



MSc 203
University Paris Dauphine

C++ Project

Thibaut JEREBITZ
Florian PERIGNE
Valentin ROCHEREAU

C++ Project
MSc 203
University Paris Dauphine

Thibaut JEREBITZ
Florian PERIGNE
Valentin ROCHEREAU

January 24, 2020

Abstract

The goal of the project is to solve the Black Scholes PDE.
In this document, we develop the mathematics needed to implement a C++ code to solve the PDE.

Outlines

Abstract	1
1 Introduction	3
2 Dirichlet	4
3 Neumann	5
4 Userguide	6

1 Introduction

The price of a European option satisfies the *Black Scholes equation*

$$0 = \frac{\partial f}{\partial t}(s, t) + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 f}{\partial s^2}(s, t) + rs \frac{\partial f}{\partial s}(s, t) - rf(s, t)$$

with the final condition $f(s, T) = V(s)$ where $V(s)$ is the payout.

Applying the variable change $x = \log(s)$ yields :

$$\frac{\partial f}{\partial t}(x, t) = -\frac{1}{2}\sigma^2 \frac{\partial^2 f}{\partial x^2}(x, t) + \left(\frac{1}{2}\sigma^2 - r\right) \frac{\partial f}{\partial x}(x, t) + rf(x, t)$$

with the final condition $f(x, T) = V(x) = V(e^x)$.

A common method to solve this PDE is to use finite differences to approximate the derivatives and then solve the equation on a discrete mesh.

When developping the different formula, we will have to set boundaries conditions for the points on the border of the mesh. We will develop two possible solutions to set these conditions :

- Dirichlet method (part 2)
- Neumann method (part 3)

2 Dirichlet

$$\forall n \in \llbracket 1, N \rrbracket, \forall i \in \llbracket 1, x_{max} \rrbracket,$$

$$f_i^n = f(t_n, x_i) \quad (1)$$

$$\frac{f_i^{n+1} - f_i^n}{dt} = \theta L_i^n + (1 - \theta) L_i^{n+1} \quad (2)$$

Let $A_1^{(1-\theta)}, B_-^{(1-\theta)}, B_+^{(1-\theta)}, A_2^\theta, B_-^\theta$ and B_+^θ take the following values :

$$\begin{aligned} A_1^{(1-\theta)} &= \frac{1}{dt} - \frac{\sigma^2}{dx^2}(1 - \theta) - r(1 - \theta) \\ B_-^{(1-\theta)} &= \frac{\sigma^2}{2dx^2}(1 - \theta) - \left(\frac{\sigma^2}{2} - r\right)(1 - \theta) \frac{1}{2dx} \\ B_+^{(1-\theta)} &= \frac{\sigma^2}{2dx^2}(1 - \theta) + \left(\frac{\sigma^2}{2} - r\right)(1 - \theta) \frac{1}{2dx} \\ A_2^\theta &= \frac{1}{dt} + \frac{\theta\sigma^2}{dx^2} + r\theta \\ B_-^\theta &= -\theta \frac{\sigma^2}{2dx^2} + \left(\frac{\sigma^2}{2} - r\right)\theta \frac{1}{2dx} \\ B_+^\theta &= -\theta \frac{\sigma^2}{2dx^2} - \left(\frac{\sigma^2}{2} - r\right)\theta \frac{1}{2dx} \end{aligned} \quad (3)$$

Then we obtain :

$$f_i^{n+1} A_1^{(1-\theta)} + f_{i+1}^{n+1} B_-^{(1-\theta)} + f_{i-1}^{n+1} B_+^{(1-\theta)} = f_i^n A_2^\theta - f_{i+1}^n B_-^\theta - f_{i-1}^n B_+^\theta \quad (4)$$

The previous formula gives us the following matrix system :

$$\begin{aligned} &\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ B_+^{(1-\theta)} & A_1^{(1-\theta)} & B_-^{(1-\theta)} & 0 & 0 & \cdots & 0 & 0 \\ 0 & B_+^{(1-\theta)} & A_1^{(1-\theta)} & B_-^{(1-\theta)} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} f_0^{n+1} \\ f_1^{n+1} \\ \vdots \\ f_{x_{max}-1}^{n+1} \\ f_{x_{max}}^{n+1} \end{pmatrix} \\ &= \begin{pmatrix} e^{rdt} & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ -B_+^\theta & A_2^\theta & -B_-^\theta & 0 & 0 & \cdots & 0 & 0 \\ 0 & -B_+^\theta & A_2^\theta & -B_-^\theta & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & e^{rdt} \end{pmatrix} \begin{pmatrix} f_0^n \\ f_1^n \\ \vdots \\ f_{x_{max}-1}^n \\ f_{x_{max}}^n \end{pmatrix} \end{aligned} \quad (5)$$

3 Neumann

$$\forall n \in \llbracket 1, N \rrbracket, \forall i \in \llbracket 1, x_{max} \rrbracket,$$

$$\frac{df}{dx}(t_n, x_0) = k_1 \quad (6)$$

$$\frac{df}{dx}(t_n, x_0) = \frac{f_1^n - f_0^n}{dx} \Leftrightarrow f_0^n = f_1^n - k_1 dx \quad (7)$$

$$\frac{df}{dx}(t_n, x_{max}) = k_2 \quad (8)$$

$$\frac{df}{dx}(t_n, x_0) = \frac{f_{x_{max}}^n - f_{x_{max}-1}^n}{dx} \Leftrightarrow f_{x_{max}}^n = f_{x_{max}-1}^n + k_2 dx \quad (9)$$

Using the same formula (equation 4):

$$f_i^{n+1} A_1^{(1-\theta)} + f_{i+1}^{n+1} B_-^{(1-\theta)} + f_{i-1}^{n+1} B_+^{(1-\theta)} = f_i^n A_2^\theta - f_{i+1}^n B_-^\theta - f_{i-1}^n B_+^\theta \quad (10)$$

Taking the left part of the equation 10, and using equation 7 we have :

$$f_i^{n+1} A_1^{(1-\theta)} + f_{i+1}^{n+1} B_-^{(1-\theta)} + f_{i-1}^{n+1} B_+^{(1-\theta)} = f_1^n A_2^\theta - f_2^n B_-^\theta - (f_1^n - k_1 dx) B_+^\theta \quad (11)$$

$$= f_1^n (A_2^\theta - B_+^\theta) - f_2^n B_-^\theta + k_1 B_+^\theta dx \quad (12)$$

Taking the right part of the equation 10, and using equation 9 we have :

$$f_i^n A_2^\theta - f_{i+1}^n B_-^\theta - f_{i-1}^n B_+^\theta = f_{x_{max}-1}^n A_2^\theta - f_{x_{max}}^n B_-^\theta - f_{x_{max}-2}^n B_+^\theta \quad (13)$$

$$= f_{x_{max}-1}^n (A_2^\theta - B_-^\theta) - k_2 B_-^\theta dx - f_{x_{max}-2}^n B_+^\theta \quad (14)$$

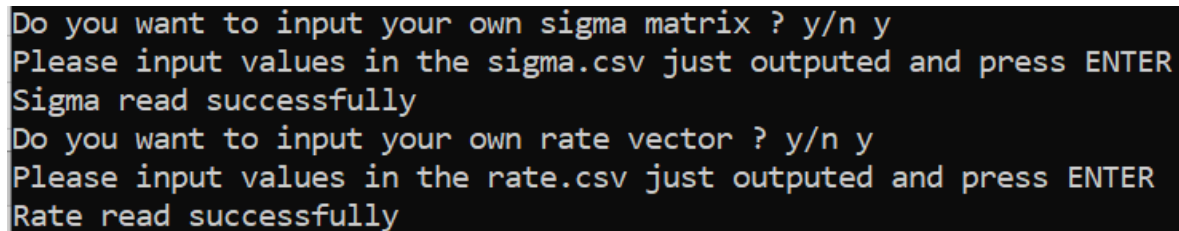
The previous formulas give us the following matrix system :

$$= \begin{pmatrix} -\frac{1}{dx} & \frac{1}{dx} & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ B_+^{(1-\theta)} & A_1^{(1-\theta)} & B_-^{(1-\theta)} & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & B_+^{(1-\theta)} & A_1^{(1-\theta)} & B_-^{(1-\theta)} & 0 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & B_+^{(1-\theta)} & A_1^{(1-\theta)} & B_-^{(1-\theta)} \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & -\frac{1}{dx} & \frac{1}{dx} \end{pmatrix} \begin{pmatrix} f_0^{n+1} \\ f_1^{n+1} \\ \vdots \\ f_{x_{max}-1}^{n+1} \\ f_{x_{max}}^{n+1} \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ B_+^\theta dx & A_2^\theta - B_+^\theta & -B_-^\theta & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & -B_+^\theta & A_2^\theta & -B_-^\theta & 0 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & -B_+^\theta & A_2^\theta & -B_-^\theta \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & -B_+^\theta & A_2^\theta - B_-^\theta \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} k_1 \\ f_1^n \\ \vdots \\ f_{x_{max}-1}^n \\ k_2 \end{pmatrix} \quad (15)$$

4 Userguide

Our program offers the possibility to the user to input the volatility matrix and the rates vector in an excel (csv) file. To do so, the program will ask the user if he wants to use this feature (see figure 1). If the user answers "yes" (y) then the program creates two .csv files to be completed by the user. One for the volatility and the other one for the rates. To input its parameters, the user can open the .csv file in excel, and convert the column A "from text to columns". User will find an empty matrix with the X in lines (all spot prices) and Y in columns (all the times considered). Once saved the user continues to run the program by pressing "ENTER".

A screenshot of a terminal window with a black background and yellow text. The text shows a sequence of prompts and user responses. First, it asks 'Do you want to input your own sigma matrix ? y/n', followed by the user input 'y'. Then it says 'Please input values in the sigma.csv just outputed and press ENTER', followed by 'Sigma read successfully'. Next, it asks 'Do you want to input your own rate vector ? y/n', followed by the user input 'y'. Then it says 'Please input values in the rate.csv just outputed and press ENTER', followed by 'Rate read successfully'.

```
Do you want to input your own sigma matrix ? y/n y
Please input values in the sigma.csv just outputed and press ENTER
Sigma read successfully
Do you want to input your own rate vector ? y/n y
Please input values in the rate.csv just outputed and press ENTER
Rate read successfully
```

Figure 1: Options available