ME5470 : Introduction to Parallel Scientific Computing

Homework 1 - Report

Question 1:

(a) The size for the output file ‘array_000080_asc.out’ (Format = ASCII) comes out to be ~320MB while that for ‘array_000080_bin.out’ (Format = Binary) was ~123MB.

(b) To estimate the size of the array in memory:

1. **Memory Size Calculation:**
    - A `double` takes **8 bytes**.
    - The total number of elements in the array is n^2.
    - Hence, the size in memory is: Memory size=8×n2 bytes.
2. **Size on Disk for ASCII vs Binary:**
    - **ASCII Format:** Each double is stored as a string representation with 15-decimal precision. The size per element depends on the number of characters used, typically around **20–25 bytes per double**, including spaces/newlines. Disk size (ASCII)≈25×n2 bytes = 25x4000x4000 bytes ~ 320MB.
    - **Binary Format:** Each double is stored in **8 bytes**, matching its in-memory size: Disk size (binary)=8×n2 bytes = 8x4000x4000 ~ 123MB.
3. **Comparison of Sizes:**
    - **Memory:** The size is always 8×n2 ~ 123MB.
    - **Disk:**
        - ASCII files are significantly larger due to text representation, approximately **3x–4x** the size of binary files.
        - Binary files store the exact data, making them much smaller.
4. **Best Format for Large Data:**
    - **Binary format** is better suited for saving large datasets because:
        - It consumes less disk space.
        - It is faster to write and read since no conversions between numbers and text are required.
    - **ASCII format** may be preferable for smaller datasets when human readability is important, but it becomes impractical for large datasets due to its increased size and slower I/O operations.

Question 2:

Output images:

(a) When ‘n = 3’ :

```
priyesh@Priyesh:~/github-classroom/IITHME5470/hw1-priyeshj1$ ./q2.out
vec_000003_000001.in : Yes : -6.000000000000000
vec_000003_000002.in : Yes : -6.000000000000000
vec_000003_000003.in : Yes : -1.000000000000000
vec_000003_000004.in : Not an eigenvector
```

(b) When ‘n = 5’:

```
priyesh@Priyesh:~/github-classroom/IITHME5470/hw1-priyeshj1$ ./q2.out
vec_000005_000001.in : Yes : 0.268098080462330
vec_000005_000002.in : Not an eigenvector
vec_000005_000003.in : Yes : 0.986875024534868
vec_000005_000004.in : Yes : 1.399038515259468
```

(c) When ‘n = 50’:

```
priyesh@Priyesh:~/github-classroom/IITHME5470/hw1-priyeshj1$ ./q2.out
vec_000050_000001.in : Not an eigenvector
vec_000050_000002.in : Yes : 0.479628234701048
vec_000050_000003.in : Yes : 1.337887289556923
vec_000050_000004.in : Not an eigenvector
```

(d) When ‘n = 80’:

```
priyesh@Priyesh:~/github-classroom/IITHME5470/hw1-priyeshj1$ ./q2.out
vec_000080_000001.in : Yes : 0.333017754867211
vec_000080_000002.in : Yes : 0.493141980754358
vec_000080_000003.in : Yes : 0.939274515847899
vec_000080_000004.in : Not an eigenvector
```

The code checks if x to eigenvector of a matrix(A) by:

1. Calculating b=Ax.
2. Finding the first non-zero component in x and using it to compute λ=bi/xi (if all components are zero, the vector is not valid).
3. Verifying if each component of b satisfies bi=λxi within a small tolerance.

If all conditions are met, x is confirmed as an eigenvector with eigenvalue λ.