End Term Report

Name: Shixue Feng   ID: Shixuef2   UIN: 672862297

First-Part: European Down-and-Out Continuous Barrier Options Pricing

1. First we use the “get-four-paths-for-the-price-of-one-path" method to create 4 different option price to simulate the path:

```
double put_option_price = 0.0;
double call_option_price = 0.0;
double put_option_price_adj = 0.0;
double call_option_price_adj = 0.0;

for (int i = 0; i < no_of_trials; i++)
{
    double current_stock_price1 = initial_stock_price;
    double current_stock_price2 = initial_stock_price;
    double current_stock_price3 = initial_stock_price;
    double current_stock_price4 = initial_stock_price;

    double current_stock_price1_adj = initial_stock_price;
    double current_stock_price2_adj = initial_stock_price;
    double current_stock_price3_adj = initial_stock_price;
    double current_stock_price4_adj = initial_stock_price;

    double p_d1 = 1.0; double p_d2 = 1.0; double p_d3 = 1.0; double p_d4 = 1.0;

    for (int j = 0; j < no_of_barriers; j++)
    {
        // create the unit normal variates using the Box-Muller Transform
        double x = get_uniform();
        double y = get_uniform();
        double a = sqrt(-2.0 * log(x)) * cos(6.283185307999998 * y);
        double b = sqrt(-2.0 * log(x)) * sin(6.283185307999998 * y);
```

Then we use the two Gaussian random variables to generate four simulation paths:

```
current_stock_price1_adj = current_stock_price1_adj * exp(delta_R + delta_SD * a);
current_stock_price2_adj = current_stock_price2_adj * exp(delta_R - delta_SD * a);
current_stock_price3_adj = current_stock_price3_adj * exp(delta_R + delta_SD * b);
current_stock_price4_adj = current_stock_price4_adj * exp(delta_R - delta_SD * b);
```

In this process, we have to check whether the price reach the barrier.

```
current_stock_price1 = ((current_stock_price1 * exp(delta_R + delta_SD * a)) > barrier_price) ? (current_stock_price1 * exp(delta_R + delta_SD * a)) : 0;
current_stock_price2 = ((current_stock_price2 * exp(delta_R - delta_SD * a)) > barrier_price) ? (current_stock_price2 * exp(delta_R - delta_SD * a)) : 0;
current_stock_price3 = ((current_stock_price3 * exp(delta_R + delta_SD * b)) > barrier_price) ? (current_stock_price3 * exp(delta_R + delta_SD * b)) : 0;
current_stock_price4 = ((current_stock_price4 * exp(delta_R - delta_SD * b)) > barrier_price) ? (current_stock_price4 * exp(delta_R - delta_SD * b)) : 0;
```

2. At the same time, we can also use 1-pc to simulate the path with the formula in the lesson pdf:

$$p_c = \begin{cases} 1 & \text{if } S_0 \leq B \text{ or } S_T \leq B \\ e^{-\frac{2 \log \frac{S_0}{B} \log \frac{S_T}{B}}{\sigma^2 T}} & \text{otherwise.} \end{cases}$$

Since the code is too long, I will not give a screen shot here.

3. Last we use the traditional way to get the Theoretical Pricing as the lecture's note.

Second-Part: European Down-and-Out Discrete Barrier Options Pricing

The only difference from continuous barrier is at the method 2 that we no longer check whether it touches the barrier every time point. We only check it at certain time point.

The method will be the Brownian Bridge:

In general when $X(t_1) = a$ and $X(t_2) = b$, the **distribution** of $X(t_i)$ for $t_i \in (t_1, t_2)$ is normally distributed with mean μ_i

$$\mu_i = a + \frac{t_i - t_1}{t_2 - t_1} \times (b - a),$$

and variance

$$\sigma_i^2 = \frac{(t_i - t_1)(t_2 - t_i)}{t_2 - t_1}.$$

Then as the lecture note we can get a pd to compute the probability that the price path never touched any discrete barrier located at any time.

Result

(1)

```
Microsoft Windows [版本 10.0.18362.535]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\Users\HP>cd C:\Users\HP\source\repos\1\Debug

C:\Users\HP\source\repos\1\Debug>1 1 0.05 0.25 50 40 1000000 1000 20
European Down-and-out Continuous Barrier Options Pricing via Monte Carlo Simulation
--------------------------------
Expiration Time (Years) = 1
Risk Free Interest Rate = 0.05
Volatility (%age of stock value) = 25
Initial Stock Price = 50
Strike Price = 40
Barrier Price = 20
Number of Trials = 1000000
Number of Divisions = 1000
--------------------------------
--------------------------------
The average Call Price by explict simulation = 12.7058
The Call Price using (1-p)-adjustment term   = 12.7058
Theoretical Call Price                       = 12.7063
--------------------------------
The average Put Price by explict simulation = 0.752092
The Put Price using (1-p)-adjustment term   = 0.751843
Theoretical Call Price                      = 0.751885

C:\Users\HP\source\repos\1\Debug>
```

(2)

```
C:\Users\HP\source\repos\1\Debug>1 1 0.05 0.25 50 40 2000000 25 20
--------------------------------
European Down-and-out Discrete Barrier Options Pricing via Monte Carlo Simulation
--------------------------------
Expiration Time (Years) = 1
Risk Free Interest Rate = 0.05
Volatility (%age of stock value) = 25
Initial Stock Price = 50
Strike Price = 40
Barrier Price = 20
Number of Trials = 2000000
Number of Discrete Barriers = 25
--------------------------------
--------------------------------
The average Call Price by explict simulation of price paths           = 12.7037
The average Call Price with Brownian-Bridge Correction on final price = 12.7037
--------------------------------
The average Put Price by explict simulation of price paths           = 0.751963
The average Put Price with Brownian-Bridge Correction on final price = 0.752612

C:\Users\HP\source\repos\1\Debug>
```