C++ Programming for Financial Engineering Level 9 Group C Writeup

QuantNet

Minghan Li

10/28/2019

a)

The Program runs and compiles correctly with no error.

b)

Batch 1 Output

N	NSim	Call Price	Call Error	Call SD	Call SE	Put Price	Put Error	Put SD	Put SE
100	10000	2.1378	0.00443	4.54234	0.045423	5.90649	0.06021	6.05727	0.060573
100	100000	2.13628	0.002911	4.5113	0.014266	5.81293	0.033352	6.02739	0.01906
100	1000000	2.12751	0.00586	4.50672	0.004507	5.85292	0.006638	6.05189	0.006052
100	3000000	2.1341	0.00073	4.51319	0.002606	5.8388	0.007483	6.0481	0.003492
100	5000000	2.13097	0.0024	4.50742	0.002016	5.84311	0.00317	6.04795	0.002705
300	10000	2.09975	0.033616	4.4639	0.044639	5.85913	0.012848	6.08277	0.060828
300	100000	2.13566	0.002293	4.51716	0.014285	5.83552	0.010758	6.04398	0.019113
300	1000000	2.13082	0.002552	4.51	0.00451	5.84304	0.003245	6.05552	0.006056
300	3000000	2.13857	0.005197	4.52396	0.002612	5.85667	0.010393	6.05223	0.003494
300	5000000	2.1354	0.002031	4.51644	0.00202	5.84766	0.00138	6.04882	0.002705
500	10000	2.10715	0.026217	4.48322	0.044832	5.85358	0.007301	6.05935	0.060594
500	100000	2.10944	0.023925	4.49796	0.014224	5.84509	0.001187	6.04499	0.019116
500	1000000	2.13675	0.003382	4.52079	0.004521	5.83382	0.012457	6.04626	0.006046
500	3000000	2.1344	0.001031	4.51821	0.002609	5.84921	0.002931	6.04762	0.003492
500	5000000	2.13301	0.000358	4.51733	0.00202	5.84339	0.002892	6.04524	0.002704

Batch 2 Output

N	NSim	Call Price	Call Error	Call SD	Call SE	Put Price	Put Error	Put SD	Put SE
100	10000	7.94097	0.024605	13.212	0.13212	8.01361	0.048045	10.4813	0.104813
100	100000	7.96498	0.000585	13.1449	0.041568	7.90847	0.0571	10.366	0.03278
100	1000000	7.94918	0.016392	13.1285	0.013129	7.98184	0.016269	10.4166	0.010417
100	3000000	7.97322	0.007648	13.144	0.007589	7.95591	0.009659	10.4103	0.00601
100	5000000	7.96664	0.001069	13.1302	0.005872	7.9636	0.001974	10.4095	0.004655
300	10000	7.94383	0.021739	13.0063	0.130063	7.97574	0.010166	10.4996	0.104996
300	100000	7.97248	0.006908	13.153	0.041593	7.94682	0.018748	10.3993	0.032886
300	1000000	7.96503	0.000542	13.1342	0.013134	7.96506	0.00051	10.421	0.010421
300	3000000	7.98299	0.017423	13.1686	0.007603	7.98431	0.018737	10.4157	0.006013
300	5000000	7.97186	0.006295	13.1526	0.005882	7.97018	0.004611	10.4071	0.004654
500	10000	7.8416	0.123967	13.0796	0.130796	7.98273	0.017157	10.4258	0.104258
500	100000	7.88892	0.07665	13.1016	0.041431	7.96749	0.001924	10.396	0.032875
500	1000000	7.97471	0.009142	13.1616	0.013162	7.9458	0.019768	10.4014	0.010401
500	3000000	7.96751	0.001943	13.1554	0.007595	7.97122	0.005648	10.4044	0.006007
500	5000000	7.96467	0.000899	13.1525	0.005882	7.96003	0.005539	10.4004	0.004651

Batch 1 Call Error

Sum of Call Error	Column Labels		
Row Labels	100	300	500
10000	0.00442991	0.0336158	0.0262173
100000	0.00291068	0.00229279	0.0239251
1000000	0.00586036	0.00255153	0.00338211
3000000	0.000729957	0.00519677	0.00103106
5000000	0.00239991	0.00203111	0.000357667

Batch 1 Put Error

Sum of Put Error	Column Labels		
Row Labels	100	300	500
10000	0.0602095	0.0128484	0.00730088
100000	0.0333517	0.0107578	0.00118686
1000000	0.00663845	0.00324454	0.0124567
3000000	0.00748342	0.0103929	0.00293082
5000000	0.00316995	0.00137984	0.00289199

Batch 2 Call Error

Sum of Call Error	Column Labels		
Row Labels	100	300	500
10000	0.0246047	0.0217385	0.123967
100000	0.000585104	0.00690842	0.0766502
1000000	0.0163919	0.000541869	0.00914151
3000000	0.00764802	0.0174228	0.00194264
5000000	0.00106943	0.00629452	0.000899264

Batch 2 Put Error

Sum of Put Error	Column Labels		
Row Labels	100	300	500
10000	0.0480449	0.0101663	0.0171568
100000	0.0570998	0.0187482	0.00192368
1000000	0.0162688	0.000509549	0.0197676
3000000	0.00965922	0.018737	0.00564824
5000000	0.00197431	0.00461137	0.00553912

From the outputs above, we can that most of the smallest errors occur in the bottom right of the error tables, indicating that the increase of both number of simulations and number of steps should increase the simulation accuracy. However, a close examination reveals that for a given small N (N=100), the increase of number of steps towards infinity don't necessarily improve the accuracy. For example, for N = 100, errors decrease significantly for increasing small NSIM, but as NSIM grows to 5000000, the error does not show sign of decreasing. On the other hand, for a large N, increasing NSIM indeed deceases error significantly.

For a given NSIM small, N has almost no relationship with the magnitude of error. It is only when NSIM reaches the millions that an increase in N start to reduce the error for call and put options for Batch 2, but the effect soon diminishes. For Batch 1, increase in N shows no sign of decreasing the error.

In general, large N and NSIM tend to produce estimations closer to the exact solution, but their effect on the accuracy of the estimations is only evident when both are large. For a given NSIM, the effect of N is not observed. Increase in N does not decrease error. This is perhaps due to the short expiry time of Batch 1 and Batch 2. Based on the result, I would speculate that there is optimal N relative to each expiry time. N larger than this optimal value will no longer improve the accuracy of the simulation. For further research, I would try larger N on options with larger expiry time to confirm my speculation.

c)

Batch 4 Output

N	NSim	Call Price	Call Error	Put Price	Put Error
100	10000	88.0412	4.13449	1.30114	0.053644
100	100000	88.9885	3.18721	1.279	0.031501
100	1000000	89.1499	3.02582	1.29374	0.046237
100	3000000	89.4347	2.74101	1.28919	0.041691
100	5000000	89.3608	2.81494	1.29059	0.043088
300	10000	91.3979	0.7778	1.25873	0.011229
300	100000	91.3263	0.849405	1.25881	0.011314
300	1000000	90.8128	1.36291	1.26351	0.016011
300	3000000	91.5105	0.665164	1.26448	0.016977
300	5000000	91.3423	0.833375	1.26235	0.014851
500	10000	88.6215	3.55423	1.25855	0.011047
500	100000	90.4535	1.72218	1.25788	0.010381
500	1000000	91.6388	0.536944	1.25267	0.005169
500	3000000	91.6354	0.540345	1.25661	0.009114
500	5000000	91.5506	0.625083	1.25413	0.006632
500	10000000	91.6058	0.569876	1.25662	0.009122
800	1000000	92.8896	0.713892	1.25152	0.004019

Batch 4 Call Error

Sum of Call Error	**Column Labels**			
Row Labels	**100**	**300**	**500**	**800**
10000	4.13449	0.7778	3.55423	
100000	3.18721	0.849405	1.72218	
1000000	3.02582	1.36291	0.536944	0.713892
3000000	2.74101	0.665164	0.540345	
5000000	2.81494	0.833375	0.625083	
10000000			0.569876	

Batch 4 Put Error

Sum of Put Error	Column Labels			
Row Labels	100	300	500	800
10000	0.0536439	0.0112293	0.0110474	
100000	0.0315008	0.0113135	0.0103809	
1000000	0.0462369	0.0160114	0.00516939	0.00401921
3000000	0.0416909	0.016977	0.00911405	
5000000	0.0430879	0.014851	0.00663189	
10000000			0.00912245	

For put option, the accuracy is already two places behind the decimal point when N reaches 500, and NSIM reaches 100000. Further increase of N to 800 improve the accuracy even more. On the other hand, the accuracy for call option is much harder to improve. Although increase in NSIM and N do seem to improve accuracy slightly, the accuracy is nowhere approaching two places behind the decimal point. Even when NSIM is increased to 10 million, the accuracy still does not show significant improvement. In the end, the cost of computing such a large number of simulations is simply too great to achieve the desired accuracy. This phenomenon is worth an investigation. Perhaps N is supposed to be set much larger in order to achieve the accuracy.

The error output for Batch 4 confirms my speculation in part b. Since Batch 4 has a much longer expiry time than Batch 1 and Batch 2, we expect higher number of steps to obtain an optimal simulation. As the output shows, the accuracy only increases as N increases, indicating that the optimal n has not yet been achieved.