Coursework 1

Junbiao Li - 209050796

October 26, 2023

1 Question 1

1.1 a) No-Arbitrage Condition

First, we need to verify whether the given (u, d) pairs satisfy the no-arbitrage condition. According to the reference material, the no-arbitrage condition is as follows:

$$d < 1 + \rho < u$$

The monthly interest rate, denoted by ρ_m, can be derived from the annual interest rate represented by ρ, as follows:

$$\rho_m = (1 + \rho)^{1/12} - 1$$

The calculated value is $\rho_m \approx 0.00439$.

For all four pairs of (u, d) values, it is found that they satisfy the no-arbitrage condition $d < 1 + \rho_m < u$.

- For $u = 1.020, d = 0.980$: $0.98 < 1 + 0.00439 < 1.02$ holds true.
- For $u = 1.006, d = 0.985$: $0.985 < 1 + 0.00439 < 1.006$ holds true.
- For $u = 1.025, d = 0.975$: $0.975 < 1 + 0.00439 < 1.025$ holds true.
- For $u = 1.013, d = 0.987$: $0.987 < 1 + 0.00439 < 1.013$ holds true.

Therefore, these factors all satisfy the no-arbitrage condition and can be utilized for subsequent option pricing.

1.2 b) Option Pricing

1.2.1 Calculating q-Probabilities

For each (u, d) pair, there are corresponding risk-neutral probabilities q_u and q_d. These probabilities calculated as follows:

$$q_u = \frac{1 + \rho_m - d}{u - d}, \quad q_d = 1 - q_u$$

The calculated values are:

1. For $u = 1.020, d = 0.980$: $q_u = 0.610, q_d = 0.390$
2. For $u = 1.006, d = 0.985$: $q_u = 0.923, q_d = 0.077$
3. For $u = 1.025, d = 0.975$: $q_u = 0.588, q_d = 0.412$
4. For $u = 1.013, d = 0.987$: $q_u = 0.669, q_d = 0.331$

1.2.2 Time 0 Option Price of the European Call Option.

Recall that:

$$\text{call option claim } = (S_T - k)+$$

After calculating the q-probabilities, we need to use these probabilities and the (u, d) pairs to step through the option values for each path saparately using two for loops(shown in Code 1). Given a specific path that consists of a series of u and d, we can calculate the stock price at the end of the path S_T using the following formula:

$$S_T = S_0 \times \prod_{i=1}^{20} Y_i$$

Where Y_i is either u or d at the i th moment in the path.

The value of the option under the path is then calculated using the following formula:

$$\text{Option Value at } S_T = \max(S_T - K, 0)$$

Finally, we need to weight these option values using risk-neutral probabilities and discount them back to the $t = 0$ time using the following formula:

Code Block 1: Use two for loops to iterate through all possible paths.

```
V0_normal = 0;
% Generate all possible paths (each path is a sequence of u's and d's)
all_paths = dec2bin(0:(2^n_periods - 1), n_periods) - '0';
% Iterate through all paths to calculate their contributions to the option ...
    price
for i = 1:size(all_paths, 1)
    path = all_paths(i, :);
    S = S0;
    path_prob = 1;  % Initialize the probability of this path occurring
    % Calculate the final stock price along this path
    for t = 1:length(path)
        index = mod(t - 1, 4) + 1;  % Determine which (u, d) pair to use
        u = ud_pairs(index, 1);
        d = ud_pairs(index, 2);
        q_u = q_probabilities(index, 1);
        q_d = q_probabilities(index, 2);
        % Update the stock price and path probability
        if path(t) == 1
            S = S * u;
            path_prob = path_prob * q_u;
        else
            S = S * d;
            path_prob = path_prob * q_d;
        end
    end
    % Calculate the option value for this path
    option_value = max(S - K, 0) * path_prob;
    % Update V0 for the normal option
    V0_normal = V0_normal + option_value;
end
```

$$V_0 = \frac{\sum (\text{Option Value at } S_T \times \text{Probability of the Path})}{(1+\rho_m)^{20}}$$

In traversing all possible paths using two loops, we calculated the option value for each path following the steps above. By aggregating the weighted option values of all the paths and discounting them back to $t = 0$, **we obtain the $t = 0$ time-unarbitrage-free price of the European Call Option $V_0 = 0.2641$.**

The complete MATLAB code is shown in Appendix A.

Code Block 2: Add one if statement to check if the stock price S_T at the end of each path is greater than or equal to L.

```
V0_barrier = 0;
% Iterate through all paths to calculate their contributions to the option ...
    price
for i = 1:size(all_paths, 1)
    %... same as the normal option
    % Calculate the final stock price along this path
    for t = 1:length(path)
        %... same as the normal option
    end
    %... same as the normal option
    % Check if this path leads to a stock price ≥ L at maturity
    if S ≥ L
        V0_barrier = V0_barrier + option_value;
    end
end
```

1.3 c) Price of the Barrier Option

Based on the calculation of the price of the regular option, the Barrier Option can only be exercised if the stock price reaches or exceeds a given level L at expiration. Therefore, we add an 'if' judgment statement which is in Code Block 2 to the code to check if the stock price S_T at the end of each path is greater than or equal to L. Only in this case will the option value of the path be included in the calculation of the barrier option price V_0 at the time of $t = 0$.

Ultimately, we can get the Price of the Barrier Option V_0 is found to be 0.2145

1.4 d) Price of the Lookback Put Option

To calculate the price of this option, we must traverse all potential price paths and assess each one. In particular, we are required to:

1. Track the maximum price ($S_{\max}$) of the underlying asset along each path.

2. Compute the price of the underlying asset at maturity (S_T).

3. Use $S_{\max}$ and S_T to calculate the option value for each path which is

$$\text{Option Value at } S_T = \max(S_{\max} - S_T, 0)$$

4. Sum the weighted option values for all paths and discount back to $t = 0$.

We weight these option values in the same way as regular options, using q-probabilities, and discount back to $t = 0$. **In this way, we obtain the no-arbitrage price $V_0 = 0.1251$ of the lookback put option at time $t = 0$.**

A MATLAB Code

Code Block 3: Complete MATLAB Code

```
% Define ud_pairs array
ud_pairs = [
    1.02, 0.98;
    1.006, 0.985;
    1.025, 0.975;
    1.013, 0.987
];

% Given parameters
annual_interest_rate = 0.054;  % Annual interest rate of 5.4%
S0 = 5.35;  % Initial stock price
K = 5.65;  % Strike price of the European call option
n_periods = 20;  % Number of periods in the binomial tree
L = 6;

% Calculate the monthly interest rate
monthly_interest_rate = (1 + annual_interest_rate)^(1/12) - 1;
q_probabilities = (1 + monthly_interest_rate - ud_pairs(:,2)) ./ ...
    (ud_pairs(:,1) - ud_pairs(:,2));
q_probabilities = [q_probabilities, 1 - q_probabilities];

% Initialize V0 for the normal, barrier, and lookback put options
V0_normal = 0;
V0_barrier = 0;
V0_lookback_put = 0;

% Generate all possible paths (each path is a sequence of u's and d's)
all_paths = dec2bin(0:(2^n_periods - 1), n_periods) - '0';

% Iterate through all paths to calculate their contributions to the option ...
    price
for i = 1:size(all_paths, 1)
    path = all_paths(i, :);
    S = S0;
    S_max = S0;  % Initialize the maximum stock price for this path
    path_prob = 1;  % Initialize the probability of this path occurring
```

```
    % Calculate the final stock price along this path
    for t = 1:length(path)
        index = mod(t - 1, 4) + 1;  % Determine which (u, d) pair to use
        u = ud_pairs(index, 1);
        d = ud_pairs(index, 2);
        q_u = q_probabilities(index, 1);
        q_d = q_probabilities(index, 2);

        % Update the stock price and path probability
        if path(t) == 1
            S = S * u;
            path_prob = path_prob * q_u;
        else
            S = S * d;
            path_prob = path_prob * q_d;
        end

        % Update the maximum stock price for this path
        S_max = max(S, S_max);
    end

    % Calculate the option value for this path
    option_value = max(S - K, 0) * path_prob;

    % Update V0 for the normal option
    V0_normal = V0_normal + option_value;

    % Check if this path leads to a stock price ≥ L at maturity
    if S ≥ L
        V0_barrier = V0_barrier + option_value;
    end

    % Calculate the lookback put option value for this path
    lookback_put_value = max(S_max - S, 0) * path_prob;

    % Update V0 for the lookback put option
    V0_lookback_put = V0_lookback_put + lookback_put_value;
end

% Discount the option price back to time 0
V0_normal = V0_normal / ((1 + monthly_interest_rate) ^ n_periods);
V0_barrier = V0_barrier / ((1 + monthly_interest_rate) ^ n_periods);
V0_lookback_put = V0_lookback_put / ((1 + monthly_interest_rate) ^ ...
    n_periods);

% Display the results
V0_normal
V0_barrier
V0_lookback_put

% Expected output:
% V0_normal = 0.2641
% V0_barrier = 0.2145
% V0_lookback_put = 0.1251
```