RELAZIONE OTTAVA ESERCITAZIONE

NOTE LE COPPIE (X(i),Y(i)), i=0,.....,n DOVE I PUNTI X(i) POSSONO ESSERE

a) EQUIDISTANTI NELL'INTERVALLO [-1,1]

b) ZERI DEL POLINOMIO DI CHEBISHEV

ED Y(i), i=0,....,n SONO DATI DA Y(i)=F(X(i)), CIOE' IL VETTORE OTTENUTO VALUTANDO NEI PUNTI X(i) LE SEGUENTI FUNZIONI:

$$f(x) = \frac{1}{1+25x^2}$$

$$f(x) = |x|$$

$$f(x) = \frac{x}{x^2+0.01}$$

CALCOLARE IL POLINOMIO INTERPOLATORE DI NEWTON E VALUTARLO (UTILIZZANDO LO SCHEMA DI HORNER), AL CRESCERE DI n, NUMERO DI OSSERVAZIONI CONOSCIUTE, IN m PUNTI (m > n), SCELTI EQUIDISTANTI NELL'INTERVALLO [-1,1].

COSA SUCCEDE AL CRESCERE DEL NUMERO DELLE OSSERVAZIONI CONOSCIUTE?

IL TIPO DI SCELTA NEI PUNTI X(i) INFLUENZA IL COMPORTAMENTO DEL POLINOMIO INTERPOLATORE?

USANDO IL COMANDO PLOT(z,pn) EFFETTUARE IL GRAFICO DEL POLINOMIO INTERPOLATORE E COMMENTARE I RISULTATI.

CALCOLARE IL VETTORE FZ(i), COME F(Z(i)), E CALCOLATE L'ERRORE ER=ABS(PN – FZ), E VISUALIZZATE IL GRAFICO CON IL COMANDO PLOT(Z,ER). COSA POTETE CONCLUDERE?

CODICE MATLAB,ANALISI E RISULTATI:

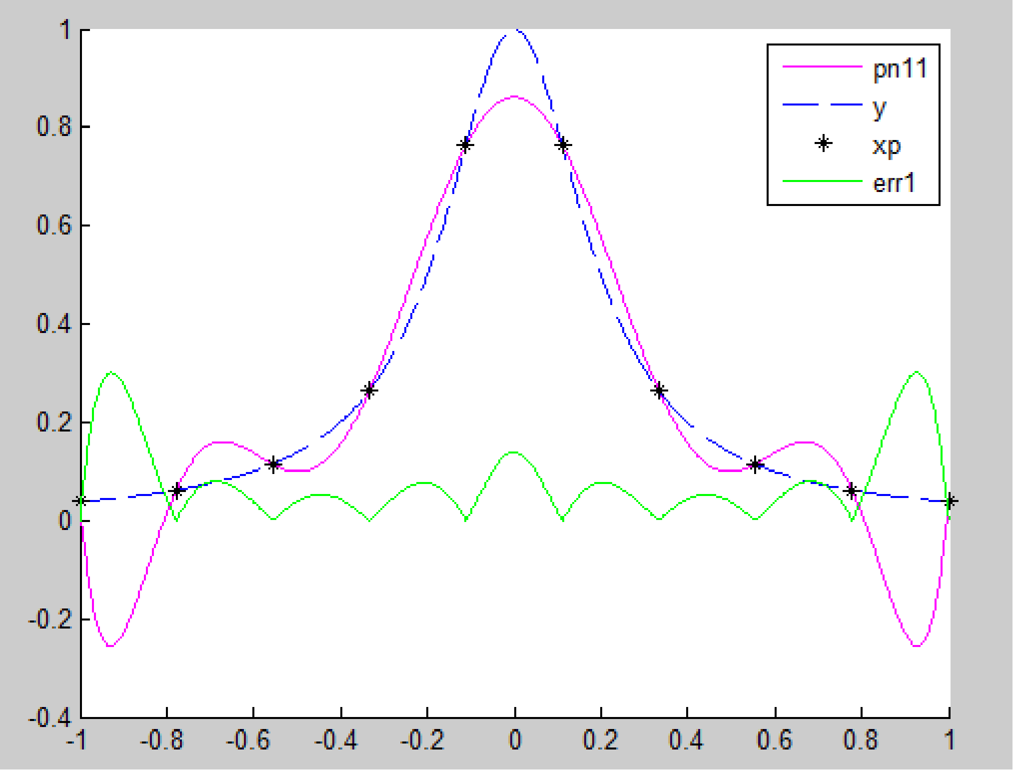
```
function y = interpolazione(n)
clearvars -except n;
close all; clc
x1 = linspace(-1,1,n);
i = [0:n];
%chebischev
x2 = cos(((1 + 2.*i)./(2*(n + 1)))*pi);
%coppie di punti
y11 = f1(x1);
y12 = f1(x2);
y21 = f2(x1);
y22 = f2(x2);
y31 = f3(x1);
y32 = f3(x2);
%funzione vera con tanti punti
z = linspace(-1,1,1000);
yz1 = f1(z);
yz2 = f2(z);
yz3 = f3(z);
%differenze divise
d11 = differenze_divise(n,x1,y11);
d12 = differenze_divise(n,x2,y12);
d21 = differenze_divise(n,x1,y21);
d22 = differenze_divise(n,x2,y22);
d31 = differenze_divise(n,x1,y31);
d32 = differenze_divise(n,x2,y32);
%polinomio interpolatore
pn11 = schema_horner(d11,z,x1);
pn12 = schema_horner(d12,z,x2);
pn21 = schema_horner(d21,z,x1);
pn22 = schema_horner(d22,z,x2);
pn31 = schema_horner(d31,z,x1);
pn32 = schema_horner(d32,z,x2);
%errore
err11 = abs(pn11 - yz1);
err12 = abs(pn12 - yz1);
err21 = abs(pn21 - yz2);
err22 = abs(pn22 - yz2);
err31 = abs(pn31 - yz3);
err32 = abs(pn32 - yz3);

figure(1)
hold on
plot(z,pn11,'m');
plot(z,pn12,'c');
plot(z,yz1,'b--');
plot(x1,y11,'k*');
plot(x2,y12,'ko');
plot(z,err11,'g');
plot(z,err12,'r');
legend('pn11','pn12','y','xp','xpC','err1','err2');

figure(2)
hold on
plot(z,pn21,'m');
plot(z,pn22,'c');
plot(z,yz2,'b--');
plot(x1,y21,'k*');
plot(x2,y22,'ko');
plot(z,err21,'g');
plot(z,err22,'r');
```

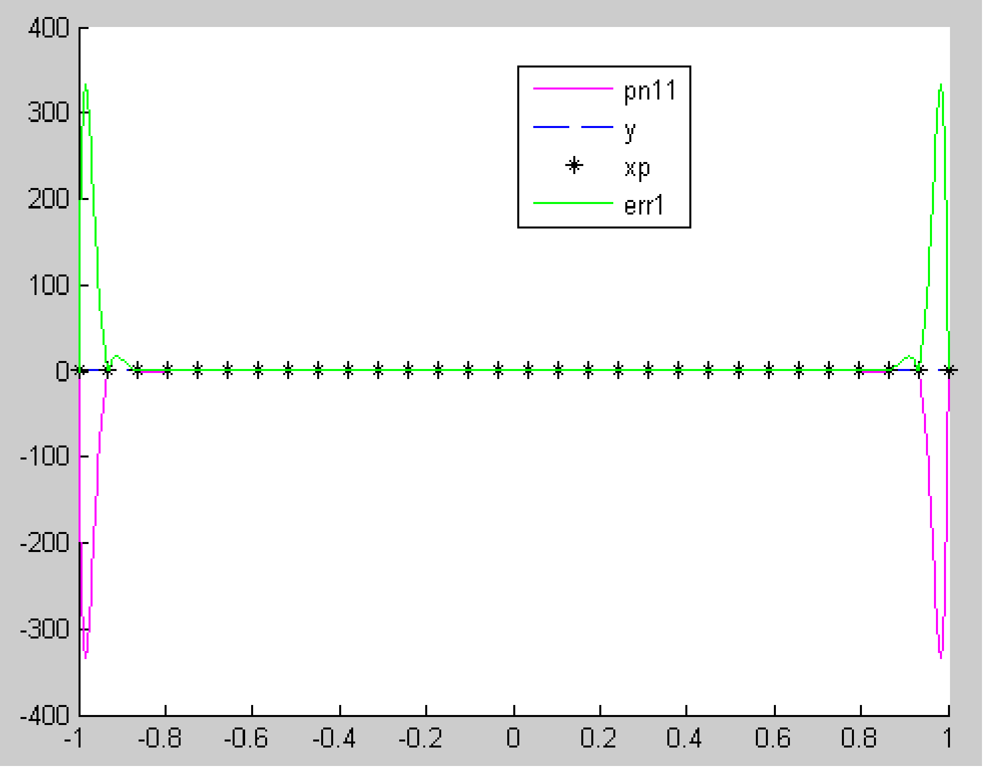
```
legend('pn21','pn22','y','xp','xpC','err1','err2');

figure(3)
hold on
plot(z,pn31,'m');
plot(z,pn32,'c');
plot(z,yz3,'b--');
plot(x1,y31,'k*');
plot(x2,y32,'ko');
plot(z,err31,'g');
plot(z,err32,'r');
legend('pn31','pn32','y','xp','xpC','err1','err2');
end

function d = differenze_divise(n,x,y)
d = zeros(n + 1);
for k=0:1:n-1
    d(k+1) = y(k+1);
end

for i=1:1:n
    for k=n-1:-1:i
        d(k + 1) = (d(k + 1) - d(k)) / (x(k + 1) - x(k + 1 - i));
    end
end
end

function pn = schema_horner(d,z,x)
m = length(z);
n = length(x);
pn = zeros(1,m);
for k=1:1:m
    pn(k) = d(n);
    for i=n-1:-1:1
        pn(k) = d(i) + (z(k) - x(i)) * pn(k);
    end
end
end



function y = f1(x)
y = 1./(1+25.*x.^2);
end

function y = f2(x)
y = abs(x);
end

function y = f3(x)
y = x ./ (x.^2 + 0.01);
end
```

RISULTATI:

Si prova a calcolare il polinomio interpolatore prima con pochi punti e poi con più punti, per vedere l'andamento del polinomio interpolatore rispetto alla funzione teorica.

(interpolazione(10) interpolazione(30))



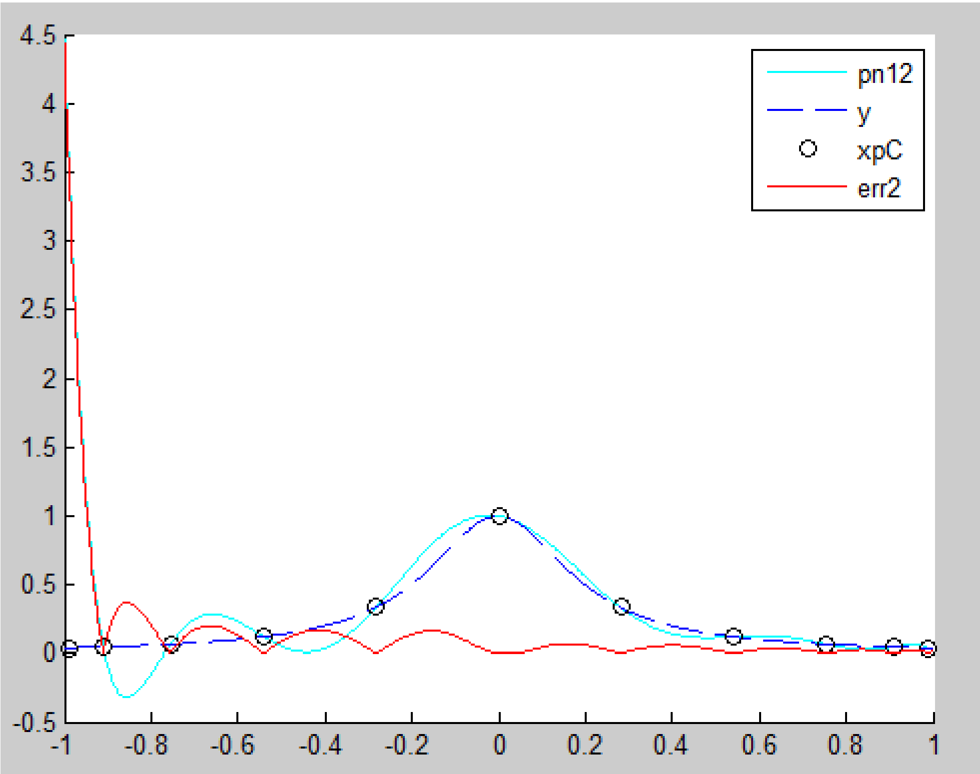


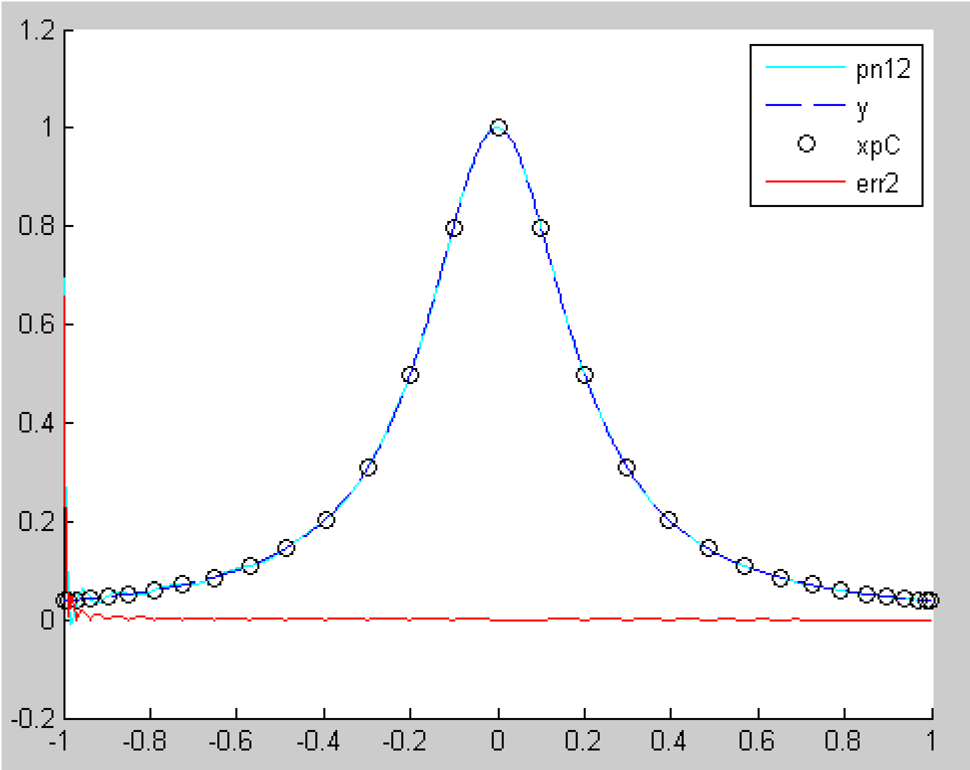
Questo è il grafico del polinomio interpolatore della prima funzione con punti scelti equidistanti nell'intervallo [-1,1].

Come si può notare, con pochi punti di interpolazione il polinomio interpolatore ha una buona approssimazione della funzione teorica, l'errore che si commette è costante.

Con molti punti di interpolzione, invece, il polinomio interpolatore approssima molto bene la funzione nei punti centrali della funzione, ma nei punti estremi l'errore si amplifica tantissimo (nel grafico non si distingue più la curva perchè l'errore alto costringe MatLab ad usare una scala più piccola per le ordinate).

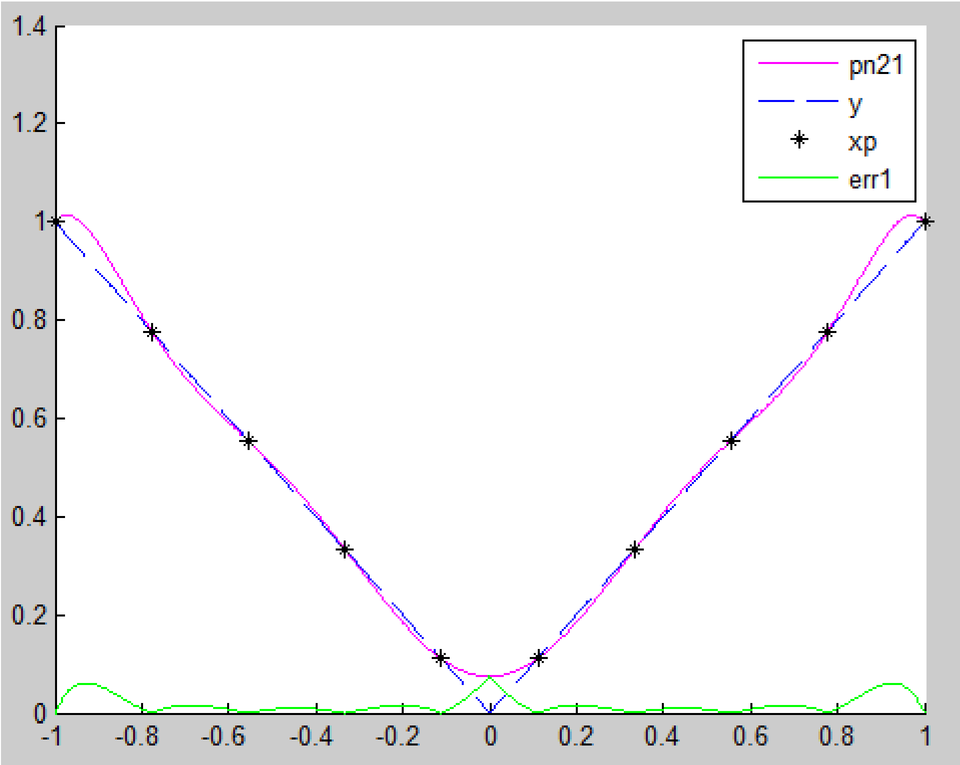
I 2 grafici seguenti mostrano come il polinomio interpolatore approssima la funzione se i punti sono scelti come zeri del polinomio di Chebishev:

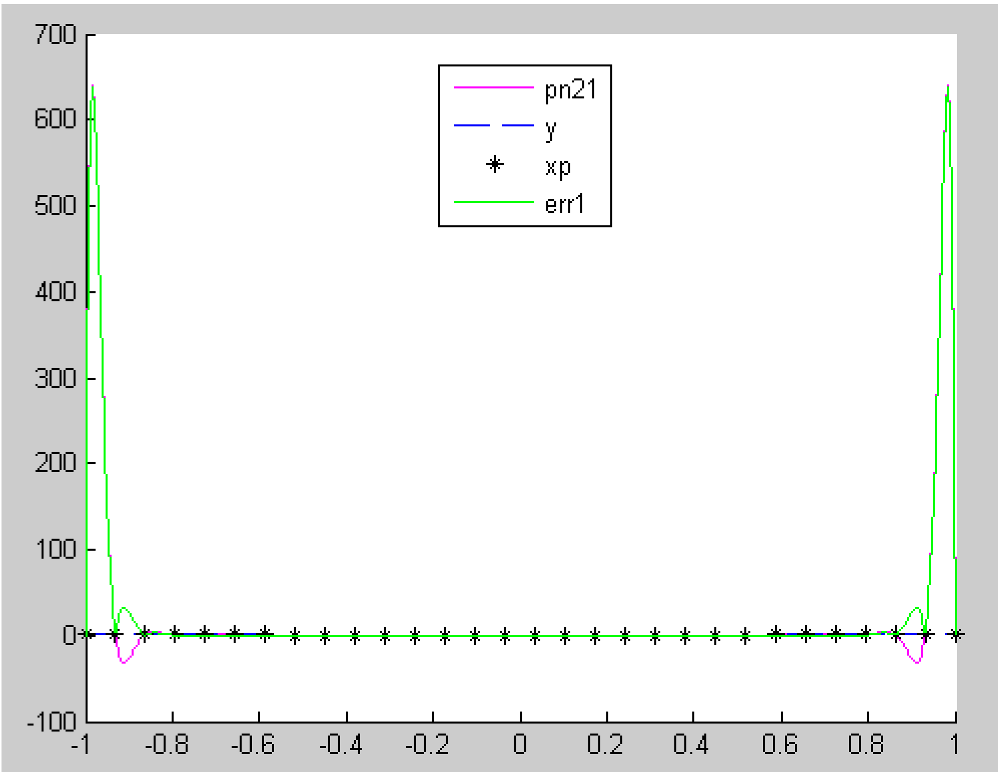


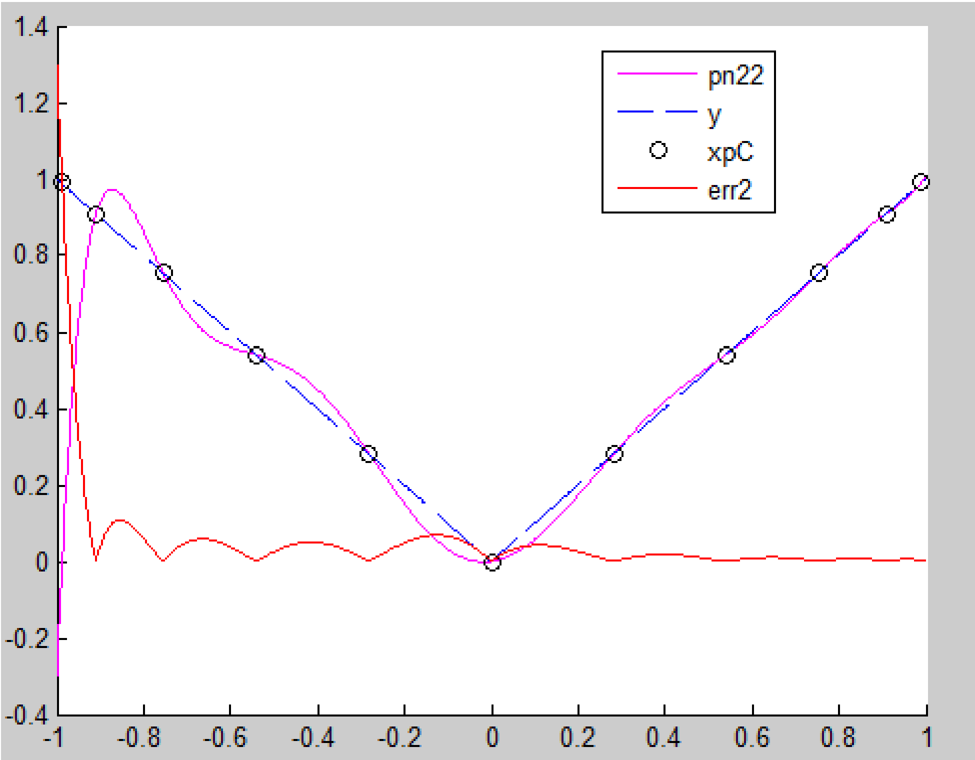


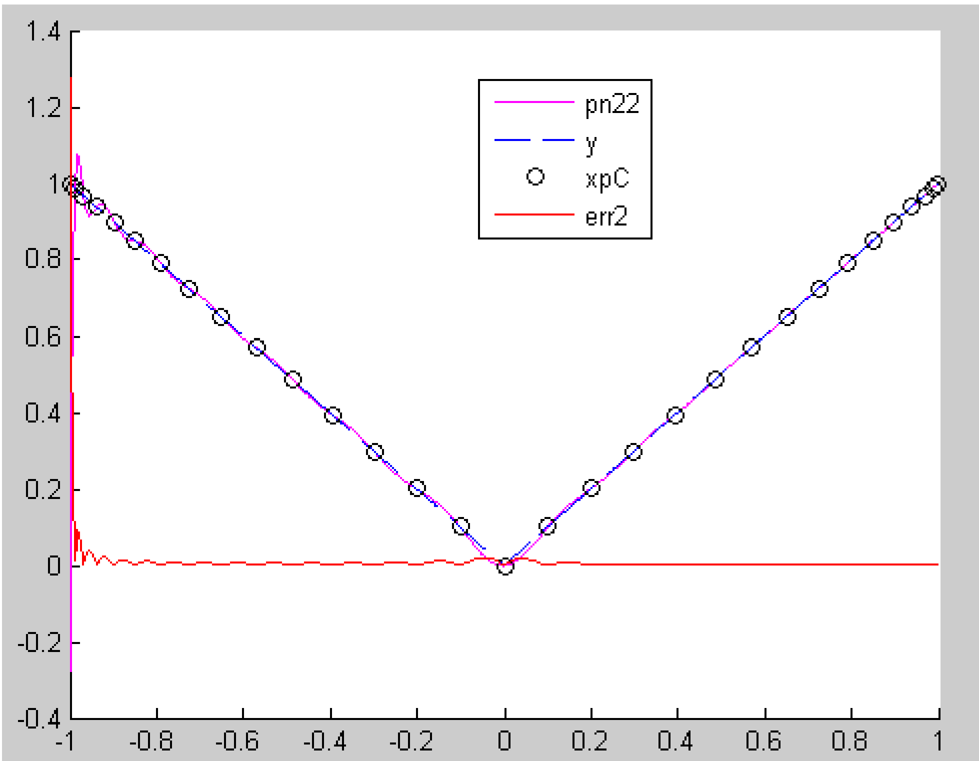
Da questi 2 grafici si vede invece come, con l'aumentare dei punti di interpolazione, il polinomio interpolatore converge alla funzione teorica.

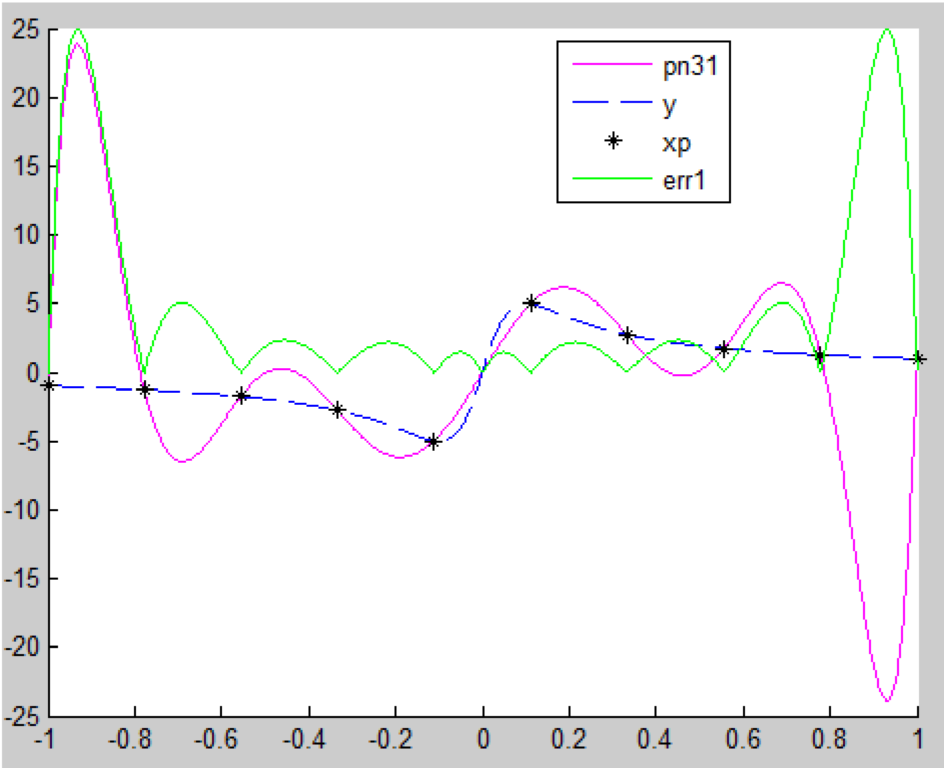
Le stesse considerazioni si possono fare anche per le altre 2 funzioni (basta vedere i grafici).

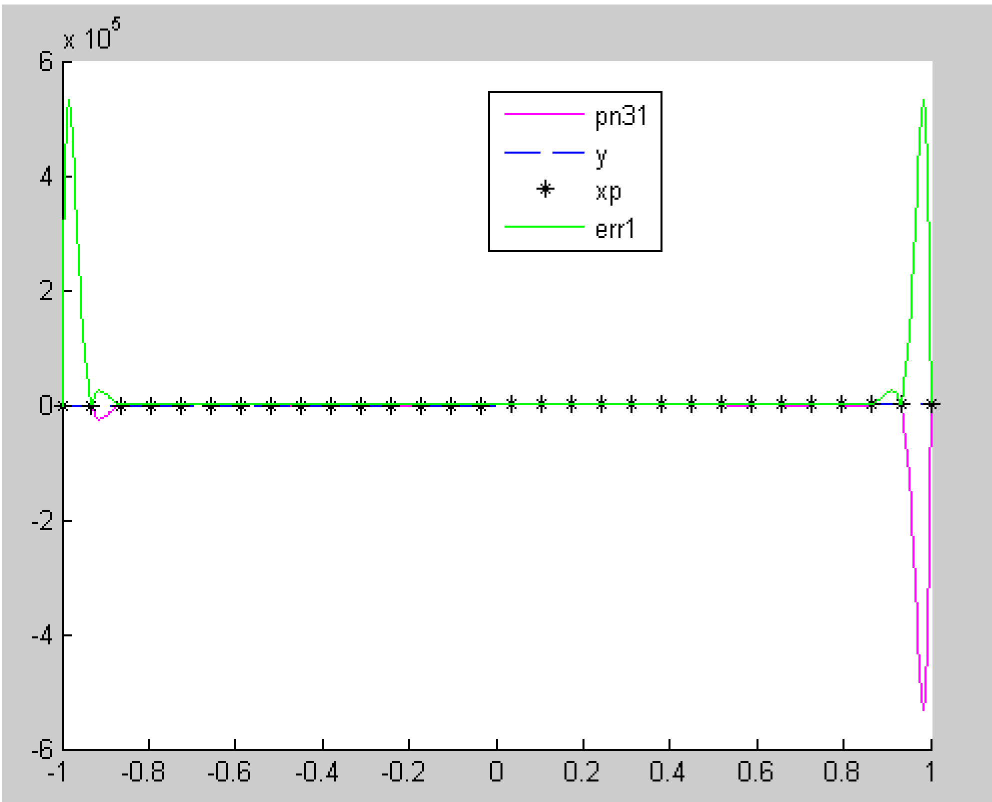

pn21
y
xp
err1

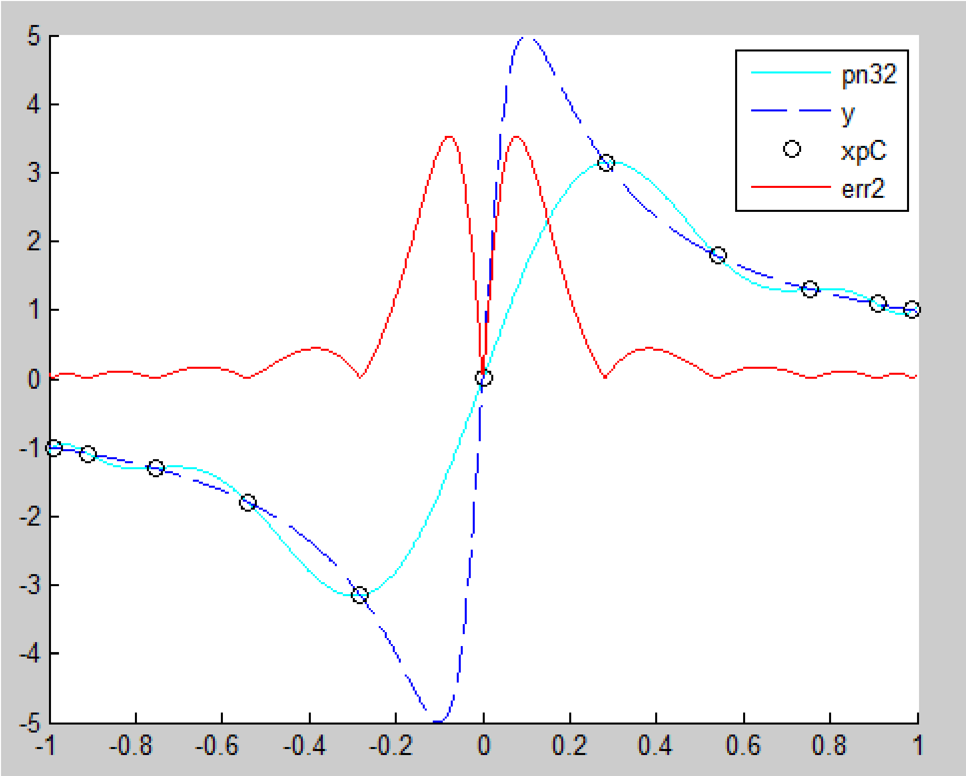

pn21
y
xp
err1


pn22
y
xpC
err2

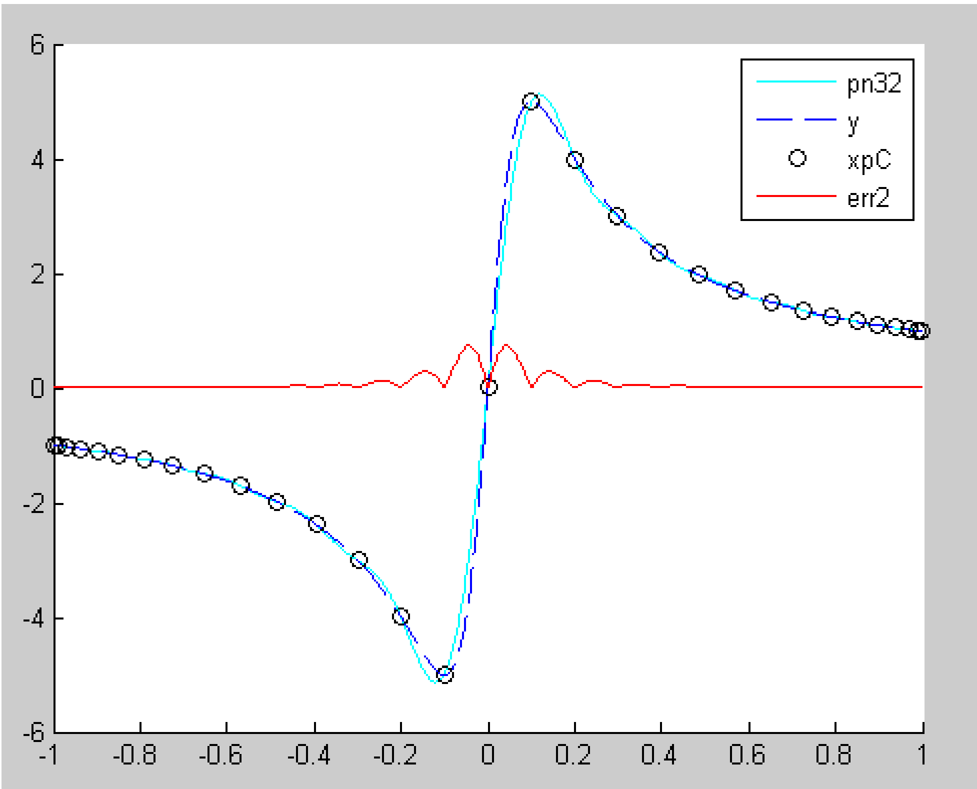

pn22
y
xpC
err2


pn31
y
xp
err1
25
20
15
10
5
0
-5
-10
-15
-20
-25
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1


x 10^5
pn31
y
xp
err1
6
4
2
0
-2
-4
-6
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1



pn32
y
xpC
err2
5
4
3
2
1
0
-1
-2
-3
-4
-5
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1



pn32
y
xpC
err2
6
4
2
0
-2
-4
-6
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

UTILIZZANDO LA FORMULA DEI TRAPEZI COMPOSITA E DI SIMPSON COMPOSITA SI RISOLVANO I SEGUENTI INTEGRALI APPLICANDO LA TECNICA DI RICERCA AUTOMATICA DI H IN MODO DA OTTENERE UNA PRECISIONE DI 10^-5.

$$\int_0^4 2^x\,dx \qquad \int_1^2 \frac{1}{1+x}\,dx \qquad \int_0^1 \frac{1+x}{1+x^3}\,dx$$

$$\int_{-1}^1 \frac{1}{1+x^2}\,dx \qquad \int_2^3 3x^3+2x+2\;dx \qquad \int_0^1 e^{x^2}\,dx$$

$$\int_0^2 \sin(10x)\,dx$$

PER OGNUNO DEGLI INTEGRALI PROPOSTI, CONFRONTARE IL NUMERO DI SUDDIVISIONI DELL'INTERVALLO DI INTEGRAZIONE RICHIESTO DAI 2 METODI PER OTTENERE UNA STIMA DELL'INTEGRALE SECONDO LA PRECISIONE RICHIESTA.

CODICE MATLAB, ANALISI E RISULTATI:

```matlab
function y = fi1(x)
y = 2.^x;
end

function y = fi2(x)
y = 1./(1+x);
end

function y = fi3(x)
y = (1 + x)./(1 + x.^3);
end

function y = fi4(x)
y = 1./(1 + x.^2);
end

function y = fi5(x)
y = 3.*x.^3 + 2.*x + 2;
end

function y = fi6(x)
y = exp(1).^x.^2;
end

function y = fi7(x)
y = sin(10.*x);
end
```

```matlab
function [trapezi,ktrapezi,  htrapezi, simpson, ksimpson, hsimpson] =
calcola_integrale(f,a,b,err)
%h = 0.01; %passo
%calcolo primo integrale
h = (b - a);
x = a:h:b;
y = feval(f,x);
area1 = h*(y(1)/2 + sum(y(2:end-1)) + y(end)/2);
%calcolo secondo integrale
h = (b - a)/2;
x = a:h:b;
y = feval(f,x);
area2 = h*(y(1)/2 + sum(y(2:end-1)) + y(end)/2);
%calcolo errore
errore = abs(area2 - area1);
k = 2;
%ciclo finchè non si trova un errore accettabile
while(errore > err)
area1 = area2;
h = (b - a)/2^k;
x = a:h:b;
y = feval(f,x);
area2 = h*(y(1)/2 + sum(y(2:end-1)) + y(end)/2);
%calcolo errore
errore = abs(area2 - area1);
k = k + 1;
end

trapezi = area1;
ktrapezi = k;
htrapezi = h;

%stessa cosa per simpson
%calcolo primo integrale
h = (b - a);
x = a:h:b;
y = feval(f,x);
area1 = h/3*(y(1) + 4*sum(y(2:2:end-1)) + 2*sum(y(3:2:end-1)) + y(end));
%calcolo secondo integrale
h = (b - a)/2;
x = a:h:b;
y = feval(f,x);
area2 = h/3*(y(1) + 4*sum(y(2:2:end-1)) + 2*sum(y(3:2:end-1)) + y(end));
%calcolo errore
errore = abs(area2 - area1);
k = 2;
%ciclo finchè non si trova un errore accettabile
while(errore > err)
area1 = area2;
h = (b - a)/2^k;
x = a:h:b;
y = feval(f,x);
area2 = h/3*(y(1) + 4*sum(y(2:2:end-1)) + 2*sum(y(3:2:end-1)) + y(end));
%calcolo errore
errore = abs(area2 - area1);
k = k + 1;
end

simpson = h/3*(y(1) + 4*sum(y(2:2:end-1)) + 2*sum(y(3:2:end-1)) + y(end));
ksimpson = k;
hsimpson = h;

end
```

RISULTATI

PRIMA FUNZIONE:

```
[area_trapezi,passi_trapezi,H_trapezi,area_simpson,passi_simpson,H_simpson] = 
calcola_integrale(@fi1,0,4,1e-5)

area_trapezi =  21.640438834066117

passi_trapezi =    12

H_trapezi =   0.001953125000000 (lunghezza del sottointervallo)

area_simpson =  21.640426036702600

passi_simpson =     7

H_simpson =   0.062500000000000 (lunghezza del sottointervallo)
```

SECONDA FUNZIONE:

```
[area_trapezi,passi_trapezi,H_trapezi,area_simpson,passi_simpson,H_simpson] = 
calcola_integrale(@fi2,1,2,1e-5)

area_trapezi =   0.405476410516339

passi_trapezi =     7

H_trapezi =   0.015625000000000 (lunghezza del sottointervallo)

area_simpson =   0.405465512025951

passi_simpson =     4

H_simpson =   0.125000000000000 (lunghezza del sottointervallo)
```

TERZA FUNZIONE:

```
[area_trapezi,passi_trapezi,H_trapezi,area_simpson,passi_simpson,H_simpson] =
calcola_integrale(@fi3,0,1,1e-5)

area_trapezi =   1.209189403568015


passi_trapezi =     9


H_trapezi =   0.003906250000000 (lunghezza del sottointervallo)


area_simpson =   1.209199639733792


passi_simpson =     6


H_simpson =   0.031250000000000 (lunghezza del sottointervallo)
```

QUARTA FUNZIONE:

```
[area_trapezi,passi_trapezi,H_trapezi,area_simpson,passi_simpson,H_simpson] =
calcola_integrale(@fi4,-1,1,1e-5)

area_trapezi =   1.570791240531876


passi_trapezi =    10


H_trapezi =   0.003906250000000 (lunghezza del sottointervallo)


area_simpson =   1.570796325612411


passi_simpson =     6


H_simpson =   0.062500000000000 (lunghezza del sottointervallo)
```

QUINTA FUNZIONE:

```
[area_trapezi,passi_trapezi,H_trapezi,area_simpson,passi_simpson,H_simpson] =
calcola_integrale(@fi5,2,3,1e-5)

area_trapezi =  55.750003576278687


passi_trapezi =    12


H_trapezi =    4.882812500000000e-004 (lunghezza del sottointervallo)


area_simpson =  55.750000000000000


passi_simpson =     3


H_simpson =   0.250000000000000 (lunghezza del sottointervallo)
```

SESTA FUNZIONE:

```
[area_trapezi,passi_trapezi,H_trapezi,area_simpson,passi_simpson,H_simpson] =
calcola_integrale(@fi6,0,1,1e-5)

area_trapezi =   3.194532111516764


passi_trapezi =    11


H_trapezi =    9.765625000000000e-004 (lunghezza del sottointervallo)


area_simpson =   3.194528320142944


passi_simpson =     6


H_simpson =   0.031250000000000 (lunghezza del sottointervallo)
```

SETTIMA FUNZIONE:

[area_trapezi,passi_trapezi,H_trapezi,area_simpson,passi_simpson,H_simpson] = calcola_integrale(@fi7,0,2,1e-5)

area_trapezi = 0.059184267001441

passi_trapezi = 11

H_trapezi = 0.001953125000000 (lunghezza del sottointervallo)

area_simpson = 0.059191990395684

passi_simpson = 8

H_simpson = 0.015625000000000 (lunghezza del sottointervallo)

CONSIDERAZIONI:

Dagli integrali calcolati si possono trarre le seguenti conclusioni:

- L'area degli integrali si ottiene con entrambi i metodi (trapezi e simpson)
- La formula di Simpson composita è più efficiente della formula dei trapezi composita, in quanto il numero dei passi per ottenere la soluzione è sempre minore nel metodo di Simpson rispetto al metodo dei trapezi. Di conseguenza, dato che il numero dei passi è minore, è maggiore la lunghezza del sottointervallo nel metodo di Simpson rispetto al metodo dei trapezi (più un sottointervallo è lungo, meno sottointervalli ci sono).

Questi risultati rispecchiano la teoria, infatti il metodo di Simpson trova l'area più velocemente perchè è di ordine 2 (si considerano 3 punti, gli estremi dell'intervallo più il punto medio), mentre il metodo dei trapezi è di ordine 1 (si considerano solo 2 punti, gli estremi dell'intervallo).