M5470 : Introduction to Parallel Scientific Computing

Homework 1

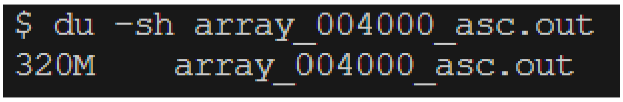
Raman Kumar

ME21BTECH11043

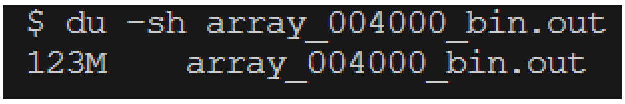
Question 1:

Part A

ASCII file size	320 MB
BINARY file size	123 MB



```
$ du -sh array_004000_asc.out
320M    array_004000_asc.out
```



```
$ du -sh array_004000_bin.out
123M    array_004000_bin.out
```

Conclusion/Understanding:

We can observe that **binary file formats** are more efficient than **ASCII formats** in terms of both storage space and execution time. Specifically:

1. **File Size**: Binary files are compact because they store raw data without any formatting or conversion, unlike ASCII files, which require extra space for human-readable representation.
2. **Execution Time**: Writing binary files is faster as it avoids the overhead of converting data to ASCII characters, making it better suited for performance-critical applications.
3. **Use Cases**:
    - **Binary Files**: Ideal for large datasets, performance optimization, or scenarios where human readability is not required.
    - **ASCII Files**: Useful for debugging, sharing, or situations where human readability and ease of editing are priorities.

Part B

Calculation for Array Size in Memory:

- For n = 4000, the array contains n2n^2 doubles.
- The size of one double is 8 bytes.

$$[\text{Memory Size} = 8 \times n^2 \text{ bytes}]$$

$$Substituting\ (n\ =\ 4000):$$

$$[\text{Memory Size} = 8 \times 4000^2 \text{ bytes} = 8 \times 16{,}000{,}000 \text{ bytes} = 128{,}000{,}000 \text{ bytes}]$$

Converting to megabytes:

$$\text{Memory Size} = \frac{128{,}000{,}000}{1{,}048{,}576} \approx 122.07 \text{ MB}$$

So, the array will take up approximately **122.07 MB** of memory.

Size on Disk:

- **ASCII File**: 320 MB (from your previous observation)
- **Binary File**: 123 MB (from your previous observation)

Comment on Size in Memory vs Size on Disk:

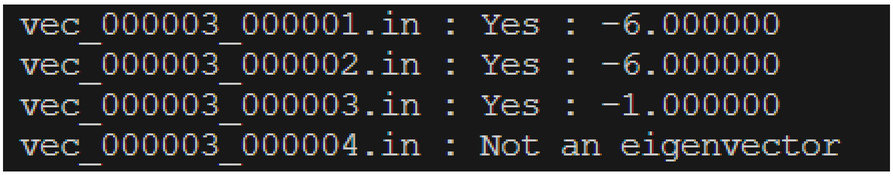
1. **Memory Size vs. Disk Size**:
   - The size of the array in memory (approximately 122.07 MB) is similar to the binary file size (123 MB). This is because binary files store data in the same raw format as it exists in memory, with minimal overhead.
   - The ASCII file, on the other hand, is significantly larger (320 MB) because of the overhead required for storing the human-readable representation of the data.
2. **Best Format for Saving Large Data**:
   - **Binary Format**: The binary format is the best choice for saving large data, as it is much more space-efficient and faster to write/read compared to ASCII. This is particularly true when dealing with large arrays or datasets that do not need to be human-readable, making it ideal for performance and storage optimization.
   - **ASCII Format**: Although larger and slower to process, ASCII is better suited for smaller datasets or when data readability and manual inspection are needed.

For large datasets, **binary format** is preferred for both space and time efficiency, while **ASCII format** can be used for ease of inspection and debugging but at the cost of increased file size and slower execution times.

Question 2)

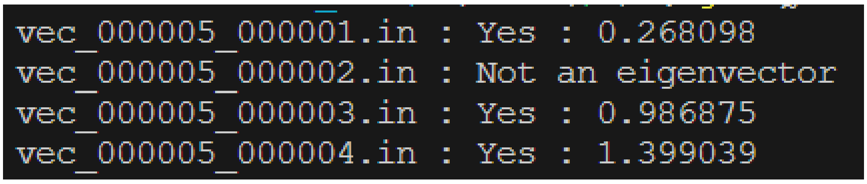
Below is the output for different n:

For n=3



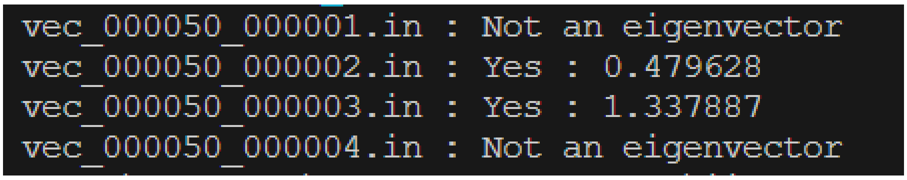
```
vec_000003_000001.in : Yes : -6.000000
vec_000003_000002.in : Yes : -6.000000
vec_000003_000003.in : Yes : -1.000000
vec_000003_000004.in : Not an eigenvector
```

For n=5



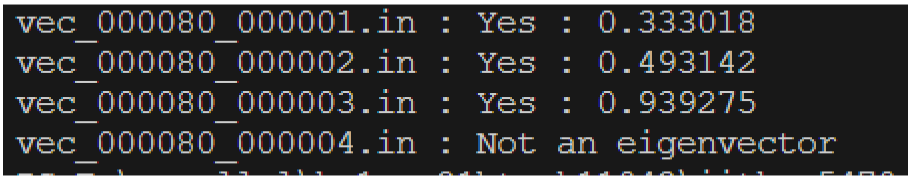
```
vec_000005_000001.in : Yes : 0.268098
vec_000005_000002.in : Not an eigenvector
vec_000005_000003.in : Yes : 0.986875
vec_000005_000004.in : Yes : 1.399039
```

For n=50



```
vec_000050_000001.in : Not an eigenvector
vec_000050_000002.in : Yes : 0.479628
vec_000050_000003.in : Yes : 1.337887
vec_000050_000004.in : Not an eigenvector
```

For n=80



```
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 : Not an eigenvector
```