

MIE 1622H: Assignment 4 – Asset Pricing

Dr. Oleksandr Romanko

March 18, 2019

Due: Wednesday, April 10, 2019, not later than 6:00 p.m.

Use **MATLAB** or **Python** for all MIE 1622H assignments.

You should hand in:

- Your report.
- Compress all of your MATLAB or Python code files and output files into a file **StudentID.zip** (which can be uncompressed by 7-zip (<http://www.7-zip.org>) software under Windows), and submit it via Quercus portal no later than 6:00 p.m. on April 10.

Where to hand in: Online via Quercus portal, both your code and report.

Introduction

European Option

The value of a European call or put option for a **无红利分配** **non-dividend-paying underlying stock** is given by the solution of the Black-Scholes equation.

By solving the Black-Scholes equation the **price** of a European **call option** is

$$C(S, t) = \mathcal{N}(d_1)S - \mathcal{N}(d_2)Ke^{-r(T-t)},$$

while the **price** of a European **put option** is

$$P(S, t) = \mathcal{N}(-d_2)Ke^{-r(T-t)} - \mathcal{N}(-d_1)S,$$

where $d_1 = \frac{1}{\sigma\sqrt{T-t}} \left[\ln\left(\frac{S}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T-t) \right]$ and $d_2 = d_1 - \sigma\sqrt{T-t}$.

Alternatively, a **price of an underlying stock** can be modeled by Monte Carlo simulations using Geometric Brownian Motion (GBM) with constant **drift μ and volatility σ** . The discretized version of GBM is known as **Geometric Random Walk** equation:

$$S_{t+1} = S_t \cdot e^{(\mu - \frac{1}{2}\sigma^2)\Delta t + \sigma\epsilon_t}.$$

We obtain prices of the European option from Monte Carlo simulations by **computing call and put payoffs** and **discounting those back to current time with the riskless short rate r** .

One may choose different number of **time steps** when computing the price of an option. For example, if maturity of an option on a stock is 1 year from today, we can compute a price of an underlying stock once at the end of the year. In this case the computation involves only one step and we will denote this method as one-step MC going forward. Alternatively, we may compute the price of an underlying every week, every month, every day of the year, in other words we chop the time interval into as many units as we need. Since we perform calculation multiple times throughout the life of an option, we will refer to this method as multiple-step MC.

In this assignment you need to compare prices of a European call and put options computed from Black-Scholes formula and from Monte Carlo simulations.

Price of underlying stock today ($t = 0$), i.e., spot price today, is $S_0 = 100$, strike at expiry is $K = 105$, years to expiry $T = 1$, risk-free rate $r = 0.05$, drift $\mu = 0.05$, volatility $\sigma = 0.2$.

Barrier Option

A barrier option is a type of option whose payoff depends on whether or not the underlying asset has reached or exceeded a predetermined price - barrier.

There are two general types of barrier options, *in* and *out* options. An *knock-out* option only pays off if the stock does not hit the barrier level throughout the life of the option. If the stock hits a specified barrier, then it has knocked out and expires worthless. A *knock-in* option on the other hand only pays out if the barrier is crossed during the life of the option.

If the barrier price is above the spot price, then the option is an up option; if the barrier is below the spot price then it is a down option. Therefore, we can categorize barrier options (put/call) as follows: down-and-out, down-and-in, up-and-out and up-and-in.

Monte Carlo simulations can be used to price barrier options in a similar manner as European options. Once multiple asset paths for GBM have been simulated, the next step is to determine the payoff for each asset path. This is done by evaluating each path to see whether it has hit the barrier. The payoff is then dependent on the type of barrier option and the knowledge of whether or not the barrier level has been hit during the life of the option. Some options will expire worthless if the barrier is reached, others will be worthless unless the barrier is reached.

In this assignment we will only focus on knock-in option: the option becomes a standard European option if the barrier was crossed some time before expiration. It will then pay if it ends up in-the-money. The barrier is \$110 for this assignment and all other variables (initial stock price, strike at expiry, years to expiry, risk-free rate, drift, volatility) remain the same as in the first part of the assignment. Note that you may use different number of scenarios for barrier option valuation.

Questions

1. (60 %) Implement pricing functions in MATLAB:

You need to implement three pricing functions:

1. Black-Scholes pricing formula for European option in the function BS_european_price.
2. Monte Carlo pricing procedure for European option in the function MC_european_price.
3. Monte Carlo pricing procedure for Barrier knock-in option in the function MC_barrier_knockin_price.

For the Monte Carlo pricing procedure choose a number of time steps and a number of scenarios.

There is a file **Option_pricing.m** on the course web-page. You are required to complete the code in the file.

2. (30 %) Analyze your results:

- Produce Black-Scholes call and put price for the given European option.
- Compute *one-step MC* call and put price for the given European option. Justify the number of paths used for computations.
- Compute *multi-step MC* call and put price for the given European option. Justify the number of steps and paths used for computations.
- Compute *one-step MC* call and put price for the given Barrier option. Justify the number of paths used for computations.
- Compute *multi-step MC* call and put price for the given Barrier option. Justify the number of steps and paths used for computations.
- Plot one chart in MATLAB that illustrates your Monte Carlo pricing procedure in the best way. Include the chart in your report.
- Compare three pricing strategies for European option and discuss their performance relative to each other.
- Explain the difference between call and put prices obtained for European and Barrier options.
- Compute prices of Barrier options with volatility increased and decreased by 10% from the original inputs. Explain the results.

3. (10 %) Discuss possible strategies to obtain the same prices from two procedures:

- Design your own procedure for choosing a number of time steps and a number of scenarios in Monte Carlo pricing for European option to get the same price (up to the cent) as given by the Black-Scholes formula.

MATLAB Script to be Completed

```
clc;
clear all;
format long

% Pricing a European option using Black-Scholes formula and Monte Carlo simulations
% Pricing a Barrier option using Monte Carlo simulations

S0 = 100;      % spot price of the underlying stock today
K = 105;       % strike at expiry
mu = 0.05;     % expected return
sigma = 0.2;   % volatility
r = 0.05;      % risk-free rate
T = 1.0;       % years to expiry
Sb = 110;      % barrier

% Define variable numSteps to be the number of steps for multi-step MC
% numPaths - number of sample paths used in simulations

% Implement your Black-Scholes pricing formula
[call_BS_European_Price, putBS_European_Price] = BS_european_price(S0, K, T, r, sigma);

% Implement your one-step Monte Carlo pricing procedure for European option
% numSteps = 1;
[callMC_European_Price_1_step, putMC_European_Price_1_step] = ...
    MC_european_price(S0, K, T, r, mu, sigma, 1, numPaths);

% Implement your multi-step Monte Carlo pricing procedure for European option
[callMC_European_Price_multi_step, putMC_European_Price_multi_step] = ...
    MC_european_price(S0, K, T, r, mu, sigma, numSteps, numPaths);

% Implement your one-step Monte Carlo pricing procedure for Barrier option
% numSteps = 1;
[callMC_Barrier_Knockin_Price_1_step, putMC_Barrier_Knockin_Price_1_step] = ...
    MC_barrier_knockin_price(S0, Sb, K, T, r, mu, sigma, numSteps, numPaths);

% Implement your multi-step Monte Carlo pricing procedure for Barrier option
[callMC_Barrier_Knockin_Price_multi_step, putMC_Barrier_Knockin_Price_multi_step] = ...
    MC_barrier_knockin_price(S0, Sb, K, T, r, mu, sigma, numSteps, numPaths);

disp(['Black-Scholes price of an European call option is ', num2str(call_BS_European_Price)])
disp(['Black-Scholes price of an European put option is ', num2str(putBS_European_Price)])
disp(['One-step MC price of an European call option is ', num2str(callMC_European_Price_1_step)])
disp(['One-step MC price of an European put option is ', num2str(putMC_European_Price_1_step)])
disp(['Multi-step MC price of an European call option is ', num2str(callMC_European_Price_multi_step)])
disp(['Multi-step MC price of an European put option is ', num2str(putMC_European_Price_multi_step)])
disp(['One-step MC price of an Barrier call option is ', num2str(callMC_Barrier_Knockin_Price_1_step)])
disp(['One-step MC price of an Barrier put option is ', num2str(putMC_Barrier_Knockin_Price_1_step)])
disp(['Multi-step MC price of an Barrier call option is ', num2str(callMC_Barrier_Knockin_Price_multi_step)])
disp(['Multi-step MC price of an Barrier put option is ', num2str(putMC_Barrier_Knockin_Price_multi_step)])

% Plot results
% figure(1);
%%%%%%%%%% Insert your code here %%%%%%%%%%
```