}{\varepsilon}\right) - \exp\left(\frac{b_i x_{i-1/2}}{\varepsilon}\right)\right);$$

dividing by the first term

$$\frac{u_{i+1}-u_i}{\Delta x} - \frac{\exp\left(\frac{b_i x_{i-1/2}}{\varepsilon}\right)}{\exp\left(\frac{b_i x_{i+1/2}}{\varepsilon}\right)}\left(\frac{u_i - u_{i-1}}{\Delta x}\right) = -f_i\frac{\varepsilon}{b_i}\left(-\frac{1}{\varepsilon} + \frac{\exp\left(\frac{b_i x_{i-1/2}}{\varepsilon}\right)}{\exp\left(\frac{b_i x_{i+1/2}}{\varepsilon}\right)}\right),$$

and rearranging the equation we obtain

$$u_{i+1} - u_i - \exp(\gamma_i)(u_i - u_{i-1}) = \frac{f_i}{b_i}\Delta x\left(1 - \exp(\gamma_i)\right),$$

with

$$\gamma_i = -\frac{b_i}{\varepsilon}\left(x_{i+1/2} - x_{i-1/2}\right),$$

which is the Il'in scheme.

4.8. Kurganov-Tadmor Schemes

Kuganov and Tadmor [KT00] introduced a high resolution scheme for nonlinear conservation laws and convection-diffusion equations. The main idea of the scheme is to use more precise information of local propagation speeds at cell boundaries in order to average non-smooth parts of the computed approximation over smaller cells than in the smooth regions. One of the advantages of treating smooth and non-smooth regions separately is that the numerical diffusion introduced by the method is independent of Δt.

We omit the full derivation but instead we only highlight important points of it and state the final fully-discrete and semi-discrete scheme.

We rewrite (4.7.1) as a system of equations with $d = 1$

$$\frac{\partial}{\partial t} u(x,t) + \frac{\partial}{\partial x} f(u(x,t)) = 0, \tag{4.8.1}$$

or the related convection-diffusion equation

$$\frac{\partial}{\partial t} u(x,t) + \frac{\partial}{\partial x} f(u(x,t)) = \frac{\partial}{\partial x} Q(u(x,t), u_x(x,t)), \tag{4.8.2}$$

with $u(x,t) = (u_1(x,t), \ldots, u_K(x,t))$ and $u_x(x,t)$ denoting the derivative of $u(x,t)$ with respect to x.

Again, an admissible mesh of site $N+1$ is defined as in (4.7.2) with a family of equidistant points x_i and a family of midpoints $I_i = \left[x_{i-1/2}, x_{i+1/2}\right]$. The step-sizes Δx and Δt have the usual definition (4.1.1) and (4.1.2) respectively.

It is assumed that a computed piecewise, linear approximation

$$\tilde{u}(x,t_n) = \sum_i \left(U_i^n - (U_x)_i^n (x - x_i)\right) \mathbf{1}_{I_i}$$

at time level t_n is already available based on cell averages U_i^n – for clarity, in this section we omit the bar notation used in Section 4.7 to denote cell averages over the interval I_i – and the approximation to the derivative $(U_x)_i^n$. The upper bound of the local speed of propagation at the boundary of the cell $x_{i+1/2}$ for the nonlinear or linearly degenerate case is given by

$$a_{i+1/2}^n = \max\left[\rho\left(\frac{\partial}{\partial u} f\left(U_{i+1/2}^+\right)\right), \rho\left(\frac{\partial}{\partial u} f\left(U_{i+1/2}^-\right)\right)\right], \tag{4.8.3}$$

where

$$U_{i+1/2}^+ = U_{i+1}^n - \frac{1}{2}\Delta x (U_x)_{i+1}^n, \tag{4.8.4}$$

$$U_{i+1/2}^- = U_i^n + \frac{1}{2}\Delta x (U_x)_i^n, \tag{4.8.5}$$

are the corresponding left and right intermediate values of $\tilde{u}(x, t_n)$ at $x_{i+1/2}$ and $\rho(A)$ here denotes the spectral radius of A.

Instead of averaging over the control volumes $I_i \times [t_n, t_n + \Delta t]$, this scheme performs the integration over variable control volumes $\left[x_{i+1/2,l}, x_{i+1/2,r}\right] \times [t_n, t_n + \Delta t]$ where

$$\begin{aligned} x_{i+1/2,l} &= x_{i+1/2} - a^n_{i+1/2}\Delta t, \\ x_{i+1/2,r} &= x_{i+1/2} + a^n_{i+1/2}\Delta t. \end{aligned}$$

Due to the finite speed of propagation, the new interval differentiates between smooth and non-smooth regions providing the non-smooth parts with a narrower control volume of spatial width $2a^n_{i+1/2}\Delta t$.

Defining $\mathcal{I}_i = \left[x_{i+1/2,l}, x_{i+1/2,r}\right]$ and $\Delta x_{i+1/2} = x_{i+1/2,r} - x_{i+1/2,l} = 2a^n_{i+1/2}\Delta t$, which denotes the width of the Riemann fan originating at $x_{i+1/2}$, and proceeding in a similar fashion as in Section 4.7 to obtain the cell averages at $t_n + \Delta t$ we can express (4.8.1) as

$$\begin{aligned} \frac{1}{\Delta x_{i+1/2}} \int_{\mathcal{I}_i} u(x, t_{n+1})\, dx &= \frac{1}{\Delta x_{i+1/2}} \int_{\mathcal{I}_i} \tilde{u}(x, t_n)\, dx \\ &\quad - \frac{1}{\Delta x_{i+1/2}} \int_{t_n}^{t_n+\Delta t} f\left(u\left(x_{i+1/2,r}, t\right)\right) \\ &\quad + \frac{1}{\Delta x_{i+1/2}} \int_{t_n}^{t_n+\Delta t} f\left(u\left(x_{i+1/2,l}, t\right)\right) dt \end{aligned} \tag{4.8.6}$$

and similarly for the point x_i over the interval $\mathcal{I}^2_i = \left[x_{i+1/2,l}, x_{i+1/2,r}\right]$ with $\Delta x_i = x_{i+1/2,l} - x_{i+1/2,r} = \Delta x - \Delta t\left(a^n_{i-1/2} + a^n_{i+1/2}\right)$

$$\begin{aligned} \frac{1}{\Delta x_i} \int_{\mathcal{I}^2_i} u(x, t_{n+1})\, dx &= \frac{1}{\Delta x_i} \int_{\mathcal{I}^2_i} \tilde{u}(x, t_n)\, dx \\ &\quad - \frac{1}{\Delta x_i} \int_{t_n}^{t_n+\Delta t} f\left(u\left(x_{i+1/2,l}, t\right)\right) \\ &\quad + \frac{1}{\Delta x_i} \int_{t_n}^{t_n+\Delta t} f\left(u\left(x_{i+1/2,r}, t\right)\right) dt. \end{aligned} \tag{4.8.7}$$

To avoid confusion, it is important to note that in the second term on the right hand side of (4.8.6) and (4.8.7), the flux is evaluated with the unknown function $u(x, t)$ whereas the first term is obtained via the known piecewise solution $\tilde{u}(x, t)$.

Equations (4.8.6) and (4.8.7) lead to the cell averages over the nonuniform grid

$\left[x_{i+1/2,l}, x_{i+1/2,r}\right]$

$$\begin{aligned} w_{i+1/2}^{n+1} &= \frac{U_i^n + U_{i+1}^n}{2} + \frac{\Delta x - a_{i+1/2}^n \Delta t}{4}\left[(U_x)_i^n - (U_x)_{i+1}^n\right] \\ &\quad - \frac{1}{2a_{i+1/2}^n}\left[f\left(U_{i+1/2,r}^{n+1/2}\right) - f\left(U_{i+1/2,l}^{n+1/2}\right)\right], \\ w_i^{n+1} &= U_i^n + \frac{\Delta t}{2}\left(a_{i-1/2}^n - a_{i+1/2}^n\right)(U_x)_i^n \\ &\quad - \frac{\lambda}{1 - \lambda\left(a_{i-1/2}^n + a_{i+1/2}^n\right)}\left[f\left(U_{i+1/2,l}^{n+1/2}\right) - f\left(U_{i+1/2,r}^{n+1/2}\right)\right], \end{aligned}$$

where $\lambda = \Delta t/\Delta x$ and the midpoints are obtained via Taylor expansion.

Finally, the nonuniform averages are projected back to the uniform grid which results in the fully discrete, second-order scheme

$$\begin{aligned} U_i^{n+1} &= \lambda a_{i-1/2}^n w_{i-1/2}^{n+1} + \left[1 - \lambda\left(a_{i-1/2}^n + a_{i+1/2}^n\right)\right] w_i^{n+1} + \lambda a_{i+1/2}^n w_{i+1/2}^{n+1} \\ &\quad + \frac{\Delta x}{2}\left[\left(\lambda a_{i-1/2}^n\right)^2 (U_x)_{i-1/2}^{n+1} - \left(\lambda a_{i+1/2}^n\right)^2 (U_x)_{i+1/2}^{n+1}\right]. \end{aligned}$$

The semi-discrete scheme is obtained by letting $\Delta t \to 0$ in the expressions for w_i^{n+1}, $w_{i+1/2}^{n+1}$ and U_i^{n+1} – c.f. [KT00] for more details. The scheme reads

$$\frac{d}{dt}U_i(t) = -\frac{1}{\Delta x}\left[H_{i+1/2}(t) - H_{i-1/2}(t)\right], \tag{4.8.8}$$

with the numerical flux given by

$$H_{i+1/2}(t) = \frac{1}{2}\left[f\left(U_{i+1/2}^+(t)\right) + f\left(U_{i+1/2}^-(t)\right)\right] - \frac{a_{i+1/2}(t)}{2}\left[U_{i+1/2}^+(t) - U_{i+1/2}^-(t)\right] \tag{4.8.9}$$

and the values $U_{i+1/2}^{\pm}(t)$ given by

$$\begin{aligned} U_{i+1/2}^+(t) &= U_{i+1}(t) - \frac{1}{2}\Delta x\,(U_x)_{i+1}(t), \\ U_{i+1/2}^-(t) &= U_i(t) + \frac{1}{2}\Delta x\,(U_x)_i(t), \end{aligned}$$

which are the semi-discrete analogous of (4.8.4) and (4.8.5) respectively. For completeness, we state also the semi-discrete analogue of (4.8.3)

$$a_{i+1/2}(t) = \max\left[\rho\left(\frac{\partial}{\partial u}f\left(U_{i+1/2}^+(t)\right)\right), \rho\left(\frac{\partial}{\partial u}f\left(U_{i+1/2}^-(t)\right)\right)\right]. \tag{4.8.10}$$

We can verify that (4.8.8) is consistent with the definition of the conservative method (4.7.4), i.e. $H_{i+1/2}(t) \equiv H(U_{i-1}(t), U_i(t), U_{i+1}(t), U_{i+2}(t))$.

The numerical viscosity, or artificial diffusion, introduced by the method is $\mathcal{O}(\Delta x^3)$ whereas for other schemes like Lax-Friedrichs it is $\mathcal{O}(\Delta x^2/\Delta t)$.

It is possible to extend the scheme (4.8.8) to convection-diffusion equations by including a reasonable numerical approximation for the dissipative flux denoted by $Q(u(x,t), u_x(x,t))$. The scheme reads

$$\frac{d}{dt}U_i(t) = -\frac{1}{\Delta x}\left[H_{i+1/2}(t) - H_{i-1/2}(t)\right] + \frac{1}{\Delta x}\left[P_{i+1/2}(t) - P_{i-1/2}(t)\right],$$

with

$$P_{i+1/2}(t) = \frac{1}{2}\left[Q\left(U_i(t), \frac{U_{i+1}(t) - U_i(t)}{\Delta x}\right) + Q\left(U_{i+1}(t), \frac{U_{i+1}(t) - U_i(t)}{\Delta x}\right)\right].$$

Finally, we would like to mention that it is a well known fact that ODEs obtained as the result of applying semi-discretization methods are always stiff. Moreover, they become arbitrarily stiff as $\Delta x \to 0$ [GRS07]. Hence, appropriate methods for stiff ODEs are needed.

5. The Black-Scholes Equation and Finite Difference Methods

We can transform the Black-Scholes equation (3.0.7) into the heat equation by letting $s = K\exp(x)$ – c.f. [Sey09] – when the quantities r, σ, and d are constant. Such transformation is possible because the variable coefficients s^j match the order of the derivative with respect to s:

$$s^j \frac{\partial^j v(s,t)}{\partial s^j}, \quad \text{for } j = 0, 1, 2.$$

Linear differential equations with such terms are known as Euler's differential equations. Nonetheless the advantage that represent having transformed the Black-Scholes into the heat equation – in terms of numerical methods, the heat equation is well studied, c.f. [GRS07] – it is only useful for plain-vanilla European options with constant coefficients. Hence, numerical methods for the equation (3.0.7) without transformation are needed.

It is useful to define the time as $t^* = T - t$ and modify Black-Scholes (3.0.7) accordingly

$$-\frac{\partial v(s,t^*)}{\partial t^*} + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 v(s,t^*)}{\partial s^2} + (r-d)\, s \frac{\partial v(s,t^*)}{\partial s} - r v(s,t^*) = 0, \qquad (5.0.1)$$

where the initial condition is the payoff (3.0.8) for a call and (3.0.9) for a put.

During this section we simply denote t^* as t and work with equation (5.0.1) instead of (3.0.7). With this definition, we are interested in the price at time $t = T$.

Black-Scholes is defined on an infinite interval for s which is impossible to represent on a computer. Instead, a large enough, finite interval is used to obtain an approximation to the solution and the boundary conditions are defined accordingly on this interval. For Example, for an European call on an interval $s \in [s_{min}, s_{max}]$ we have

$$\begin{aligned} v(s_{min}, t) &= 0, \\ v(s_{max}, t) &= s_{max}\exp(-dt) - K\exp(-rt). \end{aligned}$$

European options represent our benchmark problem because there exists an exact solution for equation (5.0.1) and therefore we can compare the exact solution

versus the numerical approximation in order to highlight strengths and weaknesses of the method being used.

5.1. An implicit Method

Let us discretize the Black-Scholes equation with FDM as introduced in Chapter 4. A mesh is defined for the price of the stock $s \in [s_{min}, s_{max}]$ with $N+2$ points s_i for $i = 0, 1, \ldots, N+1$

$$s_{min} = s_0 < s_1 < \cdots < s_N < s_{N+1} = s_{max},$$

with $\Delta s = {}^{s_{max}-s_{min}}/_{N+1}$. The time $t \in [0, T]$ is discretized in M points t_j for $j = 1, 2, \ldots, M$

$$0 = t_1 < t_2 < \cdots < t_M = T,$$

with $\Delta t = {}^{T}/_{M-1}$.

Defining $V_i^n \equiv v(s_i, t_n)$ and substituting each partial derivative in (3.0.7) by its corresponding numerical derivative at $t = t_{n+1}$ we get

$$-\frac{V_i^{n+1} - V_i^n}{\Delta t} + \frac{1}{2}\sigma^2 s_i^2 \frac{V_{i+1}^{n+1} - 2V_i^{n+1} + V_{i-1}^{n+1}}{\Delta s^2} + (r-d)\, s_i \frac{V_{i+1}^{n+1} - V_{i-1}^{n+1}}{2\Delta s} - rV_i^{n+1} = 0,$$

where we use the second-order approximation for the first derivative in order to have a method of order two in space and order one in time. Rearranging the equation leads to

$$\alpha_i V_{i-1}^{n+1} + \beta_i V_i^{n+1} + \gamma_i V_{i+1}^{n+1} = V_i^n, \tag{5.1.1}$$

with

$$\begin{aligned}
\alpha_i &= -\frac{1}{2}\left(\frac{\sigma^2 s_i^2 \Delta t}{\Delta s^2} - \frac{(r-d)\, s_i \Delta t}{\Delta s}\right), \\
\beta_i &= 1 + r\Delta t + \frac{\sigma^2 s_i^2 \Delta t}{\Delta s^2}, \\
\gamma_i &= -\frac{1}{2}\left(\frac{\sigma^2 s_i^2 \Delta t}{\Delta s^2} + \frac{(r-d)\, s_i \Delta t}{\Delta s}\right),
\end{aligned}$$

for $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, M$.

By expressing (5.1.1) in matrix form we have

$$\begin{bmatrix}
\beta_1 & \gamma_1 & & & \\
\alpha_2 & \beta_2 & \gamma_2 & & \\
& \ddots & \ddots & \ddots & \\
& & \alpha_{N-1} & \beta_{N-1} & \gamma_{N-1} \\
& & & \alpha_N & \beta_N
\end{bmatrix}
\begin{bmatrix} V_1^{n+1} \\ V_2^{n+1} \\ \vdots \\ V_{N-1}^{n+1} \\ V_N^{n+1} \end{bmatrix}
=
\begin{bmatrix} V_1^n - \alpha_1 V_0^n \\ V_2^n \\ \vdots \\ V_{N-1}^n \\ V_N^n - \gamma_N V_{N+1}^n \end{bmatrix} \tag{5.1.2}$$

with boundary conditions included. The unknown vector V^{n+1} is hence obtained by solving a tridiagonal system of equations with computational effort $\mathcal{O}(N)$ at each time step.

5.1.1. von Neumann Stability Analysis

We define the Fourier modes of the approximation as

$$V_i^n = \sum_{\varphi} c_\varphi^n \exp\left(j\varphi i \Delta s\right), \tag{5.1.3}$$

where $j^2 = -1$ is the imaginary unit and φ represents the wave number. Replacing (5.1.3) into (5.1.1) we have

$$\begin{aligned}\alpha_i \sum_{\varphi} c_\varphi^{n+1} \exp\left(j\varphi\left(i-1\right)\Delta s\right) + \beta_i \sum_{\varphi} c_\varphi^{n+1} \exp\left(j\varphi i \Delta s\right) \\ + \gamma_i \sum_{\varphi} c_\varphi^{n+1} \exp\left(j\varphi\left(i+1\right)\Delta s\right) = \\ \sum_{\varphi} c_\varphi^{n} \exp\left(j\varphi i \Delta s\right).\end{aligned}$$

Due to the linearity of the PDE, it is possible to use only one Fourier mode. Dividing the last equation by $\exp\left(j\varphi i \Delta s\right)$ we achieve the expression

$$c_\varphi^{n+1}\left[\alpha_i \exp\left(-j\varphi\Delta s\right) + \beta_i + \gamma_i \exp\left(j\varphi\Delta s\right)\right] = c_\varphi^n.$$

We are interested in the growth factor G_φ of the Fourier mode. If $|G_\varphi| \leq 1$ then the scheme is stable, otherwise the scheme is unstable and therefore not useful because initial errors are amplified. The growth factor is then defined as

$$\begin{aligned} G_\varphi &= \frac{c_\varphi^{n+1}}{c_\varphi^n} \\ &= \frac{1}{\alpha_i \exp\left(-j\varphi\Delta s\right) + \beta_i + \gamma_i \exp\left(j\varphi\Delta s\right)} \\ &= \frac{1}{\beta_i + \left(\alpha_i + \gamma_i\right)\cos\left(\varphi\Delta s\right) - j\left(\alpha_i - \gamma_i\right)\sin\left(\varphi\Delta s\right)}, \end{aligned}$$

and replacing the values for α_i, β_i and γ_i we get

$$G_\varphi = \frac{1}{1 + r\Delta t + \frac{\sigma^2 S_i^2 \Delta t}{\Delta S^2}\left(1 - \cos\left(\varphi\Delta s\right)\right) + j\frac{(r-d)s_i\Delta t}{\Delta s}\sin\left(\varphi\Delta s\right)}$$

or

$$|G_\varphi|^2 = \frac{1}{\left(1 + r\Delta t + \frac{\sigma^2 s_i^2 \Delta t}{\Delta s^2}\left(1 - \cos\left(\varphi\Delta s\right)\right)\right)^2 + \frac{(r-d)^2 s_i^2 \Delta t^2}{\Delta s^2}\sin^2\left(\varphi\Delta s\right)}.$$

From the expression for $|G_\varphi|^2$ it is easy to see that $|G_\varphi| \leq 1$ without imposing restrictions on Δt, Δs or any other parameter of the PDE because the denominator will be always greater than one. For instance:

- when $\cos(\varphi \Delta s) = -1$, then $\sin^2(\varphi \Delta s) = 0$ and the denominator is $\left(1 + r\Delta t + 2\frac{\sigma^2 s^2 \Delta t}{\Delta s^2}\right)^2 > 1$.
- when $\cos(\varphi \Delta s) = 1$, then $\sin^2(\varphi \Delta s) = 0$ and the denominator is $(1 + r\Delta t)^2 > 1$.
- when $\cos(\varphi \Delta s) = 0$, then $\sin^2(\varphi \Delta s) = 1$ and the denominator is $\left(1 + r\Delta t + \frac{\sigma^2 s^2 \Delta t}{\Delta s^2}\right)^2 + \frac{(r-d)^2 s^2 \Delta t^2}{\Delta s^2} > 1$.

The von Neumann analysis tell us that we should not expect stability problems related to the step sizes.

5.1.2. Numerical Simulation

Implementing the scheme (5.1.2) is straightforward: we only need to solve a tridiagonal system of equations for each time step. In environments like Matlab or Octave it is easy to declare sparse matrices and solve corresponding systems efficiently. When working with programming languages like C/C++, LAPACK library provides efficient algorithms to solve sparse linear systems.

As an example we take an hypothetical European call with $r = 0.05$, $d = 0$, $\sigma = 0.01$, $K = 13$, $T = 1$, $s_{min} = 10$, $s_{max} = 15$, $N = 50$, and $M = 100$ – quantities are stated without units. The result is shown in Figure 5.1.1a.

From the visual comparison between the exact solution and the numerical solution we can see that the implicit scheme delivers good results. Moreover, we can compute the error using the maximum norm

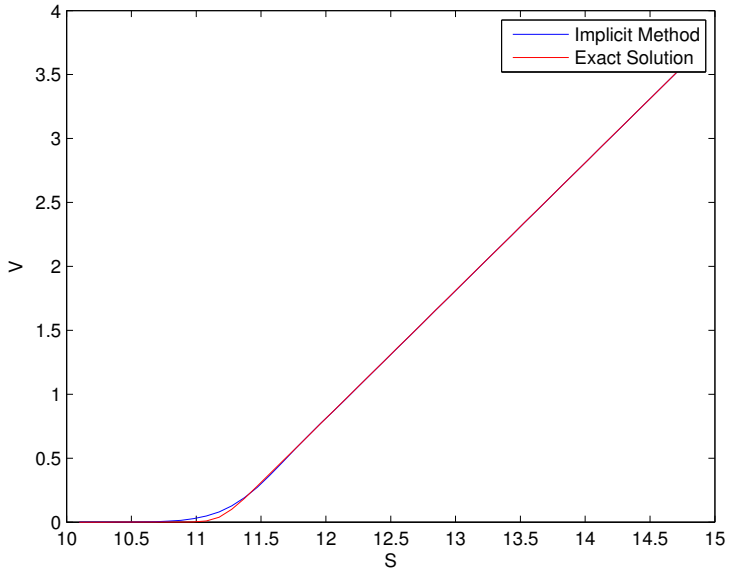
$$\left\| v(s_i, T) - V_i^M \right\|_\infty \equiv \max_{\forall i} \left| v(s_i, 0) - V_i^M \right|,$$

where $v(s,t)$ represents the exact solution and V_i^n an approximation for $v(s_i, t_n)$. For different step sizes, the Table 5.1 shows, as expected, that the error is $\mathcal{O}(\Delta s^2)$. Furthermore, the estimate is independent of the norm used to calculate the error.

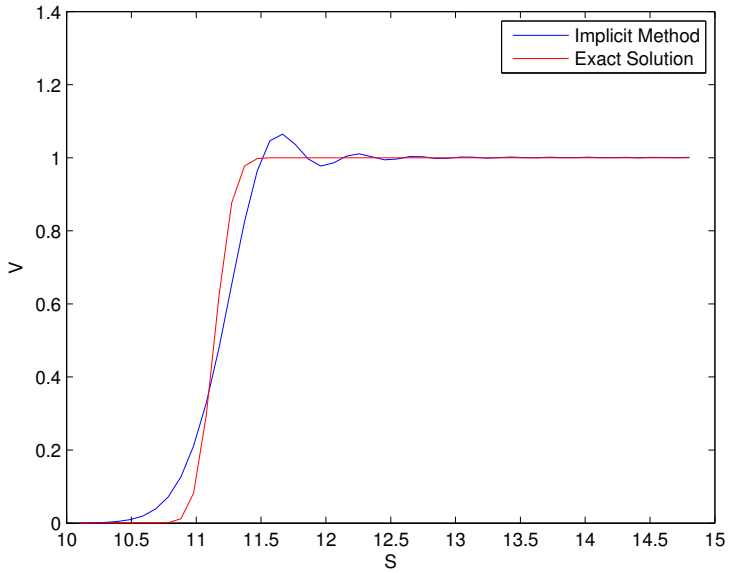
In addition to the price of the derivative, the Greeks of the price are often needed. For example, the delta of the derivative is defined as

$$\Delta = \frac{\partial v(s,t)}{\partial s},$$

which is easily obtained by applying a numerical derivative operator to V_i^n, shown in Figure 5.1.1b.



(a) Comparison between the exact solution and the numerical solution of a plain-vanilla European call with the Implicit Method.



(b) Spurious oscillations appear when obtaining the delta of the price of the derivative, defined as $\frac{\partial V}{\partial S}$.

Figure 5.1.1.: Simulation with standard methods for Black-Scholes

$N \times M$	20×200	40×200	60×200	80×200	100×200
$\left\|v\left(s_i, T\right) - V_i^M\right\|_\infty$	0.031733	0.015810	0.008507	0.004653	0.002578
Δs^2	0.056689	0.014872	0.006719	0.003810	0.002451

Table 5.1.: Error of the price for the Implicit Method with different discretization steps in the stock-price space.

Although the approximation for V_i^n is second order in space – which is shown also empirically in Table 5.1 – oscillations are observed between $s \in [12, 13]$ for the first derivative of the option price approximation V_i^n. These oscillation are *financially unrealistic* or spurious and are introduced by the numerical method.

The *Péclet number* – defined as the ratio of convection by diffusion – is a useful tool to anticipate issues with numerical simulations. For Black-Scholes we have

$$P = \Delta S \frac{rS}{\frac{1}{2}\sigma^2 S^2} = \frac{2r}{\sigma^2}\frac{\Delta S}{S} = \mathcal{O}\left(\frac{r}{\sigma^2}\right),$$

called the *mesh Péclet number*. Empirical evidence indicates that the higher the Péclet number, the higher the danger that the numerical solution exhibits oscillations [Sey09].

5.2. Exponentially Fitted Schemes

Standard FDM represent unstable solutions for the convection-diffusion equation, henceforth, Black-Scholes. In this section we use the technique, presented in Section 4.6, proposed by Il'in [Il'69] which was later applied to pricing problem with the Black-Scholes equation by Duffy [Duf06].

The same space and time discretization in Section 5.1 is used in this section.

The corresponding implicit exponential fitted scheme for Black-Scholes equation (5.0.1) is

$$-\frac{V_i^{n+1} - V_i^n}{\Delta t} + \rho\frac{1}{2}\sigma^2 s_i^2 \frac{V_{i+1}^{n+1} - 2V_i^{n+1} + V_{i-1}^{n+1}}{\Delta s^2} + (r-d)\, s_i \frac{V_{i+1}^{n+1} - V_{i-1}^{n+1}}{2\Delta s} - rV_i^{n+1} = 0,$$

where

$$\rho = \frac{1}{2}\zeta \coth\left(\frac{1}{2}\zeta\right),$$

and

$$\zeta = \frac{(r-d)\, s_i \Delta s}{1/2\sigma^2 s_i^2} = \frac{2\,(r-d)\,\Delta s}{\sigma^2 s_i}.$$

Rearranging and substituting the terms like in Section 5.1 we achieve an scheme

$$\alpha_i V_{i-1}^{n+1} + \beta_i V_i^{n+1} + \gamma_i V_{i+1}^{n+1} = V_i^n, \tag{5.2.1}$$

with

$$\begin{aligned}
\alpha_i &= -\frac{1}{2}\left(\rho\frac{\sigma^2 s_i^2 \Delta t}{\Delta s^2} - \frac{(r-d)\, s_i \Delta t}{\Delta s}\right), \\
\beta_i &= 1 + r\Delta t + \rho\frac{\sigma^2 s_i^2 \Delta t}{\Delta s^2}, \\
\gamma_i &= -\frac{1}{2}\left(\rho\frac{\sigma^2 s_i^2 \Delta t}{\Delta s^2} + \frac{(r-d)\, s_i \Delta t}{\Delta s}\right),
\end{aligned}$$

for $i = 1, 2 \ldots, N$ and $j = 1, 2, \ldots M$.

We can express (5.2.1) in matrix form

$$\begin{bmatrix}
\beta_1 & \gamma_1 & & & \\
\alpha_2 & \beta_2 & \gamma_2 & & \\
& \ddots & \ddots & \ddots & \\
& & \alpha_{N-1} & \beta_{N-1} & \gamma_{N-1} \\
& & & \alpha_N & \beta_N
\end{bmatrix}
\begin{bmatrix} V_1^{n+1} \\ V_2^{n+1} \\ \vdots \\ V_{N-1}^{n+1} \\ V_N^{n+1} \end{bmatrix}
=
\begin{bmatrix} V_1^n - \alpha_1 V_0^n \\ V_2^n \\ \vdots \\ V_{N-1}^n \\ V_N^n - \gamma_N V_{N+1}^n \end{bmatrix} \tag{5.2.2}$$

with boundary conditions included. As in Section 5.1, the unknown vector V^{n+1} is hence obtained by solving a tridiagonal system of equations with computational effort $\mathcal{O}(N)$ at each time step.

5.2.1. Numerical Simulation

The numerical method expressed by equation (5.2.2) is solved in a similar way as the method given by Equation (5.1.2). The Table 5.2 shows the error for different discretization grids. It is evident that the Exponential Fitting Method is $\mathcal{O}(\Delta s)$, as stated in [Il'69], whereas the Implicit Finite Difference Method is $\mathcal{O}(\Delta s^2)$. The introduction of the fitting factor ρ degrades the order of the method. Nevertheless, the problems with spurious oscillations are solved.

The simulation for the price of the derivative with $r = 0.05$, $d = 0$, $\sigma = 0.01$, $K = 13$, $T = 1$, $s_{min} = 10$, $s_{max} = 15$, $N = 50$, and $M = 100$ is shown in Figure 5.2.1a. From the image we immediately spot that the reduction in the order of the method is reflected in an area close to the strike price. If the price of the derivative close to the strike price is needed at $t = 0$ then we must increase N to have a better approximation. A simulation with $N = 2000$, which reduces the error $\left\| v(s_i, T) - V_i^M \right\|_\infty$ to 0.00307, is easily achieved with Matlab running on a laptop computer with average hardware.

$N \times M$	20×200	40×200	60×200	80×200	100×200
$\left\|v\left(s_i, T\right) - V_i^M\right\|_\infty$	0.10371	0.06014	0.04111	0.03051	0.02318
Δs	0.23810	0.12195	0.08197	0.06173	0.04950

Table 5.2.: Error of the price for the Exponential Fitting Method with different discretization steps in the stock-price space.

In Figure 5.2.1b we have plotted the delta the price of the derivative defined as $\partial V/\partial S$. Spurious oscillations does not exist in this case – the parameters are the same as those for the simulation used for the Implicit Method. On the other hand, the artificial diffusion introduced by the method is evident now.

5.3. Wang's Finite Volume Method

Wang [Wan04] presented a FVM for Black-Scholes with non-constant coefficients. This section highlights some parts of the derivation and the final scheme.

We would like to represent equation (3.0.7) in conservative form with homogeneous Dirichlet boundary conditions. For this propose, we add $f\left(S,t\right) = -\mathcal{L}V_0$ to both sides of Black-Scholes, where $\mathcal{L}$ is the differential operator in (3.0.7) and

$$V_0 = g_1\left(t\right) + \frac{1}{S_T}\left[g_1\left(t\right) - g_2\left(t\right)\right] S.$$

Introducing the variable $u = V - V_0$, it is possible to express the Black-Scholes PDE in the self-adjoint form:

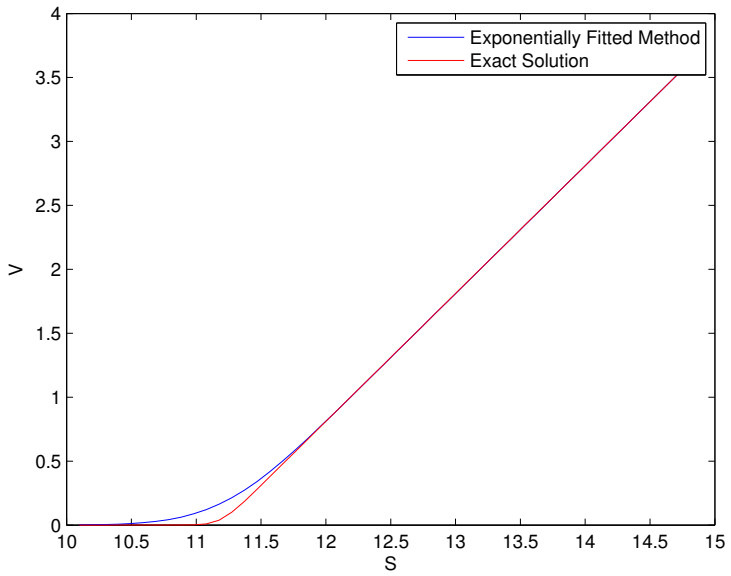
$$\frac{\partial u}{\partial t} - \frac{\partial}{\partial s}\left[a\left(t\right) s^2 \frac{\partial u}{\partial s} + b\left(s,t\right) su\right] + c\left(s,t\right) u = f\left(s,t\right), \tag{5.3.1}$$

with

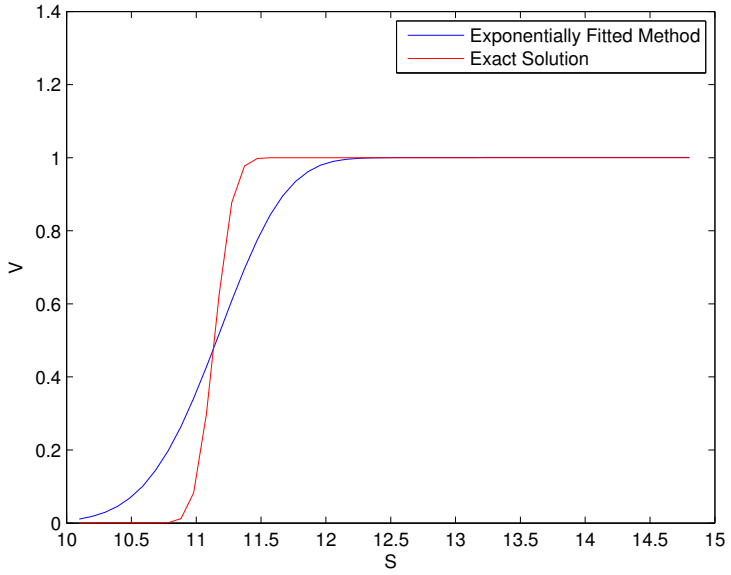
$$\begin{aligned} a\left(t\right) &= \frac{1}{2}\sigma^2\left(t\right), \\ b\left(s,t\right) &= r\left(t\right) - d\left(s,t\right) - \sigma^2\left(t\right), \\ c\left(s,t\right) &= r\left(t\right) - b\left(s,t\right) - s\frac{\partial d}{\partial s}. \end{aligned}$$

To start, an admissible mesh analogous to (4.7.2) is defined for the interval $s \in [0, s_{max}]$ with $N+2$ grid points, given by a family $I_i = \left(s_{i-1/2}, s_{i+1/2}\right)$ and a family $\left(s_i\right)_{i=0,\ldots,N+1}$

$$\begin{aligned} s_0 = s_{1/2} = 0 < s_1 < s_{3/4} < \cdots < s_{i-1/2} < s_i \\ < s_{i+1/2} < \cdots < s_N < s_{N+1/2} = s_{N+1} = s_{max}, \end{aligned} \tag{5.3.2}$$



(a) Comparison between the exact solution and the numerical solution of a plain-vanilla European call with Exponential Fitting method.



(b) No spurious oscillation observed.

Figure 5.2.1.: Exponential Fitting method applied to Black-Scholes.

with $\Delta s_i = s_{i+1} - s_i$ and $\Delta s = \max_{\forall i} (\Delta s_i)$.

Integrating the Black-Scholes equation in conservative form (5.3.1) over the cell $I_i = \left(s_{i-1/2}, s_{i+1/2}\right)$ leads to

$$\int_{I_i} \frac{\partial u}{\partial t} ds - \left[s \left(as \frac{\partial u}{\partial s} + bu \right) \right]_{s_{i-1/2}}^{s_{i+1/2}} + \int_{I_i} cuds = \int_{I_i} f ds, \tag{5.3.3}$$

Using the mid-point rule as a numerical approximation to an integral, the integrals in (5.3.3) can be replaced by their discrete equivalent

$$\begin{aligned} \int_{K_i} \frac{\partial u}{\partial t} dS &= \frac{\partial U_i}{\partial t} \cdot \left(S_{i+1/2} - S_{i-1/2} \right), \\ \int_{K_i} cudS &= c_i U_i \cdot \left(S_{i+1/2} - S_{i-1/2} \right), \\ \int_{K_i} f dS &= f_i \cdot \left(S_{i+1/2} - S_{i-1/2} \right), \end{aligned}$$

and with $l_i := S_{i+1/2} - S_{i-1/2}$, we can rewrite (5.3.3) as

$$\frac{\partial U_i}{\partial t} l_i - \left[s_{i+1/2} \rho \left(u \left(s_{i+1/2}, t \right) \right) - s_{i-1/2} \rho \left(u \left(s_{i-1/2}, t \right) \right) \right] + c_i U_i l_i = f_i l_i, \tag{5.3.4}$$

where discrete unknowns are denoted by $U_i := u(s_i, t)$, $c_i := c(s_i, t)$ and $f_i := f(s_i, t)$ for $i = 1, \ldots, N$. The flux $\rho(u(s,t))$ therefore is defined as

$$\rho(u(s,t)) := a(t) s \frac{\partial u(s,t)}{\partial s} + b(s,t) u(s,t).$$

Due to the degeneracy of $\rho(u(s,t))$ at $s = 0$, the flux must be treated separately for the degenerate and non-degenerate case.

First, an approximation for $\rho(u(s,t))$ evaluated at $S_{i+1/2}$ with $i \geq 1$ is obtained. Let us consider the following two-point boundary value problem:

$$\begin{aligned} \frac{d}{ds} \left[a(t) s \frac{dv(s)}{ds} + b_{i+1/2}(t) v(s) \right] &= 0, && (5.3.5) \\ v(s_i) &= U_i, && (5.3.6) \\ v(s_{i+1}) &= U_{i+1}, && (5.3.7) \end{aligned}$$

for $s \in (s_i, s_{i+1})$. Integrating it over the interval yields a first-order linear equation

$$\rho_i(v) := a(t) s \frac{dv(s)}{ds} + b_{i+1/2}(s,t) v(s) = c_1, \tag{5.3.8}$$

and its analytic solution is

$$v = \frac{c_1}{b_{i+1/2}(s,t)} + c_2 s^{-\alpha_i}, \tag{5.3.9}$$

where

$$\alpha_i \equiv \frac{b_{i+1/2}(s,t)}{a(t)};$$

applying boundary conditions and solving the corresponding linear system, the approximation for the flux is

$$\rho_i(u(s,t)) = b_{i+1/2}(s,t)\,\frac{s_{i+1}^{\alpha_i}\cdot U_{i+1} - s_i^{\alpha_i}\cdot U_i}{s_{i+1}^{\alpha_i} - s_i^{\alpha_i}}, \quad \text{for } i = 1,\dots N. \tag{5.3.10}$$

Now, an approximation for $\rho(u)$ at $i = 0$ is obtained. For this purpose, it is needed to reconsider (5.3.5) with an extra degree of freedom

$$\begin{aligned} \frac{d}{ds}\left[a(t)\,s\frac{dv(s)}{ds} + b_{1/2}(s,t)\,v(s)\right] &= c, && (5.3.11)\\ v(0) &= U_0, && (5.3.12)\\ v(s_1) &= U_1, && (5.3.13) \end{aligned}$$

where c is an unknown constant. Solving it analytically yields

$$\begin{aligned} \rho_0(u(s,t)) &= \left(a(t)\,s\frac{dv(s)}{ds} + b_{1/2}(s,t)\,v(s)\right)\\ &= \frac{1}{2}\left[\Big(a(t) + b_{1/2}(s,t)\Big)U_1 - \Big(a(t) - b_{1/2}(s,t)\Big)U_0\right] \end{aligned} \tag{5.3.14}$$

evaluated at $S_{1/2}$ and for all values of α_0.

Now, a fully discretized expression for the flux is possible. Substituting the discretized flux (5.3.10) and (5.3.14) into (5.3.4) yields

$$\frac{\partial U_i(t)}{\partial t} + \frac{1}{l_i}\left[e_{i,i-1}u_{i-1}(t) + e_{i,i}u_i(t) + e_{i,i+1}u_{i+1}(t)\right] = f_i, \tag{5.3.15}$$

where

$$\begin{aligned} e_{1,1} &= \frac{x_1}{4}\Big(a + b_{1+1/2}\Big) + \frac{b_{1+1/2}\cdot x_{1+1/2}\cdot x_1^{\alpha_1}}{x_2^{\alpha_1} - x_1^{\alpha_1}} + c_1 l_1, && (5.3.16)\\ e_{1,2} &= -\frac{b_{1+1/2}\cdot x_{1+1/2}\cdot x_2^{\alpha_1}}{x_2^{\alpha_1} - x_1^{\alpha_1}}, && (5.3.17) \end{aligned}$$

and

$$e_{i,i-1} = -\frac{b_{i-1/2}\cdot x_{i-1/2}\cdot x_{i-1}^{\alpha_{i-1}}}{x_i^{\alpha_{i-1}}-x_{i-1}^{\alpha_{i-1}}}, \tag{5.3.18}$$

$$e_{i,i} = \frac{b_{i-1/2}\cdot x_{i-1/2}\cdot x_i^{\alpha_{i-1}}}{x_i^{\alpha_{i-1}}-x_{i-1}^{\alpha_{i-1}}} + \frac{b_{i+1/2}\cdot x_{i+1/2}\cdot x_i^{\alpha_i}}{x_{i+1}^{\alpha_i}-x_i^{\alpha_i}} + c_i l_i, \tag{5.3.19}$$

$$e_{i,i+1} = -\frac{b_{i+1/2}\cdot x_{i+1/2}\cdot x_{i+1}^{\alpha_i}}{x_{i+1}^{\alpha_i}-x_i^{\alpha_i}}; \tag{5.3.20}$$

expressions which, although it is not explicitly expressed, depend on time because $a(t)$, $b(s,t)$ and $c(s,t)$.

These $e_{i,i}$ form the following tridiagonal matrix $E \in \mathbb{R}^{N\times N}$

$$E = \begin{bmatrix} e_{1,1} & e_{1,2} & & & & & & \\ e_{2,1} & e_{2,2} & e_{2,3} & & & & & \\ & \ddots & \ddots & \ddots & & & & \\ & & e_{i,i-1} & e_{i,i} & e_{i,i+1} & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & e_{N-1,N-2} & e_{N-1,N-1} & e_{N-1,N} \\ & & & & & e_{N,N-1} & e_{N,N} \end{bmatrix} \tag{5.3.21}$$

and with $U(t) = (U_1(t), \ldots, U_N(t))^\intercal$ and $F(t) = (f_1(t), \ldots, f_N(t))^\intercal$, equation (5.3.15) can be expressed as

$$\frac{\partial U(t)}{\partial t} + \Phi E(t) U(t) = F(t), \tag{5.3.22}$$

where

$$\Phi = \begin{bmatrix} \frac{1}{l_1} & & & \\ & \frac{1}{l_2} & & \\ & & \ddots & \\ & & & \frac{1}{l_N} \end{bmatrix}.$$

We observe that equation (5.3.22) has an unnecessary matrix-matrix multiplication. We can include the term $1/l_i$ in the expressions for $e_{i,i}$ (5.3.16) and (5.3.19)

$$e_{1,1} = \frac{1}{l_1}\frac{x_1}{4}\left(a + b_{1+1/2}\right) + \frac{1}{l_1}\frac{b_{1+1/2}\cdot x_{1+1/2}\cdot x_1^{\alpha_1}}{x_2^{\alpha_1}-x_1^{\alpha_1}} + c_1,$$

$$e_{i,i} = \frac{1}{l_i}\frac{b_{i-1/2}\cdot x_{i-1/2}\cdot x_i^{\alpha_{i-1}}}{x_i^{\alpha_{i-1}}-x_{i-1}^{\alpha_{i-1}}} + \frac{1}{l_i}\frac{b_{i+1/2}\cdot x_{i+1/2}\cdot x_i^{\alpha_i}}{x_{i+1}^{\alpha_i}-x_i^{\alpha_i}} + c_i,$$

and redefine (5.3.22) as

$$\frac{\partial U(t)}{\partial t} + \mathcal{E}(t) U(t) = F(t). \tag{5.3.23}$$

Equation (5.3.22) is a first-order linear ODE system for $U_i(t)$. Corresponding solutions are obtained by applying some of the known methods for this type of problems. For instance [DB02] reviews several state-of-the-art numerical methods to solve ODEs.

Alternatively, it is possible to use the θ-method to obtain a full discretization of ODE (5.3.23). Let us partition the time into M discrete points $j = 1, \ldots, M$ for $t \in [0, T]$ satisfying $0 = t_1 < t_2 < \cdots < t_M = T$, Δt again defined as in (4.1.2). We have

$$\frac{U^{n+1} - U^n}{\Delta t} + \theta \mathcal{E}^{n+1} U^{n+1} + (1-\theta)\,\mathcal{E}^n U^n = \theta F^{n+1} + (1-\theta)\,F^n,$$

with $\mathcal{E}^n = \mathcal{E}(t_n)$, $F^n = F(t_n)$, $U^n = U(t_n)$. Rearranging the last equation leads to

$$\left(\theta \mathcal{E}^{n+1} + \tau\right) U^{n+1} = \theta F^{n+1} + (1-\theta)\,F^n - \left((1-\theta)\,\mathcal{E}^n - \tau\right) U^n,$$

with

$$\tau = \begin{bmatrix} \frac{1}{\Delta t} & & \\ & \ddots & \\ & & \frac{1}{\Delta t} \end{bmatrix} \in \mathbb{R}^{N \times N}.$$

The fully-discrete scheme is $\mathcal{O}(\Delta x_i)$.

5.3.1. Numerical Simulation

We reproduced the results presented by Wang [Wan04] for a binary option, cash-or-nothing type, with $s_{max} = 700$, $K = 400$, $\sigma = 0.4$, $r = 0.1$, $d = 0.04$ and $N = M = 100$. The result is presented in Figure 5.3.1. As we observe, even if the initial condition is discontinuous, no spurious oscillations appear on the solution.

Despite the convenient properties that the method exhibits, we want to bring attention to an issue: we observe in equations (5.3.16)-(5.3.20) that the variable s has as exponent the term

$$\alpha_i = \frac{b(s_i, t)}{a(t)} = \frac{r(t) - d(s_i, t) - \sigma^2(t)}{\frac{1}{2}\sigma^2(t)};$$

for the case when $\sigma^2 \ll r$ we have that $\alpha_i = \mathcal{O}(r/\sigma^2)$, i.e the Péclet number and α_i are in the same order. In the example shown in Figure 5.3.1 we have that $\alpha = -1.25$, whereas for the case considered in Section 5.1 and 5.2 $\alpha = 998$.

In this scheme, when the convection dominated case is considered, the term in the denominator could be out of the range of representable numbers on a computer.

Although nothing is said in [Wan04], we found that it is possible to modify the equations for $e_{i,i}$ in order to avoid numerical issues. We note from (5.3.16)-(5.3.20)

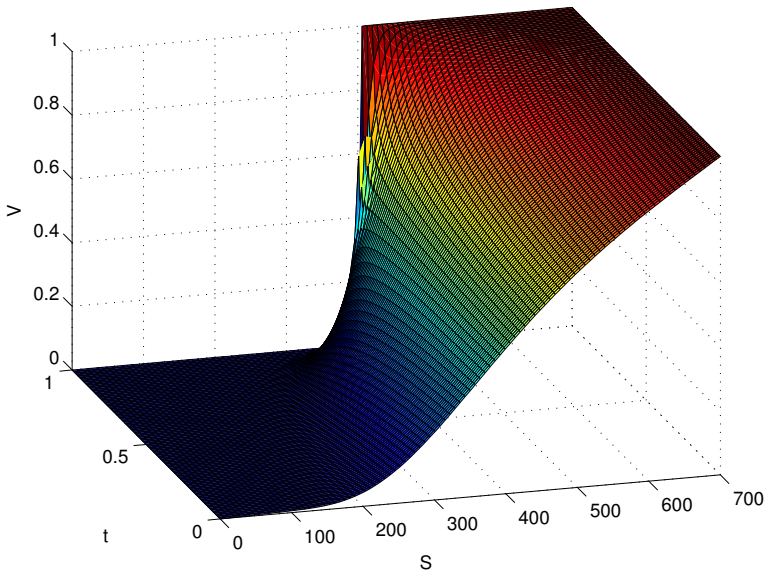


Figure 5.3.1.: Cash-or-Nothing option price obtained with Wang's scheme.

that terms of the form

$$\frac{x_i^{\alpha_i}}{x_{i+1}^{\alpha_i} - x_i^{\alpha_i}}$$

are recurrent. We can express these terms as

$$\frac{1}{\left(\frac{x_{i+1}}{x_i}\right)^{\alpha_i} - 1}, \tag{5.3.24}$$

for instance. We know that the term x_{i+1}/x_i is small because it is equivalent to

$$\frac{x_i + \Delta x_i}{x_i} = 1 + \frac{\Delta x_i}{x_i}$$

and therefore the term (5.3.24) is easier to represent on a computer for the case of a big α_i.

The modified expressions for $e_{i,i}$ are

$$\begin{aligned}
e_{1,1} &= \frac{1}{l_1}\frac{x_1}{4}\left(a + b_{1+1/2}\right) + \frac{1}{l_1}\frac{b_{1+1/2}\cdot x_{1+1/2}}{\left(\frac{x_2}{x_1}\right)^{\alpha_1} - 1} + c_1, \\
e_{2,1} &= -\frac{b_{1+1/2}\cdot x_{1+1/2}}{1 - \left(\frac{x_1}{x_2}\right)^{\alpha_1}}, \\
e_{i,i-1} &= -\frac{b_{i-1/2}\cdot x_{i-1/2}}{\left(\frac{x_i}{x_{i-1}}\right)^{\alpha_{i-1}} - 1}, \\
e_{i,i} &= \frac{1}{l_i}\frac{b_{i-1/2}\cdot x_{i-1/2}}{1 - \left(\frac{x_{i-1}}{x_i}\right)^{\alpha_{i-1}}} + \frac{1}{l_i}\frac{b_{i+1/2}\cdot x_{i+1/2}}{\left(\frac{x_{i+1}}{x_i}\right)^{\alpha_i} - 1} + c_i, \\
e_{i,i+1} &= -\frac{b_{i+1/2}\cdot x_{i+1/2}}{1 - \left(\frac{x_i}{x_{i+1}}\right)^{\alpha_i}}.
\end{aligned}$$

However, the method is just order one and therefore does not represent any advantage over the exponentially fitted scheme from Section 5.2 because Wang's scheme is more difficult to implement. In addition to that, we must be aware that as $\Delta x_i \to 0$ the terms on the denominator also tend to zero, creating further issues.

5.4. The Kurganov-Tadmor Scheme

We apply now the scheme presented in Section 4.8 to the Black-Scholes equation (5.0.1). We want to transform the Black-Scholes equation to the general form

$$\frac{\partial}{\partial t}u\left(x,t\right) + \frac{\partial}{\partial x}\mathcal{F}\left(u\right) = \frac{\partial}{\partial x}\mathcal{Q}\left(u,u_x\right) + \mathcal{S}\left(x,t,u\right)$$

where $\mathcal{S}$ is the source. To this end, the following expressions

$$\begin{aligned}
\frac{\partial}{\partial s}\left(sv\left(s,t\right)\right) &= s\frac{\partial}{\partial s}v\left(s,t\right) + v\left(s,t\right), \\
\frac{\partial}{\partial s}\left(s^2\frac{\partial}{\partial s}v\left(s,t\right)\right) &= s^2\frac{\partial^2}{\partial^2 s}v\left(s,t\right) + 2s\frac{\partial}{\partial s}v\left(s,t\right),
\end{aligned}$$

can be used to get the Black-Scholes equation to the required form

$$\frac{\partial}{\partial t}v\left(s,t\right) + \frac{\partial}{\partial s}\left(\left(\sigma^2 - r + d\right)sv\left(s,t\right)\right) = \frac{\partial}{\partial s}\left(\frac{1}{2}\sigma^2 s^2\frac{\partial}{\partial s}v\left(s,t\right)\right) + \left(\sigma^2 - 2r + d\right)v\left(s,t\right).$$

Therefore, the fluxes are defined as

$$\begin{aligned}\mathcal{F}(s,v) &:= \left(\sigma^2 - r + d\right) sv(s,t),\\ \mathcal{Q}(s,v) &:= \frac{1}{2}\sigma^2 s^2 \frac{\partial}{\partial s} v(s,t),\\ \mathcal{S}(v) &:= \left(\sigma^2 - 2r + d\right) v(s,t);\end{aligned}$$

with these definitions it is possible to proceed now applying formulae from Section 4.8 to the Black-Scholes equation.

We observe that the expression for $a_{i+1/2}(t)$ is simplified because we are dealing with a scalar case and

$$\frac{\partial}{\partial v}\mathcal{F}(s,v) \equiv \mathcal{F}_v = \left(\sigma^2 - r + d\right) s;$$

in this sense,

$$a_{i+1/2}(t) = \left|\mathcal{F}_v\left(s_{i+1/2}\right)\right|.$$

On the other hand, $\mathcal{Q}$ does not depend on $v(x,t)$ but only on the derivative $\partial v/\partial s$. Hence, the expression for P is also simplified, namely

$$P_{i+1/2}(t) = \mathcal{Q}\left(\frac{V_{i+1}(t) - V_{i-1}(t)}{2\Delta s}\right),$$

in which the second-order approximation for the derivative is used. At the boundaries, we can use the following second-order formulae to approximate the derivatives for $\mathcal{Q}$

$$\begin{aligned}\frac{\partial}{\partial s} v(s_{min},t) &= \frac{-3V_0(t) + 4V_1(t) - V_2(t)}{2\Delta s} + \mathcal{O}\left(\Delta s^2\right),\\ \frac{\partial}{\partial s} v(s_{max},t) &= \frac{V_{N-1}(t) - 4V_N(t) + 3V_{N+1}(t)}{2\Delta s} + \mathcal{O}\left(\Delta s^2\right),\end{aligned}$$

where V_0 represents the approximation at s_{min} and V_{N+1} at s_{max}.

The semi-discrete scheme for the Black-Scholes equation takes the form

$$\frac{dV_i}{dt} = -\frac{1}{\Delta s}\left[H_{i+1/2}(t) - H_{i-1/2}(t)\right] + \frac{1}{\Delta s}\left[P_{i+1/2}(t) - P_{i-1/2}(t)\right] + \mathcal{S}(v),$$

with

$$\begin{aligned}H_{i+1/2}(t) &= \frac{1}{2}\left[\mathcal{F}\left(s_{i+1/2}, V^+_{i+1/2}\right) + \mathcal{F}\left(s_{i+1/2}, V^-_{i+1/2}\right)\right]\\ &\quad -\frac{a_{i+1/2}(t)}{2}\left[V^+_{i+1/2}(t) - V^-_{i+1/2}(t)\right],\\ H_{i-1/2}(t) &= \frac{1}{2}\left[\mathcal{F}\left(s_{i-1/2}, V^+_{i-1/2}\right) + \mathcal{F}\left(s_{i-1/2}, V^-_{i-1/2}\right)\right]\\ &\quad -\frac{a_{i+1/2}(t)}{2}\left[V^+_{i-1/2}(t) - V^-_{i-1/2}(t)\right],\end{aligned}$$

and

$$\begin{aligned} V^{+}_{i+1/2}(t) &= V_{i+1}(t) - \frac{1}{2}\Delta s\,(V_s)_{i+1}(t) \\ V^{-}_{i+1/2}(t) &= V_i(t) + \frac{1}{2}\Delta s\,(V_s)_i(t) \\ V^{+}_{i-1/2}(t) &= V_i(t) - \frac{1}{2}\Delta s\,(V_s)_i(t) \\ V^{-}_{i-1/2}(t) &= V_{i-1}(t) + \frac{1}{2}\Delta s\,(V_s)_{i-1}(t)\,. \end{aligned}$$

The derivative $(V_s)_i(t)$ is approximated with a minmod limiter such that the semi-discrete scheme fulfills the Total variation diminishing (TVD) condition [KT00].

The generalized minmod limiter is defined as

$$(V_s)_i(t) = \text{minmod}\left(\theta\frac{V_i(t) - V_{i-1}(t)}{\Delta s}, \frac{V_{i+1}(t) - V_{i-1}(t)}{2\Delta s}, \theta\frac{V_{i+1}(t) - V_i(t)}{\Delta s}\right), \tag{5.4.1}$$

where $1 \leq \theta \leq 2$ and the minmod function is defined as

$$\text{minmod}(x_1, x_2, \ldots) = \begin{cases} \min_i(x_i) & \text{if } x_i > 0\ \forall i, \\ \max_i(x_i) & \text{if } x_i < 0\ \forall i, \\ 0 & \text{otherwise.} \end{cases}$$

5.4.1. European Options

To test the scheme and its properties, we would like to take as an example a convection dominated case with known analytic solution. For this reason, an European option with $r = 0.46$, $\sigma = 0.02$, $d = 0$, $K = 70$, $s_{min} = 0$, $s_{max} = 100$, and $T = 1$ is chosen. Although this setup is financially unrealistic, it is useful as a stress-test for the scheme under a high Péclet number. The value of the Péclet number for this setting is

$$Pe \propto \frac{r}{\sigma^2} = 1150.$$

In the case of an European option with Dirichlet boundary conditions, these conditions are included in the calculation of the derivative. For example, for $i = 1$ we have

$$(V_s)_1(t) = \text{minmod}\left(\theta\frac{V_1(t) - g_{s_{min}}(t)}{\Delta s}, \frac{V_2(t) - g_{s_{min}}(t)}{2\Delta s}, \theta\frac{V_2(t) - V_1(t)}{\Delta s}\right),$$

where $g_{s_{min}}(t)$ represents the prescribed boundary condition at s_{min}. A similar strategy is followed for the terms $V^{\pm}_{i+1/2}$. For instance

$$V^{-}_{1-1/2} = g_{s_{min}}(t) + \frac{1}{2}\Delta s\,(V_s)_{i-1}(t)\,.$$

N	300	400	500	600	700
$\left\|v\left(s,T\right)-V_i^M\right\|_2$	0.066468	0.054178	0.041912	0.032635	0.026116
Δs^2	0.110374	0.062189	0.039840	0.027685	0.020350

Table 5.3.: Error measured in Euclidean norm for the price of an European option obtained via second order Kurganov-Tadmor scheme.

The parameter θ is chosen problem-wise. The value $\theta = 1$ ensures non-oscillatory behavior. We found empirically that the values $\theta \in [1.5, 2]$ produce better results in this test example. This behavior is also reported in [KT00] for the scalar examples presented.

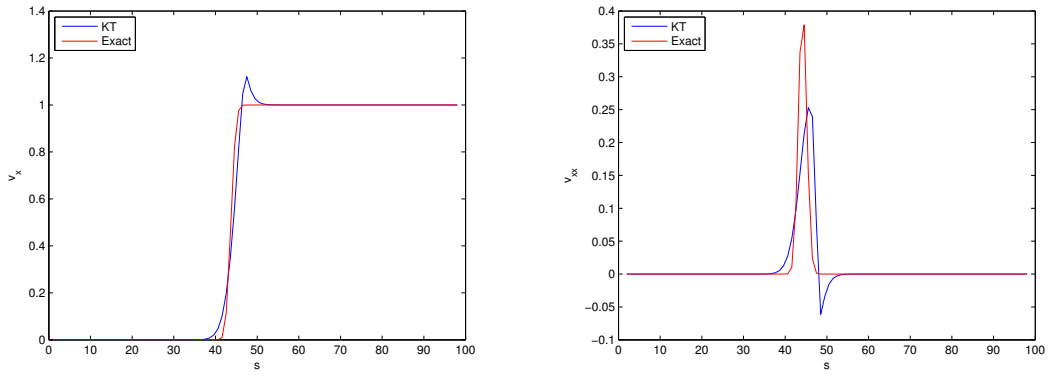
To justify our selection for the value of θ, we present three cases with $N = 100$ using an ODE integrator with automatic time step selection.

From Figures (5.4.1a), (5.4.1b) and (5.4.1c) it is observed that $\theta = 1$ provides the worst result for both the first and the second derivative. For the first derivative, $\theta = 2$ gives the best results but the second derivative is deficient in the sense that it is over estimated. The value $\theta = 1.5$ is the best for both the first and the second derivative. It is remarkable to see that the resolution obtained is already very good for $N = 100$.

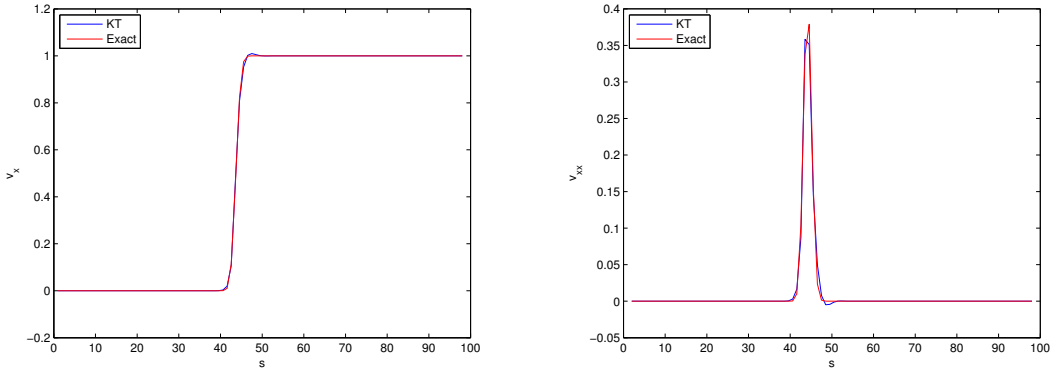
We select $\theta = 1.5$ for the minmod limiter for all the simulations and proceed with $N = 500$. The result is shown in Figure 5.4.2 and 5.4.3. The approximation for the price is quite good as in the case for $N = 100$ but it is easily spotted that the approximation for the Δ and Γ is improving fast thanks to the order of convergence of the method.

Results of the computational order of convergence for Kurganov-Tadmor scheme, measured with the euclidean norm, are shown in Table 5.3. It can be seen that the convergence behaves as theory predicts. Another way to see the computational order of convergence is by measuring how the error decreases when the step size is reduced. This is done in the Section 5.4.5 for the case of a nonlinear Black-Scholes equation.

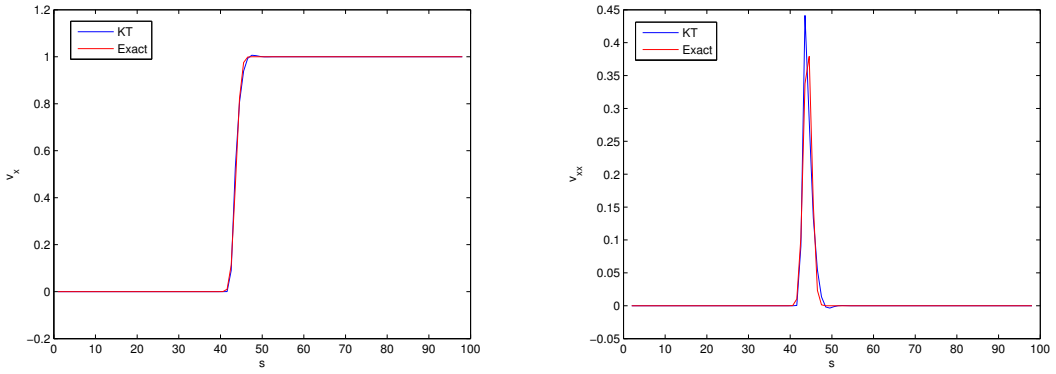
The Kurganov-Tadmor scheme gives very good results for the benchmark problem presented in this Section: a convection dominated PDE with a high Péclet number. The first derivative Δ of the approximation of the price is now free of oscillations and it is easily observed that almost no artificial diffusion is introduced. It is remarkable how good the approximation is for the second derivative of the price of the option, which is shown in Figure 5.4.3b. In other cases like the Exponentially Fitted scheme, due to the artificial diffusion introduced by the method, the second derivative is already quite a deficient approximation even if the first derivative is acceptable. It can be seen that the Kurganov-Tadmor scheme repre-



(a) First and second derivative of the price with $\theta = 1$.

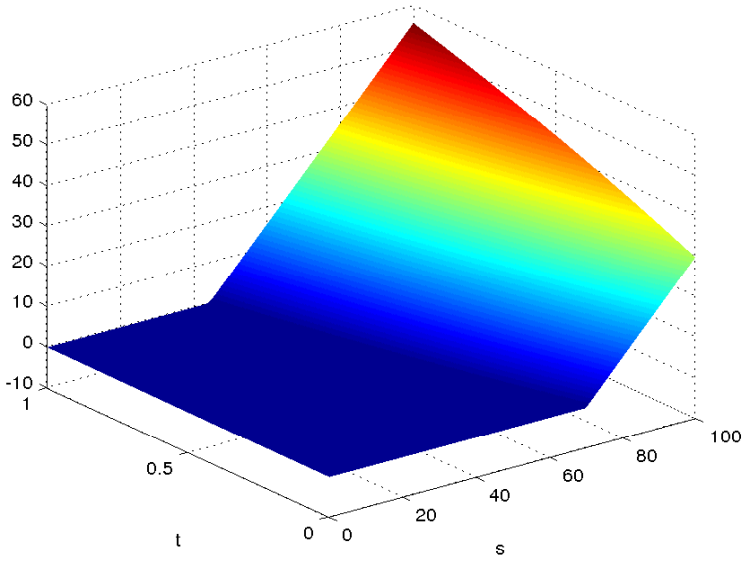


(b) First and second derivative of the price with $\theta = 1.5$.

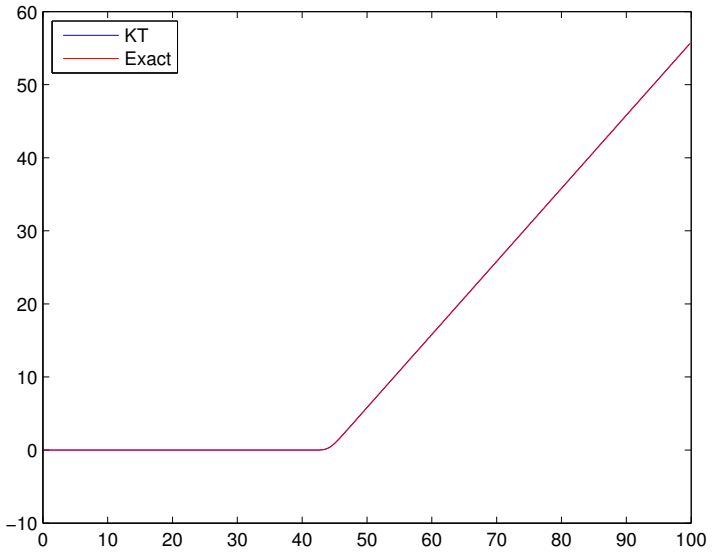


(c) First and second derivative of the price with $\theta = 2$.

Figure 5.4.1.: Results for different values of θ.

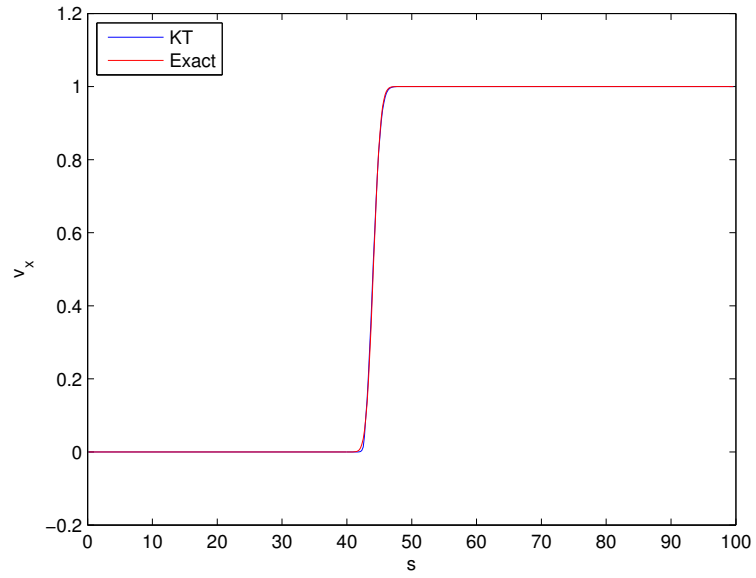


(a) Surface for $t \in [0, T]$.

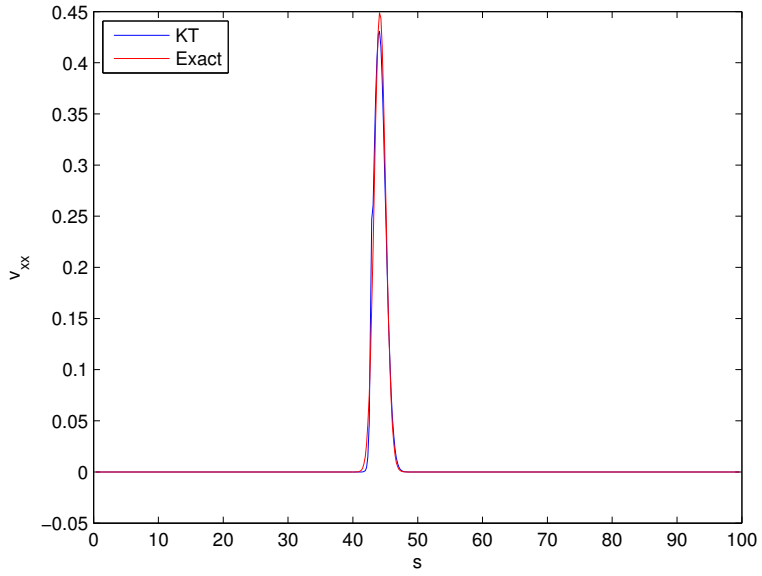


(b) Price at $t = T$.

Figure 5.4.2.: Price of the option obtained via Kurganov-Tadmor Scheme.



(a) First derivative Δ.



(b) Second derivative Γ.

Figure 5.4.3.: Greeks of the option price.

sents a big advantage in comparison to other schemes available for two reasons: the convergence properties and its flexibility. Discontinuities and non-smoothness on the initial condition are handled satisfactorily.

We now proceed to test the Kurganov-Tadmor scheme with options that present hyperbolic behavior even for realistic parameters. In addition, some of the following examples do not have an analytic solution.

5.4.2. Asian Options

In this section, the full expression (3.1.1) for the pricing of an Asian option is simulated. We perform a variable change as $t^* = T - t$ but, again, for convenience we simply denote t^* as t:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 v}{\partial s^2} + rs\frac{\partial v}{\partial s} - ru + \frac{1}{T-t}(s-a)\frac{\partial v}{\partial a} \tag{5.4.2}$$

with boundary conditions

$$\frac{\partial v}{\partial t} = -\frac{a}{T-t}\frac{\partial v}{\partial a} - rv \text{ for } s = 0, \tag{5.4.3}$$

$$\frac{\partial v}{\partial t} = \frac{s_{max} - a}{T-t}\frac{\partial v}{\partial a} \text{ for } s = s_{max}. \tag{5.4.4}$$

We note that equation (5.4.2) is not defined for $t = T$, but when this happens $s = a$ and the last term is dropped out.

Before continuing the discussion on the pricing of the option, we point out that the Kurganov-Tadmor scheme is easily extended to two spatial dimensions plus the temporal dimension [KT00]. The scheme takes the form

$$\begin{aligned}\frac{dV_{i,j}(t)}{dt} = &-\frac{1}{\Delta s}\left[H^s_{i+1/2,j}(t) - H^s_{i-1/2,j}(t)\right] - \frac{1}{\Delta a}\left[H^a_{i,j+1/2}(t) - H^a_{i,j-1/2}(t)\right] \\ &+ \frac{1}{\Delta s}\left[P^s_{i+1/2,j}(t) - P^s_{i-1/2,j}(t)\right] + \frac{1}{\Delta a}\left[P^a_{i,j+1/2}(t) - P^a_{i,j-1/2}(t)\right],\end{aligned}$$

with the usual definitions for $H(t)$ and $P(t)$.

Now we define the fluxes with the same technique as in Section 5.4.1. The convective fluxes are:

$$\begin{aligned}\mathcal{F}^s(s,v) &= \left(\sigma^2 - r\right)sv, \\ \mathcal{F}^a(s,a,v) &= -\frac{1}{T-t}(s-a)v;\end{aligned}$$

on the other hand, there is diffusion flux only for the spatial direction s

$$\mathcal{Q}^s(s,v,v_s) = \frac{1}{2}\sigma^2 s^2 \frac{\partial v}{\partial s}.$$

Finally, the source is

$$\mathcal{S} = \left(\sigma^2 - 2r + \frac{1}{T-t}\right) v.$$

Boundary conditions are of Neumann type, i.e. with derivatives both in time and in the spatial dimension a. Including expressions (5.4.3) and (5.4.4) into the discretization is not easy because the expressions H^s and H^a are elaborated. Nevertheless, we noticed that those expressions are analytically solvable and their exact solutions are defined uniquely by the initial condition, i.e the payoff. The boundaries for a fixed strike Asian put option are:

$$\begin{aligned} v(0,a,t) &= \max\left(0, K - \frac{1}{T}(T-t)\,a\right)\exp(-rt), \\ v(s_{max},a,t) &= \max\left(0, K - \frac{1}{T}\left[s_{max}t + a(T-t)\right]\right). \end{aligned}$$

For the expression (5.4.4) an analytic solution is obtained as is. For expression 5.4.3, a transformation is needed. By defining

$$\tilde{v} = v\exp(rt),$$

and substituting it into the PDE, the boundary condition takes the form

$$\frac{\partial \tilde{v}}{\partial t} = \frac{a}{T-t}\frac{\partial \tilde{v}}{\partial a},$$

which now is easily solved. An inverse transformation to get the original variable v back is straightforward obtained.

An example from [Sey09] for a fixed strike put is replicated in Figure 5.4.4 with $K = 100$, $T = 0.2$, $r = 0.05$, $\sigma = 0.25$ and $N = 50$. The value for $\theta = 1.5$ as in the case for the simulation for the European option; $s_{min} = a_{min} = 0$ and $s_{max} = a_{max} = 200$. The simulation for $t = 0.06$ is achieved in an average of 80 seconds with an ODE solver with automatic step selection for the time dimension.

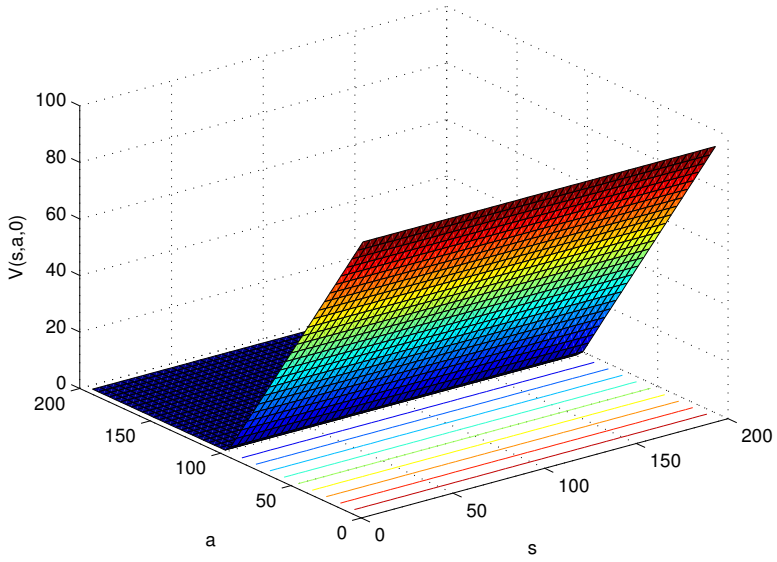
The final price and its derivative is shown in Figure 5.4.5. An analytic solution is not available to compare with, but we rely on the results from Section 5.4.1 in which the scheme performed effectively under challenging conditions.

Now we present the case of a floating strike put option with $r = 0.15$, $\sigma = 0.3$, $T = 1$ and $N = 50$. The initial condition is

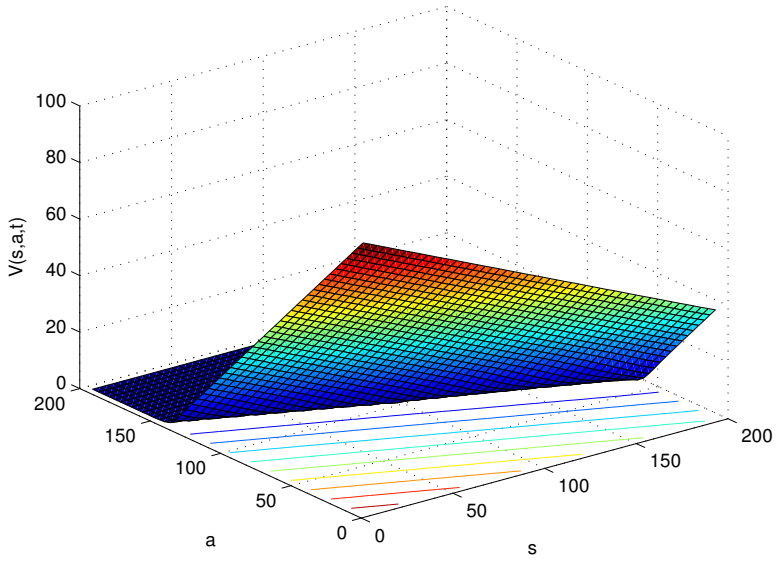
$$v(s,a,0) = (a-s)^+$$

and the boundary conditions take the form

$$\begin{aligned} v(0,a,t) &= \max\left(0, -s_{min} + \frac{1}{T}(T-t)\,a\right)\exp(-rt) \\ v(s_{max},a,t) &= \max\left(0, -s_{max} + \frac{1}{T}\left[s_{max}t + a(T-t)\right]\right). \end{aligned}$$

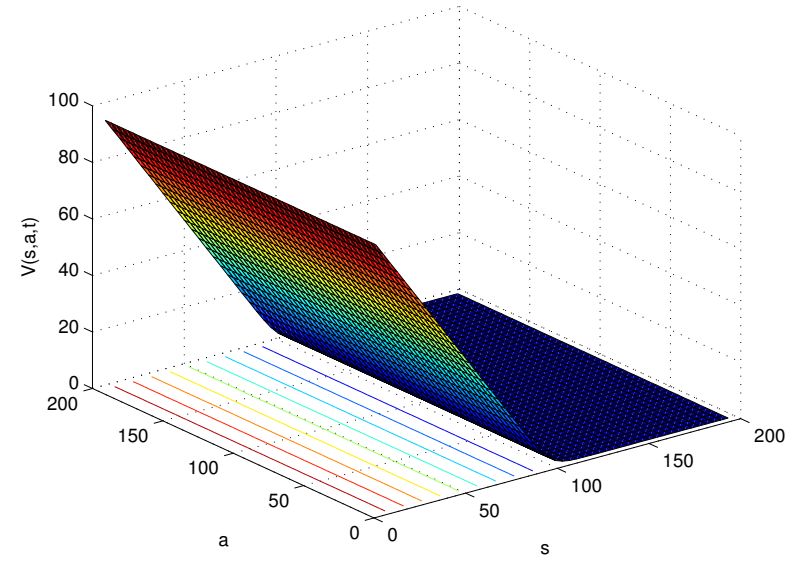


(a) Initial Condition.

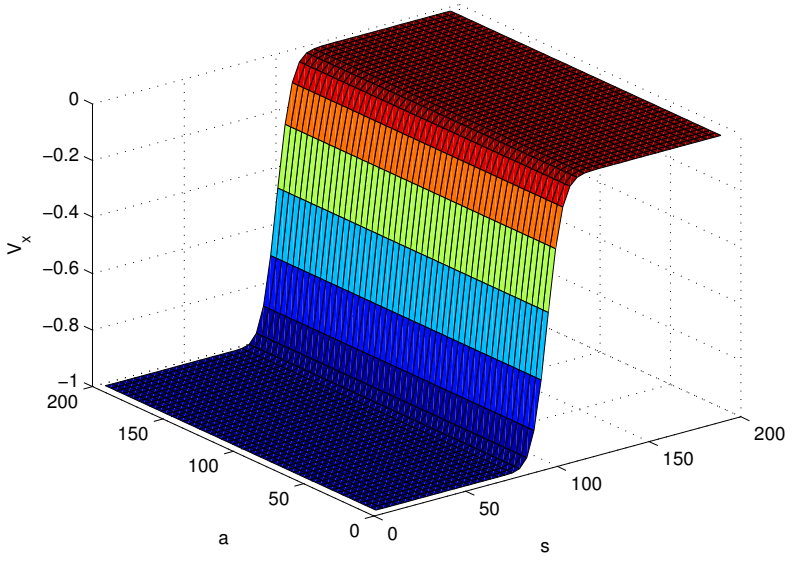


(b) Price at $t = 0.06$.

Figure 5.4.4.: Simulation of a fixed strike Asian option.



(a) Price.



(b) Derivative.

Figure 5.4.5.: Simulation of a fixed strike Asian option at $t = T$.

The final price along with the initial condition is shown in Figure 5.4.6. The derivative in this case shows some oscillation near $s = 0$, as shown in Figure 5.4.7. Further research must be done in order to understand this behavior.

For the case of the Asian options, the reader may have noticed that the simulations were performed with $N = 50$, whereas in the case for the plain vanilla European option we performed simulations with $N = 500$. The reason is because in this case we are solving an ODE with 50×50 elements, i.e. the computational cost is much higher and, in addition, the memory requirements are also high. Furthermore, for $N = 100$ the simulation is impossible to run on a normal computer: the memory is exhausted rapidly because the ODE solver saves the result matrix of size N^2 for each time step and because our implementation keeps several arrays of size N^2 during execution.

By profiling the Kurganov-Tadmor scheme implemented in Matlab – c.f. Appendix A – we could notice that the routine for the minmod derivative was taking a big part of the running time. Due to its design, Matlab is highly efficient processing operations in vector or matrix form and is rather inefficient with element-wise operations. Now, it is evident that the minmod limiter (5.4.1) requires many element-wise operations: the derivative must be analyzed at an element-by-element level at every time step. This characteristic introduces unnecessary overhead in the running time. In order to alleviate this issue, we coded the minmod routine in C and compiled it with the MEX [Mat11] utility provided with Matlab suite. In this way we take advantage of Matlab's efficiency working with matrices and vectors and low-level programming speed of execution with element-wise operations. The execution time was improved from taking $\sim$ 10 minutes in average to $\sim$ 4 minutes in average for the case of the fixed strike option. For the case of a floating strike option the improvement is more noticeable: from $\sim$ 40 minutes in average to $\sim 6 - 8$ minutes. We think this is a great gain, taking into account that the routine for the minmod limiter is fairly simple to code in C – c.f. Appendix A.7.

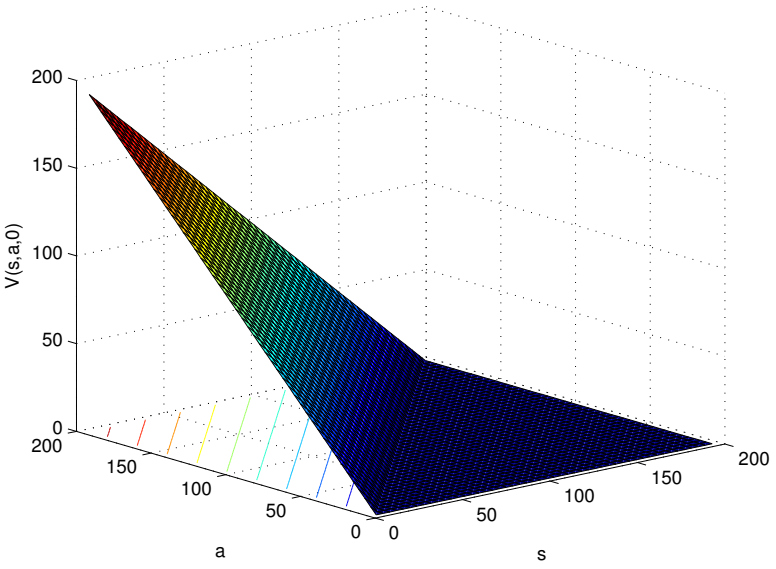
5.4.3. Similarity Reduction for Asian Options

Wilmott Reduction

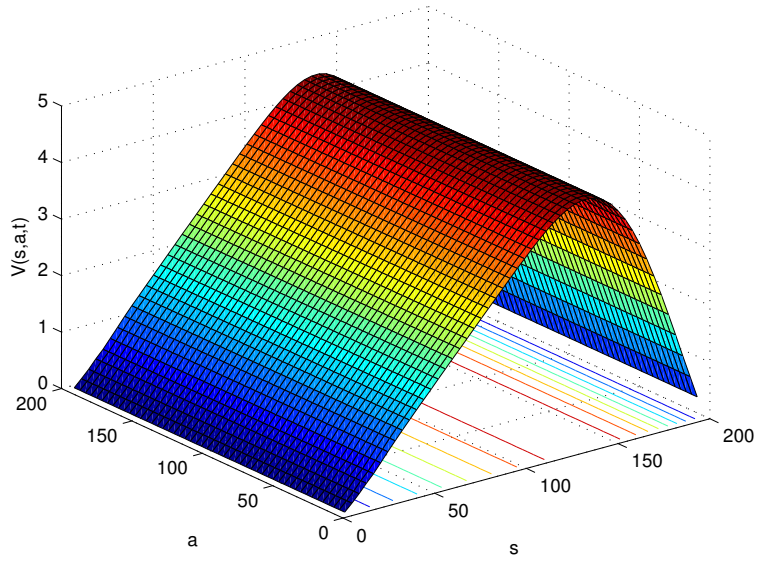
In Section 3.1.1 a reduction of the full PDE (3.1.1) for floating strike Asian options was presented. As in past sections, we first express the PDE in forward-in-time form and then we define the convective and diffusive fluxes, if any. This PDE is convection dominated for a small σ as well as for $x \to 0$.

The PDE forward in time takes the form

$$\frac{\partial y}{\partial t} = \frac{1}{2}\sigma^2 x^2 \frac{\partial^2 y}{dx^2} + (1 - rx)\frac{\partial y}{\partial x},$$



(a) Initial Condition.



(b) Price at $t = T$.

Figure 5.4.6.: Floating strike Asian option.

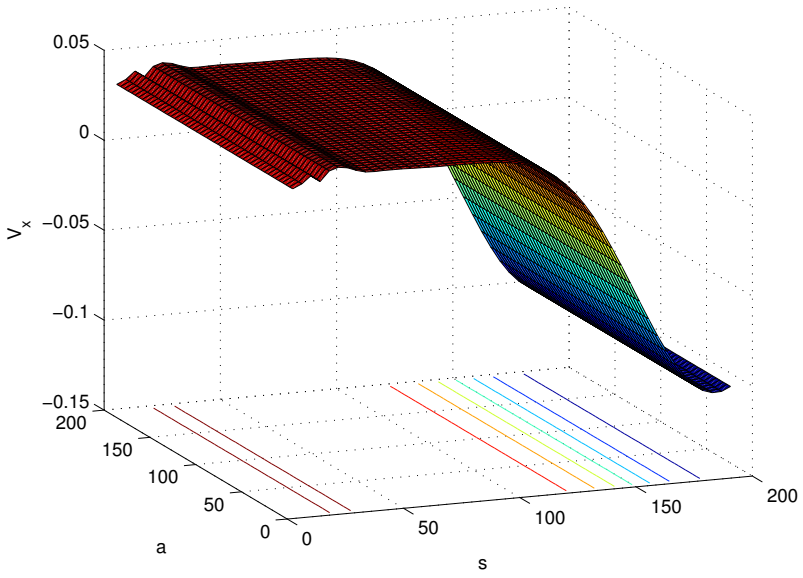


Figure 5.4.7.: Floating strike Asian option price derivative at $t = T$.

where the initial condition for a call is

$$y(x,0) = \left(1 - \frac{1}{T}x\right)^{+}.$$

The boundary condition (3.1.6) for $x = 0$ takes the form

$$\frac{\partial y}{\partial t} = \frac{\partial y}{\partial x} \tag{5.4.5}$$

and $y = 0$ for $x \to \infty$.

As in Section 5.4.2 the boundary conditions are solved analytically when needed or possible. The equation (5.4.5) has an analytic solution. The unknown boundary conditions for put options are obtained with the put-call parity.

The fluxes for this PDE are straightforward with the same technique that was used in last sections for the same purpose. Hence

$$\begin{aligned} \mathcal{F} &= -\left(1 - rx - \sigma^2 x\right) y, \\ \mathcal{Q} &= \frac{1}{2}\sigma^2 x^2 y, \\ \mathcal{S} &= \left(\sigma^2 + r\right) y. \end{aligned}$$

Comparison

In Table 5.4 we compare the results in [RS95] for a floating-strike Asian put and $s = 100$ and the results from the Kurganov-Tadmor scheme with the PDE shown in Section 3.1.2. The results provided by Rogers and Shi were obtained with routine D03PAF from NAG's library with $\Delta x = 0.005$ and are listed in column NAG-RS. Columns LB and UB are lower and upper boundaries also from Rogers-Shi. The column KT-Wilmott corresponds to an approximation obtained with the Kurganov-Tadmor scheme with the same step size. For the same Δx we observe slightly different approximations between the two schemes. The results highlighted with a star are outside the boundaries. For this example it can be seen that in general both schemes usually deliver approximation within bounds. It is also clear that the Kurganov-Tadmor scheme provides approximations that are always higher than those provided by the NAG routine. Unfortunately, we do not have enough information on the routine D03PAF to provide a comment regarding its performance because in addition to be part of a closed-source library, it has been deprecated in favor of more general routines D03PCF/D03PCA.

In Table 5.5 a list of results for a floating-strike Asian call is shown. The comparison is between a Crank-Nicolson Implicit Method (CN), a High Order Compact scheme (HOC), a Monte Carlo (MC) proposed in [KWC11] versus the Kurganov-Tadmor scheme with the Wilmott (KT-W) reduction, all of them with $N = 500$. It is easy to spot that the worst performer is the Crank-Nicolson scheme providing approximations far away from the other three schemes whereas the High Order Compact scheme results are similar to those obtained with the Kurganov-Tadmor scheme. It is interesting to notice that for the case when volatility is small, i.e. $\sigma = 0.05$, the greatest deviation between HOC and KT-W columns occurs. Furthermore, it can be see that under convection-dominated environments, the columns CN, HOC and MC are similar in contrast to the approximation obtained with Kurganov-Tadmor – observe, for example, the case for $\sigma = 0.05$ and $r = 0.2$ with $r/\sigma^2 = 80$. It is expected that the Crank-Nicolson method is affected by the hyperbolic behavior and, judging by the results listed in 5.5, it might be possible that the High-Order Compact scheme is also affected.

An analytic solution for the PDE proposed by Wilmott is not available to compare with; nevertheless, we know from experiments in Section 5.4.1 that the Kurganov-Tadmor scheme performs satisfactorily on convection-dominated environments with Péclet numbers as high as 1000, and in this sense we are confident the results from this scheme are better than the Crank-Nicolson and the High-Order Compact scheme. It is difficult to compare a finite difference method against a Monte Carlo method, but it is a good reference to have prices obtained with this method as an independent check.

σ	r	NAG-RS	KT-Wilmott	LB	UB
0.1	0.05	1.257	*1.441	1.245	1.355
	0.09	0.709	0.817	0.699	0.825
	0.15	0.271	0.294	0.252	0.415
0.2	0.05	*3.401	3.656	3.404	3.831
	0.09	2.622	2.826	2.622	3.062
	0.15	1.723	1.855	1.710	2.187
0.3	0.05	5.628	5.902	5.625	6.584
	0.09	*4.736	4.980	4.738	5.706
	0.15	3.612	3.803	3.609	4.604

Table 5.4.: Floating-strike Asian put option comparison with values listed in Table 5 from [RS95] for $s = 100$.

σ	r	CN	HOC	MC	KT-W
0.05	0.06	3.5025	3.1391	3.1509	2.9686
	0.1	5.1148	4.8784	4.8734	4.6809
	0.2	9.3988	9.3449	9.3486	9.2018
0.1	0.06	4.1353	3.8929	4.0124	3.8955
	0.1	4.0124	5.3592	5.4183	5.2841
	0.2	9.5333	9.4385	9.433	9.2962
0.2	0.06	6.1337	5.9919	6.1172	6.0316
	0.1	7.2951	7.1641	7.2625	7.1731
	0.2	10.547	10.4486	10.4894	10.3942
0.3	0.06	8.3256	8.2462	8.3155	8.2404
	0.1	9.3669	9.2902	9.3484	9.2735
	0.2	12.2035	12.1361	12.163	12.0931
0.4	0.06	10.5403	10.4921	10.5358	10.4614
	0.1	11.5081	11.4607	11.4952	11.4244
	0.2	14.0885	14.0444	14.0581	13.9955

Table 5.5.: Floating-strike Asian call option comparison with prices in Table 1 and 2 from [KWC11].

5.4.4. Rogers-Shi Reduction for Asian Options

Now, a simulation of the Rogers-Shi PDE reduction presented in Section 3.1.2 with the Kurganov-Tadmor scheme is performed. This PDE is convection dominated for small σ and for short maturity times T.

The forward-in-time PDE reads

$$\frac{\partial w}{\partial t} = \frac{1}{2}\sigma^2 x^2 \frac{\partial^2 w}{\partial x^2} - \left(\rho\left(t\right) + rx\right)\frac{\partial w}{\partial x},$$

and the boundary conditions are changed accordingly.

The fluxes are

$$\begin{aligned} \mathcal{F} &= \left(\rho\left(t\right) + rx + \sigma^2 x\right) w, \\ \mathcal{Q} &= \frac{1}{2}\sigma^2 x^2 \frac{\partial w}{\partial x}, \\ \mathcal{S} &= \left(\sigma^2 + r\right) w. \end{aligned}$$

The expression for $\rho\left(t\right)$ is defined depending on the type of the Asian option: fixed or floating strike.

For this PDE we do not provide a put-call parity because to the author's best knowledge, it does not exist yet. Rogers and Shi provide [RS95] Dirichlet-type boundary conditions for both floating and fixed strike call options.

Comparison with other schemes

A comparison between the data in [RS95] and the prices obtained with Kurganov-Tadmor scheme is shown in Table 5.6. The column RS refers to the results of a fixed strike Asian call option presented by Rogers-Shi with the NAG routine D03PAF with $\Delta x = 0.005$, $s = 100$ and $\sigma = 0.05$. The column KT1 shows the results obtained with the Kurganov-Tadmor scheme with the same step-size. The column KT2 is a simulation with $\Delta x = 0.00125$.

Looking at the column KT1 we notice that most of the time the Kurganov-Tadmor scheme yields results that are very close to or between the bounds. In contrast, the approximations on column RS are either above or below the interval defined by the bounds in all cases. Again, a star is placed to highlight a result if it is either up or down the interval defined by the bounds. It is remarkable that when comparing the approximations in column KT1 for different values of r, the Kurganov-Tadmor scheme produces better results when the hyperbolic behavior is present, i.e. for the case when $r = 0.15$ all our approximations in column KT1 are inside the bounds.

r	Strike	RS	KT1	KT2	LB	UB
0.05	95	*7.157	7.180	7.178	7.174	7.183
	100	*2.621	*2.742	2.719	2.713	2.722
	105	*0.439	*0.311	0.334	0.337	0.343
0.09	95	8.823	8.809	8.809	8.809	8.821
	100	*4.185	*4.324	4.311	4.308	4.318
	105	*1.011	0.958	0.958	0.958	0.968
0.15	95	*11.090	11.094	11.094	11.094	11.114
	100	*6.777	6.796	6.795	6.794	6.810
	105	*2.639	*2.768	2.747	2.744	2.761

Table 5.6.: Comparison of the results obtained in Table 1 in [RS95] vs. the Kurganov-Tadmor scheme.

We list column KT2 with a grid four times finer than KT1 to show the convergence of the method. The simulation is achieved in approximately 1.5 minutes. In column KT2 all the approximations are inside the bounds.

It is reported in [RS95] that as r increases, a rise in the simulation time is observed as well. Depending on the algorithm used, it could be expected that increasing r, leaving σ fixed, results in an increase on the simulation time because the convection-dominated behavior arises. This increase in simulation time is not observed with the Kurganov-Tadmor scheme.

The Table 5.8 list a comparison of price of a fixed-strike Asian call option with $r = 0.09$ obtained with the Kurganov-Tadmor scheme against two other approximations: the column labeled as Chen-Lyuu lists prices obtained with the method proposed in [CL07] which is a lower bound for the price. The values listed in column Hsu-Lyuu were obtained with lattice algorithm that exhibit quadratic-time convergence proposed in [HL07]. The column labeled as Exact is obtained with a semi-analytic method proposed in [Zha01]. As Chen and Lyuu pointed out, the reason for testing the method with such large volatilities is because many formulas and numerical schemes deteriorate its approximation as the volatility increases. At the right-hand side of each approximation we provide the difference between the price obtained and the exact value. From this error we observe that the Kurganov-Tadmor scheme gives similar approximations to those obtained by the Hsu-Lyuu lattice algorithm. We also observe that the Chen-Lyuu formula does deteriorate with as the volatility is increased whereas the performance of the the Hsu-Lyuu lattice and the Kurganov-Tadmor scheme are stable for all σ.

Given that we are using asymptotic boundary conditions, we must pay special attention to the cases when the diffusion term is big in order to fulfill those condi-

σ	Strike	MC	KT	LB	UB
0.10	95	8.91	8.91	8.91	8.95
	100	*4.91	4.92	4.92	5.10
	105	*2.06	2.07	2.07	2.34
0.30	90	*14.96	15.02	14.98	15.23
	100	*8.81	8.84	8.83	9.39
	110	*4.68	*4.67	4.70	5.37
0.50	90	*18.14	18.31	18.18	18.52
	100	*12.98	13.09	13.02	13.69
	110	*9.10	*9.16	9.18	9.97

Table 5.7.: Comparison of the results for a fixed strike Asian option in Table 6 in [RS95] vs. the Kurganov-Tadmor scheme.

tions, i.e. our discrete interval must be large enough. This is because the diffusion term is responsible for the smoothing of the solution. For example, in Figure 5.4.8a the volatility is $\sigma = 0.05$ and we observe that the solution is almost a traveling wave of the initial condition in the sense that little smoothing is observed at $t = T$. On the other hand, in Figure 5.4.8b the volatility $\sigma = 1.0$ and it is easily observed that with the interval $x \in [-1, 3]$ the boundary condition $w(x, t) = 0$ as $x \to \infty$ is not fulfilled properly and as a consequence wrong approximations are achieved. The interval used in Table 5.8 for $\sigma \in [0.05, 0.6]$ is $x \in [-1, 3]$ and for $\sigma \in (0.6, 1.0]$ is $x \in [-4, 5]$, but the value for Δx is kept fixed.

Another popular method to price derivatives is Monte Carlo method. In Table 5.7 a list of prices obtained with Monte Carlo (MC) in [RS95] is shown along with a simulation with Kurganov-Tadmor (KT) scheme for a fixed strike Asian call at $s = 100$ with $T = 1$, $r = 0.09$, $\Delta x = 0.0025$ and different volatilities. As mentioned in last section, it is complicated to compare a FDM with Monte Carlo techniques, but comparisons like the one shown in Table 5.7 serve, for example, as a good reference to check that the FDM is not affected by other unknown phenomenon. In the comparison, the Kurganov-Tadmor scheme out-performs this popular numerical technique, where it is considered a good approximation when it is inside the bounds. The approximations outside the interval of the bounds is marked with a star. It is important to remark that the simulation with the Kurganov-Tadmor scheme took ~ 20 seconds and although we do not have that information from the simulation with the Monte Carlo method, we know that the convergence for Monte Carlo could be as bad as $\mathcal{O}\left(\mathcal{N}^{1/2}\right)$ where $\mathcal{N}$ is the number of simulated paths, therefore a higher computational effort is needed, which results in slower simulations.

σ	K	Exact	Hsu-Lyuu		Chen-Lyuu		KT	
0.05	95	8.8088392	8.808717	$1.22e-04$	8.808839	$2.00e-07$	8.808983	$1.44e-04$
	100	4.3082350	4.309247	$1.01e-03$	4.308231	$4.00e-06$	4.318153	$9.92e-03$
	105	0.9583841	0.960068	$1.68e-03$	0.958331	$5.31e-05$	0.957513	$8.71e-04$
0.1	95	8.9118509	8.912238	$3.87e-04$	8.911839	$1.19e-05$	8.914974	$3.12e-03$
	100	4.9151167	4.914254	$8.63e-04$	4.915075	$4.17e-05$	4.920286	$5.17e-03$
	105	2.0700634	2.072473	$2.41e-03$	2.069930	$1.33e-04$	2.069732	$3.31e-04$
0.2	95	9.9956567	9.995661	$4.30e-06$	9.995362	$2.95e-04$	9.997251	$1.59e-03$
	100	6.7773481	6.777748	$4.00e-04$	6.776999	$3.49e-04$	6.778434	$1.09e-03$
	105	4.2965626	4.297021	$4.58e-04$	4.295941	$6.22e-04$	4.296475	$8.76e-05$
0.3	95	11.6558858	11.656062	$1.76e-04$	11.654758	$1.13e-03$	11.656542	$6.56e-04$
	100	8.8287588	8.829033	$2.74e-04$	8.827548	$1.21e-03$	8.829166	$4.07e-04$
	105	6.5177905	6.518063	$2.72e-04$	6.516355	$1.44e-03$	6.517857	$6.65e-05$
0.4	95	13.5107083	13.510861	$1.53e-04$	13.507892	$2.82e-03$	13.511028	$3.20e-04$
	100	10.9237708	10.923943	$1.72e-04$	10.920891	$2.88e-03$	10.923937	$1.66e-04$
	105	8.7299362	8.730102	$1.66e-04$	8.726804	$3.13e-03$	8.729934	$2.20e-06$
0.5	95	15.4427163	15.442822	$1.06e-04$	15.437069	$5.65e-03$	15.442907	$1.91e-04$
	100	13.0281555	13.028271	$1.15e-04$	13.022532	$5.62e-03$	13.028286	$1.30e-04$
	105	10.9296247	10.929736	$1.11e-04$	10.923750	$5.87e-03$	10.929685	$6.03e-05$
0.6	95		17.406402		17.396428		17.406609	
	100		15.128426		15.118595		15.128486	
	105		13.113874		13.103855		13.113802	
0.8	95		21.349949		21.326144		21.349981	
	100		19.288780		19.265518		19.288744	
	105		17.423935		17.400803		17.423820	
1.0	95		25.252051		25.205238		25.252053	
	100		23.367535		23.321951		23.367513	
	105		21.638238		21.593393		21.638200	

Table 5.8.: Data corresponding to Table 3 in Chen-Lyuu.

5.4.5. A Nonlinear Black-Scholes Equation

Another interesting example to show is the non-linear Black-Scholes equation proposed by Windcliff et al [WWFV07]. The non-linear PDE arises when hedging a contingent claim with an asset that is not perfectly correlated with the underlying asset. For example, the contingent claim is written on an asset with price s that cannot be traded. Instead, a reference, correlated asset is used to price the option.

Let us define ρ as the correlation between the underlying asset and the reference, λ as the *risk loading* parameter and

$$q = \begin{cases} \text{sign}\left(\frac{\partial v}{\partial s}\right) & \text{for a short position,} \\ -\text{sign}\left(\frac{\partial v}{\partial s}\right) & \text{for a long position,} \end{cases}$$

then the non-linear Black-Scholes takes the form

$$\frac{\partial v}{\partial t} = \max_{q\in[-1,1]} \left[\left(r' + q\lambda\sigma\sqrt{1-\rho^2} \right) s\frac{\partial v}{\partial s} + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 v}{\partial s^2} - rv \right] \tag{5.4.6}$$

for a short position and

$$\frac{\partial v}{\partial t} = \min_{q\in[-1,1]} \left[\left(r' + q\lambda\sigma\sqrt{1-\rho^2} \right) s\frac{\partial v}{\partial s} + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 v}{\partial s^2} - rv \right] \tag{5.4.7}$$

for a long position. The term r' is a function of the drift rate μ of the stochastic process driving the asset S and reference asset's drift rate μ' , namely

$$r' = \mu - (\mu' - r)\frac{\sigma\rho}{\sigma'}$$

with σ' defined as the volatility of the reference asset.

The boundary conditions are

$$\frac{\partial v}{\partial t} = -rv \text{ for } s \to 0, \tag{5.4.8}$$

$$v = As\exp\left(\left(r' - r + q\lambda\sigma\sqrt{1-\rho^2}\right)t\right) + B\exp(-rt) \text{ for } s \to \infty \tag{5.4.9}$$

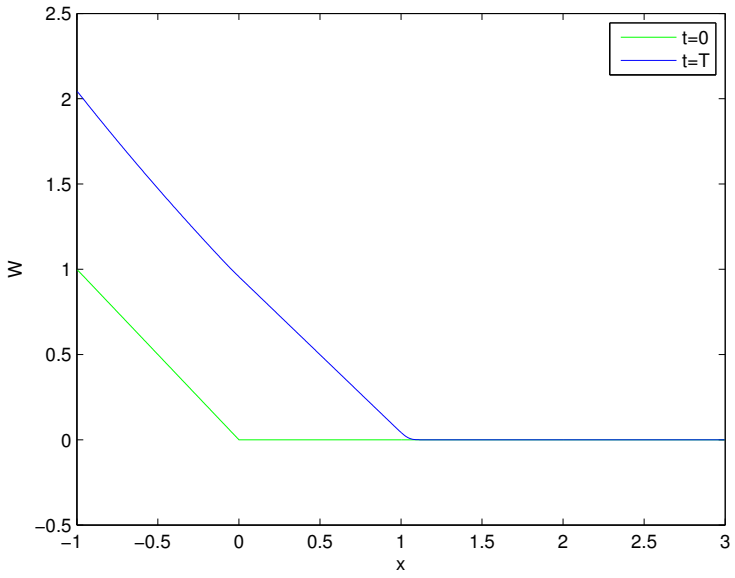
where A and B depend on the initial condition, i.e. the payoff.

For discretization purposes we can write equations (5.4.6) and (5.4.7) as

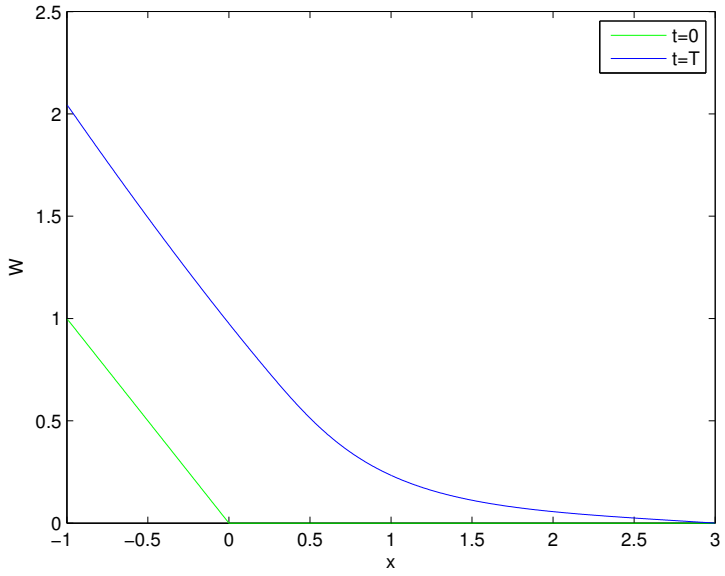
$$\frac{\partial v}{\partial t} = \left(r' + q\lambda\sigma\sqrt{1-\rho^2} \right) s\frac{\partial v}{\partial s} + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 v}{\partial s^2} - rv.$$

The fluxes are

$$\begin{aligned} \mathcal{F} &= -\left(r' + q\lambda\sigma\sqrt{1-\rho^2} - \sigma^2 \right) sv, \\ \mathcal{Q} &= \frac{1}{2}\sigma^2 s^2 \frac{\partial v}{\partial s}, \\ \mathcal{S} &= \left(\sigma^2 - r' - q\lambda\sigma\sqrt{1-\rho^2} \right) v, \end{aligned}$$



(a) Small diffusion term $\sigma = 0.05$.



(b) Large diffusion term $\sigma = 1.0$.

Figure 5.4.8.: Different diffusion terms for the Rogers-Shi PDE.

r	0.05
ρ	0.9
σ	0.2
μ	0.07
σ'	0.3
$\mu' = r + (\mu - r)\,\frac{\sigma'\rho}{\sigma}$	0.077
λ	0.2
$r' = \mu - (\mu' - r)\,\frac{\sigma\rho}{\sigma'}$	0.0538
K	100
T	1

Table 5.9.: Set of parameters for a short straddle simulation.

which define all we need to apply the Kurganov-Tadmor scheme.

As an example, we perform a simulation with a short straddle option with the parameters shown in Table 5.9.

The payoff is

$$v\,(s,0) = (K-s)^+ + (s-K)^+$$

and the boundary conditions are

$$v\,(s_{min},t) = K\exp\,(-rt)$$

for s_{min}. For s_{max} at $t = 0$ we have from equation (5.4.9) that

$$\begin{aligned}(K - s_{max})^+ + (s_{max} - K)^+ &= As_{max}\exp\,(0) + B\exp\,(0)\,,\\ s_{max} - k &= As_{max} + B,\end{aligned}$$

therefore $A = 1$ and $B = -K$ and the boundary condition at s_{max} is

$$v\,(s_{max},t) = s_{max}\exp\left(\left(r' - r + q\lambda\sigma\sqrt{1-\rho^2}\right)t\right) - K\exp\,(-rt)\,.$$

A comparison of the results presented in [WWFV07] and our results obtained with the Kurganov-Tadmor scheme is shown in Table 5.10 for $s = 100$ and $t = T$.

In the column labeled as CN, the result obtained with Crank-Nicolson proposed by Windcliff is shown whereas the column KT is the price obtained with Kurganov-Tadmor scheme. A small discrepancy is between the Crank-Nicolson and the Kurganov-Tadmor scheme is evident. Nevertheless, the numerical method proposed by Windcliff is achieved with a standard central, forward or backward difference formula, i.e. the first derivative is – following Windcliff notation – approximated as

$$(V_s)^n_{i,cent} = \frac{V^n_{i+1} - V^n_{i-1}}{s_{i+1} - s_{i-1}},$$

N	CN	KT
51	17.10144	17.92125
101	17.12367	17.98633
201	17.12899	18.00273
401	17.13021	18.00702
801	17.13050	18.00799
1601	17.13058	18.00817

Table 5.10.: Comparison between Crank-Nicolson vs. Kurganov-Tadmor method with parameters in Table 5.9.

r	0.03
ρ	0.5
σ	0.7
μ	0.04
σ'	0.25
$\mu' = r + (\mu - r)\frac{\sigma'\rho}{\sigma}$	0.0317
λ	0.9
$r' = \mu - (\mu' - r)\frac{\sigma\rho}{\sigma'}$	0.0375
K	100
T	1

Table 5.11.: Second set of parameters for a short straddle simulation.

for the central difference case. Thanks to our experiments in Section 5.1.2 we know that even for the fully implicit, second-order scheme, issues arose for certain important cases – c.f. [WWFV07] Appendix A for a detailed description of the method used to obtain the values in column CN. In addition to that, the numerical viscosity introduced by the central scheme is $\mathcal{O}\left(\Delta x^{2p}/\Delta t\right)$, where p is the order of convergence.

It is difficult to say which is a better approximation or which approximation is correct but given the aforementioned issues with CN, we are biased to think that the Kurganov-Tadmor scheme achieves a better approximation.

For a short straddle but different set of parameters – listed in Table 5.11 – a list of approximations is shown in Table 5.12. As expected, a discrepancy between both methods also appears for the second set of parameters.

Because we do not have an exact formula to compare with, we calculate the computational order of convergence by observing how the error decreases when the discretization steps increase. Using the parameters in Table 5.9, the results are shown on Table 5.13.

Let V_1 be the approximation at s_i and T with N_1 discretization points and V_2

s	CN	KT
10	91.2063	93.9848
20	85.3930	87.9937
30	79.7849	82.2132
100	102.8771	103.9016

Table 5.12.: Crank-Nicolson vs Kurganov-Tadmor for different stock values with $N = 401$.

N	KT	Error	CC
21	17.39550		
41	17.87244	0.47694	
81	17.97490	0.10245	4.6552
161	17.99992	0.02503	4.0941
321	18.00607	0.00615	4.0696
641	18.00759	0.00152	4.0405
1281	18.00801	0.00041	3.7027
2561	18.00810	0.00010	4.1797

Table 5.13.: Computational convergence of the Kurganov-Tadmor scheme.

the approximation with N_2 for $N_1 < N_2$, then the error column in Table 5.13 is defined as

$$e = |V_1 - V_2| ,$$

and the column labeled as CC represents the computational order of convergence. The Kurganov-Tadmor scheme exhibits quadratic convergence, i.e. by doubling the step-size, the error is decreased by a factor of 4. This is a good property because as Windcliff states, given that the PDE is nonlinear, it is difficult to assess whether the numerical method would converge to the financially relevant solution.

6. Conclusion

Three numerical methods to solve the issues that arise when convection-dominated behavior is present were studied in this thesis. All of them have advantages and disadvantages but based on the experiments performed, we concluded that the Kurganov-Tadmor scheme is the most flexible one.

The Exponentially Fitted scheme presented in Section 5.2 is straightforward to implement: the discretization of the system leads to a three-point relation of the values on the current layer, similar to the relation obtained with the fully implicit method. Dirichlet boundary conditions are easily included into the scheme. For conditions of Neumann type, the discrete boundary condition is obtained and placed accordingly into the discretization. A sparse linear system of size N is built and can be solved efficiently with numerical libraries specially suited for that purpose: BLAS [LHKK79] and LAPACK [ABB+99]. Because the system is tridiagonal, the computational effort to solve it at each time step is $\mathcal{O}(N)$. The method is convergent of order one and artificial diffusion is introduced as can be seen in Figures 5.2.1a and 5.2.1b. Near the strike price of the option, it is evident that the approximation is deficient due to the artificial diffusion. Although the oscillation issues are solved and the linear system solution is obtained efficiently with proper libraries, a large number of grid points must be used in order to obtain a decent approximation.

Wang's finite volume method in Section 5.3 is a robust method as it was shown with an example with discontinuous payoff: no oscillations are present. When a simulation of an European option is performed, we obtain similar results to those obtained with the Exponentially Fitted scheme. Wang's method is also convergent of order one. A big disadvantage of the method is that it cannot handle very high Péclet numbers. This is because the numerical expression for the matrix E in equation (5.3.22) contains an exponent in the spatial variable which is of the same order as the Péclet number, see for instance equation (5.3.19). This characteristic is reflected in the fact that the numerical method produces extremely big numbers that cannot be represented with native data-types on a computer and instead an arbitrary-precision library must be used, which introduces more computational effort. We proposed a modification to the Wang's scheme in order to be able to properly handle convection-dominated cases with a very high Péclet number. Nevertheless, further issues appear because as $\Delta x \to 0$ the denominator of equation (5.3.24) also goes to zero, creating, again, big numbers as the result

of the division. In the limit case, a division by zero. The palliative proposed by us solves one issue but creates another. The method performs satisfactorily but only for a certain ranges of values of σ and the risk free rate. An additional point to this discussion is the fact that because Wang's scheme has convergence order one, it does not represent any advantage in comparison with the Exponentially Fitted scheme in terms of convergence properties. In terms of simplicity, Wang's scheme is elaborated and therefore more effort is required to implement it, but has the advantage of providing a semi-discretization form which is useful when it is required/desired to use an ODE solver with automatic step selection, among other advantages. Finally, Wang's scheme is not flexible in the sense that it is not possible to apply a general option type but is rather suitable only for European and American options.

The Kurganov-Tadmor scheme was presented in Section 5.4. Extensive experiments and comparisons were made with this scheme because for various reasons we think it is the best of the three methods presented. The method exhibits a convergence of order two which represents an advantage over the Exponentially Fitted and the Wang's scheme; the quadratic convergence is achieved even for the nonlinear Black-Scholes simulation as shown in Table 5.13. In general, we consider that a method of quadratic convergence is a good balance between the computational effort and the quality of an approximation.

It was found that the flexibility of Kurganov-Tadmor scheme is very convenient for the pricing problem when different PDEs are used: by transforming the pricing PDE to the general convection-diffusion equation 4.8.2 it is possible to apply the method transparently. A proof-of-concept code is shown in Appendix A and it can be seen that by defining the fluxes and the boundary conditions, practically the same code can be used regardless of the problem.

Another advantage of the Kurganov-Tadmor scheme is that the numerical viscosity introduced is only $\mathcal{O}\left(\Delta x\right)^{2p-1}$ where p is the order of convergence, i.e. an order $\mathcal{O}\left(\Delta x\right)^{3}$ for this scheme, whereas other central schemes introduce numerical viscosity $\mathcal{O}\left({}^{\Delta x}/_{\Delta t}\right)^{2p}$. This characteristic allows to obtain a semi-discrete expression of the method by letting $\Delta t \to 0$ and take advantage of the existing methods to solve ODEs. In this work, an ODE solver with automatic time step selection was used in all the experiments. One disadvantage of the semi-discrete form of the Kurganov-Tadmor scheme is that the resulting ODE system is stiff and this means that special routines for stiff ODEs must be used and in general these routines require a higher computational effort than those for non-stiff ODEs. In our simulations of Asian options with the full PDE in Section 5.4.2 it was found that the stiffness of the ODE system is affected by the type of initial condition, i.e. if we are simulating a floating or fixed strike option.

Because of the form of the numerical fluxes defined in the expression (4.8.9),

it is difficult to include the discretized boundary conditions of Neumann type – like the ones in the Wilmott reduction for Asian options – into the scheme and for this reason we decided to solve the boundary conditions analytically and use the put-call parity to obtain the boundaries when those were not available for certain instrument: for example, in Section 3.1.1, the boundary conditions for a call are presented based on boundedness assumptions on the second derivative when the price of the underlying goes to zero. The other condition is obtained from the payoff for the case when the price goes to infinity. The unknown boundary conditions for a put can be easily obtained with the put-call parity from the known ones.

We presented three PDEs for Asian options: one considered a full Black-Scholes expression for this type of option, other as a reduction of the full PDE and a third which is obtained by independent arguments.

The Wilmott similarity reduction was compared to the Rogers-Shi reduction in Table 5.4 with data originally obtained in [RS95]. This was done this way due to the lack of data available for this similarity reduction in scientific journals. The data listed in this table shows that in general the Kurganov-Tadmor scheme leads to approximations very similar to those in [RS95]. These approximations are most of the time inside the interval defined by the bounds. In Table 5.7 we compare again the Rogers-Shi PDE discretized with the Kurganov-Tadmor scheme against a Monte Carlo technique. Surprisingly, all the approximations obtained with the Kurganov-Tadmor are considerable superior to those obtained with Monte Carlo.

In Table 5.5 the same PDE with the Kurganov-Tadmor scheme was compared to other three methods with data from [KWC11]. In this table it is evident the advantages of having a proper scheme for convection dominated PDEs. By comparing with the Crank-Nicolson column it is possible to spot the difference. The other two methods lead to similar results as those obtained with the Kurganov-Tadmor scheme.

The Rogers-Shi reduction was also compared to other numerical schemes techniques. In Table 5.6 the approximations with the Kurganov-Tadmor scheme are listed. We note that this method delivers better approximations – being a good approximation the one that is inside the bounds – in comparison to the scheme used in [RS95]. If we use a finer grid, then the power of the scheme is evident: all the approximations converge to values inside the interval between LB and UP. The Table 5.8 provides plenty of data to compare with. We observe that the Kurganov-Tadmor scheme does not lose accuracy when the volatility is increased, as reported in [CL07].

While the simulations for PDEs with one temporal and one spatial variable were finished in a matter of seconds or, at most, in a few minutes, the simulation for the Asian options with the full PDE – one temporal and two spatial variables – is

much more demanding in terms of computational cost and memory management. We give two reasons for this: the first is the well-known curse of dimensionality. This phenomenon is well-known and a rule of thumb is to say that when the dimensionality $p > 3$ then it is not longer efficient to use finite difference methods. The second reason is because of our implementation, Matlab's syntax and the ODE routine. As can be seen in Appendix A the method requires to keep track of the derivative of the current values of the solution and other several quantities which are all of the same size of the solution vector `v0` and depending on the number N of discretization steps it means that several vectors of size N^2 exist in memory at execution time; this is done so because Matlab is much more efficient working with matrices and vectors than working with arrays in an element-wise manner. In addition to that, the routine used to solve the ODE system saves in memory the solution for each time step, instead of just delivering the solution at $t = T$, characteristic that causes the memory to fill up very fast. For $N = 100$, 8GB of RAM memory are rapidly exhausted causing the operating system to request space on the swap memory – in Linux – which is also rapidly exhausted, leading to instability of the operating system. We took advantage of the possibility of using mixed Matlab language and C code to achieve a more efficient technique in terms of running time with very good results. This extra effort of coding the routine for the minmod derivative is not significant due to the simple derivative expression and its almost direct translation into C code. Nevertheless, the memory issues were still there. A possible solution for the high memory requirement issue is to code the method completely in a programming language like C or C++. We think it is not entirely convenient to implement the Kurganov-Tadmor scheme in Matlab for PDEs like (3.1.1) with three dimensions, but instead we recommend low-level programming languages.

Finally, a nonlinear Black-Scholes equation was simulated. This PDE is proposed in [WWFV07] and little data are provided in the original paper. Comparisons are shown in Table 5.10 for the first set of parameters. The difference between the method proposed – a Crank-Nicolson method – and the Kurganov-Tadmor is significant. Nevertheless, we must keep in mind that the approximation of the price is obtained exactly at the strike price. From Figure 2.2.1c we observe that the payoff of the straddle is non-smooth exactly at the strike price. In addition to that, we know that Crank-Nicolson performs quite bad exactly at this non-smooth parts leading to a smeared approximations. On the other hand, we know that the Kurganov-Tadmor scheme performs adequately at discontinuities or non-smooth parts of the initial condition. The computational order of convergence is as expected, showing quadratic convergence.

Overall, there are two properties of the Kurganov-Tadmor method which are remarkable: its flexibility and its reliability. Thanks to its flexibility, we were able

to test a wide range of PDEs with the same scheme in an easy manner. Thanks to its reliability we can simulate a broad range of instruments with different payoffs, disregarding discontinuities or non-smoothness on it.

A big limitation is that in environments like Matlab it is difficult to be efficient handling memory due to both the characteristics of the scheme and the Matlab's syntax. In some cases, having a semi-discrete scheme is considered as disadvantageous because a separate routine is needed to solve the ODE system and this requires time to code if it is not readily available.

Bibliography

[ABB+99] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK users' guide*, third ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1999.

[CFL28] R. Courant, K. Friedrichs, and H. Lewy, *Über die partiellen Differenzengleichungen der Mathematischen Physik*, Mathematische Annalen **100** (1928), 32–74.

[CL07] K.W. Chen and Y.D. Lyuu, *Accurate pricing formulas for Asian options*, Applied Mathematics and Computation **188** (2007), 1711 – 1724.

[DB02] P. Deuflhard and F. Bornemann, *Scientific computing with Ordinary Differential Equations*, Springer-Verlag, 2002.

[Duf06] D. Duffy, *Finite difference methods in financial engineering: A partial differential equation approach*, Wiley, 2006.

[EGH00] R. Eymard, T. Gallouët, and R. Herbin, *Finite volume methods*, Solution of Equation in $\mathbb{R}^n$ (Part 3), Techniques of Scientific Computing (Part 3) (P.G. Ciarlet and J.L. Lions, eds.), Handbook of Numerical Analysis, vol. 7, Elsevier, 2000, pp. 713 – 1018.

[GRS07] C. Grossmann, H.G. Roos, and M. Stynes, *Numerical treatment of Partial Differential Equations*, Springer-Verlag, 2007.

[HL07] W.W.Y. Hsu and Y. Lyuu, *A convergent quadratic-time lattice algorithm for pricing European-style Asian options*, Applied Mathematics and Computation **189** (2007), 1099–1123.

[Il'69] A. M. Il'in, *Differencing scheme for a differential equation with a small parameter affecting the highest derivative*, Mathematical Notes **6** (1969), 596–602.

[KIT11] KITCO, *Historical Gold Charts and Data*, http://www.kitco.com/charts/historicalgold.html, 2011, Retrieved 31-Oct-2011.

[KT00] A. Kurganov and E. Tadmor, *New high-resolution central schemes for nonlinear conservation laws and Convection-Diffusion equations*, J. Comput. Phys **160** (2000), 241–282.

[KWC11] A. Kumar, A. Waikos, and S. P. Chakrabarty, *Pricing of average strike Asian call option using numerical PDE methods*, ArXiv e-prints (2011).

[LeV05] R.J. LeVeque, *Numerical methods for Conservation Laws*, Birkhäuser, 2005.

[LHKK79] C. L. Lawson, R. J. Hanson, D. R. Kincaid, and F. T. Krogh, *Basic linear algebra subprograms for fortran usage*, ACM Trans. Math. Softw. **5** (1979), 308–323.

[Mat11] Mathworks, *External Interfaces*, http://www.mathworks.com/help/techdoc/matlab_external/bp_kqh7.html, 2011, Retrieved 5-Dec-2011.

[OCC11] OCC, *Daily Volume Statistics – September 2011*, http://www.theocc.com/webapps/daily-volume-statistics, 2011, Retrieved 26-Oct-2011.

[Pul10] Roland Pulch, *Numerical analysis and simulation of partial differential equations*, Lecture notes, University of Wuppertal, 2010.

[Roo94] H.G. Roos, *Ten ways to generate the Il'in and related schemes*, J. Comput. Appl. Math. **53** (1994), no. 1, 43 – 59.

[RS95] L. C. G. Rogers and Z. Shi, *The value of an Asian option*, J. Appl. Prob. **32** (1995), no. 4, 1077–1088.

[Sey09] R.U. Seydel, *Tools for Computational Finance*, fourth ed., Springer-Verlag, 2009.

[Str89] J.C. Strikwerda, *Finite difference schemes and partial differential equations*, second ed., SIAM, 1989.

[Wan04] S. Wang, *A novel fitted finite volume method for the Black-Scholes equation governing option pricing*, IMA J. Numer. Anal. (2004).

[WDH94] P. Wilmott, J. Dewynne, and S. Howison, *Option pricing: Mathematical models and computation*, Oxford Financial Press, 1994.

[WWFV07] H. Windcliff, J. Wang, P. A. Forsyth, and K. R. Vetzal, *Hedging with a correlated asset: Solution of a nonlinear pricing PDE*, J. Comput. Appl. Math. **200** (2007), 86–115.

[Zha01] J.E. Zhang, *A semi-analytical method for pricing and hedging continuously sampled arithmetic average rate options*, J. Computational Finance **5** (2001), no. 1, 59–79.

A. Matlab Source Code

Note: some comments are in written as pseudo latex expressions in order to handle optimally the sub and super scripts.

A.1. Example script

```
%  This is an example of how the Kurganov-Tadmor function scheme coded in
% the function 'kt_blackscholes2' could be used to solve the Black-Scholes
% equation numerically.
%  The function 'kt_blackscholes2' receives several parameters while the
% function handle for the ODE solver needs only two parameters. For this
% purpose, we create a 'proxy' function called 'odefun'. See line 44.

params = struct( ...
    'r',      0.05, ...
    'd',      0.0, ...
    'sigma',  0.15, ...
    'K',      70, ...
    'T',      1, ...
    's_min',  0, ...
    's_max',  100, ...
    'theta',  1.5, ...
    'N',      900 ...
);

% Discretized interval with boundaries included
s_all = linspace(params.s_min, params.s_max, params.N+2);

% Discretized interval without boundaries
s = s_all(2:params.N+1)';

% Step size
ds = abs(s(2) - s(1));

% Initial Condition (European call option)
v0 = max(s - params.K, 0);

% Boundary Conditions (function handles)
bc1 = @(t) 0;
bc2 = @(t) params.s_max*exp(-params.d*t) - params.K*exp(-params.r*t);

% Function handles for the Fluxes
F  = @(x, v) (params.sigma^2 - params.r) .* x .* v;
dF = @(x, v) (params.sigma^2 - params.r) .* x;
Q  = @(x, v, vx) 0.5 * params.sigma^2 .* x.^2 .* vx;
S  = @(x, v) (params.sigma^2 - 2 * params.r) .* v;
```

```
% Solving the ODE system
odefun = @(t, v) kt_blackscholes2(t, v, s, ds, params, F, dF, Q, S, bc1, bc2);
options = odeset('RelTol', 1e-6);

tic
[t, V] = ode15s(odefun, [0, params.T], v0, options);
toc

[sg, tg] = meshgrid(s, t);

surf(sg, tg, V, 'EdgeColor', 'none');
title('Surface for the price of the option from t=0 to t=T');
xlabel('s');
ylabel('t');
zlabel('V');
```

A.2. Main function

```
function v1 = kt_blackscholes2(t, v0, x, dx, params, F, dF, Q, S, bc1, bc2)
%  The function kt_blackscholes2 implements the Kurganov-Tadmor scheme of
% second order for an European option.
%  This function is feeded to the ODE solver as the right- hand side of the
% system to be solved.
%
%  Parameters:
%       t: current time (provided by the ODE solver)
%      v0: values on the current time step (provided by the ODE solver)
%       x: space discretization
%      dx: step size
%  params: structure which contains parameters for the option
%          -params.r:     risk-free interest rate
%          -params.d:     dividends
%          -params.sigma: volatility
%          -params.K:     strike price
%          -params.T:     maturity
%          -params.s_min: minimum value for the space discretization
%          -params.s_max: maximum value for the space discretization
%          -params.theta: value of theta for the minmod limiter
%          -params.N:     number of discretization points
%       F: function implementing the convective flux. This function
%          receives as parameters x and v0
%      dF: derivative of the convective flux with respect to the solution.
%          This function receives as parameters x, v0 and the derivative of
%          v0 with respect to x
%       Q: function implementing the diffusive flux. This function receives
%          as parameters x and v0
%       S: source function. This function receives as parameters x and v0
%     bc1: function left boundary condition (modify parameters accordingly).
%     bc2: function right boundary condition (modify parameters accordingly).

% Note: This function does not check for correctness of parameters (i.e.,
% that the function handles are passed correctly or that the params
% structure contains all the required elements) because it is called many
```

```
% times by the ODE solver (some times thousands of calls) and efficiency is
% important.

% For clarity, a copy of the parameters is made. For more efficiency the
% 'params' structure could be used instead.
r     = params.r;
d     = params.d;
sigma = params.sigma;
K     = params.K;
T     = params.T;
s_min = params.s_min;
s_max = params.s_max;
theta = params.theta;
N     = params.N;

% Derivative of v0 with respect to x obtained with the minmod limiter
v0x = derivative_minmod(v0, dx, theta, bc1(t), bc2(t));

% x_{i+1/2}
x_ph = x + dx/2;

% x_{i-1/2}
x_mh = x - dx/2;

% Convective Flux ------------------------------------------------------------
% Calculating v0^{-}_{j+1/2}
vmp = v0 + dx/2*v0x;


% Extrapolation of a value of the derivative we do not have access to.
v0x_bc2 = v0x(N-1) - (s_max+dx - x(N-1))/(s_max - (s_max-dx))*(v0x(N) - v0x(N-1));

% Calculating v0^{+}_{j+1/2}
vpp         = zeros(N, 1);
vpp(1:N-1) = v0(2:N) - dx/2 * v0x(2:N);
vpp(N)      = bc2(t)      - dx/2 * v0x_bc2;

% Calculating a_{j+1/2}
ap = max(abs(dF(x_ph, vpp)), abs(dF(x_mh, vmp)));

% Calculating v0^{-}_{j-1/2}
v0x_bc1 = v0x(2) - (s_min-dx - (s_min+dx))/(s_min - (s_min+dx))*(v0x(1) - v0x(2));
vmm       = zeros(N, 1);
vmm(1)    = bc1(t)        + dx/2 * v0x_bc1;
vmm(2:N) = v0(1:N-1) + dx/2 * v0x(1:N-1);

% Calculating v0^{+}_{j-1/2}
vpm = v0 - dx/2 * v0x;

% Calculating a_{j+1/2}
am = max(abs(dF(x_ph, vpm)), abs(dF(x_mh, vmm)));

% H_{j+1/2}
Hp = 0.5*(F(x_ph, vpp) + F(x_ph, vmp)) - 0.5*ap.*(vpp - vmp);

% H_{j-1/2}
Hm = 0.5*(F(x_mh, vpm) + F(x_mh, vmm)) - 0.5*am.*(vpm - vmm);

H = -1/dx * (Hp - Hm);

% Diffusive Flux -------------------------------------------------------------
% Forward-type numerical derivative
```

```
v0x_fwd = derivative_fwd(v0, dx, bc1(t), bc2(t));

% Backwards-type numerical derivative
v0x_bwd = derivative_bwd(v0, dx, bc1(t), bc2(t));

% P_{j+1/2}
Pp = Q(x_ph, v0, v0x_fwd);

% P_{j-1/2}
Pm = Q(x_mh, v0, v0x_bwd);

P = 1/dx * (Pp - Pm);

% Output
v1 = H + P + S(x, v0);
```

A.3. Minmod Derivative

```
function vx = derivative_minmod(v, dx, theta, bc1, bc2)
% Obtains the derivative of v based on minmod limiter defined in the
% function minmod.m

N = length(v);

vx = zeros(N, 1);

% Derivative at j = 1
vx(1) = minmod( ...
        theta*(v(1) - bc1)/dx, ...
        (v(2) - bc1)/(2*dx), ...
        theta*(v(2) - v(1))/dx);

% Derivative at j = N
vx(N) = minmod( ...
        theta*(v(N) - v(N-1))/dx, ...
        (bc2 - v(N-1))/(2*dx), ...
        theta*(bc2 - v(N))/dx);

% Derivative for the rest of the grid points
for i = 2:N-1
   vx(i) = minmod(theta*(v(i) - v(i-1))/dx, ...
                  (v(i+1) - v(i-1))/(2*dx), ...
                  theta*(v(i+1) - v(i))/dx);
end
```

A.4. Minmod Limiter

```
function d = minmod(a, b, c)
% Minmod limiter as defined in Kurganov-Tadmor [KT00]
```

```
if a > 0 && b > 0 && c > 0
    d = min([a, b, c]);
elseif a <0 && b < 0 && c <0
    d = max([a, b, c]);
else
    d = 0;
end
```

A.5. Backwards Derivative

```
function vx = derivative_bwd(v, dx, bc1, bc2)
% Derivative of a function of one variable, Backwards-type
%        vx(i) = (v(i) - v(i-1)) / dx;
% The vector returned is the same size as the input. The first approximation
% to the derivative is achieved with forward-type numerical derivative.


vx = zeros(size(v));

vx(1) = (v(1) - bc1) / dx;

vx(2:end) = (v(2:end) - v(1:end-1)) / dx;
```

A.6. Forward Derivative

```
function vx = derivative_fwd(v, dx, bc1, bc2)
% Derivative of a function of one variable, forward-type
%        vx(i) = (v(i+1) - v(i)) / dx;
% The vector returned is the same size as the input.

vx = zeros(size(v));

vx(1:end-1) = (v(2:end) - v(1:end-1)) / dx;

vx(end) = (bc2 - v(end)) / dx;
```

A.7. Code for MEX compiler

When reading this code, please keep in mind that Matlab relies on numerical libraries written in Fortran, therefore the arrays are ordered column-wise. Also, the two-dimensional matrices are represented as one-dimensional array with $m \times n$ elements where m is the number of rows and n the number of columns.

```
#include <math.h>
#include "mex.h"

/* Input Arguments */
#define U_IN      prhs[0]
#define DX_IN     prhs[1]
#define THETA_IN  prhs[2]
#define BC1_IN    prhs[3]
#define BC2_IN    prhs[4]
#define DIM_IN    prhs[5]


/* Output Arguments */
#define UX_OUT  plhs[0]

/* Some Constants */
#define Y_DIM 1
#define X_DIM 2

#if !defined(MAX)
#define MAX(A, B)   ((A) > (B) ? (A) : (B))
#endif

#if !defined(MIN)
#define MIN(A, B)   ((A) < (B) ? (A) : (B))
#endif

static double minmod(double a, double b, double c);

static void minmod_derivative(
        double  ux[],
        double  u[],
        double  dx,
        double  theta,
        double  bc1[],
        double  bc2[],
        int     dim,
        int     m,
        int     n)
{

    /* Indexers */
    int i, j;

    /* Auxiliary variables */
    double a, b, c;

    /* Auxiliar variable to represent current index */
    int ci;

    if (dim == X_DIM) {

        /* Values at the boundaries */
        for (i = 0; i < m; i++) {

            /* left boundary */
            a = theta*(u[i] - bc1[i]) / dx;
            b = (u[i+m] - bc1[i]) / (2*dx);
            c = theta*(u[i+m] - u[i]) / dx;

            ux[i] = minmod(a, b, c);
```

```
            /* right boundary */
            ci = i + m*n - m; /* Current desired position*/
            a = theta*(u[ci] - u[ci-m]) / dx;
            b = (bc2[i] - u[ci-m]) / (2*dx);
            c = theta*(bc2[i] - u[ci]) / dx;
            ux[ci] = minmod(a, b, c);
        }

        for (j = 1; j < (n-1); j++) {
            for (i = 0; i < m; i++) {

                /* Current desired position*/
                ci = i + j*n;

                a = theta*(u[ci] - u[ci - m]) / dx;
                b = (u[ci+m] - u[ci-m]) / (2*dx);
                c = theta*(u[ci+m] - u[ci]) / dx;;

                ux[ci] = minmod(a, b, c);
            }
        }

      } else if (dim == Y_DIM) {

        /* no available BC in this direction, an approximation is achieved
         * with appropriate available stencil */
        for (j = 0; j < n; j++) {
            ci = j*m;
            ux[ci] = (u[ci+1] - u[ci]) / dx;
            ux[ci+m-1] = (u[ci+m-1] - u[ci+m-2]) / dx;
        }

        for (j = 1; j < n; j++) {
            for (i = 1; i < (m-1); i++) {

                ci = i + j*n;

                a = theta*(u[ci] - u[ci-1]) / dx;
                b = (u[ci+1] - u[ci-1]) / (2*dx);
                c = theta*(u[ci+1] - u[ci]) / dx;

                ux[ci] = minmod(a, b, c);
            }
        }

    } else {
        mexWarnMsgTxt("Wrong Dimension\n");
    }

    return;

}

static double minmod(double a, double b, double c) {

    double d;

    if(a>0 && b>0 && c>0) {
        d = MIN(MIN(a, b), c);
    } else if (a<0 && b<0 && c<0) {
        d = MAX(MAX(a, b), c);
    } else {
```

```
        d = 0;
    }

    return d;
}

void mexFunction( int nlhs, mxArray *plhs[], int nrhs, const mxArray *prhs[]) {

    /* This variable will contain the derivative of the function */
    double *ux;

    /* input parameters */
    double *u, *dx, *theta, *bc1, *bc2, *dim;

    /* size of the matrix */
    mwSize m, n;


    /* Check for proper number of arguments */
    if (nrhs != 6) {
        mexErrMsgTxt("Function requires six input arguments.");
    } else if (nlhs > 1) {
        mexErrMsgTxt("Too many output arguments.");
    }

    /* Size of the matrix */
    m = mxGetM(U_IN);
    n = mxGetN(U_IN);

    /* Create a matrix for the return argument */
    UX_OUT = mxCreateDoubleMatrix(m, n, mxREAL);

    /* Assign pointers to the various parameters */
    ux = mxGetPr(UX_OUT);

    u     = (double*)mxGetPr(U_IN);
    dx    = (double*)mxGetPr(DX_IN);
    theta = (double*)mxGetPr(THETA_IN);
    bc1   = (double*)mxGetPr(BC1_IN);
    bc2   = (double*)mxGetPr(BC2_IN);
    dim   = (double*)mxGetPr(DIM_IN);


    /* Do the actual computations in a subroutine */
    minmod_derivative(ux, u, *dx, *theta, bc1, bc2, (int)(*dim), m, n);
    return;
}
```