POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

# Linear System Of Equations. Part 1

## Calcoli di Processo dell' Ingegneria Chimica


**Timoteo Dinelli, Marco Mehl**

18th of October 2024.

Department of Chemistry, Materials and Chemical Enginering, G. Natta. Politecnico di Milano.
email: timoteo.dinelli@polimi.it
email: marco.mehl@polimi.it

# Linear System Of Equations

In mathematics, and particularly in linear algebra, a system of linear equations, also called a linear system, is a system composed of several linear equations that must all be verified simultaneously. A solution of the system is a vector whose elements are the solutions of the equations that make up the system, that is, such that when substituted for the unknowns make the equations identities.

From the classical representation to the matricial form:

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n}x_n = b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \ldots + a_{2,n}x_n = b_2 \\ \vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + \ldots + a_{n,n}x_n = b_n \end{cases}$$

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \ldots & a_{1,n} \\ a_{2,1} & a_{2,2} & \ldots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \ldots & a_{n,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

$$\underline{\underline{A}}\,\underline{x} = \underline{b} \quad \longrightarrow \quad \underline{x} = \underline{\underline{A}}^{-1}\,\underline{b}$$

$$\underline{x} = \underline{\underline{A}}^{-1}\,\underline{b}$$

This solution is impracticable (since requires the exact inversion of $\underline{\underline{A}}$)! So two types of method have been invented, the so called **DIRECT** methods and the **ITERATIVE** methods.
Now let's consider a $3 \times 3$ system of equations like the one below:

$$\begin{cases} 3x + 89y + 66z = 87 \\ 65y + 9z = 7 \\ 46z = 3 \end{cases} \qquad \begin{bmatrix} 3 & 89 & 66 \\ 0 & 65 & 9 \\ 0 & 0 & 46 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 87 \\ 7 \\ 3 \end{bmatrix}$$

So the goal is to get $\underline{\underline{A}}$ and after some operations get its upper triangular form.

## Gauss elimination

Given a linear system of equations in the form $\mathbf{Ax} = \mathbf{b}$, it is convenient to introduce the augmented matrix $\mathbf{A}^* = [\mathbf{A} \mid \mathbf{b}]$

$$[\mathbf{A}|\mathbf{b}] = \left[\begin{array}{ccc|c} a_{1,1}^{(0)} & \dots & a_{1,n}^{(0)} & b_1^{(0)} \\ \vdots & \ddots & \vdots & \vdots \\ a_{n,1}^{(0)} & \dots & a_{n,n}^{(0)} & b_n^{(0)} \end{array}\right]$$

After $n-1$ steps we obtain the following system:

$$[\mathbf{A}|\mathbf{b}] = \left[\begin{array}{cccc|c} a_{1,1}^{(0)} & \dots & \dots & a_{1,n}^{(0)} & b_1^{(0)} \\ 0 & a_{2,2}^{(1)} & \dots & a_{2,n}^{(1)} & b_2^{(1)} \\ \vdots & \ddots & \dots & \dots & \vdots \\ 0 & \dots & 0 & a_{n,n}^{(n-1)} & b_n^{(n-1)} \end{array}\right]$$

# Key IDEA!

Let's use triangular matrices.

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n-1} & a_{1,n} \\ 0 & a_{2,2} & \dots & a_{2,n-1} & a_{2,n} \\ 0 & 0 & \dots & a_{3,n-1} & a_{3,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & a_{n,n} \end{bmatrix} \quad \mathbf{x} = \mathbf{b}^*$$

# LU factorization/decomposition

$$\underline{\underline{A}}\,\underline{x} = \underline{b}$$

$$\underline{\underline{A}}\,\underline{x} \;=\; \underline{\underline{LU}}\,\underline{x} \;=\; \underline{b}$$

$$\underline{\underline{U}}\,\underline{x} = \underline{y} \longrightarrow \underline{\underline{L}} \quad \underline{y} = \underline{b}$$

Solution:

$$\underline{\underline{L}}\,\underline{y} = \underline{b} \longrightarrow \underline{\underline{U}}\,\underline{x} = \underline{y}$$

$$\mathsf{L} = \begin{bmatrix} 1 & 0 & 0 \\ c_{2,1} & 1 & 0 \\ c_{3,1} & c_{3,2} & 1 \end{bmatrix} \qquad \mathsf{U} = \begin{bmatrix} d_{1,1} & d_{1,2} & d_{1,3} \\ 0 & d_{2,2} & d_{2,3} \\ 0 & 0 & d_{3,3} \end{bmatrix}$$

# WHY LU?

- Once $\underline{\underline{\mathbf{A}}}$ has been decomposed, it is possible to solve multiple problems having different $\mathbf{b}$ values (useful for other numerical methods).
- Once $\underline{\underline{\mathbf{A}}}$ has been decomposed, it is possible to solve: $A^T x = c$ without doing a factorization for $\underline{\underline{\mathbf{A}}}^T$.
- The matrices $\underline{\underline{\mathbf{L}}}$ and $\underline{\underline{\mathbf{U}}}$ can be obtained using different methods than the Gauss elimination. Pay attention to the built-in function lu of MATLAB (read the documentation).
- If $\underline{\underline{\mathbf{A}}}$ is modified, techniques that update $\underline{\underline{\mathbf{L}}}$ e $\underline{\underline{\mathbf{U}}}$ accordingly to generate the new matrix $\underline{\underline{\mathbf{A}}}^*$.

# Exercises

# Triangularize U

Write a function that takes the matrix A of size ($n \times n$) and b ($n \times 1$) as the input and outputs a triangular upper matrix using the the Gauss elimination method (without pivoting/balancing).
The function: function[A,b]=triangularize_U(A,b) Will generate the augmented matrix $\underline{\underline{\mathbf{W}}}$ and, using two nested *for* loops, the first with index *i* ranging from *1* to *n-1*, the second (index *j*) from *i+1* to *n*, that will operate the linear combination that eliminates the element *(j,i)* from $\underline{\underline{\mathbf{W}}}$. A second function solve_upper_triangular_system(A,b) will take the output of the first function and will solve the system using the algorithms discussed in class (solution either by row or by column). Finally, write a function my_linear_solver(A,b) that calls the first two and returns the solution of a linear system. And compare your solution with the one obtained using the built-in function linsolve or mldivide.

To test out if your algorithm works properly use the following systems of equations:

$$\begin{cases} x + 2y - z + 2t = 3 \\ x + 2z + t = 1 \\ 2x + y - 2t = 1 \\ -z + t = 2 \end{cases}$$

$$\begin{cases} x + 45y - t = 6 \\ x - 3t = 12 \\ x + y + z + 6 = 0 \\ x - y + z + t = 12 \end{cases}$$

# LU

Let's modify the previously implemented code and implemented a function to perform the LU decomposition of a given square matrix. Then combine this function inside our linear solver and compare the results.

Thank you for the attention!