MCSC 6030G : High Performance Computing
Assignment 4: Random Matrices with MPI

Parikshit Bajpai
100693928

1 Introduction

This study is aimed at implementing the master-slave MPI model for computing maximum Eigenvalues of random matrices and finding the distribution of the Eigenvalues for a large number of matrices.

For a symmetric matrix $A[n \times n]$, with elements normally distributed according to eqn 1, the Eigenvalues for any solution toare defined as the solution to the Eigenvalue problem $\mathbf{Av} = \lambda\mathbf{v}$, where the vector $\mathbf{v}$ is called as the Eigenvector.

$$A_{ii} \sim N(0,2)(1 \leq i \leq n) \qquad A_{ij} \sim N(0,1)(1 \leq i \leq n\ 2 \leq j \leq n) \qquad A^t = A \tag{1}$$

For a random matrix A, the maximum of the Eigenvalues, λ_{max} should be a random number and the distribution of such maximum Eigenvalues can be determined.

2 Methodology

2.1 Objective

In the present assignment, the aforementioned distribution of maximum Eigenvalues has been computed for large matrices using master-slave MPI model and the speed-up and efficiency of parallelization has been examined.

2.2 Machine Configuration

The initial computations for the present study were performed on the local machine with the following specifications:

2.2.1 Local Machine

Manufacturer: Asus
Processor: Intel Core i7 -7700HQ (4 physical cores, 4 hyperthreads)
RAM: 16 GB
Operating System: Ubuntu 18.04

2.3 Implementation

The master-slave model was implemented for the present study with the master tasked with keeping a track of the numbver of computations performed, receiving and collecting the outputs computed by each slave, and issuing instructions for the ideal slaves. Each of the slaves is tasked with generating a random matrix, computing the maximum Eigenvalue and sending the results back to the master. The pseudo code explaining the adopted methodology is as follows:

```
Algorithm 1: MAIN PROGRAM
  Size of matrices ($n$)
  Number of maximum eigenvalues to be collect ($\lambda_{max}$) ($Ni$)
  Set $flag = 1$, which means that slaves can continue working
  Set $jobcount = 0$, which means that master did not receive any $\lambda_{max}$ so far
  Start implementation Master vs Slave method

  if $PROC = SLAVE$ then
      Call MKL (Math Kernel Library) to Generate matrix randomly following normal
        distribution $N(0,1)$
      Make $A$ symmetric and multiply its diagonal by 2 ($2 * A(n,n)$), adjusting matrix to
        problem proposed
      Calculate vector $W$ of eigenvalues (Using Lapack or Power Method)
      Define $\lambda_{max} = \max(\lambda_i)$ of vector $W$
      Call mpi and $SEND$ $\lambda_{max}$ to Master
      Call mpi and $RECEIVE$ $flag$ to verify if is there any job left
  end if
  if $PROC = MASTER$ then
      Define and allocate $G$ as a vector to store all $\lambda_{max}$ to be received from slaves
      Set $ith = 1$, which is the position at vector $G$ that $\lambda_{max}$ to be receive will be allocated
      while $ith < Ni$ do
          Call MPI and $RECEIVE$ $\lambda_{max}$ from slaves ($proc_i$)
          Increment $jobcount = jobcount + 1$
          if $jobcount \leq (Ni - Numproc + 1)$ then
              $flag = 1$
              Call MPI and $SEND$ $flag$ to ($proc$) that hand in the $\lambda_{max}$ at that point
              else
                  $flag = 0$
                  Call MPI and $SEND$ $flag$ to ($proc$) that hand in the $\lambda_{max}$ at that point
          Increment $ith = ith + 1$
          end if
          end do
  end if
```

For the implementation of the above code, the random numbers were seeded using the Marsenne Twister pseudorandom number generator included in the Intel Math Kernel Library (MKL). For computing the maximum eigenvalues, both the shifted power iterations and the LAPACK subroutine *DSYEV* were implemented. The LAPACK subroutine is a $\mathcal{O}(n^3)$ while the power iteration is a $\mathcal{O}(n^2)$ method. While this means the power method is normally much quicker, there can be issues related to convergence especially if the Eigenvalues are very close. Furthermore, the compute time in the power iteration would also depend on the initial approximation of the Eigenvector.

2.4 Computational Experiments

The following studies were performed as part of this experiment:

1. Analyze the random distribution of the maximum Eigenvalues for different matrix sizes and different desired number of Eigenvalues.

2. Compare wallclock times for LAPACK subroutines and shifted power method.

3. Compare wallclock times for different number of processors.

3 Results and Discussion

The obtained frequency distribution for a matrix of size 1000×1000 for 1000 matrices has been shown below in figure 1

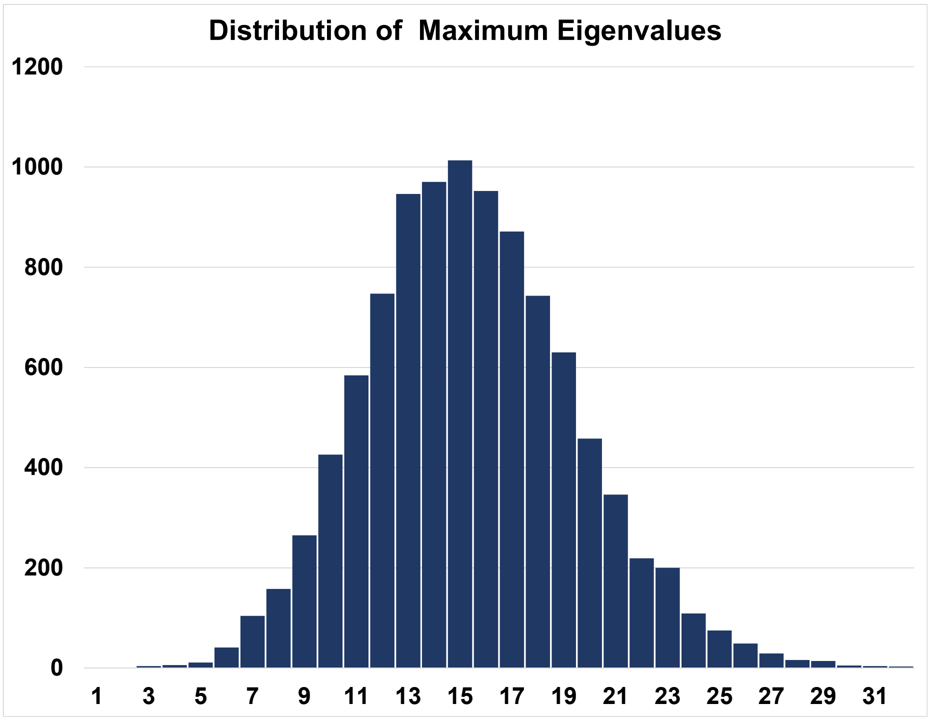

Figure 1: Distribution of maximum Eigenvalues for a matrix of size 1000×1000 for 1000 matrices.

The obtained random distributions of maximum Eigenvalues for different matrix sizes n and different numbers of desired Eigenvalues N using both BLAS and shifted power iterations for different matrix sizes are presented in figures 2 & 3.

In order to identify which distribution could fit in the outputs obtained, a distribution analysis was performed usin IBM-SPSS (Statistical Package Software) and a Komolgorov-Smirnov test was applied to the 10 Eigenvectors. Komolgovorv-Smirnov (KS) is a nonparametric test of the equality of continuous, one-dimensional probability distributions that can be used to compare a sample with a reference probability distribution (one-sample KS test), or to compare two samples (two-sample KS test). The Kolmogorov-Smirnov tests tell us that whether or not a dataset follows a specified distribution by comparing the significance of the data with a given predetermined significance level,

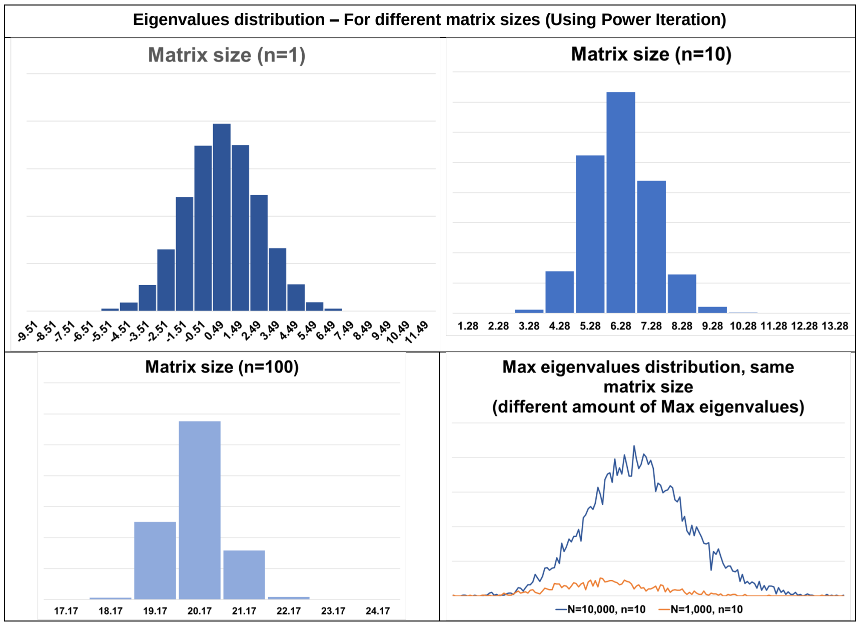

Figure 2: Distribution of maximum Eigenvalues for computations using shifted power iterations.

which in the present case was taken as 0.05. These results are presented in figure 4 The results show that the Eigenvalues generally show a Tracy-Widom distribution that has a Gaussian distribution as one limit (small matrices) and a log-normal distribution as the other limit (very large matrices). The results also show an increase in the average of maximum Eigenvalues as the matrix size is increased.

The Eigenvalues were computed for different matrix sizes n and different numbers of desired Eigenvalues N using both BLAS and shifted power iterations and the obtained wallclock times were compared. The results are presented below in figure 5 & 6.

In the available literature, Almeida et al. [1] have shown that wallclock times are inversely proportional to the number of processors and Elenin et al. [2], have shown that wallclock times as small as those of the order of 0.65s can be obtained. As seen in figure 6, the results as, sho wexpected, a decrease in wall-clock times as the number of processors is increased from one to three. However, since a further increase means that no physical cores are available and hyperthreads are used, we do not observe any further decrease in wallclock times for an increase in total number of processors beyond 4. Furthermore, the wallclock times of the power iterations were observed to be smaller than the wallclock times using LAPACK.

4 Conclusion

In conclusion, the computational experiments verify the expected outcomes of the study. MPI implementation of the master-slave model resulted in a reduced wallclock times as the number of slaves was increased. The power iteration was quicker than the LAPACK subroutine and the Eigenvalues showed a log-normal distribution with an increase in the average value as the size of the matrix was increased.

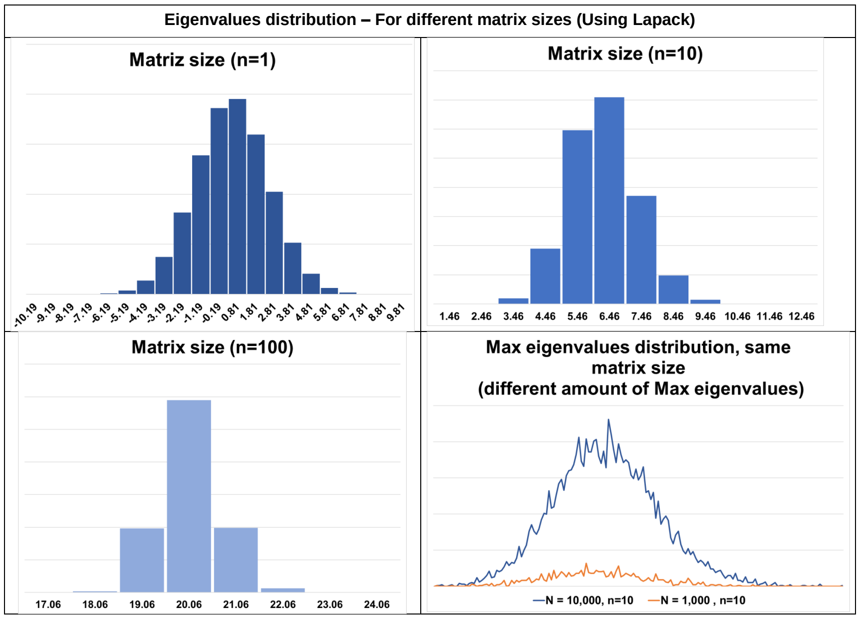

Figure 3: Distribution of maximum Eigenvalues for computations using LAPACK.

References

[1] F. Almeida, V. Blanco, A. Cabrera, and J. Ruiz, “Modeling energy consumption for master—slave applications,” *J. Supercomput.*, vol. 65, pp. 1137–1149, Sept. 2013.

[2] S. A. Elenin and M. A. ElSoud, “Evaluation of matrix multiplication on an mpi cluster,” pp. Vol 11 – No 1, International Journal of Electric Computer Sciences IJECS-IJENS, 2013.

Komolgorov-Smirnov test for Normal / Log-Normal distribution evaluation (Significance level of 0.05)

Method	Amount of Eigenvalues calculated (N)	Size of Matrix	Kolmogorov-Smirnov Statistic Test (for Normal distribution)	P-value (for Normal distribution)	Is it Normal?	Kolmogorov-Smirnov Statistic Test (for Log-Normal distribution)	P-value (for Log-Normal distribution)	Is it Log-Normal?
Lapack	1,000,000	n=1	0.000	0.150	No	0.000	0.500	No
		n=10	0.014	0.010	Yes	0.001	0.011	Yes
		n=100	0.017	0.010	Yes	0.001	0.034	Yes
	10,000	n=10	0.015	0.010	Yes	0.006	0.250	No
	1,000	n=10	0.023	0.150	No	0.013	0.500	No
Power Method	1,000,000	n=1	0.001	0.150	No	0.001	0.250	No
		n=10	0.014	0.010	Yes	0.001	0.001	Yes
		n=100	0.017	0.010	Yes	0.001	0.016	Yes
	10,000	n=10	0.020	0.010	Yes	0.008	0.077	No
	1,000	n=10	0.028	0.064	No	0.018	0.500	No

Figure 4: Analysis of the obtained maximum Eigenvalue distributions.

Experiment Wall time, Speed-up and Efficiency

Method	Matrix size (n)	Wall time 2 Procs	Wall time 4 Procs	Speed-up 4 Procs	Efficiency 4 Procs	Wall time 8 Procs	Speed-up 8 Procs	Efficiency 8 Procs
Lapack	1	77.88	57.66	1.35	0.45	56.69	1.37	0.20
	10	95.21	57.56	1.65	0.55	60.57	1.57	0.22
	100	532.46	178.26	2.99	1.00	120.61	4.41	0.63
Power Method	1	68.25	55.82	1.22	0.41	56.21	1.21	0.17
	10	83.20	55.11	1.51	0.50	56.25	1.48	0.21
	100	535.15	180.33	2.97	0.99	139.57	3.83	0.55

Figure 5: Wallclock times and performance parameters for different tests.

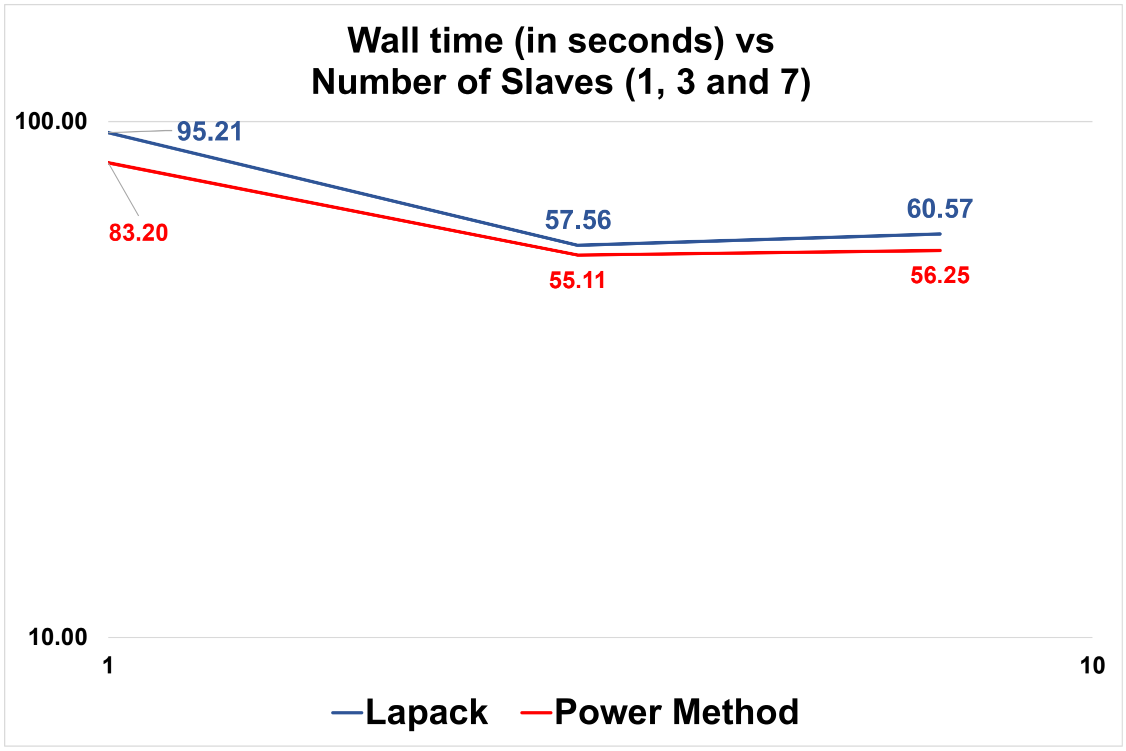

Figure 6: Comparison of wallclock times