CS 476/676 Assignment 2

Winter 2023

Instructor: Yuying Li	Office: DC3623	*yuying@uwaterloo.ca*
Lecture Times:	MW 11:30-12:50	MC 2035
Yuying Li:	OH: Thurs 3-4pm	
TA: Mohib Shirazi	DC 3594 (SciCom)	mmkshira@uwaterloo.ca
TA: Mohib Shirazi	OH: Feb 16 (Thursday): 2-3 PM	
Mohib Shirazi	OH: Feb 24 (Friday): 4-5 PM	

My Web Site: http://www.cs.uwaterloo.ca/~yuying

Due: 11am Feb 27, 2023

Programming Questions

IMPORTANT: In this and in future assignments, most of the marks for programming questions are allocated for explanations of algorithms (e.g. pseudo-code) and discussion of results. If all you hand in is the listing of the "Raw Code" or "Raw Output" by itself, you will get poor marks. Noting that all programming examples in lectures and course notes are in Matlab, coding for assignments can be done in Matlab or python. All the plots should be appropriately labeled. You should submit all code used in your assignment. Be sure to document (i.e. add liberal comments) your code. The TA will take off marks for poor documentation.

1 [(14 marks)] (Drifting Binomial Lattice)

In lectures we have considered the non-drift lattice. In this assignment we consider the Drifting Lattice described in Section 5.3 of the course notes, i.e.,

$$\begin{aligned} u &= \exp\left[\sigma\sqrt{\Delta t} + (r - \sigma^2/2)\Delta t\right] \\ d &= \exp\left[-\sigma\sqrt{\Delta t} + (r - \sigma^2/2)\Delta t\right] \\ & S_{j+1}^{n+1} = uS_j^n \ ; \ S_j^{n+1} = dS_j^n \\ q^* &= \frac{1}{2} \end{aligned} \tag{1}$$

Table 1: Some typical option parameters

σ	20%
r	3%
Time to expiry T	1 year
Strike Price	\$10
Initial asset price $S(0)$	\$10

Implement European option pricing using this binomial lattice. Your code should take only $O(N)$ storage NOT $O(N^2)$, $N = T/\Delta t$. This can be done efficiently if you store the payoff in an array of size $O(N)$ and index it appropriately.

(a) (9 marks) Write a code to price an European put option with the Drifting Lattice described (see also in Section 5.3 of the course notes). Using data in Table 1, assuming no dividend, write a binomial pricing code. Test your code for standard European call and put by comparing your results with the exact solutions from the Matlab function *blsprice* as follows. Let V_0^{exact} be the exact initial price $V(S(0),0)$ (from the Black-Scholes formula), and $V^{tree}(\Delta t)$ be the value from a lattice pricer with the time step Δt. If convergence rate is linear, then it can be shown that

$$V_0^{tree}(\Delta t) \quad = \quad V_0^{exact} + \alpha \Delta t + o(\Delta t) \tag{2}$$

where α is a constant independent of Δt. Then the ratio below

$$\lim_{\Delta t \to 0} \frac{V_0^{tree}((\Delta t)/2) - V_0^{tree}(\Delta t)}{V_0^{tree}((\Delta t)/4) - V_0^{tree}((\Delta t)/2)} \,. \tag{3}$$

would be 2. If convergence rate is quadratic, then

$$V_0^{tree}(\Delta t) \quad = \quad V_0^{exact} + \alpha (\Delta t)^2 + o((\Delta t)^2) \tag{4}$$

where α is some constant independent of Δt. What is the ratio when convergence is quadratic? Does your convergence table indicate a linear or quadratic convergence rate? Explain.

Show tables with the initial option value $V(S(0),0)$ as a function of Δt. Start off with a timestep $(\Delta t)^0 = .05$, and show the option value for $(\Delta t)^0/2, (\Delta t)^0/4, \ldots$. You should see your results converging to the *blsprice* value. Your tables should look like Table 2.

Table 2: Convergence Test

Δt	Value	Change	Ratio
.05	V_1		
.025	V_2	$V_2 - V_1$	
.0125	V_3	$V_3 - V_2$	$\frac{V_2 - V_1}{V_3 - V_2}$
...	...	...	...

(b) (5 marks) For (1), the strike price may not equal to any binomial node $S_j^N, j = 0, \ldots, N$. implement the smoothing payoff method at the $t = T$ (Section 5.4 of the notes). Repeat the experiment of question 1. What do you see?

2 [(12 marks)] (Drifting Lattice and Dividend)

Assume that the underlying stock pays dividend at the dividend payment time t_d where T is the expiry. The amount of dividend payment $\mathcal{D}$ at each t_d is specified below

$$\mathcal{D} = \max(\rho \cdot S(t_d^-), \mathcal{D}_0),$$

where $S(t_d^-)$ is the stock price immediately before dividend payment date t_d, $\rho > 0$ is a constant dividend rate, and $\mathcal{D}_0 > 0$ is a constant, specifying a dividend floor. Note that, when the stock pays dividend at t, the stock price at t^+, $S(t^+)$, becomes $S(t^-) - \mathcal{D}$, where t^- and t^+ denote immediately before and after the time of stock dividend payment respectively. Under no arbitrage, the option value immediately before dividend payment and immediately after dividend payment should remain the same.

Your handling of dividend should not require change to the lattice (1). In §5.5.2 of the course notes, the case when dividend is a fixed amount is discussed. Modify the binomial lattice call and put option pricing to handle the dividend payment case in this question using interpolation similar to the technique described in §5.5.2.

(a) (6 marks) Now, repeat Question 1 (a) (including checking for convergence), which implements smoothed payoff at T, but under above dividend payment specifications with $t_d = \frac{T}{2}$, $\mathcal{D}_0 = 0.3$ and $\rho = 0.3\%$.

(b) (6 marks) Repeat Question 1 (a) again (including checking for convergence), but assume now dividend is paid three times at $t_d = \frac{T}{4}, \frac{T}{2}, \frac{3T}{4}$ respectively with $\mathcal{D}_0 = 0.1$ and $\rho = 0.1\%$.
How do call and put values change in comparison to (a)? Describe your observations.

3 [(12 marks] (Monte Carlo)

A European down-and-out call barrier option pays out the call payoff at the expiry T when the underlying price S_t never hits the barrier $B < S_0$ in $t \in [0, T]$. Otherwise it pays nothing.

Assume that the asset price S evolves, in the risk neutral world, as follows:

$$dS = rSdt + \sigma SdZ \tag{5}$$

where r is the risk free return, σ is the volatility, and dZ is the increment of a standard Brownian motion.

(a) **(4 marks)** Consider a down-and-out call option with the expiry $T = 1$ year. The initial asset price and the strike are $S_0 = K = 100$, the down barrier is $B = 85$. The analytic formula for pricing this barrier option is

$$V(S,t) = S\left(\mathcal{N}(d_1) - (B/S)^{1+2r/\sigma^2}\left(1-\mathcal{N}(d_8)\right)\right) - Ke^{-r(T-t)}\left(\mathcal{N}(d_2) - (B/S)^{-1+2r/\sigma^2}\left(1-\mathcal{N}(d_7)\right)\right)$$

where

$$d_1 = \frac{\log(\frac{S}{K})+(r+\frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}} \qquad d_2 = d_1 - \sigma\sqrt{T-t}$$
$$d_7 = \frac{\log(\frac{SK}{B^2})-(r-\frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}} \qquad d_8 = d_7 - \sigma\sqrt{T-t}$$

where $\mathcal{N}(d)$ is the cumulative distribution function for normal distribution, which is **normcdf** in Matlab, Compute and plot the initial down-and-out call option for $S = 0.9S(0) : 2 : 1.1S(0)$. Other parameters can be found in Table 3.

Table 3: Some typical option parameters

σ	20%
r	5%
Time to expiry T	1 year
Strike	\$100
Initial asset price $S(0)$	\$100

(b) **(8 marks)** Let

$$\begin{aligned} N &= \frac{T}{\Delta t} \\ S(n\Delta t) &= S_n \end{aligned} \tag{6}$$

Then, given an initial price $S(0)$, M realizations of the path of a risky asset are generated using the algorithm

$$S_{n+1} = S_n e^{(r-\frac{1}{2}\sigma^2)\Delta t + \sigma\phi_n\sqrt{\Delta t}} \tag{7}$$

where $\{\phi_n\}$ are independent standard normals.
If the mth simulation value of $S_N = S(T)$ is denoted by $(S_N)^m$, then an approximate initial value of the option is given by (assuming initial price is $S(0)$)

$$\tilde{V}(S(0), 0) = e^{-rT}\frac{\sum_{m=0}^{M} \text{Payoff}((S_N)^m)}{M} \tag{8}$$

For down-and-out call

$$\text{Payoff}(S_N) = \begin{cases} \max(S_T - K, 0), & \text{if } S_t > B, \text{ for } \quad 0 \le t \le T \\ 0 & \text{otherwise.} \end{cases}$$

- The price simulation using (7) has no time discretization error. Does the error in the computed value $\tilde{V}(S(0), 0)$ depend on the time discretization? Explain.
- Code up this MC algorithm in Matlab to determine the initial value of the European **down-**and-out call. Vectorize your code for efficiency.
- For a fixed $\Delta t = 0.01$ plot the MC computed option values versus the number of simulations for a number of values of M. Repeat this computation for different Δt. Use M=1000:4000:45000, Δt = [0.01, 0.005, 0.0025, 0.001]. Use M as the x-axis, option value as y-axis, and plot the option values for all Δts in the same plot Show the exact price computed from (a) on the plot. Explain what you see.

Submit your code, plots, and discussion.

Graduate [(Graduate Student Question)] (15 marks).

Explain why the Monte Carlo method in Question 3 for pricing down-and-out call is slow in obtaining an accurate barrier option value. Read the recent short paper by Nouri et al, *Digital Barrier option pricing : an improved Monte Carlo algorithm*, Math Sci (2016) 10:65-70. Implement the probability approach in this paper and produce results and Figures, which are similar to Example 1, for the previous barrier option pricing, using Δt = [0.02,0.01,0.005,0.0025] and M = 100,000. Note that the exiting probability formula in the paper holds when $D = (B, +\infty)$ as well.

Comment on your observations of the computation results. Compare and discuss the improvement of the results compared to your implementation in previous barrier option pricing implementation.