DS298: Random Variates in Computation
Assignment 2

N Surya (22938)

March 10, 2024

Randomized matrix multiplication

Implementation:

1. Random square matrices of size $n = 50, 100, 200, 400, 800, 1600$ are generated using algorithm-2 (**A2**) and algorithm-3 (**A3**) as given in the question using singular value decomposition.

2. The distribution of singular values for Class-I (**C1**), Class-II (**C2**) and Class-III (**A3**) respectively are also as given in the question.

3. The number of outer products that are used to approximate the matrix multiplication through the algorithm-1 (**A1**) is $c = log_2(n), (log_2(n))^2, 0.2n$.

4. For the final results, the errors calculated in both approaches (**A2** and **A3**) are averaged over **10 trials**.

5. The code is written with good modularity to allow for minor tweaks in case of experimentation. The results are well-organized in a nested directory for ease of access and comparison. Some redundant file directories are created to aid with the modularity as well.

6. The total run-time was roughly 9 minutes for both approaches with 10 trials. **A2** takes more time than **A3** as one of the orthogonal matrices is reused in **A3**.

7. The compute time also increases with c value as expected. $log_2(n)$ consumes less time than $(log_2(n))^2$ while $(log_2(n))^2$ consumes less than $0.2n$.

8. All the plots are at the end of the report. They are referenced within the report using figure numbers.

Classwise distribution of singular values:

- Fig.1 clearly shows the trend of singular values for classes **C1**, **C2**, **C3**.

- In **C1**, the singular values decay exponentially with column indexes. This means that the matrix is effectively low rank.

- In **C2**, the values decay in a linear fashion, hence not as effectively low rank as **C1**.

- In **C3**, the matrix is clearly no where close to low rank.

This means that for a given value of c, **A1** is expected to perform better on **C1** than on **C2**, and **C2** than on **C3**. This will become evident when examining the error plots.

Rejection sampling:

The rejection sampling process also helps us see the difference between the classes. But it is not as helpful as the previous analysis as the probability mass function (PMF) is taken for the column and row norms of matrices which include the effect of all the singular values. But the analysis is still meaningful as can be observed from the plot.

- It can be readily observed from Fig.(4) that for a given n, c pair, the PMF ratio $(p_k/p_{k,max})$ gives the relative importance (*norm-wise PMF*) of columns.
- It can be observed that in **C1**, especially, some column indices seem to be more *important* than the others.
- In the other classes, the relative importance of one column index with respect to others is not too much. This can be verified using the fact that the vertical distance between the lowest and the highest PMF ratio is not as appreciable in these classes as in **C1**.

Error analysis:

As expected, as we sample more and more columns for outer products, we get lower true and relative error. This is inferred from Fig.(2) and Fig.(3). These are all scaled to be log-log plots.

True error

1. The magnitude of true error increases as the matrix size increases for all schemes. This can be observed in Fig.(2).
2. It can be seen that up to a threshold value of n, $(log_2(n))^2$ performs better than $0.2n$, which is the most conservative of the three. This is because for lower n values, $(log_2(n))^2$ is greater than $0.2n$.
3. For a given algorithm, the true errors for all the classes are comparable with the most conservative estimate of c for a given n outperforming the other c values.

Relative error

1. Fig.(3) clearly outlines the difference between the algorithms and classes.
2. The same pattern for the threshold value of n is observed here as well. But more importantly, the relative error is lower for **A3** than for **A2**, at least for the effectively low rank scenario. This is expected from the way the matrices are constructed in the algorithms. **A3** seems to preserve the *low rank-ness* of $A\&B$ more than **A2**.
3. The errors in **C1** are lesser than other classes as **C1** product is effectively low-rank, meaning that a low-rank outer product summation would approximate it well. And as expected, **C3** shows higher error than **C2** matrices in both true and relative errors.

Plots

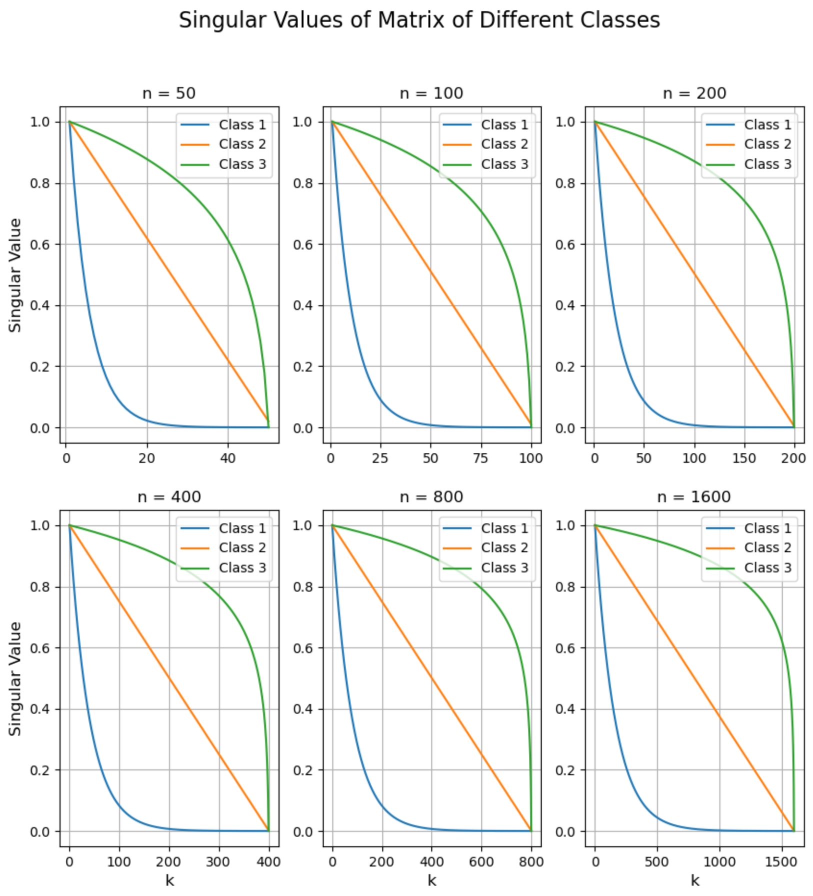
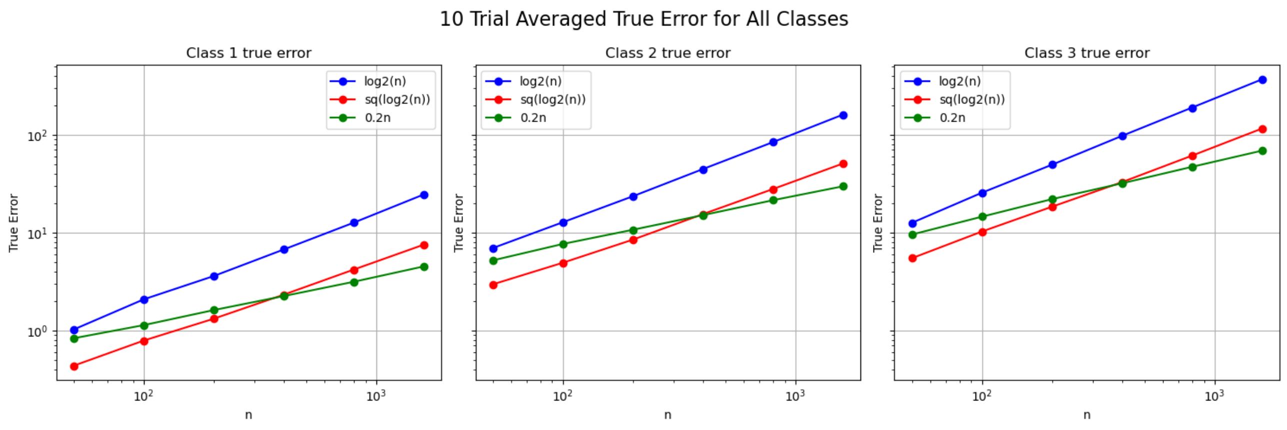
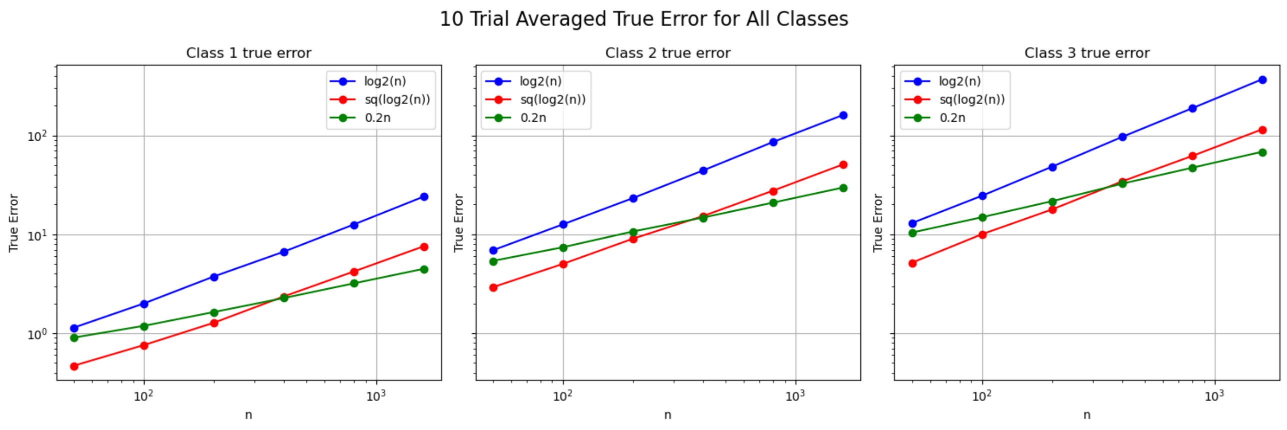


Fig. 1: Comparison of singular values for all the classes

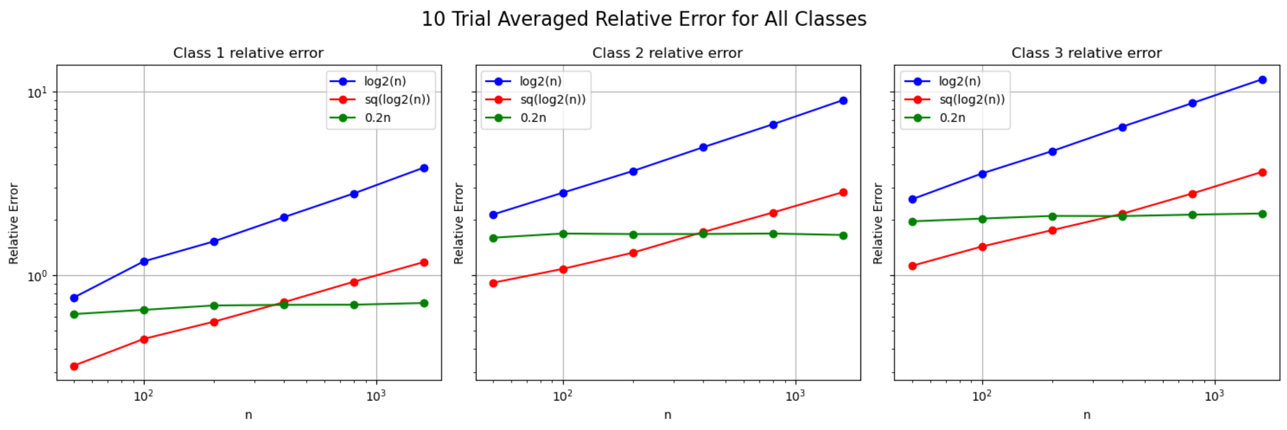


(a) Using **A2**

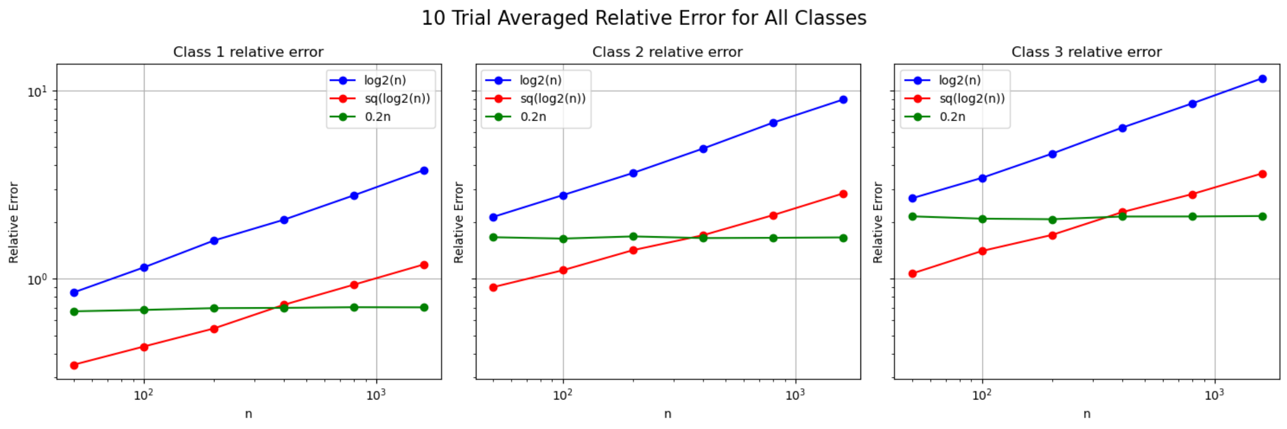


(b) Using **A3**

Fig. 2: True absolute error comparison (*frobenius norm*) log-log

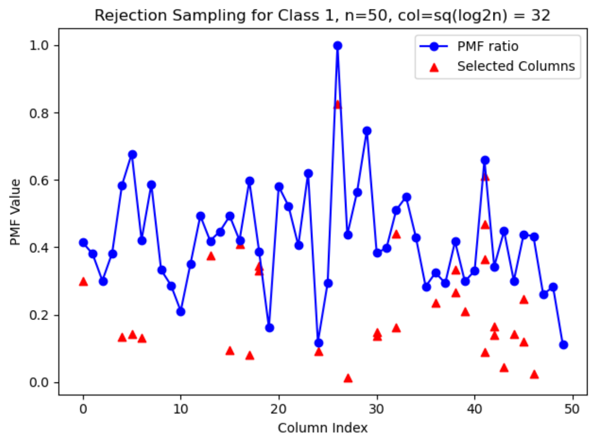


(a) Using **A2**

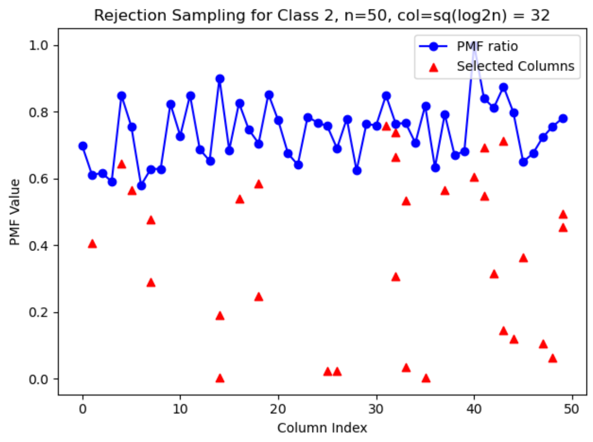


(b) Using **A3**

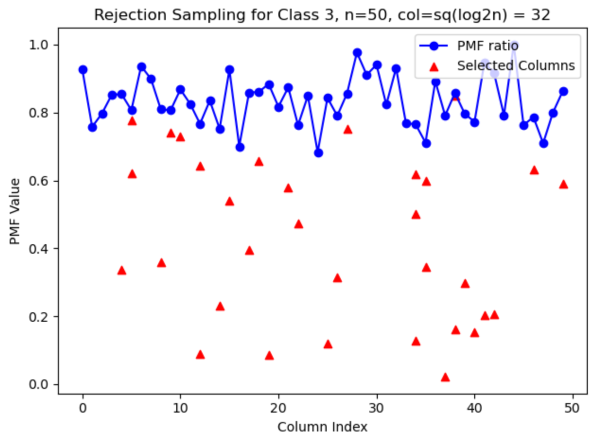
Fig. 3: Relative absolute error comparison (*frobenius norm*) log-log



(a)



(b)



(c)

Fig. 4: Rejection sampling - all classes for $(log_2(n))^2$