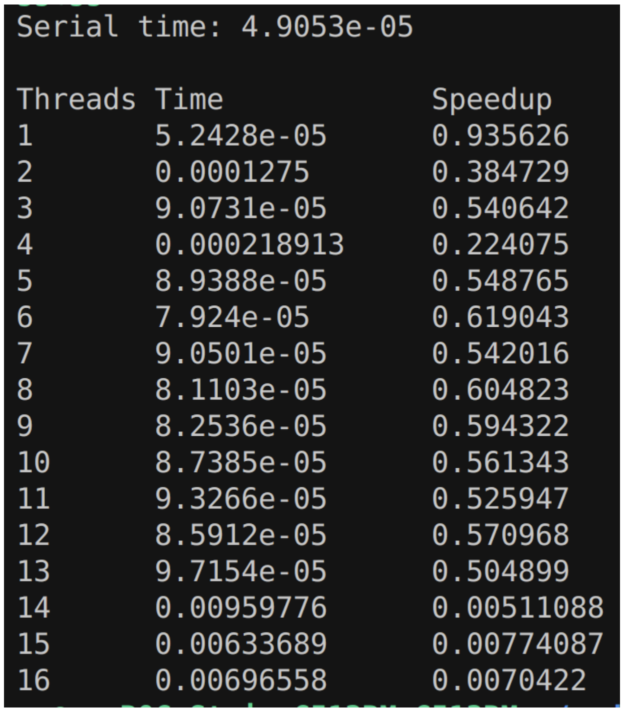
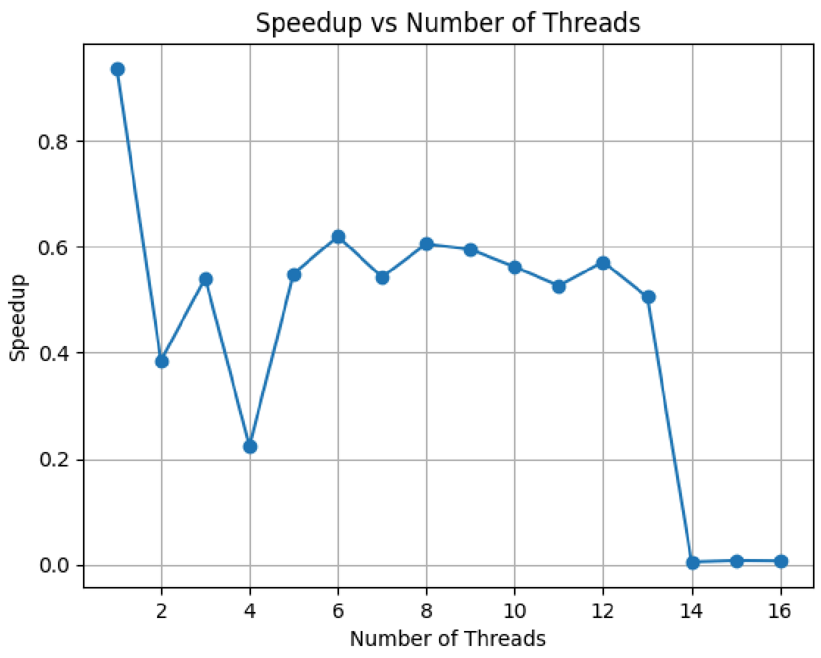
ASSIGNMENT-1
GAURANG GARG [102303134]

Question 1 -



```
Serial time: 4.9053e-05

Threads Time            Speedup
1       5.2428e-05      0.935626
2       0.0001275       0.384729
3       9.0731e-05      0.540642
4       0.000218913     0.224075
5       8.9388e-05      0.548765
6       7.924e-05       0.619043
7       9.0501e-05      0.542016
8       8.1103e-05      0.604823
9       8.2536e-05      0.594322
10      8.7385e-05      0.561343
11      9.3266e-05      0.525947
12      8.5912e-05      0.570968
13      9.7154e-05      0.504899
14      0.00959776      0.00511088
15      0.00633689      0.00774087
16      0.00696558      0.0070422
```



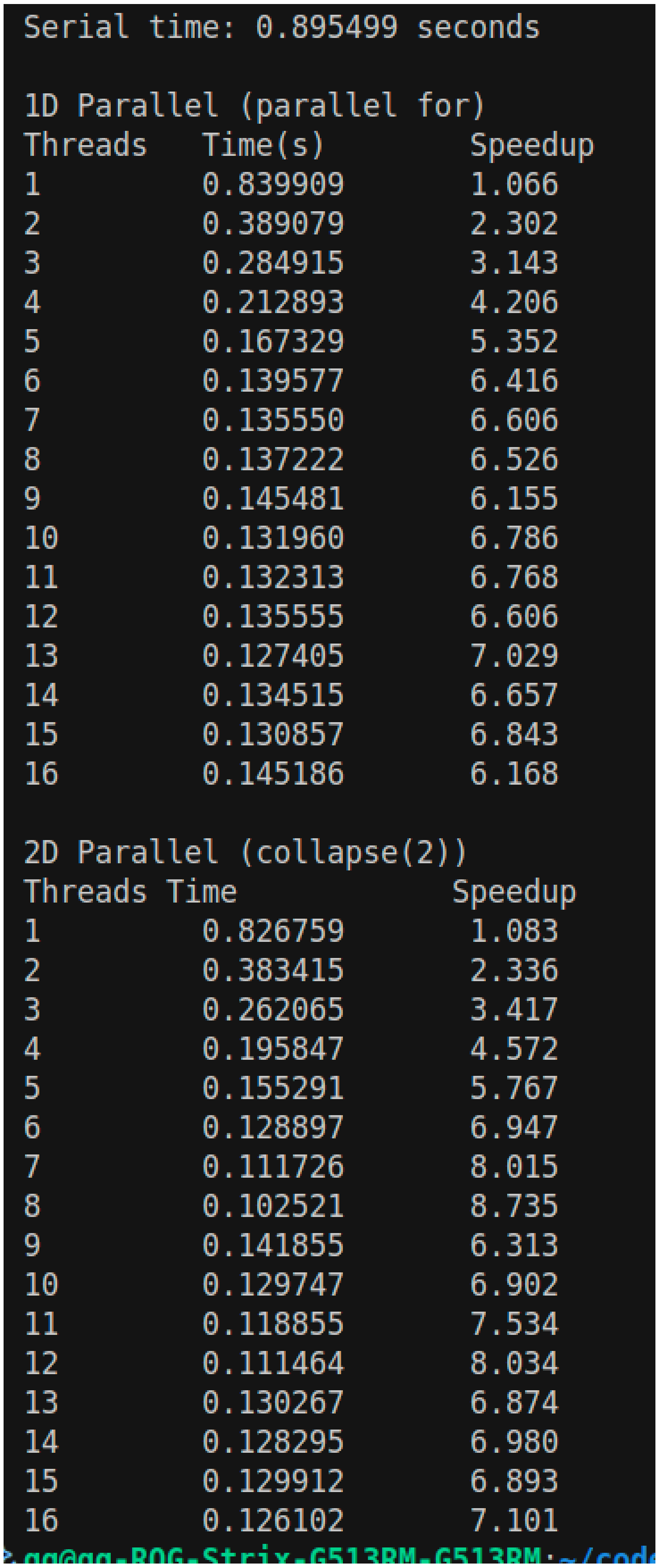
Observations :

1.The problem is extremely small [only 65,536 arithmetic operations]. This is the reason serial time is so small.

2.Parallel Overhead Exceeds Benefits. For all the parallel executions the timing is worse than serial due to OpenMP overhead.Parallel overhead includes - thread creation/destruction, work distribution, thread synchronization etc. This leads to speedup less than 1.

3. My machine has 8 cores and 16 threads. Till 8 threads, the performance is dominated by the overhead. From 9 to 13 threads there is similar poor performance. After that till 16 threads performance drops drastically as all 8 cores are subscribed with more than one thread each.

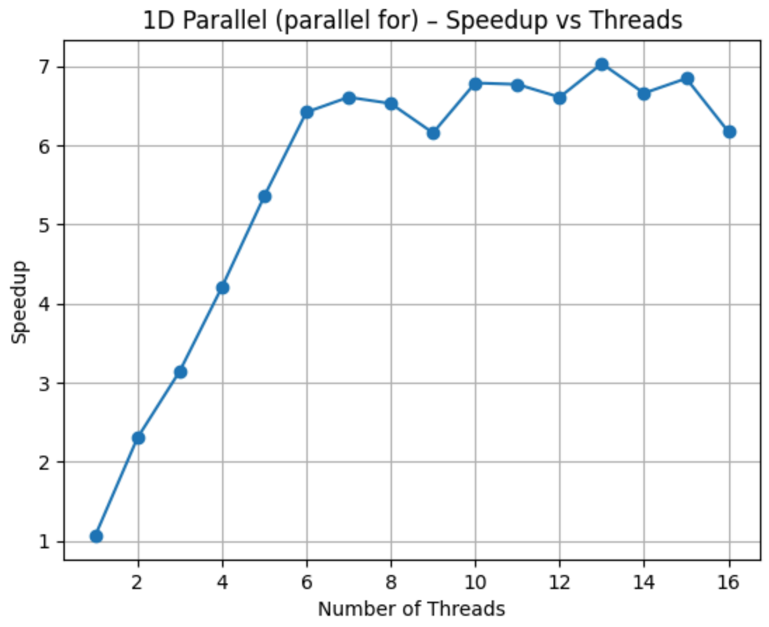
Question 2 -

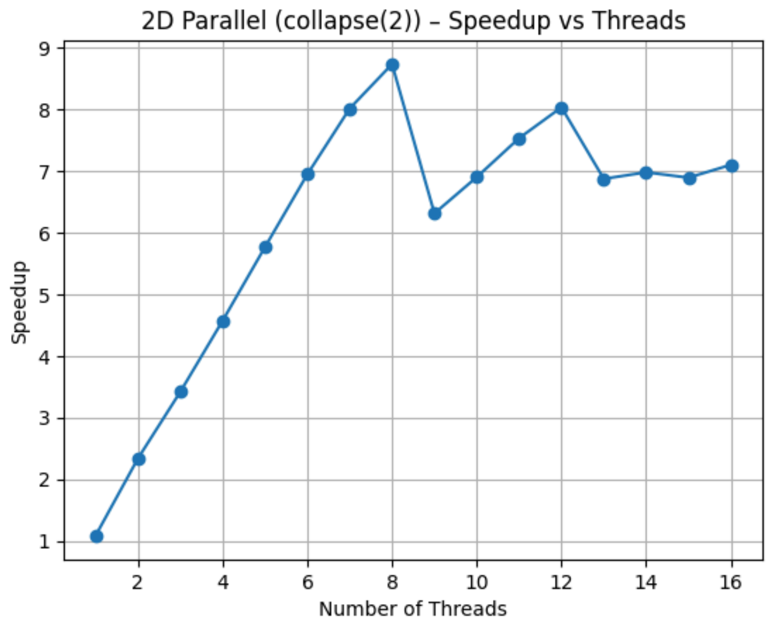


```
Serial time: 0.895499 seconds

1D Parallel (parallel for)
Threads    Time(s)        Speedup
1          0.839909       1.066
2          0.389079       2.302
3          0.284915       3.143
4          0.212893       4.206
5          0.167329       5.352
6          0.139577       6.416
7          0.135550       6.606
8          0.137222       6.526
9          0.145481       6.155
10         0.131960       6.786
11         0.132313       6.768
12         0.135555       6.606
13         0.127405       7.029
14         0.134515       6.657
15         0.130857       6.843
16         0.145186       6.168

2D Parallel (collapse(2))
Threads Time             Speedup
1          0.826759       1.083
2          0.383415       2.336
3          0.262065       3.417
4          0.195847       4.572
5          0.155291       5.767
6          0.128897       6.947
7          0.111726       8.015
8          0.102521       8.735
9          0.141855       6.313
10         0.129747       6.902
11         0.118855       7.534
12         0.111464       8.034
13         0.130267       6.874
14         0.128295       6.980
15         0.129912       6.893
16         0.126102       7.101
gg@gg-ROG-Strix-G513RM-G513RM:~/code
```





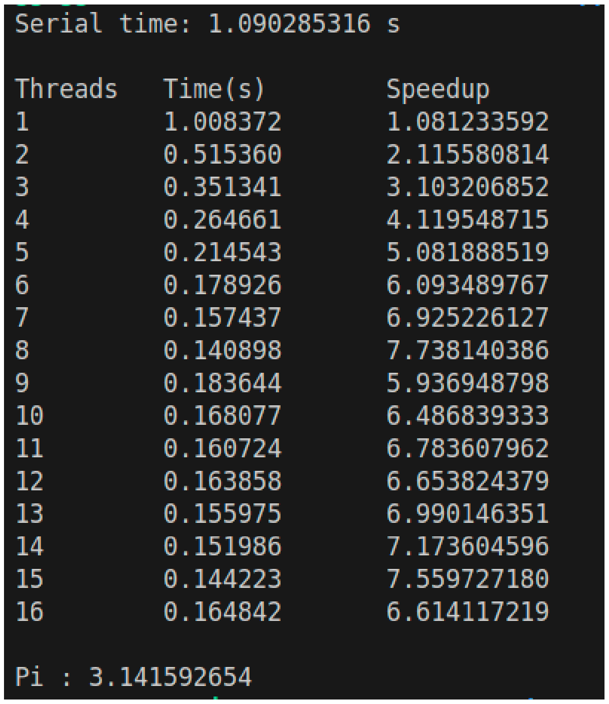
Observations :

1. For both 1D and 2D parallelization speedup increases almost linearly till 8 threads.

2.After 9 threads speed up drops because now 2 threads are competing for one cpu core. As more and more threads increase performance dips more and more. Matrix multiplication is compute and memory extensive task. In 1D there is not a dip in performance after 8 threads but a spiky fashion rather.

3.1D and 2D perform almost similarly . In 1D parallelization, only the outer loop is parallelized, so iterations are evenly distributed among threads when the outer loop is large. Using `collapse(2)` creates more parallel work but provides similar performance in this case since the workload per thread remains the same. Two-dimensional parallelization is beneficial only when the outer loop has few iterations. Performance variations at higher thread counts occur due to scheduling overhead, cache contention, and memory bandwidth limits.

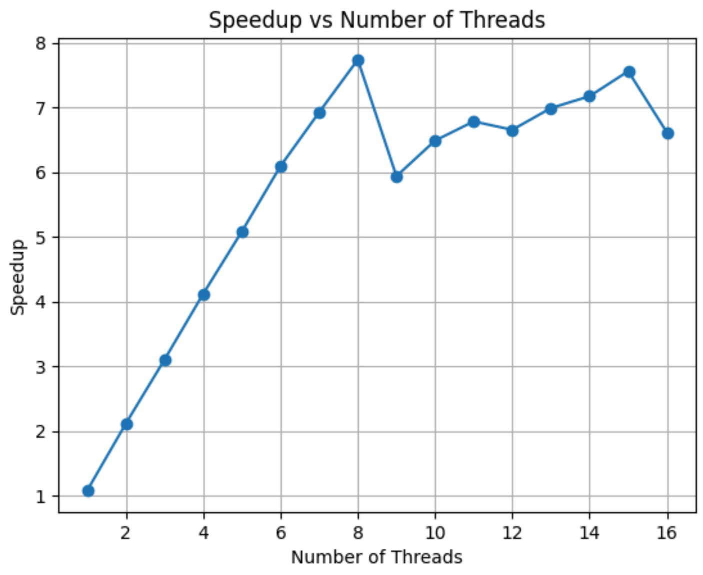
Question 3 -



```
Serial time: 1.090285316 s

Threads     Time(s)         Speedup
1           1.008372        1.081233592
2           0.515360        2.115580814
3           0.351341        3.103206852
4           0.264661        4.119548715
5           0.214543        5.081888519
6           0.178926        6.093489767
7           0.157437        6.925226127
8           0.140898        7.738140386
9           0.183644        5.936948798
10          0.168077        6.486839333
11          0.160724        6.783607962
12          0.163858        6.653824379
13          0.155975        6.990146351
14          0.151986        7.173604596
15          0.144223        7.559727180
16          0.164842        6.614117219

Pi : 3.141592654
```



Observations :

1. The results show excellent scaling up to 8 threads (your physical cores) with 7.74× speedup and 97% efficiency, reducing execution time from 1.09s to 0.141s. This near-linear performance indicates well-optimized parallel code with minimal overhead.

2. However, performance drops sharply at thread 9 (5.94× speedup) and fluctuates between 6.6-7.6× for threads 9-16, never exceeding the 8-thread peak. This occurs because hyperthreading provides minimal benefit for compute-intensive workloads.

3. **The optimal configuration is 8 threads**, matching my physical core count, as adding more threads introduces overhead without performance gains.