Equazioni di Poisson

6 marzo 2009

1 Alcuni cenni sulle equazioni di Laplace e Poisson.

Si definisce equazione alle derivate parziali *quasi-lineare* il problema

$$A\frac{\partial^2 u}{\partial x^2} + B\frac{\partial^2 u}{\partial x \partial y} + C\frac{\partial^2 u}{\partial y^2} = f(x, y, u, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}),\ (x, y) \in \Omega \tag{1}$$

dove A, B e C sono costanti.
In particolare l'equazione è

- *ellittica* se $B^2 - 4AC < 0$;
- *parabolica* se $B^2 - 4AC = 0$;
- *iperbolica* se $B^2 - 4AC > 0$.

Passiamo a considerare qualche esempio notevole. Definito il dominio Ω, si consideri l'equazione di Poisson [1, p.557], [16], [17].

$$\begin{cases} \Delta u(x, y) := \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f,\ (x, y) \in \Omega \\ u = g,\ (x, y) \in \partial\Omega \end{cases} \tag{2}$$

Nel caso particolare $f \equiv 0$, l'equazione alle derivate parziali (17), è nota come equazione di Laplace [15] [14]. La particolare condizione al bordo è detta di *Dirichlet*. E' facile osservare che (17) è di tipo quasi-lineare con $A = 1$, $B = 0$, $C = 1$ e quindi ellittica.

Lo studio riguardo l'esistenza e l'unicità della soluzione al problema di Poisson non è banale. Accanto al problema differenziale di Poisson sopracitato, si definisce una formulazione *debole*. Sia Ω un aperto limitato di $\mathbb{R}^n$ di frontiera $\partial\Omega$ di classe C^1 a tratti. Si dimostra (non facile!), che il problema di Poisson per $f \in L_2(\Omega)$ con condizione di Dirichlet $u = 0$, $(x, y) \in \partial\Omega \subset \mathbb{R}^n$, ha un'unica soluzione debole nello spazio di Sobolev $H_0^1(\Omega)$ (cioè $f \in L_2(\Omega)$, $\frac{\partial f}{\partial x_i} f \in L_2(\Omega)$, $i = 1, \ldots, n$ ed f è nulla sul bordo $\partial\Omega$ [18]) [6, p.345] [9, p.23-29]. Per una descrizione più generale, anche nel caso di $f \neq 0$, si consideri [2, p.249] dove il problema è risolto con un uso non banale del lemma di Lax-Milgram.

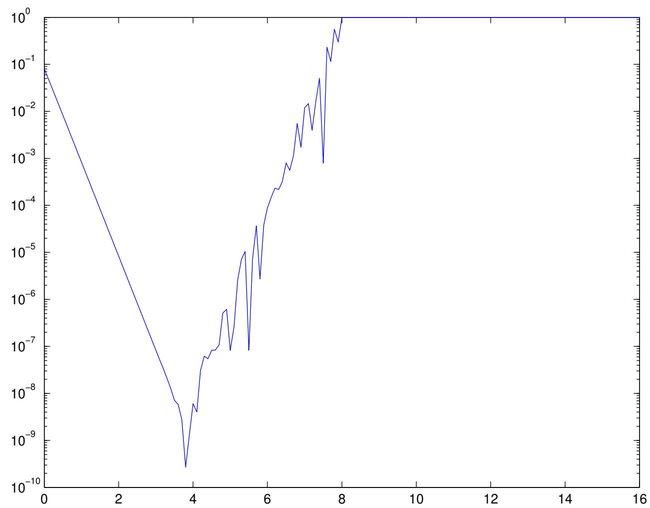

Figura 1: Grafico che illustra in scala semilogaritmica l'errore assoluto compiuto dalla formula (5) per il calcolo della derivata seconda del coseno nel punto $x = \pi$, utilizzando valori di $h = 10^{-s}$ dove s è il valore indicato nelle ascisse.

.

2 Discretizzazione del Laplaciano.

2.1 Discretizzazione del Laplaciano in dimensione uno (calcolo della derivata seconda).

Per studiare questo problema facciamo alcune osservazioni preliminari. Supponendo che $s : [a, b] \to \mathbb{R}$ sia sufficientemente regolare, dalla formula di Taylor si ha

$$s(x+h) = s(x) + hs^{(1)}(x) + \frac{h^2 s^{(2)}(x)}{2} + \frac{h^3 s^{(3)}(x)}{6} + \frac{h^4 s^{(4)}(\xi)}{24} \tag{3}$$

con $\xi \in \mathscr{I}(x, x+h)$ (dove $\mathscr{I}(a, b)$ è il più piccolo intervallo aperto contenente a, b). Similmente a (3) si ha quindi per $\epsilon \in \mathscr{I}(x, x-h)$

$$s(x-h) = s(x) - hs^{(1)}(x) + \frac{h^2 s^{(2)}(x)}{2} - \frac{h^3 s^{(3)}(x)}{6} + \frac{h^4 s^{(4)}(\epsilon)}{24}. \tag{4}$$

Sommando membro a membro (3), (4) abbiamo

$$\begin{aligned} s^{(2)}(x) &= \frac{s(x+h) - 2s(x) + s(x-h)}{h^2} + O(h^2) && (5) \\ &\approx \frac{s(x+h) - 2s(x) + s(x-h)}{h^2} && (6) \end{aligned}$$

Consideriamo la funzione

$$s(x) = \cos(x)$$

e valutiamo

$$s^{(2)}(x) = -\cos(x)$$

per $x = \pi$, qualora si usino alcuni valori di $h = 10^{-s}$ con $s = 0, 0.1, 0.2, \ldots, 16$. Noto che $s^{(2)}(\pi) = -\cos(\pi) = 1$ valuteremo l'errore assoluto (e al tempo stesso relativo) fornito dalla formula (5).

Un possibile codice Matlab `diff2experiments.m` è il seguente

```
s=0:0.1:16;
x=pi;
exactvalue=1;

for index=1:length(s)
    h_exp=s(index);
    h=10^(-h_exp);
    fxplus=f(x+h);
    fx=f(x);
    fxminus=f(x-h);
    f2approx=(fxplus-2*fx+fxminus)/(h^2);
    abserr(index)=abs(f2approx-exactvalue);
    fprintf('\n \t [h]: %2.2e [ABS. ERR.]: %2.2e',h,abserr(index))
end

semilogy(h_vett,abserr);
```

in cui `f.m` è data da

```
function fx=f(x)
fx=cos(x);
```

Si ottengono i seguenti risultati

```
[h]: 1.00e+000 [ABS. ERR.]: 8.06e-002
[h]: 7.94e-001 [ABS. ERR.]: 5.15e-002
[h]: 6.31e-001 [ABS. ERR.]: 3.27e-002
[h]: 5.01e-001 [ABS. ERR.]: 2.08e-002
[h]: 3.98e-001 [ABS. ERR.]: 1.31e-002
[h]: 3.16e-001 [ABS. ERR.]: 8.31e-003
...
[h]: 5.01e-004 [ABS. ERR.]: 2.09e-008
[h]: 3.98e-004 [ABS. ERR.]: 1.27e-008
[h]: 3.16e-004 [ABS. ERR.]: 7.19e-009
[h]: 2.51e-004 [ABS. ERR.]: 5.80e-009
[h]: 2.00e-004 [ABS. ERR.]: 2.77e-009
[h]: 1.58e-004 [ABS. ERR.]: 2.70e-010
[h]: 1.26e-004 [ABS. ERR.]: 1.26e-009
```

```
[h]: 1.00e-004 [ABS. ERR.]: 6.08e-009
[h]: 7.94e-005 [ABS. ERR.]: 4.04e-009
[h]: 6.31e-005 [ABS. ERR.]: 3.07e-008
[h]: 5.01e-005 [ABS. ERR.]: 6.21e-008
[h]: 3.98e-005 [ABS. ERR.]: 5.48e-008
[h]: 3.16e-005 [ABS. ERR.]: 8.27e-008
[h]: 2.51e-005 [ABS. ERR.]: 8.39e-008
[h]: 2.00e-005 [ABS. ERR.]: 1.09e-007
```

che dicono che la formula per il calcolo della derivata seconda è da usarsi con cautela, in quanto fornisce risultati accurati per h non eccessivamente piccolo altrimenti errori di arrotondamento *degradano* la convergenza della formula.

Quale esercizio si provi il caso in cui $f(x) = \exp(x)$, $x = 0$ (e quindi $f^{(2)}(0) = 1$). Si faccia attenzione a modificare correttamente la funzione f e le variabili `x`, `exactvalue`.

2.2 Discretizzazione del Laplaciano in dimensione due.

Siamo pronti ad introdurre una *discretizzazione* dell'operatore di Laplace

$$\Delta u := u_{xx} + u_{yy} := \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}.$$

Dalla formula unidimensionale abbiamo

$$u_{xx}(x, y) = \frac{u(x+h, y) - 2u(x, y) + u(x-h, y)}{h^2} + O(h^2), \tag{7}$$

e

$$u_{yy}(x, y) = \frac{u(x, y+h) - 2u(x, y) + u(x, y-h)}{h^2} + O(h^2), \tag{8}$$

da cui

$$\begin{aligned} \Delta u(x, y) &= \frac{u(x, y+h) - 2u(x, y) + u(x, y-h)}{h^2} \\ &+ \frac{u(x+h, y) - 2u(x, y) + u(x-h, y)}{h^2} + O(h^2) \\ &\approx \frac{u(x+h, y) + u(x-h, y) + u(x, y+h) + u(x, y-h) - 4u(x, y)}{h^2}. \end{aligned} \tag{9}$$

Facciamo una nuova verifica numerica. Sia

$$g(x, y) = \cos(x) \exp(2y)$$

e supponiamo di voler valutare $\Delta g(0,0)$. E' facile osservare che

$$\Delta g(x, y) = -\cos(x) \exp(2y) + 4\cos(x) \exp(2y)$$

e che quindi

$$\Delta g(0,0) = -1 \cdot 1 + 4 \cdot 1 \cdot 1 = 3.$$

Scriviamo quindi le 2 funzioni Matlab/Octave `diff22Dexperiments.m` e `g.m` che ci permettono di sperimentare quanto descritto:

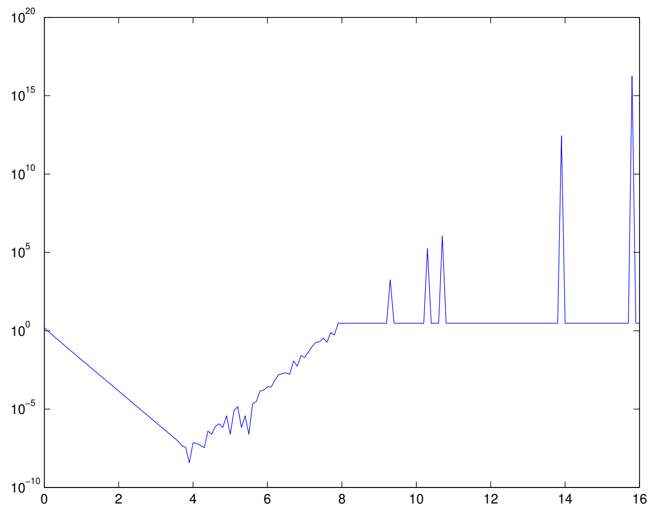

Figura 2: Grafico che illustra in scala semilogaritmica l'errore assoluto compiuto dalla formula (9) per il calcolo del laplaciano della funzione $g(x, y)$ nel punto $(0,0)$, utilizzando valori di $h = 10^{-s}$ dove s è il valore indicato nelle ascisse.

.

```
s=0:0.1:16;
x=0; y=0;

correct_value=3;

for index=1:length(s)
    h_exp=s(index);
    h=10^(-h_exp);
    g2approx=( g(x+h,y)+g(x-h,y)+g(x,y+h)+g(x,y-h)-4*g(x,y) )/(h^2);
    abserr(index)=abs(g2approx-correct_value);
    fprintf('\n \t [h]: %2.2e [ABS.ERR.]: %2.2e',h,abserr(index))
end

semilogy(s,abserr);



function gxy=g(x,y)
gxy=cos(x)*exp(2*y);
```

I risultati numerici sono i seguenti

```
[h]: 1.00e+000 [ABS.ERR.]: 1.60e+000
```

```
[h]: 7.94e-001 [ABS.ERR.]: 9.67e-001
[h]: 6.31e-001 [ABS.ERR.]: 5.93e-001
[h]: 5.01e-001 [ABS.ERR.]: 3.67e-001
[h]: 3.98e-001 [ABS.ERR.]: 2.29e-001
[h]: 3.16e-001 [ABS.ERR.]: 1.43e-001
...
[h]: 5.01e-004 [ABS.ERR.]: 3.54e-007
[h]: 3.98e-004 [ABS.ERR.]: 2.22e-007
[h]: 3.16e-004 [ABS.ERR.]: 1.42e-007
[h]: 2.51e-004 [ABS.ERR.]: 8.99e-008
[h]: 2.00e-004 [ABS.ERR.]: 4.75e-008
[h]: 1.58e-004 [ABS.ERR.]: 3.62e-008
[h]: 1.26e-004 [ABS.ERR.]: 3.79e-009
[h]: 1.00e-004 [ABS.ERR.]: 7.06e-008
[h]: 7.94e-005 [ABS.ERR.]: 6.49e-008
[h]: 6.31e-005 [ABS.ERR.]: 4.75e-008
[h]: 5.01e-005 [ABS.ERR.]: 3.45e-008
[h]: 3.98e-005 [ABS.ERR.]: 3.96e-007
[h]: 3.16e-005 [ABS.ERR.]: 2.48e-007
...
[h]: 2.51e-008 [ABS.ERR.]: 1.85e-001
[h]: 2.00e-008 [ABS.ERR.]: 7.69e-001
[h]: 1.58e-008 [ABS.ERR.]: 5.36e-001
...
```

che suggerisce nuovamente che l'utilizzo di tale discretizzazione richiede che il *passo* h non sia eccessivamente piccolo.
Consideriamo l'equazione (cf. [1, p.557])

$$\begin{cases} \Delta u(x,y) = f(x,y), \ (x,y) \in \Omega = (0,1) \times (0,1) \\ u(x,y) = g(x,y), \ (x,y) \in \partial\Omega \end{cases} \tag{10}$$

Definita la griglia di punti $\mathscr{G} = \{(x_i, y_j)\}_{i,j=0,\ldots,n+1}$

$$x_i = ih, \ y_j = jh, \ h = 1/(n+1), \ i,j = 0,\ldots,n+1$$

risulta evidente che per $i = 0$ o $j = 0$, $i = n+1$ o $j = n+1$ abbiamo un punto del bordo e quindi in virtù delle condizioni di Dirichlet in (10), il valore di u^* è determinato. Vediamo cosa succede quando il punto della griglia $\mathscr{G}$ è interno al quadrato $\Omega = (0,1) \times (0,1)$, cioè nel caso in cui (x_i, y_j) sia tale che nessuno tra gli indici i, j sia uguale a 0 oppure $n+1$.

3 Discretizzazione dell'equazione di Poisson-Laplace.

Dalla discretizzazione dell'operatore di Laplace Δ, tralasciando i termini $O(h^2)$ abbiamo

$$\Delta u(x,y) \approx \frac{u(x+h,y) + u(x-h,y) + u(x,y+h) + u(x,y-h) - 4u(x,y)}{h^2}$$

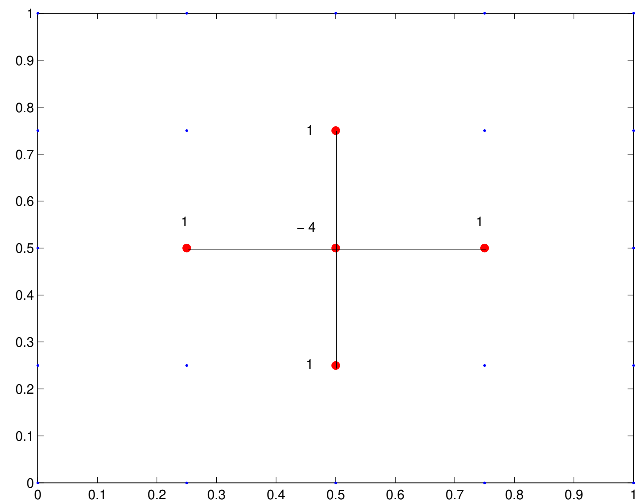

Figura 3: La molecola della citata discretizzazione del Laplaciano avente centro $(0.5, 0.5)$ e $h = 0.25$. Si ricordi di dividere ogni valore nella molecola per h^2.

.

e quindi

$$\Delta u(x_i, y_j) = \frac{u(x_{i+1}, y_j) + u(x_{i-1}, y_j) + u(x_i, y_{j+1}) + u(x_i, y_{j-1}) - 4u(x_i, y_j)}{h^2} \tag{11}$$

da cui la discretizzazione dell'equazione di Poisson

$$u(x_{i+1}, y_j) + u(x_{i-1}, y_j) + u(x_i, y_{j+1}) + u(x_i, y_{j-1}) - 4u(x_i, y_j) = h^2 f(x_i, y_j), \tag{12}$$

per $i, j = 1, \ldots, n$ con le condizioni al contorno

$$u(x_i, y_j) = g(x_i, y_j),\ i = 0,\ j = 1, \ldots, n \tag{13}$$
$$u(x_i, y_j) = g(x_i, y_j),\ i = n+1,\ j = 1, \ldots, n \tag{14}$$
$$u(x_i, y_j) = g(x_i, y_j),\ i = 1, \ldots, n,\ j = 0, j = n+1 \tag{15}$$

Purtroppo, la descrizione del sistema lineare non è troppo chiara. Vediamola scritta matricialmente. Sia B la matrice $n \times n$

$$B = \begin{pmatrix} -4 & 1 & 0 & \ldots & 0 \\ 1 & -4 & 1 & 0 & \ldots \\ 0 & 1 & -4 & 1 & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & 1 & -4 \end{pmatrix}$$

ed I la matrice identica di ordine n del tipo

$$I = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 0 & \ldots & 1 \end{pmatrix}.$$

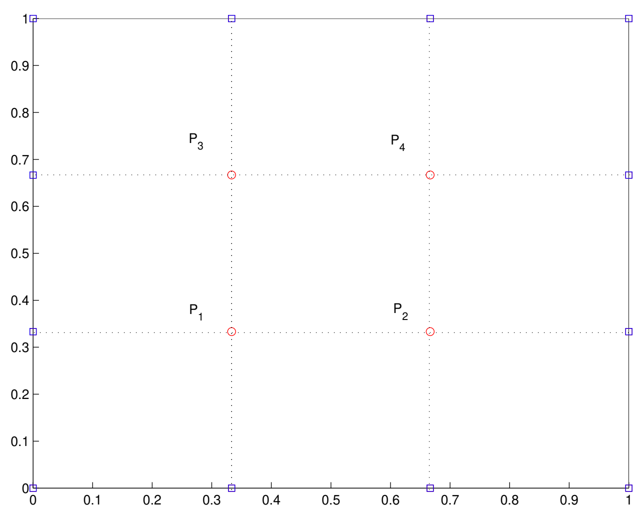

Figura 4: Una griglia avente passo $h = 1/3$ relativamente al quadrato $[0,1] \times [0,1]$.
.

Allora se b è il vettore ottenuto dai contributi dei termini dovuti a f e g in (10) e (12), definita la matrice a blocchi

$$A = \begin{pmatrix} B & I & 0 & \dots & 0 \\ I & B & I & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & 0 & I & B & I \end{pmatrix}$$

si ricava che il sistema da risolvere è $Au = b$.

Per una implementazione della matrice di Poisson A, utilizziamo la funzione `makefish`

```
function mat = makefish(siz);
% make a Poisson matrix

leng = siz*siz;
dia = zeros(siz,siz);
off = -eye(siz,siz);
 for i=1:siz, dia(i,i)=4; end;
 for i=1:siz-1, dia(i,i+1)=-1; dia(i+1,i)=-1; end;
mat = zeros(leng,leng);
 for ib=1:siz,
  mat(1+(ib-1)*siz:ib*siz,1+(ib-1)*siz:ib*siz) = dia; end;
 for ib=1:siz-1,
  mat(1+(ib-1)*siz:ib*siz,1+ib*siz:(ib+1)*siz) = off;
  mat(1+ib*siz:(ib+1)*siz,1+(ib-1)*siz:ib*siz) = off; end;
return;
```

Vediamone un esempio dalla shell di Matlab/Octave:

Figura 5: Pierre-Simon Laplace (1749-1827) e Siméon Denis Poisson (1781-1840).

```
>> makefish(2)
ans =
     4    -1    -1     0
    -1     4     0    -1
    -1     0     4    -1
     0    -1    -1     4
>>
```

Si vede subito dal 4 sulla diagonale che `makefish` non calcola A ma $-A$ e dovremo tener conto di questo dettaglio nell'implementazione.

Osserviamo che non è proprio facile determinare, fissato i, j, quali siano i punti adiacenti a (x_i, y_j) che essendo sul bordo hanno valore della soluzione noto a priori e quindi tali da contribuire attivamente al termine noto.

4 Sull'equazione di Poisson. Esempio del metodo alle differenze con 5 punti.

Facciamo un esempio sulla risoluzione dell'equazione di Poisson via metodo alle differenze con 5 punti. Sia $\Omega = [0,1] \times [0,1]$, $h = 1/3$. Siano

$$P_{i,j} = (ih, jh),\ i,j = 0,1,2,3.$$

E' chiaro che per $i = 0$ i punti $P_{0,j}$ sono sull'asse $x = 0$ (cioè l'asse y), per $i = 3$ i punti $P_{3,j}$ sono sull'asse $x = 1$, per $j = 0$ i punti $P_{i,0}$ sono sull'asse $y = 0$ (cioè l'asse x) e per $j = 3$ i punti $P_{i,3}$ sono sull'asse $y = 1$. Date le condizioni al contorno, la soluzione in questi punti è nota ed è uguale a $u_{i,j} = g(x_i, y_j)$.

I rimanenti punti $P_{i,j}$, con $i, j = 1,2$ sono interni al quadrato Ω. Per quanto visto

$$u(x_{i+1}, y_j) + u(x_{i-1}, y_j) + u(x_i, y_{j+1}) + u(x_i, y_{j-1}) - 4u(x_i, y_j) = h^2 f(x_i, y_j), \quad (16)$$

Analizziamo caso per caso queste equazioni:

1. Nel caso $i = 1$, $j = 1$ si ha

$$u(x_2, y_1) + u(x_0, y_1) + u(x_1, y_2) + u(x_1, y_0) - 4u(x_1, y_1) = h^2 f(x_1, y_1),$$

ed essendo

$$u(x_0, y_1) = g(x_0, y_1),\ u(x_1, y_0) = g(x_1, y_0)$$

portando questi due termini a secondo membro otteniamo

$$u(x_2, y_1) + u(x_1, y_2) - 4u(x_1, y_1) = h^2 f(x_1, y_1) - g(x_0, y_1) - g(x_1, y_0).$$

2. Nel caso $i = 2$, $j = 1$ si ha

$$u(x_3, y_1) + u(x_1, y_1) + u(x_2, y_2) + u(x_2, y_0) - 4u(x_2, y_1) = h^2 f(x_2, y_1),$$

ed essendo

$$u(x_3, y_1) = g(x_3, y_1),\ u(x_2, y_0) = g(x_2, y_0)$$

portando questi due termini a secondo membro otteniamo

$$u(x_1, y_1) + u(x_2, y_2) - 4u(x_2, y_1) = h^2 f(x_1, y_1) - g(x_3, y_1) - g(x_2, y_0).$$

3. Nel caso $i = 1$, $j = 2$ si ha

$$u(x_2, y_2) + u(x_0, y_2) + u(x_1, y_3) + u(x_1, y_1) - 4u(x_1, y_2) = h^2 f(x_1, y_2),$$

ed essendo

$$u(x_0, y_2) = g(x_0, y_2),\ u(x_1, y_3) = g(x_1, y_3)$$

portando questi due termini a secondo membro otteniamo

$$u(x_2, y_2) + u(x_1, y_1) - 4u(x_1, y_2) = h^2 f(x_1, y_2) - g(x_0, y_2) - g(x_1, y_3).$$

4. Nel caso $i = 2$, $j = 2$ si ha

$$u(x_3, y_2) + u(x_1, y_2) + u(x_2, y_3) + u(x_2, y_1) - 4u(x_2, y_2) = h^2 f(x_2, y_2),$$

ed essendo

$$u(x_3, y_2) = g(x_3, y_2),\ u(x_2, y_3) = g(x_2, y_3)$$

portando questi due termini a secondo membro otteniamo

$$u(x_1, y_2) + u(x_2, y_1) - 4u(x_2, y_2) = h^2 f(x_2, y_2) - g(x_3, y_2) - g(x_2, y_3).$$

Poniamo ora

$$\begin{aligned}
b_1 &:= h^2 f(x_1, y_1) - g(x_0, y_1) - g(x_1, y_0),\\
b_2 &:= h^2 f(x_1, y_2) - g(x_0, y_2) - g(x_1, y_3),\\
b_3 &:= h^2 f(x_1, y_1) - g(x_3, y_1) - g(x_2, y_0),\\
b_4 &;= h^2 f(x_2, y_2) - g(x_3, y_2) - g(x_2, y_3),
\end{aligned}$$

ordiniamo i punti da sinistra a destra, e dal basso verso l'alto (ordine lessicografico)

$$P_1 = (x_1, y_1),\ P_2 = (x_2, y_1),\ P_3 = (x_1, y_2),\ P_4 = (x_2, y_2),$$

e infine poniamo

$$u_1 = u(x_1, y_1),\ u_2 = u(x_2, y_1),\ u_3 = u(x_1, y_2),\ u_4 = u(x_2, y_2),$$

ottenendo così

$$\begin{aligned}
u_2 + u_3 - 4u_1 &= b_1, \\
u_1 + u_4 - 4u_2 &= b_2, \\
u_4 + u_1 - 4u_3 &= b_3, \\
u_3 + u_2 - 4u_4 &= b_4,
\end{aligned}$$

da cui posto

$$A = \begin{pmatrix} -4 & 1 & 1 & 0 \\ 1 & -4 & 0 & 1 \\ 1 & 0 & -4 & 1 \\ 0 & 1 & 1 & -4 \end{pmatrix}$$

basta risolvere il sistema $Au = b$ per ottenere $u(x_1, y_1)$, $u(x_2, y_1)$, $u(x_1, y_2)$, $u(x_2, y_2)$.

Notiamo che

```
>> makefish(2)
ans =
     4    -1    -1     0
    -1     4     0    -1
    -1     0     4    -1
     0    -1    -1     4
>>
```

calcola proprio $-A$, mentre il termine noto b può essere facilmente calcolato dopo aver notato che

1. i termini $-g(x_i, y_0)$ sono presenti nelle componenti b_i;
2. posto $n = 1/h$, per $i = 1, \ldots, n-1$, i termini $-g(x_i, y_n)$ sono presenti nelle componenti $b_{(n-1)^2-(n-1)+i}$;
3. per $j = 1, \ldots, n-1$, i termini $-g(x_0, y_j)$ sono presenti nelle componenti b_s con $s \equiv 1 \mod n-1$;
4. per $j = 1, \ldots, n-1$, i termini $-g(x_n, y_j)$ sono presenti nelle componenti b_s con $s \equiv 0 \mod n-1$.

5 Sull'equazione di Poisson. Implementazione del metodo alle differenze con 5 punti.

Vediamo ora un'implementazione del metodo sopra descritto, detto per ovvi motivi *a 5 punti* (cf. (12)).

Risulta importante ricordare che in [1, p. 558], si cita la seguente stima dell'errore, per soluzioni u dell'equazione di Poisson (10) almeno 4 volte differenziabili con continuità ; se u è la soluzione esatta del problema definito nel quadrato $\Omega := [0,1]\times[0,1]$ ed u_h quella ottenuta utilizzando una griglia $\mathscr{G} = \{(x_i, y_j)\}$ con $x_i = i\,h$, $y_j = j\,h$, $h = 1/(n+1)$ allora

$$|u(x_i, y_j) - u_h(x_i, y_j)| \le ch^2$$

con

$$c = (1/24)\left(\max_{(x,y)\in\Omega}\left|\frac{\partial^4 u(x,y)}{\partial x^4}\right| + \max_{(x,y)\in\Omega}\left|\frac{\partial^4 u(x,y)}{\partial y^4}\right|\right)$$

Ci si aspetta quindi dai test numerici che effettueremo un errore dell'ordine di h^2.

Salviamo in `poisson5pts.m` la funzione

```
function u=poisson5pts(n,f,g_left,g_right,g_down,g_up)

% ATKINSON. P. 559.% RICORDARSI CHE LA MATRICE DI
% MAKEFISH HA SEGNO OPPOSTO A QUELLA DA UTILIZZARE!!
A=-makefish(n);

h=1/(n+1);

x=(h:h:1-h)'; y=x;

% SOLUZIONI IN BASSO.
x_loc=x;
y_loc=zeros(size(x_loc));
b_down=feval(g_down,x_loc,y_loc);

% SOLUZIONI IN ALTO.
y_loc=ones(size(x_loc));
b_up=feval(g_up,x_loc,y_loc);

% SOLUZIONI A SINISTRA.
y_loc=x_loc; x_loc=zeros(size(x_loc));
b_left=feval(g_left,x_loc,y_loc);

% SOLUZIONI A DESTRA
x_loc=ones(size(x_loc));
b_right=feval(g_right,x_loc,y_loc);

% COMPOSIZIONE TERMINE NOTO.
b1=b_down;
```

```
% PRIMA RIGA IN BASSO.
b1(1)=b1(1)+b_left(1); b1(n)=b1(n)+b_right(1);

% PRIMA RIGA IN ALTO.
bn=b_up; bn(1)=bn(1)+b_left(n); bn(n)=bn(n)+b_right(n);

% RIGHE INTERMEDIE.
bj=[];
for j=2:(n-1)
    bjloc=zeros(n,1);
    bjloc(1)=bjloc(1)+b_left(j); bjloc(n)=bjloc(n)+b_right(j);
    bj=[bj; bjloc];
end

b=[b1; bj; bn];

% GRIGLIA LESSICOGRAFICA (METODO STANDARD).
lunghezza_x=length(x);

X=[]; Y=[];
for index=1:lunghezza_x
    X=[X; x];
    Y=[Y; y(index)*ones(size(x))];
end

fXY=feval(f,X,Y);

b_f=(h^2)*fXY;
b=b_f-b;

u=A\b;
```

e in `demopoisson5pts.m` la demo

```
% MODIFIED VERSION: MARCH 13, 2008.

demo_example=2;

switch demo_example
case 1
    f=inline('zeros(size(x))','x','y');
    g_down=inline('ones(size(x))','x','y');
    g_up=inline('ones(size(x))','x','y');
    g_left=inline('ones(size(x))','x','y');
    g_right=inline('ones(size(x))','x','y');
```

```
    solution=inline('ones(size(x))','x','y');

case 2
    f=inline('zeros(size(x))','x','y');
    g_down=inline('exp(pi*x)','x','y');
    g_up=inline('-exp(pi*x)','x','y');
    g_left=inline('cos(pi*y)','x','y');
    g_right=inline('((exp(1))^pi)*cos(pi*y)','x','y');

    solution=inline('(exp(pi*x)).*cos(pi*y)','x','y');

case 3
    f=inline('(-2*(pi^2))*sin(pi*x).*sin(pi*y)','x','y');
    g_down=inline('zeros(size(x))','x','y');
    g_up=inline('zeros(size(x))','x','y');
    g_left=inline('zeros(size(x))','x','y');
    g_right=inline('zeros(size(x))','x','y');

    solution=inline('(sin(pi*x)).*sin(pi*y)','x','y');

otherwise
    f=inline('ones(size(x))','x','y');
    g_down=inline('zeros(size(x))','x','y');
    g_up=inline('zeros(size(x))','x','y');
    g_left=inline('zeros(size(x))','x','y');
    g_right=inline('zeros(size(x))','x','y');

    solution=inline('(sin(pi*x)).*sin(pi*y)','x','y');
end

for index=2:5
    n=2^index;

    h=1/(n+1); x=(h:h:1-h)'; y=x;
    [X,Y]=meshgrid(x,y);
    X=X'; Y=Y';

    % VETTORE SOLUZIONE NEI PUNTI DELLA GRIGLIA ORDINATI IN LESSICOGRAFICO.
    u=poisson5pts(n,f,g_left,g_right,g_down,g_up);

    % USO RESHAPE COSI' LA SOLUZIONE HA LE STESSE DIMENSIONI DELLE MATRICI X, Y.
    Z=(reshape(u,n,n));

    if demo_example <=3

        V=feval(solution,X,Y);

        err(index)=norm(V(:)-Z(:),inf);
        if index == 1
            fprintf('\n \t [n]: %4.0f [ERR]: %2.2e',n,err(index));
```

```
        else
            fprintf('\n \t [n]: %4.0f [ERR]: %2.2e [RATIO]: %2.2f',...
                n,err(index),err(index-1)/err(index));
        end

    end
end

surf(X,Y,Z);
```

Alcune osservazioni:

1. Posto $h = \frac{1}{n+1}$, allora la matrice $B \in \mathbb{R}^{n \times n}$ mentre il termine noto b e il vettore soluzione u apparterranno a $\mathbb{R}^{n^2}$. dobbiamo risolvere un sistema lineare $Au = b$ con

$$A = \begin{pmatrix} B & I & 0 & \dots & 0 \\ I & B & I & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & 0 & I & B & I \end{pmatrix}$$

   e b un vettore i cui contributi dipendono dai valori che hanno sul bordo le funzioni f e g che definiscono l'equazione di Poisson

$$\begin{cases} -\left[\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right] = f(x,y), \ (x,y) \in \Omega \\ u(x,y) = g(x,y), \ (x,y) \in \partial\Omega \end{cases} \tag{17}$$

   Dobbiamo tener conto che `makefish` non calcola A ma $\bar{A} = -A$ ed è per questo che scriviamo `A=-makefish(n)`.

2. (**Per i più esperti**) La funzione `meshgrid` crea a partire da due vettori di numeri reali x, y, le ascisse X e le ordinate Y dei punti facenti parte della griglia *generata* da x ed y. In particolare la matrice di punti

$$\begin{pmatrix} (x_1, y_1) & (x_2, y_1) & \dots & (x_n, y_1) \\ \dots & \dots & \dots & \dots \\ (x_1, y_n) & (x_2, y_n) & \dots & (x_n, y_n) \end{pmatrix}$$

   viene descritta tramite la griglia di ascisse e ordinate e cioè come

$$X = \begin{pmatrix} x_1 & x_2 & \dots & x_n \\ \dots & \dots & \dots & \\ x_1 & x_2 & \dots & x_n \end{pmatrix}$$

   e

$$Y = \begin{pmatrix} y_1 & y_1 & \dots & y_1 \\ y_2 & y_2 & \dots & y_2 \\ y_3 & y_3 & \dots & y_3 \end{pmatrix}$$

Così ad esempio

```
>> h=1/3;
>> x=h:h:1-h;
>> x
x =
    0.3333    0.6667
>> y=x;
>> [X,Y]=meshgrid(x,y)
X =
    0.3333    0.6667
    0.3333    0.6667
Y =
    0.3333    0.3333
    0.6667    0.6667
>>
```

descrivendone le coordinate x, y.

Per ulteriori chiarimenti usiamo l'help di Matlab 6.1 ottenendo

```
>> help meshgrid

 MESHGRID   X and Y arrays for 3-D plots.
    [X,Y] = MESHGRID(x,y) transforms the domain specified by vectors
    x and y into arrays X and Y that can be used for the evaluation
    of functions of two variables and 3-D surface plots.
    The rows of the output array X are copies of the vector x and
    the columns of the output array Y are copies of the vector y.

    [X,Y] = MESHGRID(x) is an abbreviation for [X,Y] = MESHGRID(x,x).
    [X,Y,Z] = MESHGRID(x,y,z) produces 3-D arrays that can be used to
    evaluate functions of three variables and 3-D volumetric plots.

    For example, to evaluate the function  x*exp(-x^2-y^2) over the
    range  -2 < x < 2,  -2 < y < 2,

        [X,Y] = meshgrid(-2:.2:2, -2:.2:2);
        Z = X .* exp(-X.^2 - Y.^2);
        mesh(Z)

    ... ...
```

```
    See also SURF, SLICE, NDGRID.
```

Il comando Matlab ":" mappa invece una matrice A in un vettore $A(:)$ ottenuto *incolonnando* le colonne di A. Ad esempio

```
>> A=[1 2; 3 4; 5 6]
A =
     1     2
     3     4
     5     6
>> A(:)
ans =
     1
     3
     5
     2
     4
     6
>>
```

Se x è un vettore colonna, la parte di codice

```
lunghezza_x=length(x);
X=[]; Y=[];
for index=1:lunghezza_x
    X=[X;x];
    Y=[Y; y(index)*ones(size(x))];
end
```

è equivalente a

```
[X,Y]=meshgrid(x,y);
X=X'; Y=Y';
X=X(:); Y(:);
```

Vediamolo in un'implementazione della griglia lessografica descritta nell'esempio della sezione precedente

```
>> h=1/3;
>> x=h:h:1-h; x=x'
x =
    0.3333
    0.6667
>> y=x;
>> lunghezza_x=length(x);
```

```
X=[]; Y=[];
for index=1:lunghezza_x
    X=[X;x];
    Y=[Y; y(index)*ones(size(x))];
end
>> X
X =
    0.3333
    0.6667
    0.3333
    0.6667
>> Y
Y =
    0.3333
    0.3333
    0.6667
    0.6667
>> % METODO COL MESHGRID
>> [X,Y]=meshgrid(x,y);
X=X'; Y=Y';
>> X(:)
X =
    0.3333
    0.6667
    0.3333
    0.6667
>> Y=Y(:)
Y =
    0.3333
    0.3333
    0.6667
    0.6667
>>
```

3. una porzione di codice particolarmente complicata di `demopoisson5pts` è

    ```
    [X,Y]=meshgrid(x,y);
    X=X'; Y=Y';

    % VETTORE SOLUZIONE NEI PUNTI DELLA GRIGLIA
    % ORDINATI IN LESSICOGRAFICO.
    u=poisson5pts(n,f,g_left,g_right,g_down,g_up);

    % USO RESHAPE COSI' LA SOLUZIONE HA LE
    % STESSE DIMENSIONI DELLE MATRICI X, Y.
    Z=(reshape(u,n,n));
    ```

    L'effetto è di costruire la griglia di punti sotto forma di due matrici X, Y, e poi di definire una matrice Z che assegni a ogni punto della griglia il valore della soluzione che precedentemente era immagazzinata nel vettore u. Il comando `reshape` permette il passaggio dal vettore u alla matrice Z.

4. Notiamo l'uso in `demopoisson5pts` di `reshape`, affinchè la soluzione abbia una struttura uguale a quella fornita da meshgrid. Questa scelta è dovuta alla scelta di usare `surf` per plottare la funzione. Si ricorda che `surf` non è implementato in tutte le versioni di Octave. Funziona in Gnu Octave per PC, in versione 2.1.73.

   Vediamo un esempio di `reshape`.

```
>> u=[1;2;3;4;5;6]
u =
     1
     2
     3
     4
     5
     6
>> reshape(u,2,3)
ans =
     1     3     5
     2     4     6
>>
```

   In pratica `reshape(u,m,n)` parte da un vettore u di dimensione $m \cdot n$ (nel nostro caso $6 = 2 \cdot 3$), e costruisce colonna per colonna una matrice $m \times n$. Per ulteriori dettagli seguiamo l'help di `reshape`

```
>> help reshape

 RESHAPE Change size.
    RESHAPE(X,M,N) returns the M-by-N matrix whose elements
    are taken columnwise from X.  An error results if X does
    not have M*N elements.
    ... ...

    See also SQUEEZE, SHIFTDIM, COLON.
```

   Per quanto riguarda `surf` esegue il plot dei valori Z di una griglia strutturata in matrici X, Y come quelle costruite via meshgrid,

```
>> help surf

 SURF   3-D colored surface.
    SURF(X,Y,Z,C) plots the colored parametric surface defined by
```

```
four matrix arguments.  The view point is specified by VIEW.
The axis labels are determined by the range of X, Y and Z,
or by the current setting of AXIS.  The color scaling is determined
by the range of C, or by the current setting of CAXIS.  The scaled
color values are used as indices into the current COLORMAP.
The shading model is set by SHADING.

SURF(X,Y,Z) uses C = Z, so color is proportional to surface height.

... ...

AXIS, CAXIS, COLORMAP, HOLD, SHADING and VIEW set figure, axes, and
surface properties which affect the display of the surface.

See also SURFC, SURFL, MESH, SHADING.
```

5.1 Problema 1.

Consideriamo un semplice esempio che illustra la risoluzione dell'equazione di Poisson con un metodo alle differenze. Come in [3, p. 284], consideriamo l'equazione di Poisson nel quadrato unitario $\Omega = [0,1] \times [0,1]$

$$\begin{cases} \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0, \ (x,y) \in \Omega \\ u(x,y) = 1, \ (x,y) \in \partial\Omega \end{cases} \tag{18}$$

la cui soluzione è $u(x,y) = 1$.
Non è difficile osservare che

$$\max_{(x,y)\in\Omega} \left| \frac{\partial^4 u(x,y)}{\partial x^4} \right| = 0, \ \max_{(x,y)\in\Omega} \left| \frac{\partial^4 u(x,y)}{\partial y^4} \right| = 0$$

e quindi ci si aspetta che per qualsiasi h si abbia un errore dell'ordine della precisione di macchina.
Lanciamo da shell il primo esempio della demo, ottenendo

```
>> demopoisson5pts

         [n]:    4 [ERR]: 4.44e-016 [RATIO]: 163181252362740.50
         [n]:    8 [ERR]: 1.11e-015 [RATIO]: 0.40
         [n]:   16 [ERR]: 1.55e-015 [RATIO]: 0.71
         [n]:   32 [ERR]: 6.66e-015 [RATIO]: 0.23

>>
```

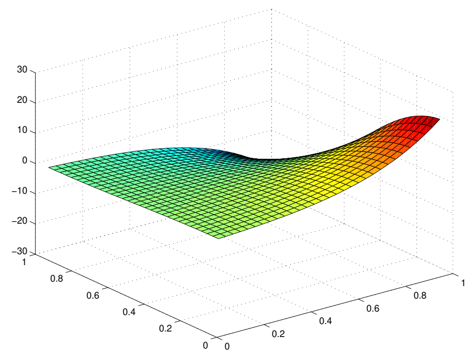

Figura 6: Soluzione del problema 2.

5.2 Problema 2.

Consideriamo un semplice esempio che illustra la risoluzione dell'equazione di Laplace nel quadrato unitario $\Omega = [0,1] \times [0,1]$ con un metodo alle differenze.

$$\left\{ \begin{array}{l} \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0, \ \ (x,y) \in \Omega \\ u(x,0) = \exp(\pi\, x), \ \ x \in [0,1] \\ u(x,1) = -\exp(\pi\, x), \ \ x \in [0,1] \\ u(0,y) = \cos(\pi\, y), \ \ y \in [0,1] \\ u(1,y) = \exp(\pi) \cdot \cos(\pi\, y), \ \ y \in [0,1] \end{array} \right. \tag{19}$$

la cui soluzione è $u(x,y) = \exp(\pi\, x) \cdot \cos(\pi\, y)$. Si nota subito che per $x, y \in [0,1]$ si ha

$$\left| \frac{\partial^4 u}{\partial x^4}(x,y) \right| = \pi^4 \left| \exp(\pi\, x) \cdot \cos(\pi\, y) \right| \leq \pi^4 \exp(\pi) \approx 2254.1$$

$$\left| \frac{\partial^4 u}{\partial y^4}(x,y) \right| = \pi^4 \left| \exp(\pi\, x) \cdot \cos(\pi\, y) \right| \leq \pi^4 \exp(\pi) \approx 2254.1$$

da cui

$$c \leq (1/24) \cdot 2254.1 \cdot 2 \approx 187.8428.$$

Quindi quale maggiorazione dell'errore assoluto in norma infinito, per $n = 3, 7, 15, 31$, avremo i valori immagazzinati qui sotto nel vettore `err`

```
>> format short e
>> c=187.8428; err=[];
>> for n=2:5, N=2^n-1; h=1/(N+1); h2=h^2; err=[err; c*h2]; end
>> err

err =

  1.1740e+001
  2.9350e+000
```

```
  7.3376e-001
  1.8344e-001

>>
```

Lanciando la demo `demopoisson5pts`, per `demoexample=2`, abbiamo

```
>> demopoisson5pts

          [n]:    4 [ERR]: 9.75e-002 [RATIO]: 0.74
          [n]:    8 [ERR]: 3.20e-002 [RATIO]: 3.04
          [n]:   16 [ERR]: 9.05e-003 [RATIO]: 3.54
          [n]:   32 [ERR]: 2.45e-003 [RATIO]: 3.69
>>
```

1. Come ci si aspettava la maggiorazione è realizzata, ma purtroppo come stima è abbastanza conservativa.
2. Nella colonna `[RATIO]` abbiamo indicato il rapporto e_{2h}/e_h dove e_h è l'errore assoluto compiuto dal metodo a 5 punti con passo h (ovvero la quantità esposte nella colonna `[ERR]` nella stessa riga di h). Il fatto che la *ratio* sia 4 non è sorprendente. Infatti se l'errore decresce come h^2 si può supporre che sia $e_h \approx \hat{c}h^2$ per qualche $\hat{c}$ indipendente da h e quindi

$$\frac{e_{2h}}{e_h} \approx \frac{\hat{c}(2h)^2}{\hat{c}h^2} \approx 4.$$

5.3 Problema 3.

Consideriamo un la risoluzione dell'equazione di Poisson nel quadrato unitario $\Omega = [0,1] \times [0,1]$ con un metodo alle differenze.

$$\begin{cases} \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = (-2\pi^2)\sin(\pi x)\sin(\pi y), \ (x,y) \in \Omega \\ u(x,0) = 0, \ x \in [0,1] \\ u(x,1) = 0, \ x \in [0,1] \\ u(0,y) = 0, \ y \in [0,1] \\ u(1,y) = 0, \ y \in [0,1] \end{cases} \tag{20}$$

la cui soluzione è $u(x,y) = \sin(\pi x)\sin(\pi y)$. Ripetendo la stima basata sulle derivate quarte della soluzione u abbiamo facilmente

$$c \leq (1/24)\,2\pi^4 \approx 8.1174$$

da cui

```
>> c=8.1174;
```

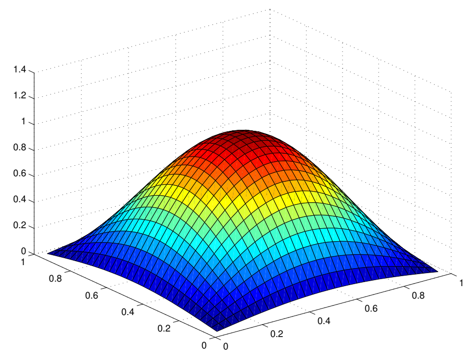

Figura 7: Soluzione del problema 3.

```
>> err=[]; for n=2:5, N=2^n; N=N-1; h=1/(N+1); h2=h^2; err=[err; c*h2]; end
>> format short e
>> err

err =

  5.0734e-001
  1.2683e-001
  3.1709e-002
  7.9271e-003

>>
```

Lanciando la demo `demopoisson5pts`, per `demoexample=3`, abbiamo

```
>> demopoisson5pts

          [n]:    4 [ERR]: 3.04e-002 [RATIO]: 2.39
          [n]:    8 [ERR]: 9.91e-003 [RATIO]: 3.06
          [n]:   16 [ERR]: 2.83e-003 [RATIO]: 3.51
          [n]:   32 [ERR]: 7.54e-004 [RATIO]: 3.75
>>
```

Rispetto al caso precedente la stima è più precisa, e la ratio di circa 4 ci dice che la convergenza è ancora dell'ordine di h^2.

6 Facoltativo: Meshgrid e i grafici bidimensionali

Non è infrequente di voler rappresentare graficamente una funzione definita su un quadrato Ω, supponiamo per semplicità $\Omega = [0,1] \times [0,1]$. Vediamone un esempio, proponendoci di voler plottare il piano

$$f(x,y) = 3x + y.$$

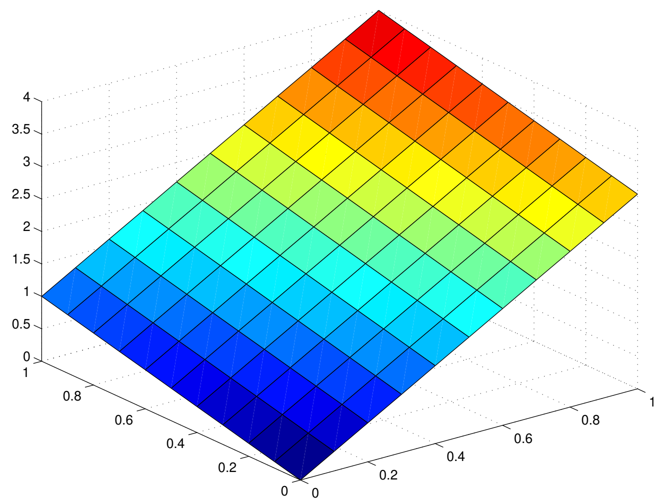

Figura 8: Il piano ottenuto con i comandi `meshgrid` e `surf`.

.

Si tratta di un piano passante per l'origine P_1=(0,0,0), per il punto P_2=(0,1,1) e P_3=(1,1,4). Descriviamo il quadrato Ω tramite una griglia non troppo fine, diciamo con passo $h_x = h_y = h = 1/10$. Vediamone i dettagli che salviamo in `my_2D_plot`:

```
h=1/10;

f=inline('3*x+y','x','y');

% DEFINIAMO IL QUADRATO [a,b] x [c,d].
a=0; b=1;
c=0; d=1;

x=0:h:1;
y=0:h:1;

[X,Y]=meshgrid(x,y);
Z=feval(f,X,Y);

surf(X,Y,Z);
```

Il risultato è descritto in figura ed è in effetti un piano che passa per P_1, P_2, P_3.

7 Online

Alcuni utili links online sugli argomenti trattati:

http://www-groups.dcs.st-and.ac.uk/~history/Biographies/Laplace.html
http://www-groups.dcs.st-and.ac.uk/~history/Biographies/Poisson.html
http://en.wikipedia.org/wiki/Laplace
http://en.wikipedia.org/wiki/Laplace\%27s_equation
http://en.wikipedia.org/wiki/Poisson
http://en.wikipedia.org/wiki/Poisson_equation

8 Frasi famose

Alcune frasi famose:

- Napoleon: You have written this huge book on the system of the world without once mentioning the author of the universe.

  Laplace: Sire, I had no need of that hypothesis.

- (about Napier's logarithms) by shortening the labours doubled the life of the astronomer (Laplace).

- Read Euler: he is our master in everything (Laplace).

- Life is good for only two things, discovering mathematics and teaching mathematics (Poisson).

References

[1] K. Atkinson, *Introduction to Numerical Analysis*, Wiley, 1989.

[2] K. Atkinson e W. Han, *Theoretical Numerical Analysis*, Springer, 2001.

[3] D. Bini, M. Capovani e O. Menchi, *Metodi numerici per l'algebra lineare*, Zanichelli, 1988.

[4] V. Comincioli, *Analisi Numerica, metodi modelli applicazioni*, Mc Graw-Hill, 1990.

[5] S.D. Conte e C. de Boor, *Elementary Numerical Analysis, 3rd Edition*, Mc Graw-Hill, 1980.

[6] D.H. Griffel, *Applied functional analysis*, Dover publications, 2002.

[7] The MathWorks Inc., *Numerical Computing with Matlab*, http://www.mathworks.com/moler.

[8] A. Quarteroni e F. Saleri, *Introduzione al calcolo scientifico*, Springer Verlag, 2006.

[9] P.A. Raviart e J.M. Thomas, *Introduzione all'analisi numerica delle equazioni alle derivate parziali*, Masson, 1989.

[10] Mac Tutor (Laplace),
http://www-groups.dcs.st-and.ac.uk/h̃istory/Biographies/Laplace.html.

[11] Mac Tutor (Poisson),
http://www-groups.dcs.st-and.ac.uk/h̃istory/Biographies/Poisson.html.

[12] A. Suli e D. Mayers, *An Introduction to Numerical Analysis*, Cambridge University Press, 2003.

[13] Wikipedia (Equazione di Poisson),
http://it.wikipedia.org/wiki/Equazione_di_Poisson.

[14] Wikipedia (Laplace),
http://it.wikipedia.org/wiki/Equazione_di_Laplace.

[15] Wikipedia (Laplace),
http://en.wikipedia.org/wiki/Laplace.

[16] Wikipedia (Laplace equation),
http://en.wikipedia.org/wiki/Laplace%27s_equation.

[17] Wikipedia (Poisson),
http://en.wikipedia.org/wiki/Poisson.

[18] Wikipedia (Poisson equation),
http://en.wikipedia.org/wiki/Poisson_equation.

[19] Wikipedia (Spazio di Sobolev),
http://it.wikipedia.org/wiki/Spazio_di_Sobolev.