# Financial Computing

1807730

March 2023

## 1 Task 1: OpenMP

1. Code to produce the power iteration times can be found in `1_matrix_1.c` and executed using `gcc -fopenmp -O9 1_matrix_1.c -lm; ./1_matrix_1`. It will produce the following table in the command line.

2. Below is the table showing the power iteration times:

| Type | N | t1 | t2 | t4 | t8 | t16 |
|---|---|---|---|---|---|---|
| Time | 256 | 0.003s | 0.003s | 0.002s | 0.001s | 0.002s |
| Speed-Up | 256 | 0.000s | -0.001s | -0.001s | -0.002s | -0.001s |
| Time | 512 | 0.004s | 0.003s | 0.002s | 0.001s | 0.002s |
| Speed-Up | 512 | 0.000s | -0.002s | -0.002s | -0.003s | -0.002s |
| Time | 1024 | 0.012s | 0.011s | 0.004s | 0.002s | 0.003s |
| Speed-Up | 1024 | 0.000s | -0.005s | -0.008s | -0.009s | -0.009s |
| Time | 2048 | 0.042s | 0.021s | 0.011s | 0.006s | 0.004s |
| Speed-Up | 2048 | 0.000s | -0.018s | -0.028s | -0.032s | -0.034s |
| Time | 4096 | 0.151s | 0.091s | 0.060s | 0.024s | 0.015s |
| Speed-Up | 4096 | 0.000s | -0.050s | -0.085s | -0.110s | -0.119s |

3. Code to produce the matrix setup times can be found in `1_matrix_2.c` and executed using `gcc -fopenmp -O9 1_matrix_2.c -lm; ./1_matrix_2`. It will produce the following table in the command line.

4. Below is the table showing the matrix setup times:

| Type | N | t1 | t2 | t4 | t8 | t16 |
|---|---|---|---|---|---|---|
| Time | 256 | 0.005s | 0.006s | 0.003s | 0.002s | 0.001s |
| Speed-Up | 256 | 0.000s | -0.001s | -0.003s | -0.004s | -0.004s |
| Time | 512 | 0.019s | 0.014s | 0.005s | 0.004s | 0.002s |
| Speed-Up | 512 | 0.000s | -0.009s | -0.014s | -0.012s | -0.015s |
| Time | 1024 | 0.073s | 0.037s | 0.018s | 0.015s | 0.012s |
| Speed-Up | 1024 | 0.000s | -0.032s | -0.049s | -0.055s | -0.064s |
| Time | 2048 | 0.288s | 0.145s | 0.101s | 0.040s | 0.026s |
| Speed-Up | 2048 | 0.000s | -0.136s | -0.214s | -0.241s | -0.259s |
| Time | 4096 | 1.159s | 0.614s | 0.305s | 0.164s | 0.111s |
| Speed-Up | 4096 | 0.000s | -0.618s | -0.862s | -1.010s | -1.091s |

5. Given that the code performs the power iteration algorithm in approximately a tenth of a second with 8 threads and $N = 4096$, you can most likely increase $N$ to $2^{15}$ before you begin to see a significant reduction in speed.

# 2 Task 2: MPI

1. Code to produce the power iteration times on the same computer can be found in `2_matrix_1.c` and executed using `mpicc -lm -o 2_matrix_1 2_matrix_1.c; mpirun -np [Threads] ./2_matrix_1 [N]`.

2. You can also produce the table for all pairs of values specified by running the shell script `computer_power_iteration_time.sh`. This can be done by running `chmod +x computer_power_iteration_time.sh; ./computer_power_iteration_time.sh`. Note that you will need to compile the C code first in order to run the shell executable.

3. Below is the power iteration times for the same computer:

| Type | N | 1 | 2 | 4 | 8 |
|---|---|---|---|---|---|
| Time | 256 | 0.007297 | 0.004496 | 0.003713 | 0.004512 |
| Speed-Up | 256 | 0.0 | 0.002801 | 0.003584 | 0.002785 |
| Time | 512 | 0.023250 | 0.012960 | 0.008197 | 0.006651 |
| Speed-Up | 512 | 0.0 | 0.010290 | 0.015053 | 0.016599 |
| Time | 1024 | 0.087499 | 0.045088 | 0.024844 | 0.015395 |
| Speed-Up | 1024 | 0.0 | 0.042411 | 0.062655 | 0.072104 |
| Time | 2048 | 0.341891 | 0.205784 | 0.089401 | 0.051577 |
| Speed-Up | 2048 | 0.0 | 0.136107 | 0.252490 | 0.290314 |
| Time | 4096 | 1.405218 | 0.784595 | 0.359617 | 0.217117 |
| Speed-Up | 4096 | 0.0 | 0.620623 | 1.045601 | 1.188101 |

4. Code to produce the matrix setup times on the same computer can be found in `2_matrix_1.c` and executed using `mpicc -lm -o 2_matrix_1 2_matrix_1.c; mpirun -np [Threads] ./2_matrix_1 [N]`.

5. You can also produce the table for all pairs of values specified by running the shell script `computer_matrix_setup_time.sh`. This can be done by running `chmod +x computer_matrix_setup_time.sh; ./computer_matrix_setup_time.sh`. Note that you will need to compile the C code first in order to run the shell executable.

6. The table of matrix setup times on the same computer can be found below.

| Type | N | 1 | 2 | 4 | 8 |
|---|---|---|---|---|---|
| Time | 256 | 0.008043 | 0.006836 | 0.006582 | 0.009191 |
| Speed-Up | 256 | 0.0 | 0.001207 | 0.001461 | -0.001148 |
| Time | 512 | 0.025244 | 0.025736 | 0.025522 | 0.028384 |
| Speed-Up | 512 | 0.0 | -0.000492 | -0.000278 | -0.003140 |
| Time | 1024 | 0.097401 | 0.099294 | 0.099154 | 0.109109 |
| Speed-Up | 1024 | 0.0 | -0.001893 | -0.001753 | -0.011708 |
| Time | 2048 | 0.394899 | 0.397619 | 0.395751 | 0.427632 |
| Speed-Up | 2048 | 0.0 | -0.002720 | -0.000852 | -0.032733 |
| Time | 4096 | 1.600617 | 1.650263 | 1.624059 | 1.783719 |
| Speed-Up | 4096 | 0.0 | -0.049646 | -0.023442 | -0.183102 |

7. We decided to use MPI on the matrix multiplication since this dominates the execution time over anything else. This involved allocating to each process the indices of the rows they would be focusing on and gathering all the data back together after the execution of each block.

8. Code to produce the power iteration times on different computers can be found in `2_matrix_1.c` and executed using `mpicc -lm -o 2_matrix_1 2_matrix_1.c; mpirun -H [Computers] ./2_matrix_1 [N]`.

9. You can also produce the table for all pairs of values specified by running the shell script `dist_power_iteration_time.sh`. This can be done by running `chmod +x dist_power_iteration_time.sh; ./dist_power_iteration_time.sh`. Note that you will need to compile the C code first in order to run the shell executable.

10. The table of power iteration times on different computers can be found below.

| Type | N | 1 | 2 | 4 | 8 |
|---|---|---|---|---|---|
| Time | 256 | 0.009405 | 0.008852 | 0.010641 | 0.026535 |
| Speed-Up | 256 | 0.0 | 0.000553 | -0.001236 | -0.017130 |
| Time | 512 | 0.023115 | 0.017017 | 0.019677 | 0.021045 |
| Speed-Up | 512 | 0.0 | 0.006098 | 0.003438 | 0.002070 |
| Time | 1024 | 0.087276 | 0.050421 | 0.051443 | 0.046599 |
| Speed-Up | 1024 | 0.0 | 0.036855 | 0.035833 | 0.040677 |
| Time | 2048 | 0.358191 | 0.178964 | 0.101727 | 0.072677 |
| Speed-Up | 2048 | 0.0 | 0.179227 | 0.256464 | 0.285514 |
| Time | 4096 | 1.485508 | 0.814531 | 0.400118 | 0.240462 |
| Speed-Up | 4096 | 0.0 | 0.670977 | 1.085390 | 1.245046 |

11. Code to produce the matrix setup times on different computers can be found in `2_matrix_1.c` and executed using `mpicc -lm -o 2_matrix_1 2_matrix_1.c; mpirun -H [Computers] ./2_matrix_1 [N]`.

12. You can also produce the table for all pairs of values specified by running the shell script `dist_matrix_setup_time.sh`. This can be done by running

`chmod +x dist_matrix_setup_time.sh;`
`./dist_matrix_setup_time.sh`. Note that you will need to compile the C code first in order to run the shell executable.

13. The table of the matrix setup times on different computers can be found below.

| Type | N | 1 | 2 | 4 | 8 |
|---|---|---|---|---|---|
| Time | 256 | 0.006411 | 0.006387 | 0.006285 | 0.006268 |
| Speedup | 256 | 0.0 | 0.000024 | 0.000126 | 0.000143 |
| Time | 512 | 0.025236 | 0.024671 | 0.025373 | 0.024936 |
| Speedup | 512 | 0.0 | 0.000565 | -0.000137 | 0.000300 |
| Time | 1024 | 0.097689 | 0.097518 | 0.097009 | 0.096958 |
| Speedup | 1024 | 0.0 | 0.000171 | 0.000680 | 0.000731 |
| Time | 2048 | 0.386132 | 0.388187 | 0.381994 | 0.382912 |
| Speedup | 2048 | 0.0 | -0.002055 | 0.004138 | 0.003220 |
| Time | 4096 | 1.526926 | 1.527217 | 1.530152 | 1.526418 |
| Speedup | 4096 | 0.0 | -0.000291 | -0.003226 | 0.000508 |

14. As $N$ increases, the time taken to execute power iteration algorithm increases with a constant number of threads / processes. As the number of threads / processes increases, the time taken decreases with a constant $N$. The time for the algorithm is lower when running the code on the same computer due to the communication time between different machines. This also means that the effect of adding extra processes is limited compared to using the same machine.