EXERCISE 1: GEM by hands

- Using the Gauss elimination method, solve, by pencil and paper, the linear system below; then check by Matlab the solution.

$$\begin{bmatrix} 4 & 0 & 12 \\ -2 & 6 & -3 \\ 1 & 2 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

- Same as before, with

$$\begin{bmatrix} -5 & 3 & 4 \\ 10 & -8 & -9 \\ 15 & 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 5 \\ 1 \end{bmatrix}$$

- Same as before, with

$$\begin{bmatrix} 1 & -3 & 4 \\ -1 & 5 & -3 \\ 4 & -8 & 23 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 12 \\ -12 \\ 58 \end{bmatrix}$$

EXERCISE 1: solution 1st system

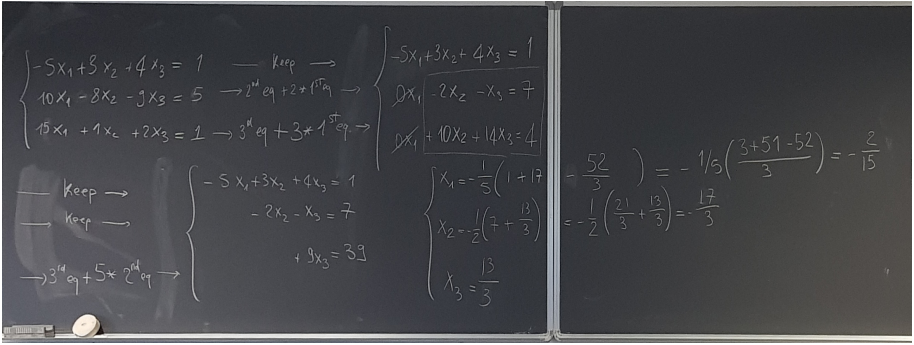
$$\begin{array}{cc}\begin{bmatrix} 4 & 0 & 12 \\ -2 & 6 & -3 \\ 1 & 2 & 5 \end{bmatrix} & \begin{array}{c} r_1 \\ r_2 \\ r_3 \end{array}\end{array} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{array}{l} \text{— nothing to do} \rightarrow \\ \text{— } r_2 + \frac{1}{2} r_1 \rightarrow \\ \text{— } r_3 - \frac{1}{4} r_1 \rightarrow \end{array}$$

$$\begin{array}{cc}\begin{bmatrix} 4 & 0 & 12 \\ 0 & 6 & +3 \\ 0 & 2 & 2 \end{bmatrix} & \begin{array}{c} \\ r_2 \\ r_3 \end{array}\end{array} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{array}{l} \text{— nothing} \rightarrow \\ \text{— nothing} \rightarrow \\ \rightarrow r_3 - \frac{1}{3} r_2 \rightarrow \end{array}$$

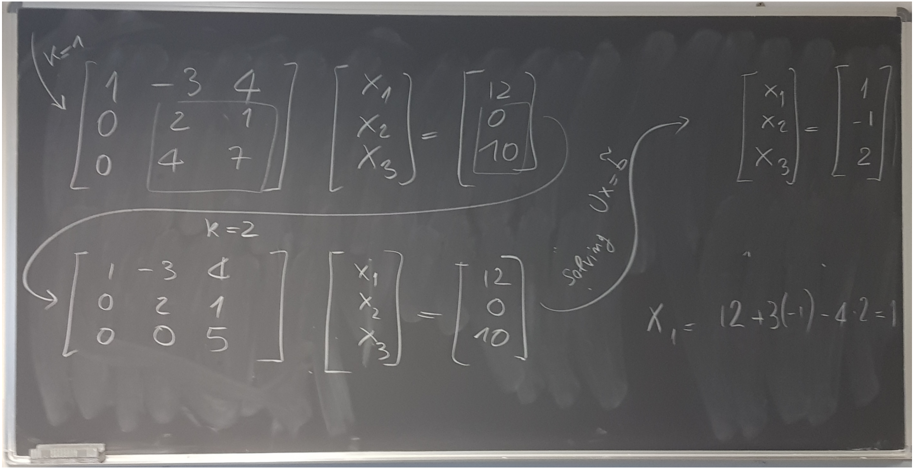
$$\begin{bmatrix} 4 & 0 & 12 \\ 0 & 6 & +3 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \simeq \begin{cases} 4x_1 + 12x_3 = 0 \\ 6x_2 + 3x_3 = 0 \\ x_3 = 1 \end{cases}$$

therefore $x_3 = 1$, $x_2 = -\frac{1}{2}$, $x_1 = -3$

EXERCISE 1: solution 2nd system



EXERCISE 1: solution 3rd system



EXERCISE 2: Matlab implementation

- Write a function that, given as input an upper triangular matrix U and a vector b, solves the system $Ux = b$ using the backsubstitution method and test it on the system

$$\begin{bmatrix} 3 & -1 & 2 \\ 0 & 1 & -5 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ -4 \\ 2 \end{bmatrix}$$

- Write a function that, given as input a matrix A and a vector b, solves the system $Ax = b$ using Gaussian elimination (without pivoting and with pivoting) and test it on the two systems:

$$\begin{bmatrix} 4 & 0 & 12 \\ -2 & 6 & -3 \\ 1 & 2 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}; \qquad \begin{bmatrix} 2 & 2 & 0 \\ 1 & 1 & -1 \\ 3 & -2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ 1 \\ 5 \end{bmatrix}$$

- Write a function that, given as input a matrix A, returns the L and U factors of the LU factorization without pivoting.

EXERCISE 3: solving linear systems and more...

- Using the LU factorization function above, write a function that returns the inverse of an input matrix.
- Modify the above function to compute the determinant of an input matrix A, and test it on the matrices in the previous slide.
- Solve a system $Ax = f$ with user-made functions above, where f arbitrary chosen and

$$A = \begin{bmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & -1 & 2 & -1 \\ 0 & 0 & \dots & -1 & 2 \end{bmatrix}$$

  In particular, compute x with either the GEM/LU methods implemented above and by explicitly computing $x = A^{-1}f$ (i.e., computing A^{-1} with the function above). Using the Matlab commands **tic** and **toc**, measure the time required by the two aproaches to compute the solution. The matrix size should be chosen large enough so that the time difference is relevant.

EXERCISE 4: sparse matrices (optional)

When the vast majority of a matrix A entries are zero, it is convenient to store only the nonzero values (and their position) in the memory. The Matlab function **sparse** can convert a non-sparse (dense) matrix into a sparse one.

- Let $Au = f$ be the matrix in the previous slide. Using the Matlab command **whos**, compare the memory usage when A is stored as dense and as sparse, for a large enough matrix size. Compare also the time spent to solve the system (use Matlab LU factorisation, and Matlab solver to invert the trinagular systems).
- Consider a similar system $Bu = f$, where

$$B = \begin{bmatrix} 2 & -1 & -1 & \dots & -1 \\ -1 & 2 & 0 & \dots & 0 \\ -1 & 0 & \ddots & & \vdots \\ \vdots & & & 2 & 0 \\ -1 & 0 & \dots & 0 & 2 \end{bmatrix}$$

  Note that A and B have the same sparisity, i.e. the same number of nonzero entries. Compare again the memory and solution time required when B is stored as sparse or as dense. Do you observe any difference with the previous case? If yes, why? You might want to compare the sparsity pattern (Matlab command **spy**) of the L U factors.