Laboratorio 1

Esercizio 1

Esercizio 1.1

Vogliamo costruire una function per implementare il metodo alle differenze finite centrate per un problema di Poisson con condizioni di Ditichlet non omogenee.

I parametri in input sono

- La lunghezza dell'intervallo L
- Il numero di sottointervalli che vogliamo N
- u0 è il valore della soluzione nel punto x=0
- uL è il valore della soluzione nel punto x=L

Vogliamo restituire come output

- x vettore con i punti della griglia in cui si approssima la soluzione
- u vettore con la soluzione numerica

Creiamo la griglia e definiamo il passo della griglia

```
fprintf("\n------------ Es. 1.1 -------------\n");
```

```
------------ Es. 1.1 -------------
```

```
fprintf("x = linspace(0,L,N+1)");
```

```
x = linspace(0,L,N+1)
```

```
fprintf(" h = L/N");
```

```
 h = L/N
```

Infatti se vogliamo N sottointervalli, allora avremo N-1 punti interni + 2 punti estremi per un totale di N+1 punti

Costruiamo la matrice A del metodo

Dalla teoria sappiamo che per i nodi interni otteniamo il seguente sistema, applicando lo schema alle differenze finite centrate

$$
\begin{cases}
\dfrac{u_2 - 2u_1 + u_0}{h^2} & = F_1, \\
\dfrac{u_3 - 2u_2 + u_1}{h^2} & = F_2, \\
 & \vdots \\
\dfrac{u_N - 2u_{N-1} + u_{N-2}}{h^2} & = F_{N-1}.
\end{cases}
$$

I valori u0 e uN sono noti perchè forniti dalle condizioni di Dirichlet.

Quindi A sarà una matrice N-1 x N-1 . In notazione matriciale, tale sistema diventa

$\mathbf{A}\mathbf{u}_* = \mathbf{F}_{\mathbf{bc}}$

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 0 & \cdots & \cdots & \cdots & 0 \\ -1 & 2 & -1 & 0 & \cdots & \cdots & \vdots \\ 0 & -1 & 2 & -1 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \cdots & \vdots \\ \cdots & \cdots & \cdots & -1 & 2 & -1 & 0 \\ \cdots & \cdots & \cdots & \cdots & -1 & 2 & -1 \\ 0 & \cdots & \cdots & \cdots & 0 & -1 & 2 \end{bmatrix}, \quad \mathbf{F}_{\mathbf{bc}} = \begin{bmatrix} h^2F_1 + u_0 \\ h^2F_2 \\ \vdots \\ \vdots \\ h^2F_{N-2} \\ h^2F_{N-1} + u_L \end{bmatrix}.$$

```
fprintf("e = ones(N-1,1)");
```

```
e = ones(N-1,1)
```

```
fprintf("A = spdiags([-e 2*e -e],[-1 0 1],N-1,N-1)");
```

```
A = spdiags([-e 2*e -e],[-1 0 1],N-1,N-1)
```

Con spdiags(B,d,m,n) abbiamo che

- B è una matrice le cui colonne vengono usate come valori per le diagonali
- d è un vettore dove si indicano i valori delle diagonali in cui inserire le colonne di B
- m,n dimensioni della matrice

Costruzione termine noto F del metodo

Dalle matrici mostrate sopra è immediato il codice che segue

```
fprintf("F = f(x(2:end-1))");
```

```
F = f(x(2:end-1))
```

```
fprintf("F = F*(h^2)");
```

```
F = F*(h^2)
```

```
fprintf("F(1) = F(1) + u0");
```

```
F(1) = F(1) + u0
```

```
fprintf("F(end) = F(end) +uL");
```

```
F(end) = F(end) +uL
```

Risolvo il sistema lineare

```
fprintf("u = A/F"); % sarebbe \
```

```
u = A/F
```

```
fprintf("u = [u0; u; uL]");
```

```
u = [u0; u; uL]
```

Esercizio 1.2

Adesso si applica il metodo creato e implementato per un esercizio "pratico"

Inserisco i dati forniti dal testo

```
%%
fprintf("\n----------------- Es 1.2 -------------------------\n")
```

```
---------------- Es 1.2 ------------------------
```

```
L = 1;
f = @(x) 96*x.^2;
u0 = 0; uL = -1;
N = 40;
u_ex = @(x) 7*x -8*x.^4;
```

Otteniamo la soluzione numerica

```
[x,uh] = poisson_dirichlet_centrato(L,N,u0,uL,f);
```

Effettuo un confronto grafico

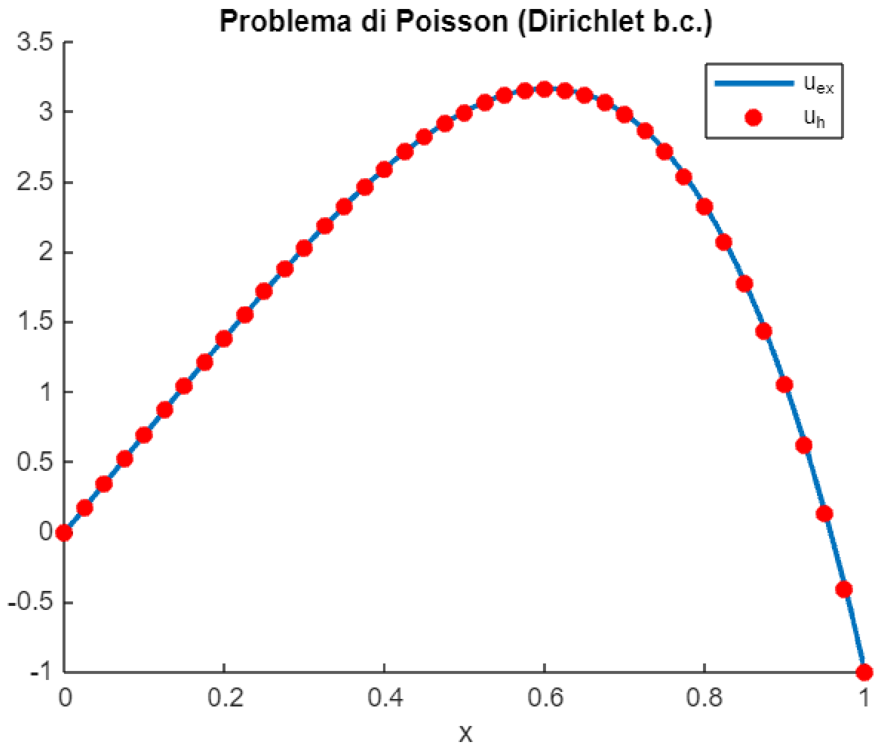
```
figure
hold on

xplot = linspace(0,L,1000);
plot(xplot,u_ex(xplot),'LineWidth',2);
plot(x,uh,'or','MarkerFaceColor','r');
```

Dove i comandi scritti

- figure permette di aprire una nuova finestra grafica
- hold on permette di sovrapporre più grafici nello stesso plot
- LineWidth permette di scegliere lo spessore della linea
- or permette di disegnare mediante cerchi (o) rossi (r)
- MarkerFaceColor, r permette di riempire i pallini col colore rosso

```
title('Problema di Poisson (Dirichlet b.c.)');
xlabel('x');
legend('u_{ex}','u_h');
```



Calcolo gli errori

Le formule proposte nella consegna per il calcolo prevedono l'uso della norma del massimo o della norma h definite così

$$e_\infty = \max_{i=0,\ldots,N+1} |u_i - u_{\text{ex}}(x_i)| \qquad \text{ed} \qquad e_h = \sqrt{h \sum_{i=1}^{N+1} |u_i - u_{\text{ex}}(x_i)|^2}$$

```
errore = max(abs( uh - u_ex(x)));
fprintf("Errore %.2e.\n",errore);
```

```
Errore 1.25e-03.
```

Esercizio 1.3

L'ultima parte dell'esercizio prevede il calcolo dell'errore e della stima dell'ordine di convergenza del metodo

```
fprintf("\n----------------- Es 1.3 -------------------------\n")
```

```
----------------- Es 1.3 -------------------------
```

```

NN = [50, 100, 200, 400,800];

errori_max = zeros(length(NN),1);
errori_h = zeros(length(NN),1);

for i = 1:length(NN)
    N = NN(i);
```

```
    h = L/N;

    [x,uh] = poisson_dirichlet_centrato(L,N,u0,uL,f);

    errori_max(i) = max(abs( uh - u_ex(x)));
    errori_h(i) = sqrt(h)*norm(uh - u_ex(x));
end
```

Abbiamo così tutti gli errori per i diversi valori di N.

Per il cacolo delle stime dell'errore usiamo le seguenti formule

$$p_{\max} \approx \log_2 \frac{e_\infty(h)}{e_\infty(h/2)}, \qquad p_h \approx \log_2 \frac{e_h(h)}{e_h(h/2)}.$$

```
p_max = log2( errori_max(1:end-1) ./ errori_max(2:end) );
p_h = log2( errori_h(1:end-1) ./ errori_h(2:end) );

fprintf("p_max:\n");
```

```
p_max:
```

```
disp(p_max);
```

```
    2.0000
    2.0000
    2.0000
    2.0000
```

```

fprintf("p_h:\n");
```

```
p_h:
```

```
disp(p_h);
```

```
    2.0000
    2.0000
    2.0000
    2.0000
```

Esercizio 2

In questo caso abbiamo un problema con condizioni al contorno misto.

In questo caso il primo nodo u0 ci viene fornita la derivata prima. In questo caso quindi le incognite del problema discreto sono

$$\mathbf{u}_* := [u_0, \ldots, u_{N-1}]^\top \in \mathbb{R}^N,$$

Per i nodi interni i=1,...,N-1 abbiamo

$$\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} = F(x_i).$$

Come prima, osserviamo che se i=N-1 allora u_{i+1} = u_N = u_L è noto e quindi

$$\frac{-2u_{N-1}+u_{N-2}}{h^2} = F(x_{N-1}) + \frac{u_L}{h^2}.$$

Abbiamo N-1 equazioni ed N incognite: per ottenere un problema ben posto va aggiunta un'altra equazione. Si presentano due metodi:

Esercizio 2.1.1

METODO UPWIND: consideriamo una discretizzazione della condizione di Neuman usando una differenza finita in avanti

$$\frac{u_1-u_0}{h} = \alpha_0$$

Implementiamo tale metodo

```
fprintf("\n------------ Es. 2.1.1 -------------\n");
```

```
------------ Es. 2.1.1 -------------
```

```
fprintf("x = linspace(0,L,N+1)");
```

```
x = linspace(0,L,N+1)
```

```
fprintf(" h = L/N");
```

```
 h = L/N
```

La matrice A e il termine noto diventano:

$$A = \begin{bmatrix} -1 & 1 & \cdots & & \cdots & \cdots & 0 \\ -1 & 2 & -1 & 0 & \cdots & \cdots & 0 \\ 0 & -1 & 2 & -1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & -1 & 2 & -1 & 0 \\ 0 & \cdots & \cdots & 0 & -1 & 2 & -1 \\ 0 & \cdots & \cdots & & \cdots & -1 & 2 \end{bmatrix},$$

$$\mathbf{F}_{\mathrm{bc}} = \begin{bmatrix} h\alpha_0 \\ h^2F_1 \\ \vdots \\ \vdots \\ h^2F_{N-2} \\ h^2F_{N-1}+u_L \end{bmatrix}.$$

```
fprintf("e = ones(N,1)");
```

```
e = ones(N,1)
```

```
fprintf("A = spdiags([-e 2*e -e],[-1 0 1],N,N)");
```

```
A = spdiags([-e 2*e -e],[-1 0 1],N,N)
```

```

fprintf("A(1,1) = -1");
```

```
A(1,1) = -1
```

```
fprintf("A(1,2) = 1");
```

```
A(1,2) = 1
```

```

fprintf("F = f(x(1:end-1))");
```

```
F = f(x(1:end-1))
```

```
fprintf("F = F*(h^2);");
```

```
F = F*(h^2);
```

```

fprintf("F(1)    = h*du0dx;")
```

```
F(1)    = h*du0dx;
```

```
fprintf("F(end) = F(end) + uL;")
```

```
F(end) = F(end) + uL;
```

```

fprintf("u = A / F;") %sarebbe \
```

```
u = A / F;
```

```
fprintf("u = [u; uL];")
```

```
u = [u; uL];
```

Svolgo Esercizio con UPWIND

Inserisco i dati forniti

```
fprintf("\n \n");
```

```

L = 1;
f = @(x) 4*(pi^2)*cos(2*pi*x);
du0dx = 2;
uL = 1;
```

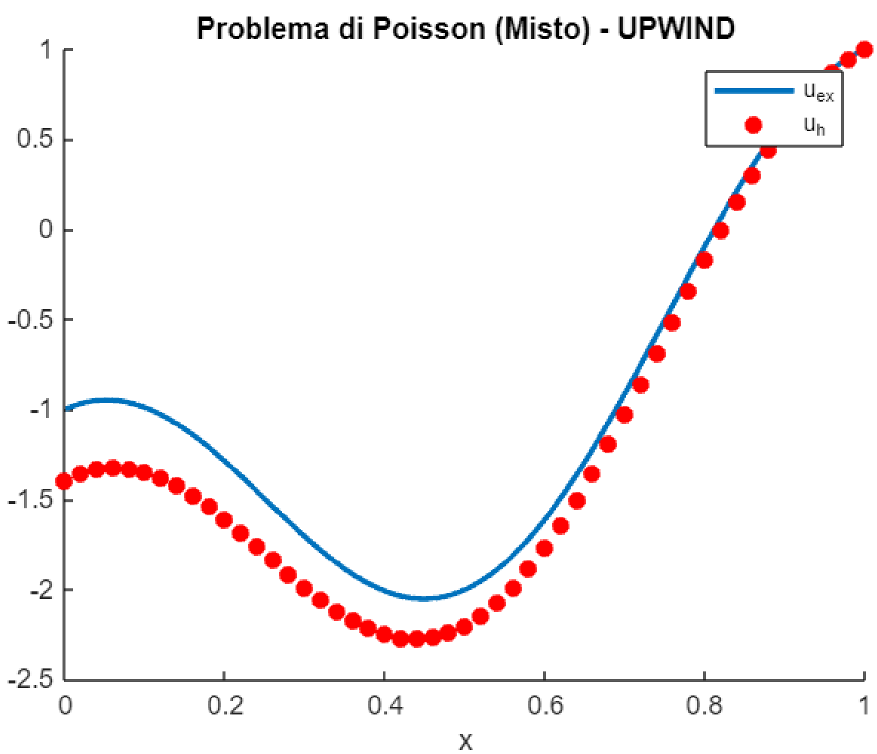
```
N = 50;
u_ex = @(x) cos(2*pi*x) + 2*x -2;
```

Risolvo col metodo upwind e plotto

```
[x,uh] = poisson_misto_upwind(L,N,du0dx,uL,f);

xplot = linspace(0,L,1000);
figure
hold on
plot(xplot,u_ex(xplot),"LineWidth",2);
plot(x,uh,'or','MarkerFaceColor','r');

title('Problema di Poisson (Misto) - UPWIND');
xlabel('x');
legend('u_{ex}','u_h')
```



Errore assoluto

```
errore = max(abs(uh - u_ex(x)));
fprintf("Errore: %.2e.\n", errore)
```

```
Errore: 3.95e-01.
```

Studio il rate di convergenza del metodo UPWIND

```
NN = [50, 100, 200, 400, 800];

errori_max = zeros(length(NN), 1);
errori_h = zeros(length(NN), 1);

for i = 1:length(NN)
    N = NN(i);
    h = L/N;

    [x, uh] = poisson_misto_upwind(L, N, du0dx, uL, f);

    errori_max(i) = max(abs(uh - u_ex(x)));
    errori_h(i) = sqrt(h)*norm(uh - u_ex(x));
end

p_max = log2(errori_max(1:end-1) ./ errori_max(2:end));
p_h = log2(errori_h(1:end-1) ./ errori_h(2:end));

fprintf("p_max:\n");
```

```
p_max:
```

```
disp(p_max);
```

```
    1.0000
    1.0000
    1.0000
    1.0000
```

```


fprintf("p_h:\n");
```

```
p_h:
```

```
disp(p_h);
```

```
    1.0141
    1.0071
    1.0036
    1.0018
```

L'approssimazione del primo ordine della condizione di Neumann deteriora l'ordine di convergenza di tutto lo schema

Esercizio 2.2.1

Il secondo metodo è il metodo dei **GHOST NODE**

Al fine di non deteriorare l'ordine di convergenza garantito dai nodi interni, lo schema alternativo prevede la discretizzazione della condizione di Neumann attraverso lo scgema centrato del secondo ordine

$$\frac{u_1 - u_{-1}}{2h} = \alpha_0$$

L'introduzione del nodo fantasma introduce una incognita in più. Introduco quindi una condizione in più imponendo l'equazione differenziale -u" = f anche nel nodo x_0 ottenendo così:

$$\begin{cases} -u_{-1} + u_1 = 2h\alpha_0, \\ -u_1 + 2u_0 - u_{-1} = h^2 F_0, \\ -u_{i+1} + 2u_i - u_{i-1} = h^2 F_i, \qquad i = 1, \ldots, N-2 \\ 2u_{N-1} - u_{N-2} = h^2 F_{N-1} + u_L, \end{cases}$$

Eliminando il nodo fantasma si ottiene

$$\begin{cases} u_1 - u_0 = \frac{1}{2} h^2 F_0 - h\alpha_0, \\ -u_{i+1} + 2u_i - u_{i-1} = h^2 F_i, \qquad i = 1, \ldots, N-2 \\ 2u_{N-1} - u_{N-2} = h^2 F_{N-1} + u_L. \end{cases}$$

Così da ottenere la seguente forma matriciale

$$A = \begin{bmatrix} 1 & -1 & \cdots & & & \cdots & 0 \\ -1 & 2 & -1 & 0 & \cdots & \cdots & 0 \\ 0 & -1 & 2 & -1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & -1 & 2 & -1 & 0 \\ 0 & \cdots & \cdots & 0 & -1 & 2 & -1 \\ 0 & \cdots & \cdots & \cdots & 0 & -1 & 2 \end{bmatrix}, \quad \mathbf{F}_{\mathrm{bc}} = \begin{bmatrix} \frac{1}{2} h^2 F_0 - h\alpha_0 \\ h^2 F_1 \\ \vdots \\ \vdots \\ h^2 F_{N-2} \\ h^2 F_{N-1} + u_L \end{bmatrix}.$$

```
fprintf("x = linspace(0, L, N+1)';")
```

```
x = linspace(0, L, N+1)';
```

```
fprintf("h = L/N;")
```

```
h = L/N;
```

```
% Costruzione della matrice A
fprintf("e = ones(5*N, 1);")
```

```
e = ones(5*N, 1);
```

```
fprintf("A = spdiags([-e 2*e -e],[-1 0 1], N, N);")
```

```
A = spdiags([-e 2*e -e],[-1 0 1], N, N);
```

```
% Correzione della matrice A (ghost node vicino a x = 0)
fprintf("A(1, 1) = 1;")
```

```
A(1, 1) = 1;
```

```
fprintf("A(1, 2) = -1;")
```

```
A(1, 2) = -1;
```

```

% Costruzione del termine noto F
fprintf("F = f(x(1:end-1));")
```

```
F = f(x(1:end-1));
```

```
fprintf("F = F*(h^2);")
```

```
F = F*(h^2);
```

```

% Correzione del termine noto (inclusione delle condizioni al bordo)
fprintf("F(1)   = 0.5*F(1)   - h*du0dx; ")% Neumann (ghost)
```

```
F(1)   = 0.5*F(1)   - h*du0dx;
```

```
fprintf("F(end) = F(end) + uL;")          % Dirichlet
```

```
F(end) = F(end) + uL;
```

```

% Risoluzione del sistema lineare
fprintf("u = A\F;")
```

```
Warning: Escaped character '\F' is not valid. See 'doc sprintf' for supported special characters.
u = A
```

```
fprintf("u = [u; uL];")
```

```
u = [u; uL];
```

Esercizio 2.2.2

```
% Dati del problema
L = 1;
du0dx = 2;
uL = 1;
f = @(x) 4*pi^2*cos(2*pi*x);

% Discretizzazione scelta
N = 50;


% Calcolo della soluzione numerica
[x, uh] = poisson_misto_centrato(L, N, du0dx, uL, f);
```

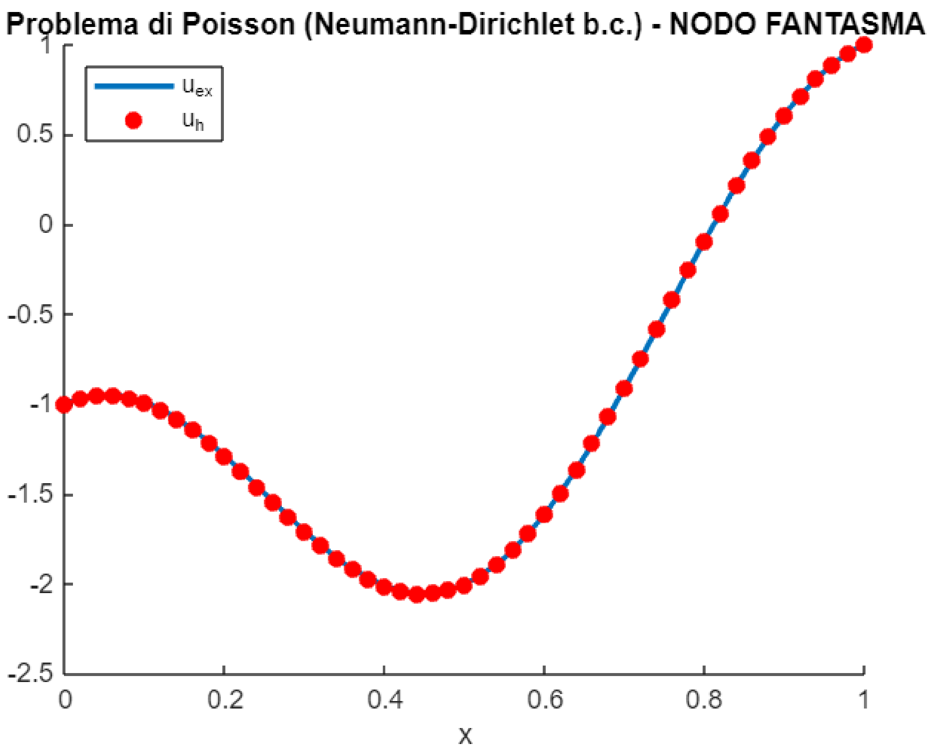
```
% Soluzione esatta (per confronto)
uex = @(x) cos(2*pi*x) + 2*x - 2;


% Confronto grafico
figure
hold on

xplot = linspace(0, L, 1000);
plot(xplot, uex(xplot), 'linewidth', 2);

plot(x, uh, 'or', 'MarkerFaceColor', 'r');

title('Problema di Poisson (Neumann-Dirichlet b.c.) - NODO FANTASMA')
xlabel('x')
legend('u_{ex}', 'u_h', 'Location', 'northwest');
```




```
errore = max(abs(uh - uex(x)));
fprintf("Errore: %.2e.\n", errore)
```

```
Errore: 2.63e-03.
```

```
%% Es. 2.2 (Rate di convergenza)

fprintf("\n--- Es. 2.2 ---\n")
```

--- Es. 2.2 ---

```
NN = [50, 100, 200, 400, 800];

errori_max = zeros(length(NN), 1);
errori_h = zeros(length(NN), 1);

for i = 1:length(NN)
    N = NN(i);
    h = L/N;

    [x, uh] = poisson_misto_centrato(L, N, du0dx, uL, f);

    errori_max(i) = max(abs(uh - uex(x)));
    errori_h(i) = sqrt(h)*norm(uh - uex(x));
end

p_max = log2(errori_max(1:end-1) ./ errori_max(2:end));
p_h = log2(errori_h(1:end-1) ./ errori_h(2:end));

fprintf("p_max:\n");
```

```
p_max:
```

```
disp(p_max);
```

```
    2.0009
    2.0002
    2.0001
    2.0000
```

```
fprintf("p_h:\n");
```

```
p_h:
```

```
disp(p_h);
```

```
    2.0009
    2.0002
    2.0001
    2.0000
```