Yeshiva University- Katz School of Science Health

Mathematics Department

Mathematics of Finance

Instructor: Marian Gidea

Project 3.1

Option Pricing via Binomial Tree

Brian Livian

November 20, 2022

Contents

1 Introduction

Option pricing is an important aspect of investing. Mathematics finance provides methods for option pricing. In this project we derive the methods for option pricing using binomial option pricing model and calculate the price of an option using a binomial tree with n = 4 steps.

2 Methodology

Some basic assumptions of the binomial tree option pricing model is given by the following:

- No arbitrage principle
- The movement of the asset when it goes up equal the movement when it goes down
- Under risk neutral probability, the probablity of the asset going up, p, equals the probability of the asset going down, 1-p = q = 1/2

Notation review:

- S(0)- today's stock price
- X- strike price of the call option
- T- expiration time
- C*- price of the option

Figure 1: One Step Binomial Tree

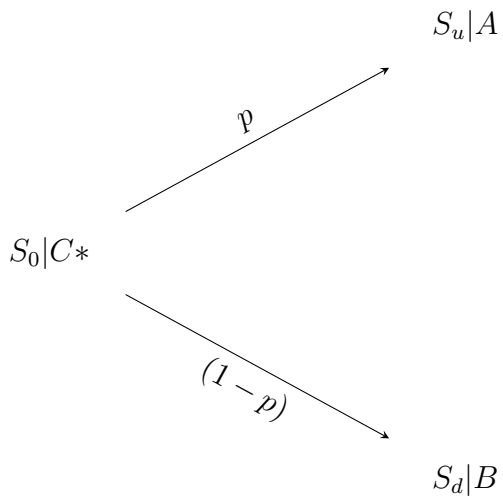

Figure 2: Binomial Tree n = 4

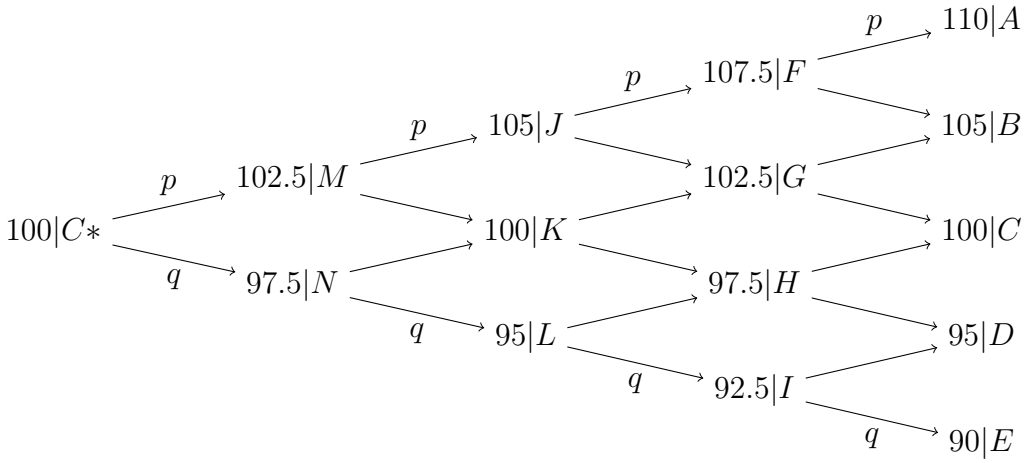

Figure 1 shows the simple case of a binomial tree with n = 1. The initial stock price is S_0. S_u is the stock price in the case where the stock goes up. Likewise, S_d is the stock price in the case where the stock goes down. A and B are the payoffs at S(T) (note: T = 1 in this case). At each step, the probability that the stock will go up is p, and the probability that the stock will go down is 1-p = q. In the case of risk netural probability, p = q = 1/2. We calculate the payoffs of S_u and S_d as following:

$$\text{Payoff}= \left\{ \begin{array}{ll} S(T)-x, & \text{if S(T) - x} > 0 \\ 0, & \text{otherwise} \end{array} \right\} (1)$$

Having the payoffs at S(T), we now work backwards in the tree, calculating the expectations at each node, ultimately reaching the option price C*. Assuming risk neutral probability, the simple case of n = 1, we calculate the option price to be C(C*) = $Payoff(S_u)$*p + $Payoff(S_d)$*q. We extend similarly for trees of $n > 1$, calculating expectation working backwards ultimately reaching the price of the option.

3 Computer Simulation & Results

Extending the methodology shown in in section 2, we run the risk neutral probability option pricing model in the case shown in figure 2. Hence, the price of the option in this case (expectation0) is $C* = 1.875$

```
expectation4 [10.  5.  0.  0.  0.]
expectation3 [7.5, 2.5, 0.0, 0.0]
expectation2 [5.0, 1.25, 0.0]
expectation1 [3.125, 0.625]
expectation0 [1.875]
```

Figure 3: Binomial Tree Results n = 4

Extending this methodology to the case of arbitrary n steps, we can study the behavior of the binomial tree option pricing model. Running this methodology for increasing values of n, we see the value of the option decreases asymptotically:

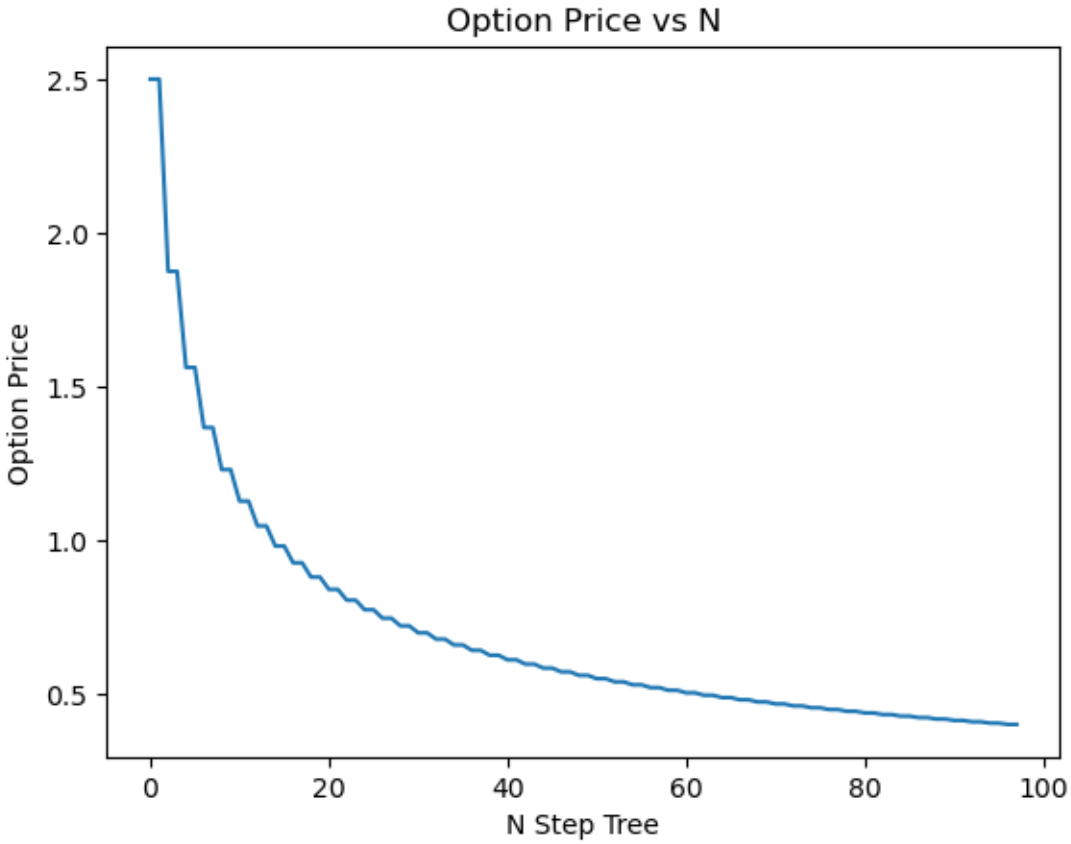

Figure 4: Option Price vs N

4 Literature Cited

[1] M. Capinski, T. Zastawniak, Mathematics for Finance, Springer Undergraduate Mathematics Series, Springer - Verlang Limited 2011

Appendix

```
g = globals()
def optionprice(S0, Su, Sd, n, p, x, printtree = True):
    q = 1-p
    # list the values of the rightmost nodes
    rightnodes = np.linspace(Su,Sd, n+1)
#     print('rightnodes', rightnodes) if printtree == True else None
    # Calculate the expectations of the right most nodes
    # This first step is simply the returns of the S: S - S0

    g[f'expectation{n}'] = np.where(rightnodes - x>0, rightnodes - x, 0)
    print('expectation{0}'.format(n), g['expectation{0}'.format(n)]) if printtree == True else None

    # Calculate the expectations for each of the other steps, going from right to left
    n-=1
    while n >=0:
        g['expectation{0}'.format(n)] = []
        for i in range(len(g['expectation{0}'.format(n + 1)])-1):
            # The expecation in risk neutral probability is simply (E(A) + E(B)) *p where p is 1/2
            g['expectation{0}'.format(n)].append(
                g['expectation{0}'.format(n+1)][i]*p + g['expectation{0}'.format(n+1)][i+1]*q
            )
        print('expectation{0}'.format(n), g['expectation{0}'.format(n)]) if printtree == True else None
        n-=1
    return(g['expectation{0}'.format(0)]) if printtree == False else None
```

```
# Print expectations at each n and ultimately the option price (expectation0)
# Case of n=4 (as seen in the image at top of notebook)
optionprice(S0 = None, Su= 110, Sd = 90, n=4, p= 1/2, x = 100, printtree = True)
```

```
expectation4 [10.  5.  0.  0.  0.]
expectation3 [7.5, 2.5, 0.0, 0.0]
expectation2 [5.0, 1.25, 0.0]
expectation1 [3.125, 0.625]
expectation0 [1.875]
```

```
# Recording the option price as n grows
optionprices = []
for n in range(2,100):
    optionprice(S0 = None, Su= 110, Sd = 90, n=n, p= 1/2, printtree = False)
    optionprices.append(g['expectation{0}'.format(0)])
```

```
plt.plot(optionprices)
plt.xlabel('N Step Tree')
plt.ylabel('Option Price')
plt.title('Option Price vs N')
```