# Projet Cpp Maths

julien.g.baudin@gmail.com

November 2019

# Introduction

We work with the Black-Scholes Partial Differential Equation to which we apply the variable change x=log(S):

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

With the final condition $f(x,T) = V(x) = V(e^S)$ where V is the payoff.

Starting from the option value at maturity, a common method to solve this PDE and retrieve the price of the option today is to use finite differences to approximate the derivatives on a discrete mesh. We are going to solve the Crank-Nicholson scheme which merges both implicit and explicit methods to combine their properties.

# Working out the maths

## Solving the problem

First, let us define:

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

The Delta Greek equals:

$$\frac{\partial f}{\partial x}(x_i, t_n) = \frac{f_{i+1}^n - f_{i-1}^n}{2dx} \tag{3}$$

The Gamma Greek equals:

$$\frac{\partial^2 f}{\partial x^2}(x_i, t_n) = \frac{f_{i+1}^n + f_{i-1}^n - 2f_i^n}{dx^2} \tag{4}$$

Then, the Theta Greek equals:

$$\frac{\partial f}{\partial t}(x_i, t_n) = \frac{f_i^{n+1} + f_i^n}{dt} \tag{5}$$

For each step on the discrete mesh, solving the Crank Nicholson scheme comes to solve the following equation:

$$\frac{f_i^{n+1} - f_i^n}{dt} = \theta L_i^n + (1-\theta)L_i^{n+1} \tag{6}$$

Where $dt$ is the time step on the mesh, index $n$ stands for the time step. Index $i$ stands for the price step for the underlying asset, and $dx$ is the price step on the mesh. $\theta$ is an input defining the contribution of both implicit and explicit schemes to the Crank-Nicholson one.

Substituting equations 2, 3 and 4 into 6, one can obtain:

$$
\begin{aligned}
f_i^{n+1} - f_i^n = dt\theta(-\frac{1}{2}\sigma^2 \frac{f_{i+1}^n + f_{i-1}^n - 2f_i^n}{dx^2} + (\frac{1}{2}\sigma^2 - r)\frac{f_{i+1}^n - f_{i-1}^n}{2dx} + rf_i^n) \\
+ dt(1-\theta)(-\frac{1}{2}\sigma^2 \frac{f_{i+1}^{n+1} + f_{i-1}^{n+1} - 2f_i^{n+1}}{dx^2} + (\frac{1}{2}\sigma^2 - r)\frac{f_{i+1}^{n+1} - f_{i-1}^{n+1}}{2dx} + rf_i^{n+1})
\end{aligned} \tag{7}
$$

After reorganising the right side of equation 7, let us denote:

$$
\begin{aligned}
A_i^n &= f_{i+1}^n(dt\theta(\frac{-\sigma^2}{2dx^2} + \frac{\sigma^2}{4dx^2} - \frac{r}{2dx}) \\
&+ f_{i-1}^n(dt\theta(\frac{-\sigma^2}{2dx^2} + \frac{\sigma^2}{4dx^2} + \frac{r}{2dx}) \\
&+ f_i^n(dt\theta(\frac{\sigma^2}{dx^2} + r))
\end{aligned} \tag{8}
$$

$$
\begin{aligned}
A_i^{n+1} &= f_{i+1}^n(dt(1-\theta)(\frac{-\sigma^2}{2dx^2} + \frac{\sigma^2}{4dx^2} - \frac{r}{2dx})) \\
&+ f_{i-1}^n(dt(1-\theta)(\frac{-\sigma^2}{2dx^2} + \frac{\sigma^2}{4dx^2} + \frac{r}{2dx})) \\
&+ f_i^n(dt(1-\theta)(\frac{\sigma^2}{dx^2} + r))
\end{aligned} \tag{9}
$$

Let us denote:

$$
\begin{aligned}
\alpha^n &= dt\theta(\frac{-\sigma^2}{2dx^2} + \frac{\sigma^2}{4dx^2} - \frac{r}{2dx}) \\
\beta^n &= dt\theta(\frac{-\sigma^2}{2dx^2} + \frac{\sigma^2}{4dx^2} + \frac{r}{2dx}) \\
\gamma^n &= dt\theta(\frac{\sigma^2}{dx^2} + r)
\end{aligned}
$$

$$
\begin{aligned}
\alpha^{n+1} &= dt(1-\theta)(\frac{-\sigma^2}{2dx^2} + \frac{\sigma^2}{4dx^2} - \frac{r}{2dx}) \\
\beta^{n+1} &= dt(1-\theta)(\frac{-\sigma^2}{2dx^2} + \frac{\sigma^2}{4dx^2} + \frac{r}{2dx}) \\
\gamma^{n+1} &= dt(1-\theta)(\frac{\sigma^2}{dx^2} + r)
\end{aligned}
$$

One can rewrite equation 6:

$$A_i^n + f_i^n = f_i^{n+1} - A_i^{n+1} \tag{10}$$

Substituting $A^n$ and $A^{n+1}$ yields:

$$f_{i+1}^n \alpha^n + f_{i-1}^n \beta^n + f_i^n \gamma^n + f_i^n = f_i^{n+1} - (f_{i+1}^{n+1} \alpha^{n+1} + f_{i-1}^{n+1} \beta^{n+1} + f_i^{n+1} \gamma^{n+1}) \tag{11}$$

Simplifying equation 11, one can obtain:

$$f_{i+1}^n \alpha^n + f_{i-1}^n \beta^n + f_i^n (\gamma^n + 1) = f_{i+1}^{n+1}(-\alpha^{n+1}) + f_{i-1}^{n+1}(-\beta^{n+1}) + f_i^{n+1}(1 - \gamma^{n+1}) \tag{12}$$

Which implies we must solve the following matrix system to find $f_i^n$ for all $i \in \{0, 1, ..., N\}$:

$$\begin{pmatrix} \gamma^n + 1 & \alpha^n & 0 & \cdots & \cdots & \cdots & 0 \\ \beta^n & \gamma^n + 1 & \alpha^n & 0 & \cdots & \cdots & 0 \\ 0 & \beta^n & \gamma^n + 1 & \alpha^n & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \alpha^n \\ 0 & 0 & \cdots & \cdots & 0 & \beta^n & \gamma^n + 1 \end{pmatrix} \begin{pmatrix} f_1^n \\ f_2^n \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_{N-1}^n \end{pmatrix} + \begin{pmatrix} \beta^n f_0^n \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ 0 \\ \alpha^n f_N^n \end{pmatrix}$$

$$= \begin{pmatrix} -\gamma^{n+1} + 1 & -\alpha^{n+1} & 0 & \cdots & \cdots & \cdots & 0 \\ -\beta^{n+1} & -\gamma^{n+1} + 1 & -\alpha^{n+1} & 0 & \cdots & \cdots & 0 \\ 0 & -\beta^{n+1} & -\gamma^{n+1} + 1 & -\alpha^{n+1} & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \alpha^{n+1} \\ 0 & 0 & \cdots & \cdots & 0 & \beta^{n+1} & \gamma^{n+1} + 1 \end{pmatrix} \begin{pmatrix} f_1^{n+1} \\ f_2^{n+1} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_{N-1}^{n+1} \end{pmatrix}$$

$$+ \begin{pmatrix} -\beta^{n+1} f_0^{n+1} \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ 0 \\ -\alpha^{n+1} f_N^{n+1} \end{pmatrix} \tag{13}$$

For the points on the mesh border, we need to set boundary conditions. One can either set values of $f$ at the boundaries, or can impose values for the derivatives. The former approach is called Dirichlet conditions, the latter is the Neumann conditions.

## Dirichlet Boundary Conditions

We impose values to $f_0^n$ and $f_N^n$ $\forall n \in \{0, 1, ..., T\}$. Let us say $f_0^n = l$ and $f_0^n = h$, substituting in equation 13 yields:

$$\begin{pmatrix} \gamma^n+1 & \alpha^n & 0 & \cdots & \cdots & \cdots & 0 \\ \beta^n & \gamma^n+1 & \alpha^n & 0 & \cdots & \cdots & 0 \\ 0 & \beta^n & \gamma^n+1 & \alpha^n & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \alpha^n \\ 0 & 0 & \cdots & \cdots & 0 & \beta^n & \gamma^n+1 \end{pmatrix} \begin{pmatrix} f_1^n \\ f_2^n \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_{N-1}^n \end{pmatrix} + \begin{pmatrix} \beta^n l \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ 0 \\ \alpha^n h \end{pmatrix}$$
$$= \begin{pmatrix} -\gamma^{n+1}+1 & -\alpha^{n+1} & 0 & \cdots & \cdots & \cdots & 0 \\ -\beta^{n+1} & -\gamma^{n+1}+1 & -\alpha^{n+1} & 0 & \cdots & \cdots & 0 \\ 0 & -\beta^{n+1} & -\gamma^{n+1}+1 & -\alpha^{n+1} & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \alpha^{n+1} \\ 0 & 0 & \cdots & \cdots & 0 & \beta^{n+1} & \gamma^{n+1}+1 \end{pmatrix} \begin{pmatrix} f_1^{n+1} \\ f_2^{n+1} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_{N-1}^{n+1} \end{pmatrix}$$
$$+ \begin{pmatrix} -\beta^{n+1} l \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ 0 \\ -\alpha^{n+1} h \end{pmatrix} \quad (14)$$

$$\underset{(N-1)\times(N-1)}{A}\underset{(N-1)\times 1}{X^n} + \underset{(N-1)\times 1}{C^n} = \underset{(N-1)\times(N-1)}{B}\underset{(N-1)\times 1}{X^{n+1}} + \underset{(N-1)\times 1}{C^{n+1}} \quad (15)$$

It remains to isolate the unknown vector $X^n$ to solve this system of linear equations:

$$\underset{(N-1)\times 1}{X^n} = \underset{(N-1)\times(N-1)}{A^{-1}}(\underset{(N-1)\times(N-1)}{B}\underset{(N-1)\times 1}{X^{n+1}} + \underset{(N-1)\times 1}{C^{n+1}} - \underset{(N-1)\times 1}{C^n}) \tag{16}$$

# Neumann Boundary Conditions

With Neumann conditions, we impose values for the first and second-order derivatives for all $n$ in $\{0, 1, ..., T\}$.

$$\frac{\partial f}{\partial x}(x_0, t_n) = k_1$$

(17)

$$\frac{\partial f}{\partial x}(x_N, t_n) = k_3$$

(18)

$$\frac{\partial^2 f}{\partial x^2}(x_0, t_n) = k_2$$

(19)

$$\frac{\partial^2 f}{\partial x^2}(x_N, t_n) = k_4$$

(20)

Then, for $f_0^n$, substituting constants $k_1$ and $k_2$ into equation 6, one obtains:

$$\begin{aligned} f_0^{n+1} - f_0^n = dt\theta(-\frac{1}{2}\sigma^2 k_1 + (\frac{1}{2}\sigma^2 - r)k_2 + rf_0^n \\ + dt(1-\theta)(-\frac{1}{2}\sigma^2 k_1 + (\frac{1}{2}\sigma^2 - r)k_2 + rf_0^{n+1}) \end{aligned} \tag{21}$$

with equation 21, one obtains:

$$f_0^n = \frac{f_0^{n+1} - dt\theta(-\frac{1}{2}\sigma^2 k_1 + (\frac{1}{2}\sigma^2 - r)k_2) - dt(1-\theta)(-\frac{1}{2}\sigma^2 k_1 + (\frac{1}{2}\sigma^2 - r)k_2 + r f_0^{n+1})}{(1 + dt\theta r)} \tag{22}$$

$$f_0^n = \frac{f_0^{n+1} - dt(-\frac{1}{2}\sigma^2 k_1 + (\frac{1}{2}\sigma^2 - r)k_2 + (1-\theta) r f_0^{n+1})}{(1 + dt\theta r)} \tag{23}$$

Same idea with $f_N^n$, replacing derivatives by constants $k_3$ and $k_4$, it yields:

$$\begin{aligned} f_N^{n+1} - f_N^n = dt\theta(-\frac{1}{2}\sigma^2 k_3 + (\frac{1}{2}\sigma^2 - r)k_4 + r f_N^n \\ + dt(1-\theta)(-\frac{1}{2}\sigma^2 k_3 + (\frac{1}{2}\sigma^2 - r)k_4 + r f_N^{n+1}) \end{aligned} \tag{24}$$

Which leads to the same type of result:

$$f_N^n = \frac{f_N^{n+1} - dt(-\frac{1}{2}\sigma^2 k_3 + (\frac{1}{2}\sigma^2 - r)k_4 + (1-\theta) r f_0^{n+1})}{(1 + dt\theta r)} \tag{25}$$

We can now substitute equations 23 and 25 into equation 13 and solve this matrix system just like we did in equation 16.

# Determination of the greeks

Now that we have computed the option value at each point of the grid (in particular at time t=0 and t=1), we can compute an approximation of the option greeks at time t=0 using the following formulas:

- **Delta of the option (central difference approximation):**

The use of the central difference approximation is more accurate than either the forward or backward difference. In particular, its error is $\mathcal{O}(dx^2)$.

$$\forall i \in \{1, ..., N-1\}, \delta_i^0 = \frac{f_{i+1}^0 - f_{i-1}^0}{2dx}$$

- **Gamma of the option (central difference approximation):**

The error in this approximation is also $\mathcal{O}(dx^2)$.

$$\forall i \in \{1, ..., N-1\}, \gamma_i^0 = \frac{f_{i+1}^0 - 2f_i^0 + f_{i-1}^0}{dx^2}$$

- **Theta of the option (forward difference approximation):**

The forward difference approximation is the usual approximation of theta in the Crank-Nicolson scheme.

$$\forall i \in \{1, ..., N-1\}, \theta_i^0 = \frac{f_i^1 - f_i^0}{dt}$$

- **Management of the boundary conditions:**

- In the case of Dirichlet boundary conditions, the user inputs constant boundary values. Thus we consider a null value for the derivative of the option price on each point of these boundaries. We then assume a null Delta, Gamma and Theta on these boundaries.

- In the case of Neumann boundary conditions, the user inputs K1, K2, K3, K4 which represent the first and second derivative of the option price with respect to x on each point of the boundaries. We then simply consider these values as Delta and Gamma on the boundaries. On the other hand, we compute theta on the boundaries with a forward difference approximation formula, as if it was a regular point of the grid.

- **Approach for computing the Vega of the option:**

Let's consider the value of the option at time 0 for a given value i of the underlying: $f_i^0$. A simple approach to compute the Vega of the option would then be to compare its value obtained with a volatility $\sigma$ and its value obtained with a volatility $\sigma + \epsilon$. $\epsilon$ represents the level of precision of our computation. Then we could obtain the Vega of the option using the following formula:

$$\forall \epsilon > 0, \forall i \in \{0, ..., N\}, \vartheta_i^0 = \frac{f_{i+1}^0(\sigma + \epsilon) - f_i^0(\sigma)}{\epsilon}$$