Homework 1

Gayathri Shreeya Patnala — co21btech11010
ME5470: Introduction to Parallel Scientific Computing

January 19, 2025

Question 1

The objective is to analyze the sizes of files generated by a C program for storing a matrix in both ASCII and binary formats.

Results:

The table shows the file sizes in disc and memory for different matrix sizes.

Value of `n`	Disc Size (ASCII)	Disc Size (Binary)	Memory Usage
2000	80 MB	31 MB	30 MB
3000	180 MB	69 MB	68 MB
4000	320 MB	123 MB	122 MB
5000	501 MB	191 MB	190 MB

Table 1: File sizes for different values of `n`.

Observations:

- For input size $n = 4000$, we observe that the binary file size (123 MB) closely matches the in-memory size (122 MB).
- However, for ASCII format, the file size (320 MB) is significantly larger, around 3X times the size in-memory.
- These observations are consistent with all values of `n` as we can see in the table above.
- It was also observed that the program takes noticeably longer time to execute when the format flag selected is ASCII format compared to binary format. This is especially true for larger input values of `n`

Inference:

- The binary file size on disc closely matches the in-memory size as both store raw binary data.
- In ASCII format, each double is converted to a human-readable string which requires significantly more bytes than its binary representation.
- Binary format is best suited for saving large datasets because it uses minimal storage and is also faster to write to and read from disk since there is no conversion overhead either.
- ASCII format is useful for smaller datasets, or when human readability is required, but it is inefficient in terms of storage and speed for large data.

Question 2

The objective is to verify whether a given vector is an eigenvector for the matrix and compute eigenvalue if true. A few important points to note about the algorithm:

- The algorithm has to make sure that it only compares the result of non-zero division.
- The algorithm also needs to account for floating point division imprecision. This is the reason why we are using `if (fabs(product[i] - temp * vec[i]) > 1e-6)` instead of directly equating (`==`).

Results:

The following are the results of the program to verify the eigenvectors:

Additional:

I have also included code to calculate the time taken for the verification of eigenvector. These times are logged in the file `timetaken.log`. The times taken are also included below along with a trend line showing how the time taken varies with the matrix size `n`.

File Name	Is eigenvector	Value
vec_000003_000001.in	Yes	-6.000000
vec_000003_000002.in	Yes	-6.000000
vec_000003_000003.in	Yes	-1.000000
vec_000003_000004.in	No	-
vec_000005_000001.in	Yes	0.268098
vec_000005_000002.in	No	-
vec_000005_000003.in	Yes	0.986875
vec_000005_000004.in	Yes	1.399039
vec_000050_000001.in	No	-
vec_000050_000002.in	Yes	0.479628
vec_000050_000003.in	Yes	1.337887
vec_000050_000004.in	No	-
vec_000080_000001.in	Yes	0.333018
vec_000080_000002.in	Yes	0.493142
vec_000080_000003.in	Yes	0.939275
vec_000080_000004.in	No	-

Table 2: Eigenvector Check Results

n	Average Time Taken
3	0.00000025
5	0.00000025
50	0.00000325
80	0.00000800

Table 3: Average Time Taken for Different Values of n

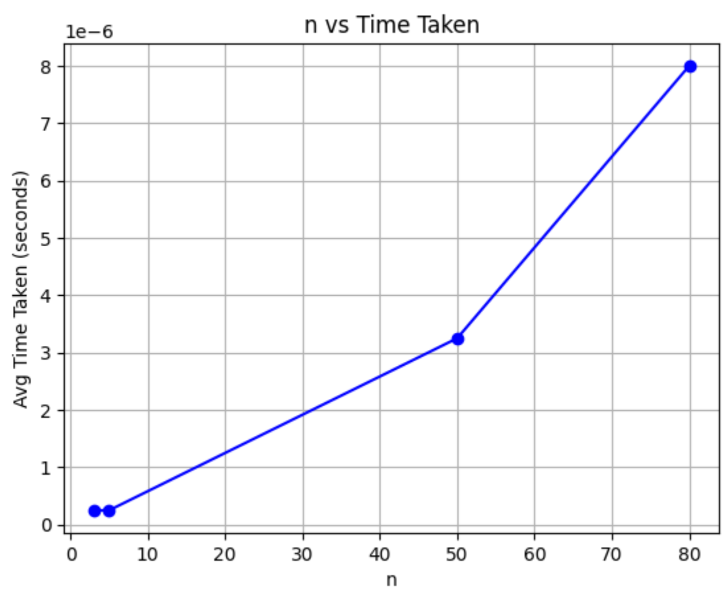

Figure 1: Time taken for verifying eigenvector