Black–Scholes Option Pricing Model

C++ Implementation and Documentation

December 9, 2025

1 Introduction

This document provides a concise description of the Black–Scholes model and its implementation in C++. The aim of the project is to compute theoretical prices for European call and put options given a set of market parameters. The final program is built using a Makefile and exposes a command-line interface for pricing.

2 Model Assumptions

The Black–Scholes model is based on the following assumptions:

- The underlying asset price follows a geometric Brownian motion.
- The risk-free interest rate r is constant.
- The volatility σ of the asset is constant.
- The market is frictionless, with no transaction costs.
- Options are European (exercise only at maturity).

3 Mathematical Formulation

Let S be the current price of the underlying asset, K the strike price, r the risk–free interest rate, σ the volatility and T the time to maturity (in years). We define:

$$d_1 = \frac{\ln(S/K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}}, \tag{1}$$

$$d_2 = d_1 - \sigma\sqrt{T}. \tag{2}$$

The price of a European call option is:

$$C = S\Phi(d_1) - Ke^{-rT}\Phi(d_2), \tag{3}$$

and the price of a European put option is:

$$P = Ke^{-rT}\Phi(-d_2) - S\Phi(-d_1), \tag{4}$$

where $\Phi(x)$ denotes the cumulative distribution function of the standard normal distribution.

4 Implementation Summary

The pricing logic is implemented within the `BlackScholes` class. The class provides public methods:

- `double callPrice()`
- `double putPrice()`

The cumulative normal distribution is calculated using the error function, provided by the C++ standard math library in `<cmath>`.

5 Usage

After compilation, the executable can be used from the command line:

```
./black_scholes_cli --type call --S 100 --K 100 --r 0.05 --sigma 0.2 --T 1
```

Example output:

```
Option Price: 10.4506
```

6 Build Instructions

The included Makefile automates the build:

```
make
```

This produces two binaries:

- $\texttt{black}_{s}choles_{c}li–command - lineapplication$
- $\texttt{bs}_{t}est–unit - styletestprogram$

7 Conclusion

The project provides a clean and extensible implementation of the Black–Scholes pricing formula. The structure supports future enhancements such as Greeks, dividend yield, implied volatility, and Monte Carlo validation.