Name: Elena Zoppellari
MSc: Physics of Data
Student ID: 2076715

Course: Stochastic Methods for Finance
Professor: Martino Grasselli
Date: April 7, 2023

Computing Option Prices using Discrete and Continuous Time Models

1 Introduction

In finance, computing the price of an option can be done using different methods, such as discrete time or continuous time models. This report focuses on three methods: the Binomial Model, Leisen-Reimer Model, and Black-Scholes formula. While the first two are formulas in discrete time, the Black-Scholes model is in continuous time, as it is derived from the convergence of the discrete models.
To compute the price of an option using these models, a VBA scripts can be written with the initial asset price, interest rate, strike price, maturity time, volatility, and number of steps as inputs. For this report, we assume initial values of stock $S_0 = 100$, strike $K = 100$, risk-free interest rate $r = 0.01$, a maturity time of $T = 1$ year, and a volatility of $\sigma = 0.2$. The VBA scripts are reported in the Appendix.

2 Theoretical Models

2.1 The Multi-Step Binomial Model

To price a call option using a multistep binomial model, one can first discretize the underlying asset's price movement over time into a binomial tree. The tree is constructed by assuming that the asset price can either go up or down by a certain factor at each time step. Specifically, if S is the current asset price, then at the next time step it can either be $u \cdot S$ (with probability q) or $d \cdot S$ (with probability $1 - q$), where u and d are the up and down factors, respectively, and p is the probability of an up move. These factors can be calculated using the asset's volatility σ and the time interval between steps Δt as follows:

$$u = e^{\sigma\sqrt{\Delta t}}, \quad d = e^{-\sigma\sqrt{\Delta t}}, \quad p = \frac{e^{r\Delta t} - d}{u - d},$$

where r is the risk-free interest rate.
Once the tree is constructed, the option price at each node of the tree can be calculated using the call option's payoff function $f(S) = \max(S - K, 0)$, where K is the strike price. Starting from the final nodes of the tree, the option price can be recursively calculated backward in time until the initial node is reached. Finally, the option price at the initial node of the tree represents the fair price of the call option under the multistep binomial model.

2.2 The Black & Scholes Model

The Black & Scholes model is a widely used method for pricing options in continuous time. It assumes that the underlying asset follows a geometric Brownian motion with constant drift μ and volatility σ, and that the risk-free interest rate is constant at r. Under these assumptions, the price p^{call} of a European call option with strike price K and maturity T can be expressed as:

$$p^{call} = S \cdot \Phi(d_1) - K \cdot e^{-r(T-t)}\Phi(d_2),$$

where $\Phi(\cdot)$ is the standard normal cumulative distribution function, and d_1 and d_2 are defined as:

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

The formula implies that the value of the call option depends on the current asset price S, the time to maturity T, the strike price K, the risk-free interest rate r, and the volatility σ. The Black-Scholes model assumes that the option can be exercised only at maturity, and that the underlying asset's price movements are continuous and follow a log-normal distribution.

2.3 Leisen and Reimer method

The Leisen-Reimer model is similar to the Binomial model, as both models use an underlying price binomial tree centered around the option's strike price at expiration. The logic and calculation of tree nodes and option prices are the same as in other binomial models. The primary difference lies in the calculation of tree parameters (u, d, and q).
In fact, probabilities must be calculated before move sizes because the former are inputs for the latter. The risk-neutral weight q is calculated using the Peizer-Pratt inversion function $h^{-1}(d_2)$ (where d_2 is the same as in the Black-Scholes formula) which provides discrete binomial estimates for the continuous normal cumulative distribution function.
Once we have calculated q, we can compute u and d using the following formulas:

$$\begin{cases} u = e^{r\cdot\Delta T} \cdot \frac{q'}{q} \\ d = e^{r\cdot\Delta T} \cdot \frac{1-q'}{1-q} \end{cases} \tag{1}$$

where ΔT is the maturity time, calculated as T/n, and $q' = h^{-1}(d1)$, where d_1 is as well the same calculated in Section 2.2. Note that in this case, the continuous dividend yield is assumed to be equal to 0.

3 Methodology and Results

As mentioned in the introduction, the variables applied to the three models in this paper are reported in Table 3.
In order to compare the three models, the price of a call option has been calculated, in particular for the two discrete models (Binomial and Leisen-Reimer) it has been highlighted the result of the price at different time steps and it has been used the result from the Black & Scholes formula

S_0	K	r	T	σ
100	100	0.01	1	0.2

Table 1: Variable's values for the three models

#n of step	p_{bin}^{call}	$\%err_{bin}$	$p_{L\&R}^{call}$	$\%err_{L\&R}$
5	8.824	4.637	8.42093	0.14695
10	8.350	0.989	8.43040	0.03471
50	8.439	0.073	8.43316	0.00188
100	8.443	0.115	8.43328	0.00056
200	8.441	0.096	8.43331	0.00022
500	8.437	0.045	8.43332	0.00012

Table 2: Prices for a call for binomial and Leisen-Reimer method, with relative percentage error in comparison with the Black Scholes result.

as a true reference value. All the results were obtained using the VBA scripts reported in the Appedix, which are based on the models descripted in Section 2.
The price of a call option using Black & Scholes model then results:

$$p_{BS}^{call} = 8.43333. \tag{2}$$

Table 2 reports some results from the application of the Binomial model and its Leisen and Reimer correction, along with their corresponding percentage errors.
To highlight the different behavior of the binomial model and the Leisen-Reimer method, we observe the variation of the price and the relative percentage error during the first 10 steps, shown in Figures 1 and 2.

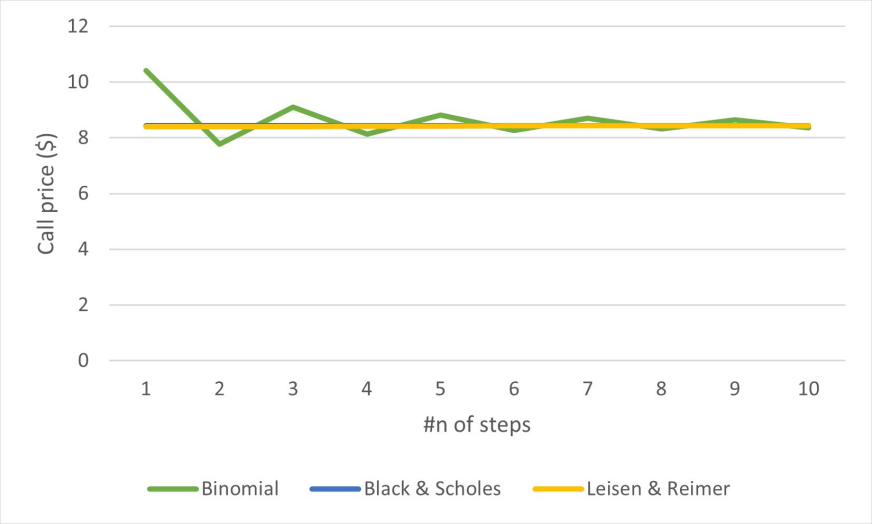

Figure 1: Call prices dependence by number of steps for the three models for the first 10 steps

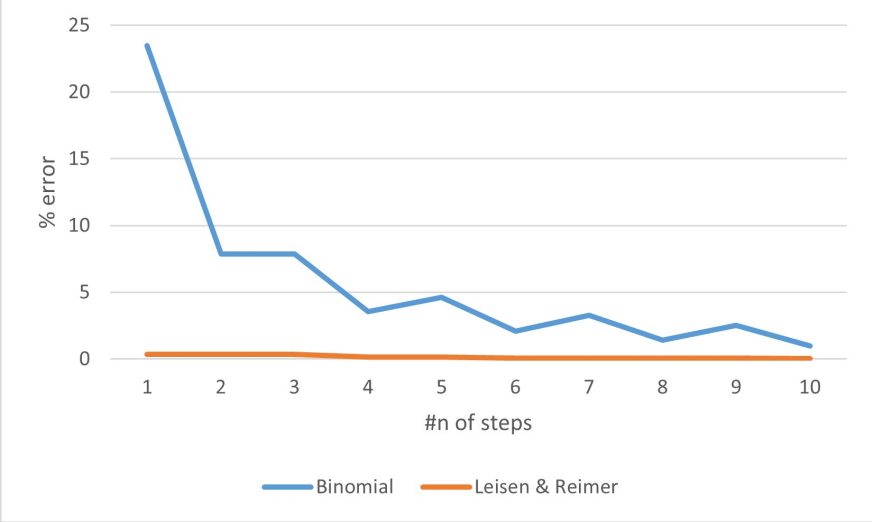

Figure 2: Percentage error for the first 10 steps for Binomial model and Leisen and Reimer correction

While the Leisen and Reimer method converges after a few steps, the binomial model requires at least 50 steps to converge successfully, as highlighted in Figures 3 and 4.

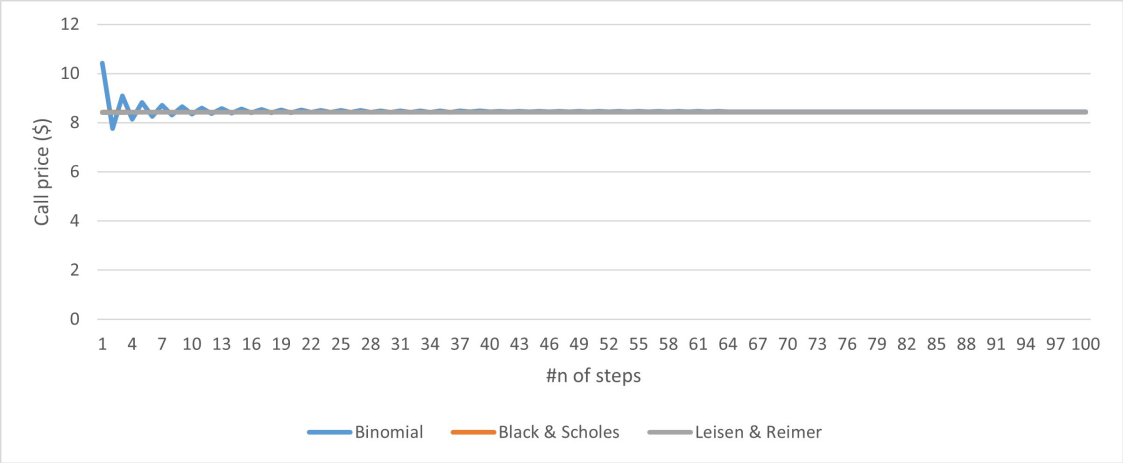

Figure 3: Call prices dependence by number of steps for the three models for the first 100 steps

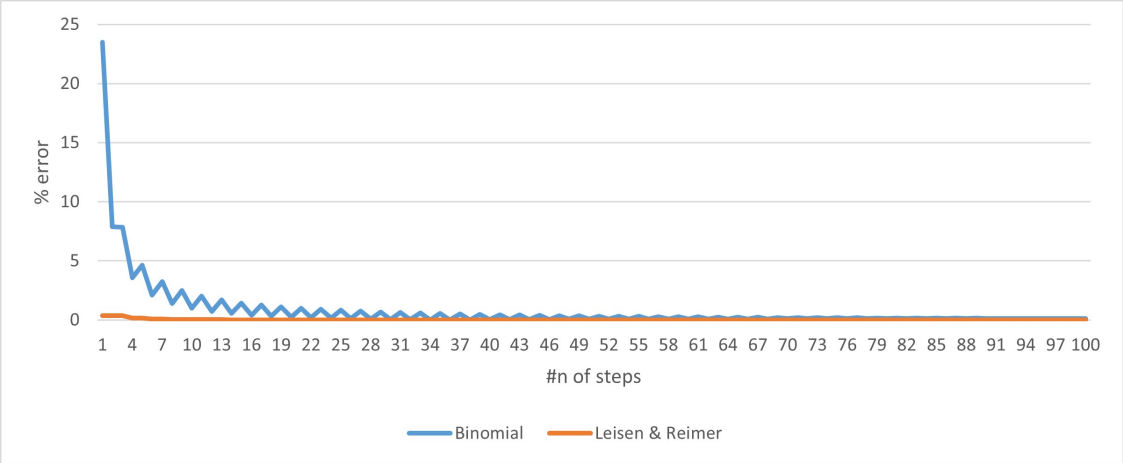

Figure 4: Percentage error for the first 100 steps for Binomial model and Leisen and Reimer correction

4 Conclusions

Comparing the two different methods, it is clear that the Leisen-Reimer method is more accurate than the Binomial model. In particular, its main benefit is greater precision with a smaller number of steps compared to the Binomial model. For example, with just $n = 10$ steps, the Leisen-Reimer method provides an error of 0.03% while the binomial model 0.99%, as reported in Table 3. As n grows, both the Binomial Model's and Leisen-Reimer method's results converge to the Black-Scholes formula, but the Binomial model remains less accurate. For $n = 500$, the Binomial Model only produces a percentage error of 0.04 %, result that the Leisen-Reimer method outperforms at only 10 steps. However, both methods converge to the Black-Scholes formula for $n \gg 1$.
Hence, if one has to choose one discrete method to describe the call option, it is better to choose the Leisen-Reimer model than the Binomial model, as it needs fewer computational resources (it gives good results with small n) and is more accurate. Naturally, if we could take $n = \infty$ steps,

both methods would give the same results, but computationally, this is not possible.

5 Appendix

In this appendix the VBA Script used to obtain the data presented in Section 3 are reported.

```
Function BS(S, K, T, r, sd)
    Dim a As Single
    Dim b As Single
    Dim c As Single
    Dim d1 As Single
    Dim d2 As Single

    a = Log(S / K)
    b = (r + 0.5 * sd   2) * T
    c = sd * (T   0.5)
    d1 = (a + b) / c
    d2 = d1 - sd * (T   0.5)
    BS = S * Application.NORMSDIST(d1) - K * Exp(-r * T) * Application.
       NORMSDIST(d2)

End Function
```

Listing 1: Black and Scholes model

```
Function Binomial(S, K, T, r, sd, n As Integer)
    Dim sdd As Single
    Dim j As Integer
    Dim rr As Single
    Dim q As Single
    Dim u As Single
    Dim d As Single
    Dim ttt As Single
    Dim sum As Single
    Dim nj As Double
    Dim firstBicomp As Single

    rr = Exp(r * (T / n)) - 1
    sdd = sd * Sqr(T / n)
    u = Exp(rr + sdd)
    d = Exp(rr - sdd)
    q = (1 + rr - d) / (u - d)

    sum = 0
    For j = 0 To n
        nj = Application.COMBIN(n, j)
        ttt = nj * (q   j) * ((1 - q)   (n - j)) * (S * (u   j) * (d
           (n - j)) - K)
```

```
        If ttt ¡ 0 Then ttt = 0
        sum = sum + ttt
    Next j

    Binomial = sum / ((1 + rr)   n)

End Function
```

Listing 2: Binomial Model

```
Function LR(S As Double, K As Double,T As Double, r As Double, sd As
   Double, n As Integer) As Variant
    Dim OptionValue() As Double
    Dim d1 As Double, d2 As Double
    Dim hd1 As Double, hd2 As Double
    Dim u As Double, d As Double, p As Double
    Dim dt As Double, Df As Double
    Dim i As Integer, j As Integer
    n = Application.Odd(n)

    ReDim OptionValue(0 To n)

    d1 = (Log(S / K) + (r + sd   2 / 2) * T) / (sd * Sqr(T))
    d2 = d1 - sd * Sqr(T)

    '// Preizer-Pratt inversion
    hd1 = 0.5 + Sgn(d1) * (0.25 - 0.25 * Exp(-(d1 / (n + 1 / 3 + 0.1 /
       (n + 1)))   2 * (n + 1 / 6)))   0.5

    hd2 = 0.5 + Sgn(d2) * (0.25 - 0.25 * Exp(-(d2 / (n + 1 / 3 + 0.1 /
       (n + 1)))   2 * (n + 1 / 6)))   0.5

    dt = T / n
    p = hd2
    u = Exp(r * dt) * hd1 / hd2
    d = (Exp(r * dt) - p * u) / (1 - p)
    Df = Exp(-r * dt)

    For i = 0 To n
         OptionValue(i) = Application.Max(0, (S * u   i * d   (n - i) -
             K))
    Next

    For j = n - 1 To ((n + 1) / 2) Step -1
        For i = (j - ((n - 1) / 2)) To j
            OptionValue(i) = (p * OptionValue(i + 1) + (1 - p) *
               OptionValue(i)) * Df
        Next
```

```
    Next

    For j = ((n + 1) / 2) - 1 To 0 Step -1
        For i = 0 To j + 1
            OptionValue(i) = (p * OptionValue(i + 1) + (1 - p) *
               OptionValue(i)) * Df
        Next
    Next

    LR = OptionValue(0)

End Function
```

Listing 3: Leisen-Reimer method