# 05 - Sistemi lineari

## Sistema Lineare

$$Ax = b$$

dove

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}$$

e quindi:

- $a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1$
- ...
- $a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n = b_n$

Esiste un unica soluzione $\iff det(A) \neq 0$.
Se è verificata questa condizione il problema è **ben posto**.

Il sistema lineare risulta un problema **ben condizionato** quando il numero di condizionamento della matrice $K(A) = ||A|| \times ||A^{-1}||$ non risulta molto elevato.

---

## Metodi diretti

Assegnato il sistema lineare $Ax = b$ con le solite condizioni:

- $A$ quadrata $n \times n$
- $det(A) \neq 0$

Si utilizzano un numero finito di passi per trasformarlo in un sistema lineare equivalente (ossia avente la stessa soluzione) ma di più facile risoluzione.

Vedremo che il sistema lineare equivalente a cui ci riconduciamo sarà un **sistema triangolare**.

---

### Metodi di sostituzioni per la risoluzione di sistemi lineari con matrice dei coefficenti in forma triangolare

#### Metodo di sostituzione in avanti:

Necessaria matrice **triangolare inferiore**:

$$\begin{pmatrix} a_{11} & 0 & ... & ... & 0 \\ a_{21} & a_{22} & 0 & ... & 0 \\ ... \\ a_{n1} & a_{n2} & a_{n3} & ... & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ ... \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ ... \\ b_n \end{pmatrix}$$

Abbiamo quindi:

- $a_{11}x_1 = b_1$
- $a_{n1}x_1 + a_{n2}x_2 = b_2$
- ...
- $a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n = b_n$

Da cui segue che una generica $i - esima$ riga si può ricavare come segue:

$$(\sum_{j=1}^{i-1} a_{ij} \times x_j) + a_{ii}x_i = b_i$$

da cui:

$$x_i = (\frac{b_i - (\sum_{j=1}^{i-1} a_{ij} \times x_j)}{a_{ii}}) \ \forall i = 2, ..., n$$

Quindi in pseudocodice sarebbe:

```
x = b[1] / a[1][1]
for i in range(2, n + 1):
    x = (b[i] - sommatoria()) / a[i][i]
```

Osservazioni: gli elementi diagonali sono tutti diversi da zero se richiedo che la matrice del sistema linerare sia **non singolare** ($det(A) \neq 0$).

**Costo computazionale**

All' $i$-esimo passo abbiamo:

- $i - 1$ moltiplicazioni
- $1$ divisione

Ovvero $i$ operazioni ad ogni passo.
Quindi il costo sarà dato da:

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} = O(\frac{n^2}{2})$$

---

### Metodo di sostituzione all'indietro:

Necessaria matrice **triangolare superiore**:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & a_{22} & a_{23} & \dots & a_{2n} \\ \dots \\ 0 & 0 & 0 & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{pmatrix}$$

Abbiamo quindi:

- $a_{11}x_1 + ... + a_{1n}x_n = b_1$
- ...
- $a_{nn}x_n = b_n$

Da cui segue che una generica $i - esima$ riga si può ricavare come segue:

$$(\sum_{j=i+1}^{n} a_{ij} \times x_j) + a_{ii}x_i = b_i$$

da cui:

$$x_i = (\frac{b_i - (\sum_{j=i+1}^{n} a_{ij} \times x_j)}{a_{ii}}) \ \forall i = 2, ..., n$$

Quindi in pseudocodice sarebbe:

```
x = b[n] / a[n][n]
for i in range(n - 1, 1, -1):
    x = (b[i] - sommatoria()) / a[i][i]
```

#### Costo computazionale

All' $i$-esimo passo abbiamo:

- $n - i$ moltiplicazioni
- $1$ divisione

Ovvero $n - i + 1$ operazioni ad ogni passo.
Quindi il costo sarà dato da:

$$\sum_{i=1}^{n} n - i + 1 = \sum_{j=1}^{n} j = \frac{n(n+1)}{2} = O(\frac{n^2}{2})$$

---

### Strategie per sostituzione del sistema ad un sistema equivalente ma triangolare superiore

Esistono 3 metodi.
Caratteristica comune ai 3 metodi diretti che vedremo per ricondurre il sistema (*) aò sistema (**) è la fattorizzazione di $A$ della forma:

$$A = BC$$

con $C$ **triangolare superiore** e $B$ o **triangolare inferiore o ortogonale**.

Utilizzando la fattorizzazione:

$$A = BC$$

possiamo rischivere il sistema (*) :

$$Ax = b => BCx = b$$

Denotiamo successivamente con $y$:

$$y = Cx$$

Abbiamo che:

1. $By = b$ => determino $y$
2. $Cx = y$ => determino $x$

**Osservazione 1**: Nel caso in cui $B$ sia triangolare inferiore, si trova $y$ risolvendo 1) con il metodo delle sostituzioni in avanti

**Osservazione 2**: Nel caso in cui $B$ sia ortogonale si trova $y$ come:

$$y = bB^T$$

Una volta ottenuto $y$ posso risolvere $x$ tramite sostituzioni all'indietro poichè $C$ è triangolare superiore.

---

## Metodo di Gauss (eliminazione Gaussiana)

$$A = LU$$

con:

- $L$ triangolare inferiore o diagonale 1
- $U$ triangolare superiore

**Condizioni**:
Data $A \in \mathbb{R}^{n \times n}$, le condizioni che dobbiamo richiedere per garantire l'esistenza di $L$ ed $U$ sono:

$$det(A(1:k, 1:k)) \neq 0, \forall k = 1, ..., n$$

ovvero tutte le **sottomatrici principali di testa** ottenute ritagliando le prime $k$ righe e $k$ colonne abbiano determinante diverso da 0.

---

**Teorema**

Se i determinanti delle $n-1$ sottomatrici principali di testa di $A$ sono diversi da zero, allora esiste ed è unica la fattorizzazione $LU$ della matrice $A$ dove $L$ è una matrice triangolare inferiore con elementi diagonali uguali ad 1 e $U$ è una matrice triangolare superiore.

---

### Procedimento

Input: $A$ $(n \times n)$

$A^{(1)} := A$
for $k = 1 : n-1$:

1. costruire i moltiplicatori del passo $k$

$$m_{rk}^{(k)} = (\frac{a_{rk}^{(k)}}{a_{kk}^{(r)}}), \forall r = k+1, ..., n$$

2. costruire la matrice elementare di Gauss del passo $k$

$$M^{(k)} = \begin{pmatrix} 1 & 0 & ... & ... & 0 \\ -m_{k+1,k}^{(k)} & 1 & 0 & ... & 0 \\ ... \\ -m_{n,k}^{(k)} & ... & ... & ... & 0 \end{pmatrix} = I - m^{(k)} e_k^T$$

con

$$m^{(k)} = \begin{pmatrix} 0 \\ ... \\ 0 \\ m_{k+1,k}^{(k)} \\ ... \\ m_{n,k}^{(k)} \end{pmatrix}$$

ed

$$e_k^T = \begin{pmatrix} 0 \\ ... \\ 0 \\ 1 \\ 0 \\ ... \\ 0 \end{pmatrix}$$

3. aggiornare la matrice $A^{(k)}$ moltiplicandola a sinistra per $M^{(k)}$

$$A^{(k+1)} = M^{(k)} \times A^{(k)}$$

Output:

$U := A^{(n)}$ triangolare superiore

$$L = \begin{pmatrix} 1 & 0 & ... & ... & 0 \\ -m_{2,1}^{(1)} & 1 & 0 & ... & 0 \\ -m_{3,1}^{(1)} & -m_{3,2}^{(2)} & 0 & ... & 0 \\ ... \\ -m_{n,1}^{(1)} & -m_{n,2}^{(2)} & ... & -m_{n,k}^{(k)} & 0 \end{pmatrix}$$

dove $L$ è triangolare inferiore.

Abbiamo quindi che:

$$U = A^{(n)} = M^{(n-1)} \times M^{(n-2)} \times ... \times M^{(2} \times M^{(1)} \times A$$

Osserviamo che tutte le $M^{(k)}$ sono matrici invertibili e:

$$(M^{(k)})^{-1} = I + m^{(k)} e_k^T$$

e quindi:

$$U = (M^{(n-1)} \times ... \times M^{(1)})A$$

da cui segue (passando per gli inversi):

$$A = ((M^{(1)})^{-1} \times ... \times (M^{(n-1)})^{-1})U = LU$$

dove di consguenza:

$$L = ((M^{(1)})^{-1} \times ... \times (M^{(n-1)})^{-1})$$

---

### Psudocodice finale

Input : $A$ $n$ x $n$

> $A^{(1)} := A$
> for $k = 1 : n-1$:
>
> > $m_{rk}^{(k)} = (\frac{a_{rk}^{(k)}}{a_{kk}^{(r)}}), \forall r = k+1, ..., n$
> > # per la matrice $A^{(k+1)}$ vanno calcolati soltagli gli elementi:
> > $a_{i,j}^{(k+1)} = a_{i,j}^{(k)} - m_{i,k}^{(k)} a_{k,j}^{(k)}$
> > # con $i = k+1, ..., n$ e $j = k+1, ..., n$

Output: $L, U$

---

#### Costo computazionale

$$\sum_{k=1}^{n-1} [(n-k) + (n-k)^2] \simeq \sum_{k=1}^{n-1} (n-k)^2 = \sum_{j=1}^{n-1} j^2 \simeq \frac{n^3}{3}$$

---

### Problemi dell eliminazione di Gauss

Per non avere elevati errori algoritmici nel calcolo della soluzione occorre evitare che i pivot siano piccoli (ovvero i moltiplicatori grandi).

### Strategia pivoting parziale

al passo $k$-esimo scambio le righe in modo da avere in posizione pivotale il valore più grande in modo da non avere i moltiplicatori grandi (vanno applicate anche le trasformazioni al vettore colonna $b$)
Lo scambio di righe viene fatto tramite la moltiplicazioni di **matrici di permutazione** denominate $p^{(k)}$

Abbiamo quindi che al passo $k$-esimo:

$$A^{(k+1)} = M^{(k)} p^{(k)} A^{(k)}$$

e

$$b^{(k+1)} = M^{(k)} p^{(k)} b^{(k)}$$

infine:

$$U := A^{(n)} = M^{(n-1)} p^{(n-1)} A^{(n-1)}$$

**Ma come trovare la L?**

$$U = A^{(n)} = (M^{(n-1)} p^{(n-1)} A^{(n-1)})(M^{(n-2)} p^{(n-2)} A^{(n-2)})...(M^{(1)} p^{(1)} A)$$

ristrutturando algebricamente:

$$U = A^{(n)} = M^{(n-1)} M'^{(n-2)} \; ... \; M'^{(1)} p^{(n-1)} p^{(n-2)} ... \; p^{(1)} A$$

dove le varie matrici $M'^{(i)}$ sono date da:

$$M^{(i-1)} := S^{(i)} M^{(i-1)} (S^{(i)})^{-1}$$

dove:

- $S^{(n-1)} = p^{(n-1)}$
- $S^{(n-2)} = p^{(n-1)} p^{(n-2)}$
- $S^{(j)} = p^{(n-1)} p^{(n-2)} p^{(n-3)} ... p^{(j)}$

quindi:

$$M'^{(i-1)} = S^{(i)} (I - m^{(i-1)} e_{i-1}^T)(S^{(i)})^{-1}$$

$$= [S^{(i)} (S^{(i)})^{-1}] - [(S^{(i)} m^{i-1})(e_{i-1}^T (S^{(i)})^{-1})]$$

$$= I - m'^{(i-1)} e_{i-1}^T$$

dove $m'^{(i)}$ contiene i moltiplicatori del vettore originale opportunamente permutati:

$$m'^{(i-1)} = \begin{pmatrix} 0 \\ ... \\ 0 \\ * \\ ... \\ * \end{pmatrix}$$

i primi $i-1$ sono $0$.

Concludendo:

$$U = (M^{(n-1)}M'^{(n-2)}...M'^{(1)})(p^{(n-1)}p^{(n-2)}...p^{(1)})A$$
$$(M^{(n-1)}M'^{(n-2)}...M'^{(1)})^{-1}U = (p^{(n-1)}p^{(n-2)}...p^{(1)})A$$

dove:

$$(M^{(n-1)}M'^{(n-2)}...M'^{(1)})^{-1} = L$$

e

$$(p^{(n-1)}p^{(n-2)}...p^{(1)}) = p$$

---

### Pseudocodice (Pivoting Parziale)

Input: $A$

> $A^{(1)} = A$
> FOR $k = 1 : n-1$:
>
> > a partire dalla matrice $A^{(k)}$ determino $r$ t.c:
> > $|a_{r,k}^{(k)}| = max_{i=k,...n}|a_{i,k}^{(k)}|$
> >
> > costruisco la matrice di permutazione $p^{(k)}$ tale che $p^{(k)}A^{(k)}$ ha le righe $r$ e $k$ scambiate
> >
> > costruisco la matrice elementare di Gauss $M^{(k)}$ t.c:
> > $A^{(k+1)} = M^{(k)}p^{(k)}A^{(k)}$
> > soddisfa la proprietà $a_{i,k}^{(k+1)} = 0, \forall i = k+1, ..., n$

Output:

- $U := A^{(n)}$
- $L := (M^{(n-1)})^{-1}(M'^{(n-2)})^{-1}...(M'^{(1)})^{-1}$
- $p = p^{(n-1)}p^{(n-2)}...p^{(1)}$

---

### Costo computazionale

Si aggiunge solamente il costo dei confronti che ha un costo di $O(\frac{n^2}{2})$ ma poichè il costo di Gauss è di per se $O(\frac{n^3}{3})$ l'utilizzo del pivoting parziale non va ad influire sul costo totale poichè di ordine inferiore.

---

### Vantaggi

- Nella matrice $L$ tutti i valori sono la diagonali sono più piccoli di 1 poichè il pivot è sempre il numero più grande.
- $max_{i,j}|U_{i,j}| \leq 2^{n-1}max_{i,j}|A_{i,j}|$

Queste due proprietà garantiscono la stabilità.

---

## Metodo di Householder e fattorizzazione QR

Sia dato un sistema lineare:

$$Ax = b$$

**Teorema**:
$\exists\, Q$ matrice $n \times n$, ortogonale ($QQ^T = Q^TQ = I$) e $R$ matrice $n \times n$, triangolare superiore, non singolare ($rg(R) = n$) tali che:

$$A = QR$$

La trasformazione si basa su delle **matrici di trasformazioni** (chiamate **matrici di Householder**) tali che:

$$H_{n-1}H_{n-2}...H_1A = R$$

e quindi:

$$A = (H_{n-1}H_{n-2}...H_1)^{-1}R$$

da cui segue:

$$(H_{n-1}H_{n-2}...H_1)^{-1} = Q$$

$$(H_{n-1})^{-1}(H_{n-2})^{(-1)}...(H_1)^{-1} = Q$$

$$(H_1)^T(H_2)^T...(H_{n-1})^T = Q$$

poichè le $H$ sono simmetriche ed ortogonali:

$$(H_1)(H_2)...(H_{n-1}) = Q$$

---

### Costo computazionale

$$O(\frac{2}{3}n^3)$$

---

### Procedimento

Sfruttando la fattorizzazione $QR$ la risoluzione del sistema lineare $Ax = b$ diventa:

$$QRx = b$$

1. $Qy = b =>$ si calcola $y = Q^T b$
2. $Rx = y =>$ è un sistema con matrice dei coefficenti triangolare superiore, non singolare, che si risolve con il metodo delle sostituzioni all'indietro

---

### Proprietà fattorizzazione $QR$

- é più stabile rispetto alla fattorizzazione $LU$ con pivoting parziale
- La fattorizzazione $QR$ non è unica
- La fattorizzazione $QR$ si può usare anche per matrici non quadrate, ad esempio quindi matrici rettangolari (sistemi sovradeterminati)

---

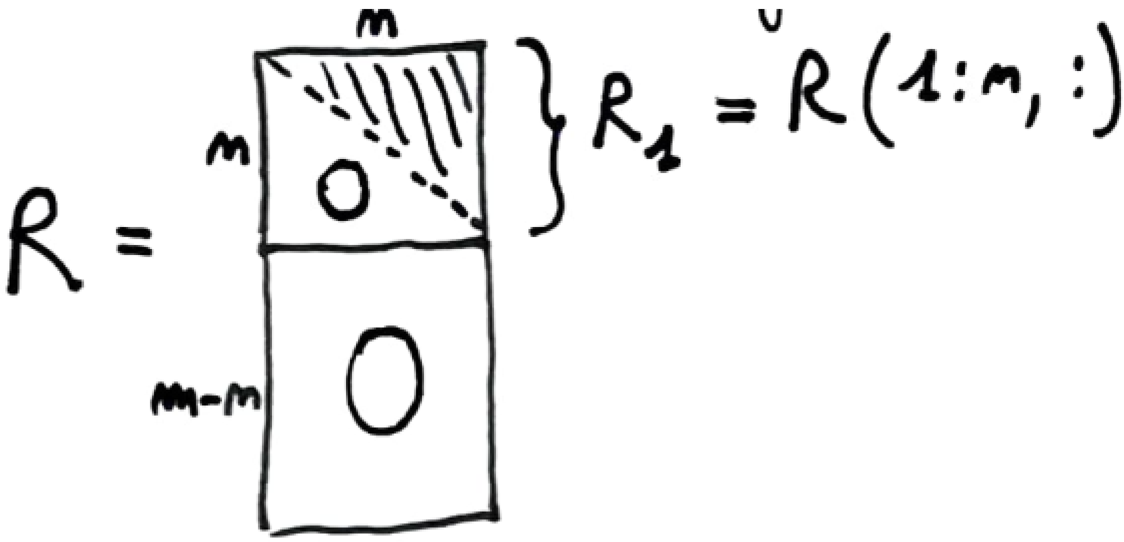
### Matrici rettangolari "alte"

numero righe > numero colonne (numero equazioni > numero incognite)

**Teorema**:

Sia $A \in \mathbb{R}^{m \times n}$ con $m > n$ e $rg(A) = n$.
Allora esistono:

- $Q \in \mathbb{R}^{m \times m}$ ortogonale
- $R \in \mathbb{R}^{m \times n}$ "trapezoidale superiore"




con $R_1 \in \mathbb{R}^{n \times n}$ triangolare superiore, non singolare, tali che:

$$A = QR$$

dove il **numero di trasformazioni di Householder** è:

$$r = min(m-1, n)$$

---

#### Costo computazionale

$$O(mn^2 - \frac{n^3}{3})$$