

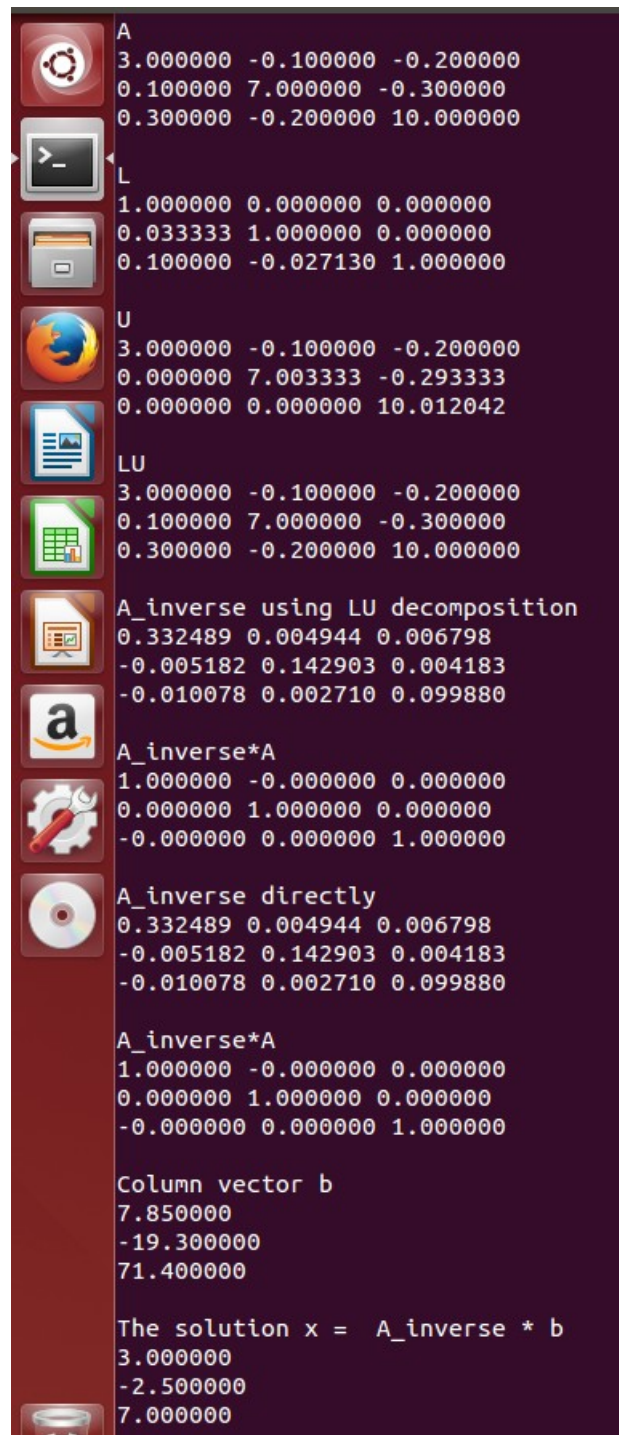
EE1103 Numerical Methods

Assignment 9 – LU Decomposition

Rajat Vadiraj Dwaraknath – EE16B033

27th October, 2016

The LU decomposition of a square matrix **A** was found using Gaussian elimination. This was implemented in the .c file **luDecomp.c**. It also has a function to invert a given square matrix. A screenshot of the required output is shown below:



```
A
3.000000 -0.100000 -0.200000
0.100000 7.000000 -0.300000
0.300000 -0.200000 10.000000

L
1.000000 0.000000 0.000000
0.033333 1.000000 0.000000
0.100000 -0.027130 1.000000

U
3.000000 -0.100000 -0.200000
0.000000 7.003333 -0.293333
0.000000 0.000000 10.012042

LU
3.000000 -0.100000 -0.200000
0.100000 7.000000 -0.300000
0.300000 -0.200000 10.000000

A_inverse using LU decomposition
0.332489 0.004944 0.006798
-0.005182 0.142903 0.004183
-0.010078 0.002710 0.099880

A_inverse*A
1.000000 -0.000000 0.000000
0.000000 1.000000 0.000000
-0.000000 0.000000 1.000000

A_inverse directly
0.332489 0.004944 0.006798
-0.005182 0.142903 0.004183
-0.010078 0.002710 0.099880

A_inverse*A
1.000000 -0.000000 0.000000
0.000000 1.000000 0.000000
-0.000000 0.000000 1.000000

Column vector b
7.850000
-19.300000
71.400000

The solution x = A_inverse * b
3.000000
-2.500000
7.000000
```

Therefore, the following solutions were obtained:

$$\begin{aligned}x_1 &= 3 \\x_2 &= -2.5 \\x_3 &= 7.8\end{aligned}$$

The matrix inversion was done by two methods – direct Gauss elimination and **LU** decomposition.

The **LU** decomposition involved inverting the **L** and **U** matrices individually using forward and backward substitution respectively, and multiplying their inverses in the reverse order to obtain the inverse of the original matrix **A**.

The direct elimination method incorporates the forward substitution steps implicitly as the elimination steps are being done, and then finally backward substitutes to invert the resulting upper triangular matrix to obtain the inverse of **A**.

We conclude by observing that the Gaussian elimination method for matrix inversion grows as **$O(n^3)$** , which is much faster than the cofactor and determinant method which grows **exponentially**.