MA 374 (2021) Financial Engineering Lab Lab 03

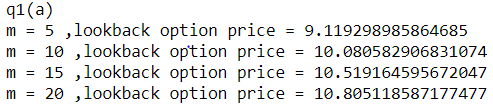
**Name:** Udandarao Sai Sandeep

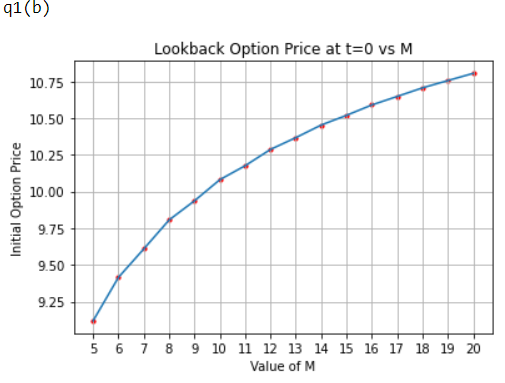
**Roll Number:** 180123063

**Dept.:** Mathematics and Computing

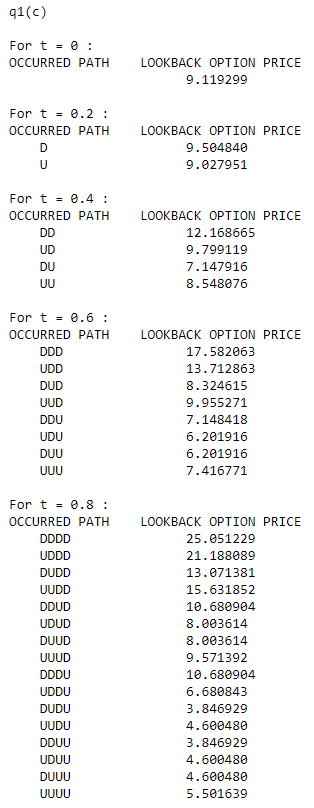
**Q1.** The standard brute force solution for finding the lookback option involves traversing over all known paths, hence leading to exponential time complexities (**2M**). Python was able to calculate the initial lookback option prices only when **M ≤ 20**. Higher values of M exceeded computational capacities of Jupyter Python notebook. The formula used for calculating the option price is as follows:

The values of initial lookback option prices are as follows:



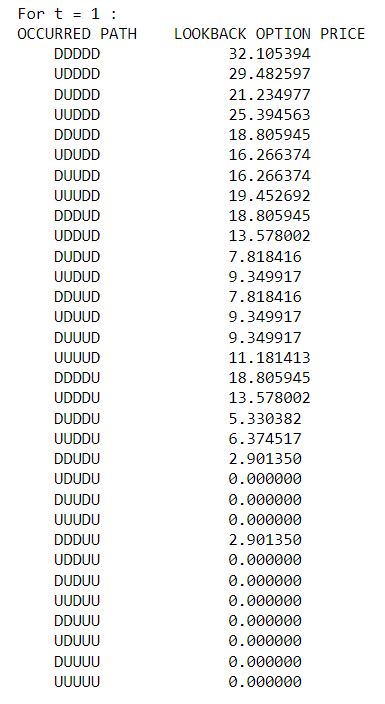


For **Q1(b)**, the initial lookback option prices were calculated for the values of **M** between the **range 5 to 20**. A graph of **Option Price vs M** was plotted out. It can be seen that with **increasing value of M**, the **option price also increasing**. However, the graph followed a **concave downward** pattern (a **negative second order derivative**), indicating that as the value of M increases, the curve will eventually flatten out.

For Q1(c), The value of **M has been set to 5.**

For each value of **t**, the **number of remaining time intervals** and **the number of time intervals that have already occurred** were calculated. All possible case paths till time **t** were taken into consideration.

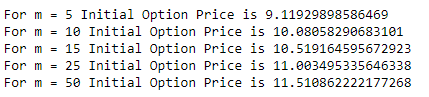
For each possible case, the maximum stock price up till time **t** and **S(t)** was calculated. Considering this as the initial stock price, the option price was calculated (for the remaining time period).

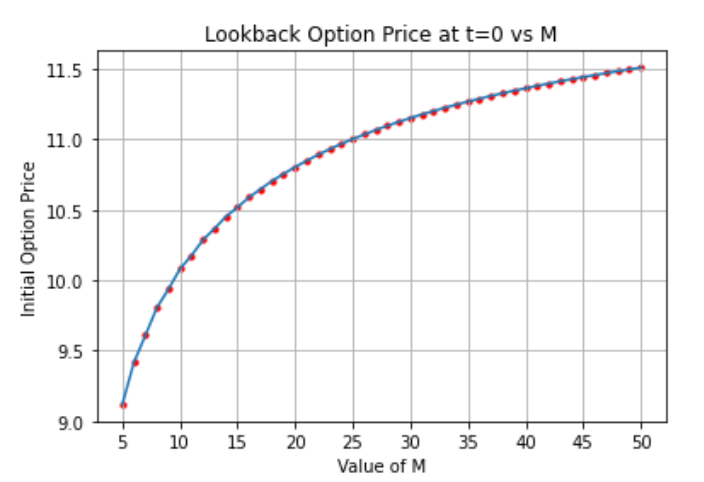
The results obtained are as follows:

**Q2.**

For Q2, an efficient **Markov-based binomial algorithm** (as seen in Shreve) was employed to calculate the Lookback Option Prices for various paths. A **recursive algorithm** was used, which keeps track of the **maximum Stock Price** till now, and the **current Stock Price** as different paths are being explored. A **storage data structure (dictionary)** was used to memorize the answers at each stage, in order to avoid repetitive calculations. The time complexity of the program was reduced form **O(2n)** to **Polynomial Time Complexity.**

The values of Option Prices obtained are as follows:





For **Q2**, the initial lookback option prices were calculated for the values of **M** between the **range 5 to 50**. A graph of **Option Price vs M** was plotted out. It can be seen that with **increasing value of M**, the **option price also increasing**. However, the graph followed a **concave downward** pattern (a **negative second order derivative**), indicating that as the value of M increases, the curve will eventually flatten out.

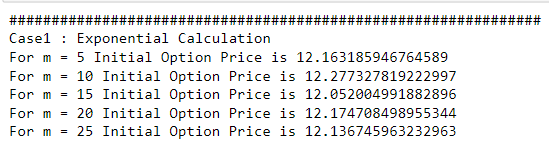
Comparison Between Algorithms used in **Q1** vs **Q2**:

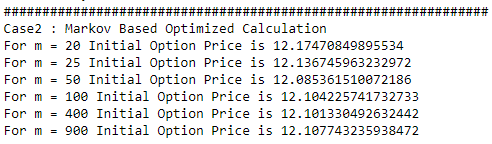
|  |  |  |
| --- | --- | --- |
|  | Straightforward Exponential Algorithm | Markov-based Binomial Algorithm |
| Values of M it can handle | **1 ≤ M ≤ 20** | **1 ≤ M ≤ 50** |
| Time Complexity | **O(2M)** | **Polynomial Time Complexity** |
| Time Taken |  |  |

**Q3.**

In order to calculate the Initial European Call Option Price, two different Algorithms were employed. First was the **straightforward exponential** (all possible paths explored) having a time complexity of **O(2M)**. The second algorithm employs a **Markov based efficient binomial algorithm** (involving a **recursive** approach). In this approach, a separate data structure was used to **memorize** already calculated answers to avoid recessive function calls. This algorithm has a time complexity of **O(M2)**.

The values of the Option Prices calculated through both methods are as follows:





Comparison Between Algorithms used in **Q1** vs **Q2**:

|  |  |  |
| --- | --- | --- |
|  | Straightforward Exponential Algorithm | Markov-based Binomial Algorithm |
| Values of M it can handle | **1 ≤ M ≤ 25** | **1 ≤ M ≤ 2000** |
| Time Complexity | **O(2M)** | **O(M2)** |
| Time Taken |  |  |