Option Calculator

Assignment 17

Gianmarco Mulazzani

Applied Numerical Finance - 20247
A.Y. 2020/21

September 2021

Overview

1. Introduction
2. Background Theory
3. Practical Implementation
4. Conclusion

Introduction

- A user-friendly interface for option pricing has been created using **Python** and **Tkinter** library.
- The user is allowed to insert different inputs and choose between American and European Call and Put Options.
- As regards models implemented, these are the **Binomial Tree Model** (for both Europeans and Americans) and the **Black-Scholes Model** (only for European Options).
- Python scripts of reference are: `Options.py` in which functions are defined, and `Option_Calculator.py` with the graphical implementation.

Background Theory: Black-Scholes Model (1)

In this slide, the main formulas used for the implementation of the Black-Scholes Model are reported.

$$C_0 = Se^{-qT}\mathcal{N}(d_1) - Ke^{-rT}\mathcal{N}(d_2) \tag{1}$$

$$P_0 = Ke^{-rT}\mathcal{N}(-d_2) - Se^{-qT}\mathcal{N}(-d_1) \tag{2}$$

$$\text{with} \quad d_1 = \frac{ln\left(\frac{S}{K}\right) - \left(r - q + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} \quad \text{and} \quad d_2 = d_1 - \sigma\sqrt{T} \tag{3}$$

Note that (1) refers to Call Options while (2) is used to price Put Options.

Background Theory: Black-Scholes Model (2)

In this slide, the main formulas used for the computation of Greeks are stated below (here, the case of a Call Option is considered).

$$\Delta = \frac{\partial C_0}{\partial S} = e^{-qT}\mathcal{N}(d_1) \tag{4}$$

$$\mathcal{V} = \frac{\partial C_0}{\partial \sigma} = Ke^{-rT}\mathcal{N}'(d_2)\sqrt{T}\frac{1}{100} \tag{5}$$

$$\Theta = \frac{\partial C_0}{\partial T} = \left[qSe^{-qT}\mathcal{N}(d_1) - rKe^{-rT}\mathcal{N}(d_2) - \frac{\sigma}{2\sqrt{T}}Ke^{-rT}\mathcal{N}'(d_2)\right]\frac{1}{365} \tag{6}$$

$$\rho = \frac{\partial C_0}{\partial r} = TKe^{-rT}\mathcal{N}(d_2)\frac{1}{100} \tag{7}$$

$$\Gamma = \frac{\partial \Delta}{\partial S} = \frac{1}{S\sigma\sqrt{T}}e^{-qT}\mathcal{N}'(d_1) \tag{8}$$

Background Theory: Binomial Tree Model

In this slide, the Greeks computed under the Binomial Model are reported. While the pricing formulas are given for granted (refer to the report for more details).

$$\Delta = \frac{C_1(u) - C_1(d)}{S_1(u) - S_1(d)} \tag{9}$$

$$\Theta = \frac{C_2(ud) - C_0}{2 \; \Delta t \; 365} \tag{10}$$

$$\mathcal{V} = \frac{C_0(\sigma + 1\%) - C_0(\sigma - 1\%)}{100 \; 2\%} \tag{11}$$

$$\rho = \frac{C_0(r + 1\%) - C_0(r - 1\%)}{100 \; 2\%} \tag{12}$$

$$\Gamma = \frac{\Delta_1(u) - \Delta_1(d)}{S_1(u) - S_1(d)} \tag{13}$$

Practical Implementation (1)

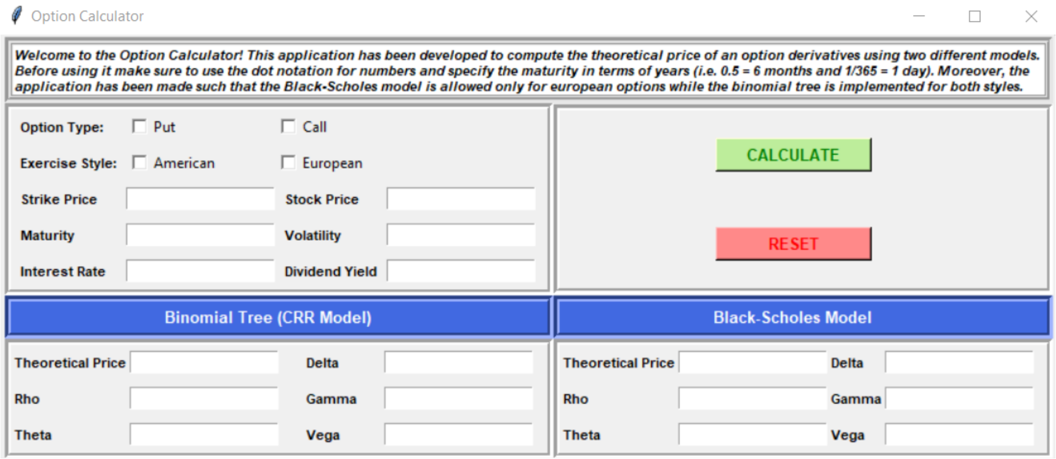

Figure: GUI of the Option Calculator

Practical Implementation (2)

- The previous figure represents the final interface that appears running the `Option_Calculator.py` file. This has been developed using Tkinter (`ttk`) and, without entering into the details, the main underlying idea to understand is that this library works by defining objects (called `widgets`) and positioning them within pre-defined frames (called `frame`).

- However, the **key script** developed is the one called `Options.py` because here, the models described above are implemented. The Black-Scholes Model is developed by just writing the known formulas using common mathematical notation. On the other hand, the Binomial Tree Model requires the definition of zero matrices that are then populated using backward recursion formulas implemented using nested `for` loops.

Conclusion

Overall, the code works correctly, and results are the ones desired. Moreover, for European Options, it is possible to empirically check the goodness of results obtained with the Binomial Tree comparing them to the Black-Scholes Model. Some interesting improvements could be:

- Enriching the possible models used, like the **Jump Diffusion Model**;
- Implement the possibility to price also Options different from standard plain vanilla ones;
- Make the interest rate automatically retrieved from an external source given the maturity specified.