Assignment 1

Mathematical Foundations For Data Science

Akhil Sudhakaran

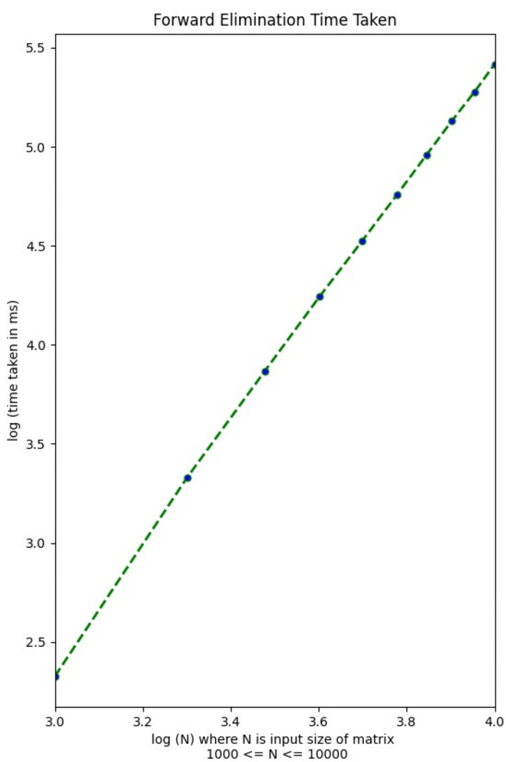
2021MT12054

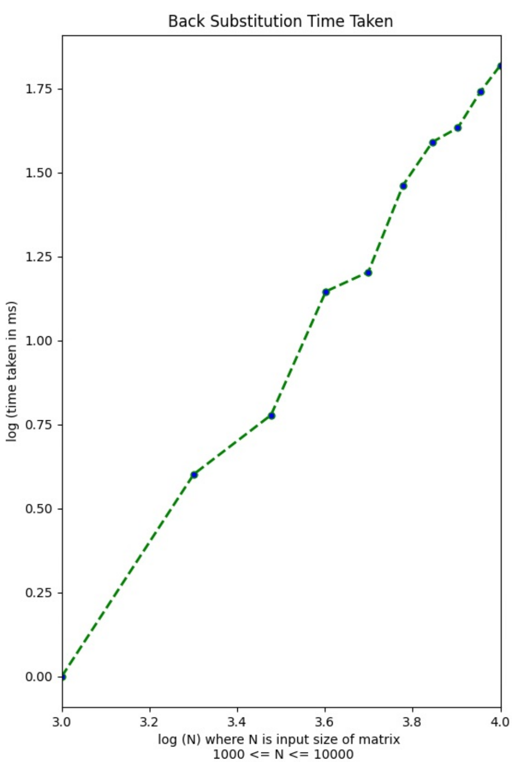
Q1(a)

Draw the log–log plot of n versus the time taken for forward elimination and backward substitution (as separate graphs) by taking values of n between 1000 and 10000 in steps of 1000. Determine the time taken for a single computation in your machine (by averaging over 1000 runs) and compare the time taken with the actual time derived in the class. This should give the time taken for the partial pivoting.

→ NOTE: THE METRICS USED HERE ARE COLLECTED WITH GAUSSIAN ELIMINATION I HAD IMPLEMENTED IN C++ WITH PARTIAL PIVOTING. PYTHON WAS USED TO PLOT THE COLLECTED METRICS IN A LOG-LOG PLOT USING MATPLOTLIB. THE CODE AND THE METRICS COLLECTED (FOR MY MACHINE) CAN BE FOUND IN THE GIT REPOSITORY HERE: https://git.io/JuAyL

→ BELOW YOU CAN SEE 2 LOG-LOG PLOTS WHERE THE X AXIS DENOTES THE INPUT SIZE OF THE MATRIX "N" WHERE N RANGES BETWEEN 1000 TO 10,000 (IN STEPS OF 1000 AS MENTIONED), AND Y AXIS DENOTES THE TIME TAKEN BY FORWARD ELIMINATION AND BACK SUBSTITUTION RESPECTIVELY





BY RUNNING SEVERAL OPERATIONS ON MY MACHINE, THE TIME TAKEN FOR A SINGLE COMPUTATION BY AVG. OVER 1000 RUNS = 0.3170 nS

$$\boxed{C = 0.3170 \times 10^{-9}\ S} \quad (1)$$

FROM DISCUSSIONS IN CLASS, THE NUMBER OF COMPUTATIONS FOR FORWARD ELIMINATION WHEN INPUT SIZE OF MATRIX = N IS GIVEN AS :

$$\underbrace{\frac{N(N-1)(2N-1)}{6}}_{\text{(ADDITIONS)}} + \underbrace{\frac{N(N-1)(2N-1)}{6}}_{\text{(MULTIPLICATIONS)}} + \underbrace{\frac{N(N-1)}{2}}_{\text{(DIVISIONS)}} \simeq O\left(\frac{2}{3}N^3\right) \quad (2)$$

NOTE THAT (2) DOES NOT INCLUDE TIME FOR PARTIAL PIVOTING

FROM THE PLOT SHOWN ABOVE FOR FORWARD ELIMINATION, THE TIME TAKEN FOR FORWARD ELIMINATION FOR A MATRIX OF SIZE 1000 × 1000 (N = 1000) [BY CODE IN MY COMPUTER] = 212 mS

$$\boxed{T_F = 0.212\ S} \quad (3)$$

LET TIME TAKEN FOR PARTIAL PIVOTING = T_P AND N = 1000

FROM (1), (2) AND (3) WE CAN SAY THAT:

$$\boxed{\frac{2}{3}N^3 \times C + T_P = T_F}$$

$$\Rightarrow T_P = T_F - \frac{2}{3}N^3 C \Rightarrow T_P = 0.212 - \frac{2}{3} \times 1000^3 \times 0.3170 \times 10^{-9}$$

$$\Rightarrow T_P = 0.212 - \frac{2}{3} \times 0.3170$$

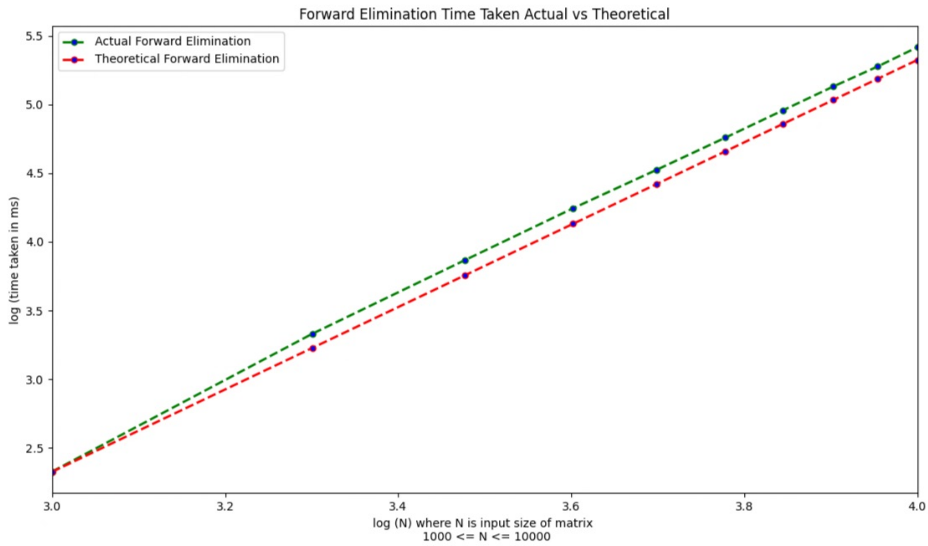
$$\Rightarrow T_P = 6.667 \times 10^{-4}\ S \Rightarrow \boxed{T_P = 0.6667\ mS}$$

TIME REQUIRED FOR PARTIAL PIVOTING FOR A 1000×1000 MATRIX IS APPROXIMATELY 0.6667 mS

BELOW ARE THE METRICS COLLECTED FOR TIME TAKEN FOR FORWARD ELIMINATION, BACK SUBSTITUTION BY CODE AND THE THEORETICAL TIME TAKEN FOR FORWARD ELIMINATION (FOR C = 0.3170 mS) FOR SEVERAL VALUES OF N:

N	TIME TAKEN BY FORWARD ELIMINATION IN CODE (mS)	TIME TAKEN BY FORWARD ELIMINATION THEORETICALLY (mS)	TIME TAKEN BY BACK SUBSTITUTION IN CODE (mS)
1000	212	211.175	1
2000	2144	1690.03	4
3000	7357	5704.57	6
4000	17552	13522.80	14
5000	33541	26412.70	16
6000	57401	45642.30	29
7000	91063	72479.60	39
8000	135565	108193	43
9000	189467	154049	55
10000	261535	211317	66

USING THE ABOVE METRICS WE CAN PLOT A LOG-LOG PLOT OF N VS TIME, FOR THE FORWARD ELIMINATION TIME TAKEN BY CODE AND THE THEORETICAL TIME OBTAINED WITH C:



FROM THESE DETAILS WE CAN INFER THAT THE TIME TAKEN FOR PARTIAL PIVOTING INCREASES AS N INCREASES.

Q1(b)

Solve the system A5×5x = b, with random entries and display your results.

→ THE RANDOM $A_{5\times5}|b$ MATRIX GENERATED BY CODE IS SHOWN BELOW:

$$\underbrace{\begin{bmatrix} -5.4058 & 8.0986 & -2.1016 & -6.2509 & 6.2680 \\ -8.1548 & -5.7482 & -4.1585 & 2.7407 & 0.5418 \\ -4.4106 & -4.7404 & -3.5668 & -6.4902 & -0.4430 \\ 8.7821 & 0.7141 & -4.6117 & -0.4947 & 0.9228 \\ -5.8557 & 6.9388 & -1.5193 & 7.1342 & 5.9202 \end{bmatrix}}_{A} \; \underbrace{\begin{bmatrix} -7.7855 \\ -2.6209 \\ 5.7870 \\ -6.6115 \\ -3.3575 \end{bmatrix}}_{b}$$

THE CODE DISPLAYS THE FOLLOWING MATRIX AFTER FORWARD ELIMINATION:

$$\begin{bmatrix} 8.7821 & 0.714071 & -4.61168 & -0.494739 & 0.922793 & -6.61152 \\ 0 & 8.53819 & -4.94037 & -6.55549 & 6.83602 & -11.8552 \\ 0 & 0 & -0.3038 & 12.4974 & 0.598793 & 2.52963 \\ 0 & 0 & 0 & -356.352 & -13.0614 & -73.7028 \\ 0 & 0 & 0 & 0 & 0.257393 & -13.4192 \end{bmatrix}$$

THE SOLUTION VECTOR CALCULATED AFTER BACK SUBSTITUTION:

$$\boxed{X = \begin{bmatrix} -10.0257 \\ 28.1127 \\ -23.9653 \\ 2.1176 \\ -52.1356 \end{bmatrix}}$$

Q2

Implement Gauss Jordan method in code and find the inverse of a 6 x 6 random matrix which is non-singular.

→ THE RANDOM $A_{6\times6}|I$ MATRIX GENERATED BY CODE IS SHOWN BELOW:

$$\left[\begin{array}{cccccc|cccccc}
6.5343 & 4.7668 & -2.7533 & 2.7367 & 5.3331 & -5.8520 & 1 & 0 & 0 & 0 & 0 & 0 \\
-3.3795 & 8.5150 & -5.8418 & 6.6935 & -2.1089 & -2.4567 & 0 & 1 & 0 & 0 & 0 & 0 \\
1.4274 & -3.2727 & 3.4050 & 5.2256 & 4.9136 & -1.1586 & 0 & 0 & 1 & 0 & 0 & 0 \\
2.5987 & 8.7759 & 5.3332 & -4.7166 & 0.1228 & -6.2648 & 0 & 0 & 0 & 1 & 0 & 0 \\
7.1193 & 8.5643 & -5.5102 & 0.7762 & -4.8795 & -0.9502 & 0 & 0 & 0 & 0 & 1 & 0 \\
-3.3497 & 5.7070 & -4.1591 & -8.7062 & -3.6795 & 6.9901 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}\right]$$

→ THE $I|A^{-1}$ MATRIX GENERATED AFTER GAUSS JORDAN BY CODE IS:

$$\left[\begin{array}{cccccc|cccccc}
1 & 0 & 0 & 0 & 0 & 0 & 0.0077468 & -0.06607 & 0.053438 & -0.008873 & 0.093014 & -0.0031869 \\
0 & 1 & 0 & 0 & 0 & 0 & -0.014297 & 0.043019 & 0.11109 & 0.050403 & 0.025931 & 0.070263 \\
0 & 0 & 1 & 0 & 0 & 0 & -0.10049 & -0.0093517 & 0.1461 & 0.076813 & 0.03538 & 0.010454 \\
0 & 0 & 0 & 1 & 0 & 0 & -0.054974 & 0.048732 & 0.11869 & -0.011439 & 0.047866 & -0.012969 \\
0 & 0 & 0 & 0 & 1 & 0 & 0.10242 & -0.00028254 & 0.085085 & -0.009366 & -0.069224 & 0.08195 \\
0 & 0 & 0 & 0 & 0 & 1 & -0.058964 & -0.0118 & 0.21446 & -0.018877 & 0.067634 & 0.11737
\end{array}\right]$$

$$\boxed{A^{-1} = \begin{bmatrix}
0.0077468 & -0.06607 & 0.053438 & -0.008873 & 0.093014 & -0.0031869 \\
-0.014297 & 0.043019 & 0.11109 & 0.050403 & 0.025931 & 0.070263 \\
-0.10049 & -0.0093517 & 0.1461 & 0.076813 & 0.03538 & 0.010454 \\
-0.054974 & 0.048732 & 0.11869 & -0.011439 & 0.047866 & -0.012969 \\
0.10242 & -0.00028254 & 0.085085 & -0.009366 & -0.069224 & 0.08195 \\
-0.058964 & -0.0118 & 0.21446 & -0.018877 & 0.067634 & 0.11737
\end{bmatrix}}$$

NOTE: CODE IMPLEMENTATIONS CREATED AS PART OF THIS ASSIGNMENT CAN BE FOUND HERE: https://git.io/JuAyL