

Esercitazione 9

Interpolazione con elemento di Clough-Tocher

Esercizio 1

Fornire un esempio di interpolazione con l'elemento di Clough-Tocher.

Il seguente codice valuta l'interpolante di Clough-Tocher in modo simbolico:

```
% Calcolo dell'elemento di Clough-Tocher
close all
clear all
clc
```

```
% DATI GEOMETRICI
% _____
```

```
% generiche variabili spaziali
syms x y
```

```
for i=1:3
    % nodi del triangolo x-i
    eval(['syms x-',num2str(i)]);
    eval(['syms y-',num2str(i)]);
end
```

```
% Baricentro
syms x_m y_m
x_m = (x_1 + x_2 + x_3)/3;
y_m = (y_1 + y_2 + y_3)/3;
```

```
% Punto medio dei singoli semi-lati
syms x_12 x_13 x_23 x_m1 x_m2 x_m3
syms y_12 y_13 y_23 y_m1 y_m2 y_m3
```

```
x_12 = (x_1+x_2)/2; y_12 = (y_1+y_2)/2;
x_13 = (x_1+x_3)/2; y_13 = (y_1+y_3)/2;
```

```

x_23 = (x_3+x_2)/2; y_23 = (y_3+y_2)/2;

x_m1 = (x_m+x_1)/2; y_m1 = (y_m+y_1)/2;
x_m2 = (x_m+x_2)/2; y_m2 = (y_m+y_2)/2;
x_m3 = (x_m+x_3)/2; y_m3 = (y_m+y_3)/2;

% Normali di un triangolo (lati esterni)
syms nx_1 ny_1 nx_2 ny_2 nx_3 ny_3
syms nx_4 ny_4 nx_5 ny_5 nx_6 ny_6

nx_1 = (y_3-y_2)/sqrt((y_2-y_3)^2+(x_2-x_3)^2);
ny_1 = -(x_3-x_2)/sqrt((y_2-y_3)^2+(x_2-x_3)^2);

nx_2 = (y_1-y_3)/sqrt((y_1-y_3)^2+(x_1-x_3)^2);
ny_2 = -(x_1-x_3)/sqrt((y_1-y_3)^2+(x_1-x_3)^2);

nx_3 = (y_2-y_1)/sqrt((y_2-y_1)^2+(x_2-x_1)^2);
ny_3 = -(x_2-x_1)/sqrt((y_2-y_1)^2+(x_2-x_1)^2);

nx_4 = -(y_m-y_3)/sqrt((y_m-y_3)^2+(x_m-x_3)^2);
ny_4 = (x_m-x_3)/sqrt((y_m-y_3)^2+(x_m-x_3)^2);

nx_5 = (y_1-y_m)/sqrt((y_1-y_m)^2+(x_1-x_m)^2);
ny_5 = -(x_1-x_m)/sqrt((y_1-y_m)^2+(x_1-x_m)^2);

nx_6 = (y_2-y_m)/sqrt((y_2-y_m)^2+(x_2-x_m)^2);
ny_6 = -(x_2-x_m)/sqrt((y_2-y_m)^2+(x_2-x_m)^2);

% DATI DI INTERPOLAZIONE
% _____

for i=1:3
    eval(['syms f_', num2str(i)]);
    eval(['syms fx_', num2str(i)]);
    eval(['syms fy_', num2str(i)]);
    eval(['syms fn_', num2str(i)]);
end

% INCOGNITE DELL'INTERPOLANTE
% _____

allvar = [];
for j=1:3
    for i=1:10
        strvar = ['p', num2str(j), '-', num2str(i)];
        eval(['syms ', strvar]);
    end
end

```

```

        eval(['allvar = [allvar,',strvar,'];']);
    end
end

% Definizione dei tre polinomi e relative derivate
for i=1:3
    eval(['syms P_',num2str(i)]);
    eval(['syms Px_',num2str(i)]);
    eval(['syms Py_',num2str(i)]);
end
for i=1:3
    % grado 0
    grad = 0;
    strpol = ['p',num2str(i),'_1'];

    index = 2;
    % in base al grado
    for grad = 1:3,
        for j=0:grad
            strpol = [strpol, ' + p',num2str(i),'_',num2str(index),'*x^',num2str(grad-j),
            index = index + 1;
        end
    end

    eval(['P_',num2str(i),' = ',strpol]);
end

% Derivate dei polinomi
for i=1:3
    eval(['Px_',num2str(i),' = diff(P_',num2str(i),'_x)']);
    eval(['Py_',num2str(i),' = diff(P_',num2str(i),'_y)']);
end

% EQUAZIONI
% -----

alleq = [];
for i=1:30
    streq = ['eq_',num2str(i)];
    eval(['syms ',streq]);
    eval(['alleq = [alleq,',streq,'];']);
end

% POLINOMIO P1
% -----
eq_1 = subs(P_1,{x,y},{x_2,y_2}) - f_2;
eq_2 = subs(P_1,{x,y},{x_3,y_3}) - f_3;

eq_3 = subs(Px_1,{x,y},{x_2,y_2}) - fx_2;

```

```

eq_4 = subs(Px_1,{x,y},{x_3,y_3}) - fx_3;

eq_5 = subs(Py_1,{x,y},{x_2,y_2}) - fy_2;
eq_6 = subs(Py_1,{x,y},{x_3,y_3}) - fy_3;

eq_7 = subs(Px_1,{x,y},{x_23,y_23})*nx_1 - subs(Py_1,{x,y},{x_23,y_23})*ny_1 - fn_1;

% POLINOMIO P2
% -----
eq_8 = subs(P_2,{x,y},{x_1,y_1}) - f_1;
eq_9 = subs(P_2,{x,y},{x_3,y_3}) - f_3;

eq_10 = subs(Px_2,{x,y},{x_1,y_1}) - fx_1;
eq_11 = subs(Px_2,{x,y},{x_3,y_3}) - fx_3;

eq_12 = subs(Py_2,{x,y},{x_1,y_1}) - fy_1;
eq_13 = subs(Py_2,{x,y},{x_3,y_3}) - fy_3;

eq_14 = subs(Px_2,{x,y},{x_13,y_13})*nx_2 - subs(Py_2,{x,y},{x_13,y_13})*ny_2 - fn_2;

% POLINOMIO P3
% -----
eq_15 = subs(P_3,{x,y},{x_1,y_1}) - f_1;
eq_16 = subs(P_3,{x,y},{x_2,y_2}) - f_2;

eq_17 = subs(Px_3,{x,y},{x_1,y_1}) - fx_1;
eq_18 = subs(Px_3,{x,y},{x_2,y_2}) - fx_2;

eq_19 = subs(Py_3,{x,y},{x_1,y_1}) - fy_1;
eq_20 = subs(Py_3,{x,y},{x_2,y_2}) - fy_2;

eq_21 = subs(Px_3,{x,y},{x_12,y_12})*nx_3 - subs(Py_3,{x,y},{x_12,y_12})*ny_3 - fn_3;

% NODO CENTRALE
% -----
eq_22 = subs(P_1,{x,y},{x_m,y_m}) - subs(P_2,{x,y},{x_m,y_m});
eq_23 = subs(P_2,{x,y},{x_m,y_m}) - subs(P_3,{x,y},{x_m,y_m});

eq_24 = subs(Px_1,{x,y},{x_m,y_m}) - subs(Px_3,{x,y},{x_m,y_m});
eq_25 = subs(Py_1,{x,y},{x_m,y_m}) - subs(Py_3,{x,y},{x_m,y_m});

eq_26 = subs(Px_2,{x,y},{x_m,y_m}) - subs(Px_3,{x,y},{x_m,y_m});
eq_27 = subs(Py_2,{x,y},{x_m,y_m}) - subs(Py_3,{x,y},{x_m,y_m});

% NODI LUNGO I LATI INTERNI DEL TRIANGOLO
% -----
eq_28 = (subs(Px_1,{x,y},{x_m3,y_m3})*nx_4 + subs(Py_1,{x,y},{x_m3,y_m3})*ny_4) - (subs
eq_29 = (subs(Px_1,{x,y},{x_m2,y_m2})*nx_6 + subs(Py_1,{x,y},{x_m2,y_m2})*ny_6) - (subs
eq_30 = (subs(Px_2,{x,y},{x_m1,y_m1})*nx_5 + subs(Py_2,{x,y},{x_m1,y_m1})*ny_5) - (subs

```

```

% Risoluzione
streq = char(alleg(1));
for i=2:length(alleg)
    streq = [streq, ',', char(alleg(i))];
end

strvar = char(allvar(1));
for i=2:length(allvar)
    strvar = [strvar, ',', char(allvar(i))];
end

%eval(['S = solve(',streq,',',',',strvar,')']);

J = jacobian([eq_1,eq_2,eq_3,eq_4,eq_5,...
    eq_6,eq_7,eq_8,eq_9,eq_10,...
    eq_11,eq_12,eq_13,eq_14,eq_15,...
    eq_16,eq_17,eq_18,eq_19,eq_20,...
    eq_21,eq_22,eq_23,eq_24,eq_25,...
    eq_26,eq_27,eq_28,eq_29,eq_30],allvar);

vectorize(J)

F = -([eq_1,eq_2,eq_3,eq_4,eq_5,...
    eq_6,eq_7,eq_8,eq_9,eq_10,...
    eq_11,eq_12,eq_13,eq_14,eq_15,...
    eq_16,eq_17,eq_18,eq_19,eq_20,...
    eq_21,eq_22,eq_23,eq_24,eq_25,...
    eq_26,eq_27,eq_28,eq_29,eq_30] .' - J*allvar. ');

for i=1:30
    F(i) = simple(F(i));
end

vectorize(F)

```

Si veda lo script principale main_ct per un esempio di interpolazione.

```

% programma dimostrativo base per l'elemento di Clough-Tocher
close all
clear all
clc

% funzione test e sue derivate direzionali
f = inline('x.^2 + y.^2 + x.*y','x','y');
fx = inline('2*x + y','x','y');
fy = inline('2*y + x','x','y');

% I punti di un triangolo
xtri = [0 2 0]';

```

```

ytri = [0 0 1]';

% Restituisci i dati geometrici del triangolo
gdata = getgeomdata(xtri,ytri);

% Restituisce i dati per l'interpolante
fdata = getfdata(gdata,f,fx,fy);

% Disegno dati geometrici relativi ad un triangolo
plotgdata(gdata);

% Calcolo dell'interpolante
P = ct_interp(gdata,fdata);

% Interpolazione sul trinagolo
% [X,Y] = meshgrid(sort(rand(1,100))*2,sort(rand(1,50))*1);
[X,Y] = meshgrid(0:0.01:2,0:0.01:1);
xydata = [X(:),Y(:)];

Zval = ct_eval(P,gdata,fdata,xydata);
Zval = reshape(Zval,size(X));

figure;
mesh(X,Y,Zval);

```

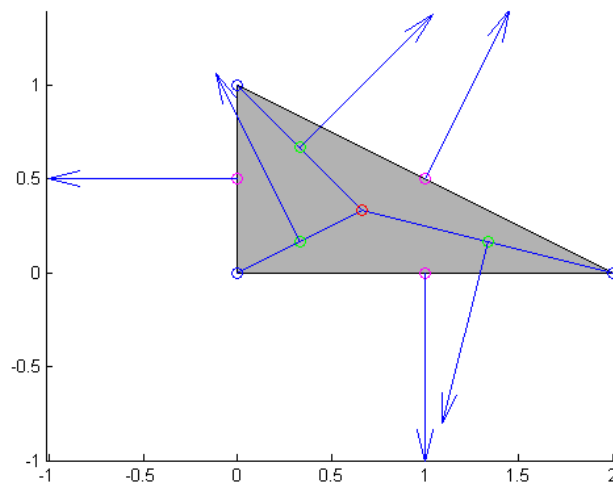


Figura 1: Elemento di Clough-Tocher.

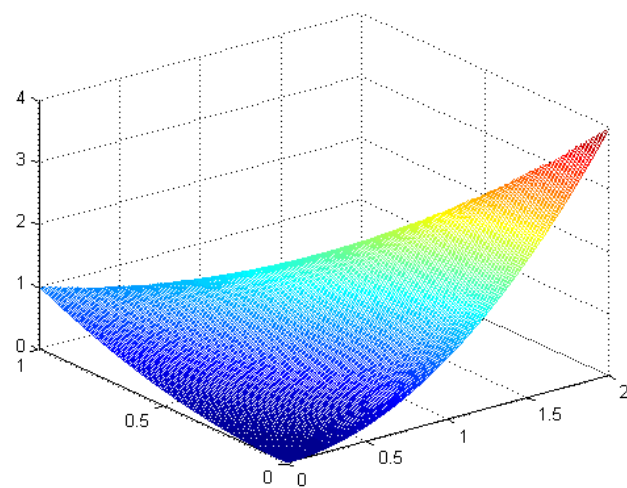


Figura 2: Esempio di interpolazione sull'elemento di Clough-Tocher.