# CS 476/676 Assignment 2

Winter 2022


| | | |
|---|---|---|
| Instructor: Yuying Li | Office: DC3623 | *yuying@uwaterloo.ca* |
| Lecture Times: | MW 4-5:20pm. | |
| Yuying Li: | OH: Tues 3-4pm | . |
| TA: Chendi Ni | OH: Feb 12 (Fri), 11 am - 12 pm | chendi.ni@uwaterloo.ca |
| Chendi Ni | OH: Feb 17 (Wed), 12 pm - 1 pm | |
| Chendi Ni | OH:Feb 19 (Fri), 1 pm - 2 pm | |

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


## Due: Feb 22, 2022

### 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) ] (Binomial Lattice)

In lectures we have considered the non-drift lattice

$$\begin{array}{lll} u = e^{\sigma\sqrt{\Delta t}}, & d = e^{-\sigma\sqrt{\Delta t}}, & q^* = \frac{e^{r\Delta t}-d}{u-d}, \\ S_{j+1}^{n+1} = uS_j^n, & S_j^{n+1} = dS_j^n & \end{array} \tag{1}$$

In this question, you will implement European option pricing under a discrete dividend for a binomial lattice specified in (1).

Table 1: Some typical option parameters

| | |
|---|---|
| $\sigma$ | 20% |
| $r$ | 3% |
| Time to expiry $T$ | 1 year |
| Strike Price | \$10 |
| Initial asset price $S(0)$ | \$10 |
| $\mathcal{D}_0$ | 0.5 |

Assume that the underlying stock pays dividend at in 4 months, i.e., the dividend payment time $t_d = \frac{T}{3}$, where $T$ is the expiry

The amount of dividend payment $\mathcal{D}$ is specified as 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.

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) Using data in Table 1 and your binomial pricing code but assuming $\rho = \mathcal{D}_0 = 0$ (i.e. no dividend), test your code for standard European call and put by comparing your results with the exact solutions from the Matlab function *blsprice*. Show tables with the initial option value $V(S(0), 0)$ as a function of $\Delta t$. 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

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

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 $\Delta t$. 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 $\alpha$ is a constant independent of $\Delta t$. 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 $\alpha$ is some constant independent of $\Delta t$. What is the ratio when convergence is quadratic? Does your convergence table indicate a linear or quadratic convergence rate? Explain.

(b) Generate tables of fair values of the same call and put options using $\Delta t = 0.005$, assuming dividend yield $\rho = 0, 2\%, 4\%, 8\%$ respectively. How do call and put values change with the dividend yield $\rho$?

**2** [(12 marks)] (Drifting Lattice)

**(a)** Now, price an European put option with the Drifting Lattice described in Section 5.3 of the course notes, i.e.,

$$\begin{aligned} u &= \exp\left[\sigma\sqrt{\Delta t} + (r - \sigma^2/2)\Delta t\right] \\ d &= \exp\left[-\sigma\sqrt{\Delta t} + (r - \sigma^2/2)\Delta t\right] \\ & \quad S_{j+1}^{n+1} = uS_j^n \ ; \ S_j^{n+1} = dS_j^n \\ q^* &= \frac{1}{2} \end{aligned} \tag{5}$$

Use the data in Table 1. Show a convergence table as in Table 2. Explain what you see.

**(b)** For (5), 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?

**3** [(12 marks)] (Monte Carlo).

A European cash-or-nothing put barrier option either pays out a pre-specified cash amount $x$ when the underlying price $S_T$ at $T$ satisfies $S_T < K$ and the underlying price $S_t$ never hits the barrier in $[0, T]$. Otherwise it pays nothing.

**(a). (4 marks)** Consider a down-and-out cash-or-nothing put option with the expiry of 6 month. The analytic formula for pricing this barrier option is

$$z_1 = \frac{\log(S/K)}{\sigma\sqrt{T}} + \frac{\sigma\sqrt{T}}{2} \qquad z_2 = \frac{\log(S/B)}{\sigma\sqrt{T}} + \frac{\sigma\sqrt{T}}{2}$$
$$y_1 = \frac{\log(B^2/(SK))}{\sigma\sqrt{T}} + \frac{\sigma\sqrt{T}}{2} \qquad y_2 = \frac{\log(B/S)}{\sigma\sqrt{T}} + \frac{\sigma\sqrt{T}}{2}$$

$$V(S,0) = xe^{-rT}\left(\mathcal{N}(-z_1+\sigma\sqrt{T}) - \mathcal{N}(-z_2+\sigma\sqrt{T}) + (S/B)\mathcal{N}(y_1-\sigma\sqrt{T}) - (S/B)\mathcal{N}(y_2-\sigma\sqrt{T})\right)$$

In Matlab, the cumulative distribution function for normal distribution $\mathcal{N}(d)$ is **normcdf**. Assume the strike price is $K = 102$, the down barrier is $B = 100$, the cash payout is $x = \$15$. Compute and plot the initial down-and-out cash-or-nothing put option for $S = 100 : 2 : 120$. Values of parameters $\sigma$ and $r$ are in Table 1.

**(b). (8 marks)** As we derived in class, for the purposes of pricing options, we assume that the asset price $S$ evolves as follows:

$$dS = rSdt + \sigma SdZ \tag{6}$$

where $r$ is the risk free return, $\sigma$ is the volatility, and $dZ$ is the increment of a standard Brownian motion. Let the expiry time of an option be $T$, and let

$$\begin{aligned} N &= \frac{T}{\Delta t} \\ S(t_n) &= S(n\Delta t) \end{aligned} \tag{7}$$

Then, given an initial price $S(0)$, $M$ realizations of the path $S(t_n),\ n = 0, 1, \ldots, N$ of the risky asset are generated using the algorithm

$$S(t_{n+1}) = S(t_n)e^{(r-\frac{1}{2}\sigma^2)\Delta t + \sigma\phi_n\sqrt{\Delta t}},\ n = 0, \ldots, N-1 \tag{8}$$

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

$$\tilde{V}(S(0),0) = e^{-rT}\frac{\sum_{m=0}^{M}\ \text{Payoff}((S(T))^m)}{M} \tag{9}$$

For down-and-out cash-or-nothing put

$$\text{Payoff}(S^N) = \begin{cases} x, & \text{if } S(T) < K \text{ and } S(t_n) > B, \text{ for } n = 1, \cdots, N \\ 0 & \text{otherwise.} \end{cases}$$

- The price simulation using (8) 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 cash-or-nothing European put.

- Assume that the initial asset price $S(0) = 105$ and other parameters the same as in (a). For each of $\Delta t = \frac{1}{200}, \frac{1}{400}, \frac{1}{800}, \frac{1}{1600}, \frac{1}{3200}, \frac{1}{6400}$, plot the computed option values using MC for $M = 1000 : 3000 : 100000$ respectively. Show the exact price computed from (a) on the plot. Explain what you see.

Submit a listing of your code, plots, and discussion.

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

Explain why the Monte Carlo method in Question 3 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 $\Delta t$ = [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.