1 Introduction

2 History and Derivation of the Black-Scholes Equation

3 Numerical Methods for Solving the Black-Scholes Equation

3.1 Crank-Nicolson Method

The Crank-Nicolson Finite Difference Method is an implicit method used for numerically solving partial differential equations and is in fact commonly used to solve the Black-Scholes equation **??**. The method is essentially a combination of the Forward Euler and Backward Euler methods, taking the average of the two to achieve a higher order of accuracy. The Crank-Nicolson method is unconditionally stable and second-order accurate in both time and space, making it a popular choice for solving parabolic PDEs like the Black-Scholes equation **??**.

3.1.1 History

The Crank-Nicolson method was developed in the 1940s by John Crank and Phyllis Nicolson as a way to improve the accuracy of numerical solutions to heat equations. Since then, it has been widely adopted in various fields, including finance, for solving the Black-Scholes equation **??**. The method's stability and accuracy make it particularly well-suited for pricing options, where precise calculations are crucial. The derivation of the Crank-Nicolson method involves discretizing the time and space domains and applying finite difference approximations to the derivatives in the PDE. By averaging the spatial derivatives at the current and next time steps, the method achieves its second-order accuracy **??**. We have that for a general parabolic PDE of the form:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} \tag{1}$$

the Crank-Nicolson method can be expressed as:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{\alpha}{2}\left(\frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{(\Delta x)^2} + \frac{u_{i+1}^{n} - 2u_i^{n} + u_{i-1}^{n}}{(\Delta x)^2}\right) \tag{2}$$

where u_i^n represents the numerical solution at spatial point i and time step n, Δt is the time step size, and Δx is the spatial step size. This leads to a system of linear equations that can be solved at each time step to obtain the numerical solution **??**. It is easy to see that this method is a combination of the Forward Euler and Backward Euler methods, taking the average of the two as seen below:

$$\text{Forward Euler: } \frac{u_i^{n+1} - u_i^n}{\Delta t} = \alpha \frac{u_{i+1}^{n} - 2u_i^{n} + u_{i-1}^{n}}{(\Delta x)^2}$$
$$\text{Backward Euler: } \frac{u_i^{n+1} - u_i^n}{\Delta t} = \alpha \frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{(\Delta x)^2}.$$

As mentioned earlier, the Crank-Nicolson method is unconditionally stable, meaning that it remains stable regardless of the choice of time step size Δt and spatial step size Δx. To see this is true we can take the Fourier transform of the Crank-Nicolson scheme and analyze the amplification factor. The amplification factor G can be derived as follows:

$$G = \frac{1 - \frac{\alpha \Delta t}{2} k^2}{1 + \frac{\alpha \Delta t}{2} k^2} \tag{3}$$

where k is the wave number. The magnitude of the amplification factor is given by:

$$|G| = \left|\frac{1 - \frac{\alpha \Delta t}{2} k^2}{1 + \frac{\alpha \Delta t}{2} k^2}\right| \tag{4}$$

Since the denominator is always greater than the numerator for all values of Δt and k, we have that $|G| \leq 1$ for all choices of Δt and k. This implies that the Crank-Nicolson method is unconditionally stable, as the numerical solution will not grow unbounded over time regardless of the time step size Δt and spatial step size Δx chosen.

3.1.2 Implementation of CN Method

Because I hate my life, I decided to implement to CN Method as a class in Python. The class takes in the following parameters:

- `S_max`: The maximum stock price considered in the grid.
- `K`: The strike price of the option.
- `T`: The time to maturity of the option.
- `r`: The risk-free interest rate.
- `sigma`: The volatility of the underlying asset.
- `S_steps`: The number of spatial grid points (stock prices).
- `T_steps`: The number of time steps.

The class then constructs the spatial and temporal grids, initializes the option price matrix, and sets up the boundary and initial conditions. The main method of the class is `crank_nicolson`, which implements the Crank-Nicolson algorithm to compute the option prices at each grid point. The method constructs the tridiagonal matrix system based on the Crank-Nicolson discretization and solves it iteratively for each time step using a linear solver. Finally, the class provides a method to retrieve the computed option prices for analysis and visualization.

4 Results and Discussion

5 Conclusion