MA374: Financial Engineering Lab

Lab03

Dev Sandip Shah

200123074

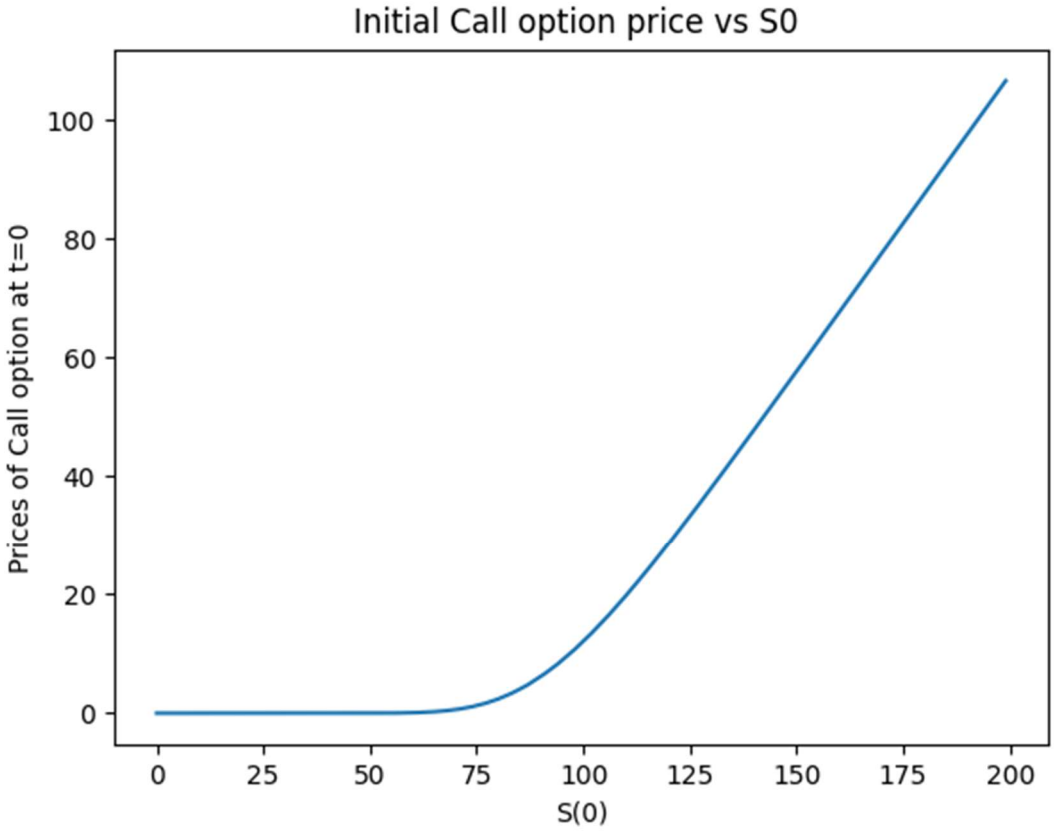
Question 1

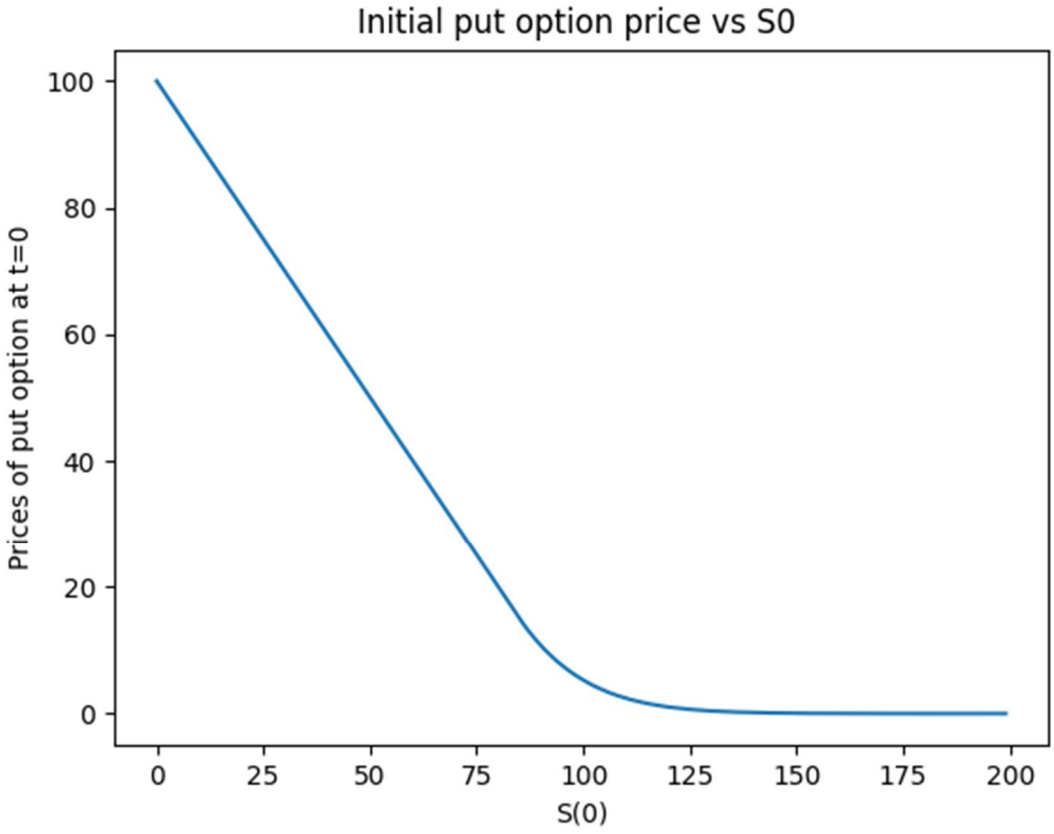
Given expression for u and d are:

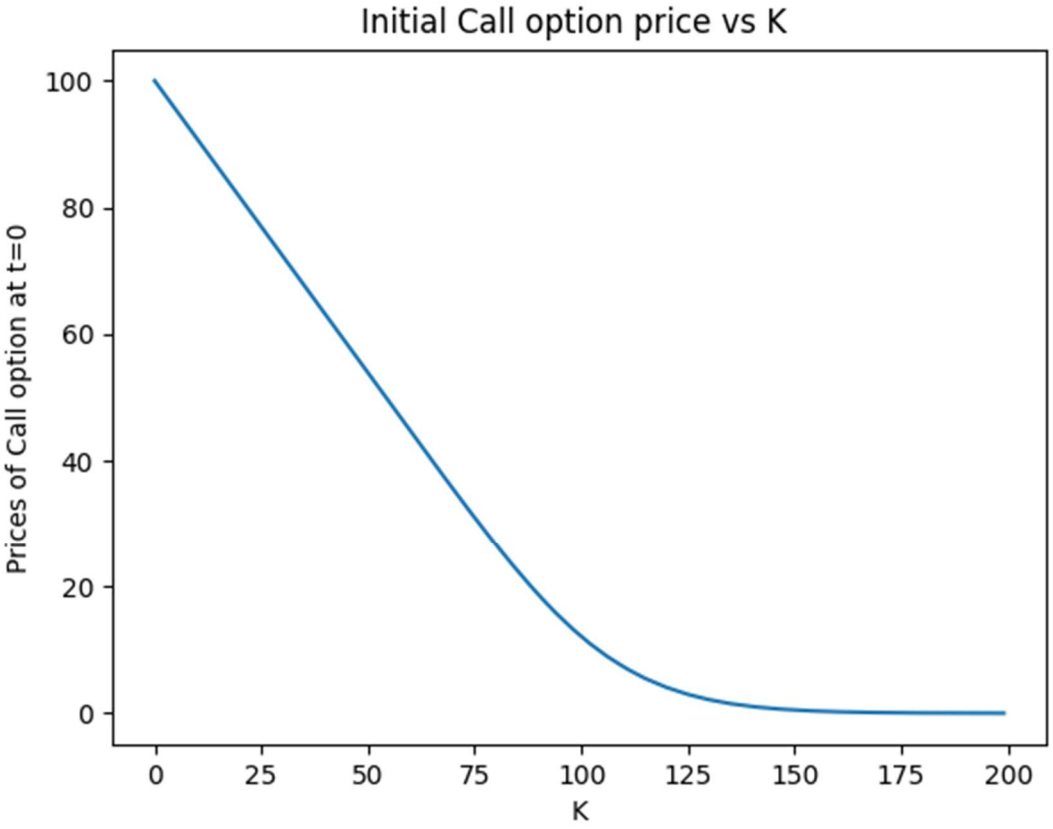
$$u = e^{\sigma\sqrt{\Delta t}+\left(r-\frac{1}{2}\sigma^2\right)\Delta t} \text{ and } d = e^{-\sigma\sqrt{\Delta t}+\left(r-\frac{1}{2}\sigma^2\right)\Delta t}, \quad \text{where } \Delta t = {}^{T}/_{M},$$
$$M \text{ here is the number of subintervals in } [0, T].$$

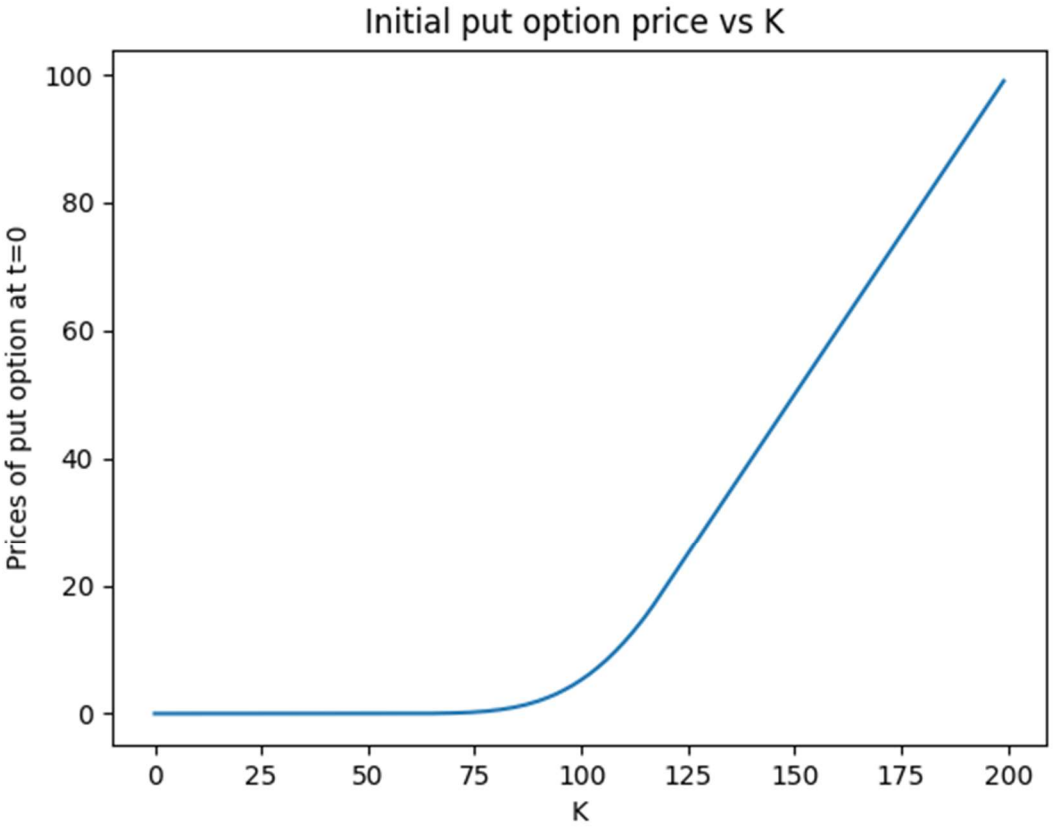
Initial price of the American Call option price = 12.123047074012304 and Put option price = 5.27983714598915.

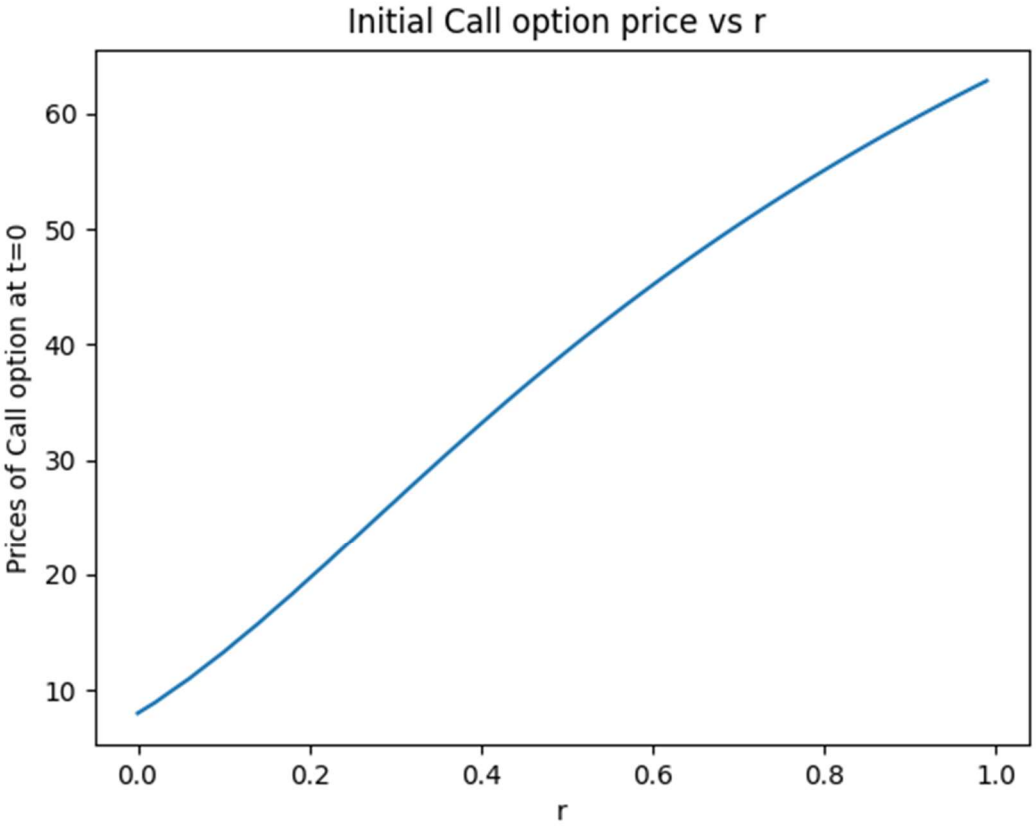
(Graphs in next page)

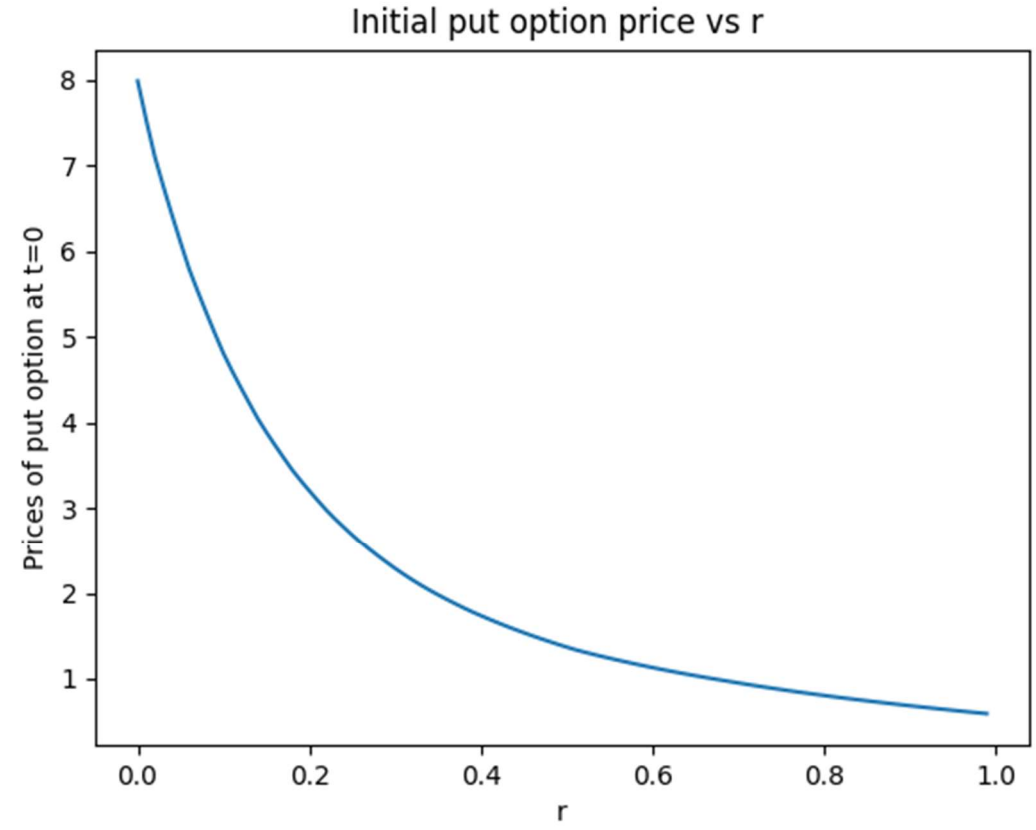

Initial Call option price vs S0
Prices of Call option at t=0
S(0)
0
20
40
60
80
100
0
25
50
75
100
125
150
175
200


Initial put option price vs S0
Prices of put option at t=0
S(0)
0
20
40
60
80
100
0
25
50
75
100
125
150
175
200

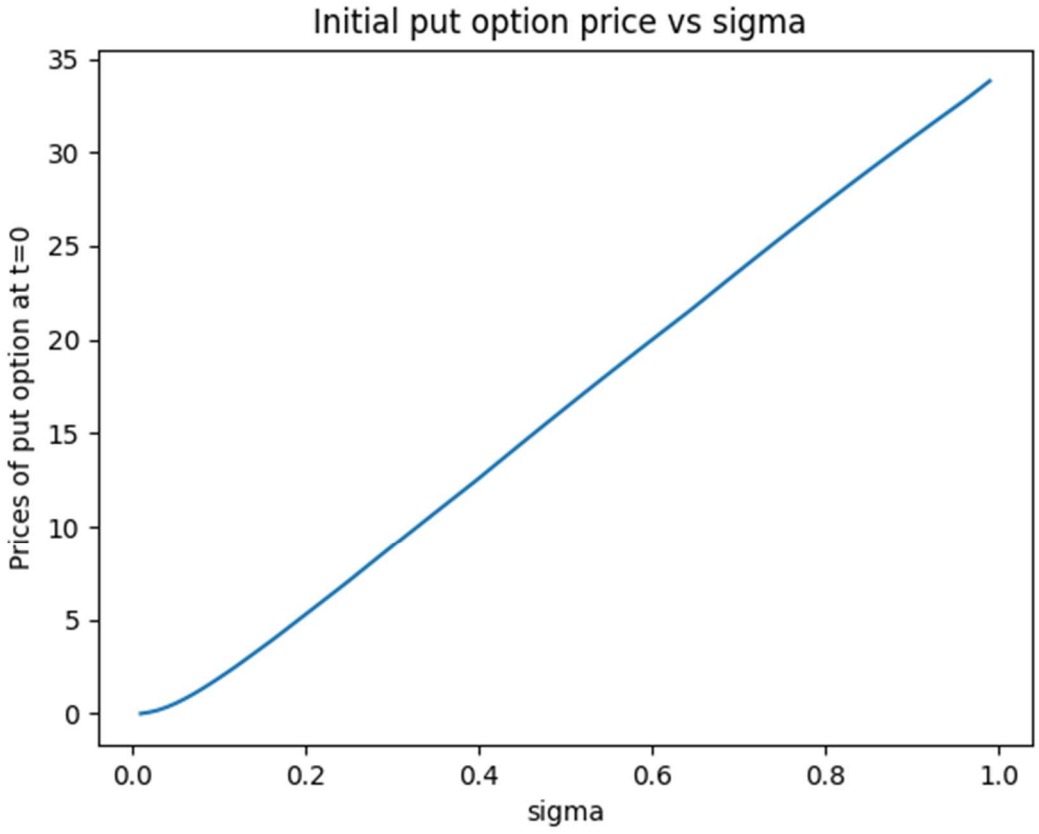

Initial Call option price vs K
Prices of Call option at t=0
100
80
60
40
20
0
0
25
50
75
100
125
150
175
200
K

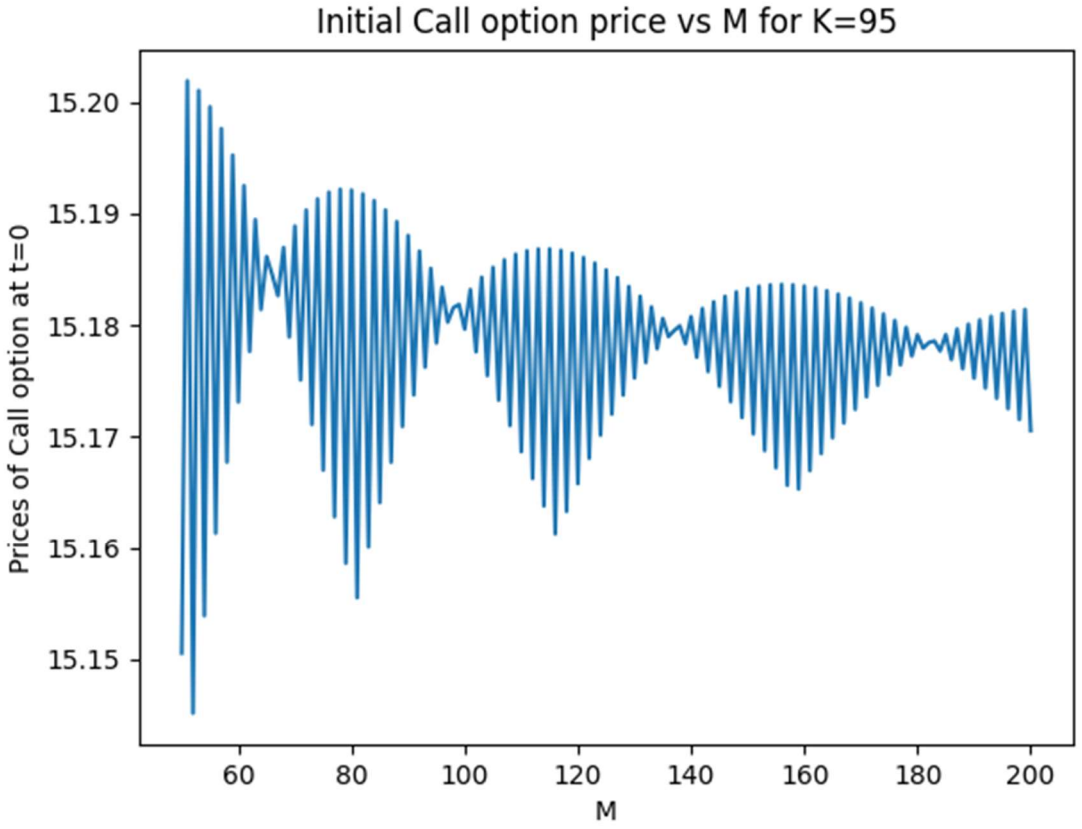

Initial put option price vs K
Prices of put option at t=0
100
80
60
40
20
0
0
25
50
75
100
125
150
175
200
K

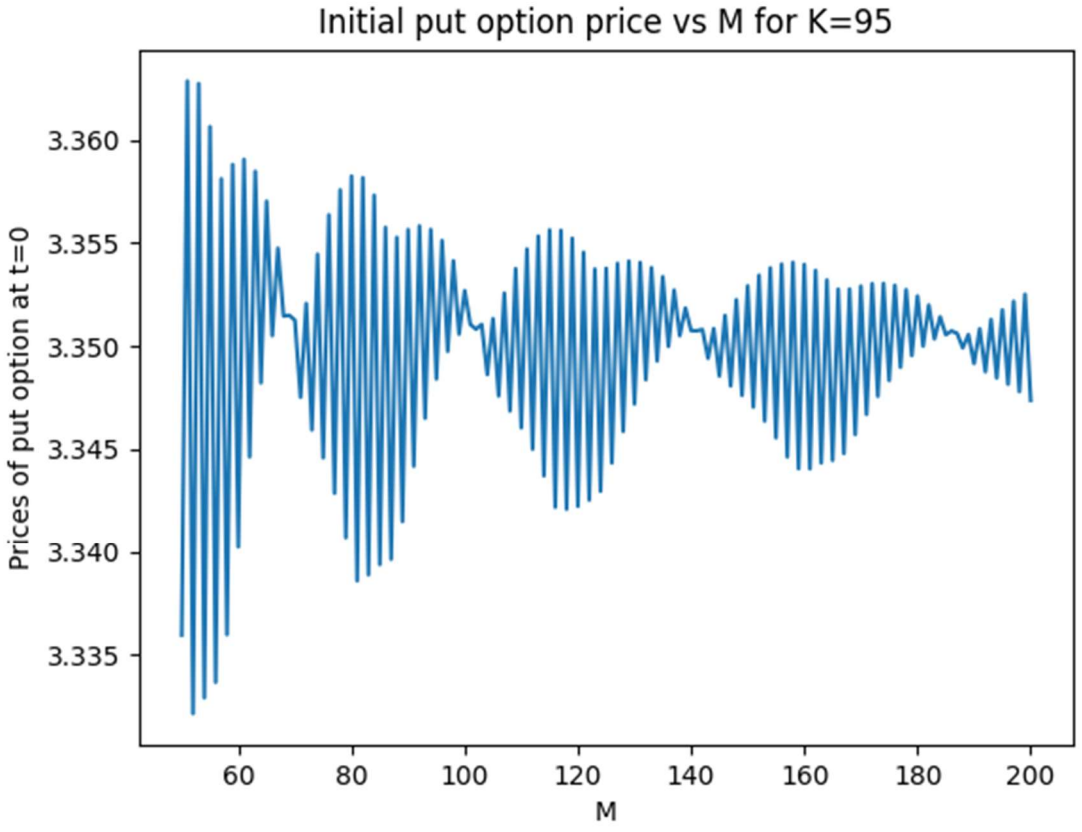

Initial Call option price vs r
Prices of Call option at t=0
60
50
40
30
20
10
0.0
0.2
0.4
0.6
0.8
1.0
r

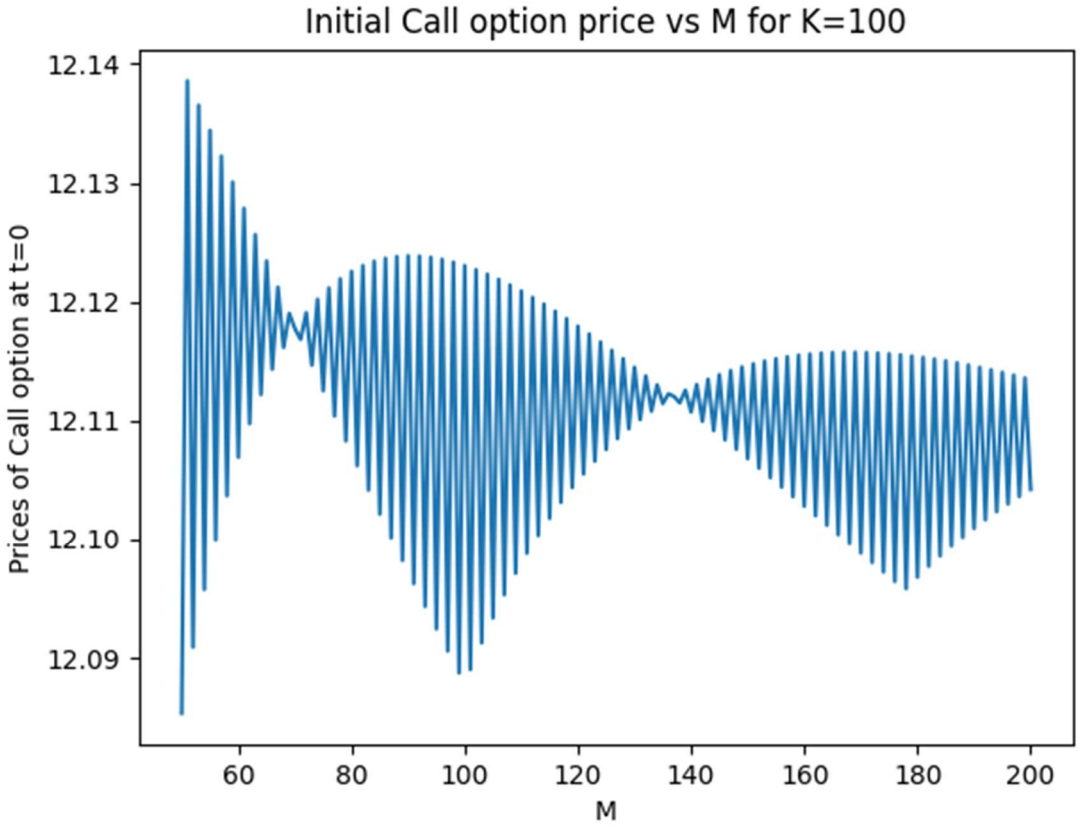

Initial put option price vs r
Prices of put option at t=0
8
7
6
5
4
3
2
1
0.0
0.2
0.4
0.6
0.8
1.0
r

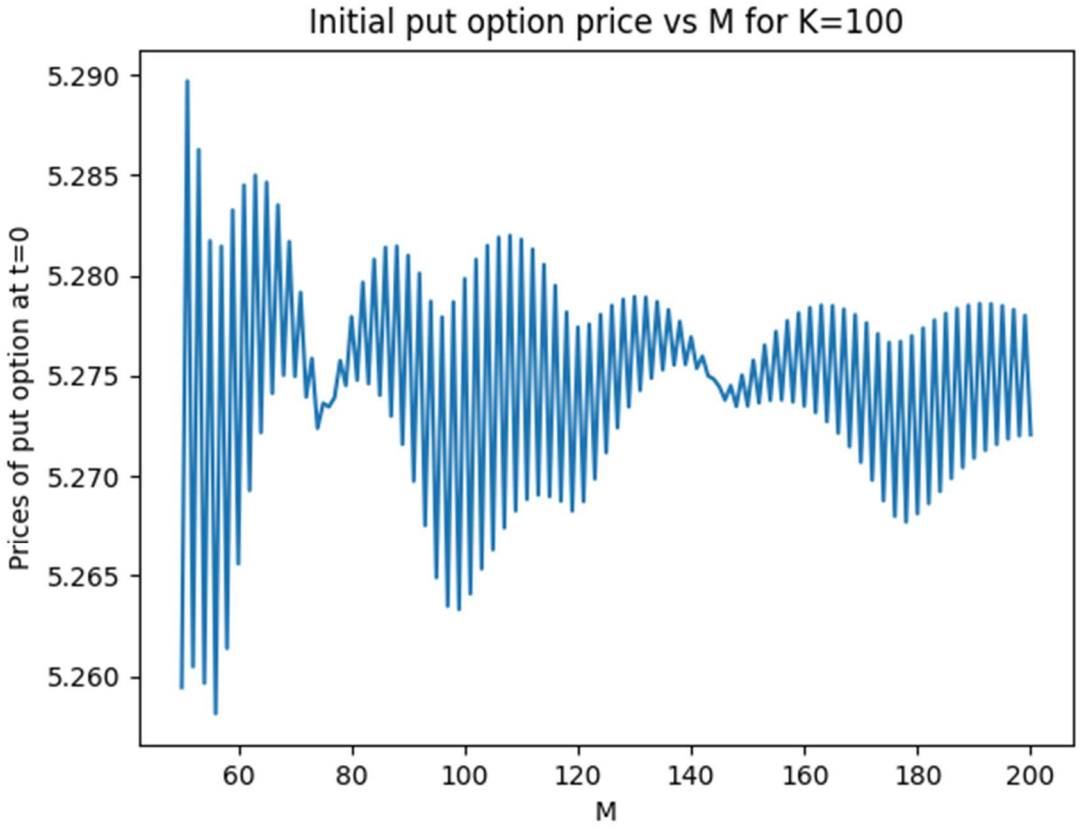

Initial Call option price vs sigma
Prices of Call option at t=0
40
35
30
25
20
15
10
0.0
0.2
0.4
0.6
0.8
1.0
sigma

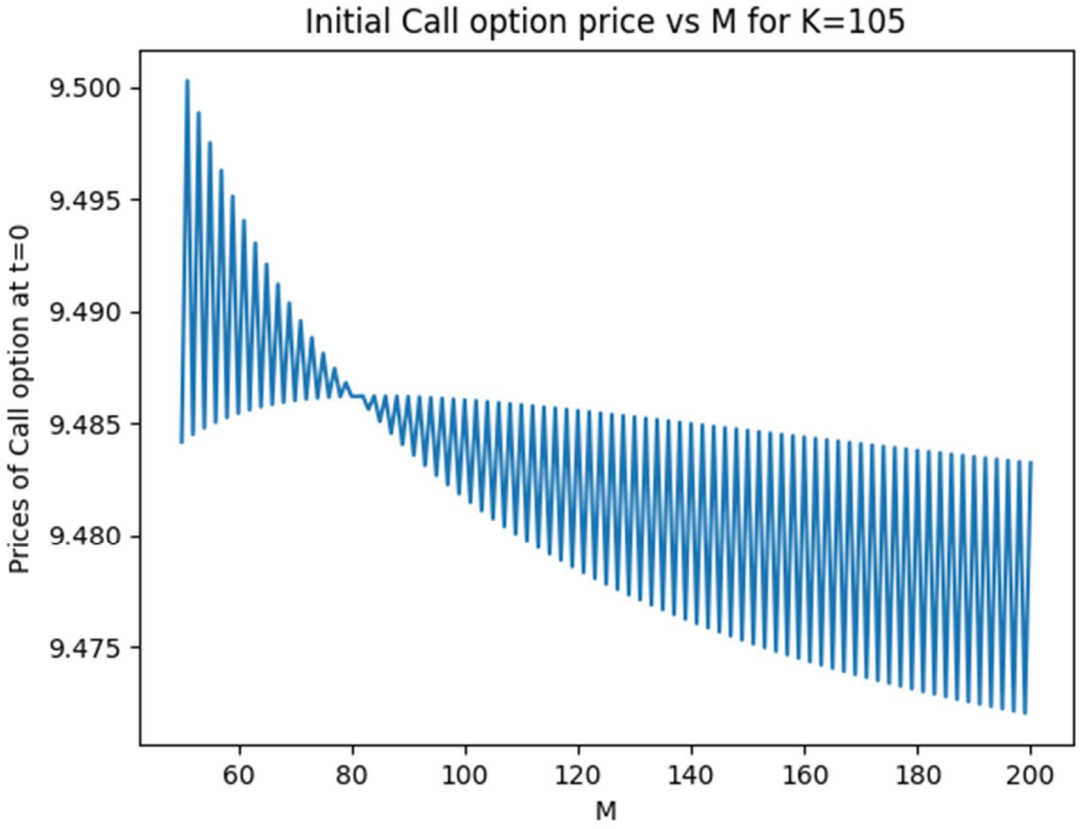

Initial put option price vs sigma
Prices of put option at t=0
35
30
25
20
15
10
5
0
0.0
0.2
0.4
0.6
0.8
1.0
sigma

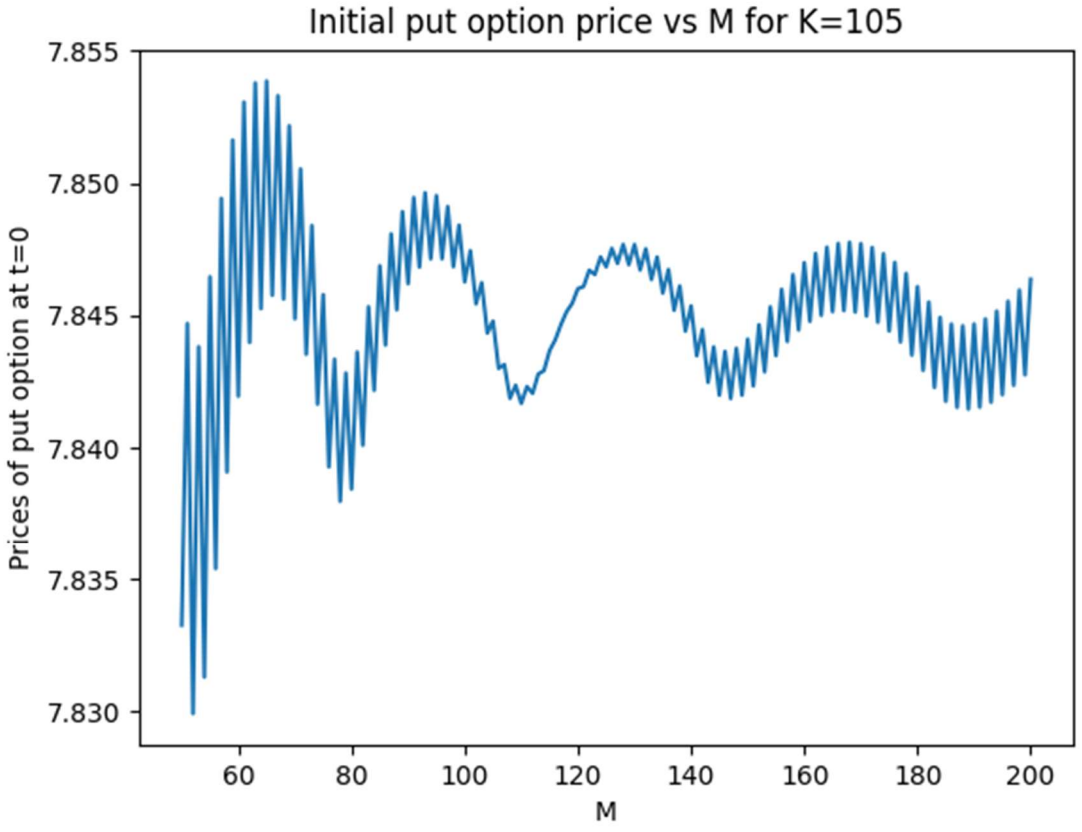

Initial Call option price vs M for K=95
Prices of Call option at t=0
15.20
15.19
15.18
15.17
15.16
15.15
60
80
100
120
140
160
180
200
M


Initial put option price vs M for K=95
Prices of put option at t=0
3.360
3.355
3.350
3.345
3.340
3.335
60
80
100
120
140
160
180
200
M


Initial Call option price vs M for K=100
Prices of Call option at t=0
12.14
12.13
12.12
12.11
12.10
12.09
60
80
100
120
140
160
180
200
M


Initial put option price vs M for K=100
Prices of put option at t=0
5.290
5.285
5.280
5.275
5.270
5.265
5.260
60
80
100
120
140
160
180
200
M


Initial Call option price vs M for K=105
Prices of Call option at t=0
9.500
9.495
9.490
9.485
9.480
9.475
60
80
100
120
140
160
180
200
M


Initial put option price vs M for K=105
Prices of put option at t=0
7.855
7.850
7.845
7.840
7.835
7.830
60
80
100
120
140
160
180
200
M

Question 2

For the European Option, we use the following data,

$$S(0) = 100, T = 1, r = 8\%, \sigma = 20\%$$

The payoff of the lookback option is given as,

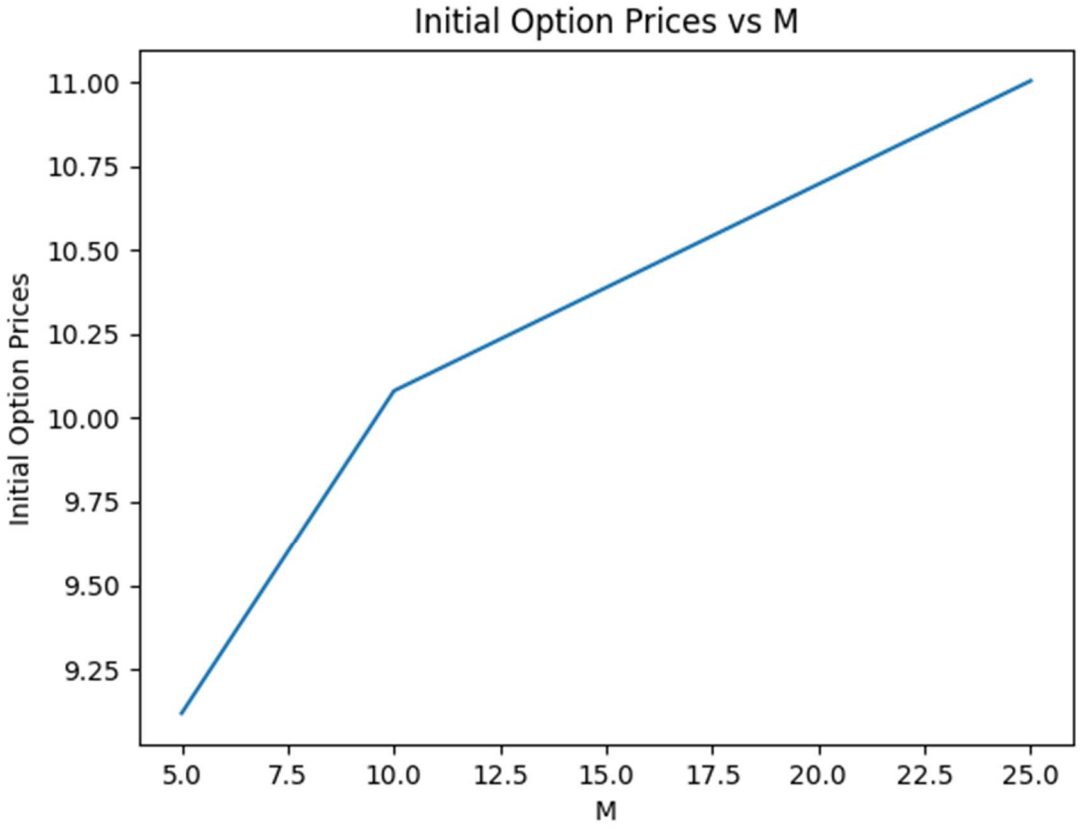
$$V = \max_{0 \le i \le M} S(i) - S(M)$$

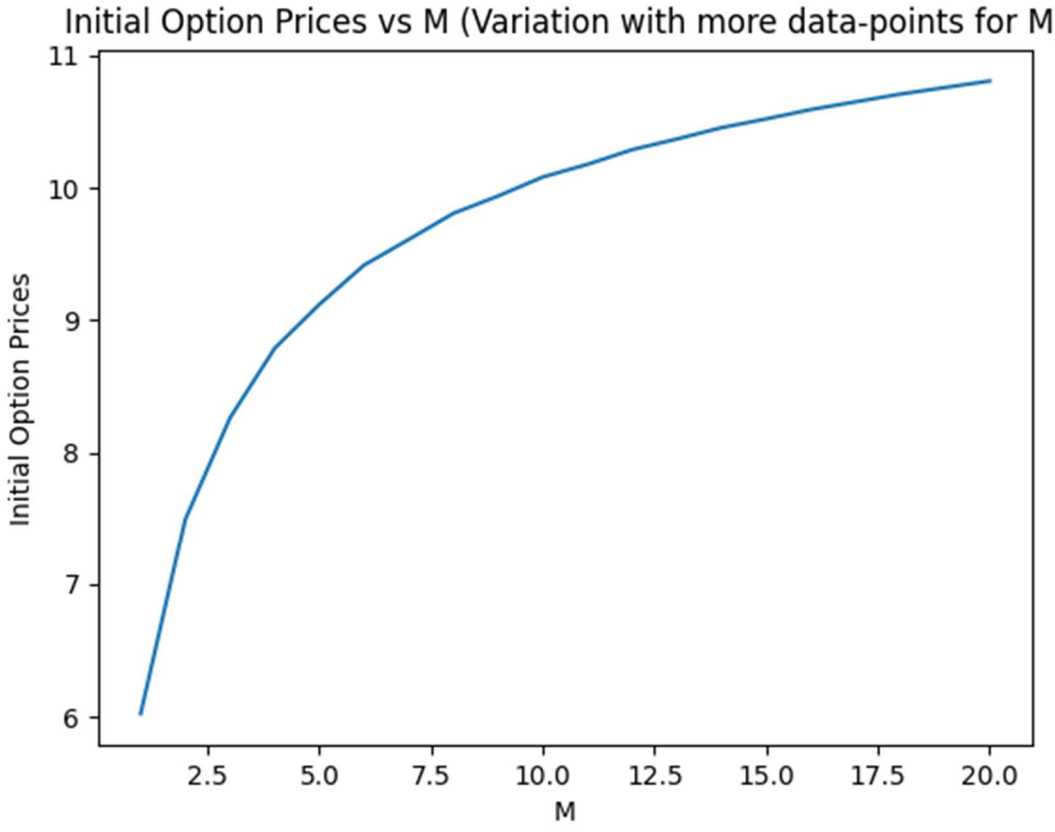
$S(i) = S(i\Delta t)$ and $u = e^{\sigma\sqrt{\Delta t} + \left(r - \frac{1}{2}\sigma^2\right)\Delta t}$ and $d = e^{-\sigma\sqrt{\Delta t} + \left(r - \frac{1}{2}\sigma^2\right)\Delta t}$, $where\ \Delta t = {}^{T}/_{M}$, $M\ here\ is\ the\ number\ of\ subintervals\ in\ [0, T]$.

a)

```
PS C:\Users\Dev Sandip Shah\IITG\Sem6\FE Lab\Lab-3> python Question_2.py
----------------------  sub-part(a)  ----------------------


*********  Executing for M = 5  *********

No arbitrage exists for M = 5
Initial Price of Loopback Option        = 9.119298985864683
Execution Time                          = 0.0 sec



*********  Executing for M = 10  *********

No arbitrage exists for M = 10
Initial Price of Loopback Option        = 10.080582906831
Execution Time                          = 0.002010345458984375 sec



*********  Executing for M = 25  *********

No arbitrage exists for M = 25
Initial Price of Loopback Option        = 11.00349533564633
Execution Time                          = 128.67568111419678 sec
```

b) Screenshots-


Initial Option Prices vs M
Initial Option Prices
11.00
10.75
10.50
10.25
10.00
9.75
9.50
9.25
5.0
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0
M


Initial Option Prices vs M (Variation with more data-points for M)
Initial Option Prices
11
10
9
8
7
6
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
M

c) Below image shows the table for all intermediate values for M = 5. Here each row denotes the option price at each intermediate level. Last level has 32 values, coming in 2 lines in image.

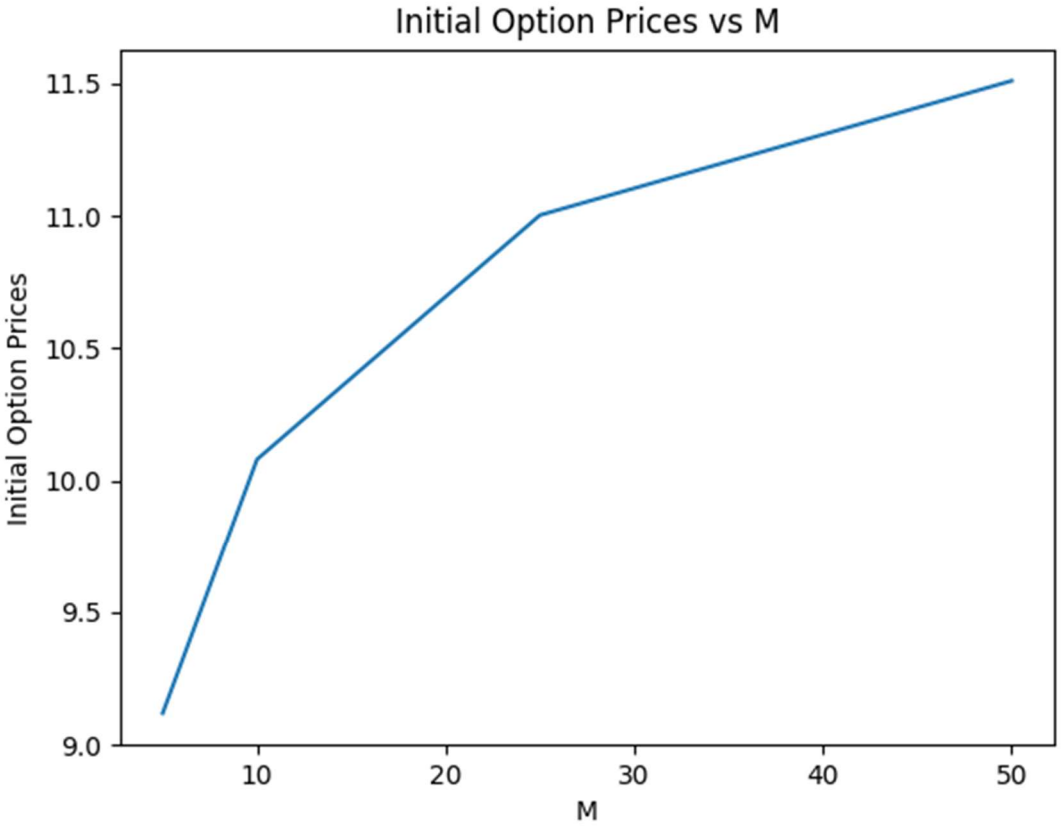
```
----------------------  sub-part(c)  ----------------------
At t = 0
Index no = 0    Price = 9.119298985864683

At t = 1
Index no = 0    Price = 9.027951165547751
Index no = 1    Price = 9.504839866450853

At t = 2
Index no = 0    Price = 8.548076183576441
Index no = 1    Price = 9.799118753547026
Index no = 2    Price = 7.147915756774744
Index no = 3    Price = 12.168664659721792

At t = 3
Index no = 0    Price = 7.416771005131011
Index no = 1    Price = 9.955271272957816
Index no = 2    Price = 6.201916453882752
Index no = 3    Price = 13.712862965988533
Index no = 4    Price = 6.201916453882752
Index no = 5    Price = 8.32461466963314
Index no = 6    Price = 7.14841820819012
Index no = 7    Price = 17.582062714095418
```

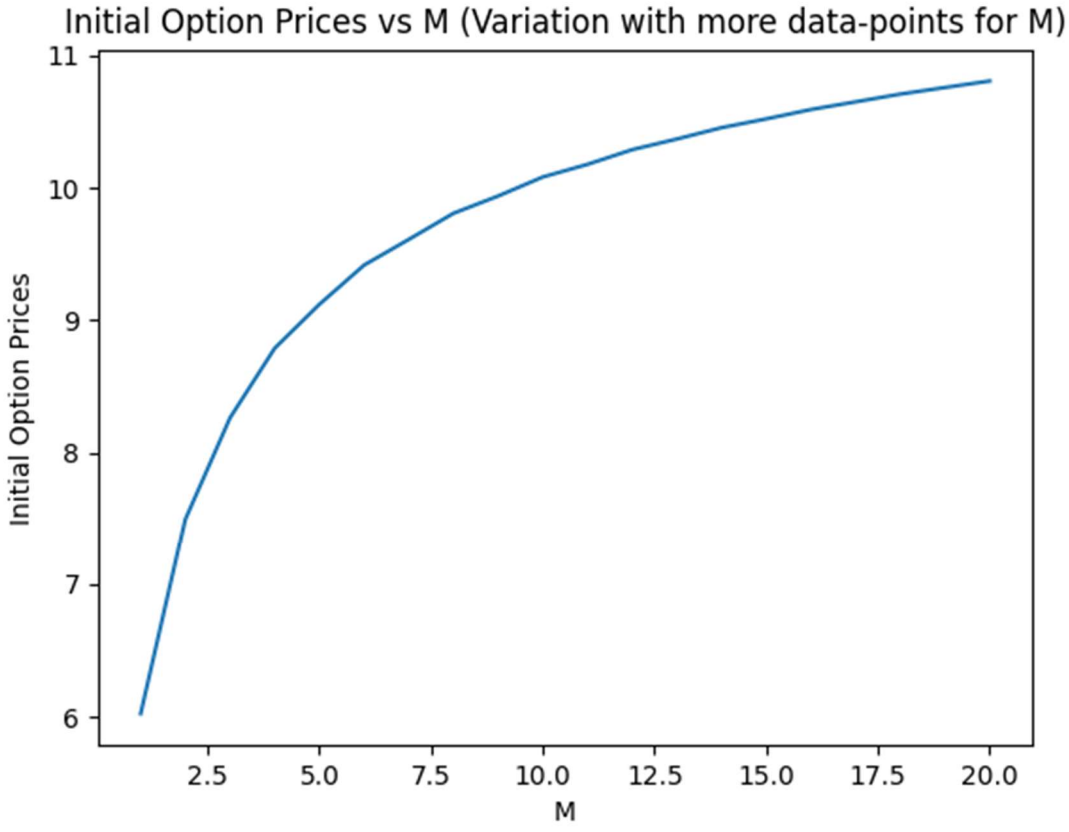
```
At t = 4
Index no = 0    Price = 5.501638813873981
Index no = 1    Price = 9.571391531700229
Index no = 2    Price = 4.600479677676438
Index no = 3    Price = 15.631851880479827
Index no = 4    Price = 4.600479677676438
Index no = 5    Price = 8.003613780975444
Index no = 6    Price = 6.6808429992566465
Index no = 7    Price = 21.18808934534565
Index no = 8    Price = 4.600479677676438
Index no = 9    Price = 8.003613780975444
Index no = 10   Price = 3.8469288844156075
Index no = 11   Price = 13.071380970928788
Index no = 12   Price = 3.8469288844156075
Index no = 13   Price = 10.68090442602997
Index no = 14   Price = 10.68090442602997
Index no = 15   Price = 25.051229457037028
```

```
At t = 5
Index no = 0    Price = 0.0
Index no = 1    Price = 11.181413117784501
Index no = 2    Price = 0.0
Index no = 3    Price = 19.452691543130413
Index no = 5    Price = 9.349916553291678
Index no = 6    Price = 6.374517470614265
Index no = 7    Price = 25.39456347506497
Index no = 8    Price = 0.0
Index no = 9    Price = 9.349916553291678
Index no = 10   Price = 0.0
Index no = 11   Price = 16.266373556657385
Index no = 12   Price = 0.0
Index no = 13   Price = 13.578002496522686
Index no = 14   Price = 13.578002496522686
Index no = 15   Price = 29.48259712227059
Index no = 16   Price = 0.0
Index no = 17   Price = 9.349916553291678
Index no = 18   Price = 0.0
Index no = 19   Price = 16.266373556657385
Index no = 20   Price = 0.0
Index no = 21   Price = 7.8184160295867144
Index no = 22   Price = 5.330382286201839
Index no = 23   Price = 21.234976911949744
Index no = 24   Price = 0.0
Index no = 25   Price = 7.8184160295867144
Index no = 26   Price = 2.9013504971397026
Index no = 27   Price = 18.805945122887607
Index no = 28   Price = 2.9013504971397026
Index no = 29   Price = 18.805945122887607
Index no = 30   Price = 18.805945122887607
Index no = 31   Price = 32.10539403853048
```

Question 3

```
PS C:\Users\Dev Sandip Shah\IITG\Sem6\FE Lab\Lab-3> python Question_3.py
----------------------  sub-part(a)  ----------------------


*********  Executing for M = 5  *********

No arbitrage exists for M = 5
Initial Price of Loopback Option        = 9.119298985864683
Execution Time                          = 0.0 sec


*********  Executing for M = 10  *********

No arbitrage exists for M = 10
Initial Price of Loopback Option        = 10.080582906831
Execution Time                          = 0.0010004043579101562 sec



*********  Executing for M = 25  *********

No arbitrage exists for M = 25
Initial Price of Loopback Option        = 11.00349533564633
Execution Time                          = 0.04801034927368164 sec



*********  Executing for M = 50  *********

No arbitrage exists for M = 50
Initial Price of Loopback Option        = 11.510862222177286
Execution Time                          = 2.702096939086914 sec
```


Initial Option Prices vs M
Initial Option Prices
M
11.5
11.0
10.5
10.0
9.5
9.0
10
20
30
40
50


Initial Option Prices vs M (Variation with more data-points for M)
Initial Option Prices
M
11
10
9
8
7
6
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

```
----------------------  sub-part(c)  ----------------------
At t = 0
Intermediate state = (100, 100)         Price = 9.119298985864683

At t = 1
Intermediate state = (110.676651999383, 110.676651999383)              Price = 9.027951165547751
Intermediate state = (92.54800352077254, 100)           Price = 9.504839866450853

At t = 2
Intermediate state = (122.49321297792528, 122.49321297792528)          Price = 8.548076183576441
Intermediate state = (102.42903178906215, 110.676651999383)            Price = 9.799118753547026
Intermediate state = (102.42903178906214, 102.42903178906214)          Price = 7.147915756774744
Intermediate state = (85.65132955680926, 100)           Price = 12.168664659721792

At t = 3
Intermediate state = (135.57138705044142, 135.57138705044142)          Price = 7.416771005131011
Intermediate state = (113.3650230595177, 122.49321297792528)           Price = 9.955271272957816
Intermediate state = (113.3650230595177, 113.3650230595177)            Price = 6.201916453882752
Intermediate state = (94.79602394643446, 110.676651999383)             Price = 13.712862965988533
Intermediate state = (113.36502305951768, 113.36502305951768)          Price = 6.201916453882752
Intermediate state = (94.79602394643445, 102.42903178906214)           Price = 8.32461466963314
Intermediate state = (94.79602394643445, 100)           Price = 7.14841820819012
Intermediate state = (79.26859549382432, 100)           Price = 17.582062714095418

At t = 4
Intermediate state = (150.04587225655362, 150.04587225655362)          Price = 5.501638813873981
Intermediate state = (104.91706553244704, 110.676651999383)            Price = 6.6808429992566465
Intermediate state = (87.73182757949854, 110.676651999383)             Price = 21.18808934534565
Intermediate state = (125.46861206060267, 125.46861206060267)          Price = 4.600479677676438
Intermediate state = (104.91706553244703, 113.36502305951768)          Price = 8.003613780975444
Intermediate state = (104.91706553244701, 104.91706553244701)          Price = 3.8469288844156075
Intermediate state = (87.73182757949853, 102.42903178906214)           Price = 13.071380970928788
Intermediate state = (87.73182757949853, 100)           Price = 10.68090442602997
Intermediate state = (73.36150254849147, 100)           Price = 25.051229457037028
```

```
At t = 5
Intermediate state = (166.06574787682462, 166.06574787682462)          Price = 0.0
Intermediate state = (138.86445913876912, 150.04587225655362)          Price = 11.181413117784501
Intermediate state = (138.8644591387691, 138.8644591387691)            Price = 0.0
Intermediate state = (116.118695507311, 135.57138705044142)            Price = 19.452691543130413
Intermediate state = (116.118695507311, 125.46861206060268)            Price = 9.349916553291678
Intermediate state = (116.11869550731102, 122.49321297792528)          Price = 6.374517470614265
Intermediate state = (97.09864950286031, 122.49321297792528)           Price = 25.39456347506497
Intermediate state = (116.11869550731102, 116.11869550731102)          Price = 0.0
Intermediate state = (97.09864950286031, 113.3650230595177)            Price = 16.266373556657385
Intermediate state = (97.09864950286031, 110.676651999383)             Price = 13.578002496522686
Intermediate state = (81.1940548771124, 110.676651999383)              Price = 29.48259712227059
Intermediate state = (116.11869550731099, 125.46861206060267)          Price = 9.349916553291678
Intermediate state = (116.11869550731099, 116.11869550731099)          Price = 0.0
Intermediate state = (97.0986495028603, 113.36502305951768)            Price = 16.266373556657385
Intermediate state = (116.11869550731097, 116.11869550731097)          Price = 0.0
Intermediate state = (97.0986495028603, 104.91706553244701)            Price = 7.8184160295867144
Intermediate state = (97.0986495028603, 102.42903178906214)            Price = 5.330382286201839
Intermediate state = (81.19405487711239, 102.42903178906214)           Price = 21.234976911949744
Intermediate state = (97.0986495028603, 100)            Price = 2.9013504971397026
Intermediate state = (81.19405487711239, 100)           Price = 18.805945122887607
Intermediate state = (67.89460596146952, 100)           Price = 32.10539403853048
```

Maximum value of M for the algorithm to run in reasonable in time:

- For binomial: 15
- For Markov: 50

Time Complexity

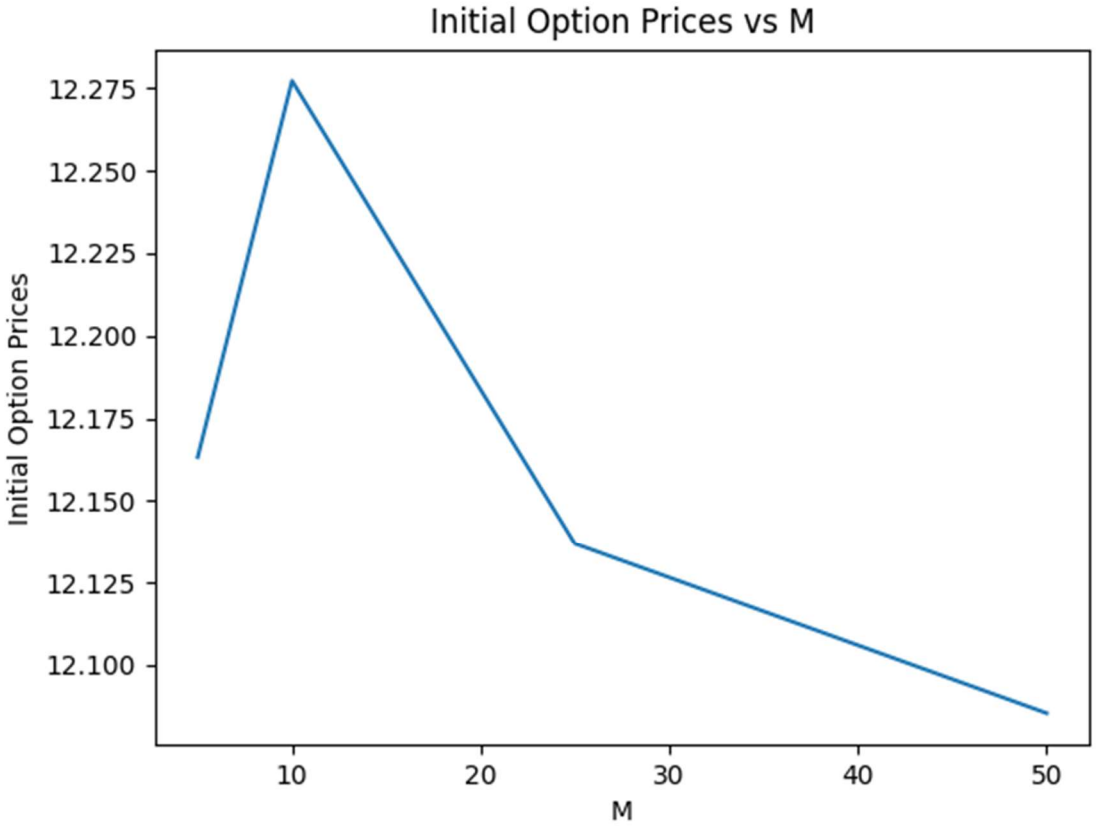
- Time complexity for binomial algorithm is O(2^M) because we are exploring every path of the binomial tree.
- Markov algorithm depends on 2 states, the current stock price and maximum stock price encountered along the path till now. Time complexity of this algorithm is O(M^4), because number of unique paths is bounded by O(M^2) and hence, maximum stock prices is also bounded by O(M^2).

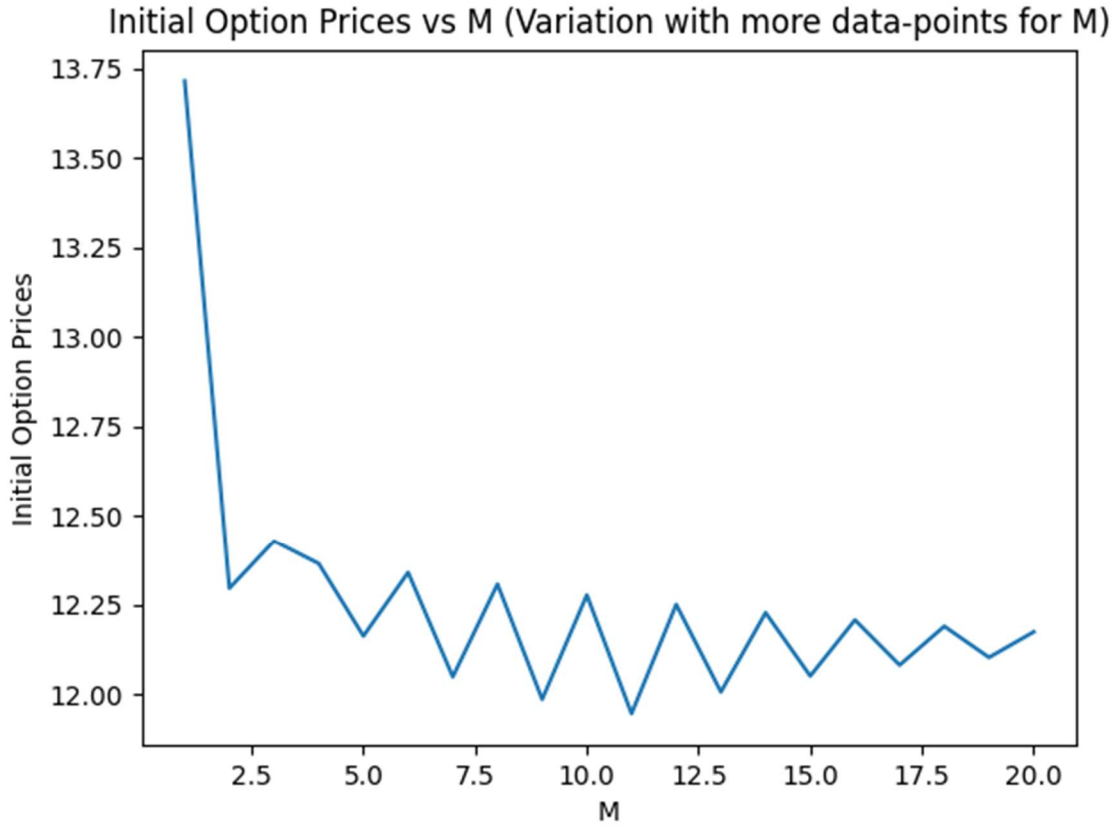
Question 4

Output on running the code, following is the output:

```
PS C:\Users\Dev Sandip Shah\IITG\Sem6\FE Lab\Lab-3> python Question_4.py
----------------------  sub-part(a)  ----------------------

######################  Unoptimised Binomial Algorithm executing  ######################
No arbitrage exists for M = 5
European Call Option            = 12.16318594676458
Execution Time                  = 0.0 sec

No arbitrage exists for M = 10
European Call Option            = 12.27732781922299
Execution Time                  = 0.001832723617553711 sec

No arbitrage exists for M = 25
European Call Option            = 12.136745963232949
Execution Time                  = 65.72803044319153 sec



######################  Efficient Binomial Algorithm executing (Markov Based)  ######################
No arbitrage exists for M = 5
European Call Option            = 12.163185946764584
Execution Time                  = 0.0 sec

No arbitrage exists for M = 10
European Call Option            = 12.277327819222982
Execution Time                  = 0.0 sec

No arbitrage exists for M = 25
European Call Option            = 12.136745963232947
Execution Time                  = 0.0009098052978515625 sec

No arbitrage exists for M = 50
European Call Option            = 12.0853615100722
Execution Time                  = 0.0 sec
```

```
######################  Most Efficient Binomial Algorithm executing (Markov Based) ######################
No arbitrage exists for M = 5
European Call Option            = 12.163185946764584
Execution Time                  = 0.0 sec

No arbitrage exists for M = 10
European Call Option            = 12.277327819222986
Execution Time                  = 0.0 sec

No arbitrage exists for M = 25
European Call Option            = 12.136745963232956
Execution Time                  = 0.0 sec

No arbitrage exists for M = 50
European Call Option            = 12.085361510072197
Execution Time                  = 0.0010001659393310547 sec
```


Initial Option Prices vs M
Initial Option Prices
12.275
12.250
12.225
12.200
12.175
12.150
12.125
12.100
10
20
30
40
50
M


Initial Option Prices vs M (Variation with more data-points for M)
Initial Option Prices
13.75
13.50
13.25
13.00
12.75
12.50
12.25
12.00
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
M

```
----------------------  sub-part(c)  ----------------------
At t = 0
Index no = 0    Price = 12.163185946764584

At t = 1
Index no = 0    Price = 18.65868251160212
Index no = 1    Price = 6.0592900974208455

At t = 2
Index no = 0    Price = 27.525444303544514
Index no = 1    Price = 10.392778619897372
Index no = 2    Price = 1.9207528986659217

At t = 3
Index no = 0    Price = 38.72072884252166
Index no = 1    Price = 17.21677529537563
Index no = 2    Price = 3.9032313677700126
Index no = 3    Price = 0.0

At t = 4
Index no = 0    Price = 51.633140251025104
Index no = 1    Price = 27.055880055074176
Index no = 2    Price = 7.9318974975518906
Index no = 3    Price = 0.0
Index no = 4    Price = 0.0

At t = 5
Index no = 0    Price = 66.06574787682459
Index no = 1    Price = 38.86445913876909
Index no = 2    Price = 16.118695507311017
Index no = 3    Price = 0
Index no = 4    Price = 0
Index no = 5    Price = 0
```

In a reasonable amount of time, we found that

- $M_{max} = 20$ for Binomial
- $M_{max} = 1000\ approx.$ for Markov

Below is the table showing the comparison time and initial Option Value.

Time Complexity

- Time complexity for binomial algorithm is O(2^M) because we are exploring every path of the binomial tree.
- Markov algorithm depends on 2 states, the step number and count of up steps encountered along the path till now. Time complexity of this algorithm is O(M^3), because number of unique states is bounded by O(M^2) and hence, number of up states is also bounded by O(M).