SISTEMI LINEARI

Letizia SCUDERI

Dipartimento di Scienze Matematiche, Politecnico di Torino

letizia.scuderi@polito.it

A.A. 2017/2018

Sistemi lineari

La risoluzione di un sistema lineare è un problema che si presenta in moltissime applicazioni,

- sia esplicitamente come modello (formulazione matematica) di un fenomeno fisico;
- sia come passo intermedio o finale nella risoluzione numerica del modello in questione, rappresentato, per esempio, da equazioni differenziali.

L'applicazione stessa di metodi numerici per l'approssimazione di dati e di funzioni, per la risoluzione di equazioni non lineari, di equazioni differenziali e per il calcolo degli autovalori di matrici, può richiedere la risoluzione di sistemi lineari.

Prima di analizzare i metodi numerici per la risoluzione di sistemi lineari, si richiamano alcune nozioni riguardanti vettori e matrici.

Norme di vettore e di matrice

Sia $\mathbf{x} = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$ un vettore colonna.

Nel seguito verranno considerate le seguenti norme di vettore:

- $||\mathbf{x}||_1 = \sum_{i=1}^{n} |x_i|$;
- $||\mathbf{x}||_2 = \sqrt{\sum_{i=1}^{n} x_i^2} = \sqrt{\mathbf{x}^T\mathbf{x}}$, **norma euclidea**;
- $||\mathbf{x}||_\infty = \max_{1 \le i \le n} |x_i|$.

Comandi Matlab

- `norm(x,1)` fornisce la norma `1` del vettore `x`;
- `norm(x,2)` oppure `norm(x)` fornisce la norma `2` del vettore `x`;
- `norm(x,inf)` fornisce la norma `inf`inito del vettore `x`.

Sia $\mathbf{A} = (a_{ij})_{i,j=1,\ldots,n} \in \mathbb{R}^{n,n}$ una matrice quadrata di ordine n. Nella trattazione dei sistemi lineari verranno utilizzate le seguenti norme di matrice:

- $||\mathbf{A}||_1 = \max\limits_{1\le j\le n} \sum\limits_{i=1}^{n} |a_{ij}|$;
- $||\mathbf{A}||_\infty = \max\limits_{1\le i\le n} \sum\limits_{j=1}^{n} |a_{ij}|, \quad (||\mathbf{A}||_\infty = ||\mathbf{A}^T||_1)$.

La norma 2 di matrice $||\mathbf{A}||_2$, detta **norma spettrale**, verrà definita in seguito.

Comandi Matlab

- `norm(A,1)` fornisce la norma `1` della matrice `A`;
- `norm(A,inf)` fornisce la norma `inf`inito della matrice `A`.

Definizione

Date una norma di matrice e una di vettore, si dice che le due norme sono **compatibili** se

$$||\mathbf{Ax}|| \leq ||\mathbf{A}||\,||\mathbf{x}||$$

per ogni $\mathbf{A} \in \mathbb{R}^{n,n}$ e per ogni $\mathbf{x} \in \mathbb{R}^n$.

Le norme 1 e infinito precedentemente definite per vettori e per matrici sono compatibili (come pure la norma 2).

Per le suddette norme valgono le seguenti proprietà:

- $||\mathbf{AB}|| \leq ||\mathbf{A}||\,||\mathbf{B}||$;
- $||\mathbf{I}|| = 1$ con $\mathbf{I}$ matrice identità.

Si elencano alcune forme speciali di matrici che si presenteranno in seguito:

Definizione

Una matrice $\mathbf{A} \in \mathbb{R}^{n,n}$ si dice **triangolare superiore** se

$$\mathbf{A} = \begin{pmatrix} \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star \\ 0 & 0 & \star & \star & \star \\ 0 & 0 & 0 & \star & \star \\ 0 & 0 & 0 & 0 & \star \end{pmatrix} \qquad a_{ij} = 0, \ i > j$$

Definizione

Una matrice $\mathbf{A} \in \mathbb{R}^{n,n}$ si dice **triangolare inferiore** se

$$\mathbf{A} = \begin{pmatrix} \star & 0 & 0 & 0 & 0 \\ \star & \star & 0 & 0 & 0 \\ \star & \star & \star & 0 & 0 \\ \star & \star & \star & \star & 0 \\ \star & \star & \star & \star & \star \end{pmatrix} \qquad a_{ij} = 0, \ i < j$$

Definizione

Una matrice $\mathbf{A} \in \mathbb{R}^{n,n}$ si dice **diagonale** se

$$\mathbf{A} = \begin{pmatrix} \star & 0 & 0 & 0 & 0 \\ 0 & \star & 0 & 0 & 0 \\ 0 & 0 & \star & 0 & 0 \\ 0 & 0 & 0 & \star & 0 \\ 0 & 0 & 0 & 0 & \star \end{pmatrix} \qquad a_{ij} = 0, \ i \neq j$$

Definizione

Una matrice $\mathbf{A} \in \mathbb{R}^{n,n}$ si dice **tridiagonale** se

$$\mathbf{A} = \begin{pmatrix} \star & \star & 0 & 0 & 0 \\ \star & \star & \star & 0 & 0 \\ 0 & \star & \star & \star & 0 \\ 0 & 0 & \star & \star & \star \\ 0 & 0 & 0 & \star & \star \end{pmatrix} \qquad a_{ij} = 0, \ |i - j| > 1$$

Definizione

Una matrice $\mathbf{A} \in \mathbb{R}^{n,n}$ si dice **Hessenberg superiore** se

$$\mathbf{A} = \begin{pmatrix} \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star \\ 0 & 0 & \star & \star & \star \\ 0 & 0 & 0 & \star & \star \end{pmatrix} \qquad a_{ij} = 0, \ \ i > j + 1$$

Si definisce di **Hessenberg inferiore** la matrice i cui elementi al di sopra della codiagonale superiore sono tutti nulli.

Definizioni

Si definisce **trasposta** della matrice $\mathbf{A}$ e si denota con $\mathbf{A}^T$, la matrice i cui elementi sono definiti da $(\mathbf{A}^T)_{ij} = (\mathbf{A})_{ji}$.

Una matrice $\mathbf{A} \in \mathbb{R}^{n,n}$ si dice **simmetrica** se $\mathbf{A}^T = \mathbf{A}$.

Una matrice $\mathbf{A} \in \mathbb{R}^{n,n}$ si dice **ortogonale** se $\mathbf{A}^T\mathbf{A} = \mathbf{A}\mathbf{A}^T = \mathbf{I}$.

Definizione

Una **matrice a blocchi**, o **matrice partizionata a blocchi**, è una matrice descritta in termini di sue sottomatrici anziché in termini dei suoi elementi.

Esempio

$$\mathbf{A} = \left(\begin{array}{cc|c} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ \hline a_{31} & a_{32} & a_{33} \end{array}\right) = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix}$$

con

$$\mathbf{A}_{11} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \ \mathbf{A}_{12} = \begin{pmatrix} a_{13} \\ a_{23} \end{pmatrix}, \ \mathbf{A}_{21} = \begin{pmatrix} a_{31} \\ a_{32} \end{pmatrix}^T, \ \mathbf{A}_{22} = (a_{33})$$

in tal caso si dice che $\mathbf{A}$ è una matrice 2×2 a blocchi.

È evidente che la partizione a blocchi di una matrice possa essere fatta in vari modi.

Per esempio, denotando con $\mathbf{a}_j$, $j = 1, \ldots, n$, il $j-$esimo vettore colonna, ogni matrice $\mathbf{A}$ con n colonne può essere riscritta come vettore riga:

$$\mathbf{A} = (\mathbf{a}_1 \ \mathbf{a}_2 \ \ldots \ \mathbf{a}_n);$$

analogamente, denotando con $\mathbf{a}_i^T$, $i = 1, \ldots, m$, l'$i-$esimo vettore riga, ogni matrice $\mathbf{A}$ con m righe può essere riscritta come vettore colonna:

$$\mathbf{A} = \begin{pmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \\ \vdots \\ \mathbf{a}_m^T \end{pmatrix}$$

La matrice **trasposta di una matrice partizionata a blocchi** si ottiene nel seguente modo:

$$\mathbf{A}^T = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \dots & \mathbf{A}_{1n} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \dots & \mathbf{A}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{m1} & \mathbf{A}_{m2} & \dots & \mathbf{A}_{mn} \end{pmatrix}^T = \begin{pmatrix} \mathbf{A}_{11}^T & \mathbf{A}_{21}^T & \dots & \mathbf{A}_{m1}^T \\ \mathbf{A}_{12}^T & \mathbf{A}_{22}^T & \dots & \mathbf{A}_{m2}^T \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{1n}^T & \mathbf{A}_{2n}^T & \dots & \mathbf{A}_{mn}^T \end{pmatrix}$$

La partizione a blocchi di una matrice consente talvolta di descrivere meglio la matrice e di effettuare più agevolmente i calcoli.

Quando il numero dei blocchi e le loro dimensioni sono compatibili, le operazioni tra matrici a blocchi possono essere eseguite formalmente trattando i blocchi come scalari.

Esempio

Siano

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{pmatrix}$$

due matrici con la stessa partizione a blocchi; si ha allora

$$\mathbf{A} + \mathbf{B} = \begin{pmatrix} \mathbf{A}_{11} + \mathbf{B}_{11} & \mathbf{A}_{12} + \mathbf{B}_{12} \\ \mathbf{A}_{21} + \mathbf{B}_{21} & \mathbf{A}_{22} + \mathbf{B}_{22} \end{pmatrix}, \ \alpha\mathbf{A} = \begin{pmatrix} \alpha\mathbf{A}_{11} & \alpha\mathbf{A}_{12} \\ \alpha\mathbf{A}_{21} & \alpha\mathbf{A}_{22} \end{pmatrix}.$$

Inoltre,

$$\mathbf{AB} = \begin{pmatrix} \mathbf{A}_{11}\mathbf{B}_{11} + \mathbf{A}_{12}\mathbf{B}_{21} & \mathbf{A}_{11}\mathbf{B}_{12} + \mathbf{A}_{12}\mathbf{B}_{22} \\ \mathbf{A}_{21}\mathbf{B}_{11} + \mathbf{A}_{22}\mathbf{B}_{21} & \mathbf{A}_{21}\mathbf{B}_{12} + \mathbf{A}_{22}\mathbf{B}_{22} \end{pmatrix}.$$

Osservazione

Quest'ultima proprietà è alla base degli algoritmi utilizzati dai calcolatori, poiché permette di ricondurre il calcolo di un prodotto tra matrici a somme di prodotti tra matrici di ordine più piccolo.

È particolarmente utile nel caso di matrici sparse, cioè con un gran numero di elementi nulli, nel qual caso si cercano partizioni in cui molti blocchi siano interamente nulli.

Definizione

Una **matrice triangolare superiore (inferiore) a blocchi** è una matrice quadrata che ha blocchi quadrati sulla diagonale e i cui blocchi sotto (sopra) la diagonale principale contengono solo zeri:

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \dots & \mathbf{A}_{1n} \\ \mathbf{O} & \mathbf{A}_{22} & \dots & \mathbf{A}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{O} & \mathbf{O} & \dots & \mathbf{A}_{nn} \end{pmatrix}$$

con $\mathbf{O}$ matrice identicamente nulla.

Per le matrici triangolari a blocchi vale la relazione:

$$det(\mathbf{A}) = \prod_{i=1}^{n} det(\mathbf{A}_{ii}).$$

Un caso particolare di matrice triangolare a blocchi è la **matrice diagonale a blocchi**, una matrice quadrata che ha blocchi quadrati sulla diagonale e i cui altri blocchi contengono solo zeri.

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{O} & \dots & \mathbf{O} \\ \mathbf{O} & \mathbf{A}_{22} & \dots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{O} & \mathbf{O} & \dots & \mathbf{A}_{nn} \end{pmatrix}$$

Definizione

Si definisce **matrice di permutazione**, una matrice ottenuta permutando le righe della matrice identità.

Osservazione

Le matrici di permutazione in ogni riga e ogni colonna hanno un solo elemento diverso da zero e uguale a 1.

Le matrici di permutazione sono matrici ortogonali.

Come mostra l'esempio che segue, se si moltiplica una matrice **A** o un vettore **b** a **sinistra** per un'opportuna matrice di permutazione **P**, si possono realizzare scambi di righe in **A** oppure di componenti in **b**.

Esempio

Consideriamo la seguente matrice di permutazione

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

e calcoliamo i prodotti **PA** e **Pb**, con $\mathbf{A} \in \mathbb{R}^{3,3}$ e $\mathbf{b} \in \mathbb{R}^3$. Si ha

$$\mathbf{PA} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \begin{pmatrix} a_{31} & a_{32} & a_{33} \\ a_{21} & a_{22} & a_{23} \\ a_{11} & a_{12} & a_{13} \end{pmatrix}$$

e

$$\mathbf{Pb} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} b_3 \\ b_2 \\ b_1 \end{pmatrix}$$

Definizione

Una matrice **A** si dice a **diagonale dominante per righe** se

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| \quad \text{per} \quad i = 1, \ldots, n$$

Esempio

La matrice

$$\mathbf{A} = \begin{pmatrix} 4 & 1 & -1 \\ 2 & -7 & 1 \\ 3 & -2 & 9 \end{pmatrix}$$

è a diagonale dominante per righe perché

$$|4| > |1| + |-1|, \qquad |-7| > |2| + |1|, \qquad |9| > |3| + |-2|$$

Definizione

Una matrice **A** si dice a **diagonale dominante per colonne** se

$$|a_{jj}| > \sum_{\substack{i=1 \\ i \neq j}}^{n} |a_{ij}| \quad \text{per} \quad j = 1, \ldots, n$$

Esempio

La matrice

$$\mathbf{A} = \begin{pmatrix} 9 & 1 & -1 \\ 2 & -7 & 1 \\ 3 & -2 & 4 \end{pmatrix}$$

è a diagonale dominante per colonne, perché

$$|9| > |2| + |3|, \qquad |-7| > |1| + |-2|, \qquad |4| > |-1| + |1|,$$

ma non è a diagonale dominante per righe.

Definizione

Una matrice **simmetrica A** si dice **definita positiva** se

$$\mathbf{x}^T \mathbf{A}\mathbf{x} > 0$$

per ogni $\mathbf{x} \neq \mathbf{0}$.

In seguito, verrà fornita una caratterizzazione delle matrici simmetriche e definite positive che, dal punto di vista pratico, risulta essere più conveniente per il riconoscimento delle suddette matrici.

Osservazione

La matrice $\mathbf{A} = \mathbf{B}^T\mathbf{B}$, con $\mathbf{B} \in \mathbb{R}^{n,n}$ avente determinante non nullo, è simmetrica e definita positiva.

Condizionamento di un sistema lineare

Si consideri il sistema lineare di n equazioni nelle n incognite $x_1, \ldots, x_n$:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \quad \ldots \quad \ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases}$$

In forma matriciale si ha

$$\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \iff \mathbf{Ax} = \mathbf{b},$$

Si assuma che il sistema assegnato abbia una e una sola soluzione; in tal caso, la matrice $\mathbf{A}$ si dice **non singolare**.

Prima di descrivere i metodi numerici di base per la risoluzione del suddetto sistema, si esamina il condizionamento del problema.

Si denotino con $\bar{\mathbf{A}}$ e $\bar{\mathbf{b}}$ i dati perturbati e con $\bar{\mathbf{x}}$ la soluzione in **aritmetica esatta** del sistema perturbato

$$\bar{\mathbf{A}}\bar{\mathbf{x}} = \bar{\mathbf{b}}$$

Nello studio del condizionamento si deve confrontare l'errore relativo

$$||\mathbf{x} - \bar{\mathbf{x}}||/||\mathbf{x}||$$

associato alla soluzione perturbata $\bar{\mathbf{x}}$, con gli errori relativi

$$||\mathbf{A} - \bar{\mathbf{A}}||/||\mathbf{A}|| \quad \text{e} \quad ||\mathbf{b} - \bar{\mathbf{b}}||/||\mathbf{b}||$$

associati ai dati perturbati.

Si può dimostrare il seguente risultato.

Teorema

Se $||\mathbf{A} - \bar{\mathbf{A}}|| < 1/(2||\mathbf{A}^{-1}||)$, il sistema $\bar{\mathbf{A}}\bar{\mathbf{x}} = \bar{\mathbf{b}}$ ammette una e una sola soluzione e

$$\frac{||\mathbf{x} - \bar{\mathbf{x}}||}{||\mathbf{x}||} \leq 2\,K(\mathbf{A}) \left(\frac{||\mathbf{A} - \bar{\mathbf{A}}||}{||\mathbf{A}||} + \frac{||\mathbf{b} - \bar{\mathbf{b}}||}{||\mathbf{b}||} \right),$$

ove

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

viene definito **numero di condizionamento** del sistema lineare $\mathbf{Ax} = \mathbf{b}$.

Si osservi che

$$K(\mathbf{A}) = ||\mathbf{A}||\,||\mathbf{A}^{-1}|| \geq ||\mathbf{A}\mathbf{A}^{-1}|| = ||\mathbf{I}|| = 1$$

Pertanto,

- se $K(\mathbf{A}) \approx 1$, la matrice $\mathbf{A}$ è detta **ben condizionata** e a piccole perturbazioni sui dati corrispondono perturbazioni sulla soluzione al più dello stesso ordine di grandezza di quelle sui dati;
- se $K(\mathbf{A}) >> 1$, la matrice si dice **mal condizionata** e a piccole perturbazioni sui dati *possono* corrispondere grandi perturbazioni sulla soluzione.

Esempi classici di sistemi lineari mal condizionati sono quello associato alla **matrice di Hilbert**

$$\mathbf{H}_n = \begin{pmatrix} 1 & \frac{1}{2} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n+1} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \cdots & \frac{1}{2n-1} \end{pmatrix}$$

e quello alla **matrice di Vandermonde**

$$\mathbf{V}_n = \begin{pmatrix} x_1^n & \cdots & x_1 & 1 \\ x_2^n & \cdots & x_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_{n+1}^n & \cdots & x_{n+1} & 1 \end{pmatrix}$$

ove $x_i \neq x_j$ per $i \neq j$.

Comandi Matlab

- `cond(A,1)` fornisce il numero di condizionamento in norma `1` del sistema $\mathbf{Ax} = \mathbf{b}$;
- `cond(A,inf)` fornisce il numero di condizionamento in norma `inf`inito del sistema $\mathbf{Ax} = \mathbf{b}$;
- `hilb(n)` genera la matrice di Hilbert $\mathbf{H}_n$ di ordine `n`;
- `vander(x)` genera la matrice di Vandermonde $\mathbf{V}_n$ associata al vettore `x` con componenti $x_1, \ldots, x_{n+1}$.

L'esempio che segue mostra che il condizionamento di un sistema lineare non dipende dalla norma scelta. Pertanto, se un sistema è ben/mal condizionato rispetto alla norma 1, lo è pure, per esempio, rispetto alla norma infinito.

Esempio 1

Calcoliamo, mediante istruzioni Matlab, il condizionamento in norma 1 e in norma infinito della matrice di Vandermonde associata a un vettore di n elementi equispaziati in $[0, 1]$. Scegliamo $n = 4$ e $n = 10$.

```
>> x = linspace(0,1,4);
>> A = vander(x);
>> K1 = cond(A,1)
K1 =
    180
>> Kinf = cond(A,inf)
Kinf =
    216
>> x = linspace(0,1,10);
>> B = vander(x);
>> K1 = cond(B,1)
K1 =
    4.2412e+07
>> Kinf=cond(B,inf)
Kinf =

    4.8184e+07
```

I numeri di condizionamento nelle norme 1 e infinito, per ciascuna matrice di Vandermonde calcolata, hanno lo stesso ordine di grandezza.

Esempio 2

In questo esempio evidenziamo gli effetti che il cattivo condizionamento di un sistema lineare ha sulla soluzione del sistema stesso. A tal scopo, risolviamo il sistema lineare

$$\mathbf{Ax} = \mathbf{b}$$

con $\mathbf{A} = \mathbf{H}_n$ matrice di Hilbert di ordine $n = 2, 3, ..., 14$ e $\mathbf{b}$ definito in modo tale che il sistema abbia il vettore unitario $\mathbf{u}$ come soluzione. Pertanto, poniamo

$$b_i = \sum_{j=1}^{n} a_{ij}, \;\; i = 1, \ldots, n$$

... continua esempio

Per ogni valore di n, calcoliamo il numero di condizionamento $K_\infty(\mathbf{H}_n)$ del sistema e l'errore relativo associato alla soluzione $\mathbf{x}$ in norma infinito. Osserviamo che, essendo la matrice di Hilbert simmetrica, si ha $K_\infty(\mathbf{H}_n) = K_1(\mathbf{H}_n)$.

Per la risoluzione del sistema lineare $\mathbf{Ax} = \mathbf{b}$ useremo il comando Matlab `x = A\b`, che implementa il metodo delle eliminazioni di Gauss.

```
ord = 2:14;
for n = 1:length(ord)
   A = hilb(ord(n));
   b = sum(A,2);
   x = A\b;
   u = ones(ord(n),1);
   err(n) = norm(u-x,inf)/norm(u,inf);
   Kinf(n) = cond(A,inf);
end
```

... continua esempio

I valori ottenuti sono stati riportati nella tabella sottostante.

n	$\|\|\mathbf{u}-\mathbf{x}\|\|_\infty / \|\|\mathbf{u}\|\|_\infty$	$K_\infty(\mathbf{A})$
2	7.7716e−16	2.7000e+01
3	1.2434e−14	7.4800e+02
4	2.3259e−13	2.8375e+04
5	3.1360e−12	9.4366e+05
6	4.9815e−10	2.9070e+07
7	3.2718e−08	9.8519e+08
8	6.9786e−07	3.3873e+10
9	2.9759e−05	1.0997e+12
10	6.6607e−04	3.5354e+13
11	3.0946e−02	1.2300e+15
12	1.9342e−01	3.7520e+16
13	6.7285e+00	3.9019e+17
14	8.6973e+00	2.7983e+18

Osserviamo che, al crescere di n, aumenta il numero di condizionamento e, conseguentemente, diminuisce l'accuratezza della soluzione.
Le inevitabili perturbazioni di $\mathbf{A}$ e di $\mathbf{b}$, dovute alla loro rappresentazione in aritmetica floating-point con precisione finita, generano un errore nella soluzione, che è tanto più grande, quanto più è elevato il numero di condizionamento del sistema lineare.

Esempio 3

Siano (x_i, y_i), $i = 1, \ldots, n+1$, $n+1$ coppie di punti tali che $x_i \neq x_j$ per $i \neq j$, e sia

$$p_n(x) = c_1 x^n + c_2 x^{n-1} + \ldots + c_n x + c_{n+1}$$

la rappresentazione monomiale del polinomio interpolante i dati (x_i, y_i). Per calcolare i coefficienti di $p_n(x)$ si impongono le seguenti condizioni di interpolazione

$$p_n(x_i) = y_i, \;\; i = 1, \ldots, n+1$$

... continua esempio

Si ottengono così $n+1$ equazioni lineari nelle incognite c_i:

$$c_1 x_i^n + c_2 x_i^{n-1} + \ldots + c_n x_i + c_{n+1} = y_i, \quad i = 1, \ldots, n+1$$

I coefficienti c_i sono dunque soluzione del sistema lineare

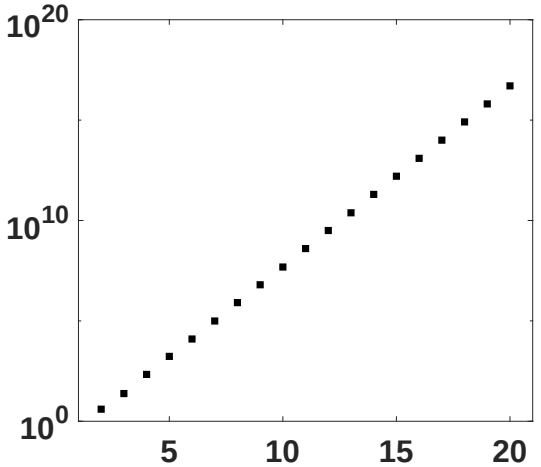
$$\begin{pmatrix} x_1^n & \ldots & x_1 & 1 \\ x_2^n & \ldots & x_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_{n+1}^n & \ldots & x_{n+1} & 1 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n+1} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n+1} \end{pmatrix} \iff \mathbf{V_n c} = \mathbf{y}$$

la cui matrice dei coefficienti è una matrice di Vandermonde.

... continua esempio

In figura è riportato il numero di condizionamento K_∞ (in norma infinito) della matrice di Vandermonde generata dal vettore $\mathbf{x}$ di $n = 2, \ldots, 20$ elementi equispaziati in $[0, 1]$.
Si nota che il condizionamento della matrice peggiora al crescere dell'ordine n e, per questo motivo, non è possibile utilizzare la rappresentazione monomiale del polinomio interpolante quando n è grande.

Figura: $K_\infty(\mathbf{V}_n)$ per $n = 2, 3, \ldots, 20$.



Metodi numerici

Si descrivono ora alcuni metodi numerici e gli algoritmi, che implementano i suddetti metodi, per la risoluzione di sistemi lineari di ordine n.

Per ciascun algoritmo verrà fornito il **costo computazionale**, ovvero il numero di operazioni aritmetiche (addizioni e moltiplicazioni) che esso richiede per la sua esecuzione.

Nel caso dei sistemi lineari il costo è legato alla dimensione n del sistema e, in generale, viene quantificato il numero di operazioni per valori di n grandi.

Metodi numerici per sistemi diagonali

Vengono dapprima esaminati i metodi per la risoluzione di sistemi lineari, la cui matrice dei coefficienti ha una struttura particolare, per esempio diagonale, triangolare superiore oppure triangolare inferiore.

Se la matrice dei coefficienti **A** è **diagonale**, con $a_{ii} \neq 0,\ i = 1, \ldots, n$, la soluzione del sistema lineare

$$\begin{cases} a_{11}x_1 & & & & & = b_1 \\ & a_{22}x_2 & & & & = b_2 \\ & & a_{33}x_3 & & & = b_3 \\ & & & \ddots & & \vdots \\ & & & & a_{nn}x_n & = b_n \end{cases}$$

si ricava immediatamente mediante le seguenti n divisioni:

$$x_i = \frac{b_i}{a_{ii}}, \quad i = 1, \ldots, n$$

Metodi numerici per sistemi triangolari

Se la matrice dei coefficienti **A** è **triangolare superiore**, con $a_{ii} \neq 0,\ i = 1, \ldots, n$, la soluzione del sistema lineare

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + & \ldots + & a_{1n}x_n & = b_1 \\ & \ldots & \ldots & \ldots \\ & a_{n-1n-1}x_{n-1} + & a_{n-1n}x_n & = b_{n-1} \\ & & a_{nn}x_n & = b_n \end{cases}$$

si ottiene ricavando l'incognita x_n dall'ultima equazione, x_{n-1} dalla penultima equazione (dopo aver sostituito in essa il valore di x_n),..., x_1 dalla prima (dopo aver sostituito in essa i valori di $x_2, \ldots, x_n$):

$$x_n = \frac{b_n}{a_{nn}}$$

$$x_{n-1} = \frac{b_{n-1} - a_{n-1n}x_n}{a_{n-1n-1}}$$

$$\vdots$$

$$x_1 = \frac{b_1 - \sum_{j=2}^{n} a_{1j}x_j}{a_{11}}$$

Tale procedura è nota con il nome di **metodo di sostituzione all'indietro**.

L'algoritmo che implementa il suddetto metodo è dunque:

$$x_n = \frac{b_n}{a_{nn}}$$

$$x_i = \frac{b_i - \sum_{j=i+1}^{n} a_{ij} x_j}{a_{ii}} \quad i = n-1, \ldots, 1$$

Il costo computazionale, in termini delle operazioni aritmetiche di moltiplicazione (e di addizione), è essenzialmente $n^2/2$ per n grande.

Di seguito si riporta l'algoritmo, in linguaggio Matlab, che implementa il metodo di sostituzione all'indietro.

Function Matlab

```
function x = indietro(A,b)
n = length(b);
x = zeros(n,1);
x(n) = b(n)/A(n,n);
for i = n-1:-1:1
  s = 0;
  for j = i+1:n
    s = s+A(i,j)*x(j);
  end
  x(i) = (b(i)-s)/A(i,i);
end
```

Un'alternativa che sfrutta la capacità di Matlab di operare in modo ottimizzato direttamente sui vettori (attraverso la libreria BLAS, Basic Linear Algebra Subprograms) è la seguente:

```
s = A(i,i+1:n)*x(i+1:n)
```

Se la matrice dei coefficienti **A** è **triangolare inferiore**, con $a_{ii} \neq 0,\ i = 1, \ldots, n$, la soluzione del sistema lineare

$$\begin{cases} a_{11}x_1 & & & & = b_1 \\ a_{21}x_1 + & a_{22}x_2 & & & = b_2 \\ & \ldots & \ldots & & \\ a_{n1}x_1 + & a_{n2}x_2 + & \ldots + & a_{nn}x_n & = b_n \end{cases}$$

si ottiene ricavando l'incognita x_1 dalla prima equazione, x_2 dalla seconda equazione (dopo aver sostituito in essa il valore di x_1),..., x_n dall'ultima (dopo aver sostituito in essa i valori di $x_1, \ldots, x_{n-1}$):

$$x_1 = \frac{b_1}{a_{11}}$$

$$x_2 = \frac{b_2 - a_{21}x_1}{a_{22}}$$

$$\vdots$$

$$x_n = \frac{b_n - \sum_{j=1}^{n-1} a_{nj}x_j}{a_{nn}}$$

Tale procedura è nota con il nome di **metodo di sostituzione in avanti**.

L'algoritmo che implementa il suddetto metodo è dunque:

$$x_1 = \frac{b_1}{a_{11}}$$

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} a_{ij} x_j}{a_{ii}} \quad i = 2, \ldots, n$$

Il costo computazionale, in termini delle operazioni aritmetiche di moltiplicazione (e di addizione), è essenzialmente $n^2/2$ per n grande.

Di seguito si riporta l'algoritmo, in linguaggio Matlab, che implementa il metodo di sostituzione in avanti.

Function Matlab

```
function x = avanti(A,b)
n = length(b);
x = zeros(n,1);
x(1) = b(1)/A(1,1);
for i = 2:n
  s = A(i,1:i-1)*x(1:i-1);
  x(i) = (b(i)-s)/A(i,i);
end
```

Metodo delle eliminazioni di Gauss

Il metodo delle eliminazioni di Gauss, insieme alla regola di Cramer, rappresenta un metodo classico dell'Algebra Lineare per la risoluzione di un sistema lineare $\mathbf{Ax} = \mathbf{b}$.

Il costo computazionale in termini di moltiplicazioni del metodo delle eliminazioni di Gauss è essenzialmente $n^3/3$; quello della regola di Cramer è invece dell'ordine di $(n+1)!$.

Gli algoritmi con complessità fattoriale sono proibitivi anche per i calcolatori più avanzati oggi disponibili.

Ad esempio, per risolvere un sistema di $n = 24$ equazioni, un calcolatore in grado di eseguire 10^{15} operazioni al secondo impiega

$\approx 4.6 \cdot 10^{-12}$ **secondi** con il metodo delle eliminazioni di Gauss,

≈ 50 **anni** con la regola di Cramer.

Pertanto, anche per sistemi di modeste dimensioni, l'applicazione della regola di Cramer risulta impraticabile.

Se la matrice dei coefficienti $\mathbf{A}$ non ha una struttura particolare, per risolvere il sistema lineare $\mathbf{Ax} = \mathbf{b}$ di ordine n, si può utilizzare il **metodo delle eliminazioni di Gauss**.

Il metodo delle eliminazioni di Gauss consta essenzialmente di due fasi:

1. trasformazione, in $n-1$ passi, del sistema assegnato $\mathbf{Ax} = \mathbf{b}$ nel sistema $\mathbf{Ux} = \bar{\mathbf{b}}$ equivalente a quello assegnato (ovvero che ammette la stessa soluzione $\mathbf{x}$), con $\mathbf{U}$ matrice triangolare superiore;
2. risoluzione del sistema $\mathbf{Ux} = \bar{\mathbf{b}}$ mediante la tecnica di sostituzione all'indietro.

Il metodo di Gauss si basa sulle seguenti proprietà dei sistemi lineari:

- la soluzione rimane invariata se si scambiano tra loro due equazioni del sistema;
- la soluzione rimane invariata se si sostituisce a un'equazione del sistema una combinazione lineare dell'equazione stessa con un'altra.

Per la descrizione del metodo delle eliminazioni di Gauss consideriamo il seguente sistema lineare di ordine $n = 4$:

$$\mathbf{Ax} = \mathbf{b} \Longleftrightarrow \begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 = b_3 \\ a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 = b_4 \end{cases}$$

Poniamo $a_{ij}^{(1)} := a_{ij}$ e $b_i^{(1)} := b_i$ per ogni $i, j = 1, \ldots, 4$.

PASSO k = 1

Supponiamo $a_{11}^{(1)} \neq 0$. In caso contrario, se la matrice **A** è non singolare, esiste certamente $a_{r1}^{(1)} \neq 0$ con $2 \leq r \leq 4$ e scambiamo la prima equazione con l'r-esima equazione.

Quindi eliminiamo l'incognita x_1 nelle equazioni $i = 2, 3, 4$, sostituendo l'i-esima equazione con l'equazione che si ottiene sottraendo all'i-esima stessa la prima equazione moltiplicata per $m_{i1} := a_{i1}^{(1)}/a_{11}^{(1)}$:

$$\begin{array}{rl} & a_{i1}^{(1)}x_1 + \quad a_{i2}^{(1)}x_2 + a_{i3}^{(1)}x_3 + a_{i4}^{(1)}x_4 = \quad b_i^{(1)} \\ m_{i1} & (a_{11}^{(1)}x_1 + \quad a_{12}^{(1)}x_2 + a_{13}^{(1)}x_3 + a_{14}^{(1)}x_4) = \quad m_{i1}\, b_1^{(1)} \\ \hline & \qquad\qquad a_{i2}^{(2)}x_2 + a_{i3}^{(2)}x_3 + a_{i4}^{(2)}x_4 = \quad b_i^{(2)} \end{array}$$

ove

$$a_{ij}^{(2)} = a_{ij}^{(1)} - m_{i1}\, a_{1j}^{(1)}, \quad b_i^{(2)} = b_i^{(1)} - m_{i1}\, b_1^{(1)}, \quad i, j = 2, 3, 4.$$

Otteniamo così il sistema lineare:

$$\begin{cases} a_{11}^{(1)} x_1 + a_{12}^{(1)} x_2 + a_{13}^{(1)} x_3 + a_{14}^{(1)} x_4 = b_1^{(1)} \\ \qquad\quad a_{22}^{(2)} x_2 + a_{23}^{(2)} x_3 + a_{24}^{(2)} x_4 = b_2^{(2)} \\ \qquad\quad a_{32}^{(2)} x_2 + a_{33}^{(2)} x_3 + a_{34}^{(2)} x_4 = b_3^{(2)} \\ \qquad\quad a_{42}^{(2)} x_2 + a_{43}^{(2)} x_3 + a_{44}^{(2)} x_4 = b_4^{(2)} \end{cases}$$

PASSO k = 2

A meno di uno scambio della seconda equazione con una successiva, possiamo supporre $a_{22}^{(2)} \neq 0$.

Quindi eliminiamo l'incognita x_2 nelle equazioni $i = 3, 4$, sostituendo l'i-esima equazione con l'equazione che si ottiene sottraendo all'i-esima stessa la seconda equazione moltiplicata per $m_{i2} := a_{i2}^{(2)} / a_{22}^{(2)}$. Abbiamo così:

$$\begin{cases} a_{11}^{(1)} x_1 + a_{12}^{(1)} x_2 + a_{13}^{(1)} x_3 + a_{14}^{(1)} x_4 = b_1^{(1)} \\ a_{22}^{(2)} x_2 + a_{23}^{(2)} x_3 + a_{24}^{(2)} x_4 = b_2^{(2)} \\ a_{33}^{(3)} x_3 + a_{34}^{(3)} x_4 = b_3^{(3)} \\ a_{43}^{(3)} x_3 + a_{44}^{(3)} x_4 = b_4^{(3)} \end{cases}$$

ove

$$a_{ij}^{(3)} = a_{ij}^{(2)} - m_{i2} a_{2j}^{(2)}, \quad b_i^{(3)} = b_i^{(2)} - m_{i2} b_2^{(2)}, \quad i, j = 3, 4.$$

PASSO k = 3

Supponiamo, a meno di uno scambio, $a_{33}^{(3)} \neq 0$.

Quindi eliminiamo l'incognita x_3 nell'equazione $i = 4$, sostituendo l'i-esima equazione con l'equazione che si ottiene sottraendo all'i-esima stessa la terza equazione moltiplicata per $m_{i3} := a_{i3}^{(3)}/a_{33}^{(3)}$. Otteniamo così il seguente sistema triangolare superiore

$$\begin{cases} a_{11}^{(1)}x_1 + a_{12}^{(1)}x_2 + a_{13}^{(1)}x_3 + a_{14}^{(1)}x_4 = b_1^{(1)} \\ a_{22}^{(2)}x_2 + a_{23}^{(2)}x_3 + a_{24}^{(2)}x_4 = b_2^{(2)} \\ a_{33}^{(3)}x_3 + a_{34}^{(3)}x_4 = b_3^{(3)} \\ a_{44}^{(4)}x_4 = b_4^{(4)} \end{cases} \iff \mathbf{U}\mathbf{x} = \bar{\mathbf{b}}$$

ove

$$a_{ij}^{(4)} = a_{ij}^{(3)} - m_{i3}a_{3j}^{(3)}, \quad b_i^{(4)} = b_i^{(3)} - m_{i3}b_3^{(3)}, \quad i,j = 4.$$

Risolviamo infine il sistema $\mathbf{U}\mathbf{x} = \bar{\mathbf{b}}$ con la tecnica di sostituzione all'indietro.

Il seguente schema di calcolo riassume le trasformazioni del metodo delle eliminazioni di Gauss:

$$k = 1, \ldots, n-1$$

$$i = k+1, \ldots, n \quad \begin{cases} m_{ik} & = a_{ik}^{(k)} / a_{kk}^{(k)} \\ a_{ij}^{(k+1)} & = a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}, \ j = k+1, \ldots, n \\ b_i^{(k+1)} & = b_i^{(k)} - m_{ik} b_k^{(k)} \end{cases}$$

$$\begin{cases} x_n = b_n^{(n)} / a_{nn}^{(n)} \\ x_k = (b_k^{(k)} - \sum_{j=k+1}^{n} a_{kj}^{(k)} x_j) / a_{kk}^{(k)}, \quad k = n-1, \ldots, 1 \end{cases}$$

Il costo computazionale del metodo delle eliminazioni di Gauss, in termini delle operazioni aritmetiche di moltiplicazione (e di addizione), è essenzialmente $n^3/3$ per n grande.

Se **A** è **simmetrica** e **non si effettuano scambi**, si dimostra che la sottomatrice degli elementi $a_{ij}^{(k)}$:

$$\begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1k}^{(1)} & \dots & a_{1n}^{(1)} \\ & a_{22}^{(2)} & \dots & a_{2k}^{(2)} & \dots & a_{2n}^{(2)} \\ & & \dots & & \dots & \\ & & & a_{kk}^{(k)} & \dots & a_{kn}^{(k)} \\ & & & \vdots & \vdots & \vdots \\ & & & a_{nk}^{(k)} & \dots & a_{nn}^{(k)} \end{pmatrix}$$

per ogni k, è ancora simmetrica. Pertanto, al passo k, tenendo conto che la simmetria è preservata, è possibile generare soltanto la metà degli elementi $a_{ij}^{(k+1)}$ (per esempio, quelli della parte triangolare superiore). In tal caso il costo del metodo delle eliminazioni di Gauss si dimezza ed è essenzialmente $n^3/6$.

Osserviamo che:

- a ogni passo k gli elementi $a_{ij}^{(k)}$ e $b_i^{(k)}$ possono essere memorizzati all'interno della matrice $\mathbf{A}$ e del vettore $\mathbf{b}$, nelle posizioni ij e i, rispettivamente;
- i moltiplicatori m_{ik} possono essere memorizzati nella matrice $\mathbf{A}$ nella posizione ik.

Nell'ipotesi in cui $a_{kk}^{(k)} \neq 0$ per ogni $k = 1, \ldots, n-1$, l'algoritmo che implementa il metodo delle eliminazioni di Gauss in linguaggio Matlab, è il seguente.

Function Matlab

```
function x = gauss_noscambi(A,b)
n = length(b);
for k = 1:n-1
  for i = k+1:n
    A(i,k) = A(i,k)/A(k,k);
    for j = k+1:n
      A(i,j) = A(i,j)-A(i,k)*A(k,j);
    end
    b(i) = b(i)-A(i,k)*b(k);
  end
end
x = zeros(n,1);
x(n) = b(n)/A(n,n);
for i = n-1:-1:1
  x(i) = (b(i)-A(i,i+1:n)*x(i+1:n))/A(i,i);
end
```

Affinché il metodo delle eliminazioni di Gauss possa procedere, è necessario che al passo k risulti $a_{kk}^{(k)} \neq 0$.

Si dimostra che tale condizione è soddisfatta nei seguenti casi:

- **A** è a diagonale dominante per righe;
- **A** è a diagonale dominante per colonne;
- **A** è simmetrica e definita positiva.

Esempio

Risolviamo il seguente sistema lineare

$$\mathbf{Ax} = \mathbf{b} \iff \begin{pmatrix} 2 & -1 & 1 & -2 \\ 0 & 2 & 0 & -1 \\ 1 & 0 & -2 & 1 \\ 0 & 2 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 4 \end{pmatrix}$$

con soluzione $\mathbf{x} = (1, 1, 1, 1)^T$, applicando l'algoritmo che implementa il metodo delle eliminazioni di Gauss.

... continua esempio

PASSO k = 1

Osserviamo che $a_{11} \neq 0$ e, quindi, procediamo con le trasformazioni. Memorizziamo i moltiplicatori m_{i1} al posto di a_{i1}, $i = 2, 3, 4$ e generiamo gli elementi a_{ij} con $i, j = 2, 3, 4$:

$$\mathbf{A} = \begin{pmatrix} 2 & -1 & 1 & -2 \\ 0 & 2 & 0 & -1 \\ 1/2 & 1/2 & -5/2 & 2 \\ 0 & 2 & 1 & 1 \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 4 \end{pmatrix}$$

... continua esempio

PASSO k = 2

Osserviamo che $a_{22} \neq 0$. Memorizziamo i moltiplicatori m_{i2} al posto di a_{i2}, $i = 3, 4$ e generiamo gli elementi a_{ij} con $i, j = 3, 4$:

$$\mathbf{A} = \begin{pmatrix} 2 & -1 & 1 & -2 \\ 0 & 2 & 0 & -1 \\ 1/2 & 1/4 & -5/2 & 9/4 \\ 0 & 1 & 1 & 2 \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} 0 \\ 1 \\ -1/4 \\ 3 \end{pmatrix}$$

... continua esempio

PASSO k = 3

Osserviamo che $a_{33} \neq 0$. Memorizziamo il moltiplicatore m_{i3} al posto di a_{i3}, $i = 4$ e generiamo l'elemento $a_{4,4}$:

$$\mathbf{A} = \begin{pmatrix} 2 & -1 & 1 & -2 \\ 0 & 2 & 0 & -1 \\ 1/2 & 1/4 & -5/2 & 9/4 \\ 0 & 1 & -2/5 & 29/10 \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} 0 \\ 1 \\ -1/4 \\ 29/10 \end{pmatrix}$$

Applichiamo la tecnica di sostituzione all'indietro:

$$\begin{aligned} x_4 &= (29/10)/(29/10) = 1 \\ x_3 &= (-1/4 - 9/4)/(-5/2) = 1 \\ x_2 &= (1 - (-1))/2 = 1 \\ x_1 &= -(-2 + 1 - 1)/2 = 1 \end{aligned}$$

... continua esempio

Consideriamo ora le matrici **L** e **U** così definite: **L** è la matrice triangolare inferiore i cui elementi sono definiti da

$$\ell_{ii} = 1, \quad \ell_{ij} = a_{ij} \text{ per } i > j$$

e **U** è la matrice triangolare superiore, i cui elementi sono

$$u_{ij} = a_{ij} \text{ per } j \geq i$$

... continua esempio

Osserviamo che il prodotto

$$\mathbf{LU} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1/2 & 1/4 & 1 & 0 \\ 0 & 1 & -2/5 & 1 \end{pmatrix} \begin{pmatrix} 2 & -1 & 1 & -2 \\ 0 & 2 & 0 & -1 \\ 0 & 0 & -5/2 & 9/4 \\ 0 & 0 & 0 & 29/10 \end{pmatrix}$$

genera la matrice di partenza

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

Fattorizzazione PA=LU e applicazioni

L'esempio precedente mostra che il metodo delle eliminazioni di Gauss, **quando non richiede scambi**, realizza la seguente fattorizzazione della matrice **A**:

$$\mathbf{A} = \mathbf{LU}$$

ove **U** è la matrice triangolare superiore del sistema equivalente al sistema di partenza e **L** è la matrice triangolare inferiore dei moltiplicatori con diagonale unitaria.

Per determinare le matrici **L** e **U** della fattorizzazione $\mathbf{A} = \mathbf{LU}$, occorre aggiungere nell'algoritmo di Gauss (implementato nella `function` Matlab `gauss_noscambi`), al termine delle trasformazioni degli elementi della matrice **A**, soltanto le due istruzioni che generano **L** e **U** a partire dalla matrice trasformata **A**.

Function Matlab

```
function [L,U] = elleu(A)
n = size(A,1);
for k = 1:n-1
  for i = k+1:n
    A(i,k) = A(i,k)/A(k,k);
    for j = k+1:n
      A(i,j) = A(i,j)-A(i,k)*A(k,j);
    end
  end
end
L = tril(A,-1)+eye(n);
U = triu(A);
```

Se al passo k del metodo delle eliminazioni di Gauss si ha $a_{kk}^{(k)} = 0$, il metodo può procedere solo se si scambia la k-esima equazione con la r-esima con $k < r \leq n$ tale che $a_{rk}^{(k)} \neq 0$.

In questo caso, cioè **in caso di scambi**, il metodo delle eliminazioni di Gauss realizza la seguente fattorizzazione della matrice **A**:

$$\mathbf{PA} = \mathbf{LU}$$

ove **U** è triangolare superiore, **L** è triangolare inferiore con diagonale unitaria e **P** è una matrice di permutazione, definita dagli scambi richiesti dal metodo.

Poiché l'introduzione degli scambi non comporta alcun calcolo, il costo computazionale della fattorizzazione $\mathbf{PA} = \mathbf{LU}$ della matrice **A** è uguale a quello della fattorizzazione $\mathbf{A} = \mathbf{LU}$ e vale all'incirca $n^3/3$ per n grande.

Per garantire una **migliore stabilità numerica** dell'algoritmo di Gauss conviene operare uno scambio di equazioni anche quando $a_{kk}^{(k)}$, pur essendo diverso da zero, è "piccolo" in valore assoluto.

In tal caso, infatti, il moltiplicatore $m_{ik} = a_{ik}^{(k)} / a_{kk}^{(k)}$ risulterebbe "grande" e potrebbe amplificare eccessivamente gli errori di arrotondamento, che si generano durante l'esecuzione dell'algoritmo.

Pertanto, per garantire una maggiore accuratezza della soluzione, conviene utilizzare moltiplicatori m_{ik} "piccoli", e questi si ottengono con una scelta opportuna degli elementi **pivot** $a_{kk}^{(k)}$:

ad ogni passo k conviene individuare l'indice di riga r per il quale risulta

$$|a_{rk}^{(k)}| = \max_{k \leq i \leq n} |a_{ik}^{(k)}|$$

e scambiare la k-esima equazione con l'r-esima equazione.

Tale strategia è nota sotto il nome di **pivoting parziale**.

Il pivoting è **superfluo** quando:

- **A** è a diagonale dominante per colonne;
- **A** è simmetrica e definita positiva.

L'esempio che segue evidenzia l'importanza della strategia del pivoting quando si risolve un sistema lineare con il metodo delle eliminazioni di Gauss.

Esempio

Consideriamo il sistema lineare $\mathbf{Ax} = \mathbf{b}$ di ordine $n = 18$, con

$$a_{ij} = \cos((j-1)\theta_i), \qquad \theta_i = \frac{2i-1}{2n}\pi, \qquad b_i = \sum_{j=1}^{n} a_{ij}$$

la cui soluzione esatta è $\mathbf{x} = (1, 1, \ldots, 1)^T$. Si ha $K_\infty(\mathbf{A}) \approx 17$.

Denotata con $\bar{\mathbf{x}}$ la soluzione del sistema calcolata mediante il metodo delle eliminazioni di **Gauss senza pivoting** e con $\bar{\mathbf{x}}_p$ la soluzione calcolata mediante il metodo di **Gauss con pivoting parziale**, si ha

$$\frac{||\mathbf{x} - \bar{\mathbf{x}}||_\infty}{||\mathbf{x}||_\infty} \approx 10^{-9} \quad \text{e} \quad \frac{||\mathbf{x} - \bar{\mathbf{x}}_p||_\infty}{||\mathbf{x}||_\infty} \approx 10^{-15}$$

Pertanto il metodo delle eliminazioni di Gauss con pivoting parziale fornisce una soluzione più accurata.

Comandi Matlab

- `x = A\b` calcola la soluzione `x` di $\mathbf{Ax} = \mathbf{b}$ con il metodo delle eliminazioni di Gauss con pivoting parziale. È importante sapere che il comando richiama un algoritmo specifico a seconda delle caratteristiche della matrice `A` (diagonale, triangolare, simmetrica e definita positiva,...); se la matrice non soddisfa nessuna delle caratteristiche previste, allora viene calcolata una generica fattorizzazione $\mathbf{PA} = \mathbf{LU}$.
- `[L,U,P] = lu(A)` calcola i fattori `L`, `U`, e `P` della fattorizzazione $\mathbf{PA} = \mathbf{LU}$ di `A`.

Applicazioni della fattorizzazione $PA = LU$

Descriviamo ora alcune applicazioni della fattorizzazione $\mathbf{PA} = \mathbf{LU}$.

- **Applicazione 1. Risoluzione del sistema lineare $\mathbf{Ax} = \mathbf{b}$**

Per la risoluzione di un sistema lineare $\mathbf{Ax} = \mathbf{b}$ si può procedere nel seguente modo:

$$\mathbf{Ax} = \mathbf{b} \Longrightarrow \mathbf{PAx} = \mathbf{Pb} \Longrightarrow \mathbf{L}\underbrace{\mathbf{Ux}}_{\mathbf{y}} = \mathbf{Pb}$$

da cui segue la risoluzione di due sistemi triangolari

$$\begin{cases} \mathbf{Ly} = \mathbf{Pb} & \Rightarrow \mathbf{y} \\ \mathbf{Ux} = \mathbf{y} & \Rightarrow \mathbf{x} \end{cases}$$

Se i fattori $\mathbf{L}, \mathbf{U}, \mathbf{P}$ della fattorizzazione $\mathbf{PA} = \mathbf{LU}$ sono noti, il costo della risoluzione del sistema lineare $\mathbf{Ax} = \mathbf{b}$ è soltanto n^2; altrimenti, il costo è pari a quello della fattorizzazione $\mathbf{PA} = \mathbf{LU}$, cioè $n^3/3$ (essendo trascurabile, per n grande, il costo della risoluzione dei due sistemi lineari triangolari, rispetto al costo della fattorizzazione).

- **Applicazione 2. Calcolo del determinante di una matrice**

Per il calcolo del determinante di $\mathbf{A}$ si può così procedere:

$$\det(\mathbf{PA}) = \det(\mathbf{LU}) \Longrightarrow \det(\mathbf{P})\det(\mathbf{A}) = \det(\mathbf{L})\det(\mathbf{U})$$

Poiché $\det(\mathbf{L}) = 1$, $\det(\mathbf{U}) = \prod_{i=1}^{n} u_{ii}$ e si dimostra che $\det(\mathbf{P}) = (-1)^s$ ove s è il numero totale degli scambi di equazioni richiesti dal metodo, si ha

$$\det(\mathbf{A}) = (-1)^s \prod_{i=1}^{n} u_{ii}$$

Questa procedura è alla base del comando Matlab `det(A)`, che restituisce il determinante della matrice `A`.

- **Applicazione 3. Calcolo dell'inversa di una matrice**

Un metodo efficiente e ottimale dal punto di vista del costo computazionale per il calcolo dell'inversa di $\mathbf{A}$ è il seguente:

$$\mathbf{PA} = \mathbf{LU} \Longrightarrow (\mathbf{PA})^{-1} = (\mathbf{LU})^{-1} \Longrightarrow \mathbf{A}^{-1}\mathbf{P}^{-1} = \mathbf{U}^{-1}\mathbf{L}^{-1}$$

da cui

$$\mathbf{A}^{-1} = \mathbf{U}^{-1}\mathbf{L}^{-1}\mathbf{P}$$

Il costo computazionale del metodo descritto è circa pari a n^3 per n grande.

Questa procedura è alla base del comando Matlab `inv(A)`, che restituisce l'inversa della matrice `A`.

Osserviamo che, poiché il costo del calcolo dell'inversa è il triplo del costo del metodo delle eliminazioni di Gauss, non è conveniente dal punto di vista del costo computazionale, risolvere il sistema lineare $\mathbf{Ax} = \mathbf{b}$ mediante il calcolo $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$.

- **Applicazione 4. Risoluzione di più sistemi lineari $\mathbf{A}\mathbf{x}_i = \mathbf{b}_i$**

Per la risoluzione di p sistemi lineari aventi tutti la stessa matrice dei coefficienti $\mathbf{A}$, un algoritmo efficiente consiste nel calcolare la fattorizzazione $\mathbf{PA} = \mathbf{LU}$ e procedere con la risoluzione di due sistemi triangolari per ciascun sistema assegnato.

Il costo di tale procedura è circa $n^3/3 + pn^2$ per n grande. Se p è molto più piccolo di n, il costo pn^2 dei $2p$ sistemi lineari triangolari è trascurabile rispetto al costo della fattorizzazione.

La risoluzione di ciascun sistema lineare mediante il metodo delle eliminazioni di Gauss avrebbe, invece, richiesto un costo all'incirca pari a $pn^3/3$.

Per esempio, per $p = 3$,

$$\begin{cases} \mathbf{A}\mathbf{x}_1 = \mathbf{b}_1 \\ \mathbf{A}\mathbf{x}_2 = \mathbf{b}_2 \\ \mathbf{A}\mathbf{x}_3 = \mathbf{b}_3 \end{cases}$$

si determina $\mathbf{PA} = \mathbf{LU}$ e quindi si calcola $\mathbf{x}_i$, $i = 1, 2, 3$, risolvendo i seguenti due sistemi triangolari:

$$\begin{cases} \mathbf{L}\mathbf{y} = \mathbf{P}\mathbf{b}_i & \Rightarrow \mathbf{y} \\ \mathbf{U}\mathbf{x}_i = \mathbf{y} & \Rightarrow \mathbf{x}_i \end{cases}$$

Fattorizzazione di Choleski e applicazioni

Per le matrici simmetriche e definite positive si può dimostrare il seguente teorema.

Teorema

Sia **A** una matrice **simmetrica e definita positiva**. Si dimostra che per essa esiste ed è unica la fattorizzazione

$$\mathbf{A} = \mathbf{R}^T \mathbf{R}$$

ove **R** è una matrice triangolare superiore con elementi positivi sulla diagonale principale.
Tale fattorizzazione è detta **fattorizzazione di Choleski**.

Il calcolo del fattore $\mathbf{R}$ si ottiene mediante un algoritmo il cui costo computazionale è essenzialmente $n^3/6$ per n grande.

Comando Matlab

`R = chol(A)` calcola il fattore `R` triangolare superiore della fattorizzazione di Choleski $\mathbf{A} = \mathbf{R}^T\mathbf{R}$, della matrice simmetrica e definita positiva `A`.

Di seguito vengono descritte alcune applicazioni della fattorizzazione di Choleski.

- **Applicazione 1. Risoluzione del sistema lineare $\mathbf{Ax} = \mathbf{b}$**

Per la risoluzione del sistema lineare $\mathbf{Ax} = \mathbf{b}$, con $\mathbf{A}$ simmetrica e definita positiva, nota la fattorizzazione di Choleski $\mathbf{A} = \mathbf{R}^T\mathbf{R}$, un algoritmo efficiente è il seguente:

$$\mathbf{Ax} = \mathbf{b} \Longrightarrow \mathbf{R}^T \underbrace{\mathbf{Rx}}_{\mathbf{y}} = \mathbf{b} \Longrightarrow \begin{cases} \mathbf{R}^T\mathbf{y} = \mathbf{b} & \Rightarrow \mathbf{y} \\ \mathbf{Rx} = \mathbf{y} & \Rightarrow \mathbf{x} \end{cases}$$

- **Applicazione 2. Calcolo dell'inversa di una matrice**

Per il calcolo dell'inversa della matrice simmetrica e definita positiva $\mathbf{A}$, un metodo efficiente e ottimale dal punto di vista del costo computazionale, è il seguente:

$$\mathbf{A} = \mathbf{R}^T\mathbf{R} \Longrightarrow \mathbf{A}^{-1} = (\mathbf{R}^T\mathbf{R})^{-1} = \mathbf{R}^{-1}(\mathbf{R}^T)^{-1}$$

Per ridurre il costo computazionale conviene scambiare l'operazione di trasposizione con quella di inversione e scrivere:

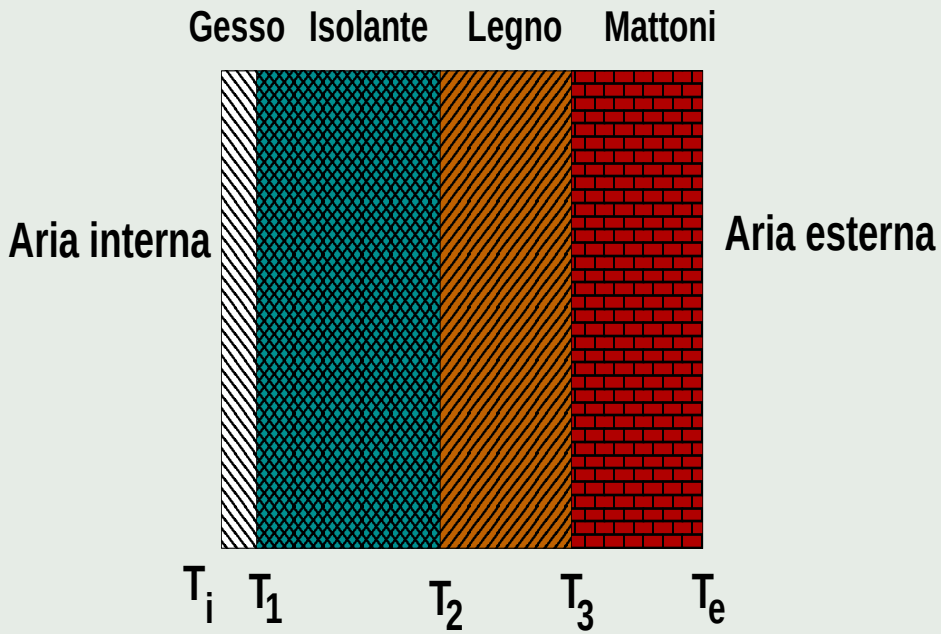
$$\mathbf{A}^{-1} = \mathbf{R}^{-1}(\mathbf{R}^{-1})^T$$

In questo modo occorre invertire soltanto il fattore triangolare $\mathbf{R}$.

Il costo computazionale del metodo è essenzialmente $2n^3/3$ per n grande.

Esempio. Trasmissione del calore.

Supponiamo di avere una parete formata da quattro strati: lo strato 1 di gesso (10 mm), lo strato 2 di fibra di vetro (125 mm), lo strato 3 di legno (60 mm) e lo strato 4 di mattoni (50 mm).



... continua esempio

Indicati con D lo spessore del materiale e con k la sua conduttività termica, la resistenza termica di una parete di $1m^2$ è data da $R = D/k$; per i materiali considerati, le resistenze termiche sono dunque $R_1 = 0.036^o\,C/watt$, $R_2 = 4.01^o\,C/watt$, $R_3 = 0.408^o\,C/watt$, $R_4 = 0.038^o\,C/watt$, rispettivamente.

Supponendo che la temperatura interna $T_i = 20^o\,C$ ed esterna $T_e = -10^o\,C$ restino costanti per un determinato periodo di tempo e che la superficie della parete sia $10m^2$, calcoliamo le temperature T_1, T_2 e T_3 tra i vari strati e l'energia termica in *watt*.

... continua esempio

Per calcolare le quantità richieste, si utilizza la seguente relazione

$$q = \frac{\Delta T}{R}$$

che, quando un materiale è sottoposto alla differenza di temperatura ΔT, correla la sua resistenza termica R al flusso q che lo attraversa.

Avendo supposto che la temperatura interna T_i e quella esterna T_e restino costanti, anche l'energia termica presente nei vari strati è costante. Quindi il flusso termico, attraverso i singoli strati, è lo stesso e si ha:

$$q = \frac{T_i - T_1}{R_1} = \frac{T_1 - T_2}{R_2} = \frac{T_2 - T_3}{R_3} = \frac{T_3 - T_e}{R_4}$$

... continua esempio

Esplicitando le tre equazioni, otteniamo il seguente sistema lineare

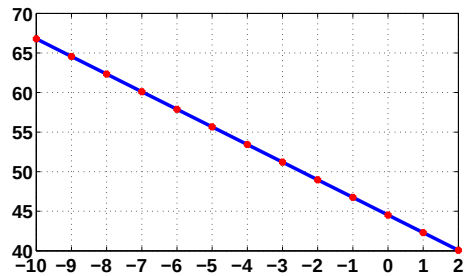
$$\begin{cases} (R_1 + R_2)T_1 - R_1 T_2 = T_i R_2 \\ R_3 T_1 - (R_2 + R_3)T_2 + R_2 T_3 = 0 \\ R_4 T_2 - (R_3 + R_4)T_3 = -T_e R_3 \end{cases}$$

la cui risoluzione fornisce $T_1 = 19.7596^o\,C$, $T_2 = -7.0214^o\,C$ e $T_3 = -9.7462^o\,C$. Usando la relazione $q = \Delta T / R$, abbiamo che il flusso termico per una parete di $10 m^2$ è pari a $66.785\,watt$.

... continua esempio

Calcoliamo ora l'energia termica al variare della temperatura esterna $T_e = -10^oC, -9^oC, ..., 1^oC, 2^oC$.

```
R = [0.036 4.01 0.408 0.038];
ti = 20;
te = [-10:1:2];
A = [R(1)+R(2) -R(1) 0;
   R(3) -(R(2)+R(3)) R(2);
   0 R(4) -(R(3)+R(4))];
[L,U,P] = lu(A);
for i = 1:length(te)
  b = [ti*R(2); 0 ; -te(i)*R(3)];
  y = L\(P*b);
  x = U\y;
  q(i) = (ti-x(1))/R(1)*10;
end
plot(te,q,'b',te,q,'r*','linewidth',3)
grid on
```



Teorema

Ogni matrice $\mathbf{A} \in \mathbb{R}^{m,n}$ è fattorizzabile nella forma

$$\mathbf{A} = \mathbf{QR}$$

con $\mathbf{Q} \in \mathbb{R}^{m,m}$ **ortogonale** ed $\mathbf{R} \in \mathbb{R}^{m,n}$ con elementi $r_{ij} = 0$ per $i > j$.

In particolare, per $m < n$ e $m > n$ si ha rispettivamente

$$\mathbf{R} = \begin{pmatrix} r_{11} & r_{12} & \dots & \dots & \dots & r_{1n} \\ 0 & r_{22} & \dots & \dots & \dots & r_{2n} \\ \vdots & \vdots & \ddots & & & \vdots \\ 0 & 0 & \dots & r_{mm} & \dots & r_{mn} \end{pmatrix}, \quad \mathbf{R} = \begin{pmatrix} r_{11} & r_{12} & \dots & r_{1n} \\ 0 & r_{22} & \dots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & r_{nn} \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{pmatrix}$$

I vettori colonna della matrice $\mathbf{Q}$ sono a due a due ortogonali.

La fattorizzazione QR di una matrice $\mathbf{A}$, in generale, **non è unica**.

Si può, tuttavia, dimostrare il seguente teorema.

Teorema

Sia $\mathbf{A} \in \mathbb{R}^{m,n}$, $m \geq n$, di **rango massimo** n. Allora il fattore $\mathbf{R}$ della fattorizzazione QR della matrice $\mathbf{A}$ si riscrive come

$$\mathbf{R} = \begin{pmatrix} \tilde{\mathbf{R}} \\ \mathbf{O} \end{pmatrix}$$

con $\mathbf{O}$ matrice nulla e $\tilde{\mathbf{R}}$ triangolare superiore **non singolare**.
Inoltre, $\mathbf{A}$ può essere scritta in un unico modo nella forma

$$\mathbf{A} = \tilde{\mathbf{Q}}\tilde{\mathbf{R}}$$

con $\tilde{\mathbf{Q}} \in \mathbb{R}^{m,n}$ avente vettori colonna ortonormali ed $\tilde{\mathbf{R}} \in \mathbb{R}^{n,n}$ triangolare superiore **non singolare con elementi diagonali positivi**.

I seguenti comandi Matlab consentono di calcolare una fattorizzazione QR di una assegnata matrice $\mathbf{A}$.

Comandi Matlab

- `[Q,R] = qr(A)`, data la matrice `A` di dimensioni $m \times n$, calcola un fattore ortogonale `Q` di dimensioni $m \times m$ e un fattore `R` di dimensioni $m \times n$ tali che $\mathbf{A} = \mathbf{QR}$.

Si descrivono ora alcune applicazioni della fattorizzazione $\mathbf{A} = \mathbf{QR}$.

- **Applicazione 1. Risoluzione di un sistema lineare determinato**

Per la risoluzione di un sistema lineare $\mathbf{Ax} = \mathbf{b}$, con $\mathbf{A} \in \mathbb{R}^{n,n}$ non singolare, noti i fattori $\mathbf{Q}$ e $\mathbf{R}$ della fattorizzazione $\mathbf{A} = \mathbf{QR}$, con $\mathbf{Q} \in \mathbb{R}^{n,n}$ ortogonale e $\mathbf{R} \in \mathbb{R}^{n,n}$ triangolare superiore e non singolare, si procede nel seguente modo:

$$\mathbf{Ax} = \mathbf{b} \Longrightarrow \mathbf{QRx} = \mathbf{b} \Longrightarrow \mathbf{Rx} = \mathbf{Q}^T\mathbf{b}$$

Si osservi che, se i fattori $\mathbf{Q}$ e $\mathbf{R}$ non sono noti, tale procedura non è conveniente dal punto di vista del costo computazionale, perché il calcolo dei suddetti fattori richiede un numero di operazioni aritmetiche superiore a quello richiesto dal calcolo dei fattori della fattorizzazione $\mathbf{PA} = \mathbf{LU}$.

- **Applicazione 2. Risoluzione di un sistema lineare sovradeterminato**

La fattorizzazione $\mathbf{A} = \mathbf{QR}$ si utilizza anche per la risoluzione del sistema lineare

$$\mathbf{Ax} = \mathbf{b}, \quad \mathbf{A} \in \mathbb{R}^{m,n}, \ \mathbf{x} \in \mathbb{R}^{n}, \ \mathbf{b} \in \mathbb{R}^{m}, \ m > n$$

Tale sistema si dice **sovradeterminato** in quanto il numero m delle equazioni lineari (*vincoli*) è superiore al numero n delle incognite (*gradi di libertà*).

Un sistema sovradeterminato può non ammettere soluzione in senso classico. In altre parole, indipendentemente dal rango, massimo o non, della matrice $\mathbf{A}$, può non esistere un vettore $\mathbf{x}$ che soddisfi contemporaneamente tutte le equazioni del sistema $\mathbf{Ax} = \mathbf{b}$ e per il quale, quindi, $\mathbf{Ax} - \mathbf{b}$ è il vettore identicamente nullo.

Se il sistema non ammette soluzione in senso classico, allora si ricercano i vettori $\mathbf{x}$ per i quali il **vettore residuo** $\mathbf{Ax} - \mathbf{b}$ è *piccolo*.

Pertanto, si fissa una norma vettoriale $||\cdot||$ (al fine di misurare la grandezza del suddetto vettore) e si ricercano i vettori $\mathbf{x}$ che minimizzano la quantità $||\mathbf{Ay} - \mathbf{b}||$ con $\mathbf{y} \in \mathbb{R}^n$.

Definizione

Si dice allora che il vettore $\mathbf{x}$ è **soluzione del sistema sovradeterminato** $\mathbf{Ax} = \mathbf{b}$ rispetto alla norma $||\cdot||$ se

$$\min_{\mathbf{y} \in \mathbb{R}^n} ||\mathbf{Ay} - \mathbf{b}|| = ||\mathbf{Ax} - \mathbf{b}||$$

Si osservi che se $||\mathbf{Ax} - \mathbf{b}|| = 0$, allora $\mathbf{Ax} - \mathbf{b} = \mathbf{0}$ e si ritrova la soluzione classica $\mathbf{x}$ del sistema $\mathbf{Ax} = \mathbf{b}$.

Se si sceglie la norma euclidea, il problema della risoluzione del sistema sovradeterminato $\mathbf{Ax} = \mathbf{b}$ viene dunque trasformato nel seguente problema di minimo $\min_{\mathbf{y}\in\mathbb{R}^n} ||\mathbf{Ay} - \mathbf{b}||_2$.

Definizione

Il problema

$$\min_{\mathbf{y}\in\mathbb{R}^n} ||\mathbf{Ay} - \mathbf{b}||_2,$$

si definisce **problema dei minimi quadrati** e un vettore $\mathbf{x}$ tale che

$$\min_{\mathbf{y}\in\mathbb{R}^n} ||\mathbf{Ay} - \mathbf{b}||_2 = ||\mathbf{Ax} - \mathbf{b}||_2$$

si dice **soluzione nel senso dei minimi quadrati**.

Il teorema che segue caratterizza l'insieme X dei vettori soluzione $\mathbf{x} \in \mathbb{R}^n$ del problema dei minimi quadrati.

Teorema

Il problema dei minimi quadrati ammette sempre soluzione, ossia l'insieme X dei vettori soluzione è non vuoto. Inoltre,

- $\mathbf{x} \in X$ se e solo se

$$\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b}$$

  Tale sistema viene detto **sistema delle equazioni normali** o **sistema normale**.

- L'insieme X si riduce ad un solo elemento $\mathbf{x}^\star$ se e solo se la matrice $\mathbf{A}$ ha rango massimo.
- Esiste uno e un solo vettore $\mathbf{x}^\star \in X$ tale che

$$||\mathbf{x}^\star||_2 = \min_{\mathbf{x} \in X} ||\mathbf{x}||_2$$

  Il vettore $\mathbf{x}^\star$ è detto **soluzione di minima norma**.

Per la risoluzione di un sistema sovradeterminato, ovvero per la risoluzione del problema dei minimi quadrati, si distinguono pertanto due casi in base al rango della matrice $\mathbf{A}$.

1) A ha rango massimo

In tal caso il problema dei minimi quadrati ammette un'unica soluzione $\mathbf{x}^\star$.

Tale soluzione può essere calcolata risolvendo il sistema delle equazioni normali.
Tenendo conto che $\mathbf{A}$ ha rango massimo, la matrice $\mathbf{A}^T\mathbf{A}$ è simmetrica e definita positiva, e quindi si può utilizzare la fattorizzazione di Choleski per risolvere il sistema normale.

Tale procedura non è tuttavia numericamente stabile, in quanto il sistema delle equazioni normali può essere mal condizionato anche quando il problema iniziale non lo è.

Un metodo stabile ed efficiente per calcolare la soluzione del problema dei minimi quadrati si basa sulla fattorizzazione $\mathbf{A} = \mathbf{QR}$.

Per la descrizione del suddetto metodo, si osservi preliminarmente che, poichè si vuole minimizzare una quantità non negativa, non è restrittivo considerare il quadrato della norma del residuo $||\mathbf{Ay} - \mathbf{b}||_2^2$.

Tenendo conto che $\mathbf{Q}$ è ortogonale e

$$||\mathbf{Q}^T\mathbf{z}||_2^2 = (\mathbf{Q}^T\mathbf{z})^T\mathbf{Q}^T\mathbf{z} = \mathbf{z}^T\mathbf{Q}\mathbf{Q}^T\mathbf{z} = ||\mathbf{z}||_2^2,$$

si ha

$$||\mathbf{Ay} - \mathbf{b}||_2^2 = ||\mathbf{Q}^T(\mathbf{Ay} - \mathbf{b})||_2^2 = ||\mathbf{Ry} - \mathbf{Q}^T\mathbf{b}||_2^2 = ||\mathbf{Ry} - \mathbf{c}||_2^2$$

ove $\mathbf{c} := \mathbf{Q}^T\mathbf{b}$.
Pertanto $||\mathbf{Ay} - \mathbf{b}||_2^2$ *è minimo se, e solo se,* $||\mathbf{Ry} - \mathbf{c}||_2^2$ *è minimo.*

Per il teorema precedentemente enunciato, poiché $\mathbf{A}$ ha rango massimo,

$$\mathbf{R} = \begin{pmatrix} \tilde{\mathbf{R}} \\ \mathbf{O} \end{pmatrix}, \ \tilde{\mathbf{R}} \in \mathbb{R}^{n,n}, \ \mathbf{O} \in \mathbb{R}^{m-n,n}$$

con $\tilde{\mathbf{R}}$ triangolare superiore e non singolare e $\mathbf{O}$ matrice nulla.
Partizioniamo quindi $\mathbf{c}$ nella seguente forma

$$\mathbf{c} = \begin{pmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{pmatrix}, \ \mathbf{c}_1 \in \mathbb{R}^n, \ \mathbf{c}_2 \in \mathbb{R}^{m-n},$$

e riscriviamo $||\mathbf{Ry} - \mathbf{c}||_2^2$ come

$$||\mathbf{Ry} - \mathbf{c}||_2^2 = \left|\left| \begin{pmatrix} \tilde{\mathbf{R}}\mathbf{y} \\ \mathbf{O} \end{pmatrix} - \begin{pmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{pmatrix} \right|\right|_2^2 = \left|\left| \begin{pmatrix} \tilde{\mathbf{R}}\mathbf{y} - \mathbf{c}_1 \\ -\mathbf{c}_2 \end{pmatrix} \right|\right|_2^2 = ||\tilde{\mathbf{R}}\mathbf{y} - \mathbf{c}_1||_2^2 + ||\mathbf{c}_2||_2^2$$

Osserviamo che, in quest'ultima uguaglianza, solo il primo addendo dipende da $\mathbf{y}$.
Pertanto $||\mathbf{Ay} - \mathbf{b}||_2^2$ è minimo se, e solo se, $||\tilde{\mathbf{R}}\mathbf{y} - \mathbf{c}_1||_2^2$ è minimo.

Essendo $\tilde{\mathbf{R}}$ non singolare, il sistema lineare $\tilde{\mathbf{R}}\mathbf{y} = \mathbf{c}_1$ ammette una e una sola soluzione $\mathbf{x}^\star$, per la quale si ha $||\tilde{\mathbf{R}}\mathbf{x}^\star - \mathbf{c}_1||_2^2 = 0$.

Ne consegue che $\mathbf{x}^\star$ è tale che

$$\min_{\mathbf{y} \in R^n} ||\mathbf{A}\mathbf{y} - \mathbf{b}||_2 = ||\mathbf{A}\mathbf{x}^\star - \mathbf{b}||_2 = ||\mathbf{c}_2||_2$$

Osserviamo che se $\mathbf{c}_2 = \mathbf{0}$, allora $\mathbf{x}^\star$ è soluzione nel senso classico; altrimenti, $\mathbf{x}^\star$ è soluzione nel senso dei minimi quadrati. La norma $||\mathbf{c}_2||_2$ fornisce la misura del residuo.

Il costo computazionale di tale metodo è pari a quello della fattorizzazione $\mathbf{A} = \mathbf{QR}$.

2) A non ha rango massimo

Osserviamo che se $\mathbf{A}$ non ha rango massimo, allora il problema dei minimi quadrati ammette infinite soluzioni. Infatti, in tal caso, se $\mathbf{x}$ è una soluzione del problema $\min_{\mathbf{y}\in\mathbb{R}^n} ||\mathbf{Ay} - \mathbf{b}||_2$, allora anche $\mathbf{x} + \mathbf{z}$, con $\mathbf{z} \neq \mathbf{0}$ e tale che $\mathbf{Az} = \mathbf{0}$, è soluzione del medesimo problema.

Tuttavia il teorema, precedentemente enunciato, assicura che fra tutti i vettori $\mathbf{x} \in X$ che minimizzano $||\mathbf{Ay} - \mathbf{b}||_2$, ne esiste solo uno che ha norma euclidea minima.

Si determina allora come soluzione del sistema lineare sovradeterminato $\mathbf{Ax} = \mathbf{b}$, il vettore $\mathbf{x}^\star$ tale che

$$||\mathbf{Ax}^\star - \mathbf{b}||_2 = \min_{\mathbf{y} \in \mathbb{R}^n} ||\mathbf{Ay} - \mathbf{b}||_2 \quad \text{e} \quad ||\mathbf{x}^\star||_2 = \min_{\mathbf{x} \in X} ||\mathbf{x}||_2$$

Osserviamo che la soluzione del problema $\min_{\mathbf{y} \in \mathbb{R}^n} ||\mathbf{Ay} - \mathbf{b}||_2$ nell'ipotesi in cui $\mathbf{A}$ abbia rango massimo, essendo unica, ha necessariamente norma euclidea minima.

Lo strumento che consente di calcolare la soluzione del problema sopra è la decomposizione ai valori singolari di $\mathbf{A}$, che verrà in seguito descritta.

Osservazioni

- Anche nel caso di sistemi sottodeterminati $\mathbf{Ax} = \mathbf{b}$, per i quali il numero delle incognite è superiore al numero delle equazioni, tra le infinite soluzioni, si considera quella di norma euclidea minima. Essa si determina tramite la decomposizione ai valori singolari.
- Come vedremo, la fattorizzazione $\mathbf{A} = \mathbf{QR}$ verrà utilizzata anche per il calcolo degli autovalori della matrice $\mathbf{A}$.

Comando Matlab

`x = A\b`, nel caso di un sistema lineare sovradeterminato $\mathbf{Ax} = \mathbf{b}$, con `A` matrice rettangolare di dimensioni $m \times n$ con $m > n$ e di rango massimo, fornisce (utilizzando la procedura precedentemente descritta) la soluzione `x` nel senso dei minimi quadrati del sistema $\mathbf{Ax} = \mathbf{b}$.

Esempio. Biomeccanica

Nella seguente tabella vengono riportati i risultati di un esperimento (P. Komarek, Biomechanics of Clinical Aspects of Biomedicine, 1993) eseguito per individuare il legame tra lo sforzo e la relativa deformazione di un campione di tessuto biologico (un disco intervertebrale).

Test	Sforzo x	Deformazione y	Test	Sforzo x	Deformazione y
1	0.00	0.00	5	0.31	0.23
2	0.06	0.08	6	0.47	0.25
3	0.14	0.14	7	0.60	0.28
4	0.25	0.20	8	0.70	0.29

A partire dai suddetti dati, stimiamo la deformazione corrispondente allo sforzo $x = 0.9MPa$ ($MPa = 100N/cm^2$, megapascal), approssimando i dati assegnati mediante un polinomio di grado 1.

... continua esempio

Sia allora $p(x) = c_1 x + c_2$ un polinomio di grado 1 e imponiamo le condizioni $p(x_i) \equiv c_1 x_i + c_2 = y_i$; usando i dati della tabella, otteniamo un sistema lineare sovradeterminato $\mathbf{Ac} = \mathbf{b}$,

$$\begin{pmatrix} 0 & 1 \\ 0.06 & 1 \\ 0.14 & 1 \\ 0.25 & 1 \\ 0.31 & 1 \\ 0.47 & 1 \\ 0.60 & 1 \\ 0.70 & 1 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0.08 \\ 0.14 \\ 0.20 \\ 0.23 \\ 0.25 \\ 0.28 \\ 0.29 \end{pmatrix}$$

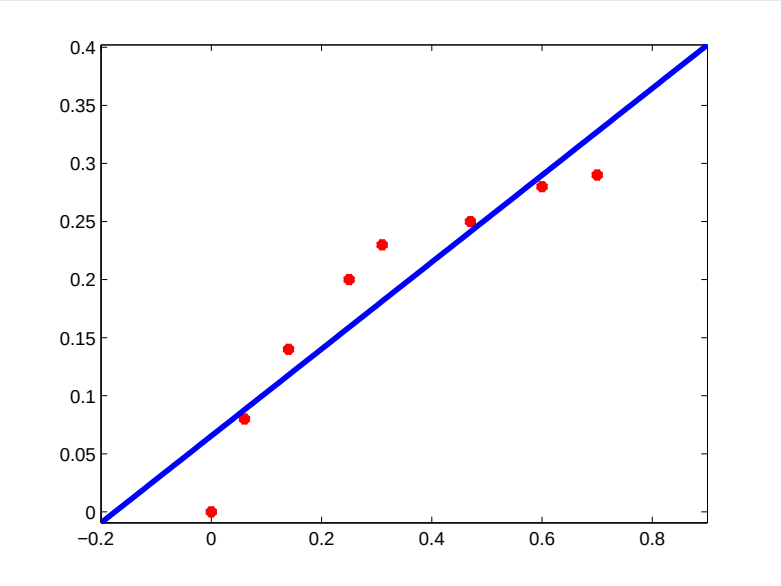
di 8 equazioni nel vettore incognito $\mathbf{c} = (c_1, c_2)^T$.

... continua esempio

Per calcolare la soluzione $\mathbf{c}^\star$ nel senso dei minimi quadrati scriviamo le seguenti istruzioni Matlab:

```
x = [0 0.06 0.14 0.25 0.31 0.47 0.60 0.70];
y = [0 0.08 0.14 0.20 0.23 0.25 0.28 0.29];
A = [x' ones(8,1)];
b = y';
[Q,R] = qr(A);
c = Q'*b;
c1 = c(1:2);
c_star = R(1:2,1:2)\c1;
z = [-.2,.9];
p = polyval(c_star,z);
plot(x,y,'r*',z,p,'linewidth',3)
axis([-0.2 0.9 min(p) max(p)])
pr = polyval(c_star,0.9)
```

... continua esempio



La retta rappresentata graficamente è detta **retta di regressione**.
La deformazione corrispondente allo sforzo $x = 0.9MPa$ è $y \approx 0.4021cm$.

Osservazioni

- La precedente procedura è automaticamente implementata nel comando `\`: l'istruzione `c_star = A\b` produce infatti il medesimo vettore $\mathbf{c}^\star$ calcolato in precedenza.
- Il vettore $\mathbf{c}^\star$ si può ottenere anche mediante l'istruzione `c_star = polyfit(x,y,1)`. I coefficienti del polinomio di grado 1, calcolati con `polyfit`, coincidono con le componenti del vettore $\mathbf{c}^\star$ che minimizza la quantità $||\mathbf{Ac} - \mathbf{b}||_2$.