Sistemi lineari

Lucia Gastaldi

Dipartimento di Matematica,
http://dm.ing.unibs.it/gastaldi/

1 febbraio 2008

Outline

Ax=b

Tre casi possibili:

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

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

$x = A\backslash b$ Indica la soluzione del sistema $Ax = b$, x e b vettori colonna.

$x = b/A$ Indica la soluzione del sistema $xA = b$, 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.
- Sistemi rettangolari sovradeterminati.
- Sistemi rettangolari sottodeterminati.

Risoluzione di sistemi triangolari

Metodo di sostituzione in avanti

L matrice triangolare inferiore.

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

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

Metodo di sostituzione all'indietro

U matrice triangolare superiore.

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

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

Algoritmo di eliminazione di Gauss

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

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

    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

Fattorizzazione LU

Teorema

Costruiamo le seguenti matrici:

$$L = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ m_{21} & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ m_{n1} & \cdots & m_{nn-1} & 1 \end{pmatrix} \quad U = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & a_{nn}^{(n)} \end{pmatrix}.$$

Allora

$$LU = A.$$

Propagazione degli errori

Esercizio

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: $a = 1$, $a = 0$, $a = 0.5\,10^{-15}$,

- calcolare la fattorizzazione LU di A mediante la function `lunopiv` che si utilizza mediante il seguente comando

```
>> [L,U]=lunopiv(A);
```

- calcolare la differenza $A - LU$;
- nel caso in cui la matrice sia non singolare, usare la fattorizzazione ottenuta per risolvere il sistema lineare $Ax = b$, la cui soluzione esatta è $x = (1, -1, 1)^T$;
- confrontare il risultato con quello che si ottiene con il comando `x=A\b`.

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

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

    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 $A = LL^T$.
`qr`	Fattorizzazione $A = QR$.
`qz`	Fattorizzazione QZ.
`eig`	Decomposizione spettrale (autovalori, autovettori).
`svd`	Decomposizione in valori singolari $A = U\Sigma V^T$.
`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 `LU=PA`.
- `[L,U]=lu(A)`:
  fornisce le matrici `L` e `U` in modo che `LU=A`.

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 `lunopiv`.

Propagazione degli errori

Esercizio

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: $a = 1$, $a = 0$, $a = 0.5\,10^{-15}$,

- calcolare la fattorizzazione LU mediante di A mediante la function `lu`;
- calcolare la differenza $A - LU$;
- nel caso in cui la matrice sia non singolare, risolvere il sistema lineare $Ax = b$, la cui soluzione esatta è $x = (1, -1, 1)^T$;
- confrontare i risultati con quelli ottenuti precedentemente usando la function `lunopiv`.

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:

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

Esercizio

Esercizio

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 $A\hat{x}_i = b + r_i$, $i = 1, 2$.
- Calcolare l'errore relativo commesso, e confrontarlo con la perturbazione relativa del termine noto.
- Verificare che il risultato ottenuto soddisfa la stima teorica.

Esempio di matrice mal condizionata

Esercizio

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 $x = (1, \ldots, 1)^T$.

- Calcolare con MATLAB la fattorizzazione LU con pivoting e risolvere il sistema al variare di n. Sia $\hat{x}$ la soluzione calcolata.
- Riportare l'errore relativo $E = \|x - \hat{x}\|/\|x\|$ in un grafico in scala semilogaritmica per diversi valori di n.
- Riportare sullo stesso grafico anche la norma della matrice $R = LU - PA$ e del residuo $\|b - A\hat{x}\|/\|b\|$.
- Per calcolare le norme usare il comando `norm`.

Traccia dell'esercizio

Per ogni `n=2:15` (`for n=2:15`)

- `A=hilb(n)` genera 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(n)=norm(x-xapp)/norm(x)` calcola l'errore relativo.
- `r=b-A*xapp` calcola il residuo.
- `res(n)=norm(r)/norm(b)` calcola la norma del residuo rapportata alla norma del termine noto.
- `K(n)=cond(A)` calcola il numero di condizionamento di `A`.
- `[L,U,P]=lu(A)` genera la fattorizzazione di A.
- `RES(n)=norm(L*U-P*A)` fornisce il residuo della fattorizzazione.

Alla fine del ciclo riportare i risultati ottenuti in un grafico in scala bilogaritmica come segue:

```
N=2:15;  semilogy(N,err,N,res,N,K,N,RES)
```

Altre funzioni di MATLAB per l'algebra lineare

Funzione	**Significato**
`hess`	Forma di Hessenberg.
`orth`	Base ortonormale dello spazio immagine di A.
`null`	Base ortonormale del nucleo della matrice A.
`subspace`	Angolo tra vettori o sottospazi.
`planerot`	Matrice rotazione nel piano.
`det`	Determinante.
`rank`	Rango.
`inv`	Inversa.
`pinv`	Pseudoinversa di Moore-penrose.
`cond`	Numero di condizionamento (norma Euclidea).
`condest`	Stima numero di condizionamento (norma 1).

Equazioni differenziali con valori ai limiti

Si consideri l'equazione differenziale

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

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 il valore approssimato di $u(x_i)$, si ottiene il seguente sistema lineare:

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

Equazioni differenziali con valori ai limiti II

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

$$Au = b$$

essendo A la matrice tridiagonale che ha i seguenti elementi

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

e il termine noto è dato da

$$\begin{aligned} b_1 &= f(x_1) + \alpha/h^2, \\ b_i &= f(x_i) \text{ per } i = 2, \ldots, N-1, \\ b_N &= f(x_N) + \beta/h^2. \end{aligned}$$

Soluzione dell'equazione differenziale

Function `eqlim`

La function `eqlim` calcola la soluzione dell'equazione differenziale con valori ai limiti

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

mediante il seguente comando:

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

Input

`f,sigma`	nome delle function che contengono l'espressione analitica di
`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` deve eseguire le seguenti operazioni:

- calcola h;
- calcola `x` vettore dei punti della mesh $x_i = a + ih$ per $i = 1, \ldots, N$;
- calcola i valori di f e σ nei punti x_i (`tn=f(x),s=sigma(x)`);
- costruisce la matrice tridiagonale A usando i comandi `diag`, `ones` e il vettore `s`;
- costruisce il termine noto aggiungendo a `tn` i valori ai limiti nella prima e ultima componente;
- risolve il sistema lineare.

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.

Esercizio

Esercizio

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 mediante dei comandi di tipo `inline`;
- calcola la soluzione usando la function `eqlim`;
- plotta la soluzione discreta insieme a quella continua,

Convergenza del metodo delle differenze finite

Esercizio

Dato `N=[10 20 40 80 160 320]`, per ciascun valore di N calcolare la soluzione dell'equazione differenziale dell'esercizio precedente, riportare la soluzione in un grafico insieme alla soluzione esatta e valutare l'errore relativo. Riportare gli errori in funzione di `N` in un grafico in scala bilogaritmica.

Traccia

- Assegnare i dati: f, σ, a, b, α, β.
- Assegnare la soluzione esatta.
- Assegnare `N=[10 20 40 80 160 320]`.
- Per ogni valore di `N`: `for i=1:length(N)`
  - Calcolare la soluzione `u` e la soluzione esatta `sol`.
  - Fare il grafico: `plot(x,u,x,sol)`.
  - Calcolare l'errore: `E(i)=norm(u-sol)/norm(sol)`.
- Riportare gli errori in funzione di `N` in un grafico in scala bilogaritmica: `loglog(N,E)`.

Calcolo dell'ordine di convergenza

Supponiamo che la stima dell'errore per un certo metodo sia

$$E(h) \approx Ch^p \quad \text{essendo } h = (b-a)/(N+1).$$

La relazione dipende quindi da due quantità incognite C e p.
Per calcolare il valore di p, valutiamo l'espressione dell'errore per due diversi valori di h, h_1 e h_2:

$$E(h_1) \approx Ch_1^p, \quad E(h_2) \approx Ch_2^p$$

Dividiamo la prima relazione per la seconda, per eliminare C:

$$\frac{E(h_1)}{E(h_2)} \approx \left(\frac{h_1}{h_2}\right)^p;$$

p si ottiene prendendo i logaritmi ad entrambi i membri:

$$\log \frac{E(h_1)}{E(h_2)} \approx p \log \frac{h_1}{h_2}$$

da cui

$$p = \frac{\log E(h_1) - \log E(h_2)}{\log h_1 - \log h_2}$$

Rappresentazione dell'ordine di convergenza

Supponiamo che la stima dell'errore per un certo metodo sia

$$E(h) \approx Ch^p \quad \text{essendo } h = (b-a)/(N+1).$$

Per rappresentare il valore di p, calcoliamo il logaritmo ad entrambi i membri, ottenendo:

$$\log E(h) \approx \log C + p \log h.$$

Quindi il grafico di $\log E(h)$ in funzione di $\log h$ è una retta con coefficiente angolare pari a p.
Poiché si ha che $\log h = \log(b-a) - \log(N+1)$, il grafico dell'errore in funzione di N, è una retta con coefficiente angolare uguale a $-p$.

`loglog`

`loglog(N,E)` produce il grafico di $\log E(h)$ in funzione di $\log N$.

Per verificare il valore di p, confrontare l'andamento di $\log E(h)$ con quello di $p \log h$ con il comando `loglog(N,E,N,1./N.^p)`.

Function di Matlab per la costruzione di matrici

diag

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

Equazione di Laplace

Problema

Sia $\Omega =]0, a[\times]0, b[$. Dati $f : \Omega \to \mathbb{R}$ e $g : \overline{\Omega} \to \mathbb{R}$, si consideri il seguente problema

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega \\ u &= g \qquad \text{su } \partial\Omega \end{aligned}$$

Dati N_x e N_y, poniamo $h_x = \dfrac{a}{N_x + 1}$ e $h_y = \dfrac{b}{N_y + 1}$.

Notazioni

$$\begin{aligned} &x_i = ih_x, \ \text{per } i = 0, 1, \cdots, N_x + 1, \\ &y_j = jh_y, \ \text{per } j = 0, 1, \cdots, N_y + 1, \\ &\mathbf{u}_{i,j} \approx u(x_i, y_j), \ \text{per } i = 0, 1, \cdots, N_x + 1, \ j = 0, 1, \cdots, N_y + 1. \end{aligned}$$

Differenze finite

Discretizzazione delle derivate seconde

$$u_{xx}(x,y) \approx \frac{u(x-h_x,y) - 2u(x,y) + u(x+h_x,y)}{h_x^2}$$
$$u_{yy}(x,y) \approx \frac{u(x,y-h_y) - 2u(x,y) + u(x,y+h_y)}{h_y^2}$$

Si ottengono $N_x \times N_y$ equazioni (per $i = 1, \cdots, N_x$ $j = 1, \cdots, N_y$)

$$-\frac{\mathbf{u}_{i-1,j} - 2\mathbf{u}_{i,j} + \mathbf{u}_{i+1,j}}{h_x^2} - \frac{\mathbf{u}_{i,j-1} - 2\mathbf{u}_{i,j} + \mathbf{u}_{i,j+1}}{h_y^2} = f(x_i, y_j),$$

alle quali si aggiungono le condizioni al bordo:

$$\begin{aligned}
&\mathbf{u}_{i,0} = g_1(x_i) && \text{per } i = 0, \cdots, N_x + 1,\\
&\mathbf{u}_{N_x+1,j} = g_2(y_j) && \text{per } j = 0, \cdots, N_y + 1,\\
&\mathbf{u}_{i,N_y+1} = g_3(x_i) && \text{per } i = 0, \cdots, N_x + 1,\\
&\mathbf{u}_{0,j} = g_4(y_j) && \text{per } j = 0, \cdots, N_y + 1.
\end{aligned}$$

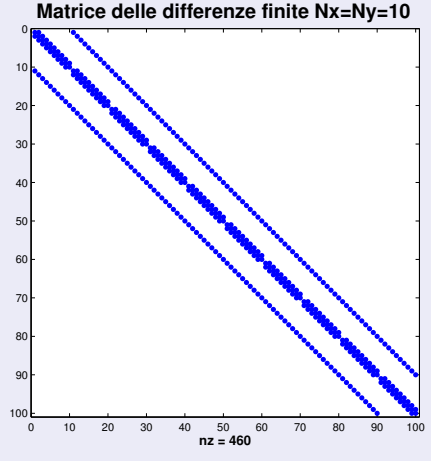
Sistema lineare

Vettore soluzione

- u vettore di $N = N_x \times N_y$ componenti: $\mathrm{u}(k) = \mathbf{u}_{i,j}$ essendo $k = (j-1) * N_x + i$;
- ricostruzione della soluzione: $\mathbf{u}_{i,j} = \mathrm{u}(k)$ per $j = \mathrm{int}(\frac{k-1}{N_x}) + 1$, $i = k - (j-1) * N_x$.

Matrice delle differenze finite

La matrice del sistema `A` è una matrice pentadiagonale con la struttura riportata in figura.
Se $h_x = h_y = h$ si costruisce con il comando
`A=delsq(numgrid('S',n+2))/hˆ2`



Costruzione della matrice delle differenze finite

Struttura della matrice delle differenze finite

Poniamo: $\alpha = 2/h_x^2 + 2/h_y^2$, $\beta = -1/h_x^2$, $\gamma = -1/h_y^2$ e $E = \gamma I_{N_x}$. La matrice A è pentadiagonale con la seguente struttura tridiagonale a blocchi:

$$A = \begin{pmatrix} D & E & \cdots & \cdots & 0 \\ E & D & E & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & E & D & E \\ 0 & \cdots & \cdots & E & D \end{pmatrix},$$

con N_y blocchi formati dalle matrici tridiagonali $D \in \mathbb{R}^{N_x \times N_x}$:

$$D = \begin{pmatrix} \alpha & \beta & \cdots & \cdots & 0 \\ \beta & \alpha & \beta & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \beta & \alpha & \beta \\ 0 & \cdots & \cdots & \beta & \alpha \end{pmatrix}.$$

Costruzione della matrice per $N_x \neq N_y$

Si può costruire la matrice per diagonali come segue.
Poniamo `Nx` $= N_x$, `Ny` $= N_y$, `hx` $= h_x$, `hy` $= h_y$, `N=Nx*Ny`

```
alfa=2/hx^2+2/hy^2;
beta=-1/hx^2;
gamma=-1/hy^2;
A=alfa*diag(ones(N,1))+beta*diag(ones(N-1,1),1)+...
        beta*diag(ones(N-1,1),-1);
A=A+gamma*diag(ones(N-Nx,1),Nx)+...
        gamma*diag(ones(N-Nx,1),-Nx);
for i=Nx:Nx:N-1
   A(i,i+1)=0;
end
for i=Nx+1:Nx:N
   A(i+1,i)=0;
end
```

Termine noto I

`b` vettore di $N = N_x \times N_y$ componenti tale che $\mathtt{b}(k) = f(ih_x, jh_y)+$ condizioni al bordo

$$\mathtt{b(1:Nx)} = \begin{pmatrix} f(x_1, y_1) + \frac{g_4(y_1)}{h_x^2} + \frac{g_1(x_1)}{h_y^2} \\ \dots \\ f(x_i, y_1) + \frac{g_1(x_i)}{h_y^2} \\ \dots \\ f(x_{N_x}, y_1) + \frac{g_2(y_1)}{h_x^2} + \frac{g_1(x_{N_x})}{h_y^2} \end{pmatrix},$$

Termine noto II

Per $j = 2, N_y - 1$

$$\texttt{b(Nx*(j-1)+1:Nx)} = \begin{pmatrix} f(x_1, y_j) + \frac{g_4(y_j)}{h_x^2} \\ \dots \\ f(x_i, y_j) \\ \dots \\ f(x_{N_x}, y_j) + \frac{g_2(y_j)}{h_x^2} \end{pmatrix}$$

Termine noto III

$$\texttt{b(Nx*(Ny-1)+(1:Nx))} = \begin{pmatrix} f(x_1, y_{N_y}) + \frac{g_4(y_{N_y})}{h_x^2} + \frac{g_3(x_1)}{h_y^2} \\ \dots \\ f(x_i, y_{N_y}) + \frac{g_3(x_i)}{h_y^2} \\ \dots \\ f(x_{N_x}, y_{N_y}) + \frac{g_2(y_{N_y})}{h_x^2} + \frac{g_3(x_{N_x})}{h_y^2} \end{pmatrix}$$

Costruzione del termine noto in Matlab

- Assegno la f: $\mathtt{b}(k) = f(ih_x, jh_y)$

```
k=0;
for j=1:Ny
   for i=1:Nx
      k=k+1;
      f(k)=effe(i*hx,j*hy);
   end
end
```

- Impongo le condizioni al bordo

```
for i=1:Nx
   f(i)=f(i)+g1(i*hx)/hy^2;
   f(Nx*(Ny-1)+i)=f(Nx*(Ny-1)+i)+g3(i*hx)/hy^2;
end
for j=1:Ny
   f((j-1)*Nx+1)=f((j-1)*Nx+1)+g4(j*hy)/hx^2;
   f(j*Nx)=f(j*Nx)+g2(j*hy)/hx^2;
end
```

Ricostruzione della soluzione

Per ricostruire la soluzione si vuole ottenere una matrice `Z` di dimensioni $N_y \times N_x$ che contiene i valori nei punti (x_i, y_j).
Tenendo conto di come è stato costruito il vettore `u` si può procedere a costruire la matrice `U` riga per riga nel modo seguente:

```
k=1;
for i=1:n
for j=1:n
Z(i,j)=u(k);
k=k+1;
end
end
```

Come tenere conto delle condizioni al bordo per rappresentare meglio la soluzione?

Esempio di grafico di una superficie

Sia data la funzione

$$f(x, y) = \sin(xy) \quad (x, y) \in [-\pi, \pi] \times [-\pi, \pi].$$

Per costruire il grafico serve la griglia dei valori (x, y) che si ottiene dai vettori `x` e `y`, mediante la function **meshgrid**.

```
>> m=20;n=25;
>> x=linspace(-pi,pi,n);
>> y=linspace(-pi,pi,m);
>> [X,Y]=meshgrid(x,y);
```

In questo modo si ottengono due matrici che contengono rispettivamente le ascisse e le ordinate dei punti della griglia. Per disegnare la funzione:

```
>> Z=sin(X.*Y);
>> mesh(X,Y,Z)
```

Grafici di superfici

Funzione	Significato
`view`	cambia l'orientamento del grafico.
`colormap`	cambia il colore al grafico.
`shading`	cambia l'ombreggiatura al grafico.
`mesh`	disegna un grafico a griglia.
`surf`	disegna un grafico di superficie.
`surfl`	disegna un grafico di superficie con ombreggiatura.
`contour`	disegna un grafico a curve di livello.
`contourf`	disegna un grafico a curve di livello riempite.
`pcolor`	disegna una scacchiera colorando le caselle.
`meshgrid`	genera i punti di una griglia.

contour

Il comando `contour` permette di disegnare le linee di livello di una superficie.
I vettori `X,Y,Z` contengono i punti della mesh e i valori della funzione rispettivamente.

`>> contour(X,Y,Z)`	disegna 9 linee di livello
`>> contour(X,Y,Z,N)`	disegna N linee di livello
`>> contour(X,Y,Z,[v v])`	disegna la linea di livello v
`>> contour(X,Y,Z,...`	disegna le linee di livello
`[v1 v2 v3 v4 ... vn])`	v1 v2 v3 v4 ... vn

Il comando `contour3` disegna le linee di livello in un grafico tridimensionale.

Rappresentazione della soluzione

- suddivisione dell'intervallo $[0, a]$ con $N_x + 2$ punti equispaziati: `X=linspace(0,a,Nx+2);`
- suddivisione dell'intervallo $[0, b]$ con $N_y + 2$ punti equispaziati: `X=linspace(0,b,Ny+2);`
- costruzione della griglia: `[x,y]=meshgrid(X,Y);`
- ricostruzione della soluzione in un array `(Nx+2)x(Ny+2)` in modo di tenere in considerazione anche le condizioni al bordo:

```
k=1;
U(1,1)=(g1(0)+g4(0))/2; U(1,Nx+2)=(g1(a)+g2(0))/2;
U(Ny+2,Nx+2)=(g2(b)+g3(a))/2; U(Ny+2,1)=(g3(0)+g4(b))/2;
U(1,2:Nx+1)=g1((1:Nx)*hx); U(Ny+2,2:Nx+1)=g3((1:Nx)*hx);
U(2:Ny+1,Nx+2)=g2((1:Ny)*hy); U(2:Ny+1,1)=g4((1:Ny)*hy);
for j=1:Ny
U(j+1,2:Nx+1)=u((j-1)*Nx+1:j*Nx)';
end
```

- `surf(x,y,U)`

Function `dirichlet`

La function `dirichlet` risolve il seguente problema di Laplace con condizioni al bordo di Dirichlet:

$$
\begin{array}{ll}
-\Delta u(x,y) = f(x,y) & (x,y) \in]0,a[\times]0,b[\\
u(x,0) = g_1(x) & x \in]0,a[, \\
u(a,y) = g_2(y) & y \in]0,b[\\
u(x,b) = g_3(x) & x \in]0,a[, \\
u(0,y) = g_4(y) & y \in]0,b[
\end{array}
$$

Il comando `help dirichlet` fornisce le istruzioni per l'uso della function `dirichlet`.

Esercizi

Esercizio 1

Usare la function `dirichlet` per risolvere il problema:

$$\begin{aligned} -\Delta u(x,y) &= 2(x(1-x)+y(1-y)) && (x,y) \in \Omega =]0,1[\times]0,1[\\ u(x,y) &= 0 && (x,y) \in \partial\Omega \end{aligned}$$

Confrontare con la soluzione esatta $u(x,y) = x(1-x)y(1-y)$.

Esercizio 2

Usare la function `dirichlet` per risolvere il problema:

$$\begin{aligned} -\Delta u(x,y) &= -e^x \left(2(1-\pi^2)\sin(\pi x)\sin(\pi y) + 2\pi \sin(\pi(x+y)) \right. \\ & && (x,y) \in \Omega =]0,1[\times]0,1[\\ u(x,y) &= 0 && (x,y) \in \partial\Omega \end{aligned}$$

Confrontare con la soluzione esatta $u(x,y) = e^x \sin(\pi x)\sin(\pi y)$.

Esercizi

Esercizio 3

Usare la function `dirichlet` per risolvere il problema:

$$
\begin{aligned}
-\Delta u(x,y) &= (12x+2y)(1-y^2)+(1-x^2)(6y+4x) && \\
& && (x,y)\in\Omega=]0,1[\times]0,1[\\
u(x,0) &= 2x(1-x^2) && x\in]0,1[, \\
u(1,y) &= 0 && y\in]0,1[\\
u(x,1) &= 0 && x\in]0,1[, \\
u(0,y) &= y(1-y^2) && y\in]0,1[
\end{aligned}
$$

Confrontare con la soluzione esatta $u(x,y)=(2x+y)(1-x^2)(1-y^2)$.

Esercizi

Esercizio 4

Modificare opportunamente la function `dirichlet` per risolvere l'equazione di Laplace su un dominio generale del tipo $\Omega =]a_1, a_2[\times]b_1, b_2[$. Usare la function `dirichlet` modificata per risolvere il problema:

$$\begin{aligned}
-\Delta u(x,y) &= 4 && (x,y) \in \Omega =]-1,1[\times]-1,1[\\
u(x,-1) &= -x^2 && x \in]-1,1[, \\
u(1,y) &= -y^2 && y \in]0,1[\\
u(x,1) &= -x^2 && x \in]-1,1[, \\
u(-1,y) &= -y^2 && y \in]0,1[
\end{aligned}$$

La soluzione esatta è: $u(x,y) = 1 - x^2 - y^2$.

Convergenza del metodo delle differenze finite applicate al caso del Laplaciano

Esercizio

Verificare l'ordine di convergenza del metodo delle differenze finite nel caso di uno dei problemi proposti negli esercizi 1-4.