**Binomial Tree Method**

The binomial tree can be used to calculate option prices. It models potential underlying stock movements and using the payoff function, derives option prices.

The binomial tree method is a faster alternative to Monte Carlo simulation, as it requires a fraction of the runs of a simulation.

The stock pricing path component of the binomial tree is derived from the binomial distribution. The option pricing part is derived from the assumptions of no arbitrage or alternatively, risk-neutrality.

The logic of the underlying stock pricing element of the binomial tree approach comes from the properties of the binomial distribution. This approximates to the normal distribution under certain conditions, enabling it to be applied to the changes in the value of underlying stock prices, as these are, under Black Scholes assumptions, normally distributed.

Binomial trees therefore provide good numerical approximations to option pricing problems under lognormal underlying stock pricing assumptions, with lower computational effort than Monte Carlo simulation.

The binomial distribution measures the probability of a number of successes in a series of Bernoulli trials. A Bernoulli trial is an experiment with two possible outcomes: success or failure. The probability of a success is a constant p, and the probability of failure is a constant q = 1-p.

The binomial distribution therefore describes, given any number of Bernoulli trials N, the probability of x successes and N-x failures, for each integer value of x between 0 and N.

In the context of underlying stock pricing paths, the Bernoulli trial is applied to the underlying stock price to model the uncertain or stochastic movement of the underlying stock price during a time interval. The underlying stock price can either move up or down. The initial underlying stock price is S0. The up movement results in a price of S^u. The downward price movement results in a price of S0*d where u>1 and d<1. The probability of an upward movement is p, and an upward movement is therefore considered a success in a Bernoulli trial framework. The probability of a downward movement is q=1-p.

The binomial distribution is described by the probability density function $P(X=x) = nC_x p^x * q^{n-x}$ where $nC_x = n!/x!(n-x)!$.

The mean of the binomial distribution is given by $pq$, and the variance is given by $npq$. In the context of underlying stock pricing, we assign x as the number of upward price movements, n as the number of time intervals into which we divide the life of the option (the more intervals the better, in terms of accuracy, but there is a trade-off with computation time).

The approximation of the normal distribution by the binomial distribution is given by $P(X=x) = P(x-0.5 <= \Theta <= x+0.5)$. Where $\Theta$ is normally distributed, and x is the number of successes under a binomial model. This approximation is reasonably accurate so long as $\operatorname{mean} + 3*\sigma <= X <= \operatorname{mean} + 3*\sigma$. From Chebyshev's inequality this means 99.7% of the area under the normal distribution can be approximated by the binomial.
Modelling the **Binomial Tree in Excel VBA**

For a multi-period model, all underlying stock prices $S(t)$ at different nodes are defined by

$$S(t) = S_0 \cdot u^j \cdot d^{i-j}$$

where $S(t) = n \cdot \text{delta}_t$, $\text{delta}_t = T / \text{number of intervals}$

[1]

Payoff of an option is max ($S(T) - K, 0$) at expiry [2]

We bring risk-neutrality and no-arbitrage at this point. Given that the payoff of an option is max ($S(T) - K, 0$) at expiry, we can calculate payoffs for the option at expiry. But these prices are for many different nodes at expiry, depending on the number of time intervals in the model.

If we are at any particular node before expiry and consider the portfolio consisting of a shorted option and a long amount of stock, delta, that has the same value whether the stock price moves up or down in the next period. If we assume no arbitrage possibilities, then the price of the option at the node must equal the expected value of the option discounted by the risk-free rate. The expected value is a vector product of the option prices of the node following up and down stock price movements, and the probability of up and down movements.

On each node the option price is given by the following equation:

$$f(i, j) = \exp (-r \cdot \text{delta}_t) \cdot [p \cdot f(i+1, j+1) + (1-p) \cdot f(i+1, j)]$$

[3]

where $\text{delta}_t$ is the length of the interval during which we assume a single stock price movement, $f(i, j)$ is the option price at time $t = i \cdot \text{delta}_t$, $i$ is the time interval, $j$ is the number of upward price movements, $f(i+1, j)$ is the option price at time $t=t+1$ when there is a downward stock movement at time $t$, $p$ is the probability of an upward stock movement at time $t$ [for $j = N$ to 1 step -1]

$$u = \exp (\sigma \cdot \text{sqr}($$delta_t$$))$$

[4]

$$d = \exp (-\sigma \cdot \text{sqr}($$delta_t$$))$$

[5]

$$a = \exp(r \cdot \text{delta}_t)$$

[6]

$$p = (a-d)/(u-d)$$

[7]

We calculate the payoffs and option prices at expiry using formulae [1], [2], [4], [5] before working backwards using formula [3], which in turn requires inputs from formulae [4]-[7] which are derived from Cox, Ross and Rubenstein.

For an **American option**, early exercise is a possibility. This will occur, for a call option whenever the payoff of the option upon immediate exercise is greater than the option price calculated from no arbitrage assumptions.

When coding a binomial tree we start at the last time interval and work backwards. This means a downward counter. In this case I use a for loop with a step -1 indicator to show that the counter is moving backwards. There are two loops: an outer loop for the time period, and an inner loop for the number of upward movements. So we look at each interval in turn and calculate its stock and option prices using the formulae above.

[function declaration]
[define stock price and option prices as 2 dimension arrays depending on indices i,j - time interval and number of option movements, respectively]
[calculate parameters delta_t, u, d, a, p]
[for i =N to 1 step -1]
How to build an Equity Option Pricer using the Binomial Tree in Excel VBA

**Binomial Tree Excel VBA Code**

Function binomial_tree(S, K, t, r, sigma, N)

Dim delta_t As Double
Dim u As Double
Dim d As Double
Dim j As Integer 'number of up movements on the tree
Dim i As Integer 'number of time intervals that have passed
Dim a As Double
Dim call_price() As Double
Dim put_price() As Double
Dim p As Double

delta_t = t / N
u = Exp(sigma * Sqr(delta_t))
d = -Exp(sigma * Sqr(delta_t))
a = Exp(r * delta_t)
p = (a - d) / (u - d)

ReDim call_price(1 To N, 1 To N)
ReDim put_price(1 To N, 1 To N)

For j = N To 1 Step -1

call_price(N, j) = WorksheetFunction.Max(S * u ^ j * d ^ (N - j) - K, 0)
put_price(N, j) = WorksheetFunction.Max(K - S * u ^ j * d ^ (N - j), 0)

[calculate stock price given j upward movements]
[calculate option payoff if i=N]
[when i<>N calculate option payoff(i,j) as function of option price(i+1, j+1), option price (i+1,j), p, r, delta_t]
[discount option payoffs to calculate option price(i,j) for any i]
[next j]

[next i]
How to build an Equity Option Pricer using the Binomial Tree in Excel VBA

```vba
For i = N - 1 To 1 Step -1
    For j = N - 1 To 1 Step -1
        call_price(i, j) = Exp(-r * delta_t) * (p * call_price(i + 1, j + 1) + (1 - p) * call_price(i + 1, j))
        put_price(i, j) = Exp(-r * delta_t) * (p * put_price(i + 1, j + 1) + (1 - p) * put_price(i + 1, j))
    Next j
    Next i

binomial_tree = Array(call_price, put_price)

End Function
```

Next Steps

- The Cox, Ross, Rubenstein binomial tree’s convergence rate can be improved upon. Jarrow and Reed propose a jump diffusion process and corresponding parameters which create a faster convergence to the analytical Black Scholes solution. Another iteration of the model would incorporate this.
- Even faster convergence to Black Scholes results is attained by the trinomial lattice method, which includes 3 potential asset price movements in the subsequent interval: up down, and same.
- An alternative approach that has been shown to have the best convergence rate of all lattice methods is the adaptive mesh model developed by Figlewski and Gao.

How VBA Developer.net Can Save You Time and money

You can get complete Excel apps from VBA Developer.net containing the code in this document, customisation, VBA development of any Excel, Access and Outlook apps, as well as C# and C++ add-ins and technical documentation.

Visit VBA Developer.net

Examples of VBA documents from VBA Developer.net
<table>
<thead>
<tr>
<th>How to build a Black Scholes VBA Option Pricer</th>
<th>How to build a Black Scholes VBA Option Pricer using Monte Carlo Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>How to build a Black Scholes C# Option Pricer</td>
<td>How To Code the Newton-Raphson Method in Excel VBA</td>
</tr>
<tr>
<td>How to build a Black Scholes VBA Option Pricer for FX Options</td>
<td>How To Model Volatility Smiles, Volatility Term Structure and the Volatility Surface in Excel VBA</td>
</tr>
<tr>
<td>How to build a Black Scholes VBA Option Pricer for Equity Options</td>
<td>How To Write Use Cases for a Portfolio Reporting VBA Tool</td>
</tr>
<tr>
<td>How to build a Black Scholes VBA Option Pricer using Monte Carlo Simulation</td>
<td>How To Write a User Interface Model For a Portfolio Reporting VBA Tool</td>
</tr>
<tr>
<td>How to build a Black Scholes VBA Option Pricer for Binary Options</td>
<td>How To Create a Semantic Object Model For a Portfolio Reporting VBA Tool</td>
</tr>
<tr>
<td>How to build a Black Scholes VBA Option Pricer for Equity Barrier Options</td>
<td>How To Normalise a Database For VBA Apps</td>
</tr>
<tr>
<td>How to build a Black Scholes VBA Option Pricer for Exotic Asian Options</td>
<td>How To Create a Database using SQL Scripts for a Portfolio Reporting VBA App</td>
</tr>
<tr>
<td>How to build a Black Scholes VBA Option Pricer for Exotic Lookback Options</td>
<td>How to Write Stored Procedures in SQL/Access/VBA for a Portfolio Reporting VBA App</td>
</tr>
<tr>
<td>How to build an Equity Option Pricer using the Binomial Tree in Excel VBA</td>
<td>How to Use Cursors in SQL for a Portfolio Reporting VBA Tool</td>
</tr>
</tbody>
</table>
How to build an Equity Option Pricer using the Binomial Tree in Excel VBA

How to code a Choleskey Decomposition in VBA (Numerical Methods for Excel)

How to Move Data from Access to Excel with SQL for a Portfolio Reporting VBA App

Portfolio Reporting VBA Tool: Inserting Data into SQL/Access Databases from Excel

Portfolio Reporting VBA Tool: Connecting Excel with SQL & Access Databases

How To Design Classes For an Option Pricer in VBA: UML Concepts

How To Design Classes for Object Orientated VBA Programming