MA 374 – Financial Engineering Lab

Lab – 3

Name - Vishisht Priyadarshi Roll No - 180123053

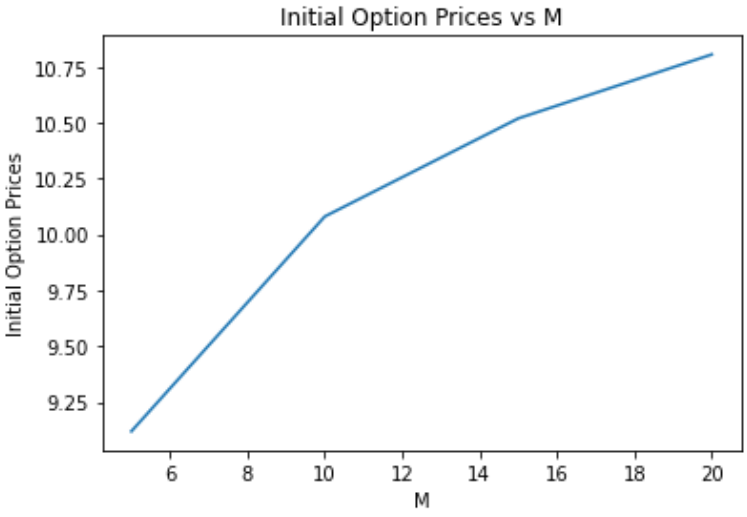
1 QUESTION - 1:

(a) The initial option prices for the Loopback Option are:

SI No	M	Initial Option Price	Execution Time (in sec)
1.	5	9.119298985864683	0.0004768371582031 25
2.	10	10.080582906831	0.0038139820098876 95
3.	25	11.00349533564633	217.85930705070496
4.	26	11.036041506227305	450.0798707008362
5.	27	11.067921312247574	928.2180182933807
6.	50	**Not feasible**	**Not feasible**

For M = 50, the algorithm is unable to calculate the option price since it has exponential time complexity. For M = 26 and 27, we can see that as we increase M by single unit, the execution time almost doubles. So, computation for M = 50 is not feasible with this algorithm for finding out the loopback option.

(b) The plots comparing the above values are as follows:



As the value of M increases, the initial option price increases, and it seems that the prices tend to converge as M is increased further.

(c) The values of the options at all intermediate time points for M = 5:
(<u>Note:</u> 't' shows time intervals wrt M)

	t = 0	**t = 1**	**t = 2**	**t = 3**	**t = 4**	**t = 5**
1.	9.12	9.03	8.55	7.42	5.50	0.00
2.	x	9.50	9.78	9.96	9.57	11.18
3.	x	x	7.15	6.20	4.60	0.00
4.	x	x	12.17	13.71	15.63	19.45
5.	x	x	x	6.20	4.60	0.00
6.	x	x	x	8.32	8.00	9.35
7.	x	x	x	7.15	6.68	6.37
8.	x	x	x	17.58	21.19	25.39
9.	x	x	x	x	4.60	0.00
10.	x	x	x	x	8.00	9.35
11.	x	x	x	x	3.85	0.00
12.	x	x	x	x	13.07	16.27
13.	x	x	x	x	3.85	0.00
14.	x	x	x	x	10.68	13.58
15.	x	x	x	x	10.68	13.58
16.	x	x	x	x	25.05	29.48
17.	x	x	x	x	x	0.00
18.	x	x	x	x	x	9.35
19.	x	x	x	x	x	0.00
20.	x	x	x	x	x	16.27
21.	x	x	x	x	x	0.00
22.	x	x	x	x	x	7.82
23.	x	x	x	x	x	5.33
24.	x	x	x	x	x	21.23
25.	x	x	x	x	x	0.00
26.	x	x	x	x	x	7.82
27.	x	x	x	x	x	2.90
28.	x	x	x	x	x	18.81
29.	x	x	x	x	x	2.90
30.	x	x	x	x	x	18.81
31.	x	x	x	x	x	18.81
32.	x	x	x	x	x	32.11

2 QUESTION - 2:

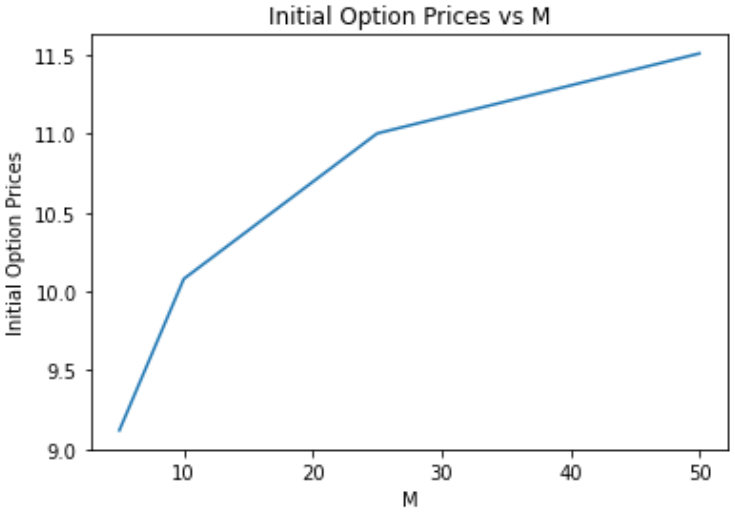
Using Markov property, we arrive at the following relation for finding out the loopback option price at t = 0:

$$v_n(s,m) = \frac{1}{R}\left[p.v_{n+1}(u.s, \max(u.s,m)) + (1-p).v_{n+1}(d.s,m)\right]$$

(a) The initial option prices for the Loopback Option are:

SI No	M	Initial Option Price	Execution Time (in sec)
1.	5	9.119298985864683	0.00028252601623535156
2.	10	10.080582906831	0.0011289119720458984
3.	25	11.00349533564633	0.06793522834777832
4.	50	11.510862222177286	4.077192544937134

(b) The plots comparing the above values are as follows:



As the value of M increases, the initial option price increases, and it seems that the prices tend to converge as M is increased further.

(c) The values of the options at all intermediate time points for M = 5 (each state is characterised by a tuple denoting - (i) stock price at that instant, and (ii) maximum stock price among all the stock prices in the current path):

(Note: 't' shows time intervals wrt M)

t = 0 :

	Intermediate state	Option Price
1.	(100, 100)	9.119298985864683

t = 1 :

	Intermediate state	Option Price
1.	(110.676651999383, 110.676651999383)	9.027951165547751
2.	(92.54800352077254, 100)	9.504839866450853

t = 2:

	Intermediate state	Option Price
1.	(122.49321297792528, 122.49321297792528)	8.548076183576441
2.	(102.42903178906215, 110.676651999383)	9.799118753547026
3.	(102.42903178906214, 102.42903178906214)	7.147915756774744
4.	(85.65132955680926, 100)	12.168664659721792

t = 3:

	Intermediate state	Option Price
1.	(135.57138705044142, 135.57138705044142)	7.416771005131011
2.	(113.3650230595177, 122.49321297792528)	9.955271272957816
3.	(113.3650230595177, 113.3650230595177)	6.201916453882752
4.	(94.79602394643446, 110.676651999383)	13.712862965988533
5.	(113.36502305951768, 113.36502305951768)	6.201916453882752
6.	(94.79602394643445, 102.42903178906214)	8.32461466963314
7.	(94.79602394643445, 100)	7.14841820819012
8.	(79.26859549382432, 100)	17.582062714095418

t = 4:

	Intermediate state	Option Price
1.	(150.04587225655362, 150.04587225655362)	5.501638813873981
2.	(125.46861206060268, 135.57138705044142)	9.571391531700229
3.	(125.46861206060268, 125.46861206060268)	4.600479677676438
4.	(104.91706553244704, 122.49321297792528)	15.631851880479827
5.	(104.91706553244704, 113.3650230595177)	8.003613780975444
6.	(104.91706553244704, 110.676651999383)	6.6808429992566465
7.	(87.73182757949854, 110.676651999383)	21.18808934534565
8.	(125.46861206060267, 125.46861206060267)	4.600479677676438

9.	(104.91706553244703, 113.36502305951768)	8.003613780975444
10.	(104.91706553244701, 104.91706553244701)	3.8469288844156075
11.	(87.73182757949853, 102.42903178906214)	13.071380970928788
12.	(87.73182757949853, 100)	10.68090442602997
13.	(73.36150254849147, 100)	25.051229457037028

t = 5:

	Intermediate state	Option Price
1.	(166.06574787682462, 166.06574787682462)	0.0
2.	(138.86445913876912, 150.04587225655362)	11.181413117784501
3.	(138.8644591387691, 138.8644591387691)	0.0
4.	(116.118695507311, 135.57138705044142)	19.452691543130413
5.	(116.118695507311, 125.46861206060268)	9.349916553291678
6.	(116.11869550731102, 122.49321297792528)	6.374517470614265
7.	(97.09864950286031, 122.49321297792528)	25.39456347506497
8.	(116.11869550731102, 116.11869550731102)	0.0
9.	(97.09864950286031, 113.3650230595177)	16.266373556657385
10.	(97.09864950286031, 110.676651999383)	13.578002496522686
11.	(81.1940548771124, 110.676651999383)	29.48259712227059
12.	(116.11869550731099, 125.46861206060267)	9.349916553291678
13.	(116.11869550731099, 116.11869550731099)	0.0
14.	(97.0986495028603, 113.36502305951768)	16.266373556657385
15.	(116.11869550731097, 116.11869550731097)	0.0
16.	(97.0986495028603, 104.91706553244701)	7.8184160295867144
17.	(97.0986495028603, 102.42903178906214)	5.330382286201839
18.	(81.19405487711239, 102.42903178906214)	21.234976911949744
19.	(97.0986495028603, 100)	2.9013504971397026
20.	(81.19405487711239, 100)	18.805945122887607
21.	(67.89460596146952, 100)	32.10539403853048

Comparative Analysis:

1. The table compares the execution time for the different algorithms:

M	Unoptimised algorithm	Markov based algorithm
5	0.000476837158203125	0.00028252601623535156
10	0.003813982009887695	0.0011289119720458984
25	217.85930705070496	0.06793522834777832
50	Not feasible	4.077192544937134

2. The highest value of M the algorithms can handle:
    a. **<u>Unoptimised Algorithm -</u>** around 25 to 30. The execution time almost doubles up as M is increased by 1 unit. At M = 50, the google colab executing this code crashes due to insufficient RAM.
    b. **<u>Markov based Algorithm -</u>** around 85. At M = 90, the google colab executing this code crashes due to insufficient RAM.

3. The unoptimised algorithm has exponential space complexity, while the Markov based algorithm does not have exponential space complexity, since it utilizes the concept of memoization based on the ideas of dynamic programming.

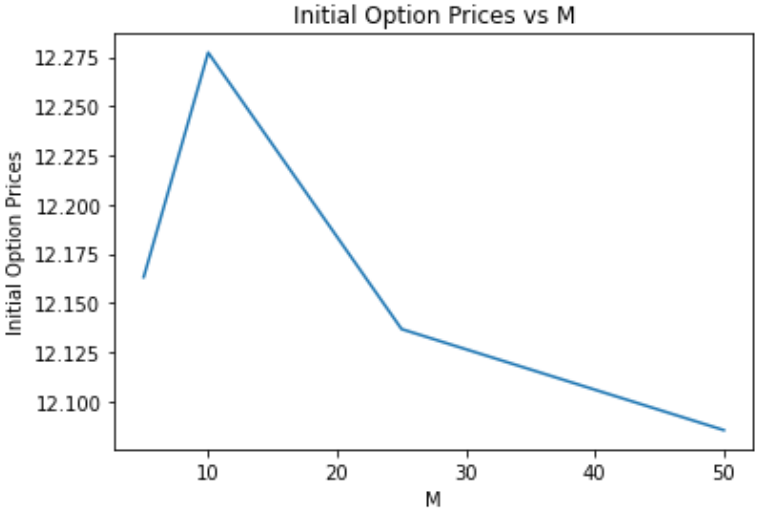
4. Unoptimised algorithm is not practical solution since the time complexity can blow up for small values of M (like 50), while the Markov based algorithm is able to cope up with this.

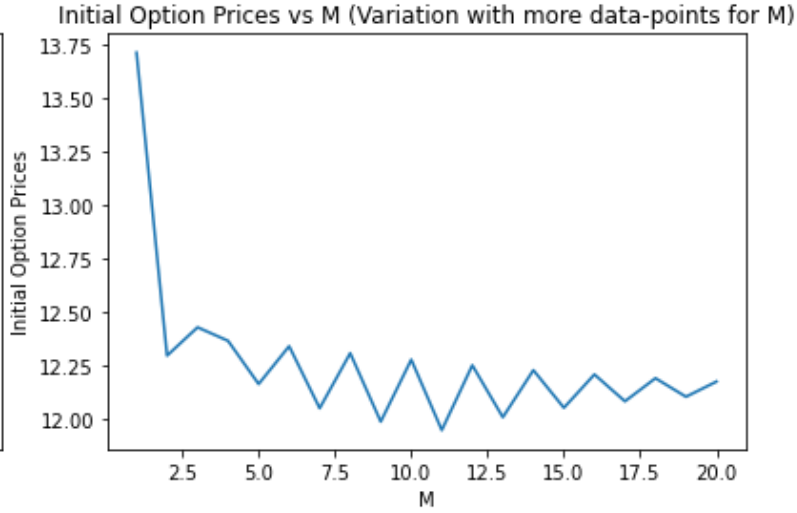
3 QUESTION – 3 :

(a) The initial option prices for the European Call Option are ***(K is taken to be 100)***:

SI No	M	Initial Option Price	Execution Time (in sec)
1.	5	12.163185946764584	8.511543273925781e-05
2.	10	12.277327819222986	9.083747863769531e-05
3.	25	12.136745963232956	0.0001544952392578125
4.	50	12.085361510072197	0.0002987384796142578

(b) The plots comparing the above values are:





The option price decreases and converges around 12.1 (roughly) as M increases.

(c) The values of the options at all intermediate time points for M = 5 is:

(<u>Note:</u> 't' shows time intervals wrt M)

	t = 0	t = 1	t = 2	t = 3	t = 4	t = 5
1.	12.16	18.66	27.53	38.72	51.63	66.07
2.	x	6.06	10.39	17.22	27.06	38.86
3.	x	x	1.92	3.90	7.93	16.12
4.	x	x	x	0.00	0.00	0.00
5.	x	x	x	x	0.00	0.00
6.	x	x	x	x	x	0.00

Comparative Analysis:

1. The table compares the execution time for the different algorithms:

M	Unoptimised	Efficient	Most Efficient
5	0.0002951622 sec	0.0001239776 sec	8.511543273e-05 sec
10	0.0033535957 sec	0.00011634826 sec	9.083747863e-05 sec
25	101.93571686 sec	0.00022888183 sec	0.00015449523 sec
50	Not Feasible	0.00058317184 sec	0.00029873847 sec

2. The highest value of M the algorithms can handle:
   a. **<u>Unoptimised Algorithm -</u>** around 25 to 30. Even at these values, it takes quite some time for algorithm to execute completely. As the value of M increases by 1 unit, execution time almost doubles up.
   b. **<u>Efficient Algorithm –</u>** around $2*10^4$
   c. **<u>Most Efficient Algorithm –</u>** around 1000, with default settings of python. The most efficient algorithm is based on the computation of "nCr (n choose r)". With the standard implementations, values of M > 1000 causes overflow issues. If the implementation is modified (the maximum_recursion_limit of python is modified), then we can compute for values upto M = 1500. But this modification comes with its own risk since it can lead to bugs like stack overflow. So, a safe value of M is around 1000 for this algorithm.

3. The unoptimised algorithm has exponential time and space complexity. The efficient algorithm has quadratic space and time complexity (in M), although space complexity can be decreased to linear. But since we also needed to print intermediate information, I implemented algorithm which has quadratic complexity. The most efficient algorithm has almost linear time complexity (after making use of memoization to calculate nCr), and linear space complexity.

4. The most efficient algorithm works on the same principle as that of efficient algorithm. The only difference lies in the fact that the most efficient algorithm summarises the computation of efficient algorithm to a formula. So, it is much easier to implement, and is faster than the efficient algorithm. But the upper limit of M is less due to computational issues like integer overflow and stack overflow.