Sistemi lineari

Lucia Gastaldi

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

Indice

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** $\backslash$ e **slash** $/$.

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.

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.

Come risolvere un sistema lineare con MATLAB

La risoluzione del sistema lineare si ottiene usando i simboli di divisione: **backslash** $\backslash$ 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.

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

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} \ell_{ij} x_j}{\ell_{ii}} \qquad \text{per } i = 2, \dots, n$$

Metodo di sostituzione all'indietro

U matrice triangolare superiore.

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

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

Algoritmo di eliminazione di Gauss

```
for k = 1, ..., n − 1
   for i = k + 1, ..., n
      m_ik = a_ik^(k) / a_kk^(k)
      for j = k + 1, ..., 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
```

$$
\begin{array}{l}
\textbf{for } k = 1, \ldots, n-1 \\
\quad \textbf{for } i = k+1, \ldots, n \\
\qquad m_{ik} = \dfrac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \\
\qquad \textbf{for } j = k+1, \ldots, n \\
\qquad\quad a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)} \\
\qquad \textbf{end} \\
\qquad b_i^{(k+1)} = b_i^{(k)} - m_{ik} b_k^{(k)} \\
\quad \textbf{end} \\
\textbf{end}
\end{array}
$$

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

cerco più piccolo p tale che $|a_{pk}^{(k)}| = \max_{k \le i \le n} |a_{ik}^{(k)}|$

scambio la riga k con la riga p

for $i = k+1, \ldots, n$

$m_{ik} = \dfrac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$

for $j = k+1, \ldots, n$

$a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}$

end

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

end

end

Le funzioni MATLAB per la fattorizzazione

Funzione	**Significato**
`lu`	Fattorizzazione $PA = LU$.
`chol`	Fattorizzazione di Cholesky $A = R^\top R$ 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 $\mathtt{L1} = \mathtt{P}^{-1}\mathtt{L}$

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`.

Propagazione degli errori

Esercizio 1

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

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

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

- usare la function `miaLU` per calcolare la fattorizzazione LU di A;
- calcolare la differenza $A - LU$;
- usare la fattorizzazione ottenuta per risolvere il sistema lineare $Ax = b$, la cui soluzione esatta è $x = (1, -1, 1)^\top$.
  Date le matrici `L` e `U` la soluzione si ottiene con i due comandi

  ```
  y=L\b        x=U\y
  ```

- 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:

$$\begin{aligned} a_{ii} &= 1, \quad \text{per } i = 1, \dots, 25 \\ a_{1j} &= 1, \quad \text{per } j = 2, \dots, 25 \\ a_{i1} &= 1, \quad \text{per } i = 2, \dots, 25 \end{aligned}$$

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

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

Le norme più usate sono:

$$\|x\|_1 = \sum_{i=1}^{n} |x_i| \qquad \texttt{norm(x,1)}$$

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

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

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:

$$\|\mathbb{I}\| = 1 \text{ per } \mathbb{I} \text{ matrice identità}$$
$$\|Ax\| \leq \|A\| \, \|x\|$$
$$\|AB\| \leq \|A\| \, \|B\|$$

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

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

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

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

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 δA e δb perturbazioni di A e di b rispettivamente e sia $x + \delta x$ la soluzion 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 ∞). 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.

Matrice mal condizionata

Esercizio 3

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

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

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

- Calcolare con MATLAB la fattorizzazione LU con pivoting e 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`.

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`.

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

Equazioni differenziali con valori ai limiti

Si consideri l'equazione differenziale

$$\begin{array}{l} -\mu u''(x) + \sigma(x)u(x) = f(x) \quad \text{per } x \in (a,b) \\ u(a) = \alpha, \quad u(b) = \beta. \end{array}$$

Suddividiamo l'intervallo $[a, b]$ in $N + 1$ parti uguali e poniamo $h = (b - a)/(N + 1)$. Poniamo poi $x_i = a + ih$.
Possiamo approssimare la derivata seconda con la seguente differenza finita del secondo ordine

$$u''(x_i) \approx \frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2} \quad \text{per } i = 1, \ldots, N.$$

Indicando con u_i^h il valore approssimato di $u(x_i)$, si ottiene il seguente sistema lineare:

$$\begin{array}{l} -\mu \dfrac{u_{i+1}^h - 2u_i^h + u_{i-1}^h}{h^2} + \sigma(x_i)u_i^h = f(x_i) \quad \text{per } i = 1, \ldots, N \\ u_0^h = \alpha \qquad u_{N+1}^h = \beta \end{array}$$

Equazioni differenziali con valori ai limiti II

Quindi si ricava la soluzione u_i^h per $i = 1, \ldots, N$ risolvendo il sistema

$$Au^h = F$$

essendo A la matrice tridiagonale che ha i seguenti elementi

$$a_{ii} = 2\mu/h^2 + \sigma(x_i), \quad a_{ii-1} = a_{ii+1} = -\mu/h^2$$

e il termine noto è dato da

$$\begin{aligned} F_1 &= f(x_1) + \mu\alpha/h^2, \\ F_i &= f(x_i) \text{ per } i = 2, \ldots, N-1, \\ F_N &= f(x_N) + \mu\beta/h^2. \end{aligned}$$

Stima dell'errore

$$\max_{1 \le i \le N} |u(x_i) - u_i^h| \le Ch^2 \max_{a \le x \le b} |u^{(4)}(x)|$$

Costruzione della matrice

Per costruire la matrice del sistema lineare che si ottiene con le differenze finite osserviamo che si può scrivere $A = K + M$ essendo

$$K = \frac{\mu}{h^2} \begin{bmatrix} 2 & -1 & \dots & \dots & \dots & \dots & 0 \\ -1 & 2 & -1 & \dots & \dots & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & -1 & 2 & -1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & -1 & 2 & -1 \\ 0 & \dots & \dots & \dots & \dots & -1 & 2 \end{bmatrix}$$

$$M = \begin{bmatrix} \sigma(x_1) & 0 & \dots & \dots & 0 \\ 0 & \sigma(x_2) & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & 0 & \sigma(x_N) \end{bmatrix}$$

Costruzione del termine noto

Il termine noto può essere ottenuto come la somma di due vettori $F = \tilde{F} + bc$: $\tilde{F}$ tiene conto del dato f sull'intervallo, mentre bc è relativo alle condizioni al bordo.

$$\tilde{F} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \dots \\ \dots \\ f(x_N) \end{bmatrix} \qquad bc = \begin{bmatrix} \mu\alpha/h^2 \\ 0 \\ \dots \\ 0 \\ \mu\beta/h^2 \end{bmatrix}$$

Dal punto di vista pratico non è necessario costruire il vettore bc, ma basta sommare alla prima e ultima componente di F il contributo derivante dalle condizioni al bordo.

Soluzione dell'equazione differenziale

Scrivere la function eqlim per calcolare la soluzione dell'equazione differenziale con valori ai limiti

$$\begin{array}{l} -\mu u''(x) + \sigma(x)u(x) = f(x) \quad \text{per } x \in [a, b] \\ u(a) = \alpha, \quad u(b) = \beta. \end{array}$$

mediante il seguente comando:

```
[x,u]=eqlim(f,sigma,mu,a,b,alfa,beta,N)
```

Input

`f,sigma`	nome delle function_handle che contengono l'espressione analitica di f e σ
`mu`	coefficiente davanti alla derivata seconda
`a,b`	estremi dell'intervallo
`alfa,beta`	valori ai limiti
`N`	numero di punti in cui si calcola la soluzione

Output

`x`	ascisse dei punti in cui si calcola la soluzione
`u`	valori della soluzione

Function `eqlim`

La function `eqlim` si compone di 5 passi:

- assegnazione della griglia di calcolo: calcolo h e i punti $x_i = a + ih$ per $i = 1, \ldots, N$;
- costruzione della matrice A come suggerito usando i comandi `ones` e `diag`;
- costruzione del termine noto (attenzione al nome);
- risoluzione del sistema lineare;
- organizzazione dell'output che tenga conto delle condizioni al bordo.

Oss Si possono completare i vettori `x` e `u` con i valori negli estremi dell'intervallo. Per fare ciò si devono aggiungere le rispettive componenti all'inizio ed alla fine del vettore, con i comandi

- `x=[a,x,b]` se `x` è un vettore riga;
- `u=[alfa;u;beta]` se `u` è un vettore colonna.

zeros e ones

`ones` e `zeros`

I comandi `ones` e `zeros` permettono di costruire un array che ha componenti tutte uguali ad 1 oppure a 0.
`b=ones(10,1)` costruisce il vettore colonna b di 10 componenti tutte uguali a 1.
`c=zeros(3,3)` costruisce una matrice 3×3 di elementi tutti uguali a 0.

DIAG

Il comando **diag** ha due funzioni diverse:

Costruzione di una matrice con una sola diagonale non nulla

Sia `V` un vettore di `N` componenti.
`diag(V,K)` è una matrice quadrata di ordine `N+ABS(K)` che ha gli elementi di `V` sulla diagonale `K`-esima.

se `K`$=0$	è la diagonale principale
se `K`>0	si trova sopra la diagonale principale
se `K`<0	si trova sotto la diagonale principale

Estrazione di una diagonale da una matrice

Sia `A` una matrice di dimensione $n \times n$ il comando

```
d=diag(A,k)
```

fornisce il vettore
$d = [a_{1,1+k}, \ldots, a_{n-k,n}]^t$ se $k \geq 0$
$d = [a_{1+k,1}, \ldots, a_{n,n-k}]^t$ se $k < 0$.

Esercizio 1

Risolvere l'equazione differenziale

$$\begin{aligned}&-u''(x) + \sin x \, u(x) = (1 - \cos^2 x - 2\cos x)e^x \quad \text{per } x \in [0, \pi]\\ &u(0) = u(\pi) = 0.\end{aligned}$$

Confrontare i risultati ottenuti con la soluzione esatta $u(x) = \sin x \, e^x$.

Scrivere un file di tipo script che:

- assegna i dati, in particolare le funzioni f e σ di tipo `@`;
- calcola la soluzione usando la function `eqlim`;
- plotta la soluzione discreta insieme a quella continua;
- calcola l'errore

$$E = \max_{1 \le i \le N} |u(x_i) - u_i^h|.$$