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

- An 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: 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.

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

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

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

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

$$V = \frac{\partial C_0}{\partial \sigma} = K e^{-rT} \mathcal{N}'(d_2) \sqrt{T}$$
 (5)

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

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

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

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

$$C_2(ud) - C_2$$
(9)

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

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

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

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

Practical Implementation (1)

$m{/}$ Option Calculator $ \Box$ $ imes$				
Welcome to the Option Calculator! This application has been developed to compute the theoretical price of an option derivatives using two different models. Before using it make sure to use the dot notation for numbers and specify the maturity in terms of years (i.e. 0.5 = 6 months and 1/365 = 1 day). Moreover, the application has been made such that the Black-Scholes model is allowed only for european options while the binomial tree is implemented for both styles.				
Option Type:	□ Put	□ Call		
Exercise Style:	American	European	CALCULA	TE
Strike Price		Stock Price		
Maturity		Volatility	RESET	
Interest Rate		Dividend Yield		
Binomial Tree (CRR Model)			Black-Scholes Model	
Theoretical Price	,	Delta	Theoretical Price	Delta
Rho		Gamma	Rho	Gamma
Theta		Vega	Theta	Vega

Figure: GUI of the Option Calculator

Practical Implementation (2)

The previous Figure represents the final interface that appears running Option_Calculator.py file. This has been developed using

- The previous Figure represents the final interface that appears running
 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. As regards the Black-Scholes Model, it is developed by
 just writing the known formulas using common mathematical notation. On the other
 hand, the Binomial Tree Model required the definition of zero matrices that are then
 populated using backward recursion formulas implemented using for loops.

Conclusion

Overall, the code works properly and results are the ones desired. Moreover, for European Options it is possible to compare the results produced by both models. Some possible improvements are:

- 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 variable automatically retrieved from an external source given the maturity specified.