

# Sistemi lineari

Lucia Gastaldi

DICATAM - Sez. di Matematica,  
<http://lucia-gastaldi.unibs.it>



UNIVERSITÀ  
DEGLI STUDI  
DI BRESCIA

- 1 Risoluzione di sistemi lineari
  - Risoluzione di sistemi lineari in Matlab
  - Metodi di risoluzione
  - Fattorizzazione
  
- 2 Analisi degli errori
  - Norme di vettore e di matrici
  - Numero di condizionamento

$$Ax=b$$

Tre casi possibili:

- ▶ Sistemi quadrati,  $m = n$ .
- ▶ Sistemi sovradeterminati,  $m > n$ .
- ▶ Sistemi sottodeterminati,  $m < n$ .

# Come risolvere un sistema lineare con MATLAB

La risoluzione del sistema lineare si ottiene usando i simboli di divisione: **backslash** `\` e **slash** `/`.

$x = A \backslash b$  indica la soluzione di  $Ax = b$ ,  $x$  e  $b$  vettori colonna.

$x = b / A$  indica la soluzione di  $xA = b$ ,  $x$  e  $b$  vettori riga.

L'operatore **backslash** usa algoritmi differenti per trattare diversi tipi di matrici:

- ▶ Permutazioni di matrici triangolari.
- ▶ Matrici simmetriche e definite positive.
- ▶ Matrici quadrate, non singolari e piene.
- ▶ Matrici quadrate, non singolari e sparse.
- ▶ Sistemi rettangolari sovradeterminati.
- ▶ Sistemi rettangolari sottodeterminati.

# Risoluzione di sistemi triangolari

## Metodo di sostituzione in avanti

$L$  matrice triangolare inferiore.

$$x_1 = \frac{b_1}{l_{11}}$$
$$x_i = \frac{b_i - \sum_{j=1}^{i-1} l_{ij}x_j}{l_{ii}} \quad \text{per } i = 2, \dots, n$$

## Metodo di sostituzione all'indietro

$U$  matrice triangolare superiore.

$$x_n = \frac{b_n}{u_{nn}}$$
$$x_i = \frac{b_i - \sum_{j=i+1}^n u_{ij}x_j}{u_{ii}} \quad \text{per } i = n-1, \dots, 1$$

# Algoritmo di eliminazione di Gauss

```
for  $k = 1, \dots, n - 1$   
  for  $i = k + 1, \dots, n$   
     $m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$   
    for  $j = k + 1, \dots, n$   
       $a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}$   
    end  
     $b_i^{(k+1)} = b_i^{(k)} - m_{ik} b_k^{(k)}$   
  end  
end
```

# Fattorizzazione LU

## Teorema

Costruiamo le seguenti matrici:

$$L = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ m_{21} & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ m_{n1} & \cdots & m_{nn-1} & 1 \end{pmatrix}$$

$$U = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & a_{nn}^{(n)} \end{pmatrix}.$$

Se tutti i minori principali di  $A$  sono non nulli, si ha  $LU = A$ .

Il comando `[L,U]=miaLU(A)` fornisce la fattorizzazione LU associata al metodo di eliminazione di Gauss.

# Strategia di pivoting

Per evitare possibili divisioni per 0 e per rendere l'algoritmo di eliminazione (oppure l'algoritmo di fattorizzazione LU) **stabili** rispetto alla **propagazione degli errori di arrotondamento** si usa la **strategia di pivoting** che consiste nello scambio sistematico di righe opportune.

Il risultato della fattorizzazione LU è:

$$PA = LU$$

essendo  $P$  una **matrice di permutazione** che tiene conto degli scambi di righe avvenuti.



# Algoritmo di eliminazione di Gauss con pivoting

```

for  $k = 1, \dots, n - 1$ 
  cerco più piccolo  $p$  tale che  $|a_{pk}^{(k)}| = \max_{k \leq i \leq n} |a_{ik}^{(k)}|$ 
  scambio la riga  $k$  con la riga  $p$ 
  for  $i = k + 1, \dots, n$ 
    
$$m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$$

    for  $j = k + 1, \dots, n$ 
      
$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}$$

    end
    
$$b_i^{(k+1)} = b_i^{(k)} - m_{ik} b_k^{(k)}$$

  end
end
end

```

# Le funzioni MATLAB per la fattorizzazione

---

Funzione	Significato
----------	-------------

---

lu	Fattorizzazione $PA = LU$ .
----	-----------------------------

chol	Fattorizzazione di Cholesky $A = R^T R$ con $R$ triang. sup.
------	--

qr	Fattorizzazione $A = QR$ .
----	----------------------------

schur	Decomposizione di Schur $A = UTU^H$ .
-------	---------------------------------------

---

## Uso della function `lu`

Data la matrice  $A \in \mathbb{R}^{n \times n}$ , la function `lu` fornisce il risultato della fattorizzazione nelle seguenti forme:

- ▶ `[L,U,P]=lu(A)`:  
fornisce le matrici L, U e P in modo che  $L*U=P*A$ .
- ▶ `[L1,U]=lu(A)`:  
fornisce le matrici L1 e U in modo che  $L1*U=A$ . In questo caso la matrice L1 si ottiene dalla permutazione delle righe di L mediante P ossia  $L1 = P^{-1}L$

Verificare il comportamento di `lu` sulla matrice:

$$A = \begin{pmatrix} 1 & 2 & -1 \\ 3 & 2 & 1 \\ 2 & -1 & 0 \end{pmatrix}$$

e confrontare quanto ottenuto con la function `miaLU`.

# Matrici di permutazione

## Definizione

Una **matrice di permutazione**  $P$  è ottenuta dalla matrice identità scambiando le righe e le colonne. Su ciascuna riga e colonna si trova uno ed uno solo 1 mentre tutti gli altri elementi sono nulli.

$$P = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Il prodotto  $P * A$  permuta le righe della matrice  $A$ .

Il prodotto  $A * P$  permuta le colonne della matrice  $A$ .

L'effetto della moltiplicazione per  $P$  può essere anche ottenuto usando il vettore  $p = [4 \ 1 \ 3 \ 2]$ .

I comandi  $P * A$  e  $A(p, :)$  hanno lo stesso effetto.

La matrice inversa è data da:  $P^{-1} = P^T$

# Propagazione degli errori

## Esercizio 1

Consideriamo al variare di  $a \in \mathbb{R}$  la matrice  $A$  e il vettore  $b$  dati da:

$$A = \begin{pmatrix} 1 & 1 & 3 \\ 2 & 2+a & 20 \\ 3 & 6 & 4 \end{pmatrix} \quad b = \begin{pmatrix} 3 \\ 20-a \\ 1 \end{pmatrix}$$

Dati i seguenti valori di  $a$ :  $a = 1$ ,  $a = 0$ ,  $a = 0.5 \cdot 10^{-15}$ ,

- ▶ calcolare la fattorizzazione LU di  $A$  mediante la function `miaLU`;
- ▶ calcolare la differenza  $A - LU$ ;
- ▶ usare la fattorizzazione ottenuta per risolvere il sistema lineare  $Ax = b$ , la cui soluzione esatta è  $x = (1, -1, 1)^T$ ;
- ▶ ripetere la procedura usando le apposite function di Matlab per la fattorizzazione LU e la risoluzione dei due sistemi relativi alle matrici triangolari.

# Riempimento delle matrici triangolari ottenute con LU

## Esercizio 2

Si consideri la matrice  $A \in \mathbb{R}^{25 \times 25}$  che ha i seguenti elementi:

$$a_{ii} = 1, \quad \text{per } i = 1, \dots, 25$$

$$a_{1j} = 1, \quad \text{per } j = 2, \dots, 25$$

$$a_{i1} = 1, \quad \text{per } i = 2, \dots, 25$$

Costruire la matrice usando prima il comando `speye` e poi correggendo la prima riga e la prima colonna.

Usare il comando `lu` per ottenere le matrici  $L, U, P$  che danno la fattorizzazione della matrice.

Usando i comandi `subplot` e `spy` visualizzare la distribuzione degli elementi non nulli delle matrici  $A, L, U, P$  in una stessa figura.

# Norma di vettore

Sia  $\mathbf{x}$  un vettore di dimensione  $n$ , per  $1 \leq p \leq \infty$ , il comando `norm(x,p)` fornisce il valore della norma:

$$\|\mathbf{x}\|_p = \left( \sum_{i=1}^n |x_i|^p \right)^{1/p}$$

Le norme più usate sono:

$$\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i| \quad \text{norm(x,1)}$$

$$\|\mathbf{x}\|_2 = \left( \sum_{i=1}^n |x_i|^2 \right)^{1/2} \quad \text{norm(x,2)=norm(x)}$$

$$\|\mathbf{x}\|_\infty = \max_{1 \leq i \leq n} |x_i| \quad \text{norm(x,Inf)}$$

# Norma di matrice

La moltiplicazione  $Ax$  può produrre un vettore con una norma completamente diversa da quella di  $x$ . La norma della matrice  $A$  si definisce come segue

$$\|A\| = M = \max_{x \neq 0} \frac{\|Ax\|}{\|x\|}.$$

Valgono le seguenti proprietà:

$$\|\mathbb{I}\| = 1 \text{ per } \mathbb{I} \text{ matrice identità}$$

$$\|Ax\| \leq \|A\| \|x\|$$

$$\|AB\| \leq \|A\| \|B\|$$

$\text{norm}(A, p)$  fornisce la norma di matrice per  $p = 1, 2, \infty$ :

$$\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^n |a_{ij}| \quad \|A\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}|$$

$$\|A\|_2 = \sqrt{\rho(A^T A)}$$

essendo  $\rho(A^T A) = \max_i \sigma_i$ , e  $\sigma_i$  autovalore di  $A^T A$ .



# Numero di condizionamento

## Definizione

$$K(A) = \|A^{-1}\| \|A\|$$

si dice **numero di condizionamento della matrice A**.

## Teorema

Si consideri il sistema lineare  $Ax = b$ . Siano  $\delta A$  e  $\delta b$  perturbazioni di  $A$  e di  $b$  rispettivamente e sia  $x + \delta x$  la soluzione del sistema lineare:

$$(A + \delta A)(x + \delta x) = b + \delta b.$$

Allora vale la seguente maggiorazione:

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{K(A)}{1 - K(A)\|\delta A\|/\|A\|} \left( \frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|} \right).$$

# Numero di condizionamento

$$K(A) = \|A^{-1}\| \|A\|$$

La norma  $\|A\|$  indica il rapporto massimo tra la norma del vettore  $Ax$  e quella di  $x$ .

Osserviamo che, ponendo  $Ay = x$  e  $y = A^{-1}x$ , si ha

$$\|A^{-1}\| = \max_{x \neq 0} \frac{\|A^{-1}x\|}{\|x\|} = \max_{y \neq 0} \frac{\|y\|}{\|Ay\|} = \frac{1}{\min_{y \neq 0} \frac{\|Ay\|}{\|y\|}} = \frac{1}{m}$$

Il numero  $m$  indica il rapporto minimo tra la norma di  $Ax$  e quella di  $x$ . Di conseguenza

$$K(A) = \frac{\max_{x \neq 0} \frac{\|Ax\|}{\|x\|}}{\min_{x \neq 0} \frac{\|Ax\|}{\|x\|}}.$$

## Il condizionamento in Matlab

- ▶ `cond(A)` o `cond(A,2)` calcola  $K_2(A)$  (con la norma 2). Usa `svd(A)`. Computazionalmente costoso, adatto a matrici piccole.
- ▶ `cond(A,1)` calcola  $K_1(A)$  (con la norma 1). Usa `inv(A)`. Meno lavoro che per `cond(A,2)`.
- ▶ `cond(A,Inf)` calcola  $K_\infty(A)$  (con la norma  $\infty$ ). Usa `inv(A)`. È lo stesso di `cond(A',1)`.
- ▶ `condest(A)` stima  $K_1(A)$ . Usa `lu(A)` e un algoritmo recente di Higham e Tisseur. Adatto specialmente per matrici sparse e di grandi dimensioni.
- ▶ `rcond(A)` stima  $1/K_1(A)$ . Usa `lu(A)` e un algoritmo più vecchio sviluppato in LINPACK e LAPACK.

# Esercizi

## Esercizio 3

Dati

$$A = \begin{pmatrix} 1.2969 & 0.8648 \\ 0.2161 & 0.1441 \end{pmatrix}, \quad b = \begin{pmatrix} 0.8642 \\ 0.1440 \end{pmatrix},$$

calcolare la soluzione esatta del sistema  $Ax = b$ .

Si considerino le seguenti perturbazioni  $r_1 = [-10^{-8}, 10^{-8}]^T$  e  $r_2 = [10^{-8}, 10^{-8}]^T$  al termine noto.

- ▶ Per ciascuna perturbazione calcolare la soluzione del sistema  $A\hat{x}_i = b + r_i$ ,  $i = 1, 2$  mediante il comando `x=A\b`.
- ▶ Calcolare l'errore relativo commesso, e confrontarlo con la perturbazione relativa del termine noto.
- ▶ Calcolare il numero di condizionamento di  $A$ .
- ▶ Verificare che il risultato ottenuto soddisfa la stima teorica.

# Matrice mal condizionata

## Esercizio 4

Si consideri il sistema lineare  $Ax = b$  con  $A \in \mathbb{R}^{n \times n}$  **matrice di Hilbert** di elementi

$$a_{ij} = \frac{1}{i+j-1}, \quad i = 1, \dots, n,$$

e  $b \in \mathbb{R}^n$  tale che la soluzione del sistema sia  $x = (1, \dots, 1)^\top$ .

- ▶ Risolvere il sistema al variare di  $n$ . Sia  $\hat{x}$  la soluzione calcolata.
- ▶ Calcolare il numero di condizionamento della matrice  $K$ .
- ▶ Riportare in uno stesso grafico in scala semilogaritmica le seguenti quantità al variare di  $n$ :
  - ▶ il numero di condizionamento;
  - ▶ l'errore relativo  $E = \|x - \hat{x}\|/\|x\|$ ;
  - ▶ il **residuo**  $\|b - A\hat{x}\|/\|b\|$ ;
  - ▶ la stima dell'errore  $K\|b - A\hat{x}\|/\|b\|$ .

Per calcolare le norme usare il comando **norm**.

## Comandi utili per l'esercizio

- ▶ `A=hilb(n)` fornisce la matrice di Hilbert di dimensione  $n \times n$ .
- ▶ `x=ones(n,1)` genera il vettore colonna di dimensione  $n$  che ha tutte le componenti uguali a 1.
- ▶ `b=A*x` calcola il termine noto.
- ▶ `xapp=A\b` risolve il sistema lineare.
- ▶ `err=norm(x-xapp)/norm(x)` calcola l'errore relativo.
- ▶ `r=b-A*xapp` calcola il residuo.
- ▶ `res=norm(r)/norm(b)` calcola la norma del residuo rapportata alla norma del termine noto.
- ▶ `K=cond(A)` calcola il numero di condizionamento di  $A$ .
- ▶ Il comando `semilogy` produce un grafico con la scala logaritmica in base 10 sull'asse delle  $y$ .

**Suggerimento** Per potere fare il grafico in scala semilogaritmica si devono creare i vettori `err`, `res`, `K`.