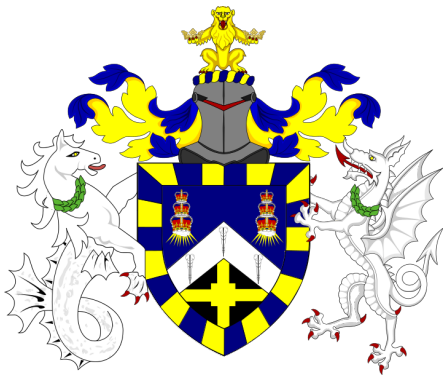
Mathematical Finance MSc Dissertation MTH775P, 2018/19

Accelerated Grids

Optimizing Solvers for Financial Partial Differential Equations

Mustafa Berke Erdis, ID 180883925

Supervisor: Dr. Sebastian del Bano Rollin



A thesis presented for the degree of
Master in Sciences in *Mathematical Finance*

School of Mathematical Sciences
and *School of Economics and Finance*
Queen Mary University of London

Declaration of original work

This declaration is made on August 9, 2019.

Student's Declaration: I, Mustafa Berke Erdis, hereby declare that the work in this thesis is my original work. I have not copied from any other students' work, work of mine submitted elsewhere, or from any other sources except where due reference or acknowledgement is made explicitly in the text, nor has any part been written for me by another person.

Referenced text has been flagged by:

1. Using italic fonts, **and**
2. using quotation marks "...", **and**
3. explicitly mentioning the source in the text.

This work is dedicated to my family.

Acknowledgements

Here you thank people that have helped you in the journey.
Vivamus vehicula leo a justo. Quisque nec augue. Morbi mauris wisi, aliquet vitae, dignissim eget, sollicitudin molestie, ligula. In dictum enim sit amet risus. Curabitur vitae velit eu diam rhoncus hendrerit. Vivamus ut elit. Praesent mattis ipsum quis turpis. Curabitur rhoncus neque eu dui. Etiam vitae magna. Nam ullamcorper. Praesent interdum bibendum magna. Quisque auctor aliquam dolor. Morbi eu lorem et est porttitor fermentum. Nunc egestas arcu at tortor varius viverra. Fusce eu nulla ut nulla interdum consectetuer. Vestibulum gravida. Morbi mattis libero sed est.

Abstract

Here you write a short summary, around 10 lines, of your work.
Vivamus vehicula leo a justo. Quisque nec augue. Morbi mauris wisi, aliquet vitae, dignissim eget, sollicitudin molestie, ligula. In dictum enim sit amet risus. Curabitur vitae velit eu diam rhoncus hendrerit. Vivamus ut elit. Praesent mattis ipsum quis turpis. Curabitur rhoncus neque eu dui. Etiam vitae magna. Nam ullamcorper. Praesent interdum bibendum magna. Quisque auctor aliquam dolor. Morbi eu lorem et est porttitor fermentum. Nunc egestas arcu at tortor varius viverra. Fusce eu nulla ut nulla interdum consectetuer. Vestibulum gravida. Morbi mattis libero sed est.

Preface

Here you write a summary of the work. A paragraph on the motivation, previous work, then maybe a brief chapter by chapter summary.

Vivamus vehicula leo a justo. Quisque nec augue. Morbi mauris wisi, aliquet vitae, dignissim eget, sollicitudin molestie, ligula. In dictum enim sit amet risus. Curabitur vitae velit eu diam rhoncus hendrerit. Vivamus ut elit. Praesent mattis ipsum quis turpis. Curabitur rhoncus neque eu dui. Etiam vitae magna. Nam ullamcorper. Praesent interdum bibendum magna. Quisque auctor aliquam dolor. Morbi eu lorem et est porttitor fermentum. Nunc egestas arcu at tortor varius viverra. Fusce eu nulla ut nulla interdum consectetuer. Vestibulum gravida. Morbi mattis libero sed est.

Queen Mary University of London
12$^{\text{th}}$ August 2019

Contents

Chapter 1

Introduction

In Ancient Greece, Thales was scorned for his poverty. Later that year, Thales utilized his skills in astrology. He forecasted the increase in olive yields. Using his limited capital, he rented oil presses in winter. Months lates, over the oil making season, many people rushed to the presses because of the high yields that Thales predicted. As he rented the presses over the winter, he forced the terms he pleased. Thales showed it was easy for philosophers to be rich if they chose it and practically used the first financial derivative product. [1]

In the modern world, financial derivatives are contracts between two or more parties. The value of the contract depends on one or several underlying assets. Commonly the assets are currencies, equities, bonds, interest rates, market indices or commodities. The vanilla call option gives the right but not the obligation to buy the underlying asset at the expiry date at a previously agreed strike price. On the other hand, the vanilla put option gives the right but not the obligation to sell the underlying asset at the expiry date at a previously agreed strike price. Essentially, Thales bought call options for oil presses. So if the olive yields didn't come as Thales expected he didn't have the obligation to use the olive presses.

1.1 Pricing Financial Derivatives

1.1.1 The Risk Neutral Approach

Definition 1.1.1. Brownian Motion

Brownian motion (also known as Wiener Process) was discovered by botanist Robert Brown as he observed a chaotic motion of particles suspended in water. [10] A Brownian motion, $B(t)$, is a continuous-time stochastic process with the following properties:

- $B(0) = 0$.
- $B(t)$ is a continuous function of t.
- For $0 \leq s < t$ the increment $B(t) - B(s)$ has normal distribution $\mathcal{N}(0, t-s)$.

Brownian motion is the basic building block in stochastic calculus and geometric Brownian motion is often used to model the stock prices.

Theorem 1.1.2. *Itô's Lemma*

Let $B(t)$ be a Brownian motion and $W(t)$ be an Ito drift-diffusion process which satisfies the stochastic differential equation:

$$dW(t) = \mu(W(t), t)dt + \sigma(W(t), t)dB(t) \tag{1.1}$$

If $f(w, t) \in C^2(\mathbb{R}^2, \mathbb{R})$ then $f(W(t), t)$ is also an Ito drift-diffusion process, with its differential given by:

$$d(f(W(t), t)) = \frac{\partial f}{\partial t}(W(t), t)dt + f'(W(t), t)dW + \frac{1}{2}f''(W(t), t)dW(t)^2 \tag{1.2}$$

With $dW(t)^2$ given by: $dt^2 = 0$, $dtdB(t) = 0$ and $dB(t)^2 = dt$.

Theorem 1.1.3. *Black-Scholes Model*

The Black-Scholes framework is a theoretical valuation formula for options. Since almost all corporate liabilities can be viewed as combinations of options, the formula is applicable to common stock, corporate bonds etc. [2] Under the assumptions of Black-Scholes framework, the call or put option price satisfies the parabolic partial differential equation:

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} = r(V - S\frac{\partial V}{\partial S}) \tag{1.3}$$

Proof. Suppose an investor sets up a self-financing portfolio at time t, with a total value $X(t)$. The portfolio consists of:

- $\Delta(t)$ share of stocks
- Remaining $X(t) - \Delta(t)S(t)$ deposit in the riskless bank account which yields continuously compounded interest at rate r.

Self-financing trading strategy has no capital influx or consumption, the value of portfolio changes as:

$$dX(t) = \Delta(t)dS(t) + r(X(t) + \Delta(t)S(t))dt \tag{1.4}$$

Black-Scholes model assumes that the stock price under the "market probability measure" follows a gBM.

$$S(t) = S(0)exp((\alpha - \sigma^2/2)t + \sigma B(t)), t \geq 0 \tag{1.5}$$

where $\sigma > 0$ is a constant volatility parameter, $\alpha > 0$ is a mean rate of return. In differential form:

$$dS(t) = \alpha S(t)dt + \sigma S(t)dB(t) \tag{1.6}$$

Putting (1.4) and (1.6) together yields

$$dX(t) = rX(t)dt + (\alpha - r)\Delta(t)S(t)dt + \sigma\Delta(t)S(t)dB(t) \tag{1.7}$$

The first term denotes the return on investment X(t) in bank, second term is the risk premium and third term appears due to the volatility of the stock market.

In order to calculate the differential for the discounted portfolio value $exp(-rt)X(t)$, we use the Itô's formula.

$$d(exp(-rt)X(t)) = -rexp(-rt)X(t)dt + exp(-rt)dX(t) \tag{1.8}$$

Substituting $dX(t)$ from (1.7)

$$(\alpha - r)\Delta(t)S(t)dt + \sigma\Delta(t)S(t)dB(t) \tag{1.9}$$

□

Remark 1.1.4. 1 sayfa anlat bunu The Black-Scholes model is still used but not entirely applicable to assets that cannot be hedged. Multi asset derivatives(equity baskets), Weather derivatives, non tradable assets, not freely tradable FX (Brazil real, Korean Won). The industry uses a different product called non deliverable forward for such.

1.1.2 Solving Financial Partial Differential Equations

Since the foundation of the world humanity tried to understand and model the nature. Differential equations serves this purpose by enabling us to describe natural phenomena for instance, heat, sound and fluid flow. The mathematical theory of partial differential equations describing financial markets plays an important role in mathematical finance.

General form of 2nd order PDEs of two independent variables is

$$au_{xx} + bu_{xy} + cu_{yy} + du_x + eu_y + fu + g = 0$$

If the equation satisfies the condition $b^2 - 4ac = 0$ it is considered a parabolic partial differential equation. Generally, financial partial differential equations

can be classified as parabolic partial differential equations.

Sometimes there are analytical solutions derive analytical bs/separation of variables heat equation Vanilla options barrier options reference (1 page)

Numerical partial differential equations is a large area of study. The subject includes components in the areas of applications, mathematics and computers. These three aspects of a problem are so strongly tied together that it is virtually impossible to consider the applied aspect of a problem without considering at least some of the mathematical counting aspects of that problem.[j.w. thomas]

On the other hand, mostly partial differential equations are too complicated to work out an analytical solution. Thus, we need to achieve a numerical solution to the problem.

The most common framework is finite difference method which tries to find approximate solutions to the problem at a discrete set of points, normally on a rectangular grid of points. Instead we try to find approximate solutions of the problem at a discrete set of points in the (x, t) plane, normally a rectangular grid of points. It is simple to construct and analyse but can compromise performance because of increased computational complexity when there are high dimensions.

But sometimes we need to use numerical methods Monte carlo high dimensions, Grids low dimensions show feynman kac writes as expectation Glassermann MC book rough calculation of error MC vs PDE grids(1-2 pages) Finally, the Monte Carlo method is used to find the numerical solution when dimensions are too high by calculating an expectation (Feynman-Kac Theorem) klebaner.

1.2 Finite Difference Methods

Parabolic partial differential equation can be denoted as

$$\frac{\partial u}{\partial t} = a(t,x)\frac{\partial^2 u}{\partial x^2} + b(t,x)\frac{\partial u}{\partial x} + c(t,x)u(t,x) + d(t,x)$$

$a(t,x)$ denotes diffusion coefficient, $b(t,x)$ convection coefficient, $c(t,x)$ reaction coefficient, $d(t,x)$ source coefficient.

1st order Central Difference

$$\frac{\partial u}{\partial x} = \frac{u_{i+1}^n - u_i^n}{\Delta t}$$

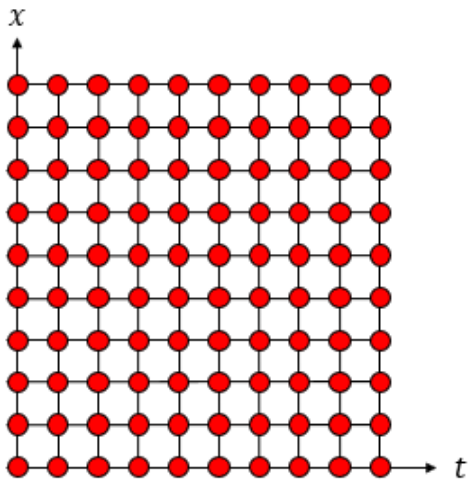


Figure 1.1: 10 x 10 grid

Utilizing the boundary conditions and initial conditions:

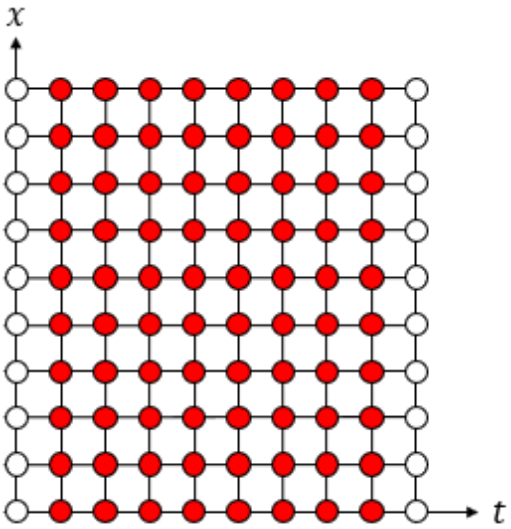


Figure 1.2: Boundary conditions.

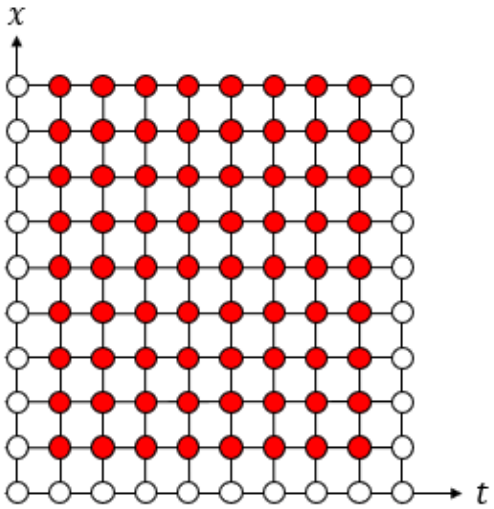


Figure 1.3: Initial conditions.

1.2.1 Explicit Method

Explicit method is a forward time, central space scheme . 1st order Forward Difference

$$\frac{\partial u}{\partial t} = \frac{u_i^{n+1} - u_i^n}{\Delta t}$$

2nd order Central Difference

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{(\Delta x)^2}$$

The parabolic partial differential equation can be generalized by applying the forward difference to the time derivative and the centred second difference.

$$u_j^{n+1} = \alpha u_{j-1}^n + \beta u_j^n + \gamma u_{j+1}^n$$

For heat equation the coefficients become

$$\alpha = r, \beta = 1 - 2r, \gamma = r$$

Black-Scholes partial differential equation the coefficients become:

$$\alpha = \frac{\sigma^2 j^2 \Delta t}{2} - \frac{rj\Delta t}{2}, \beta = 1 - \sigma^2 j^2 \Delta t - r\Delta t, \gamma = \frac{\sigma^2 j^2 \Delta t}{2} - \frac{rj\Delta t}{2}$$

where $r = \frac{\delta t}{\delta x^2}$.

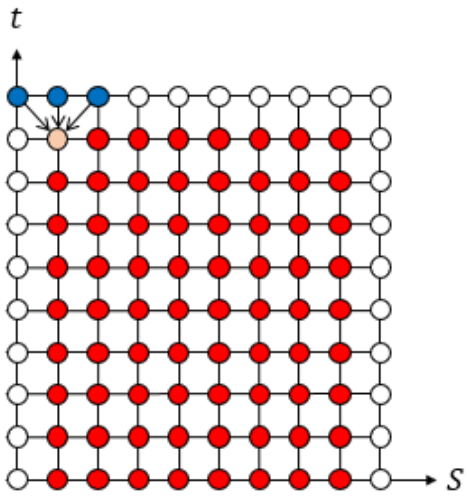


Figure 1.4: Computational stencil of explicit method

1.2.2 Crank-Nicholson Method

The crank nicolson method was introduced [3]

$$u(t,x) \approx \frac{1}{2}(u_i^{n+1} + u_i^n)$$

$$\frac{\partial u}{\partial t} \approx \frac{u_i^{n+1} - u_i^n}{\Delta t}$$

$$\frac{\partial u}{\partial x} \approx \frac{u_{i+1}^n - u_{i-1}^n + u_{i+1}^{n+1} - u_{i-1}^{n+1}}{4\Delta x}$$

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n + u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{2(\Delta x)^2}$$

$$A = a(t,x)\frac{\Delta t}{\Delta x^2}, B = b(t,x)\frac{\Delta t}{4\Delta x}, C = c(t,x)\frac{\Delta t}{2}, D = d(t,x)\Delta t$$

$$\begin{aligned}(-A-B)u_{i+1}^{n+1} + (1+2A-C)u_i^{n+1} + (-A+B)u_{i-1}^{n+1} = \\ = (A+B)u_{i+1}^n + (1-2A+C)u_i^n + (A-B)u_{i-1}^n + D \quad (1.10)\end{aligned}$$

The left hand side groups the unknowns and the right hand side groups knowns. The system of equations can be represented by a tridiagonal matrix system. Most commonly these system are solved by Thomas Algorithm.

Definition 1.2.1. Thomas Algorithm
The method is used to solve a tridiagonal matrix system invented by Llewellyn Thomas [11]. The system equations can be written as

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & ... & 0 \\ a_2 & b_2 & c_2 & 0 & ... & 0 \\ 0 & a_3 & b_3 & c_3 & 0 & 0 \\ . & . & & & & . \\ . & . & & & & . \\ . & . & & & & c_{k-1} \\ 0 & 0 & 0 & 0 & a_k & b_k \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ . \\ . \\ . \\ f_k \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ . \\ . \\ . \\ d_k \end{bmatrix}$$

The method begins by forming coefficients c_i^* and d_i^* in place of a_i, b_i and c_i as follows:

$$c_i^* = \begin{cases} \frac{c_1}{b_1} & ; i = 1 \\ \frac{c_i}{b_i - c_{i-1}^* a_i} & ; i = 2, 3, ..., k-1 \end{cases}$$

$$d_i^* = \begin{cases} \frac{d_1}{b_1} & ; i = 1 \\ \frac{d_i - d_{i-1}^* a_i}{b_i - c_{i-1}^* a_i} & ; i = 2, 3, ..., k-1 \end{cases}$$

With these new coefficients, the matrix equation can be rewritten as:

$$\begin{bmatrix} 1 & c_1^* & 0 & 0 & ... & 0 \\ 0 & 1 & c_2^* & 0 & ... & 0 \\ 0 & 0 & 1 & c_3^* & 0 & 0 \\ . & . & & & & . \\ . & . & & & & . \\ . & . & & & & c_{k-1}^* \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ . \\ . \\ . \\ f_k \end{bmatrix} = \begin{bmatrix} d_1^* \\ d_2^* \\ d_3^* \\ . \\ . \\ . \\ d_k^* \end{bmatrix}$$

The algorithm for the solution of these equations is now straightforward and works 'in reverse':

$$f_k = d_k^*, \qquad f_i = d_k^* - c_i^* x_{i+1}, \qquad i = k-1, k-2, ..., 2, 1$$

1.2.3 Rannacher Trick

Using the Crank nicholson

1.2.4 Alternating Direction Implicit Method

The alternate direction implicit (ADI) method is used to numerically solve two dimensional parabolic PDEs. ADI schemes give us advantages of implicit finite difference method and computationally only requires to solve tridiagonal matrices.

ADI Peaceman and Rachford in 1955 [8]

Operator splitting is a natural and old idea. When a PDE or system of PDEs contains different terms expressing different physics, it is natural to use different numerical methods for different physical processes. This can op-

timize and simplify the overall solution process. The idea was especially popularized in the context of the Navier-Stokes equations and reaction-diffusion PDEs. Common names for the technique are operator splitting, fractional step methods, and split-step methods. We shall stick to the former name. In the context of nonlinear differential equations, operator splitting can be used to isolate nonlinear terms and simplify the solution methods. [4] A related technique, often known as dimensional splitting or alternating direction implicit (ADI) methods, is to split the spatial dimensions and solve a 2D or 3D problem as two or three consecutive 1D problems, but this type of splitting is not to be further considered here.

$$\delta x^2 u_{i,j}^n = \frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{\Delta x^2} \tag{1.11}$$

$$\frac{u_{i,j}^{n+1} + u_{i,j}^n}{\Delta t} = \delta x^2 u_{i,j}^n + \delta y^2 u_{i,j}^n \tag{1.12}$$

$$\frac{u_{i,j}^{n+1} + u_{i,j}^n}{\Delta t} = \delta x^2 u_{i,j}^{n+1} + \delta y^2 u_{i,j}^{n+1} \tag{1.13}$$

Divide each time step in half Implicit x, Explicit y:

$$\frac{u_{i,j}^{n+1/2} + u_{i,j}^n}{0.5\Delta t} = \frac{\delta x^2 u_{i,j}^{n+1/2}}{\Delta x^2} + \frac{\delta y^2 u_{i,j}^n}{\Delta y^2} \tag{1.14}$$

Explicit x, Implicit y:

$$\frac{u_{i,j}^{n+1} + u_{i,j}^{n+1/2}}{0.5\Delta t} = \frac{\delta x^2 u_{i,j}^{n+1/2}}{\Delta x^2} + \frac{\delta y^2 u_{i,j}^{n+1}}{\Delta y^2} \tag{1.15}$$

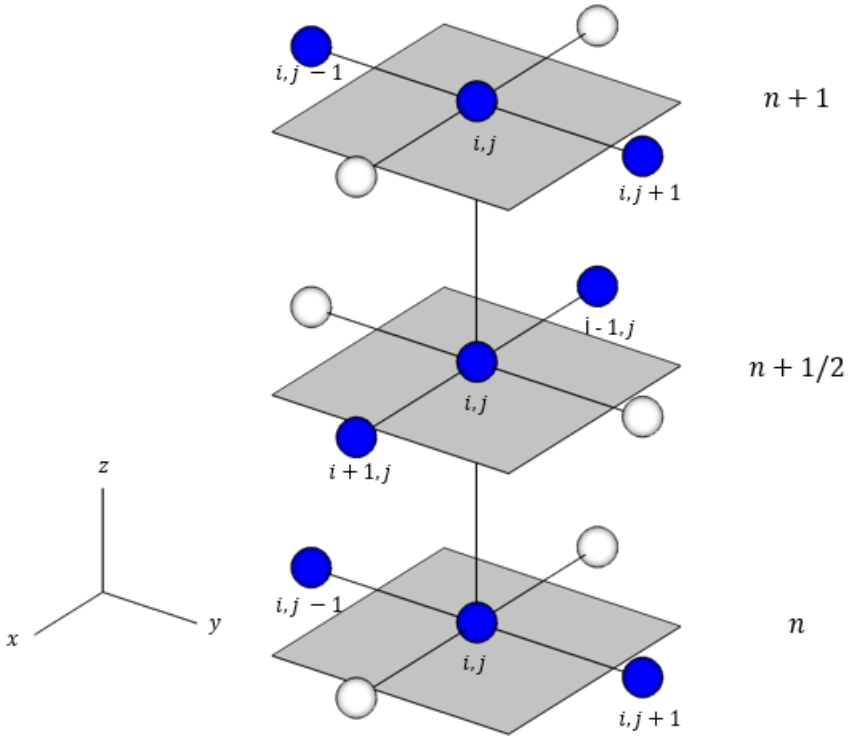


Figure 1.5: Computational stencil of alternating direction implicit method

1.3 Optimizations

Derivative pricing in the real world is a computationally intensive task. The existing numerical methods for partial differential equations are all constrained by the computational complexity. Being fast when evaluating new information is critical for the operations of hedge funds and investment banks. referansbul Therefore optimizing the existing numerical methods with hardware and software that can be installed on a trading floor is crucial. Main optimization techniques that can be implemented can be listed as follows.

1.3.1 Compilers

Intel Compiler, VS Compiler, gcc

1.3.2 32 bit and 64 bit

1.3.3 Optimization Switches

1.3.4 Tridiagonal Solvers

Thomas Algorithm, cyclic reduction double sweep.

1.3.5 Threading

High performance computing techniques that can be implemented for multi-threading with Open Multi-Processing(OpenMP) and compiler intrinsics.

1.3.6 OpenMP

1.3.7 AVX and Intrinsics

CPUs are pipelining and use of SSE/SIMD kusswurm registers with Advanced Vector Extensions(AVX 512),

1.3.8 GPGPU

In the case of General Purpose GPUs, CUDA or Open Computing Language(OpenCL) can be utilized but can be challenging because of the requirement of delicate memory management.

1.3.9 Cloud Applications

Chapter 2

Optimizing Solvers

Numerical analysis and computer simulations will be undertaken to put theory and observation together to gain insight into the workings of numerical solutions of partial differential equations. First step was deriving toy examples that can be calculated by hand and Excel. Following the verifications, porting the toy examples in C++ and utilize high performance computing principles.

2.1 Heat Equation

Most basic case for parabolic differential equations is the heat equation. In the experiments the following initial and boundary conditions where $T = 0.075$ and $x_{max} = 1.0$ will be used.

$$u_t(x,t) = u_{xx}(x,t) \tag{2.1}$$

$$u(0,t) = u(x_{max},t) = 0, \quad 0 \leq t \leq T \tag{2.2}$$

$$u(x,0) = sin(\pi x), \quad 0 \leq x \leq x_{\max} \tag{2.3}$$

2.1.1 Analytical Solution of Heat Equation

Certain kinds of partial differential equations allows us to find an analytical solution with the help of the Separation of Variables technique [5] .

$$u(x,t) = X(x)T(t) \tag{2.4}$$

$$u_{xx}(x,t) = X^{''}(x)T(t) \tag{2.5}$$

$$u_t(x,t) = X(t)T^{'}(t) \tag{2.6}$$

Using the partial derivatives the equation $u_t = u_{xx}$ becomes

$$\frac{T^{'}(t)}{T(t)} = \frac{X^{''}(x)}{X(x)} \tag{2.7}$$

Right hand side only depends on x and the left hand side depends only on t. Therefore, the equation is valid only when each side is equal to a constant, which we set to . Rearranging terms gives us the following equations:

$$\frac{T^{'}(t)}{T(t)} = \frac{X^{''}(x)}{X(x)} = \lambda \tag{2.8}$$

$$X^{''}(x) = \lambda X(x), \quad T^{'}(t) = \lambda T(t) \tag{2.9}$$

$$X(0) = X(1) = 0 \tag{2.10}$$

Solving for X(x) is an example case of Sturm-Liouville problem [7]. However the $\lambda < 0$ and $\lambda = 0$ cases result in trivial solutions, thus they are discarded. Solving for $\lambda > 0$ yields

$$X(x) = c_1 sin(kx) + c_2 cos(kx) \tag{2.11}$$

The boundary conditions leads to $c_2 = 0, c_1 sin(k) = 0 \rightarrow k = 0, \pi, 2\pi, ...n\pi$ where n is an integer.

Solving for $T(t)$ gives the solution

$$T(t) = c_3 exp(-n^2\pi^2 t) \tag{2.12}$$

$$u(x,t) = \sum_{n=1}^{\infty} b_n exp(-n^2\pi^2 t) sin(n\pi x) \tag{2.13}$$

where we have set $b_n = c_1 c_3$. The initial condition gives

$$u(x,0) = sin(\pi x) = \sum_{n=1}^{\infty} b_n sin(n\pi x) \tag{2.14}$$

which is a Fourier sine series. Thus, the coefficient b_n is chosen such that

$$b_n = 2\int_0^1 sin(\pi x) sin(\pi n x) dx = \frac{2sin(\pi n)}{\pi - \pi n^2} \tag{2.15}$$

Combining the solutions

$$\sum_{n=1}^{\infty} \frac{2sin(\pi n)}{\pi - \pi n^2} exp(-n^2\pi^2 t) sin(n\pi x) = exp(-\pi t) sin(\pi x) \tag{2.16}$$

Analytical solution is utilised to calculate errors for the solutions using different grid sizes, the grid with the lowest error is used for further tests. Figure 2.1 demonstrates the error compared to the grid sizes. The optimal grid size for this specific problem is 50 by 50.

Grid size depends on the parameters prove this duffy book.

After choosing the optimal grid size the resulting grids can be visualized as

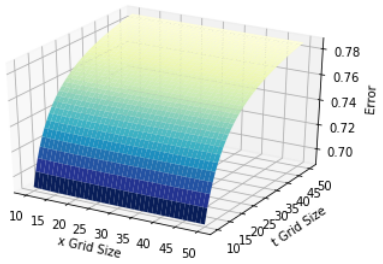


Figure 2.1: Error of grid sizes.

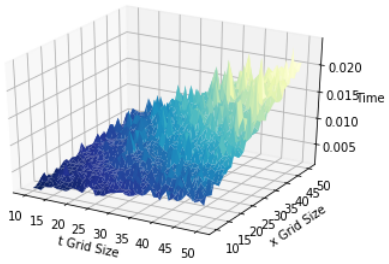


Figure 2.2: Timing grid sizes.

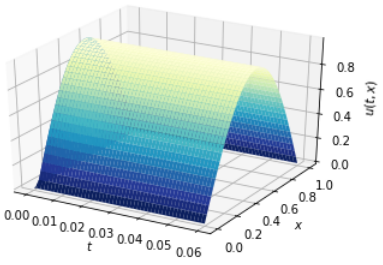


Figure 2.3: Explicit method.

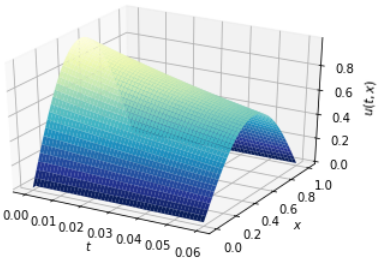


Figure 2.4: Crank nicolson method.

2.2 Black-Scholes

Black-Scholes model for call option

$$\frac{\partial C}{\partial t} = rS\frac{\partial C}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} - rC \tag{2.17}$$

$$C(0,t) = 0,\ C(S_{\max},t) = S_{\max} - Ke^{-r(T-t)}, \quad 0 \le t \le T \tag{2.18}$$

$$C(S,T) = \max(S-K,0), \quad 0 \le S \le S_{\max} \tag{2.19}$$

Parameter	Value
Strike Price	1.0
Volatility	20 %
Risk Free Rate	5 %
Time to Expiry	2.0
Maximum Share Price	2.0

Table 2.1: Black-Scholes model testing parameters.

2.2.1 Analytical Solution

Derive analytical solution optimal grid size

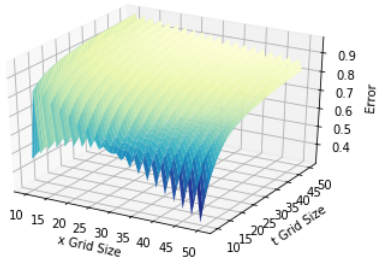


Figure 2.5: Explicit method.

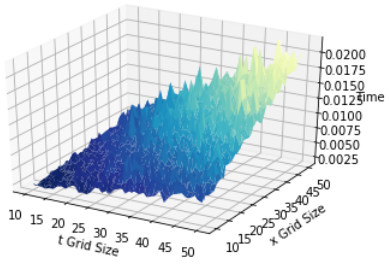


Figure 2.6: Crank nicolson method.

After choosing the optimal grid size the resulting grids can be visualized as

2.3 2 Dimensional Heat Equation Base Case

2 Dimensional Heat Equation to test ADI method:

$$\frac{\partial f}{\partial t} = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{2.20}$$

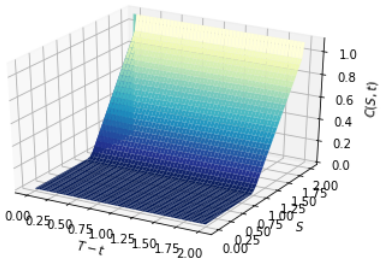


Figure 2.7: Explicit method.

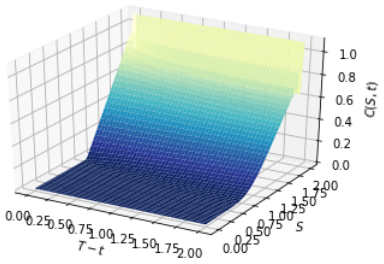


Figure 2.8: Crank nicolson method.

Initial Condition

$$f(x, y, 0) = 1 \tag{2.21}$$

Boundary Conditions

$$f(x, 0, t) = f(x, 1, t) = 0 \tag{2.22}$$

$$f(x, 0, t) = f(x, 1, t) = 0 \tag{2.23}$$

2.4 Timing the Code

Measuring execution time intervals accurately is an important aspect to compare the efficiency and speed of different environments and implementations.

2.4.1 Windows Application Programming Interface

Windows Application Programming Interface (API) is the lowest level of interaction between applications and the windows operating system. Thus every program is built upon the API. Mostly, the interaction is hidden, the runtime and support libraries manage it in the background. [9]

The APIs can be easily used in C++ environment. Timing can be achieved by "QueryPerformanceCounter" or "QueryPerformanceFrequency"

functions. Respectively, they retrieve a high resolution time stamp and the frequency of the performance counter. The functions can be easily utilized for time measurements.

2.4.2 Chrono Library

Using the Windows API for just timing the code is slightly excessive given the amount of work it takes. Luckily, Chrono library was introduced part of the C++11s standard library. Timers and clocks might differ on distinct systems, thus Chrono library is designed to work effortlessly with date and time.

The "high resolution clock" provides the smallest possible tick period and with the now method, returns a value corresponding to the calls point in time. Once the start and end time of the code is recorded , the duration::count method is used to get the elapsed time.

2.5 Optimization Experiments

This section documents the performance of attempted optimizations. Experiments are conducted at W307 computer laboratory, Queen Mary University of London. Each computer has Windows 10 Enterprise 64 bit, 16 GB of RAM, Intel Core i7-6700 CPU with 4 cores clocked at 3.40 GHz.

The source code is written in C++ and compiled with Microsoft Visual Studio Enterprise 2017, Version 15.3.3 in the release mode. External tools utilized in the tests include Intel Compiler, version 18.0.3 and Intel Math Kernel Library, version X .

2.5.1 Tridiagonal Solvers

While using the Crank Nicolson and Alternating Direction Implicit methods the computationally intensive part of program is solving tridiagonal systems.

Previously defined Thomas Algorithm is the most commonly used method 1.2.1.

Intel Math Kernel Library implements routines for solving systems of linear equations from the standard LAPACK library. Variety of matrix types are supported by the routines. Specifically gtsv function is utilized from the package. Using Gaussian elimination with partial pivoting, gstv computes the solution to the system of linear equations with a tridiagonal coefficient matrix. [6]

2.5.2 Compilers and Solution Platforms

32 bit vs 64 bit VS vs Intel compiler

Environment	Explicit Method	Crank Nicolson
Visual Studio Compiler x86	0.022081	0.02035
Visual Studio Compiler x64	0.018627	0.01888
Intel Compiler x86	0.023165	0.023723
Intel Compiler x64	0.019535	0.017451

Table 2.2: Averaging 1000 timings for Black-Scholes.

Testing	Explicit Method Timing	Crank Nicolson Timing
Visual Studio Compiler x86	0.022625	0.022441
Visual Studio Compiler x64	0.018933	0.016687
Intel Compiler x86	0.023076	0.023719
Intel Compiler x64	0.017271	0.017271

Table 2.3: Averaging 1000 timings for Black-Scholes and adding random $\epsilon < 10^{-7}$ to step size in order to avoid compiler optimizations.

In practical terms, you would use X64 when: You need to directly address more than 4GB of memory, or You need very fast (native) processing of 64 bit numerical quantities (including double-precision floating-point numbers)

X86 (32 bit) is suitable for most everything else.

2.5.3 Visual Studio Optimization Switches

Chapter 3

Conclusions

Appendix A

Usage of chrono class

Should code example be in appendix or stay here?

```
auto start = std::chrono::high_resolution_clock::now();
    Portion of code to be timed
auto finish = std::chrono::high_resolution_clock::now();
std::chrono::duration<double> elapsed = finish - start;
std::cout << "Elapsed time: " << elapsed.count() << " s\n";
```

Appendix B

Implementation of the `FiniteDifferenceMethod` class

Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dictum tortor, vel consectetuer odio sem sed wisi.

Appendix C

Additional details on the Gundermanian determinant

Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dictum tortor, vel consectetuer odio sem sed wisi.

Bibliography

[1] Saeed Amen. *Trading Thalesians: What the Ancient World Can Teach Us About Trading Today*, pages 39–60. Palgrave Macmillan UK, London, 2014.

[2] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973.

[3] J. Crank and P. Nicolson. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Advances in Computational Mathematics*, 6(1):207–226, Dec 1996.

[4] Jim Douglas, Jr. Alternating direction methods for three space variables. *Numer. Math.*, 4(1):41–63, December 1962.

[5] D.J. Duffy. *Finite Difference Methods in Financial Engineering: A Partial Differential Equation Approach*. The Wiley Finance Series. Wiley, 2006.

[6] Intel. Intel math kernel library for c, gtsv. https://software.intel.com/en-us/mkl-developer-reference-c-gtsv, 2019. [Online; accessed 7-August-2019].

[7] Matthew J. Hancock. The 1-d heat equation (18.303, linear partial differential equations). https://ocw.mit.edu/courses/mathematics/18-303-linear-partial-differential-equations-fall-2006/

lecture-notes/heateqni.pdf, 2006. [Online; accessed 7-August-2019].

[8] D. W. Peaceman and H. H. Rachford. The numerical solution of parabolic and elliptic differential equations. *Journal of the Society for Industrial and Applied Mathematics*, 3(1):28–41, 1955.

[9] Charles Petzold. *Programming Windows, Fifth Edition*. Microsoft Press, Redmond, WA, USA, 5th edition, 1998.

[10] Sriram Ramaswamy. Pollen grains, random walks and einstein. *Resonance*, 5, 03 2000.

[11] Llewellyn Hilleth Thomas. Elliptic problems in linear difference equations over a network. *Watson Sci. Comput. Lab. Rept., Columbia University, New York*, 1, 1949.