Name: Fei Liu
UNI: fl2312
Tele: 9176557918

Report for HW5

Part 1 Implementation of algorithm:

For a), the closed formula is exactly the same as in HW1. The Black-Scholes formula is as below:

$$C(S,t) = N(d_1)\, S - N(d_2)\, Ke^{-r(T-t)},$$
$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T-t)}{\sigma\sqrt{T-t}}$$
$$d_2 = \frac{\ln\left(\frac{S}{K}\right) + \left(r - \frac{\sigma^2}{2}\right)(T-t)}{\sigma\sqrt{T-t}} = d_1 - \sigma\sqrt{T-t}.$$

For b), I apply the explicit method directly using below formula:

$$\frac{V_i^k - V_i^{k+1}}{\Delta t} + a_i^k\left(\frac{V_{i+1}^k - 2V_i^k + V_{i-1}^k}{\Delta S^2}\right) + b_i^k\left(\frac{V_{i+1}^k - V_{i-1}^k}{2\Delta S}\right) + c_i^k V_i^k$$
$$= O(\Delta t, \Delta S^2),$$

And from this formula, it can be derived that below formula is true:

$$V_i^{k+1} = A_i^k V_{i-1}^k + (1 + B_i^k)V_i^k + C_i^k V_{i+1}^k,$$

where

$$A_i^k := \nu_1 a_i^k - \frac{1}{2}\nu_2 b_i^k,$$
$$B_i^k := -2\nu_1 a_i^k + c_i^k \Delta t,$$
$$C_i^k := \nu_1 a_i^k + \frac{1}{2}\nu_2 b_i^k,$$
$$\nu_1 := \frac{\Delta t}{\Delta S^2},$$
$$\nu_2 := \frac{\Delta t}{\Delta S}.$$

In the ExplicitFD.h, I defined the A, B, C coefficients above as thisA, thisB, and thisC, they take more parameters than they actually need. That is because I want to save them for future amendment just like I did for EulerScheme1.h before. The function pointers are passing those functions into the main ExplicitFD function as well as ExplicitFDBarrier function. The difference is just at the if condition I test at every step. I actually ceate ExplictiFDAmerican function as well. But, since it is not graded, I didn't include it in the ExplictiFDmain file. But you can have a look. Because American and European call option values are the same, they actually gives the same option price in the end.

The way I implemented them is just like setting up a matrix to fill eventually. Starting filling the ending payoff to the first column in the matrix and push it back. The tricky part is to realize that the front part of the matrix is actually the rear part of a real time line. As long as that is done, the remaining job is just to put the index in the right place and find the closest grid value to the spot price and return corresponding value as function output.

For c), the closed formed barrier option formula is available from previous homework. I skip the description here.

For d), The barrier option actually impose an if condition to everything step of the loop to test the condition and see if we need to break and assign 0 there. The final result is pretty good.

Part 2 Benchmark

(Note that I will put the analysis of change given different Nt, Ns, and Smax to next section)

I think the Benchmark also applied the same formula listed above. But, I am not sure how to export them out from the excel file. So, I just list the values available. I didn't found Explicit method in those files either.

	My Code with standard 10000 simulations (mean)	Haug's VBA Code
Closed-form European Style Vanilla call option price	9.05705	**9.0571**
Explicit FD European-style vanilla call price	9.06218	**N/A**
The closed-form European-style Barrer call price	0.0517846	**0.0511**

Explicit FD European-style Barrier call price	0.0564949	N/A

Part 3 Impact of change of parameters:

For Nt, I change it to 50. The corresponding values for explicit method change to 7.54657 * 10^17 and -1.43119* 10^8. Which is ridiculous.

For Ns, I change it to 150. The corresponding values for explicit method change to 1.88095* 10^225 and -1.3697* 10^151. Which is even more ridiculous.

For Smax, I change it to 400. The corresponding values for explicit method change to 7.29147 and 0.0537854. Which is not as ridiculous but is off.

The stability condition is

$$\Delta t \leq \frac{1}{\sigma^2 I^2}.$$

The right hand side is 0.0044444 if the Ns is 50. The time interval is 0.003968 in the case of 252 but not in the case of 50. So, the values go crazy. The right hand side Is 4.93827* 10^(-4) , which is obviously not satisfied and that's why we go crazy again. The safe way to go is just like what the professor told us to do.

I constructed a set of value with time steps 600 and space steps 80. This one satisfy the above condition. The values as 9.02481 and 0.415527. Not bad compared with the crazy case above. The accuracy can be Improved if the time steps go smaller.