

Relazione Matematica Applicata e Computazionale

Studente:Domenico Scognamiglio

Matricola:0124000017



Indice

1 Numeri Complessi	6
1.1 Simulare i movimenti di una forma del tetris durante una partita:cioé generare a caso una forma(quadrato,asta,'Z','L',...) e mentre questa scende l'utente gioca governando i tre movimenti della forma mediante bottoni sulla figura (uicontrol() in MATLAB). [liv.3]	6
1.2 Come ristabilire l'ordine naturale nelle radici fornite da roots()? [liv.1]	7
1.3 Perche' risulta sempre ($\forall n, \forall z$) $\sum(r) == 0$ e $\prod(r) == -1^{n-1} \cdot z$? [liv.1]	8
1.4 Visualizzare le radici in 3D mediante il Calcolo Simbolico (ezsurf(),ezmesh(),...). [liv.1]	9
1.5 Scrivere una function che,fissato un valore di n,restituiscia tutte le radici n-sime dell'unità primitiva.Realizzare due algoritmi:il primo fa uso delle coordinate polari e della formula matematica ed il secondo fa uso della funzione roots(...). [liv.1]	10
1.6 Calcolare e visualizzare tutte le radici n-sime di un numero complesso verificando (anche graficamente) che le medesime si possono ottenere tramite le potenze successive di una qualsiasi radice primitiva n-sima dell'unita' scelta a caso.[liv.1]	10
2 Funzioni nel campo complesso	12
2.1 Visualizzare mediante il MATLAB Symbolic Math Toolbox alcune curve del piano (ellisse,arco di iperbole,...) descritte mediante equazioni parametriche.Nel caso di curve regolari,visualizzare anche la tangente in un punto scelto dall'utente (o scelto a caso).[liv.1]	12
2.2 Ripetere in ambiente numerico l'esercizio precedente,sapendo che ora non si ha l'espressione della funzione ma solo dei campioni (punti della curva).[liv.1]	13
2.3 Ricostruire una curva 2D (regolare) mediante interpolazione:ad esempio da alcuni campioni ricostruire la propria mano oppure il profilo nei dati contenuti nella cartella dati_interpolazione.[liv.1]	14
2.4 Produrre i grafici di alcune funzioni complesse per visualizzarne alcune caratteristiche (continuitá,periodicitá sul dominio visualizzato) scegliendo opportunamente il dominio di visualizzazione e il tipo di griglia (circolare o rettangolare).[liv.1]	14
2.5 Visualizzare la continuitá o la discontinuitá di alcune funzioni complesse in un intorno (circolare e rettangolare) $I(z_0)$ di alcuni punti z_0 del piano complesso.[liv.1]	17
2.6 Verificare e visualizzare la continuitá o la discontinuitá di alcune funzioni complesse in un intorno di alcuni punti z_0 del piano complesso mediante Symbolic Math Toolbox.[liv.1]	22
2.7 Visualizzare l'olomorfia o la non olomorfia di alcune funzioni complesse in un intorno (circolare e rettangolare), $I(z_0)$ di alcuni punti z_0 del piano complesso.[liv.1]	30
2.8 Verificare e visualizzare la derivabilitá in senso complesso di alcune funzioni mediante Symbolic Math Toolbox.[liv.1]	36
2.9 Come si calcola mediante il Symbolic Math Toolbox di MATLAB il seguente limite(in senso complesso)? [liv.1]	
$f'(z_0) = \lim_{z \rightarrow z_0} \frac{f(z) - f(z_0)}{z - z_0} = \lim_{z \rightarrow z_0} \frac{f(z + \Delta z) - f(z_0)}{\Delta z}$	
2.10 Verificare e visualizzare la validitá delle Eq. di Cauchy-Riemann in senso complesso di alcune funzioni in un intorno di alcuni punti z_0 mediante Symbolic Math Toolbox.[liv.1]	41
3 Successioni e serie	45
3.1 Mediante function MATLAB approssimare numericamente a p cifre decimali corrette la somma delle serie:	
$1 + \frac{1}{2!} + \frac{1}{3!} + \dots$	
$1 + \frac{1}{2^2} + \frac{1}{3^2} + \dots$	
$1 + \frac{1}{2^4} + \frac{1}{3^4} + \dots$	
[liv.2]	45
3.2 Mediante Symbolic Math Toolbox di MATLAB studiare la serie	
$1 - x^2 + x^4 - x^6 + \dots$	

	[liv.2]	Confrontare,evidenziandone le differenze,la bontà delle due stime calcolabili dell'errore nella serie esponenziale:	46
3.3		$ e^x - S(n, x) \approx \frac{ x ^n}{n!} \frac{n+1}{n+1-x}$ $e^{-\frac{1}{2}} \frac{ e^x - S(n, x) }{e^x} \approx \frac{ x ^n}{n!}$	
	[liv.2]	Visualizzare il campo di convergenza delle seguenti serie di potenze nel campo complesso:geometrica,armonica,esponenziale,ed ancora le serie:	48
3.4		$\sum_{n=0}^{\infty} \frac{z^n}{2^{2n}}, \quad \sum_{n=0}^{\infty} \frac{(z+1)^n}{n2^n}, \quad \sum_{n=0}^{\infty} \frac{(z-2)^n}{n^2 2^{2n}}$ $\sum_{n=0}^{\infty} (1+ni) z^n, \quad \sum_{n=0}^{\infty} (\log n)^2 z^n, \quad \sum_{n=0}^{\infty} \frac{5^n z^{3n}}{2n(2n+2)}$	
		Giustificare i risultati ottenuti.	
3.5		Studiare le precedenti serie mediante il Symbolic Math Toolbox individuandone il campo di convergenza e l'eventuale somma.Giustificare i risultati ottenuti.	49
3.6		Visualizzare il comportamento sulla frontiera delle seguenti serie di potenze nel campo complesso:geometrica,armonica,esponenziale,ed ancora le serie:	54
		$\sum_{n=0}^{\infty} \frac{z^n}{2^{2n}}, \quad \sum_{n=0}^{\infty} \frac{(z+1)^n}{n2^n}, \quad \sum_{n=0}^{\infty} \frac{(z-2)^n}{n^2 2^{2n}}$ $\sum_{n=0}^{\infty} (1+ni) z^n, \quad \sum_{n=0}^{\infty} (\log n)^2 z^n, \quad \sum_{n=0}^{\infty} \frac{5^n z^{3n}}{2n(2n+2)}$	
		Per tutte le serie visualizzare anche,sia nel cerchio di convergenza sia sulla frontiera,la serie ottenuta mediante derivazione termine a termine.	58
	Interpolazione trigonometrica e DFT		67
4.1		Per le funzioni di seguito specificate costruire (al variare del grado N=1,2,3,4,5,6,7,8) i polinomi trigonometrici interpolanti generando in modo random nodi di interpolazione.Rappresentare negli intervalli indicati la funzione,i nodi di interpolazione e il polinomio trigonometrico interpolante:	
		$f(x) = \sin(2x), x \in [0, 2\pi] \quad f(x) = \sin(8x)^2, x \in [0, \pi]$ $f(x) = x , x \in [-1, 1] \quad f(x) = x^2, x \in [-1, 1]$ $f(x) = \frac{1}{1+x^2}, x \in [-5, 5] \quad f(x) = \sin(2x) - \sin(8x), x \in [-\pi, \pi]$ $f(x) = \cos(22x), x \in [-1, 1] \quad f(x) = \cos(8\pi x)^4, x \in [-2, 2]$ $f(x) = \sin(\pi x) , x \in [-1, 1] \quad f(x) = \{x(\pi + x), x \in [-\pi, 0] \cup x(\pi - x), x \in [0, \pi]\}$	
	[liv.2]		
4.2		Per le funzioni e gli intervalli di seguito specificati,costruire efficientemente,al variare del numero M di nodi,i polinomi trigonometrici interpolanti Q e T generando i nodi di interpolazione come segue:x.i=linspace(...);x.i=rand(...). Confrontare graficamente tra loro i polinomi interpolanti quando vengano considerati approssimazioni delle funzioni dei dati:	67
		$f(x) = \sin(2x), x \in [0, 2\pi] \quad f(x) = \sin(8x)^2, x \in [0, \pi]$ $f(x) = x , x \in [-1, 1] \quad f(x) = x^2, x \in [-1, 1]$ $f(x) = \frac{1}{1+x^2}, x \in [-5, 5] \quad f(x) = \sin(2x) - \sin(8x), x \in [-\pi, \pi]$ $f(x) = \cos(22x), x \in [-1, 1] \quad f(x) = \cos(8\pi x)^4, x \in [-2, 2]$	

$f(x) = \sin(\pi x) , x \in [-1, 1]$	$f(x) = \{x(\pi + x), x \in [-\pi, 0] \cup (\pi - x), x \in [0, \pi]\}$	
[liv.1]		74
4.3 Ripetere l'esercizio precedente solo per 'molti' nodi equispaziati costruendo i coefficienti del polinomio interpolante mediante i 3 metodi. Confrontare i relativi tempi di esecuzione. [liv.3] . . .		80
4.4 Mediante circshift() simulare una stringa animata che scorre da destra a sinistra. [liv.1] . . .		81
4.5 Visualizzare in MATLAB le proprieta' della DFT mediante Symbolic Math Toolbox. [liv.3] . . .		82
4.6 Quale relazione lega i coefficienti del polinomio trigonometrico interpolante ed una DFT? [liv.1] . . .		89
4.7 Ricostruire nell'intervallo [0,3pi] mediante interpolazione opportuna su 10,20 dati, la curva 3d di equazioni parametriche:		
$x = \cos(t), y = \sin(t), z = t$		
[liv.2]		89
5 Serie e coefficienti di Fourier		91
5.1 Mediante il Symbolic Math Toolbox di Matlab costruire i coefficienti di Fourier in forma complessa		
$\gamma_k = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t) e^{-ikt} dt$		
per le funzioni $t/2, t^2, \sin(t), \cos(t)$ nell'intervallo $[-\pi, \pi]$ e per ognuna visualizzare il grafico della funzione e:		
-di tutte le ridotte fino all'ordine 41;		
-delle ultime 6 ridotte.		
$S_{N+1}(x) = \sum_{k=-\frac{N}{2}}^{+\frac{N}{2}} \gamma_k e^{ikx}$		
[liv.2]		91
5.2 Unire i due riquadri MATLAB per costruire una function		
$function[y] = period_repeat(fun, a, b, x)$		
che restituisca il valore in x della ripetizione periodica della restrizione all'intervallo $]a, b[$ di fun. [liv.2]		93
5.3 Mediante il Symbolic Math Toolbox di Matlab costruire i coefficienti di Fourier in forma complessa per le funzioni $t/2, t^2, \sin(t), \cos(t)$ in un intervallo generico $[a, b]$, stabilito in input, e per ognuna visualizzare, nell'intervallo $[a - 2(b-a), b + 2(b-a)]$ il grafico della funzione e delle ridotte della sua Serie di Fourier fino all'ordine max 40. [liv.2]		94
5.4 Approssimare numericamente nell'intervallo $[-T/2, T/2]$ alcune funzioni $f(x)$ mediante ridotta di ordine N della sua serie di Fourier costruendo i coefficienti di Fourier sia mediante integrazione numerica (function quad()) sia mediante DFT (functions fft() ed fftshift()) e confrontare i risultati ottenuti al variare di T e N. [liv.1]		96
5.5 Ripetere l'esercizio precedente supponendo di avere a disposizione solo n campioni della funzione da approssimare. [liv.2]		99
5.6 Approssimare numericamente nell'intervallo $[-T/2, T/2]$ alcune funzioni $f(x)$ mediante ridotta di ordine N della sua serie di Fourier. Per l'errore di discretizzazione in funzione di N usare le seguenti misure:		
$E_1(N) = \max f(x) - FS_n(x) , x \in \left[-\frac{T}{2}, +\frac{T}{2}\right]$		
$E_2(N) = \int_{-T/2}^{+T/2} f(x) - FS_n(x) ^2 dx$		
Visualizzare al variare di N, $E_1(N)$ e $E_2(N)$, costruendo i coefficienti di Fourier sia mediante integrazione numerica (function quad()) sia mediante DFT (function fft() ed fftshift()) e confrontare i risultati ottenuti al variare di T e N. [liv.3]		102

6 Trasformata di Fourier e applicazioni	108
6.1 Derivare l'algoritmo per l'anttrasformata.[liv.3]	108
6.2 -Al variare di n=16,32,64 come ottenere gli N+1 campioni $F(\nu)$ della trasformata di Fourier tutti nello stesso intervallo $[-\Omega/2, \Omega/2]$?Quali considerazioni possono farsi...[liv.2]	
-Approssimare numericamente in $[-T/2, T/2]$,la trasformata di Fourier di alcune funzioni $f(x)$.Visualizzare i risultati sia rispetto alla frequenza circolare che rispetto a quella angolare,confrontandoli quando possibile con la trasformata analitica e commentandoli al variare dei parametri di discretizzazione T e N.[liv.1]	
-Ripetere l'esercizio con l'anttrasformata.[liv.2]	109
6.3 Determinare,mediante la FT e senza usare spectrogram(),il numero di telefono dal suono registrato nei file: -NumTel.wav -noise1NumTel.wav -noise2NumTel.wav estraendo le frequenze dei singoli toni (STFT).[liv.3]	114
6.4 A partire dalle frequenze delle note (nel file frequenze_note_musicali.pdf),creare in Matlab un file audio (function auwrite(),wavwrite()),con al piu' 2 secondi per nota,contenente: -le 7 note musicali -un accordo (tipo do_maggiore=DO-MI-SOL).[liv.3]	123

1 Numeri Complessi

- 1.1 Simulare i movimenti di una forma del tetris durante una partita:cioé generare a caso una forma(quadrato,asta,'Z','L',...) e mentre questa scende l'utente gioca governando i tre movimenti della forma mediante bottoni sulla figura (uicontrol() in MATLAB). [liv.3]

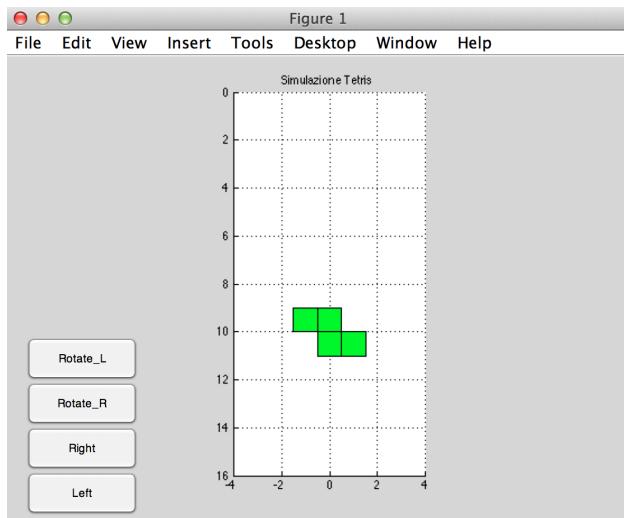
```
1 x1=-4;x2=4;y1=0;y2=16;%limiti della finestra grafica
2 V=[x1 x2 y1 y2];
3 axis equal;axis(V);axis ij;grid on;
4 qb=[0 1 1+i i 0].'; %quadrato di base
5 %FORME DEL TETRIS
6 Q1=[qb qb+1 qb+1+i qb+i];%Quadrato grande
7 Q2=[qb qb+1 qb+i qb+2i];%L
8 Q3=[qb qb+i qb+2i qb+2i+1];%L capovolta
9 Q4=[qb qb+1 qb+1+i qb+2+i];%Z
10 Q5=[qb qb+1 qb+2 qb+3];%Asta
11 Q6=[qb qb+1 qb+2 qb+1+i];%T
12 N=6;
13 %Centro le figure rispetto al loro baricentro
14 Q1=Q1-real(mean(sum(Q1(1:end-1,:))/4));
15 Q2=Q2-real(mean(sum(Q2(1:end-1,:))/4));
16 Q3=Q3-real(mean(sum(Q3(1:end-1,:))/4));
17 Q4=Q4-real(mean(sum(Q4(1:end-1,:))/4));
18 Q5=Q5-real(mean(sum(Q5(1:end-1,:))/4));
19 Q6=Q6-real(mean(sum(Q6(1:end-1,:))/4));
20 %scelta forma casuale
21 forma=cat(3,Q1,Q2,Q3,Q4,Q5,Q6);
22 ind=5;
23 Q=forma(:,:,ind);
24 patch(real(Q),imag(Q),'g');
25 title('Simulazione Tetris');
26 while all(all(imag(Q)<y2))
    %Fino a che non arrivo all'estremita' inferiore del tavolo da gioco
27     hL=uicontrol('Style','pushbutton','Position',[20 10 100 40],...
28         'String','Left','Callback','eval('''Q=Q-1;'')');
29     hL2=uicontrol('Style','pushbutton','Position',[20 50 100 40],...
30         'String','Right','Callback','eval('''Q=Q+1;'')');
31     hL3=uicontrol('Style','pushbutton','Position',[20 90 100 40],...
32         'String','Rotate_R','Callback','rotate_right');
33     hL4=uicontrol('Style','pushbutton','Position',[20 130 100 40],...
34         'String','Rotate_L','Callback','rotate_left');
35     pause(1);
36     Q=Q+i;%Faccio scendere di un'unita' per volta il blocco
37     clf;
38     patch(real(Q),imag(Q),'g');%e lo visualizzo
39     axis equal;axis(V);axis ij;grid on;title('Simulazione Tetris')
40 end
41 %-----
42 %rotate_right
43 %Questa procedura ruota il blocco di 90 gradi verso destra
44 baricentro=sum(Q(1:end-1,:))/4;
45 %baricentri dei singoli blocchetti che compongono il blocco
46 baricentro=mean(baricentro);
47 %baricentro del blocco nella posizione attuale
48 Q=Q-baricentro;
49 %Traslazione del blocco in modo da centrare l'origine del sistema di
```

```

51 %riferimento con il baricentro del blocco
52 Q=Q*(-i);
53 %Rotazione
54 Q=Q+baricentro;
55 %Riposizionamento nella vecchia posizione
56 %-----
57 %rotate_left
58 %Questa procedura ruota il blocco di 90 gradi verso sinistra
59 baricentro=sum(Q(1:end-1,:))/4;
60 baricentro=mean(barcenetro);
61 Q=Q-baricentro;
62 Q=Q*i;
63 Q=Q+baricentro;

```

ESECUZIONE DEL PROGRAMMA



1.2 Come ristabilire l'ordine naturale nelle radici fornite da roots()?[liv.1]

```

1 z=2+3i;
2 n=8;%voglio calcolare le 8 radici ottave di z
3 k=(0:n-1)';
4 w=roots([1 zeros(1,n-1) -z]);
5 ro=abs(z)^(1/n);
6 teta=angle(z)/n+2*k*pi/n;
7 w1=ro*exp(i*teta);
8 disp(' Radici con roots Radici con formula matematica');
9 disp([w w1]);
10 disp(' Arg roots Arg formula');
11 disp([angle(w) angle(w1)]);
12 [ang,ind]=sort(angle(w));
13 i=1;
14 while ang(i)<0
15     ind=circshift(ind,-1);
16     i=i+1;

```

```

17 end
18 disp('Radici con roots ordinate');
19 disp(w(ind));

```

ESECUZIONE DEL PROGRAMMA

```

Radici con roots      Radici con formula matematica
-0.9255 + 0.7221i   1.1650 + 0.1438i
-1.1650 - 0.1438i   0.7221 + 0.9255i
-0.1438 + 1.1650i   -0.1438 + 1.1650i
-0.7221 - 0.9255i   -0.9255 + 0.7221i
0.1438 - 1.1650i   -1.1650 - 0.1438i
0.7221 + 0.9255i   -0.7221 - 0.9255i
0.9255 - 0.7221i   0.1438 - 1.1650i
1.1650 + 0.1438i   0.9255 - 0.7221i

Arg roots  Arg formula
2.4790    0.1228
-3.0187   0.9082
1.6936    1.6936
-2.2333   2.4790
-1.4479   -3.0187
0.9082    -2.2333
-0.6625   -1.4479
0.1228    -0.6625

Radici con roots ordinate
1.1650 + 0.1438i
0.7221 + 0.9255i
-0.1438 + 1.1650i
-0.9255 + 0.7221i
-1.1650 - 0.1438i
-0.7221 - 0.9255i
0.1438 - 1.1650i
0.9255 - 0.7221i

```

1.3 Perche' risulta sempre ($\forall n, \forall z$) $\text{sum}(r) == 0$ e $\text{prod}(r) == -1^{n-1} \cdot z$? [liv.1]

Sia z la radice ennesima di un numero complesso $z = \sqrt[n]{\zeta}$.

Le n radici del polinomio $z^n - \zeta = 0$ costituiscono le n radici del numero complesso. Le formule di Viéte affermano che:

Se $P(x) = a_n X^n + a_{n-1} X^{n-1} + \dots + a_1 X + a_0$ é un polinomio di grado $n \geq 1$ con coefficienti complessi (cioé i numeri $a_0, a_1, \dots, a_{n-1}, a_n$ sono complessi con $a_n \neq 0$) per il teorema fondamentale dell'algebra $P(x)$ ha n radici complesse (non necessariamente distinte) x_1, x_2, \dots, x_n .

$$x_1 + x_2 + \dots + x_n = -\frac{a_{n-1}}{a_n}$$

$$x_1 x_2 \dots x_n = (-1)^n \frac{a_0}{a_n}$$

Somma delle Radici
Prodotto delle radici

Quindi siccome a_{n-1} é sempre uguale a 0 nel polinomio $z^n - \zeta = 0$ la somma delle radici di un numero complesso é sempre uguale a 0. Il prodotto delle radici invece é pari a $(-1)^n (\frac{-\zeta}{1}) = (-1)^{n-1} \zeta$.

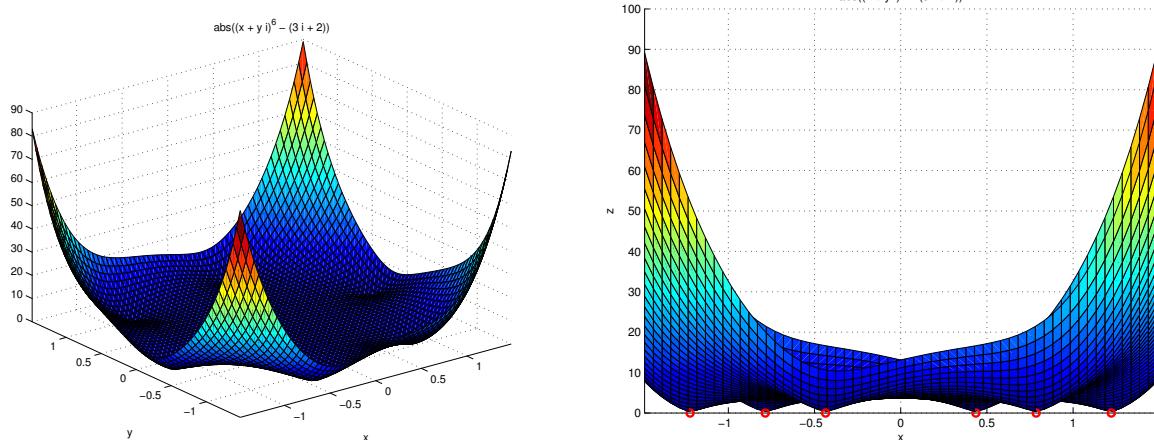
1.4 Visualizzare le radici in 3D mediante il Calcolo Simbolico (ezsurf(),ezmesh(),...). [liv.1]

```

1 %Visualizzare le radici in 3D mediante il Calcolo Simbolico
2 %(ezsurf(),ezmesh(),...)
3 n=6;
4 z0=2+3i;
5 %Voglio calcolare le 6 radici seste di z
6 r=roots([1,zeros(1,n-1),-z0]);
7 syms x y;
8 z=x+i*y;
9 fz=z^n-z0;
10 %Costruisco una funzione complessa che vale 0 nelle radici n-sime di z0
11 ezsurf(abs(fz), [-abs(r(1))*1.2, abs(r(1))*1.2]);
12 figure
13 hold on
14 ezsurf(abs(fz), [-abs(r(1))*1.2, abs(r(1))*1.2]);
15 view([0 0]);
16 plot(real(r),imag(r),'or','LineWidth',2);
17 zlabel('z');
18 %visualizzo dei cerchi rossi in corrispondenza delle radici

```

ESECUZIONE DEL PROGRAMMA



1.5 Scrivere una function che,fissato un valore di n,restituisca tutte le radici n-sime dell'unità primitiva.Realizzare due algoritmi:il primo fa uso delle coordinate polari e della formula matematica ed il secondo fa uso della funzione roots(...). [liv.1]

```

1 %VERSIONE CON ROOTS(...)
2 function [ w ] = primitive_roots( n )
3 k=(0:n-1)';
4 w1=roots([1,zeros(1,n-1),-1]);
5 [ang,ind]=sort(angle(w1));
6 %ordino le radici con lo stesso ordine della formula matematica
7 i=1;
8 while ang(i)<0
9     ind=circshift(ind,-1);
10    i=i+1;
11 end
12 w=w1(ind);
13 %w contiene le radici ordinate
14 mcd=gcd(n,k);
15 w=w(find(mcd==1));
16 %restituisco le radici primitive,ossia quelle
17 %tali che il massimo comun divisore tra n e k e' pari a 1
18 end
19 %-----
20 %VERSIONE CON FORMULA MATEMATICA
21 function [ w ] = primitive_roots2( n )
22 k=(0:n-1)';
23 teta=2*k*pi/n;
24 w=exp(i*teta);
25 mcd=gcd(n,k);
26 w=w(find(mcd==1));
27 end

```

1.6 Calcolare e visualizzare tutte le radici n-sime di un numero complesso verificando (anche graficamente) che le medesime si possono ottenere tramite le potenze successive di una qualsiasi radice primitiva n-sima dell'unità scelta a caso.[liv.1]

```

1 if ~exist('z', 'var') %UTILE PER LA PUBLISH
2 z=input('Inserire il valore del numero complesso:');
3 end
4 if ~exist('n', 'var')
5 n=input('Inserire il valore di n:');
6 end
7 k=(0:n-1)';
8 s=sprintf('Radici %d-esime di %d+%di',n,real(z),imag(z));
9 w=roots([1,zeros(1,n-1),-z]);%CALCOLO LE N RADICI DEL NUMERO COMPLESSO
10 p=primitive_roots(n);%CALCOLO LE RADICI PRIMITIVE N-SIME DELL'UNITA'
11 hold on;axis equal;grid on;
12 title(s);
13 compass(z,'b');
14 h=compass(w,'y');
15 set(h,'LineWidth',2);

```

```

16 p2=p(1).^k;%POTENZE SUCCESSIVE DI UNA DELLE RADICI PRIMITIVE
17 compass(p2,'g');
18 %MOLTIPLICO UNA DELLE RADICI PER TUTTE LE RADICI N-SIME DELL'UNITÀ
19 p2=p2*abs(w(1));%OMOTETIA
20 compass(p2,'r');
21 p2=p2*exp(i*angle(w(1)));%ROTAZIONE
22 compass(p2,'k');
23 disp([w p2]);
24 xlabel('x');ylabel('y');

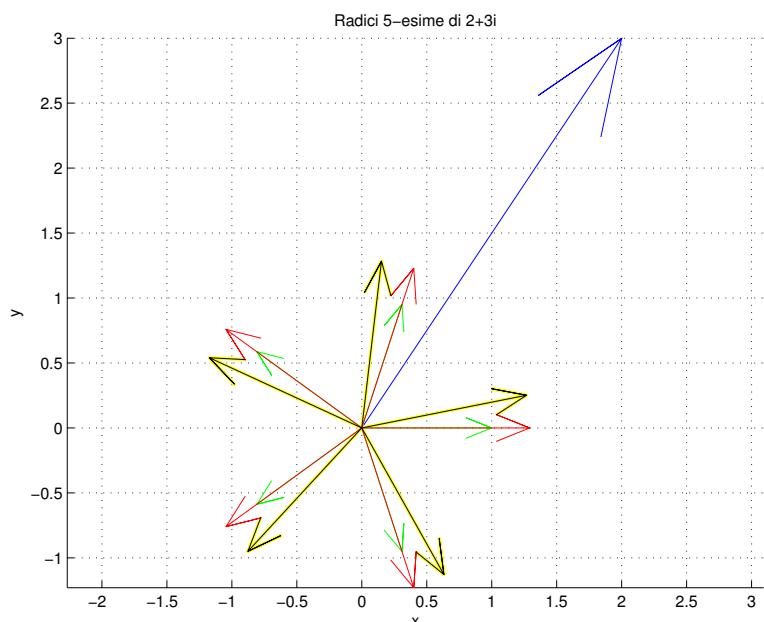
```

ESECUZIONE DEL PROGRAMMA

```

-1.1738 + 0.5408i -1.1738 + 0.5408i
-0.8771 - 0.9492i -0.8771 - 0.9492i
0.1516 + 1.2835i 0.6317 - 1.1275i
0.6317 - 1.1275i 1.2675 + 0.2524i
1.2675 + 0.2524i 0.1516 + 1.2835i

```



$Z = 2 + 3i$ e' rappresentato dalla freccia piú lunga blu,mentre quelle gialle sono le 5 radici di Z . A partire da una radice primitiva n-sima dell'unità sono state generate tutte le altre radici dell'unità e sono state disegnate in verde.In seguito sono state moltiplicate tali radici primitive,prima per il modulo della radice $w(1)$,[omotetia,frecce rosse],e in seguito per l'argomento[rotazione,frecce nere]ottenendo così le 5 radici di Z calcolate precedentemente con roots come si può notare anche dalla stampa delle radici.

2 Funzioni nel campo complesso

- 2.1 Visualizzare mediante il MATLAB Symbolic Math Toolbox alcune curve del piano (ellisse, arco di iperbole,...) descritte mediante equazioni parametriche. Nel caso di curve regolari, visualizzare anche la tangente in un punto scelto dall'utente (o scelto a caso). [liv.1]

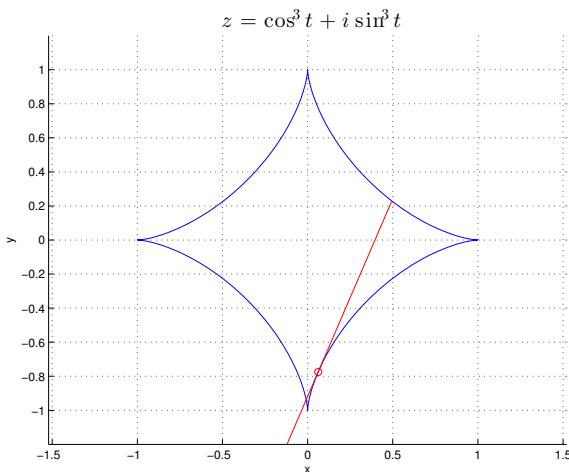
```

1 syms t real
2 if ~exist('scelta','var')
3     scelta=input('Scegliere la funzione:');
4 end
5 switch scelta
6 case 1
7     z=cos(t)^3+i*sin(t)^3;
8     n_regular=pi/2;%punti di non regolarita'
9     str='$z=\cos^3\{t\}+i\sin^3\{t\}$';
10 case 2
11     z=4*cos(t)-2*cos(2*t)+i*(4*sin(t)-2*sin(2*t));
12     n_regular=2*pi;%punti di non regolarita'
13     str='$z=4\cos\{t\}-2\cos\{2t\}+i(4\sin\{t\}-2\sin\{2t\})$';
14 otherwise
15     disp('Inserimento errato');
16     exit
17 end
18 hold on;grid on;axis([-1.2 1.2 -1.2 1.2]);
19 ezplot(real(z),imag(z)); %visualizzo la curva
20 p=2*pi*rand(1); %genero un numero casuale tra [0,2*pi]
21 pz=subs(z,'t',p); %valuto la funzione nel punto p
22 plot(real(pz),imag(pz),'or'); %visualizzo il punto sul grafico
23 d=diff(z); %calcolo la derivata della funzione
24 dp=subs(d,'t',p); %la valuto nel punto
25 if mod(p,n_regular)~=0 %se non mi trovo su punti di irregolarita'
26     retta_tang=pz+dp*t; %calcolo e visualizzo la retta tangente
27     h=ezplot(real(retta_tang),imag(retta_tang),[-1,1]);
28     set(h,'Color','r');
29     title(str,'interpreter','latex','fontsize',18);
30 else
31     disp(sprintf('Curva non regolare in %f\n.',p));
32 end

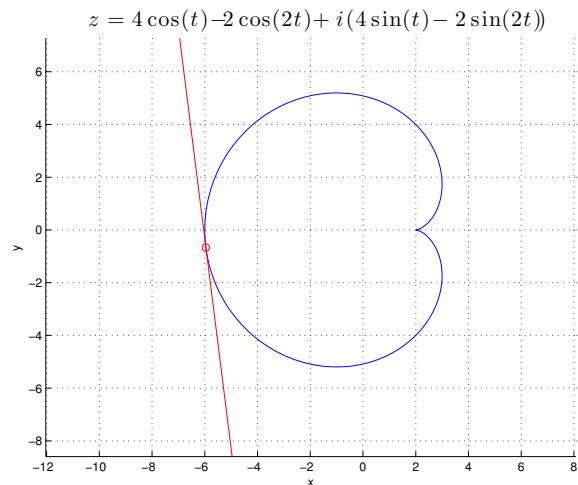
```

ESECUZIONE DEL PROGRAMMA

(SCELTA=1)



(SCELTA=2)



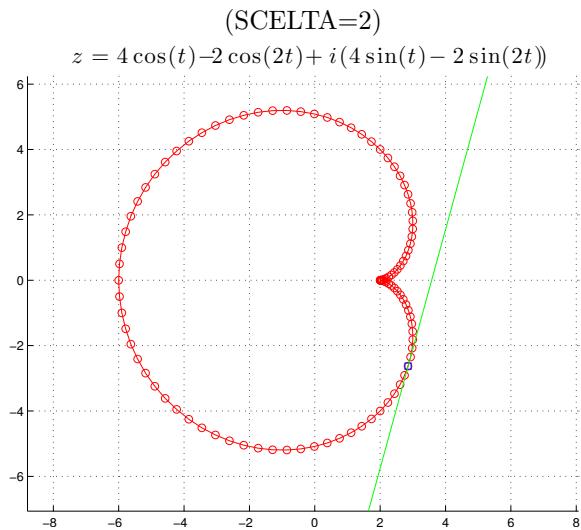
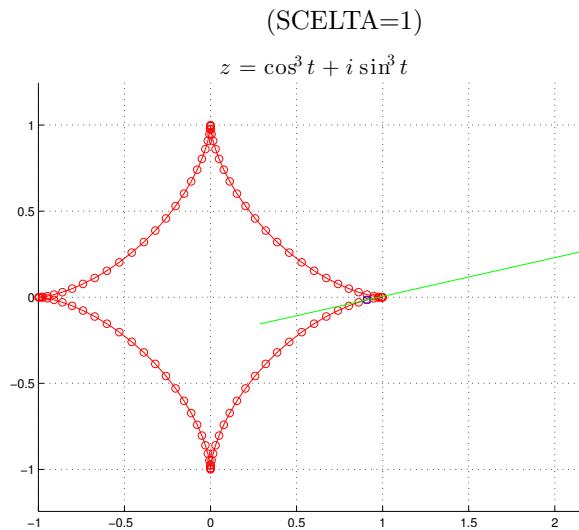
2.2 Ripetere in ambiente numerico l'esercizio precedente,sapendo che ora non si ha l'espressione della funzione ma solo dei campioni (punti della curva).[liv.1]

```

1 t=linspace(0,2*pi,101);
2 if ~exist('scelta','var')
3     scelta=input('Scegliere la funzione:');
4 end
5 switch scelta
6     case 1
7         z=cos(t).^3+i*sin(t).^3;
8         n_regular=pi/2;%punti di non regolarita'
9         str='$z=\cos^3\{t\}+i\sin^3\{t\}$';
10    case 2
11        z=4*cos(t)-2*cos(2*t)+i*(4*sin(t)-2*sin(2*t));
12        n_regular=2*pi;%punti di non regolarita'
13        str='$z=4\cos(t)-2\cos(2t)+i(4\sin(t)-2\sin(2t))$';
14    otherwise
15        disp('Inserimento errato');
16        exit
17    end
18 hold on;grid on;axis equal;
19 plot(real(z),imag(z),'or-'); %visualizzo i campioni
20 p=randi(101,1); %genero un numero casuale tra [1,101]
21 plot(real(z(p)),imag(z(p)),'sb'); %visualizzo il punto sul grafico
22 der=diff(z)./diff(t); %approssimo le derivate
23 if mod(t(p),n_regular)~=0 %se non mi trovo su punti di irregolarit
24     retta_tang=z(p)+der(p)*[-1,2]; %calcolo e visualizzo la retta tangente
25     plot(retta_tang,'g');
26     title(str,'interpreter','latex','fontsize',18);
27 else
28     disp(sprintf('Curva non regolare in t=%f\n',t(p)));
29 end

```

ESECUZIONE DEL PROGRAMMA



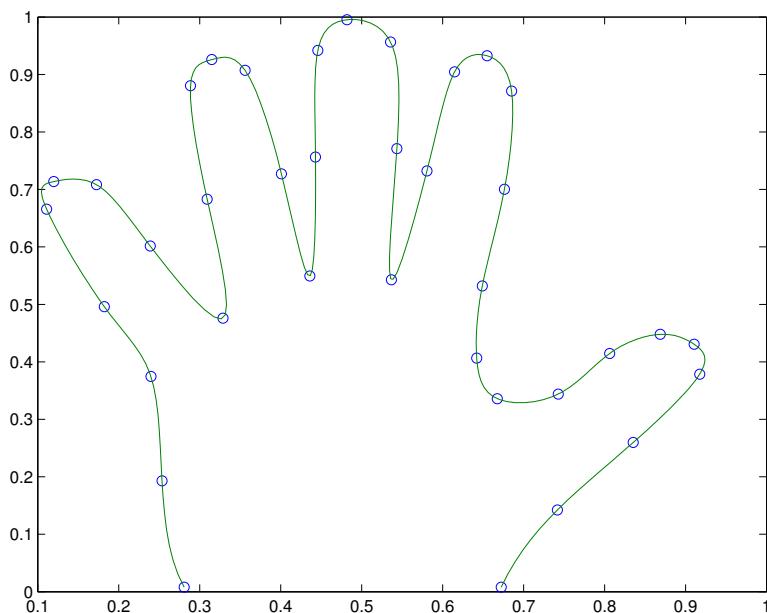
2.3 Ricostruire una curva 2D (regolare) mediante interpolazione: ad esempio da alcuni campioni ricostruire la propria mano oppure il profilo nei dati contenuti nella cartella dati_interpolazione.[liv.1]

```

1 x=mano(:,1);
2 y=mano(:,2);
3 i=(1:length(x));
4 tt=linspace(i(1),i(end),250);
5 xspline=spline(i,x,tt);%interpolo con spline cubica separatamente le ascisse
6 yspline=spline(i,y,tt);%e le ordinate
7 plot(x,y,'o',xspline,yspline);

```

ESECUZIONE DEL PROGRAMMA



2.4 Produrre i grafici di alcune funzioni complesse per visualizzarne alcune caratteristiche (continuitá, periodicitá sul dominio visualizzato) scegliendo opportunamente il dominio di visualizzazione e il tipo di griglia (circolare o rettangolare).[liv.1]

```

1 if ~exist('scelta','var')
2     scelta=input('Inserire scelta:');
3 end
4 switch scelta
5     case 1
6         f=inline('1./(z.^2-1)');
7         dim_griglia=2;%semiampiezza griglia rettangolare
8         str='Re $\frac{1}{z^2-1}$';
9     case 2
10        f=inline('sin(z)./abs(z)');
11        dim_griglia=2*pi;
12        str='Re $\frac{\sin(z)}{|z|}$';

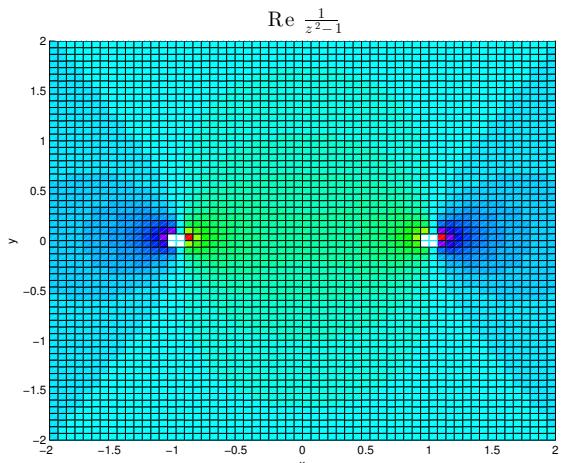
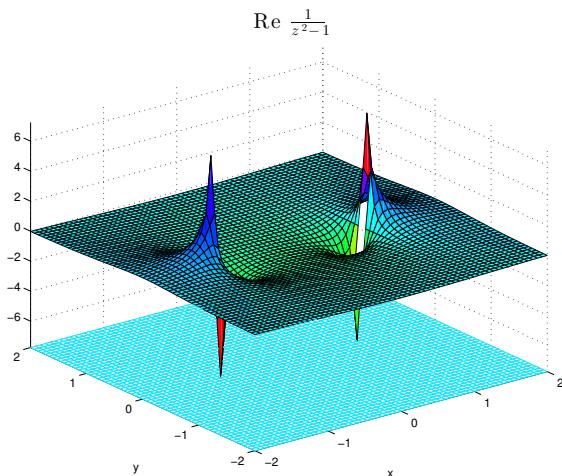
```

```

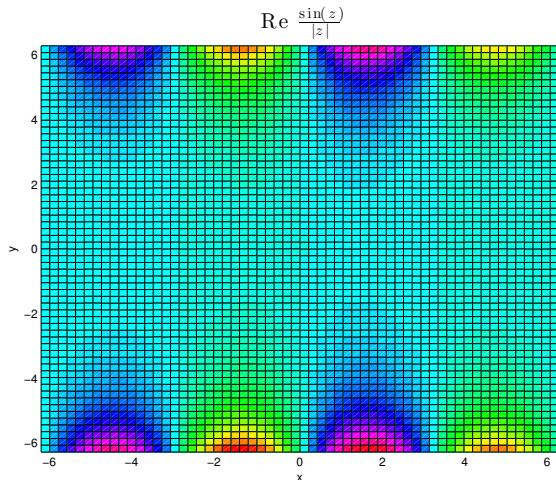
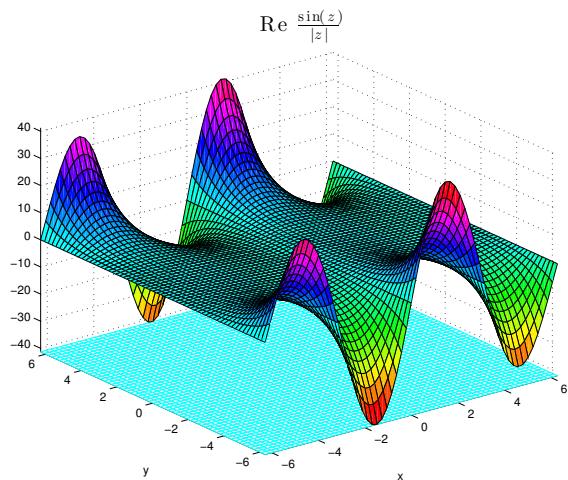
13 case 3
14     f=inline('exp(1./z)');
15     dim_griglia=5;
16     str='Re ${e}^{\frac{1}{z}}$';
17 case 4
18     f=inline('log(z)');
19     dim_griglia=5;
20     str='Re $\log(z)$';
21 otherwise
22     disp('Inserimento errato');
23     exit;
24 end
25 [x,y]=meshgrid(linspace(-dim_griglia,dim_griglia,61),...
26     linspace(-dim_griglia,dim_griglia,61));
27 z=x+i*y;
28 fz=f(z);
29 cplxmap(z,real(fz)*(1+i));
30 title(str,'interpreter','latex','fontsize',18);
31 axis tight
32 figure
33 cplxmap(z,real(fz)*(1+i));
34 view(2);
35 title(str,'interpreter','latex','fontsize',18);
36 axis tight

```

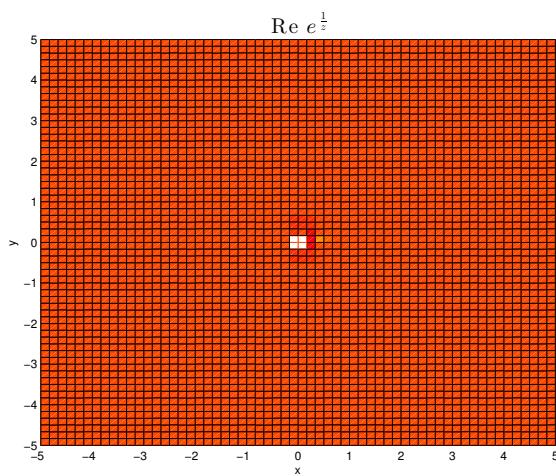
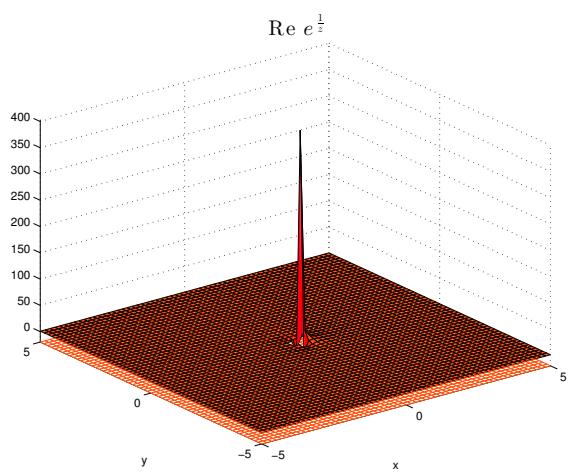
ESECUZIONE DEL PROGRAMMA(SCELTA=1, $\frac{1}{z^2-1}$)



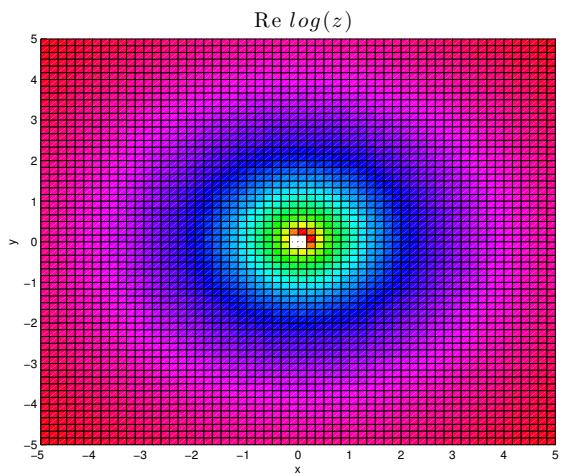
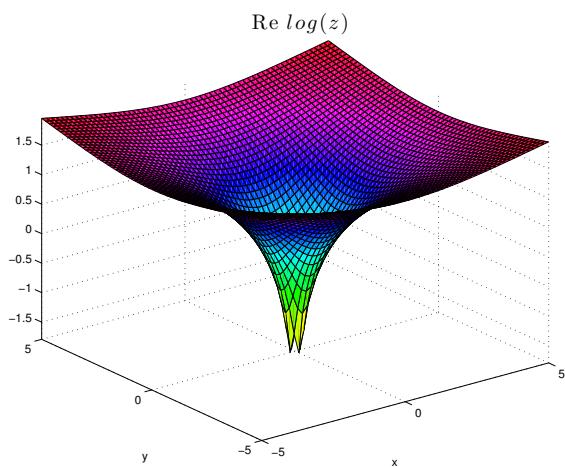
ESECUZIONE DEL PROGRAMMA(SCELTA=2, $\frac{\sin(z)}{|z|}$)



ESECUZIONE DEL PROGRAMMA(SCELTA=3, $e^{\frac{1}{z}}$)



ESECUZIONE DEL PROGRAMMA(SCELTA=4, $\log(z)$)



2.5 Visualizzare la continuitá o la discontinuitá di alcune funzioni complesse in un intorno (circolare e rettangolare) $I(z_0)$ di alcuni punti z_0 del piano complesso.[liv.1]

```

1 if ~exist('scelta','var')
2     scelta=input('Inserire scelta:');
3 end
4 switch scelta
5     case 1
6         f=inline('1./(z.^2-1)');
7         punti=[1,2+3i];
8         %il primo punto e' un punto di discontinuita' mentre il
9         %secondo punto e' un punto di continuita' della funzione
10        str='Re $\frac{1}{\{z\}^2-1}$';
11    case 2
12        f=inline('sin(z)./abs(z)');
13        punti=[0,2+3i];
14        str='Re $\frac{\sin\{z\}}{\left|z\right|}$';
15    case 3
16        f=inline('exp(1./z)');
17        punti=[0,2+3i];
18        str='Re ${e}^{\frac{1}{z}}$';
19    case 4
20        f=inline('log(z)');
21        punti=[0,2+3i];
22        str='Re $\log(z)$';
23    otherwise
24        disp('Inserimento errato');
25        exit;
26    end
27    for n=[1,2]
28        [x,y]=meshgrid(linspace(real(punti(n))-0.5,real(punti(n))+0.5,61),...
29                        linspace(imag(punti(n))-0.5,imag(punti(n))+0.5,61));
30        %Costruisco la griglia quadrata dove valutare la funzione di

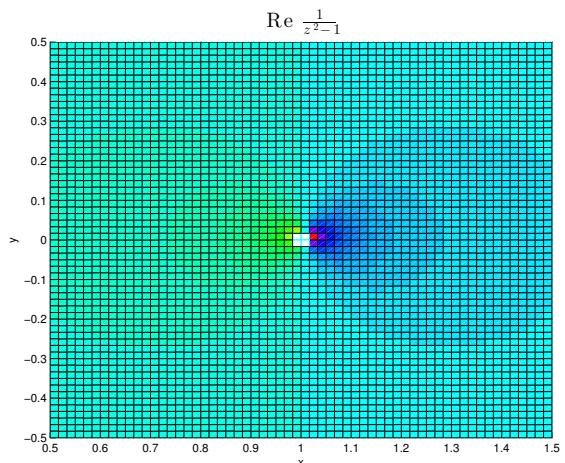
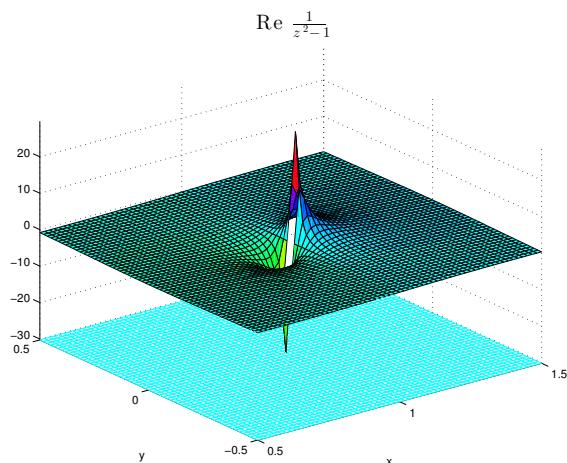
```

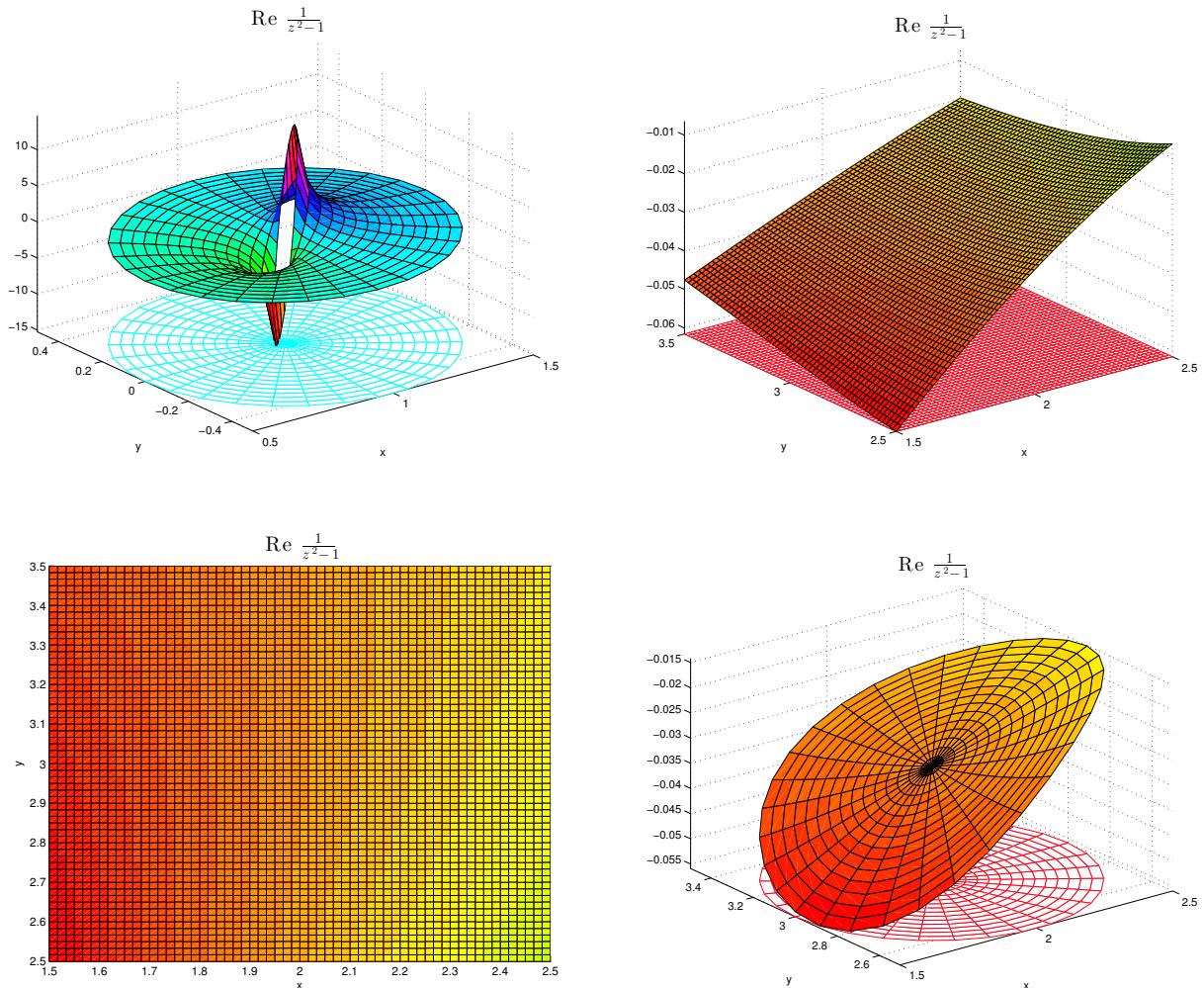
```

31 %ampiezza 0.5 centrata nel punto in esame
32 z=x+i*y;
33 fz=f(z);
34 figure
35 cplxmap(z,real(fz)*(1+i));
36 title(str,'interpreter','latex','fontsize',18);
37 xlabel('x');ylabel('y');
38 axis tight
39 figure
40 cplxmap(z,real(fz)*(1+i));
41 view(2);
42 title(str,'interpreter','latex','fontsize',18);
43 xlabel('x');ylabel('y');
44 axis tight
45 z=punti(n)+0.5*cplxgrid(15);
46 %Costruisco la griglia circolare di ampiezza 0.5 centrata nel punto in
47 %esame
48 fz=f(z);
49 figure
50 cplxmap(z,real(fz)*(1+i));
51 title(str,'interpreter','latex','fontsize',18);
52 xlabel('x');ylabel('y');
53 axis tight
54 end

```

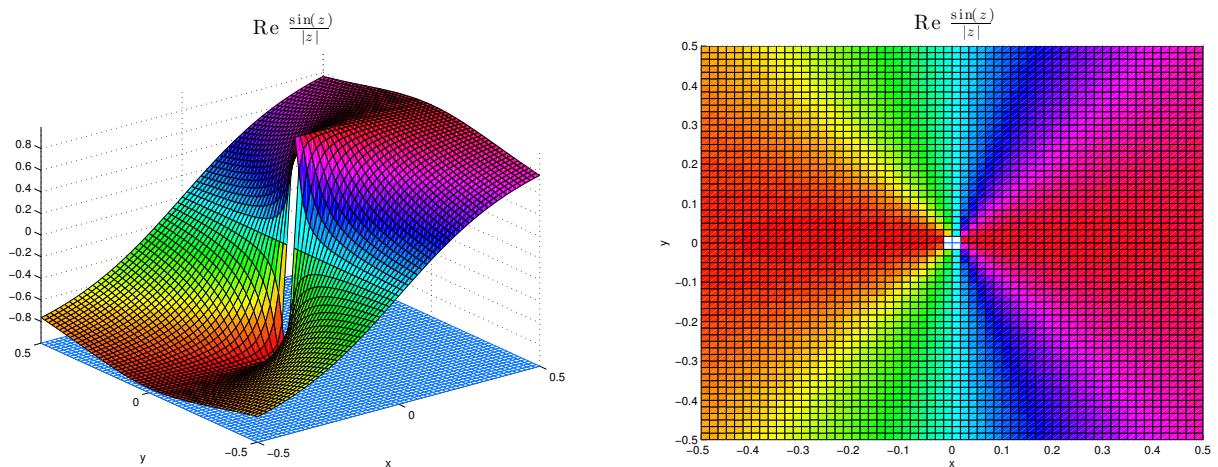
ESECUZIONE DEL PROGRAMMA(SCELTA=1, $\frac{1}{z^2-1}$, punti[(1), (2 + 3i)])

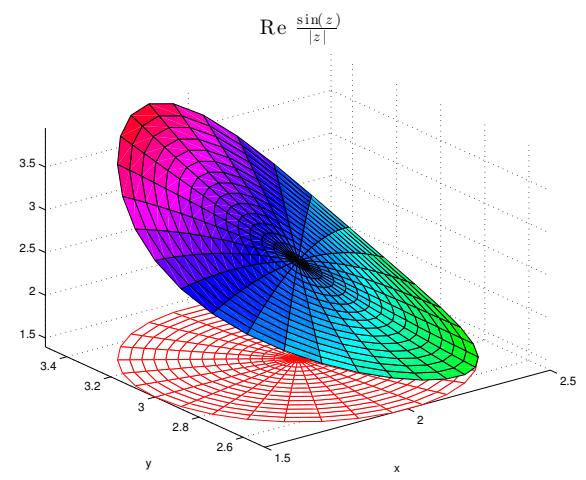
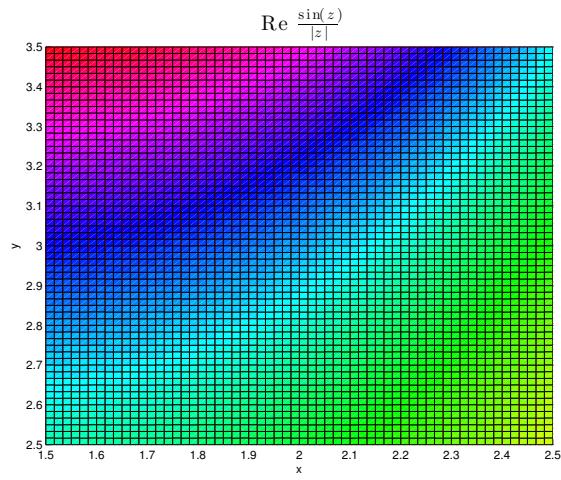
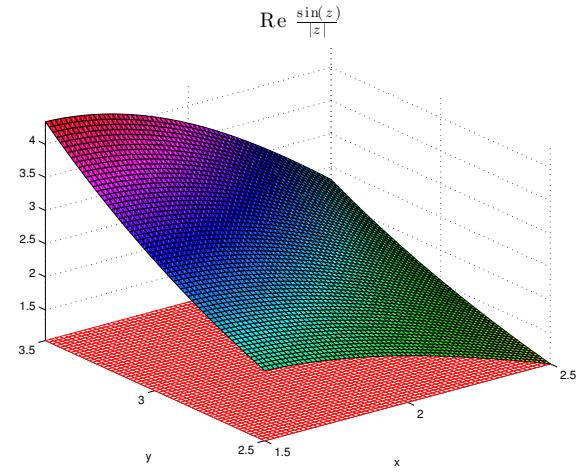
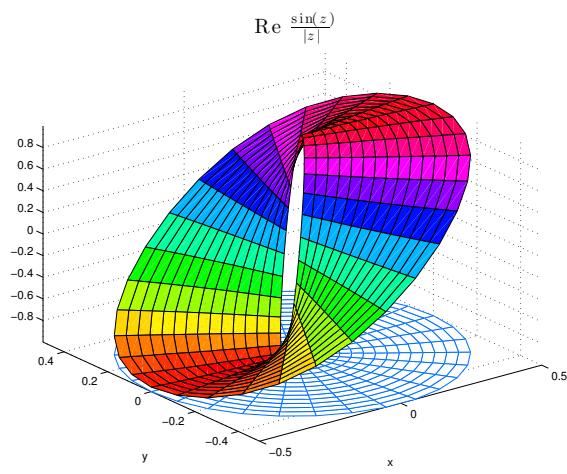




Le prime tre figure sono relative al punto 1, mentre le altre al punto $2 + 3i$. Come si può notare da tali figure la funzione non è continua nel primo punto mentre lo è nel secondo.

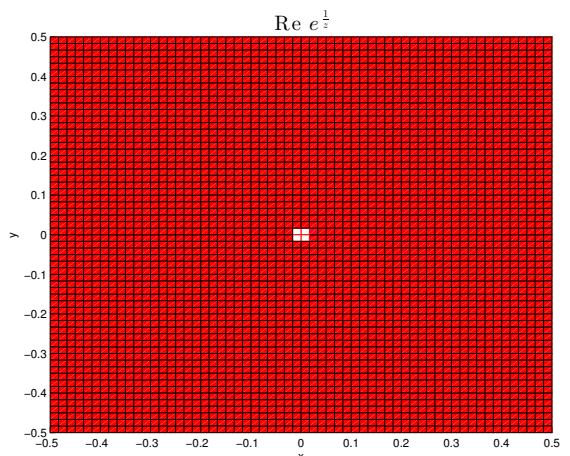
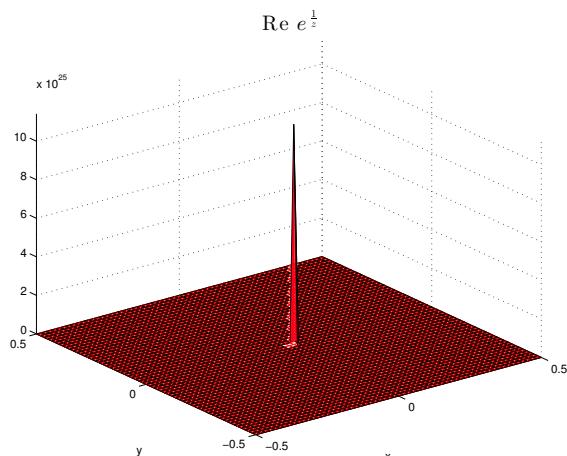
ESECUZIONE DEL PROGRAMMA (SCELTA=2, $\frac{\sin(z)}{|z|}$, punti[(0), (2 + 3i)])

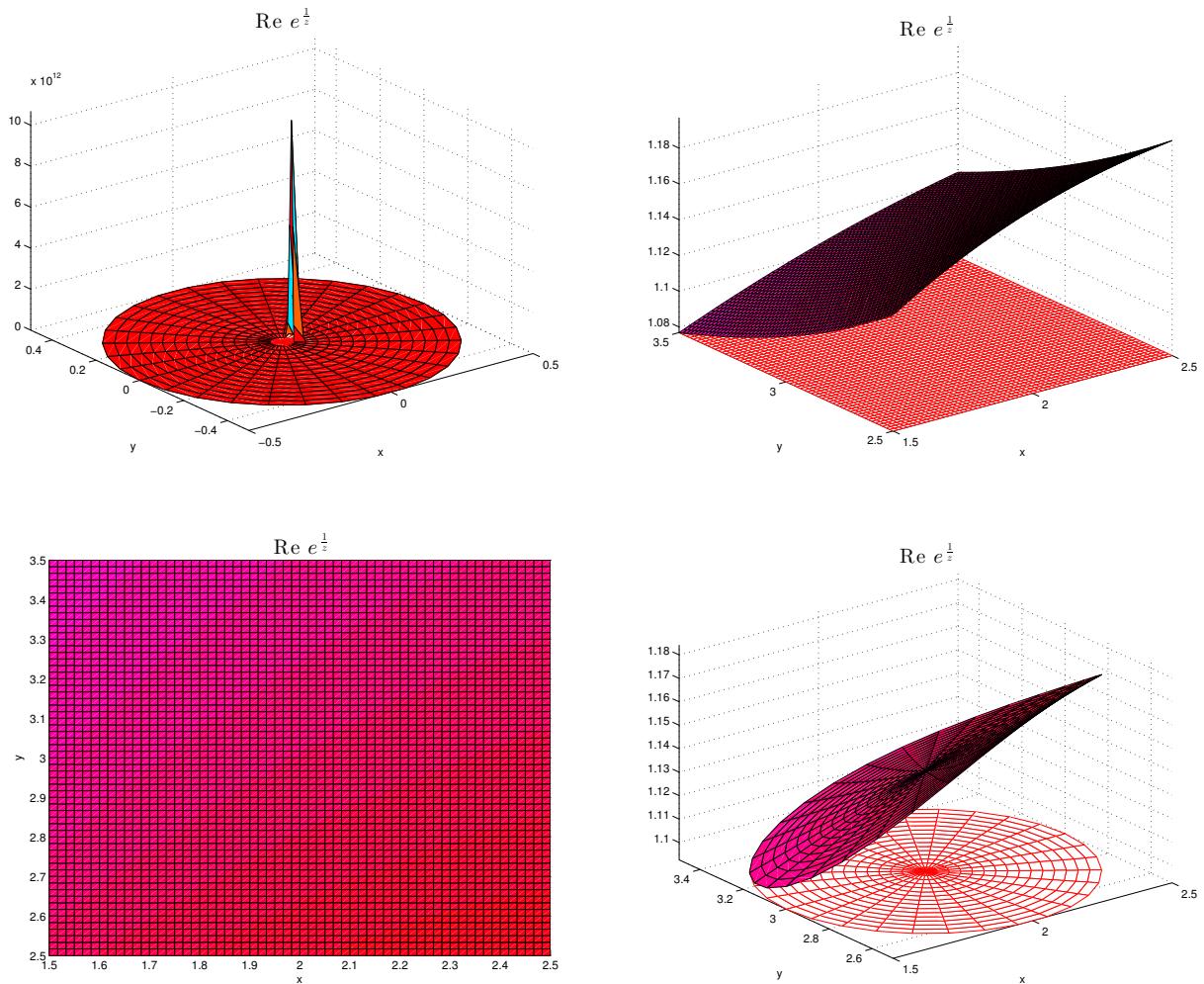




Le prime tre figure sono relative al punto 0 , mentre le altre al punto $2 + 3i$. Come si può notare da tali figure la funzione non è continua nel primo punto mentre lo è nel secondo.

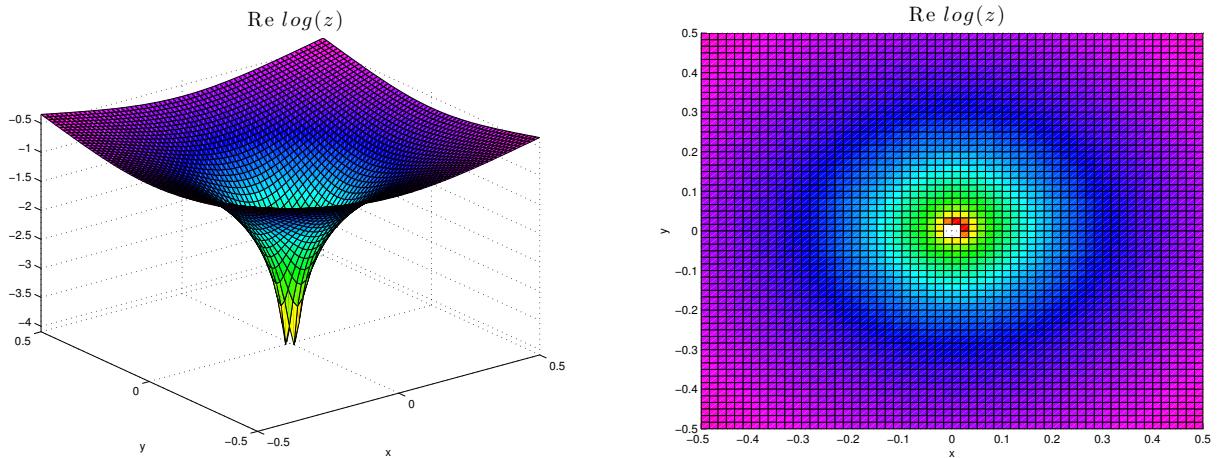
ESECUZIONE DEL PROGRAMMA (SCELTA=3, $e^{\frac{1}{z}}$, punti[(0) , $(2 + 3i)$])

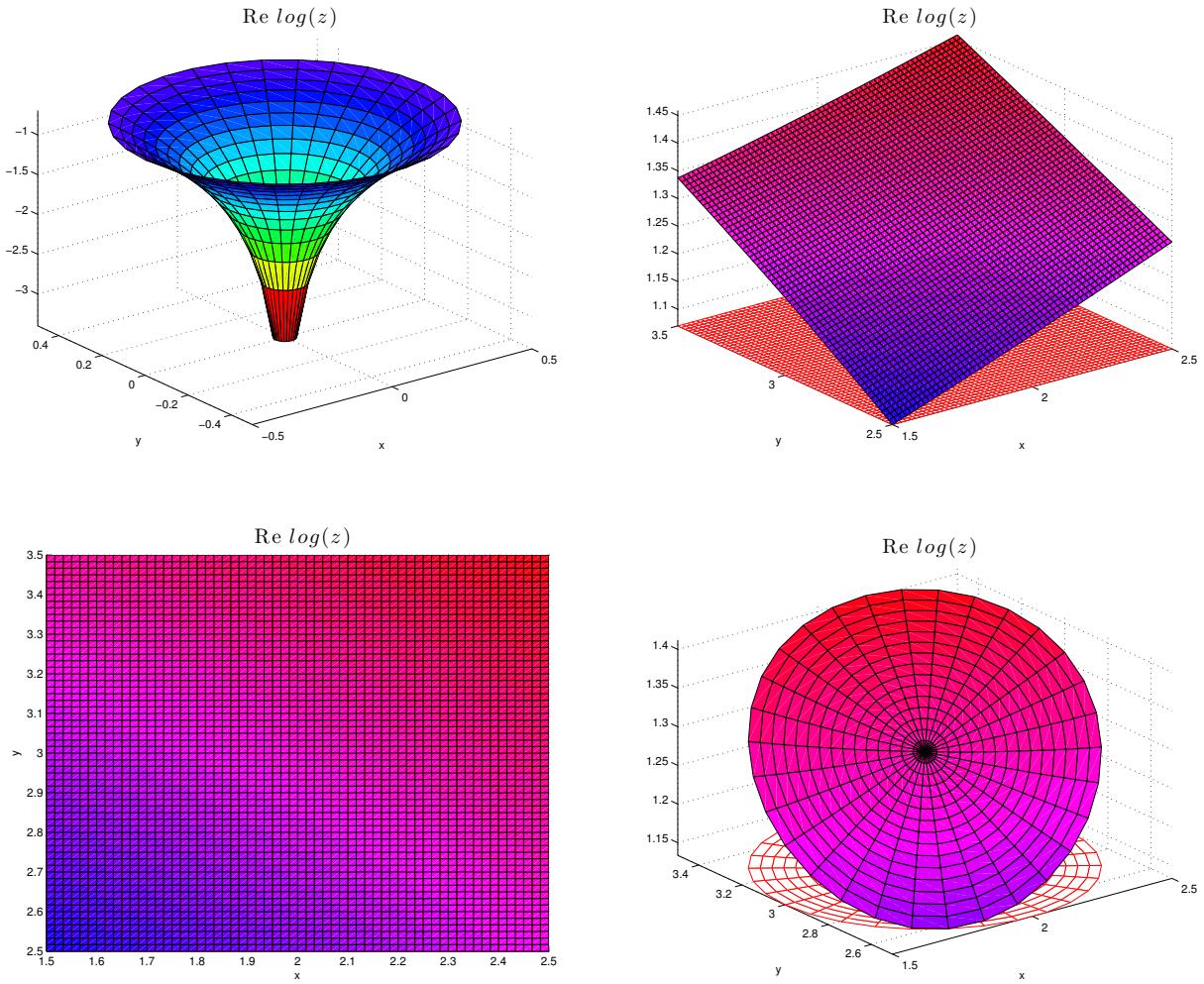




Le prime tre figure sono relative al punto 0 , mentre le altre al punto $2 + 3i$. Come si può notare da tali figure la funzione non è continua nel primo punto mentre lo è nel secondo.

ESECUZIONE DEL PROGRAMMA(SCELTA=4, $\log(z)$, punti[$(0), (2 + 3i)$])





Le prime tre figure sono relative al punto 0 , mentre le altre al punto $2 + 3i$. Come si può notare da tali figure la funzione non è continua nel primo punto mentre lo è nel secondo.

2.6 Verificare e visualizzare la continuità o la discontinuità di alcune funzioni complesse in un intorno di alcuni punti z_0 del piano complesso mediante Symbolic Math Toolbox. [liv.1]

```

1 syms x y theta real;
2 syms rho positive;
3 z=x+i*y;
4 if ~exist('scelta','var')
5     scelta=input('Inserire scelta:');
6 end
7 switch scelta
8 case 1
9     f=1/(z^2-1);
10    punti=[1,2+3i];
11    %Punti di interesse
12    str='Re $\frac{1}{\{z\}^2-1}$';
13 case 2

```

```

14      f=sin(z)/abs(z);
15      punti=[0,2+3i];
16      %Punti di interesse
17      str='Re $\frac{\sin{(z)}}{\left|z\right|}$';
18  case 3
19      f=exp(1/z);
20      punti=[0,2+3i];
21      %Punti di interesse
22      str='Re ${e}^{\frac{1}{z}}$';
23  case 4
24      f=log(z);
25      punti=[0,2+3i];
26      %Punti di interesse
27      str='Re $\log(z)$';
28 otherwise
29     disp('Inserimento errato');
30     exit;
31 end
32 for n=[1,2]
33 figure
34 ezsurf(real(f),[real(punti(n))-0.2,real(punti(n))+0.2, ...
35           imag(punti(n))-0.2,imag(punti(n))+0.2]);
36 %visualizzo la funzione in un intorno rettangolare del punto,di
37 %semiasse 0.2
38 title(str,'interpreter','latex','fontsize',18);
39 axis tight;xlabel('x');ylabel('y');
40 figure(copyobj(gcf,0));
41 view(2);
42 z=punti(n)+rho*exp(i*theta);
43 fz=subs(f,'x+i*y',z);
44 figure
45 ezsurf(real(z),imag(z),real(fz),[0 0.2 -pi pi]);
46 %visualizzo la funzione in un intorno circolare centrato nel punto,con
47 %raggio 0.2
48 title(str,'interpreter','latex','fontsize',18);
49 axis tight;xlabel('rho');ylabel('theta');
50 figure(copyobj(gcf,0));
51 view(2);
52 fprintf('Limite per rho che tende a 0,punto=%d+%di\n',...
53       real(punti(n)),imag(punti(n)));
54 disp(vpa(simplify(subs(fz,rho,0)),5));
55 fprintf('Funzione valutata nel punto=%d+%di\n',...
56       real(punti(n)),imag(punti(n)));
57 disp(vpa(subs(f,{x,y},{real(punti(n)),imag(punti(n))}),5));
58 %verifico se il limite per rho->0 e' uguale al valore della funzione
59 %nel punto in modo da verificare la continuita'.
60 end

```

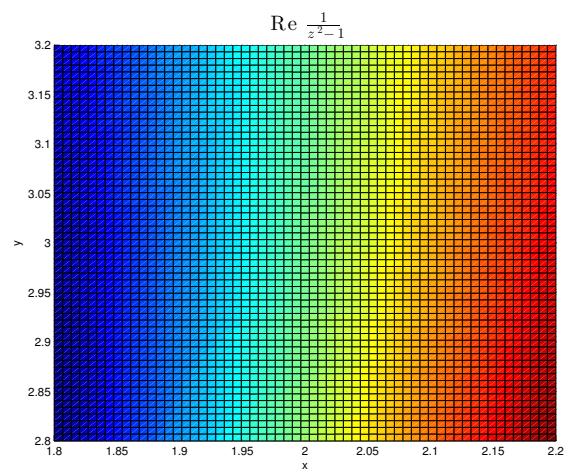
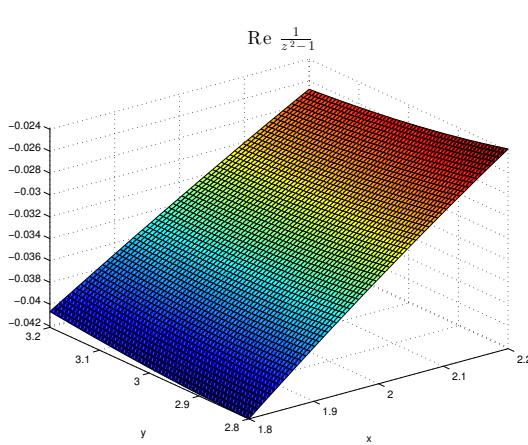
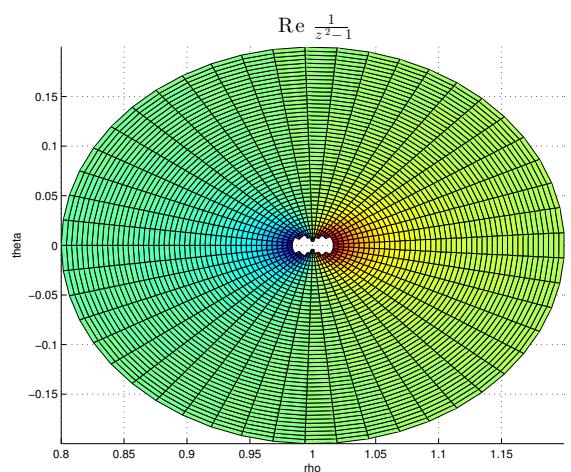
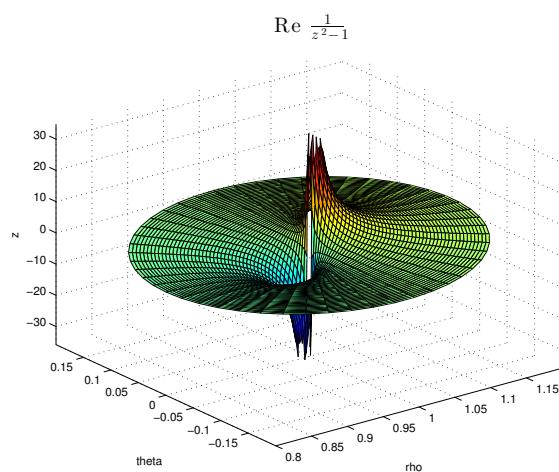
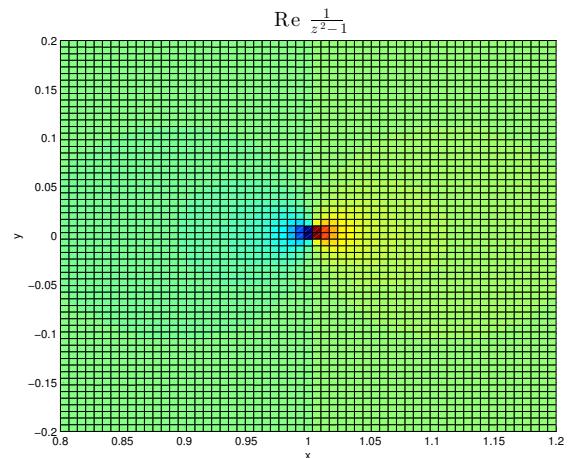
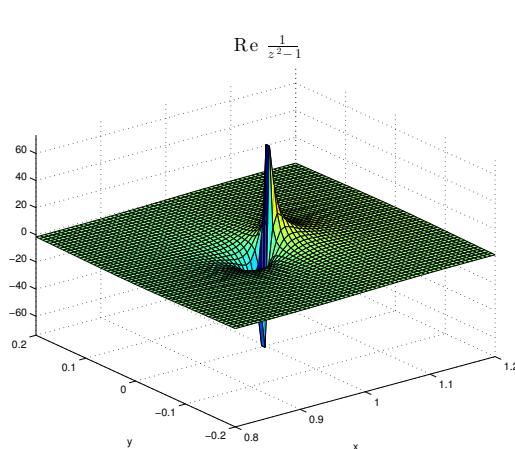
ESECUZIONE DEL PROGRAMMA(SCELTA=1, $\frac{1}{z^2-1}$, punti[(1),(2 + 3i)])

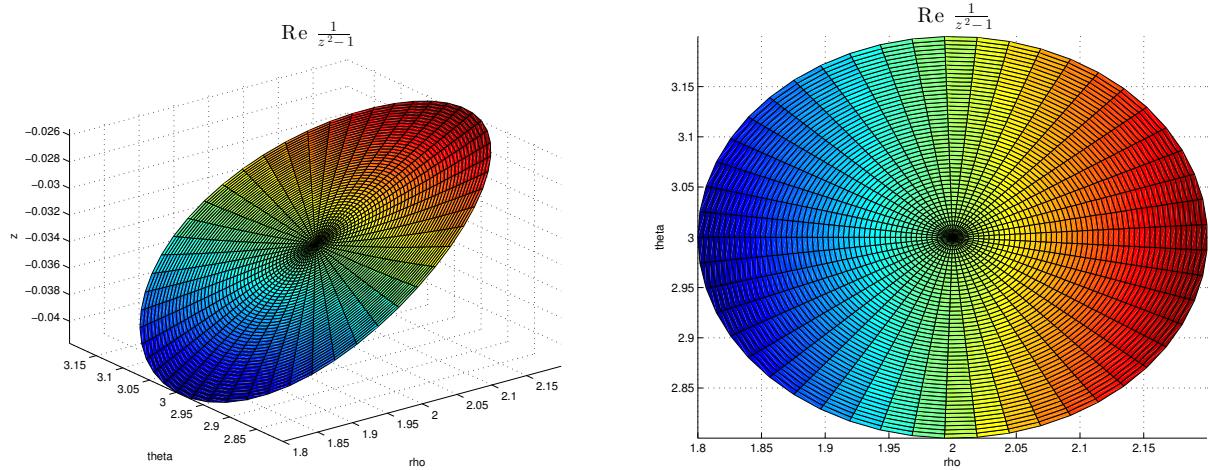
Limite per rho che tende a 0,punto=1+0i
Inf

Funzione valutata nel punto=1+0i
Inf

Limite per rho che tende a 0, punto=2+3i
 - 0.033333 - 0.066667*i

Funzione valutata nel punto=2+3i
 - 0.033333 - 0.066667*i





Il limite non esiste nel primo punto (1) la funzione quindi non è continua in tale punto. Il secondo limite esiste ed è pari al valore della funzione nel punto, quindi la funzione è continua nel punto $2+3i$.

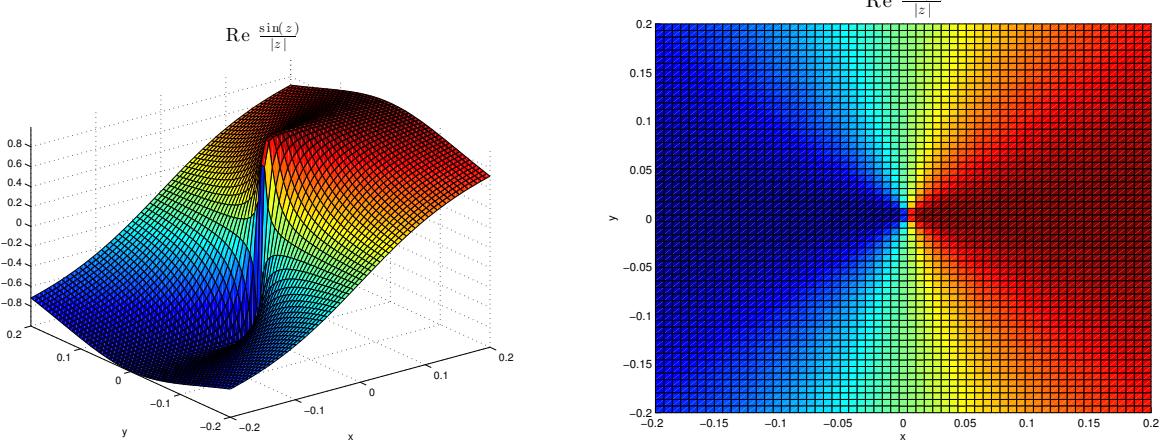
ESECUZIONE DEL PROGRAMMA(SCELTA=2, $\frac{\sin(z)}{|z|}$, punti[(0), (2 + 3i)])

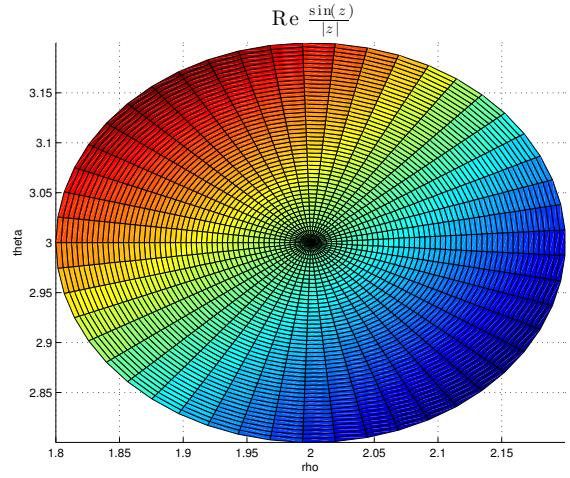
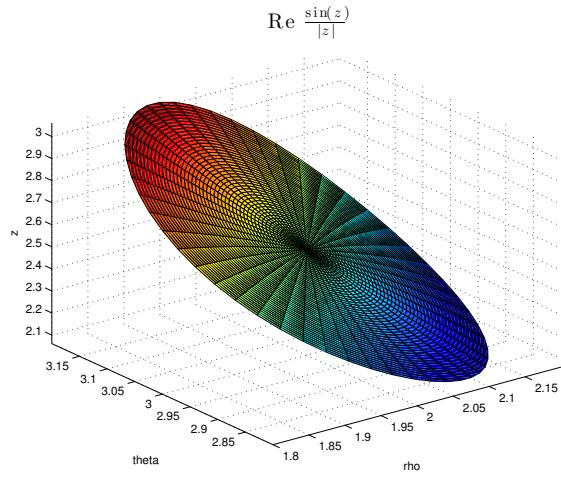
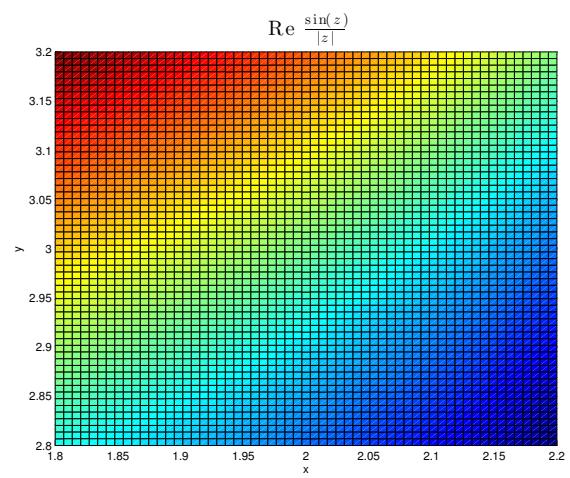
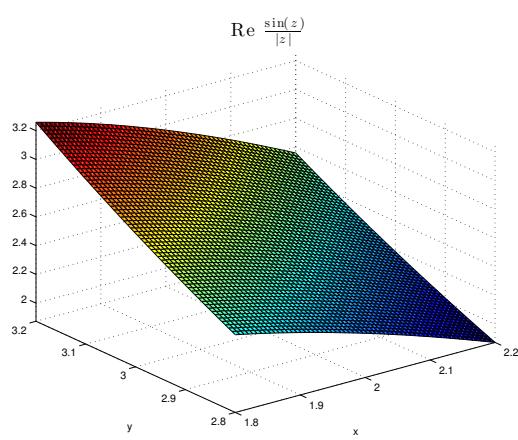
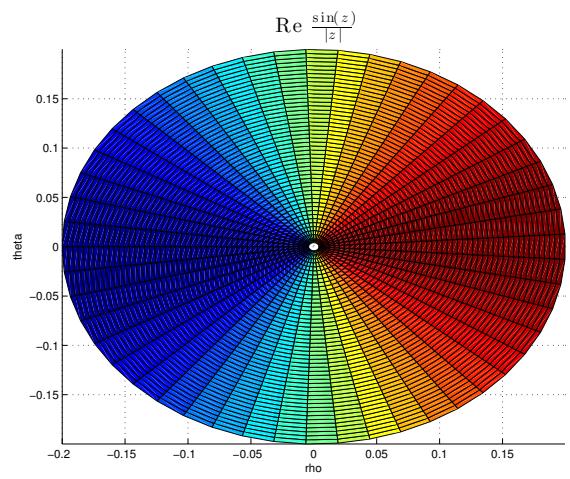
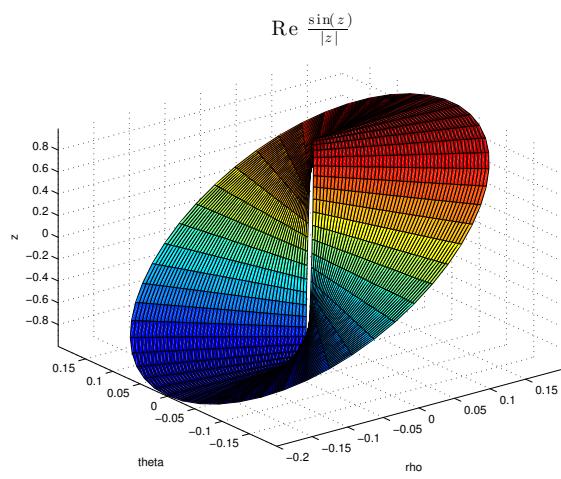
Limite per rho che tende a 0, punto=0+0i
NaN

Funzione valutata nel punto=0+0i
NaN

Limite per rho che tende a 0, punto=2+3i
2.539 - 1.1562*i

Funzione valutata nel punto=2+3i
2.539 - 1.1562*i





Il limite non esiste nel primo punto (0) la funzione quindi non è continua in tale punto. Il secondo limite esiste ed è pari al valore della funzione nel punto, quindi la funzione è continua nel punto $2+3i$.

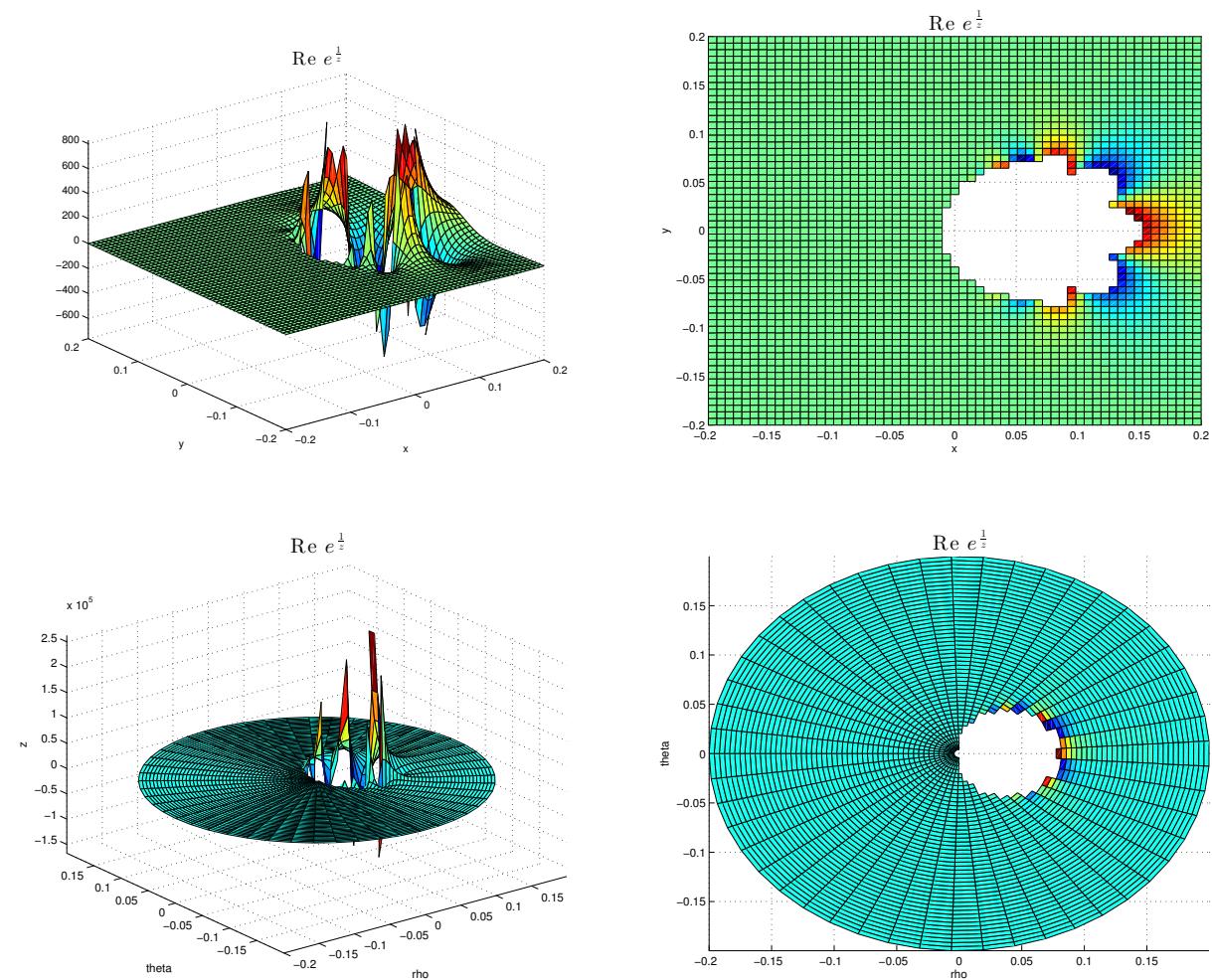
ESECUZIONE DEL PROGRAMMA(SCELTA=3, $e^{\frac{1}{z}}$, punti[(0), (2 + 3i)])

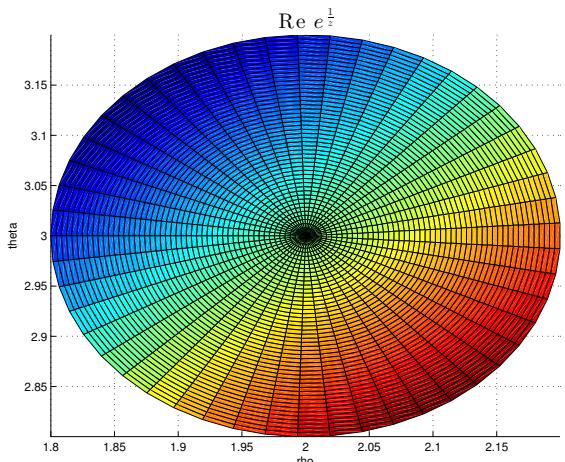
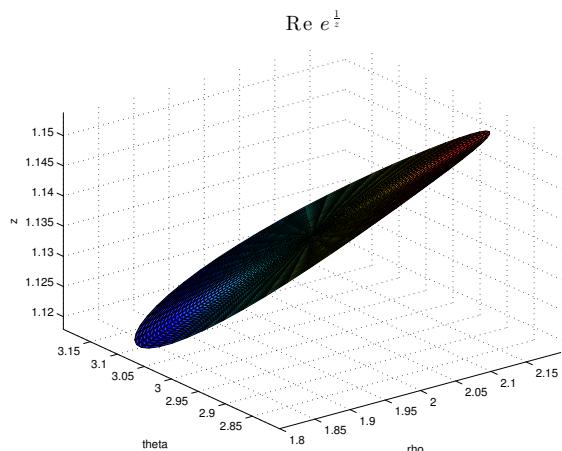
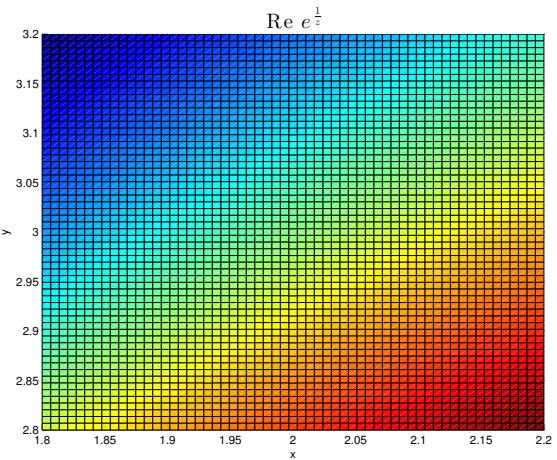
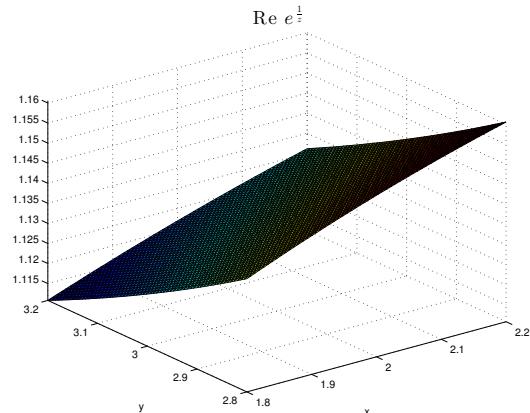
Limite per rho che tende a 0, punto=0+0i
Inf

Funzione valutata nel punto=0+0i
Inf

Limite per rho che tende a 0, punto=2+3i
1.1354 - 0.26677*i

Funzione valutata nel punto=2+3i
1.1354 - 0.26677*i





Il limite non esiste nel primo punto (0) la funzione quindi non è continua in tale punto. Il secondo limite esiste ed è pari al valore della funzione nel punto, quindi la funzione è continua nel punto $2+3i$.

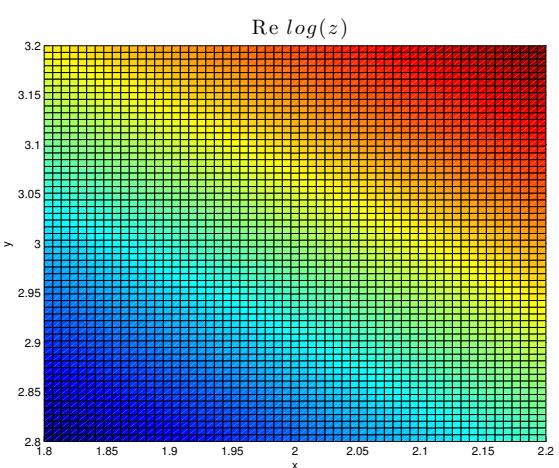
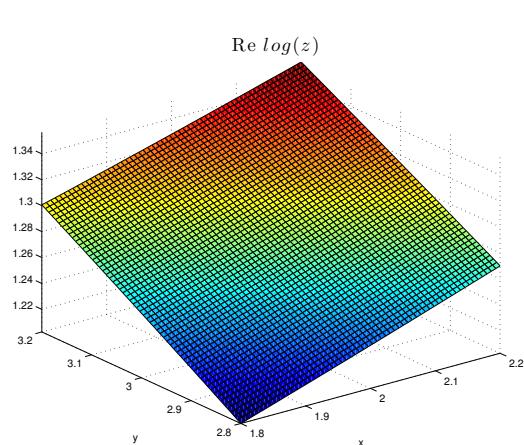
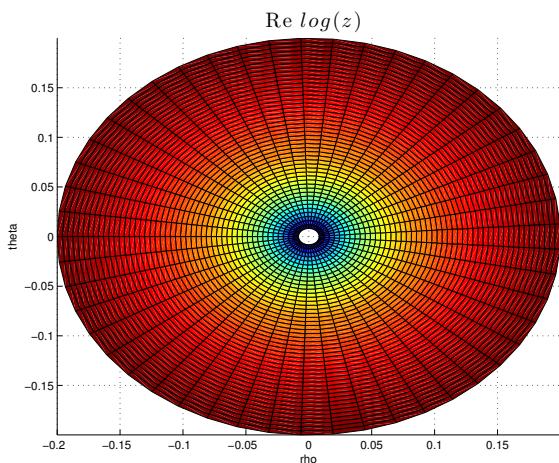
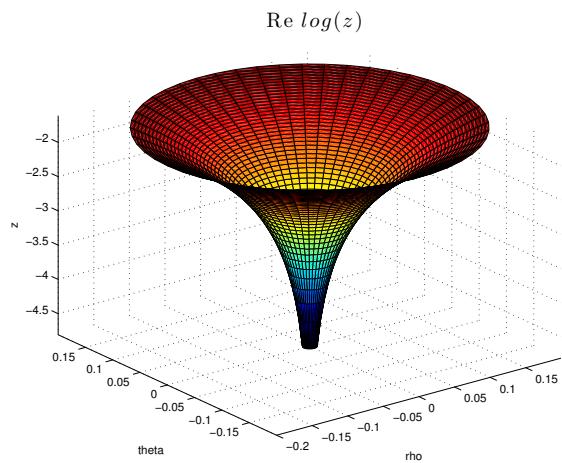
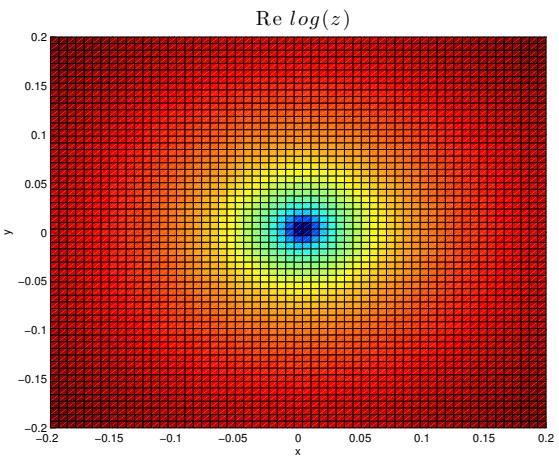
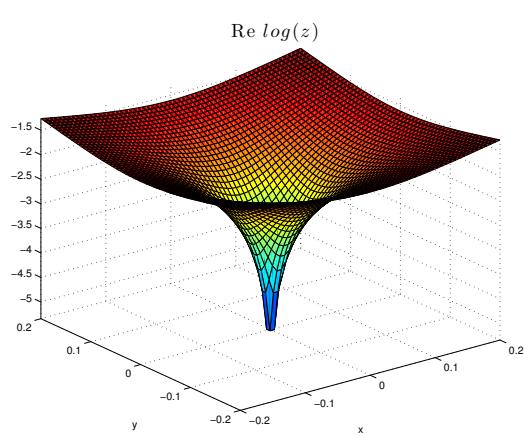
ESECUZIONE DEL PROGRAMMA(SCELTA=4, $\log(z)$, punti[(0), (2 + 3i)])

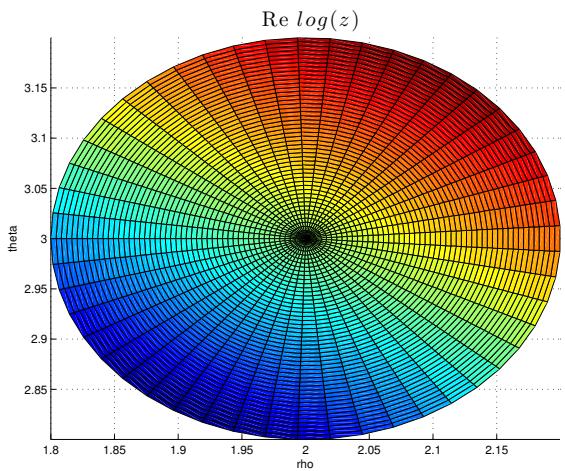
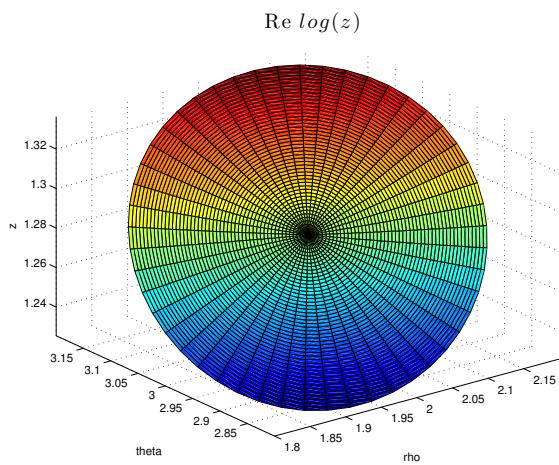
Limite per rho che tende a 0, punto=0+0i
-Inf

Funzione valutata nel punto=0+0i
-Inf

Limite per rho che tende a 0, punto=2+3i
1.2825 + 0.98279*i

Funzione valutata nel punto=2+3i
1.2825 + 0.98279*i





Il limite non esiste nel primo punto (0) la funzione quindi non è continua in tale punto. Il secondo limite esiste ed è pari al valore della funzione nel punto, quindi la funzione è continua nel punto $2+3i$.

2.7 Visualizzare l'olomorfia o la non olomorfia di alcune funzioni complesse in un intorno (circolare e rettangolare), $I(z_0)$ di alcuni punti z_0 del piano complesso. [liv.1]

```

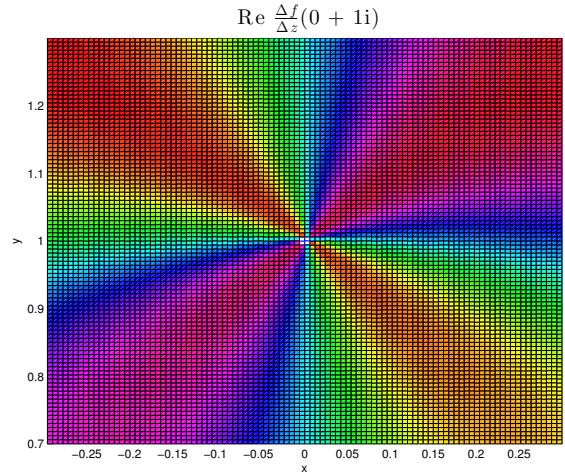
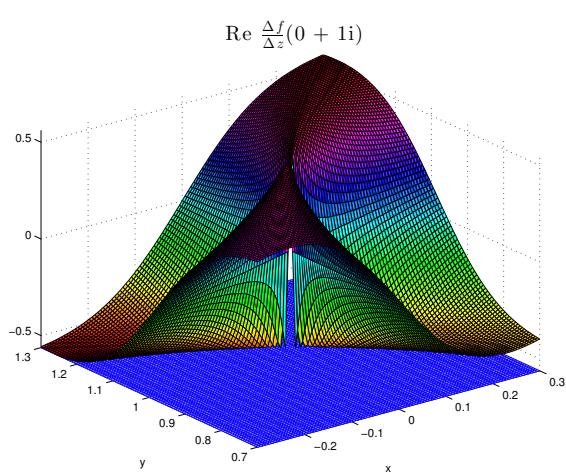
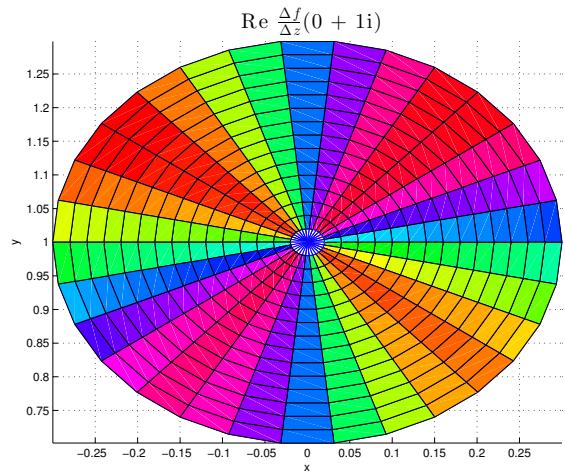
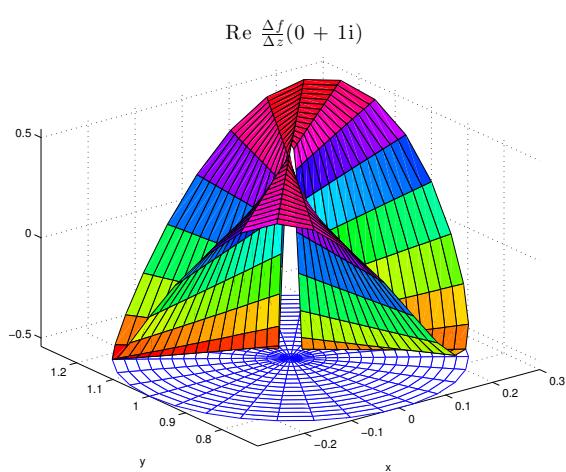
1 if ~exist('scelta','var')
2     scelta=input('Inserire scelta:');
3 end
4 switch scelta
5     case 1
6         f=inline('abs(z)');
7         punti=[i,2+3i];
8     case 2
9         f=inline('imag(z)');
10        punti=[4+2i,2+3i];
11    case 3
12        f=inline('exp(z)');
13        punti=[4+2i,2+3i];
14    case 4
15        f=inline('sin(z)');
16        punti=[4+2i,2+3i];
17    otherwise
18        disp('Inserimento errato');
19        exit;
20 end
21 for n=[1,2]
22     r=0.3;
23     z0=punti(n);
24     I=z0+r*cplxgrid(15);
25     %intorno centrato in z0 di raggio r
26     dz=I-z0;
27     Ri=(f(I)-f(z0))./dz;
28     %Rapporto incrementale
29     figure
30     cplxmap(I,real(Ri)*(1+i));
31     str=['Re $\frac{\Delta f}{\Delta z}$',...
32         sprintf(' (%d + %di)',real(punti(n)),imag(punti(n)))];
```

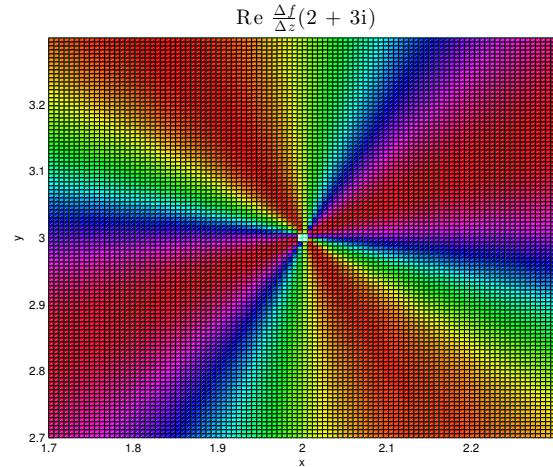
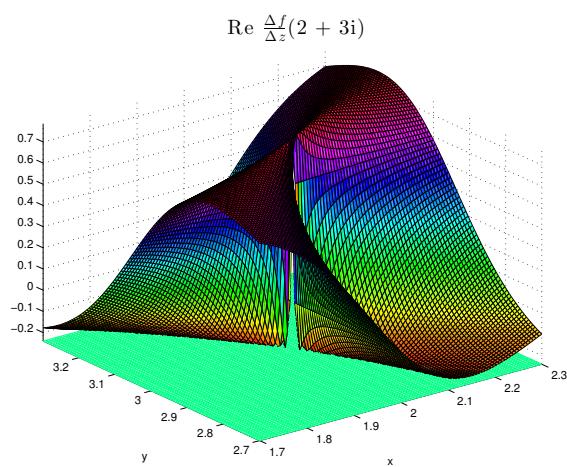
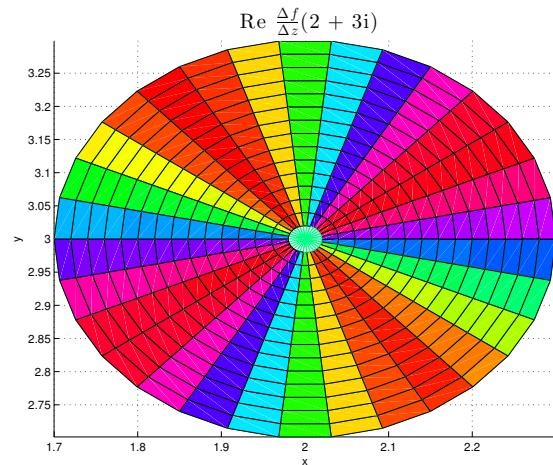
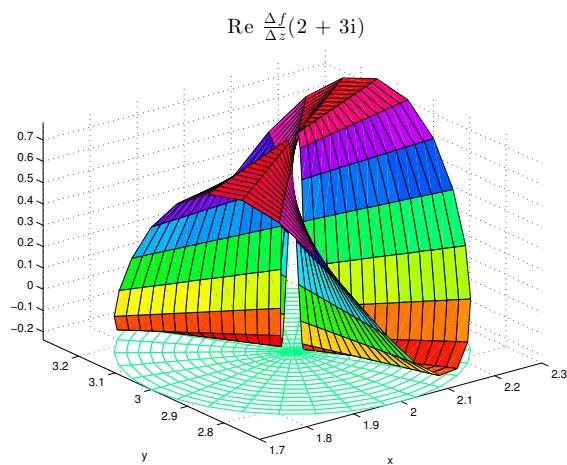
```

33 title(str,'interpreter','latex','fontsize',18);
34 axis tight;xlabel('x');ylabel('y');
35 figure(copyobj(gcf,0));
36 view(2);
37 [x,y]=meshgrid(linspace(real(z0)-r,real(z0)+r,101),...
38 linspace(imag(z0)-r,imag(z0)+r,101));
39 %griglia rettangolare centrata in z0 di semiampiezza r
40 I=x+i*y;
41 dz=I-z0;
42 Ri=(f(I)-f(z0))./dz;
43 %Rapporto incrementale
44 figure
45 cplxmap(I,real(Ri)*(1+i));
46 axis tight;xlabel('x');ylabel('y');
47 title(str,'interpreter','latex','fontsize',18);
48 figure(copyobj(gcf,0));
49 view(2);
50 end

```

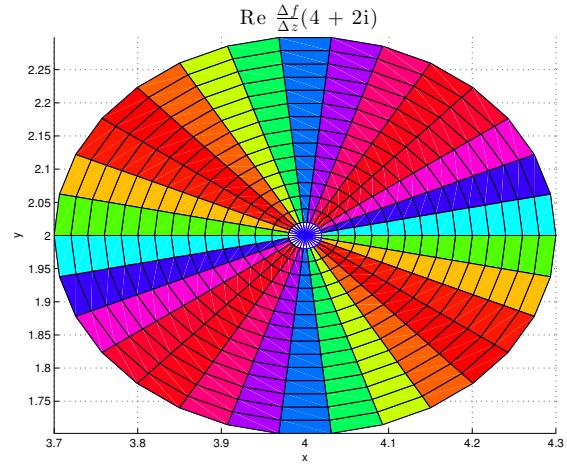
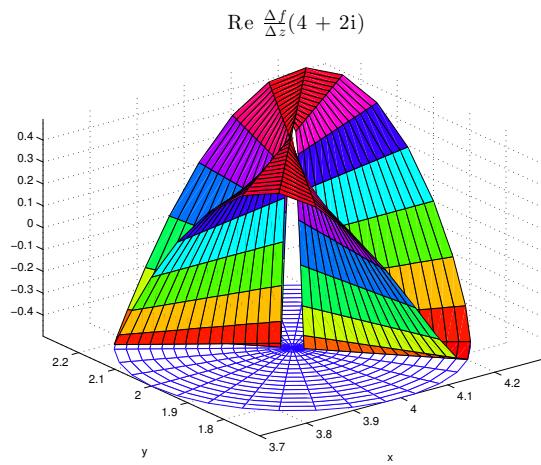
ESECUZIONE DEL PROGRAMMA(SCELTA=1, $f(z) = |z|$, punti[(i),(2 + 3i)])

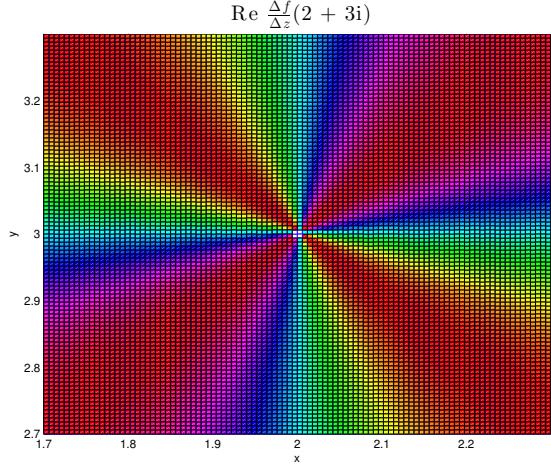
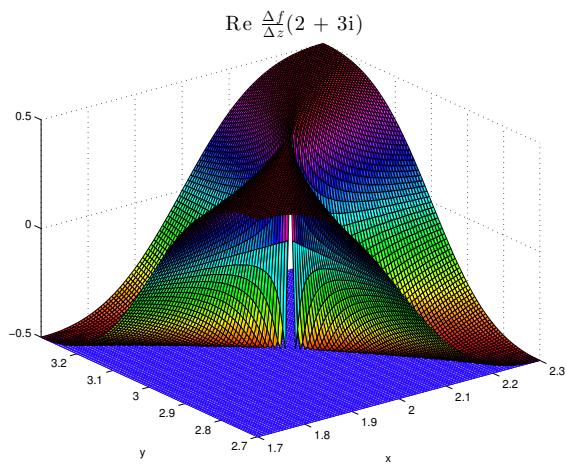
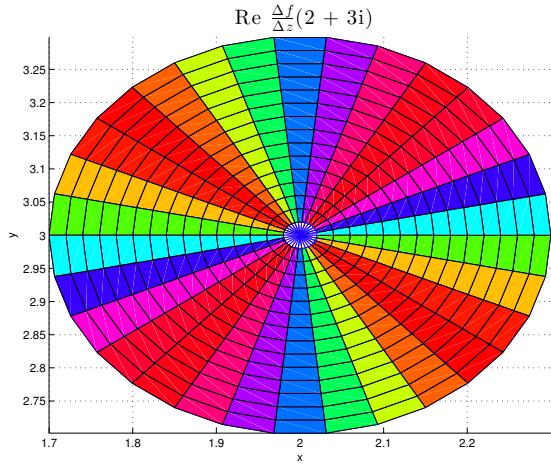
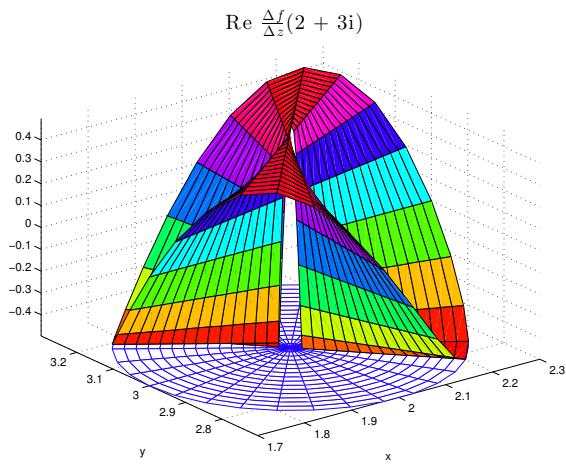
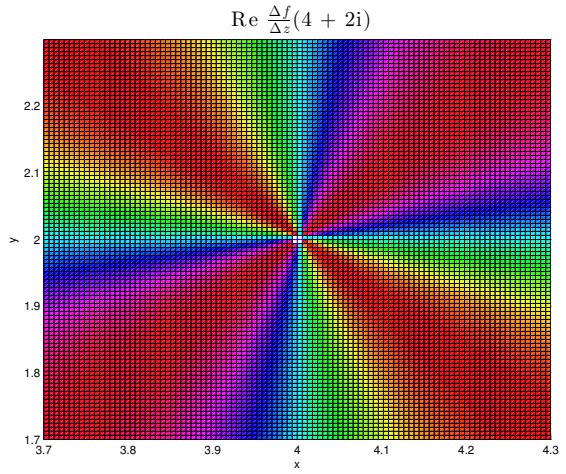
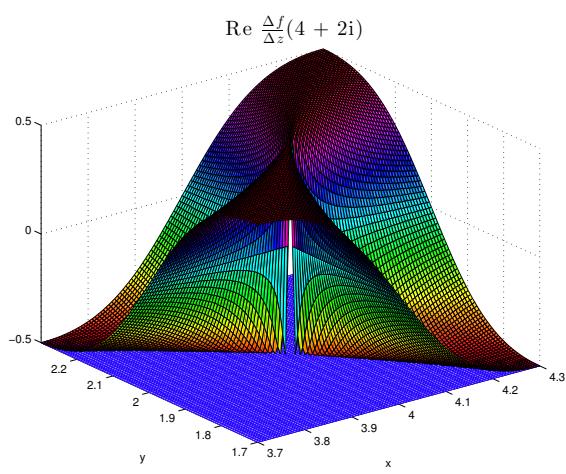




Come si osserva dalle immagini precedenti il limite del rapporto incrementale nei 2 punti non esiste. La funzione $|z|$ non è olomorfa in tali punti.

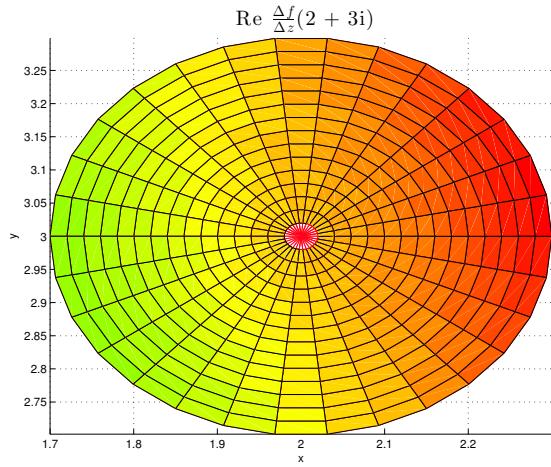
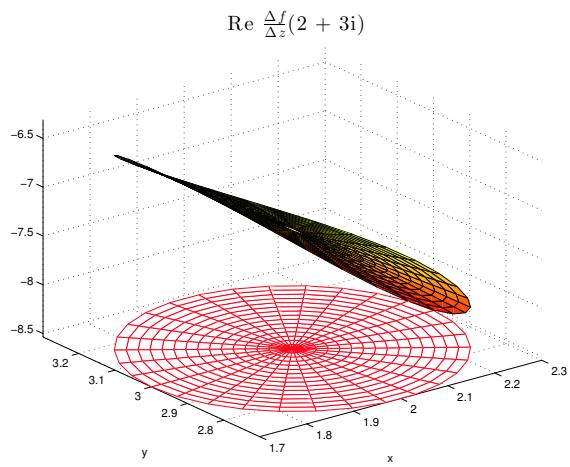
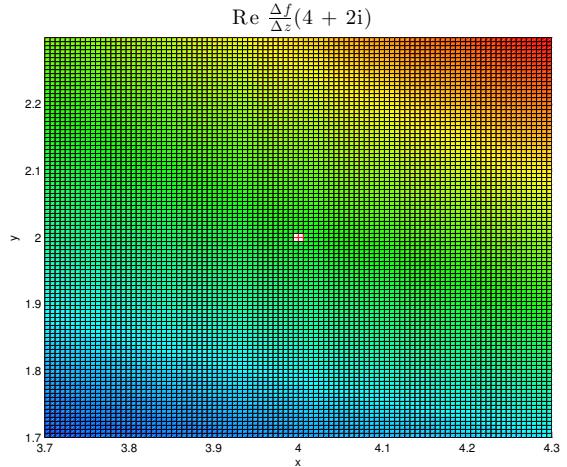
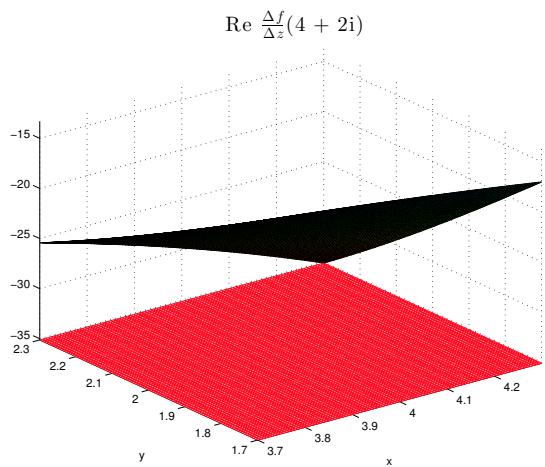
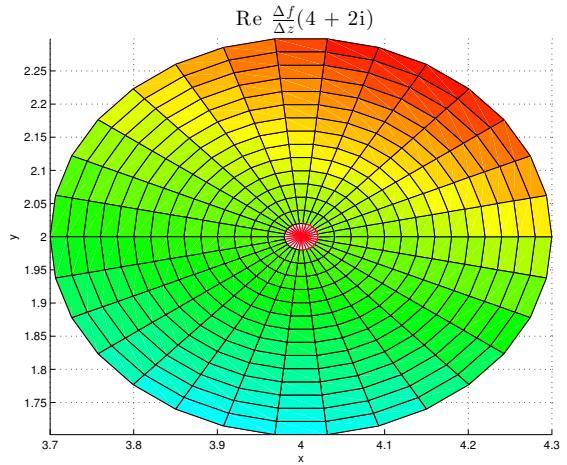
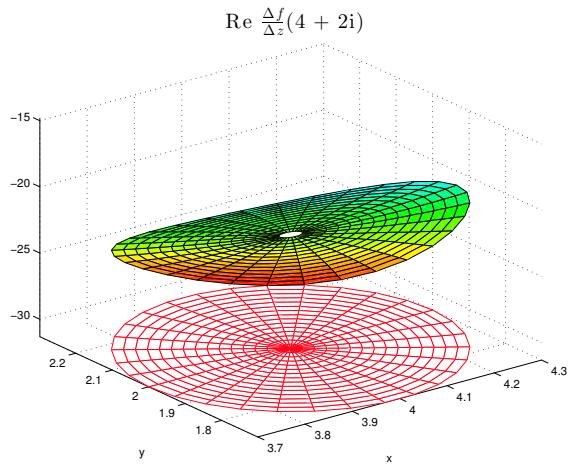
ESECUZIONE DEL PROGRAMMA(SCELTA=2,f(z) = $imag(z)$, punti $[(4 + 2i), (2 + 3i)]$)

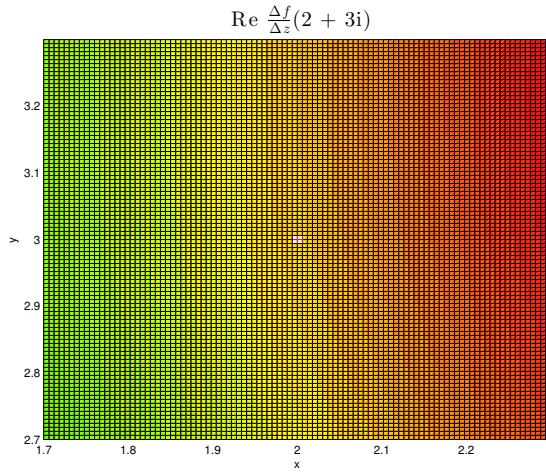
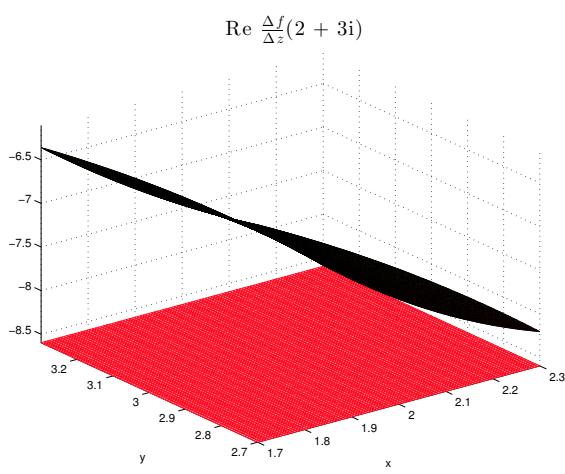




Anche in questo caso per entrambi i punti non esiste il limite del rapporto incrementale. La funzione $\text{imag}(z)$ non è olomorfa in tali punti.

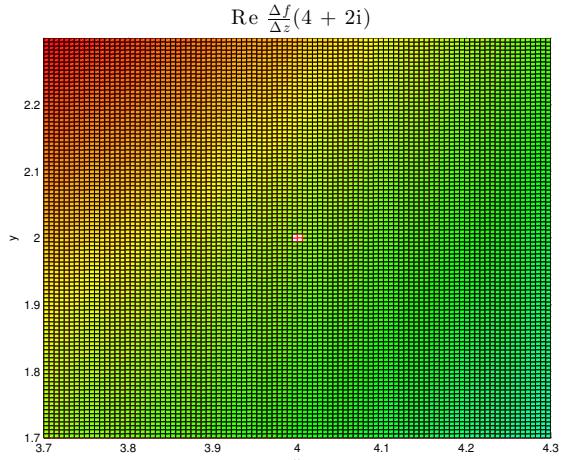
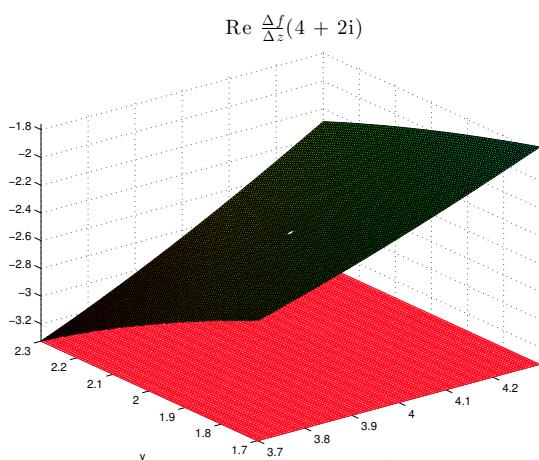
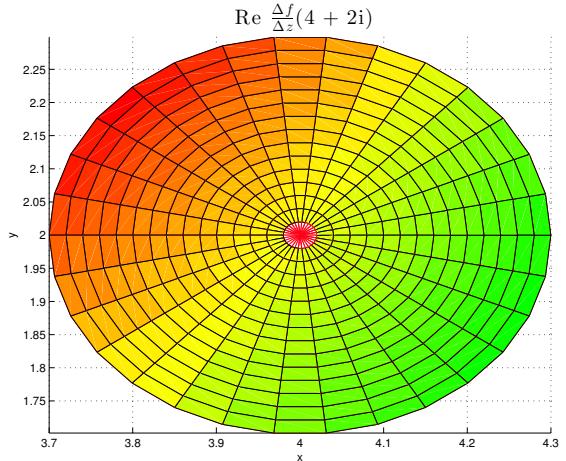
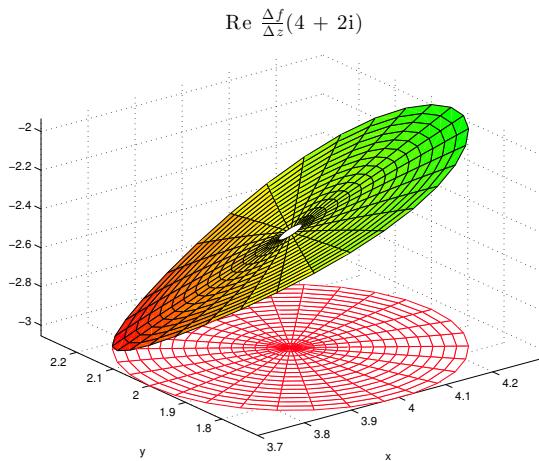
ESECUZIONE DEL PROGRAMMA(SCELTA=3, $f(z) = e^z$, punti $[(4 + 2i), (2 + 3i)]$)

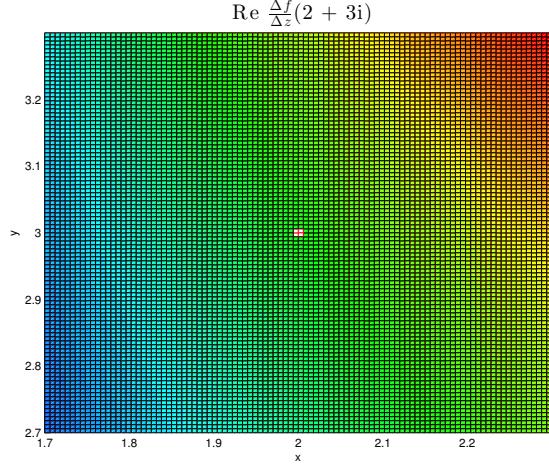
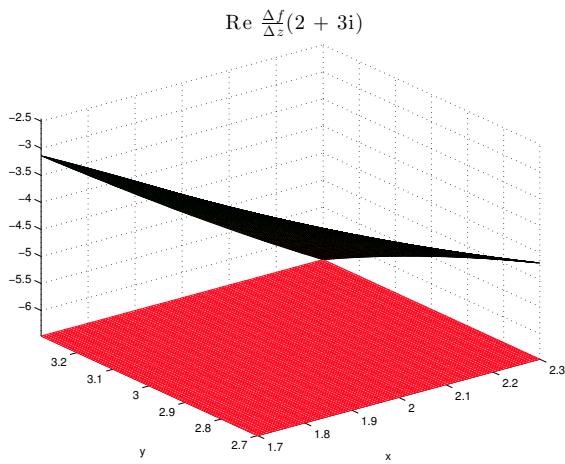
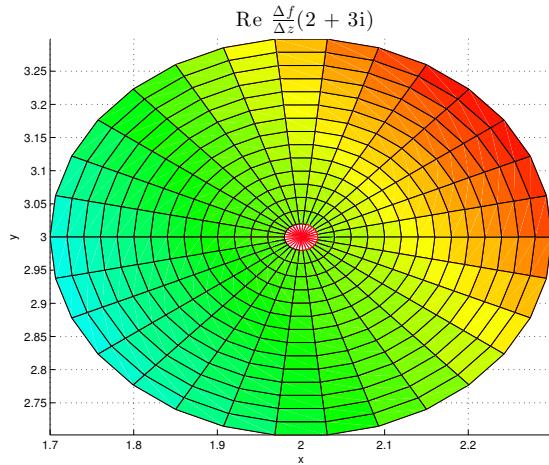
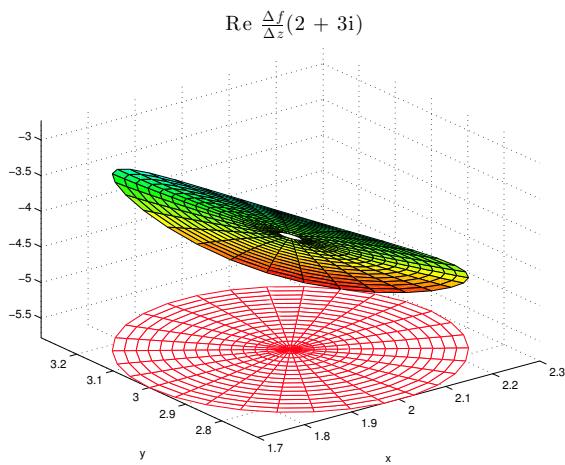




Per entrambi i punti esiste il limite del rapporto incrementale. La funzione è olomorfa in tali punti.

ESECUZIONE DEL PROGRAMMA (SCELTA=4, $f(z) = \sin(z)$, punti[$(4 + 2i), (2 + 3i)$])





Per entrambi i punti esiste il limite del rapporto incrementale. La funzione è olomorfa in tali punti.

2.8 Verificare e visualizzare la derivabilità in senso complesso di alcune funzioni mediante Symbolic Math Toolbox.[liv.1]

```

1 syms x1 y1 theta real
2 syms rho positive
3 if ~exist('scelta','var')
4     scelta=input('Inserire scelta:');
5 end
6 switch scelta
7 case 1
8     f=inline('1./(z.^2-1)');
9     punti=[4+2i,2+3i];
10 case 2
11     f=inline('imag(z)');
12     punti=[4+2i,2+3i];
13 case 3
14     f=inline('abs(z)');
15     punti=[4+2i,2+3i];
16 case 4
17     f=inline('sin(z)');

```

```

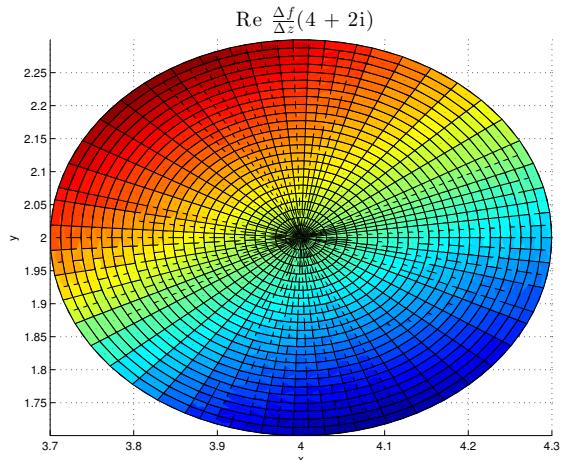
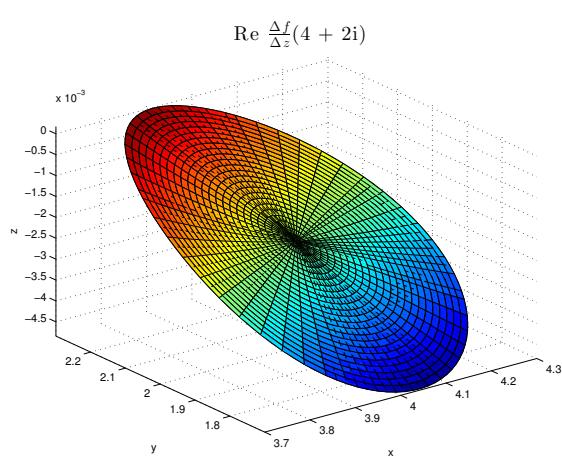
18     punti=[4+2i,2+3i];
19 otherwise
20     disp('Inserimento errato');
21     exit;
22 end
23 for n=[1,2]
24     z0=punti(n);
25     r=0.3;
26     dz=rho*exp(i*theta);
27     %intorno centrale intorno a z0 di raggio r
28     Ri=simplify((f(z0+dz)-f(z0))/dz);
29     %Rapporto incrementale
30     figure
31     ezsurf(real(dz)+real(z0),imag(dz)+imag(z0),real(Ri),[-r r -pi,pi])
32     str=['Re  $\frac{\Delta f}{\Delta z}$ ',...
33         sprintf(' (%d + %di)',real(z0),imag(z0))];
34     title(str,'interpreter','latex','fontsize',18);
35     axis tight;xlabel('x');ylabel('y');
36     figure(copyobj(gcf,0));
37     view(2);
38     %calcolo il limite per rho che tende a zero del rapporto incrementale
39     z1=x1+i*y1;
40     dz=rho*exp(i*theta);
41     Ri=simplify((f(z1+dz)-f(z1))/dz);
42     Ri=simplify(limit(Ri,rho,0));
43     Ri=subs(Ri,{x1,y1},{real(z0),imag(z0)} );
44     fprintf('Limite del rapporto incrementale per rho->0,punto(%d+%di)\n',...
45             ,real(z0),imag(z0));
46     disp(Ri);
47 end

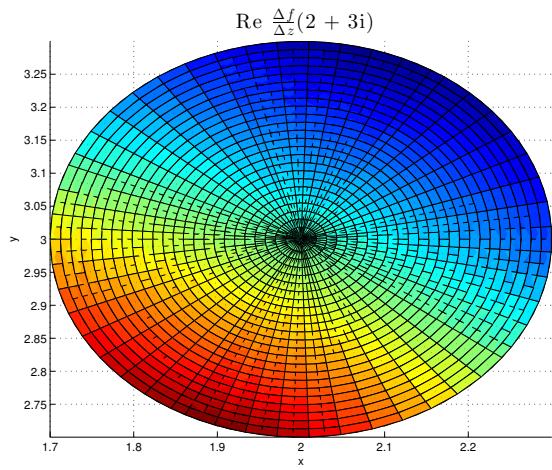
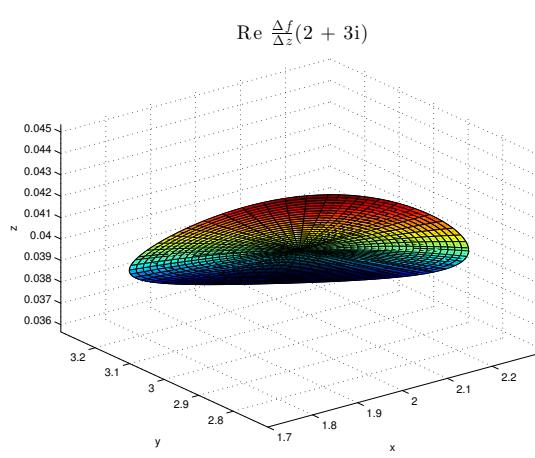
```

ESECUZIONE DEL PROGRAMMA (SCELTA=1, $f(z) = \frac{1}{z^2-1}$, punti[(4 + 2i), (2 + 3i)])

Limite del rapporto incrementale per rho->0, punto (4+2i)
 $-0.0023 + 0.0236i$

Limite del rapporto incrementale per rho->0, punto (2+3i)
 $0.0400 + 0.0022i$



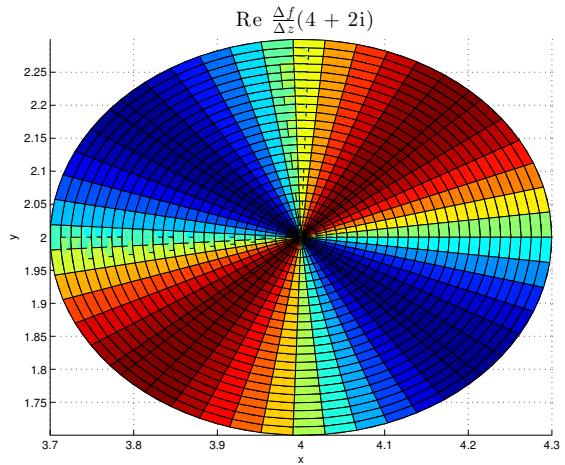
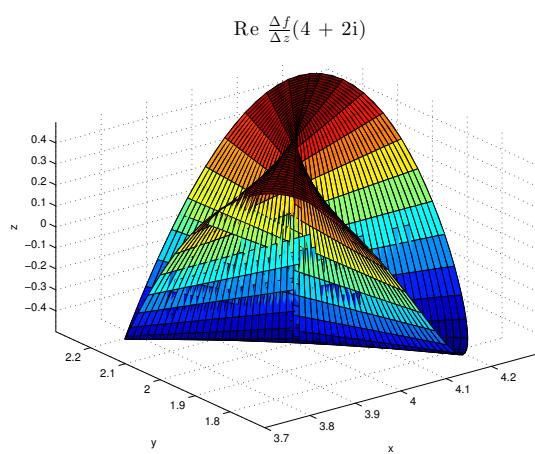


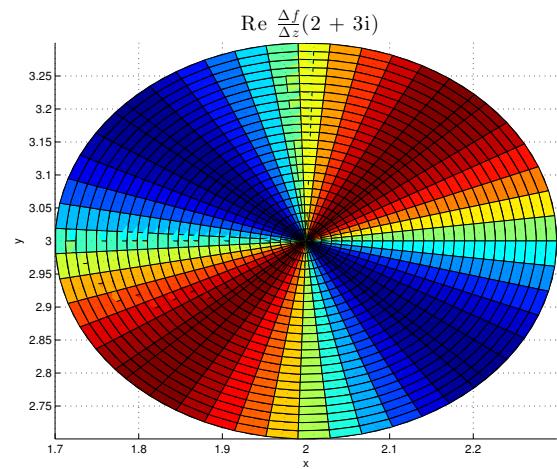
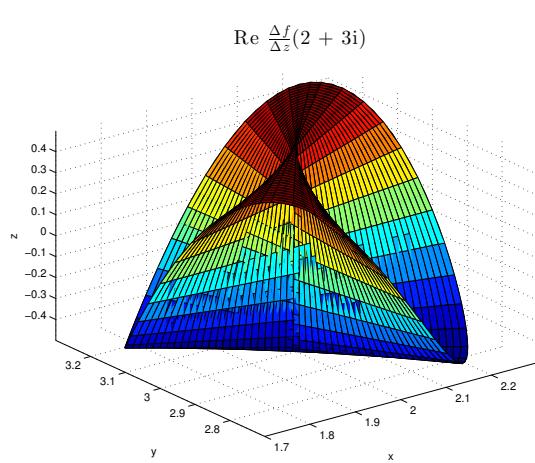
Il limite del rapporto incrementale esiste e non dipende da theta. La funzione é olomorfa in entrambi i punti.

ESECUZIONE DEL PROGRAMMA(SCELTA=2,f(z) = $imag(z)$, punti[(4 + 2i), (2 + 3i)])

Limite del rapporto incrementale per rho->0, punto (4+2i)
 $\sin(\theta * y1) / y1 - \sin(\theta)^y1 * i$

Limite del rapporto incrementale per rho->0, punto (2+3i)
 $\sin(\theta * x1) / x1 - \sin(\theta)^x1 * i$



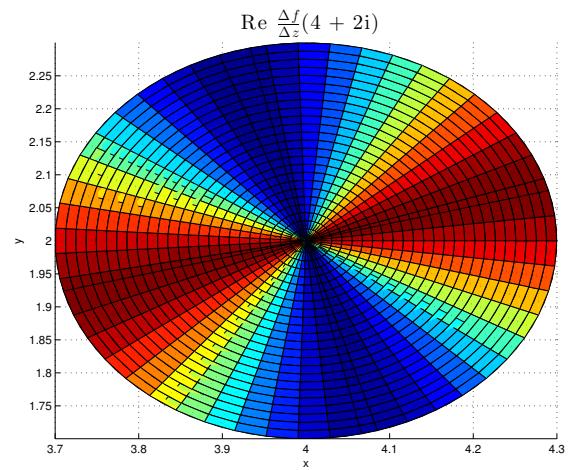
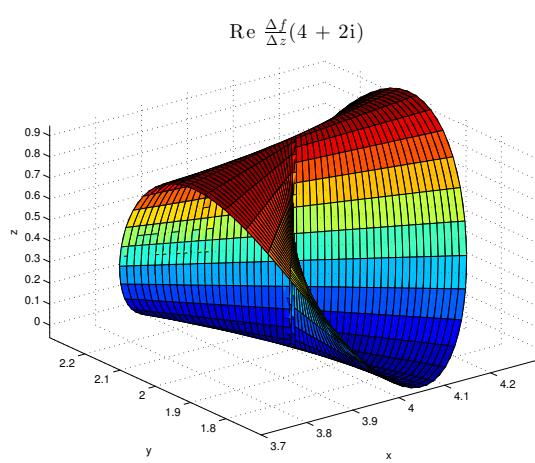


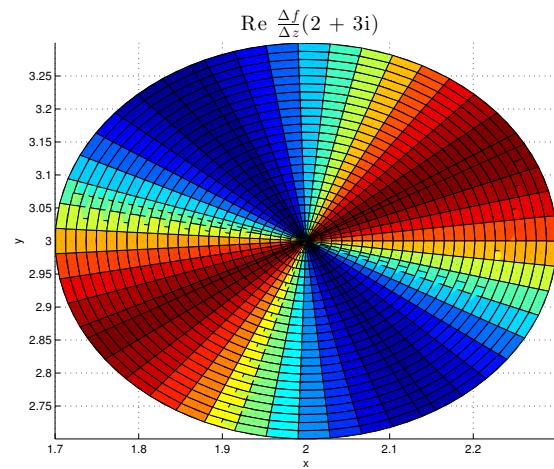
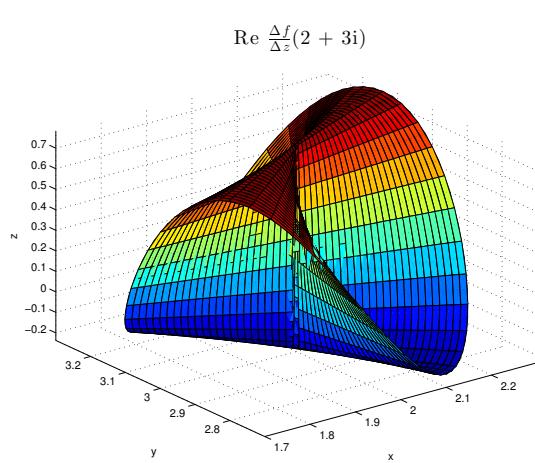
In questo caso non esiste il limite in senso complesso, in quanto dipende da ϑ . La funzione $\operatorname{imag}(z)$ non è olomorfa nei punti studiati.

ESECUZIONE DEL PROGRAMMA(SCELTA=3, $f(z) = |z|$, punti $[(4 + 2i), (2 + 3i)]$)

Limite del rapporto incrementale per rho->0, punto(4+2i)
 $(5^{(1/2)} * \exp(-\theta * 2i) * (\cos(\theta) * \sin(\theta) * (64 - 32i) + \cos(\theta)^2 * (64 - 32i) + \sin(\theta)^2 * (16 - 8i))) / 200$

Limite del rapporto incrementale per rho->0, punto(2+3i)
 $(13^{(1/2)} * \exp(-\theta * 2i) * (\cos(\theta) * \sin(\theta) * (24 - 36i) + \cos(\theta)^2 * (8 - 12i) + \sin(\theta)^2 * (18 - 27i))) / 169$



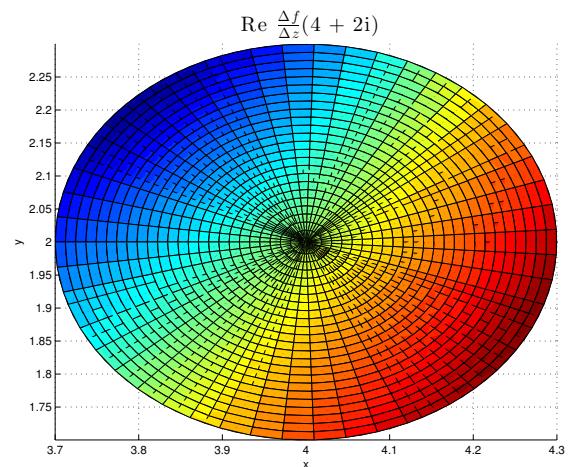
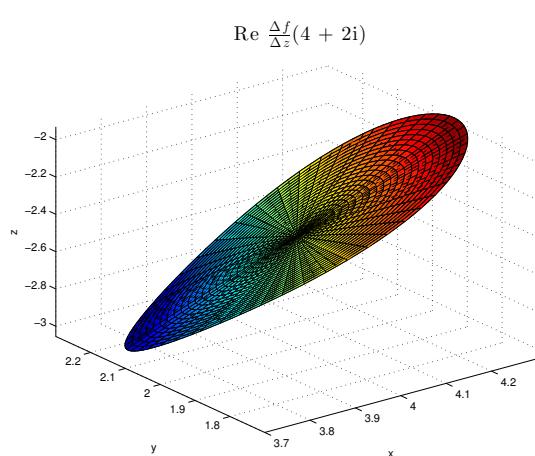


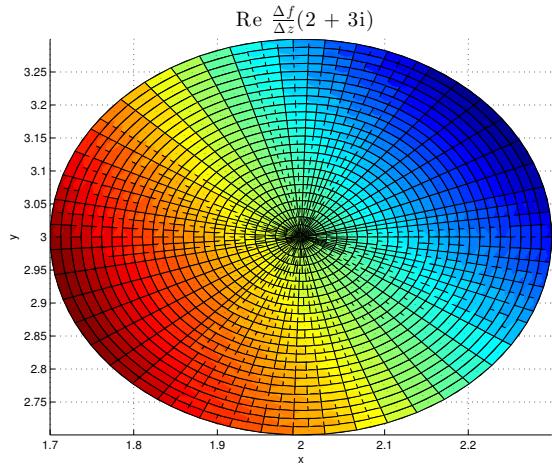
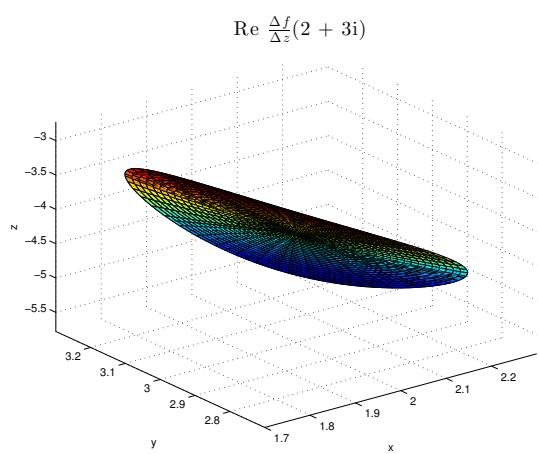
Neanche in questo caso esiste il limite del rapporto incrementale, a causa della dipendenza dello stesso da ϑ . La funzione non è olomorfa nei punti studiati.

ESECUZIONE DEL PROGRAMMA(SCELTA=4, $f(z) = \sin(z)$, punti[$(4 + 2i), (2 + 3i)$])

Limite del rapporto incrementale per rho->0, punto $(4+2i)$
 $-2.4591 + 2.7448i$

Limite del rapporto incrementale per rho->0, punto $(2+3i)$
 $-4.1896 - 9.1092i$





In questo ultimo caso esiste il limite del rapporto incrementale. La funzione è olomorfa nei 2 punti.

2.9 Come si calcola mediante il Symbolic Math Toolbox di MATLAB il seguente limite (in senso complesso)? [liv.1]

$$f'(z_0) = \lim_{z \rightarrow z_0} \frac{f(z) - f(z_0)}{z - z_0} = \lim_{z \rightarrow z_0} \frac{f(z + \Delta z) - f(z_0)}{\Delta z}$$

```

1 syms x0 y0 theta real; syms rho positive
2 %definisco un generico numero complesso
3 z0=x0+i*y0;
4 %e una funzione complessa
5 f=inline('z^3');
6 %intorno circolare
7 dz=rho*exp(i*theta);
8 %rapporto incrementale
9 ri=simplify((f(z0+dz)-f(z0))/dz);
10 %limite del rapporto incrementale per rho->0
11 disp('Limite del rapporto incrementale per rho->0');
12 l=limit(ri,rho,0);
13 disp(l);

```

ESECUZIONE DEL PROGRAMMA

Limite del rapporto incrementale per rho->0
 $3*(x_0 + y_0*i)^2$

2.10 Verificare e visualizzare la validitá delle Eq. di Cauchy-Riemann in senso complesso di alcune funzioni in un intorno di alcuni punti z_0 mediante Symbolic Math Toolbox.[liv.1]

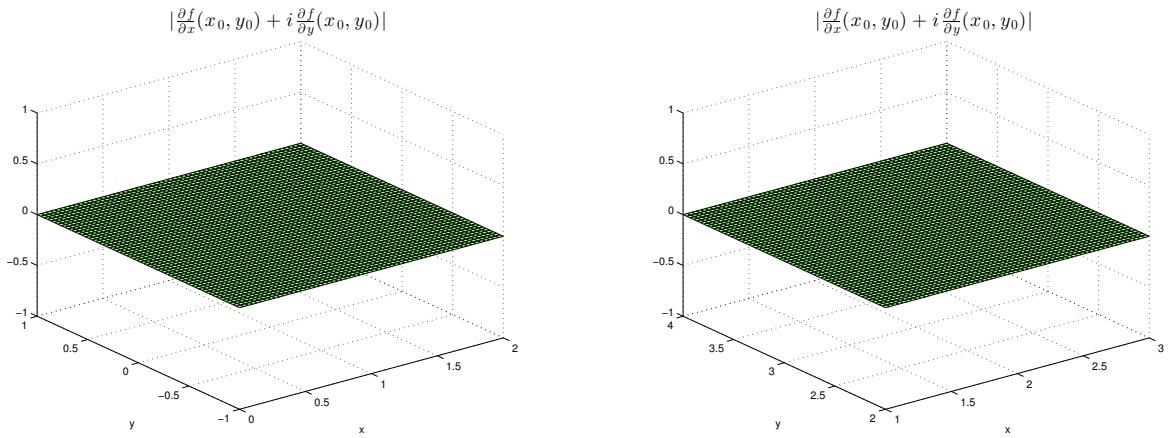
```

1 syms x y real
2 z=x+i*y;
3 if ~exist('scelta','var')
4     scelta=input('Inserire scelta:');
5 end
6 switch scelta
7     case 1
8         f=1/(z^2-1);
9         punti=[1,2+3i];
10    case 2
11        f=imag(z);
12        punti=[4+2i,2+3i];
13    case 3
14        f=conj(z);
15        punti=[0,2+3i];
16    case 4
17        f=sin(z);
18        punti=[4+2i,2+3i];
19    otherwise
20        disp('Inserimento errato');
21        exit;
22 end
23 for n=[1,2]
24     z0=punti(n);
25     d=diff(f,x)+i*diff(f,y);
26     %Equazione di Cauchy-Riemann in forma complessa
27     dz0=subs(d,{x y},{real(z0) imag(z0)} );
28     fprintf('Equazione valutata nel punto z0=%f+%fi:%d\n',...
29             ,real(z0),imag(z0),abs(dz0));
30 figure
31     %Visualizzo il grafico del valore assoluto di d.
32     %Nei punti del grafico tali che f(x,y)==0 la funzione e' olomorfa
33     ezsurf(abs(d),[real(z0)-1 real(z0)+1 imag(z0)-1 imag(z0)+1]);
34     title('$|\frac{\partial f}{\partial x}({x}_0,{y}_0)|$',...
35             , 'interpreter','latex','fontsize',18);
36     axis tight
37     if abs(dz0)==0
38         fprintf('Funzione olomorfa in z=%f+%fi\n',real(z0),imag(z0));
39     else
40         fprintf('Funzione non olomorfa in z=%f+%fi\n',real(z0),imag(z0));
41     end
42 end
43

```

ESECUZIONE DEL PROGRAMMA(SCELTA=1, $f(z) = \frac{1}{z^2-1}$, punti[(1), (2 + 3i)])

Equazione valutata nel punto z0=1.000000+0.000000i:NaN
 Funzione non olomorfa in z=1.000000+0.000000i
 Equazione valutata nel punto z0=2.000000+3.000000i:0
 Funzione olomorfa in z=2.000000+3.000000i



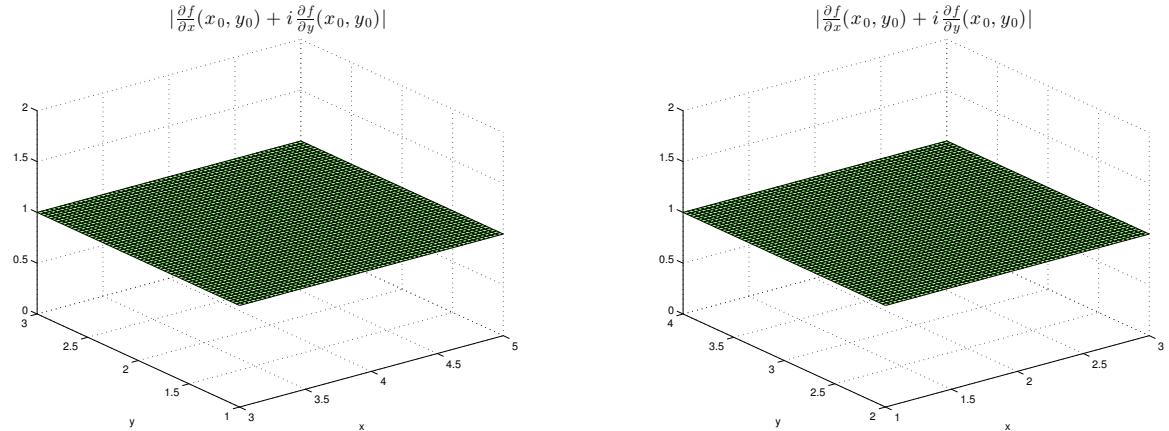
ESECUZIONE DEL PROGRAMMA(SCELTA=2, $f(z) = \text{imag}(z)$, punti[$(4 + 2i), (2 + 3i)$])

Equazione valutata nel punto $z0=4.000000+2.000000i:1$

Funzione non olomorfa in $z=4.000000+2.000000i$

Equazione valutata nel punto $z0=2.000000+3.000000i:1$

Funzione non olomorfa in $z=2.000000+3.000000i$



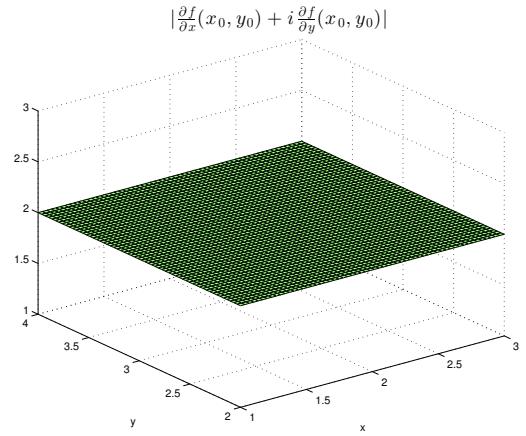
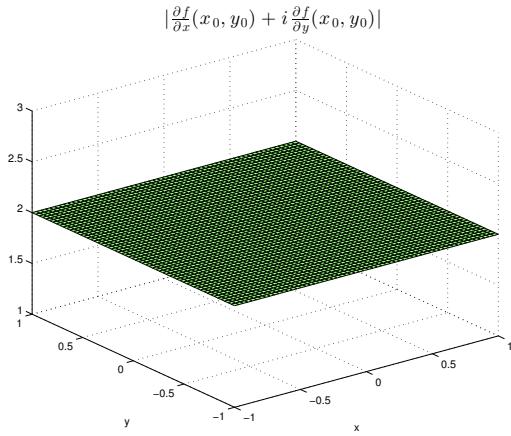
ESECUZIONE DEL PROGRAMMA(SCELTA=3, $f(z) = \text{conj}(z)$, punti[$(0), (2 + 3i)$])

Equazione valutata nel punto $z0=0.000000+0.000000i:2$

Funzione non olomorfa in $z=0.000000+0.000000i$

Equazione valutata nel punto $z0=2.000000+3.000000i:2$

Funzione non olomorfa in $z=2.000000+3.000000i$



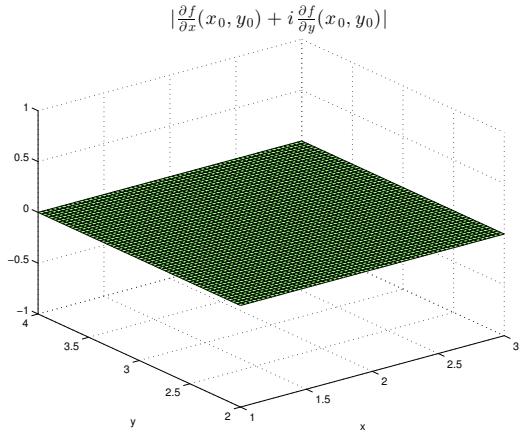
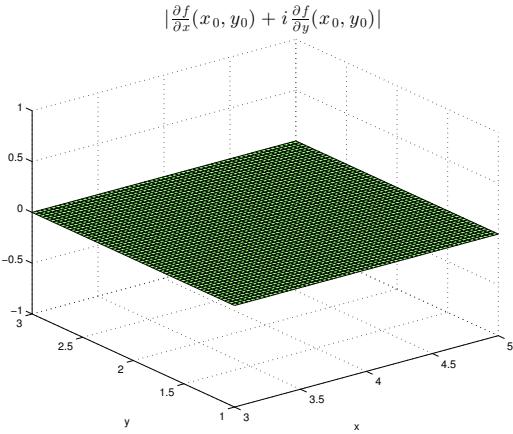
ESECUZIONE DEL PROGRAMMA (SCELTA=4, $f(z) = \sin(z)$, punti[$(4 + 2i), (2 + 3i)$])

Equazione valutata nel punto $z0=4.000000+2.000000i:0$

Funzione olomorfa in $z=4.000000+2.000000i$

Equazione valutata nel punto $z0=2.000000+3.000000i:0$

Funzione olomorfa in $z=2.000000+3.000000i$



3 Successioni e serie

3.1 Mediante function MATLAB approssimare numericamente a p cifre decimali corrette la somma delle serie:

$$1 + \frac{1}{2!} + \frac{1}{3!} + \dots$$
$$1 + \frac{1}{2^2} + \frac{1}{3^2} + \dots$$
$$1 + \frac{1}{2^4} + \frac{1}{3^4} + \dots$$

[liv.2]

```
1 syms k positive;
2 funzioni={'1/k!', '1/k^2', '1/k^4'};
3 if ~exist('scelta','var')
4     scelta=menu('Scelta funzione',funzioni);
5     if ~exist('p', 'var')
6         p=input('Inserire il valore di p:');
7     end
8     switch scelta
9         case 1
10            f=@(x) 1./gamma(x+1);
11            ak=1/gamma(k+1);%uso gamma per approssimare il fattoriale
12        case 2
13            f=@(x) 1./x.^2;
14            ak=1/k.^2;
15        case 3
16            f=@(x) 1./x.^4;
17            ak=1/k.^4;
18        otherwise
19            disp('Scelta Errata');
20        end
21 %Applico il teorema di confronto con un integrale
22 i=1;
23 while quadgk(f,i,inf)>=p
24     i=i+1;
25 end
26 fprintf('Scelta:%d,p=%e\nOrdine ridotta=%d\n',scelta,p,i);
27 n=1:i;
28 an=f(n);
29 Sn=(cumsum(an));
30 ris=Sn(end);
31 fprintf('Ridotta=%d\n',ris);
32 Sn=symsum(ak,k,1,inf);
33 fprintf('Il resto :%d\n',subs(Sn-ris));
34 end
```

ESECUZIONE DEL PROGRAMMA(SCELTA=1,p=1e-3)

```
Scelta:1,p=1.000000e-03
Ordine ridotta=6
Ridotta=1.718056e+00
Il resto e':2.262729e-04
```

ESECUZIONE DEL PROGRAMMA(SCELTA=2,p=1e-4)

```
Scelta:2,p=1.000000e-04
Ordine ridotta=10000
Ridotta=1.644834e+00
Il resto e':9.999500e-05
```

ESECUZIONE DEL PROGRAMMA(SCELTA=3,p=1e-2)

```
Scelta:3,p=1.000000e-02
Ordine ridotta=4
Ridotta=1.078752e+00
Il resto :3.571305e-03
```

3.2 Mediante Symbolic Math Toolbox di MATLAB studiare la serie

$$1 - x^2 + x^4 - x^6 + \dots$$

[liv.2]

```
1 syms n x y k positive;
2 ak=y^k;
3 S=symsum(ak,k,0,inf);
4 disp(' Limite serie:');
5 S=subs(S,y,-x^2);
%Somma della serie
6 pretty(S);
7 Sn=symsum(ak,k,0,n);
8 Sn=subs(Sn,y,-x^2);
9 %Ridotta n-sima della serie
10 disp(sprintf('\n Ridotta n-sima'));
11 pretty(Sn);
12 hold on;grid on;axis tight;
13 h(1)=ezplot(S,[-1 1]);
14 set(h(1),'Color','k');
15 set(h(1),'LineWidth',2);
16 set(h(1),'DisplayName','Limite');
17 cmap = hsv(6);
18 for i=1:6
19 c=Sn;
20 c=subs(c,n,i);
21 h(i+1)=ezplot(c,[-1 1]);
%Visualizzo la ridotta di ordine i
22 set(h(i+1),'Color',cmap(i,:));
23 str=sprintf('n=%d',i);
24 set(h(i+1),'DisplayName',str);
25 end
26 legend(h);
27 title('$1 - {x}^{2} + {x}^{4} - {x}^{6} \dots $',...
28 'interpreter','latex','fontsize',18);
29 axis([-1 1 0 1]);
30 disp('lim n=>+inf Sn(x)');
%Vedo dove converge la successione delle somme parziali
```

```

34 pretty(simplify(limit(Sn,n,inf)));
35 ERR=abs(S-Sn);
36 disp('lim n=>+inf |F(x)-Fn(x)|');
37 %Verifico se la convergenza e' uniforme
38 pretty(simplify(limit(ERR,n,inf)));

```

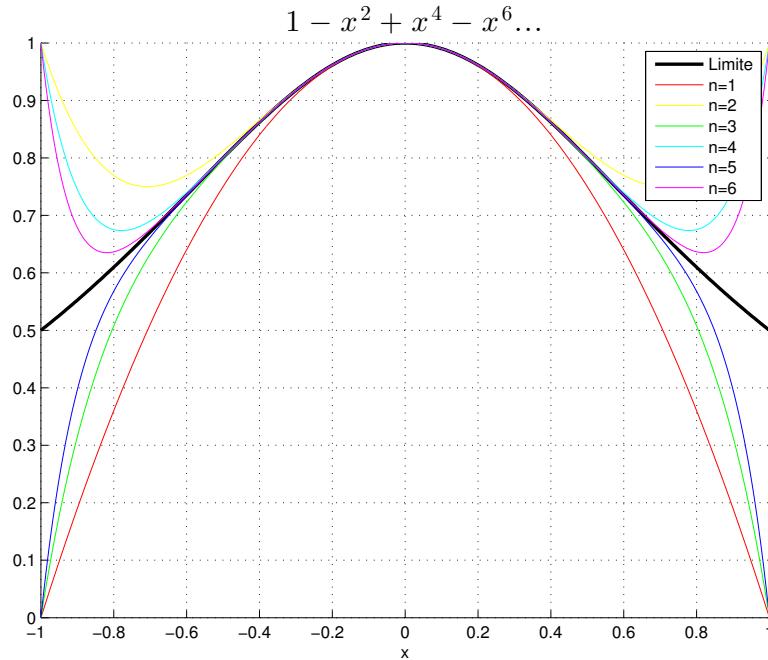
ESECUZIONE DEL PROGRAMMA

Limite serie:

$$\frac{1}{x^2 + 1}$$

Ridotta n-sima

$$\begin{aligned} & \frac{2}{x^2} \frac{(-x)^n}{n^2 + 1} \\ & \text{lim } n \Rightarrow +\infty \quad S_n(x) \\ & \text{piecewise} \left| \begin{array}{c} / \quad 1 \quad \quad \quad 2 \quad \quad \quad \backslash \\ \hline \text{if } x^2 < 1 \\ | \quad 2 \\ \backslash x^2 + 1 \quad \quad \quad / \end{array} \right. \\ & \text{lim } n \Rightarrow +\infty \quad |F(x) - F_n(x)| \\ & \text{piecewise}(1/2 \text{ if } x == 1, \text{ Inf if } 1 < x, 0 \text{ if } x^2 < 1) \end{aligned}$$



La serie da studiare è una serie geometrica di termine $-x^2$. La successione delle somme parziali converge per $-1 < x < 1$ e il limite dell'errore tende a 0 per n che tende a inf. La serie converge uniformemente nell'intervallo $]-1, 1[$ è indeterminata per $x=1$ e $x=-1$ e diverge per tutti gli altri valori di x.

3.3 Confrontare, evidenziandone le differenze, la bontà delle due stime calcolabili dell'errore nella serie esponenziale:

$$|e^x - S(n, x)| \approx \frac{|x|^n}{n!} \frac{n+1}{n+1-x} \quad e \quad \frac{|e^x - S(n, x)|}{e^x} \approx \frac{|x|^n}{n!}$$

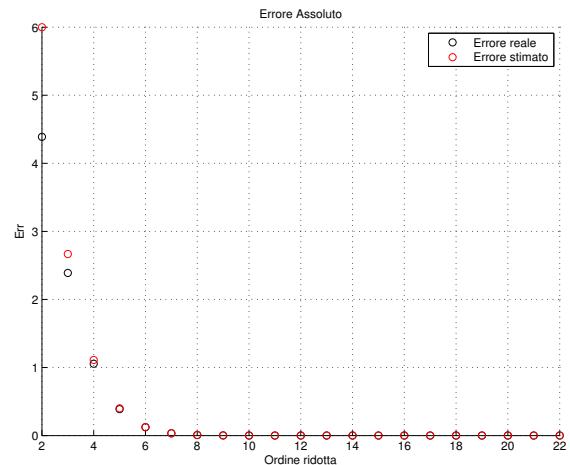
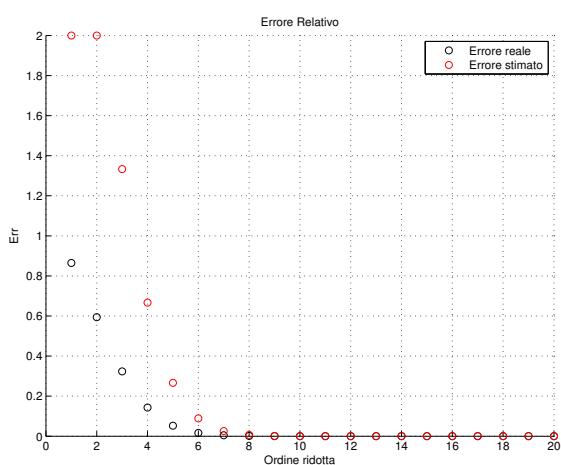
[liv.2]

```

1 syms x;
2 x0=2;
3 hold on;title('Errore Relativo');grid on;
4 xlabel('Ordine ridotta');ylabel('Err');
5 for i=1:20
6 T=taylor(exp(x),'Order',i);%ridotta di exp(x) di ordine i
7 Sn=subs(T,x,x0);%valuto la ridotta in x0
8 Err=abs(exp(x0)-Sn)/exp(x0);%calcolo l'errore relativo
9 Err_stim=abs(x0^i)/factorial(i);%e quello stimato
10 plot(i,Err,'ok',i,Err_stim,'or');%infine li visualizzo entrambi
11 end
12 legend('Errore reale','Errore stimato');
13 figure;
14 title('Errore Assoluto');grid on;
15 xlabel('Ordine ridotta');ylabel('Err');
16 a=abs(x0);
17 %La stima dell'errore assoluto e' valida per x<n+1
18 for i=a:(a+20)
19 T=taylor(exp(x),'Order',i);
20 Sn=subs(T,x,x0);
21 Err=abs(exp(x0)-Sn);
22 Err_stim=(abs(x0^i)/factorial(i))*((i+1)/(i+1-x0));
23 hold on;
24 plot(i,Err,'ok',i,Err_stim,'or');
25 end
26 legend('Errore reale','Errore stimato');

```

ESECUZIONE DEL PROGRAMMA($x_0 = 2$)



3.4 Visualizzare il campo di convergenza delle seguenti serie di potenze nel campo complesso:geometrica,armonica,esponenziale,ed ancora le serie:

$$\sum_{n=0}^{\infty} \frac{z^n}{2^{2n}}, \quad \sum_{n=0}^{\infty} \frac{(z+1)^n}{n2^n}, \quad \sum_{n=0}^{\infty} \frac{(z-2)^n}{n^2 2^{2n}}$$

$$\sum_{n=0}^{\infty} (1+ni) z^n, \quad \sum_{n=0}^{\infty} (\log n)^2 z^n, \quad \sum_{n=0}^{\infty} \frac{5^n z^{3n}}{2n(2n+2)}$$

Giustificare i risultati ottenuti.[liv.1]

```

1 n=200;
2 funzioni={'z^n','z^n/n','z^n/n!', 'z^n/(2^(2n))','(z+1)^n/(n*2^n)',...
3 '(z-2)^n/(n^2*2^(2n))','(1+ni)z^n','(logn)^2*z^n',...
4 '(5^n*z^(3n))/(2n(2n+2))'};
5 scelta=menu('Scelta funzione',funzioni);
6 switch scelta
7 case 1%z^n
8     p=1;%raggio di convergenza
9     z0=0;%punto iniziale(centro della serie)
10    k=1;
11    str='Ridotta 200-sima di $\sum_{n=0}^{\infty}\{z^n\}$';
12    an=ones(n,1);%coefficiente n-simo della serie
13 case 2%z^n/n
14     p=1;%raggio di convergenza
15     z0=0;
16     k=1;
17     str='Ridotta 200-sima di $\sum_{n=1}^{\infty}\{\frac{z^n}{n}\}$';
18     an=fliplr(1./(1:n));%coefficiente n-simo della serie
19     an=[an,0];
20     %aggiungo lo 0 in quanto nella polyval il termine di grado 0 non
21     %deve esserci(la somma va da 1 a n)
22 case 3%z^n/n!
23     p=2;
24     %raggio di convergenza infinito,scelgo un numero per l'intervalllo
25     %di visualizzazione
26     z0=0;
27     k=1;
28     str='Ridotta 200-sima di $\sum_{n=0}^{\infty}\{\frac{z^n}{n!}\}$';
29     an=fliplr(1./gamma(2:n+1));%coefficiente n-simo della serie
30 case 4%z^n/(2^(2n))
31     p=4;%raggio di convergenza
32     z0=0;
33     k=1;
34     str='Ridotta 200-sima di $\sum_{n=0}^{\infty}\{\frac{z^n}{2^{2n}}\}$';
35     an=fliplr(1./4.^((0:n-1)));%coefficiente n-simo della serie
36 case 5%(z+1)^n/(n*2^n)
37     p=2;%raggio di convergenza
38     z0=-1;
39     k=1;
40     str='Ridotta 200-sima di $\sum_{n=1}^{\infty}\{\frac{(z+1)^n}{n2^n}\}$';
41     an=fliplr(1./((1:n).*2.^((1:n))));
42     %coefficiente n-simo della serie
43     an=[an,0];
44

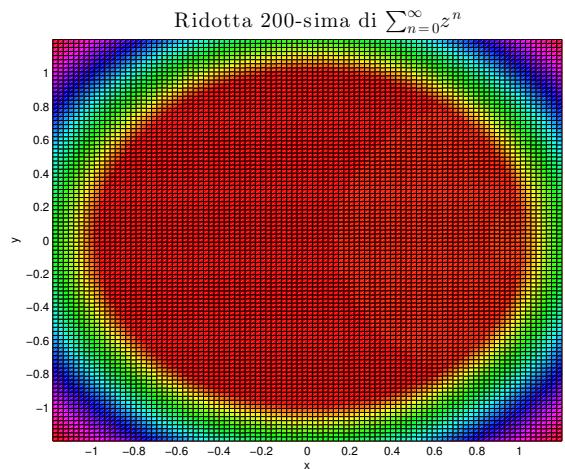
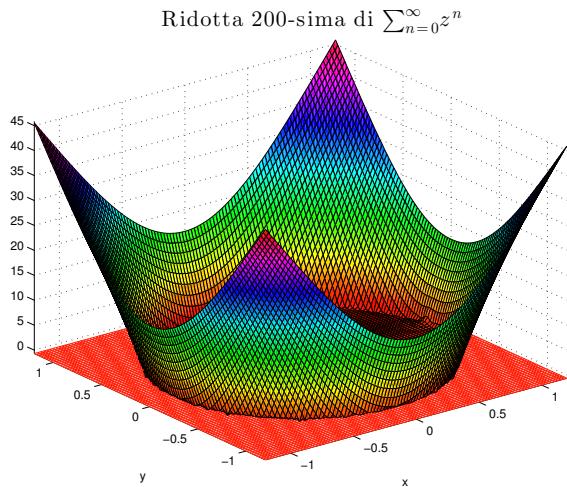
```

```

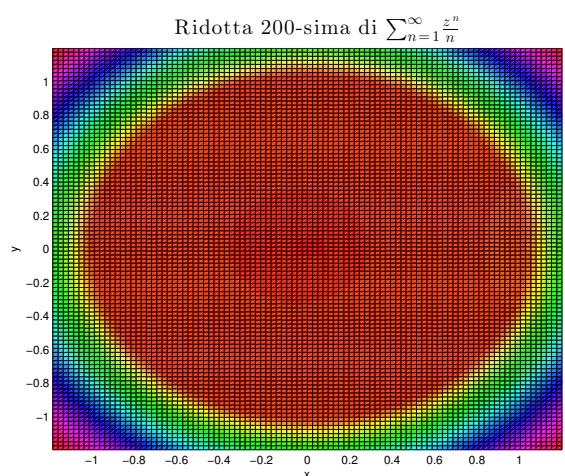
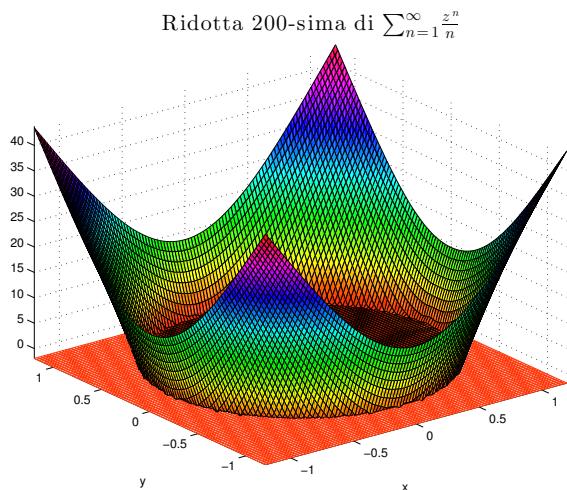
46 case 6%(z-2)^n/(n^2*2^(2n)
47     p=4;%raggio di convergenza
48     z0=2;
49     k=1;
50     str='Ridotta 200-sima di $\sum_{n=1}^{\infty}
51     {\frac{(z-2)^n}{n^{22^{(2n)}}}}$';
52     an=fliplr(1./(((1:n).^2).*((2.^((1:n)*2))));%
53     %coefficiente n-simo della serie
54     an=[an,0];
55 case 7%(1+ni)z^n
56     p=1;%raggio di convergenza
57     z0=0;
58     k=1;
59     str='Ridotta 200-sima di $\sum_{n=0}^{\infty}
60     {\left(1+ni\right)^n}z^n$';
61     an=fliplr(1+(0:n-1).*i);%coefficiente n-simo della serie
62 case 8%(logn)^2*z^n
63     p=1;%raggio di convergenza
64     z0=0;
65     k=1;
66     str='Ridotta 200-sima di $\sum_{n=1}^{\infty}
67     {\left(\log{n}\right)^2z^n}$';
68     an=fliplr(log(1:n).^2);%coefficiente n-simo della serie
69     an=[an,0];
70 case 9%(5^n*z^(3n))/(2n(2n+2))
71     p=(1/5)^(1/3);%raggio di convergenza
72     z0=0;
73     k=3;
74     str='Ridotta 200-sima di $\sum_{n=1}^{\infty}
75     {\frac{5^n z^{3n}}{2n \left(2n+2\right)}}$';
76     an=fliplr((5.^(1:n))./(4.*((1:n).^2+4.*((1:n))));%
77     %coefficiente n-simo della serie
78     an=[an,0];
79 otherwise
80     disp('Inserimento errato');
81     exit;
82 end
83 [x,y]=meshgrid(linspace(real(z0)-p-0.2,real(z0)+p+0.2,99),...
84     linspace(imag(z0)-p-0.2,imag(z0)+p+0.2),99);
85 z=x+i*y;
86 Sn=polyval(an,(z-z0).^k);
87 figure
88 cplxmap(z,log10(abs(Sn))*(1+i));
89 title(str,'interpreter','latex','fontsize',18);
90 xlabel('x');ylabel('y');axis tight;
91 figure(copyobj(gcf,0));
92 view(2);

```

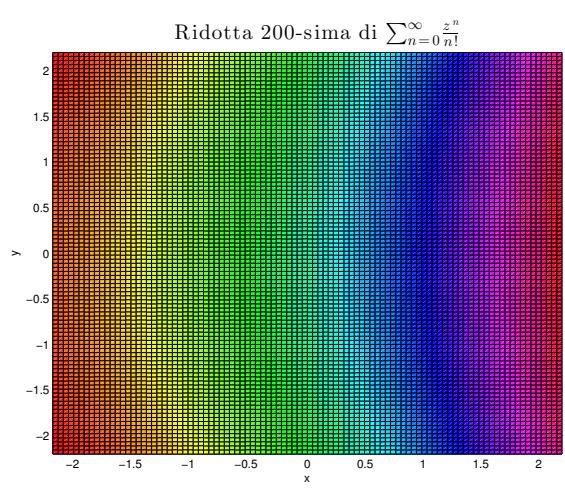
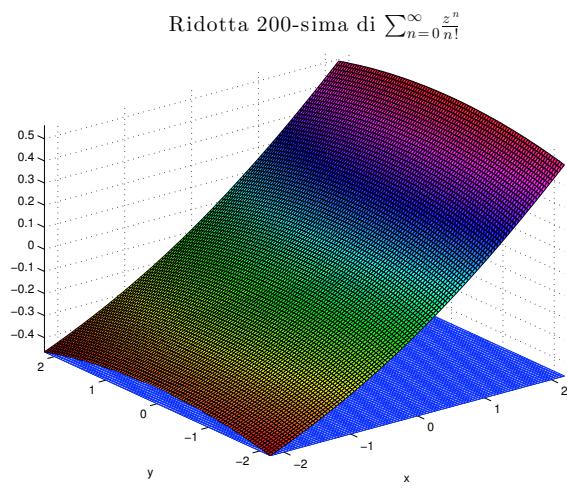
ESECUZIONE DEL PROGRAMMA(SCELTA=1)



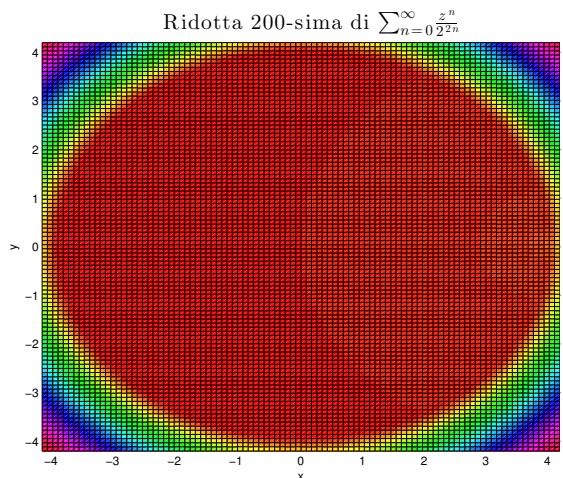
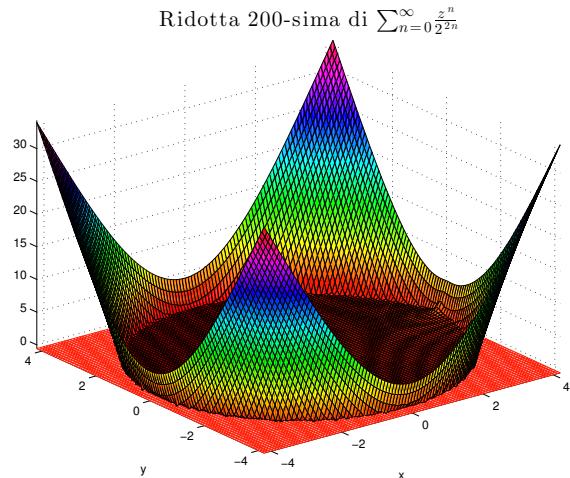
ESECUZIONE DEL PROGRAMMA(SCELTA=2)



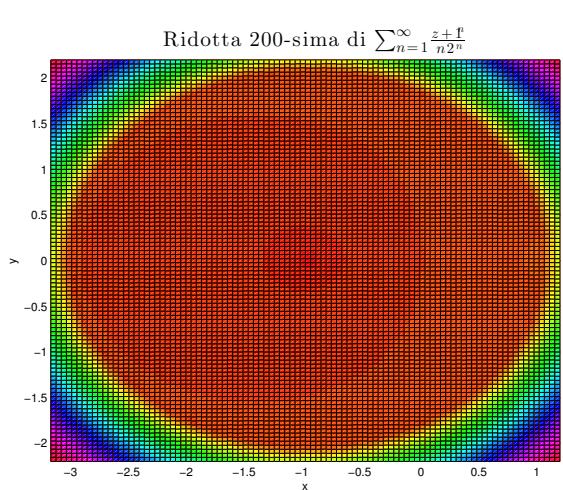
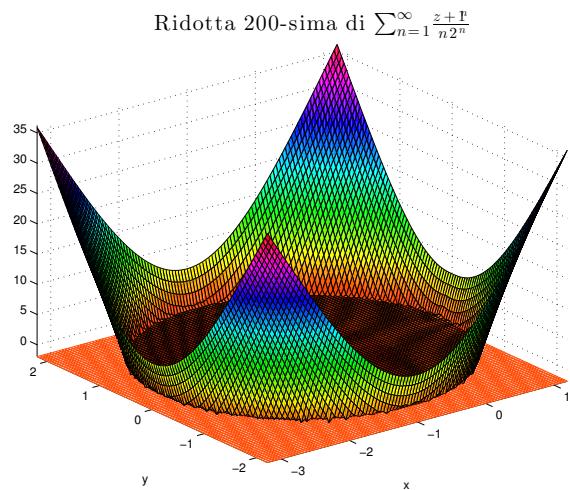
ESECUZIONE DEL PROGRAMMA(SCELTA=3)



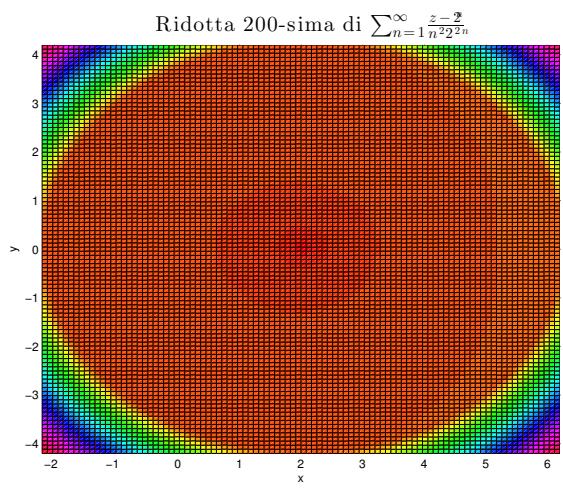
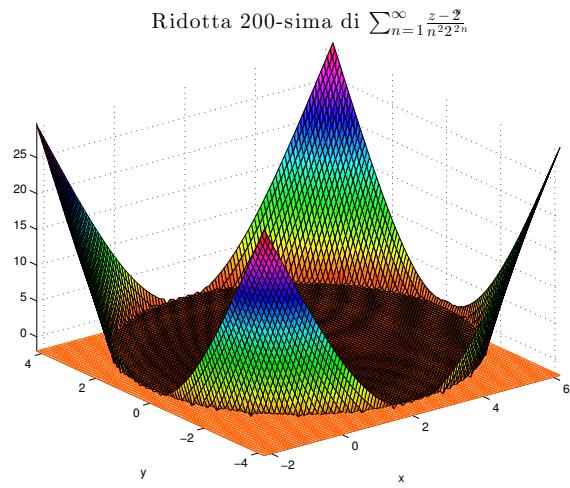
ESECUZIONE DEL PROGRAMMA(SCELTA=4)



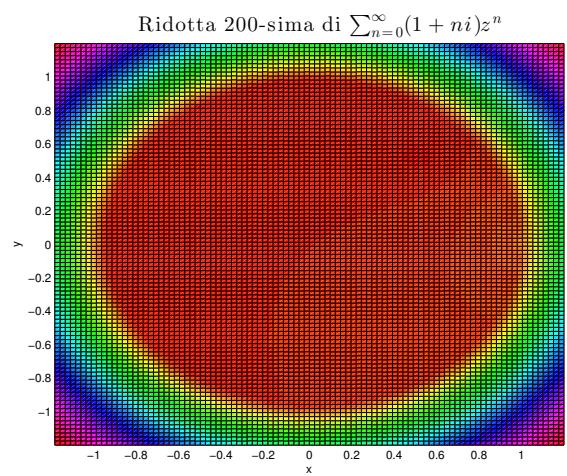
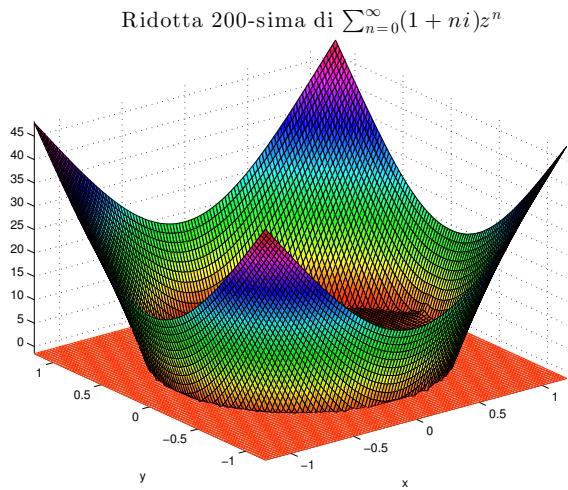
ESECUZIONE DEL PROGRAMMA(SCELTA=5)



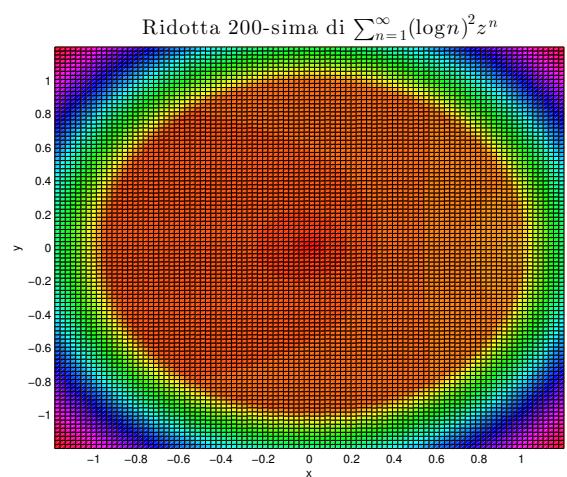
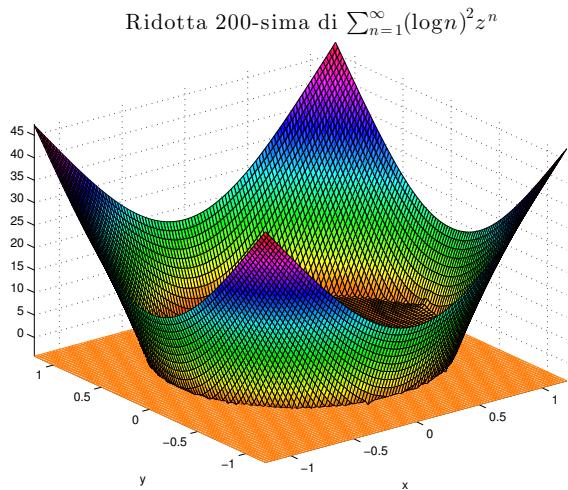
ESECUZIONE DEL PROGRAMMA(SCELTA=6)



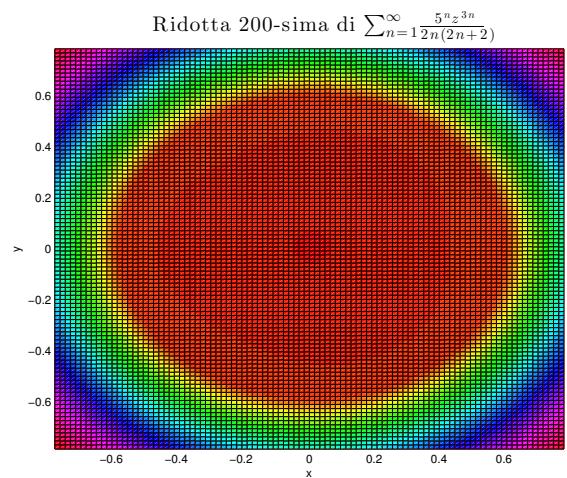
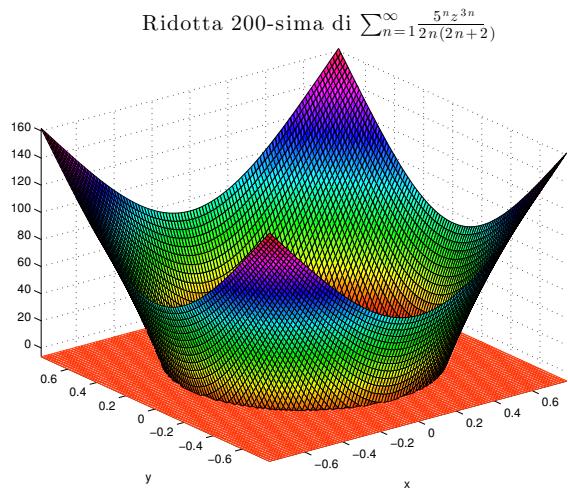
ESECUZIONE DEL PROGRAMMA(SCELTA=7)



ESECUZIONE DEL PROGRAMMA(SCELTA=8)



ESECUZIONE DEL PROGRAMMA(SCELTA=9)



3.5 Studiare le precedenti serie mediante il Symbolic Math Toolbox individuando il campo di convergenza e l'eventuale somma. Giustificare i risultati ottenuti.[liv.2]

```

1 syms x y real
2 syms n positive
3 z=x+i*y;
4 funzioni={'z^n','z^n/n','z^n/n!','z^n/(2^(2n))','(z+1)^n/(n*2^n)',...
5 '(z-2)^n/(n^2*2^(2n))','(1+ni)z^n','(logn)^2*z^n',...
6 '(5^n*z^(3n))/(2n(2n+2))'};
7 scelta=menu('Scelta funzione',funzioni);
8 k=1;
9 switch scelta
10 case 1%z^n
11     an=z^n;
12     f=inline('n/n');
13     a=0;z0=0;
14     f_limite=abs(-1/(x + y*i - 1));
15 case 2%z^n/n
16     an=z^n/n;
17     f=inline('1/n');
18     a=1;z0=0;
19     f_limite=abs(-log(1 - y*i - x));
20 case 3%z^n/n!
21     an=z^n/gamma(n+1);
22     f=inline('1/gamma(n+1)');
23     a=0;z0=0;
24     f_limite=abs(exp(x + y*i));
25 case 4%z^n/(2^(2n))
26     an=simplify(abs(z^n/4^n));
27     f=inline('1/4^n');
28     a=0;z0=0;
29     f_limite=abs(-1/(abs(x + y*i)/4 - 1));
30 case 5%(z+1)^n/(n*2^n)
31     an=simplify(abs((z+1)^n/(n*2^n)));
32     f=inline('1/(n*2^n)');
33     a=1;z0=-1;
34     f_limite=abs(-log(1 - abs(x + y*i + 1)/2));
35 case 6%(z-2)^n/(n^2*2^(2n))
36     an=simplify(abs((z-2)^n/(n^2*2^(2n))));
37     f=inline('1/(n^2*4^n)');
38     a=1;z0=2;
39 case 7%(1+ni)z^n
40     an=simplify(abs((1+n*i)*z^n));
41     f=inline('1+n*i');
42     a=0;z0=0;
43 case 8%(logn)^2*z^n
44     an=simplify(abs(log(n)^2*z^n));
45     f=inline('log(n)^2');
46     a=1;z0=0;
47 case 9%(5^n*z^(3n))/(2n(2n+2))
48     an=(5^n*z^(3*n))/(4*n^2+4*n);
49     f=inline('5^n/(4*n^2+4*n)');
50     a=1;z0=0;k=3;
51 otherwise
52     disp('Inserimento errato');
53     exit;

```

```

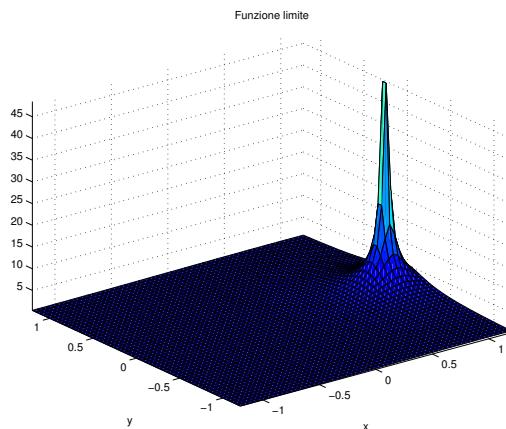
54 end
55 disp('Somma della serie');
56 S=symsum(an,n,a,inf);%Somma della serie
57 disp(S);
58 disp('Raggio di convergenza');
59 p=double((limit(f(n)/f(n+1),n,inf))^(1/k));%raggio di convergenza
60 %^(1/k) in quanto nell'ultimo caso ho z^3n mentre il calcolo del raggio di
61 %convergenza avviene rispetto a w=z^3n, quindi z=w^(1/3) e pz=pw^(1/3);
62 disp(p);
63 if p==inf
64 p=2;%per la visualizzazione della ezsurf
65 end
66 if exist('f_limite','var')
67 ezsurf(f_limite,[real(z0)-p-0.2,real(z0)+p+0.2...
68 ,imag(z0)-p-0.2,imag(z0)+p+0.2]);
69 axis tight;xlabel('x');ylabel('y');
70 title('Funzione limite');
71 end

```

ESECUZIONE DEL PROGRAMMA (SCELTA=1, $\sum_{n=0}^{\infty} z^n$)

Somma della serie
 $\text{piecewise}([1 \leq x + y*i, \text{Inf}], [abs(x + y*i) < 1, -1/(x + y*i - 1)])$

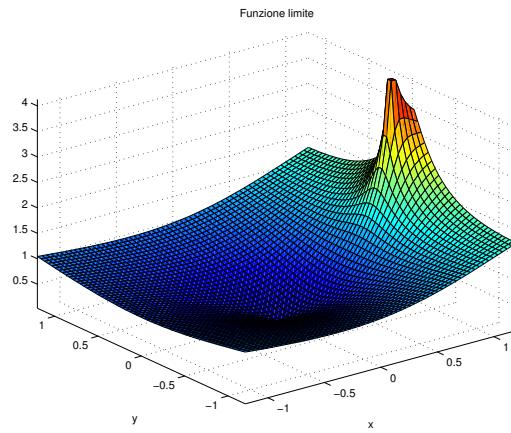
Raggio di convergenza
 1



ESECUZIONE DEL PROGRAMMA (SCELTA=2, $\sum_{n=1}^{\infty} \frac{z^n}{n}$)

Somma della serie
 $\text{piecewise}([1 \leq x + y*i, \text{Inf}], [x + y*i \approx 1 \text{ and } abs(x + y*i) \leq 1, -\log(1 - y*i - x)])$

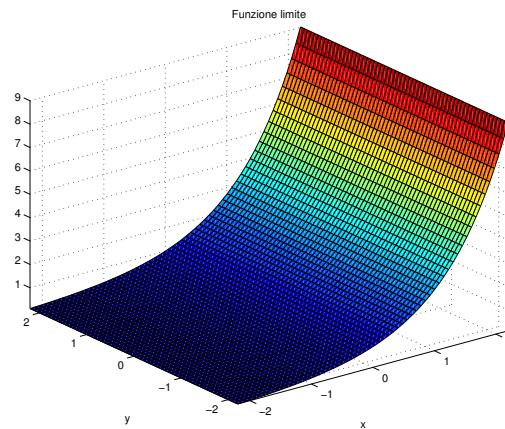
Raggio di convergenza
 1



ESECUZIONE DEL PROGRAMMA(SCELTA=3, $\sum_{n=0}^{\infty} \frac{z^n}{n!}$)

Somma della serie
 $\exp(x + y \cdot i)$

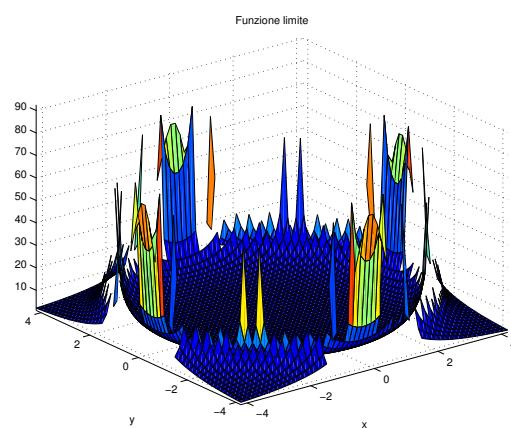
Raggio di convergenza
 Inf



ESECUZIONE DEL PROGRAMMA(SCELTA=4, $\sum_{n=0}^{\infty} \frac{z^n}{2^{2n}}$)

Somma della serie
 $\text{piecewise}([\text{abs}(x + y \cdot i) < 4, -1/(\text{abs}(x + y \cdot i)/4 - 1)], [4 \leq \text{abs}(x + y \cdot i), \text{Inf}])$

Raggio di convergenza
 4



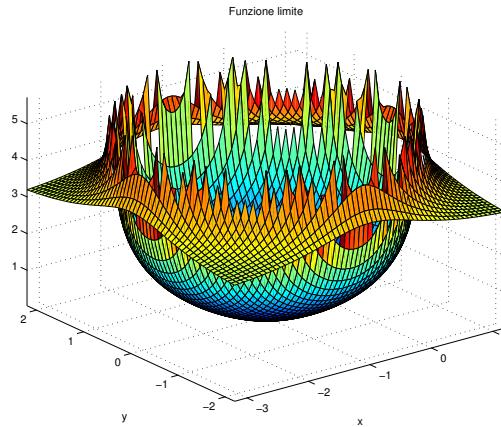
ESECUZIONE DEL PROGRAMMA(SCELTA=5, $\sum_{n=1}^{\infty} \frac{z+1^n}{n2^n}$)

Somma della serie

```
piecewise([abs(x + y*i + 1) < 2, -log(1 - abs(x + y*i + 1)/2)])
```

Raggio di convergenza

2



ESECUZIONE DEL PROGRAMMA(SCELTA=6, $\sum_{n=1}^{\infty} \frac{z-2^n}{n^2 2^{2n}}$)

Somma della serie

```
piecewise([abs(x + y*i - 2) <= 4, polylog(2, abs(x + y*i - 2)/4)])
```

Raggio di convergenza

4

ESECUZIONE DEL PROGRAMMA(SCELTA=7, $\sum_{n=0}^{\infty} (1 + ni) z^n$)

Somma della serie

```
sum(abs(x + y*i)^n * (n^2 + 1)^(1/2), n == 0 .. Inf)
```

Raggio di convergenza

1

ESECUZIONE DEL PROGRAMMA(SCELTA=8, $\sum_{n=1}^{\infty} (\log n)^2 z^n$)

Somma della serie

```
sum(log(n)^2 * (x^2 + y^2)^(n/2), n == 1 .. Inf)
```

Raggio di convergenza

1

ESECUZIONE DEL PROGRAMMA(SCELTA=9, $\sum_{n=1}^{\infty} \frac{5^n z^{3n}}{2n(2n+2)}$)

Somma della serie

```
sum((5^n * (x + y*i)^(3*n)) / (4*n^2 + 4*n), n == 1 .. Inf)
```

Raggio di convergenza

0.5848

3.6 Visualizzare il comportamento sulla frontiera delle seguenti serie di potenze nel campo complesso:geometrica,armonica/esponenziale,ed ancora le serie:

$$\sum_{n=0}^{\infty} \frac{z^n}{2^{2n}}, \quad \sum_{n=0}^{\infty} \frac{(z+1)^n}{n2^n}, \quad \sum_{n=0}^{\infty} \frac{(z-2)^n}{n^2 2^{2n}}$$

$$\sum_{n=0}^{\infty} (1+ni) z^n, \quad \sum_{n=0}^{\infty} (\log n)^2 z^n, \quad \sum_{n=0}^{\infty} \frac{5^n z^{3n}}{2n(2n+2)}$$

Per tutte le serie visualizzare anche,sia nel cerchio di convergenza sia sulla frontiera,la serie ottenuta mediante derivazione termine a termine.[liv.1]

```

1 n=200;
2 funzioni={'z^n','z^n/n','z^n/n!','z^n/(2^(2n))','(z+1)^n/(n*2^n)',...
3   '(z-2)^n/(n^2*2^(2n))','(1+ni)z^n','(logn)^2*z^n',...
4   '(5^n*z^(3n))/(2n(2n+2))'};
5 scelta=menu('Scelta funzione',funzioni);
6 switch scelta
7 case 1%z^n
8   p=1;%raggio di convergenza
9   z0=0;%punto iniziale(centro della serie)
10  k=1;
11  str='Ridotta 200-sima di $\sum_{n=0}^{\infty}\{z^n\}$';
12  an=ones(n,1);%coefficiente n-simo della serie
13  an_der=fliplr(1:n);%coefficiente n-simo della serie derivata
14 case 2%z^n/n
15   p=1;%raggio di convergenza
16   z0=0;
17   k=1;
18   str='Ridotta 200-sima di $\sum_{n=1}^{\infty}\{\frac{z^n}{n}\}$';
19   an=fliplr(1./(1:n));%coefficiente n-simo della serie
20   an=[an,0];
21   %aggiungo lo 0 in quanto nella polyval il termine di grado 0 non
22   %deve esserci(la somma va da 1 a n)
23   an_der=ones(n,1);
24 case 3%z^n/n!
25   p=2;
26   %raggio di convergenza infinito,scelgo un numero per l'intervallo
27   %di visualizzazione
28   z0=0;
29   k=1;
30   str='Ridotta 200-sima di $\sum_{n=0}^{\infty}\{\frac{z^n}{n!}\}$';
31   an=fliplr(1./gamma(2:n+1));%coefficiente n-simo della serie
32   an_der=an;
33 case 4%z^n/(2^(2n))
34   p=4;%raggio di convergenza
35   z0=0;
36   k=1;
37   str='Ridotta 200-sima di $\sum_{n=0}^{\infty}\{\frac{z^n}{2^{2n}}\}$';
38   an=fliplr(1./4.^0:n-1));%coefficiente n-simo della serie
39   an_der=fliplr((1:n)./4.^1:n));
40 case 5%(z+1)^n/(n*2^n)
41   p=2;%raggio di convergenza
42   z0=-1;
43   k=1;

```

```

45 str='Ridotta 200-sima di $\sum_{n=1}^{\infty}
46 {\frac{{z+1}^n}{n2^n}}$';
47 an=fliplr(1./((1:n).*2.^ (1:n)));%coefficiente n-simo della serie
48 an=[an,0];
49 an_der=fliplr(1./2.^ (1:n));
50 an_der=[an_der,0];
51 case 6%(z-2)^n/(n^2*2^(2n)
52 p=4;%raggio di convergenza
53 z0=2;
54 k=1;
55 str='Ridotta 200-sima di $\sum_{n=1}^{\infty}
56 {\frac{{z-2}^n}{n^{22^{2n}}}}$';
57 an=fliplr(1./(((1:n).^2).* (2.^ ((1:n)*2))));%
58 %coefficiente n-simo della serie
59 an=[an,0];
60 an_der=fliplr(1./((1:n).* (4.^ (1:n))));%
61 an_der=[an_der,0];
62 case 7%(1+ni)z^n
63 p=1;%raggio di convergenza
64 z0=0;
65 k=1;
66 str='Ridotta 200-sima di $\sum_{n=0}^{\infty}
67 {\left(1+ni\right)z^n}$';
68 an=fliplr(1+(0:n-1).*i);%coefficiente n-simo della serie
69 an_der=fliplr((1:n)+(1:n).^2.*i);
70 case 8%(logn)^2*z^n
71 p=1;%raggio di convergenza
72 z0=0;
73 k=1;
74 str='Ridotta 200-sima di $\sum_{n=1}^{\infty}
75 {\left(\log n\right)^2z^n}$';
76 an=fliplr(log(1:n).^2);%coefficiente n-simo della serie
77 an=[an,0];
78 an_der=fliplr((1:n).*log(1:n).^2);
79 an_der=[an_der,0];
80 case 9%(5^n*z^(3n))/(2n(2n+2))
81 p=(1/5)^(1/3);%raggio di convergenza
82 z0=0;
83 k=3;
84 str='Ridotta 200-sima di $\sum_{n=1}^{\infty}
85 {\frac{5^n z^{3n}}{2n \left(2n+2\right)}}$';
86 an=fliplr((5.^ (1:n))./(4.* (1:n).^2+4.* (1:n)));
87 %coefficiente n-simo della serie
88 an=[an,0];
89 an_der=fliplr((5.^ (1:n).*3.* (1:n))./(4.* (1:n).^2+4.* (1:n)));
90 an_der=[an_der,0];
91 otherwise
92 disp('Inserimento errato');
93 exit;
94 end
95 z=z0+p*cplxgrid(15);
96 Sn=polyval(an,(z-z0).^k);%valuto la serie
97 figure
98 cplxmap(z,abs(Sn)*(1+i));
99 title(str,'interpreter','latex','fontsize',18);
100 xlabel('x');ylabel('y');axis tight;
101 S1n=polyval(an_der,(z-z0).^k);%valuto la serie derivata
102 figure
103 cplxmap(z,abs(S1n)*(1+i));

```

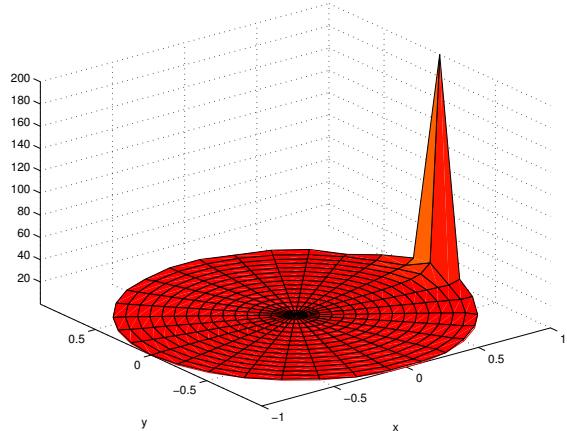
```

104 title('Serie derivata');
105 xlabel('x');ylabel('y');axis tight;
106 theta=linspace(0,2*pi,100);
107 z=z0+p*exp(i*theta);%insieme dei punti sulla frontiera
108 Sn=polyval(an,(z-z0).^k);
109 figure
110 plot(angle(z),abs(Sn));
111 grid on;
112 xlabel('$\vartheta$','interpreter','latex');
113 ylabel('$\rho$','interpreter','latex');

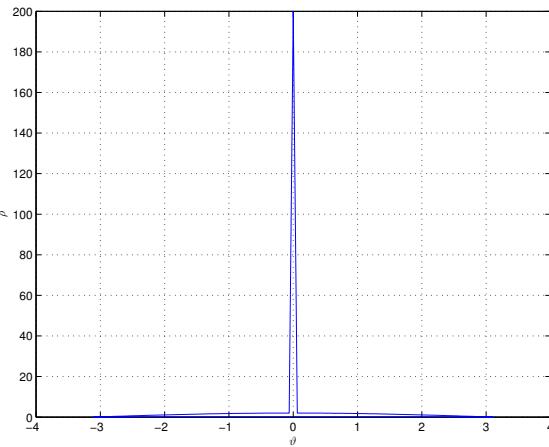
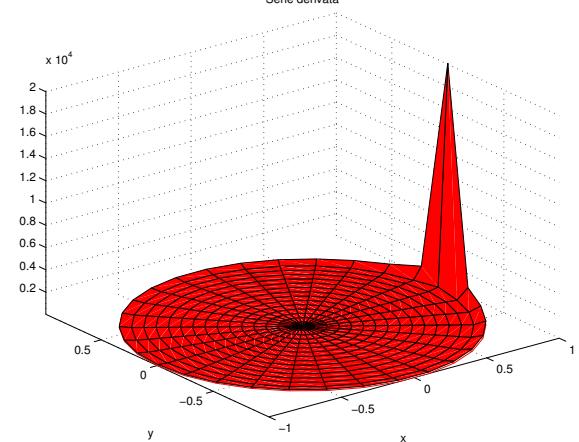
```

ESECUZIONE DEL PROGRAMMA (SCELTA=1)

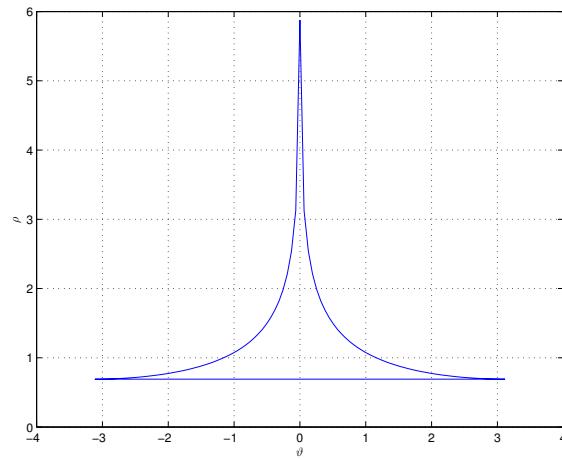
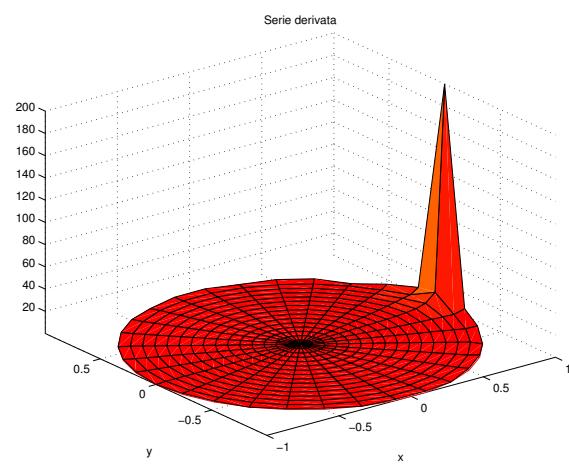
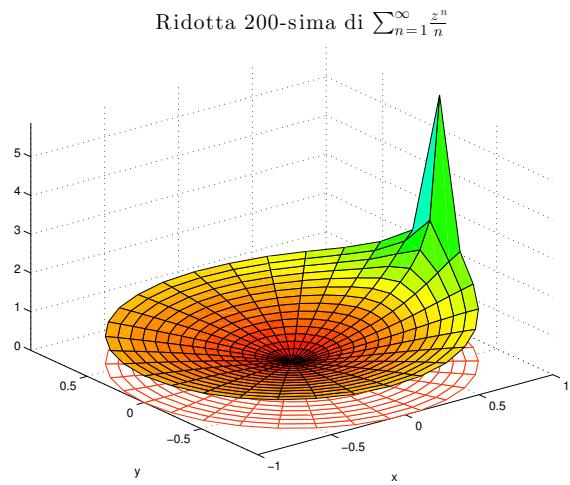
Ridotta 200-sima di $\sum_{n=0}^{\infty} z^n$



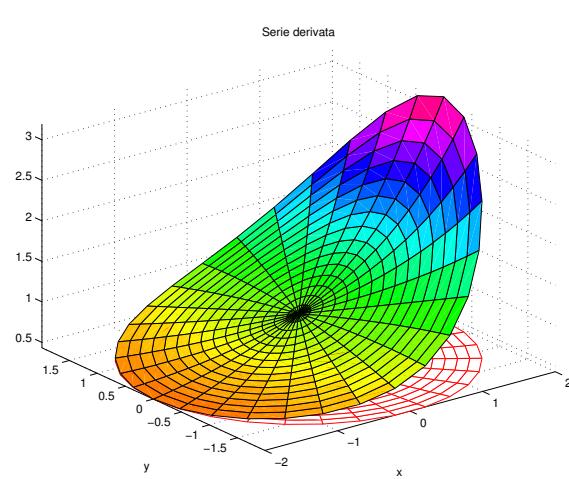
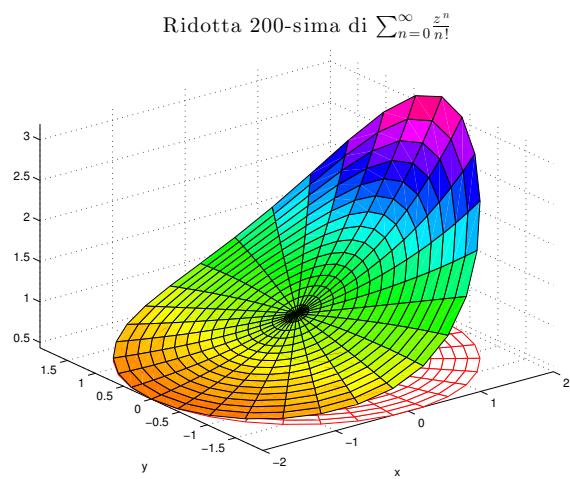
Serie derivata

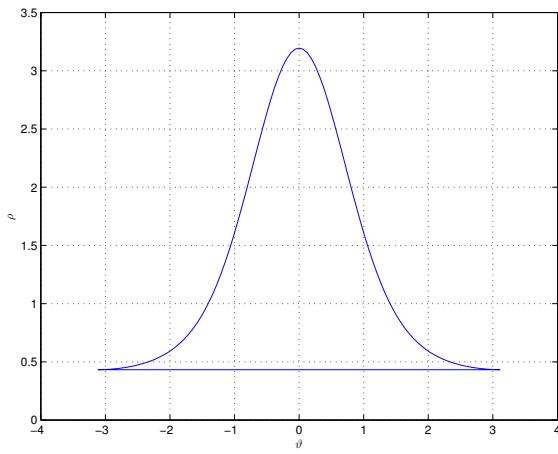


ESECUZIONE DEL PROGRAMMA(SCELTA=2)



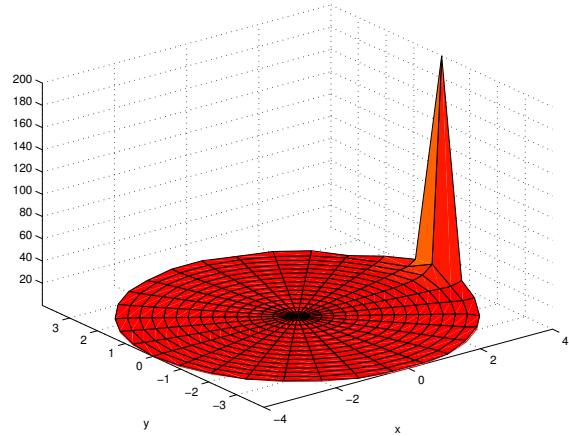
ESECUZIONE DEL PROGRAMMA(SCELTA=3)



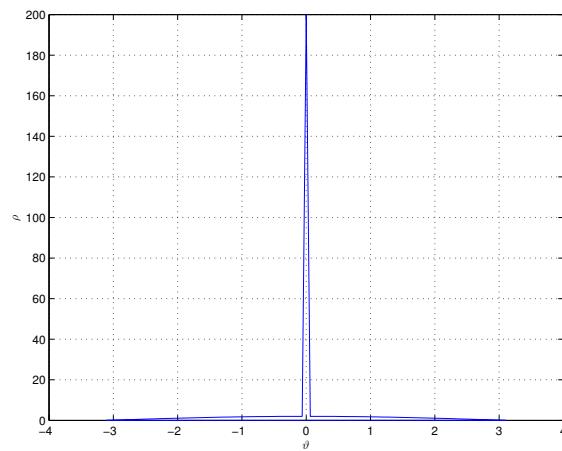
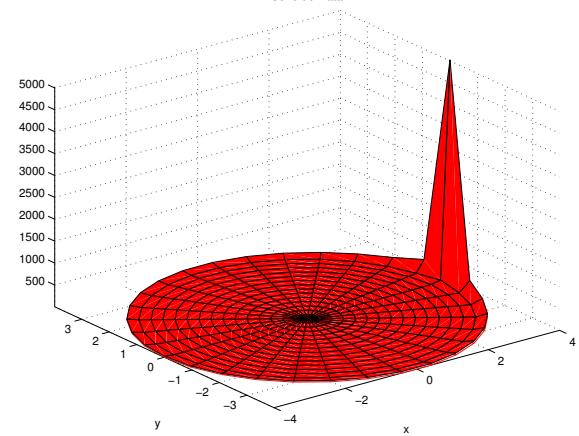


ESECUZIONE DEL PROGRAMMA(SCELTA=4)

Ridotta 200-sima di $\sum_{n=0}^{\infty} \frac{z^n}{2^{2n}}$

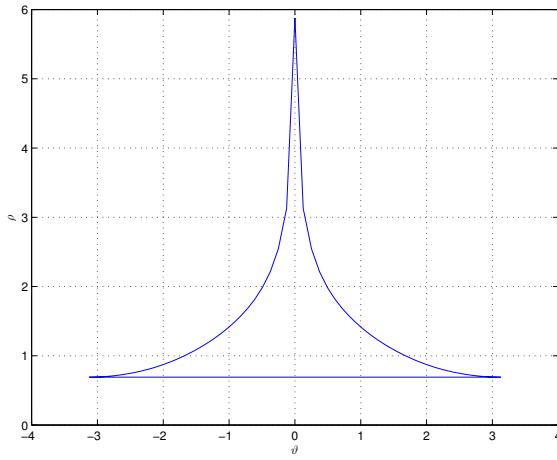
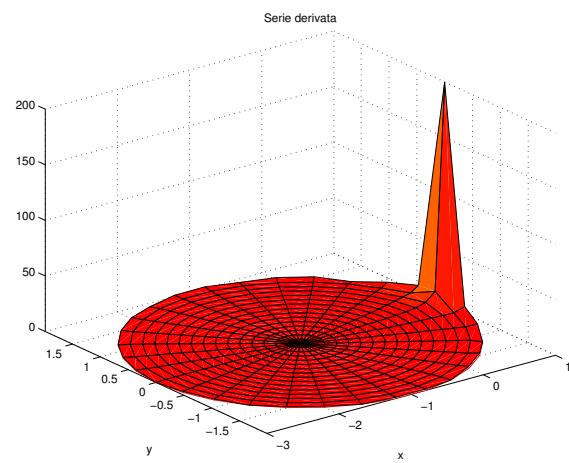
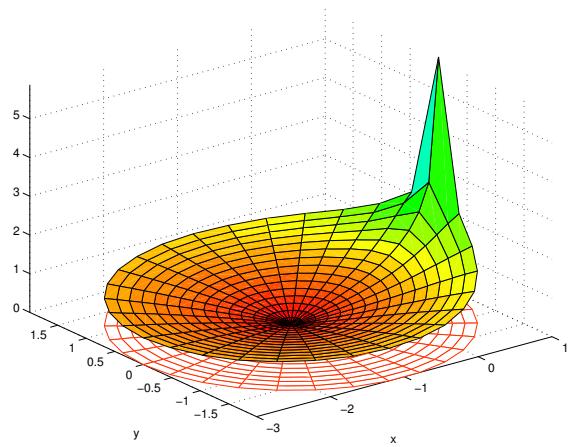


Serie derivata



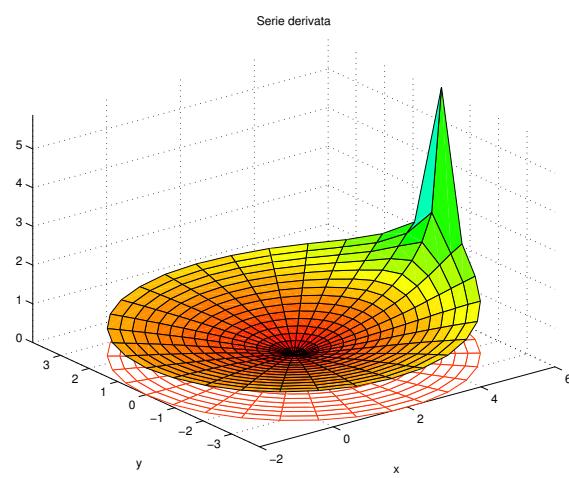
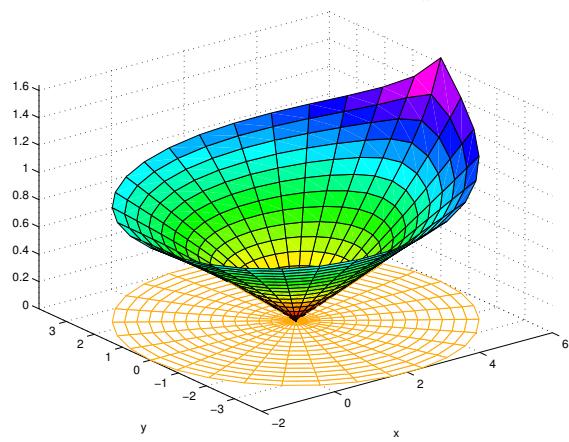
ESECUZIONE DEL PROGRAMMA(SCELTA=5)

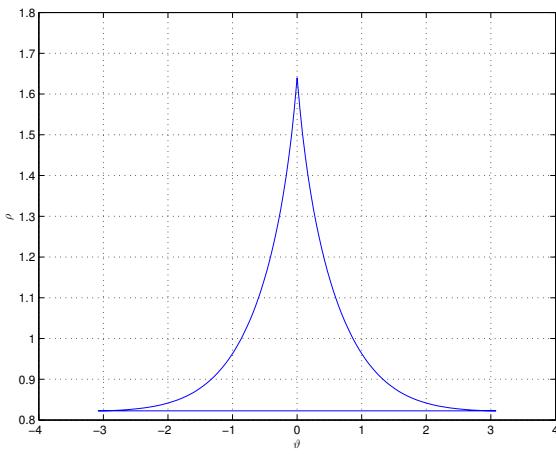
Ridotta 200-sima di $\sum_{n=1}^{\infty} \frac{z+1}{n 2^n}$



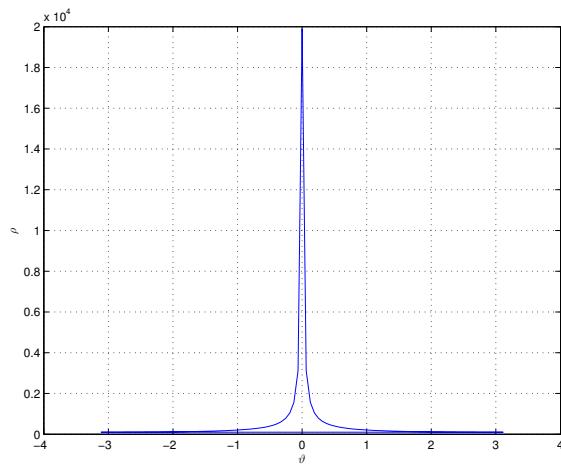
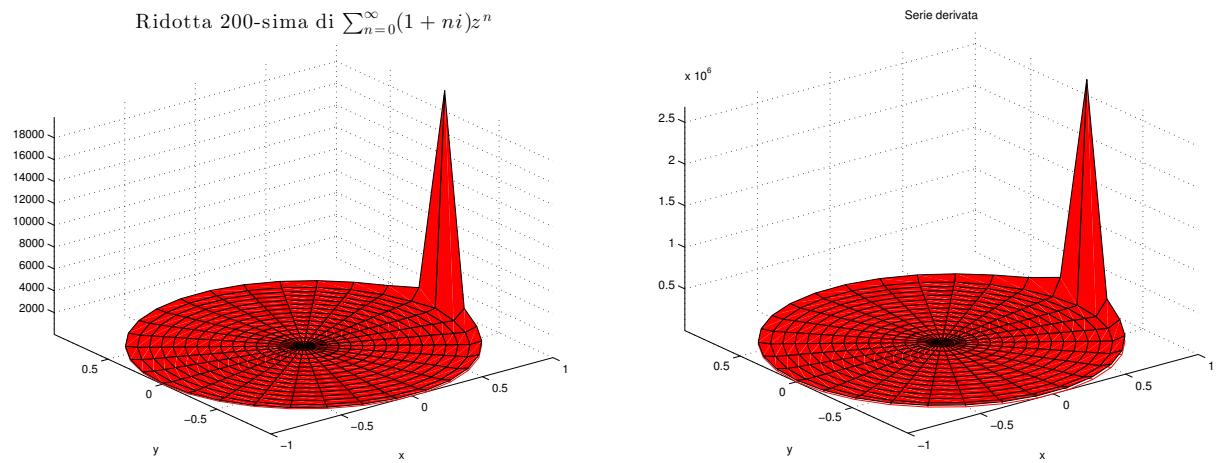
ESECUZIONE DEL PROGRAMMA(SCELTA=6)

Ridotta 200-sima di $\sum_{n=1}^{\infty} \frac{z-2}{n 2^{2n}}$



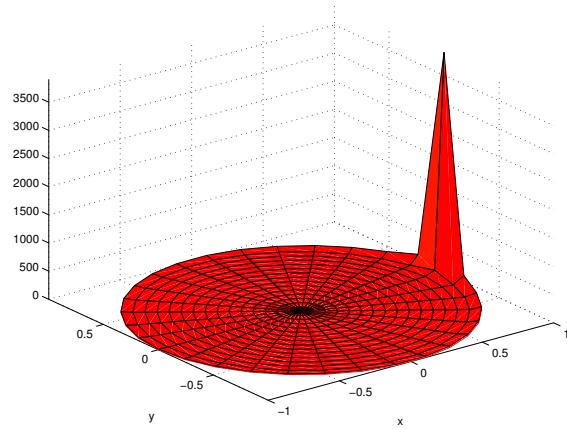


ESECUZIONE DEL PROGRAMMA(SCELTA=7)

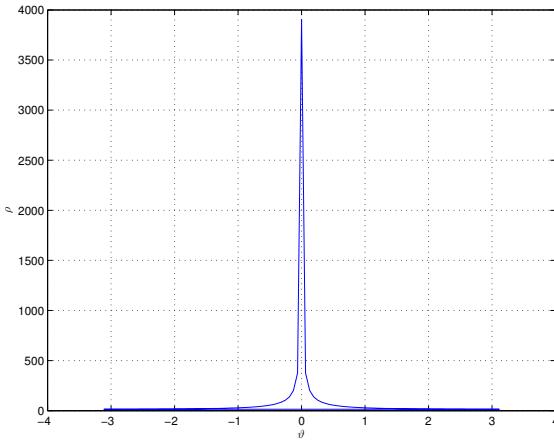
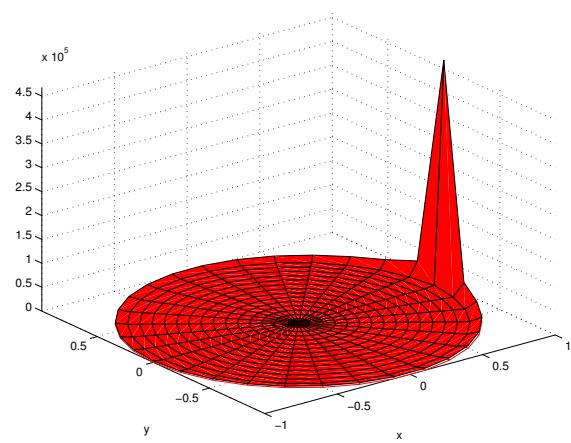


ESECUZIONE DEL PROGRAMMA(SCELTA=8)

Ridotta 200-sima di $\sum_{n=1}^{\infty} (\log n)^2 z^n$

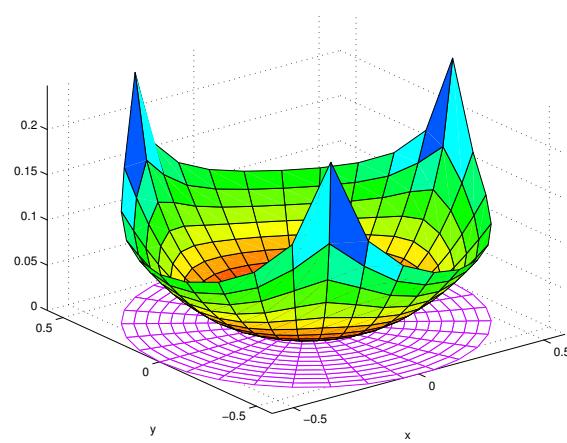


Serie derivata

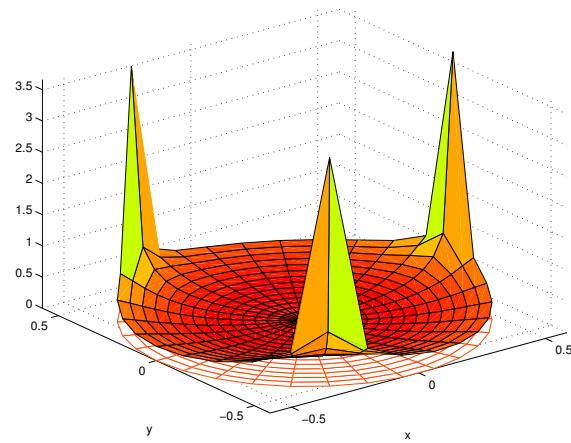


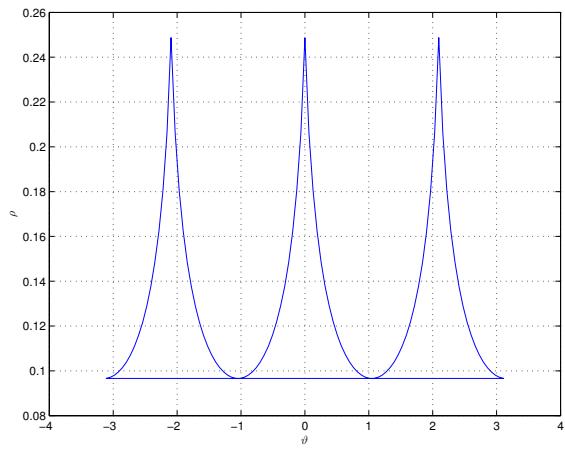
ESECUZIONE DEL PROGRAMMA(SCELTA=9)

Ridotta 200-sima di $\sum_{n=1}^{\infty} \frac{5^n z^{3n}}{2n(2n+2)}$



Serie derivata





4 Interpolazione trigonometrica e DFT

- 4.1 Per le funzioni di seguito specificate costruire (al variare del grado $N=1,2,3,4,5,6,7,8$) i polinomi trigonometrici interpolanti generando in modo random nodi di interpolazione. Rappresentare negli intervalli indicati la funzione, i nodi di interpolazione e il polinomio trigonometrico interpolante:

$$\begin{aligned} f(x) &= \sin(2x), x \in [0, 2\pi] & f(x) &= \sin(8x)^2, x \in [0, \pi] \\ f(x) &= |x|, x \in [-1, 1] & f(x) &= x^2, x \in [-1, 1] \\ f(x) &= \frac{1}{1+x^2}, x \in [-5, 5] & f(x) &= \sin(2x) - \sin(8x), x \in [-\pi, \pi] \\ f(x) &= \cos(22x), x \in [-1, 1] & f(x) &= \cos(8\pi x)^4, x \in [-2, 2] \\ f(x) &= |\sin(\pi x)|, x \in [-1, 1] & f(x) &= \{x(\pi + x), x \in [-\pi, 0] \cup (\pi - x), x \in [0, \pi]\} \end{aligned}$$

[liv.2]

```

1 if ~exist('f', 'var')
2 f=input('Scegliere la funzione:');
3 end
4 switch f
5   case 1
6     fx=inline('sin(2*x)');a=0;b=pi;u=0;v=2*pi;
7 %a e b sono gli estremi dell'intervallo del periodo di fx (e quindi dei dati)
8 %mentre u e v sono gli estremi dell'intervallo di rappresentazione dei grafici.
9   case 2
10    fx=inline('(sin(8*x)).^2');a=0;b=pi/8;u=0;v=pi;
11   case 3
12    fx=inline('abs(x)');a=-1;b=1;u=-1;v=1;
13   case 4
14    fx=inline('x.^2');a=-1;b=1;u=-1;v=1;
15   case 5
16    fx=inline('1./(1+x.^2)');a=-5;b=5;u=-5;v=5;
17   case 6
18    fx=inline('sin(2*x)-sin(8*x)');a=0;b=pi;u=-pi;v=pi;
19   case 7
20    fx=inline('cos(22*x)');a=0;b=pi/11;u=-1;v=1;
21   case 8
22    fx=inline('cos(8*pi*x).^4');a=0;b=1/8;u=-2;v=2;
23   case 9
24    fx=inline('abs(sin(pi*x))');a=0;b=1;u=-1;v=1;
25   otherwise
26     disp('Inserimento errato');
27     exit
28 end
29 for Npts=2:9
30   N=Npts-1;
31   k=0:N;
32   xk=u+(v-u)*rand(Npts,1);
33 %Genero Npts ascisse casuali in [u,v]
34   yk=fx(xk);
35 %e le valuto. I punti (xk,yk) costituiranno i punti da interpolare.
36   subplot(4,2,N);
37   hold on;grid on;
38   s=sprintf('numero punti=%d',Npts);

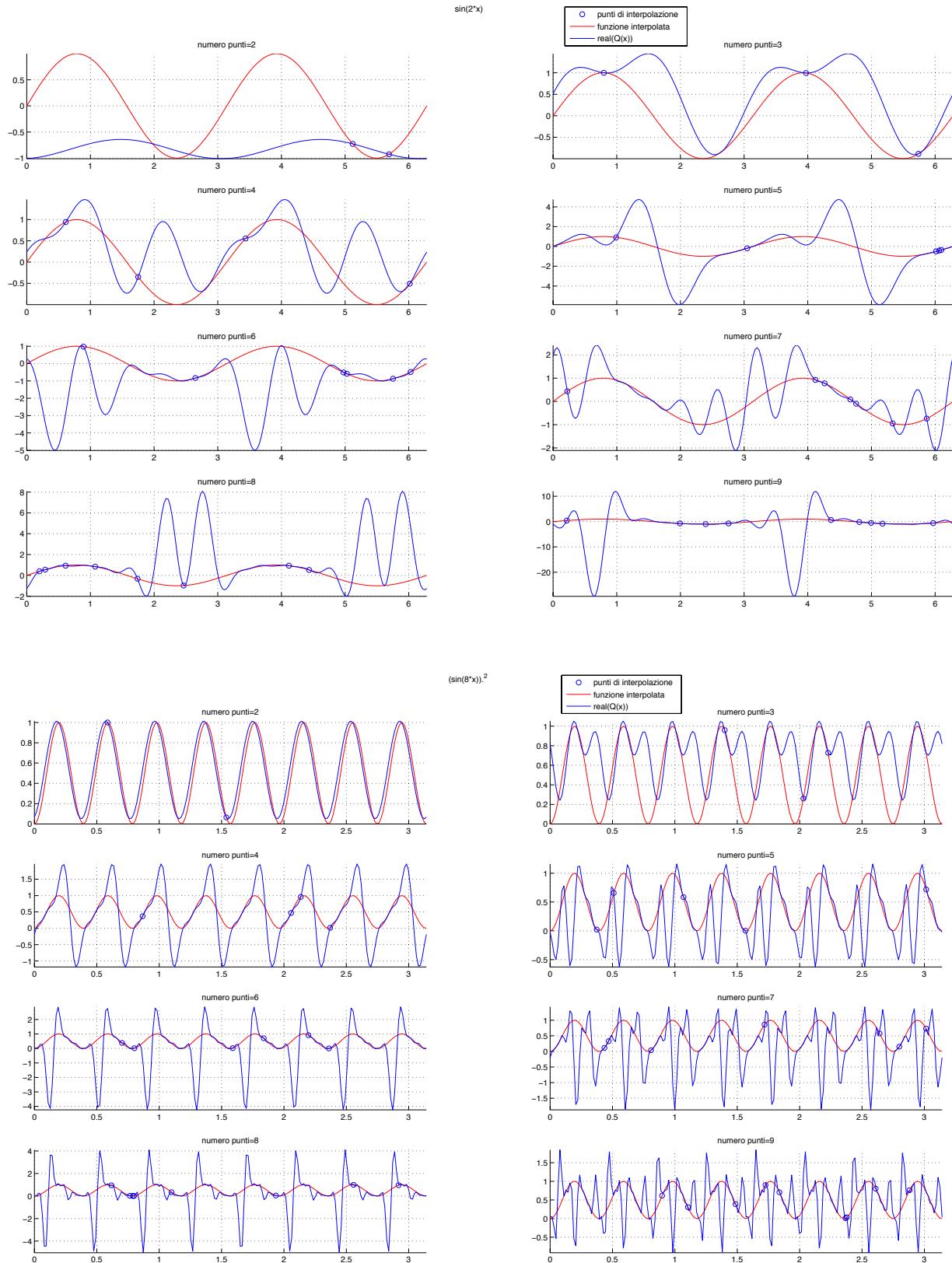
```

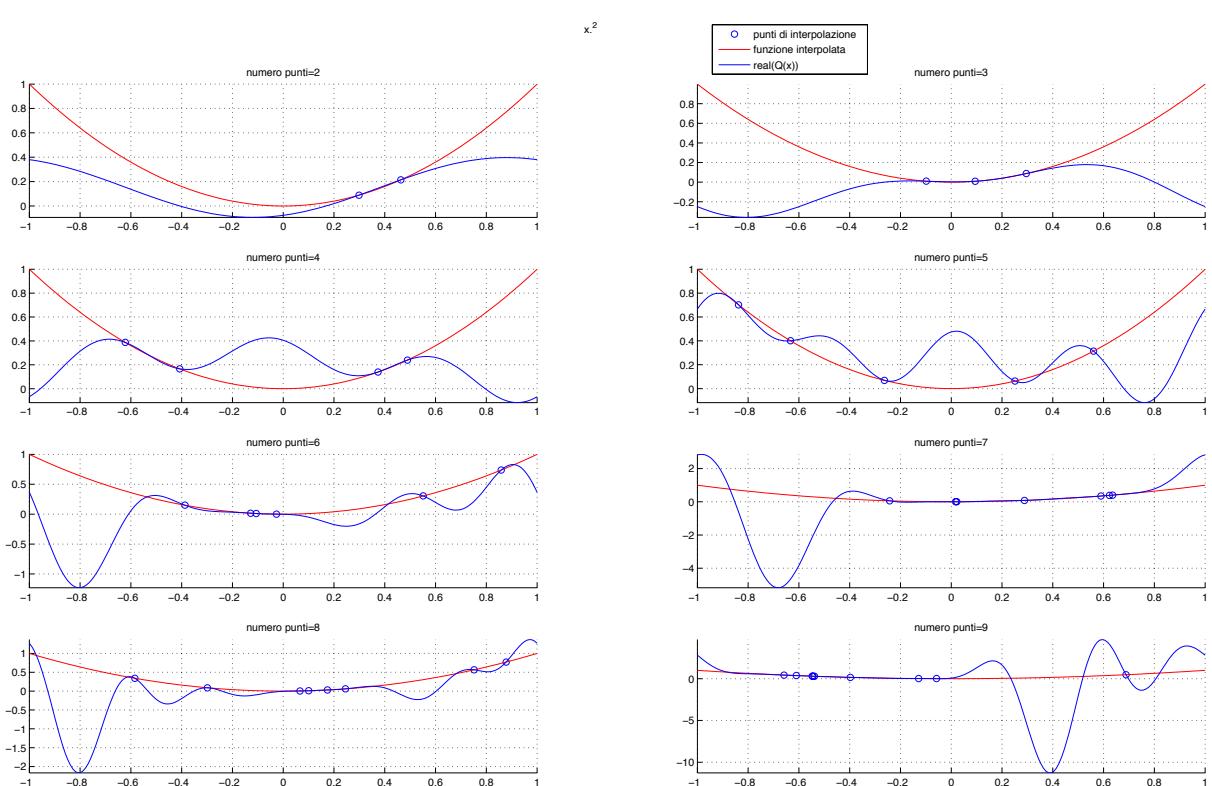
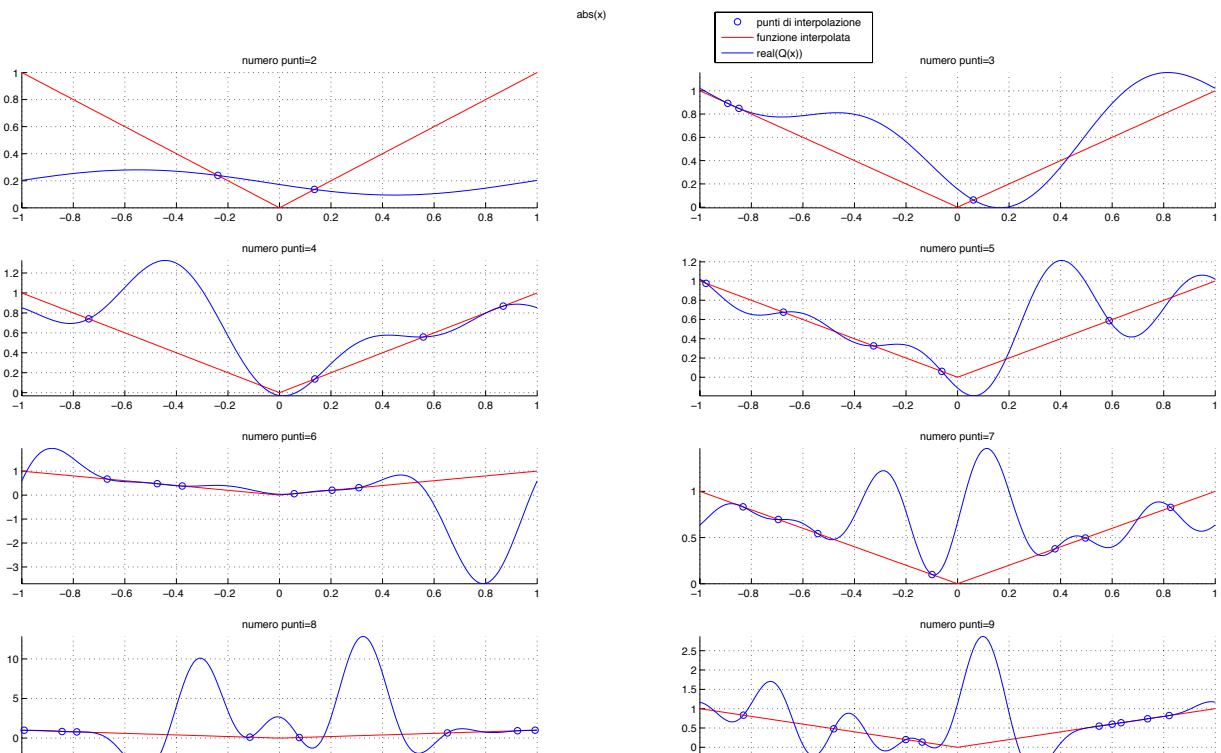
```

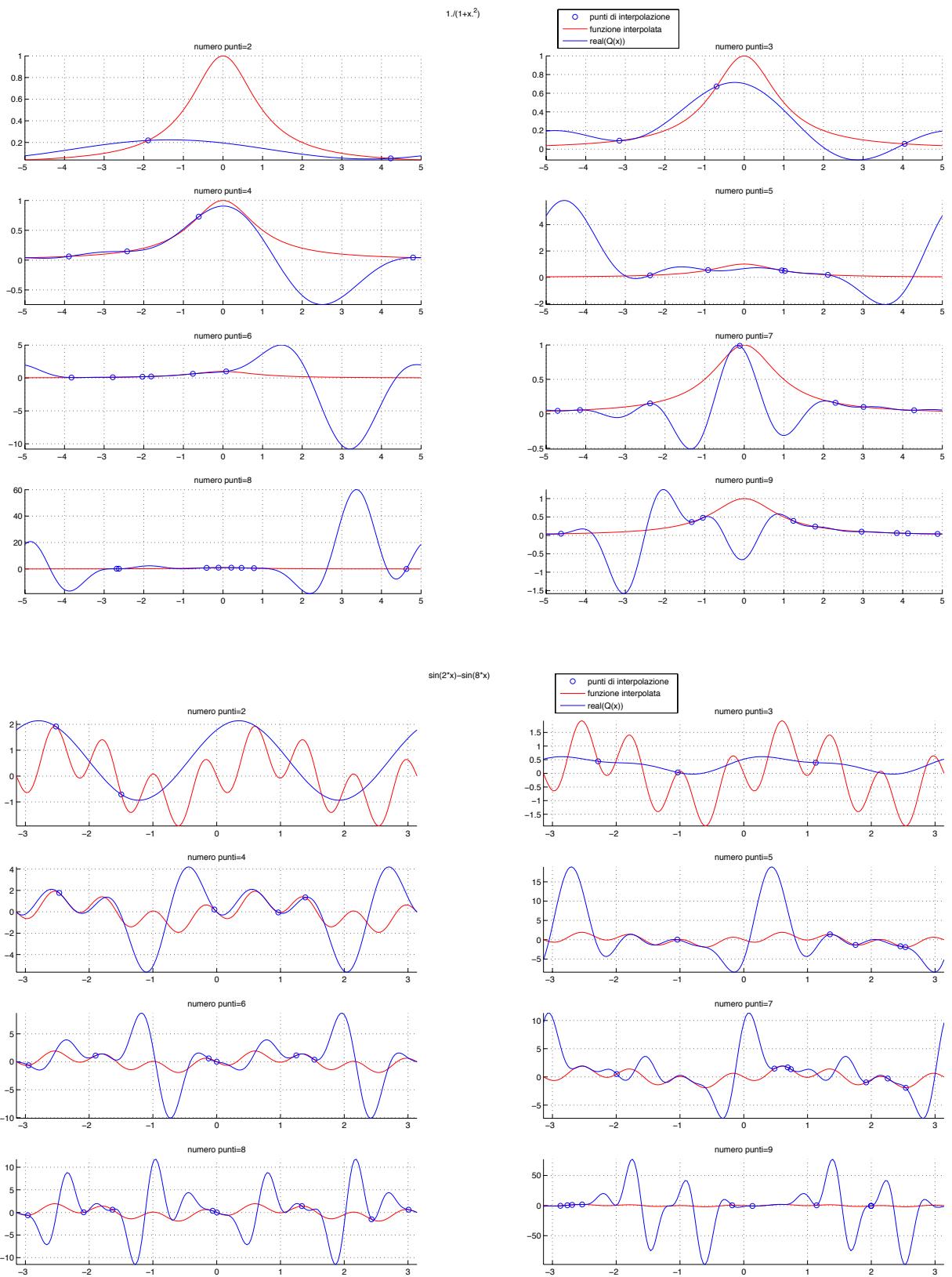
39 title(s);
40 A=exp(i*(xk-a)*((2*pi)/(b-a))*k);
41 c=A\yk;
42 t=linspace(u,v,199);
43 y=fx(t);
44 x=2*pi/(b-a)*(t-a);
45 %Siccome i dati potrebbero non essere periodici nell'intervallo [0,2*pi]
46 %proietto [a,b] in [0,2*pi].
47 Q=polyval(flipud(c),exp(i*x));
48 %Polinomio interpolante
49 plot(xk,yk,'o',t,y,'r',t,real(Q),'b');
50 axis tight
51 end
52 h1=legend('punti di interpolazione','funzione interpolata','real(Q(x))');
53 p = get(h1,'Position');
54 p(1)=0.5;p(2)=0.91;
55 set(h1,'Position',p);
56 ha=axes('Position',[0 0 1 1],'Xlim',[0 1],'Ylim',[0 1],'Box',...
57 , 'off','Visible','off',...
58 'Units','normalized','clipping','off');
59 s=char(fx);
60 text(0.5, 1,s,'HorizontalAlignment','center','VerticalAlignment','top');
61 %-----
62 %FUNZIONE A TRATTI
63 fx=inline('x.*(pi+x)');
64 fx2=inline('x.*(pi-x)');
65 a=-pi;b=pi;
66 for Npts=2:9
67 N=Npts-1;
68 k=0:N;
69 xk=a+(b-a)*rand(Npts,1);
70 x2k=xk(xk>0);
71 x1k=xk(xk<0);
72 y1k=fx(x1k);
73 y2k=fx2(x2k);
74 xk=[x1k;x2k];
75 yk=[y1k;y2k];
76 subplot(4,2,Npts-1);
77 hold on;grid on;
78 s=sprintf('numero punti=%d',Npts);
79 title(s);
80 A=exp(i*(xk-a)*((2*pi)/(b-a))*k);
81 c=A\yk;
82 t=linspace(-pi,pi,199);
83 t1=t(t<0);
84 t2=t(t>0);
85 y1=fx(t1);
86 y2=fx2(t2);
87 y=[y1 y2];
88 x=2*pi/(b-a)*(t-a);
89 Q=polyval(flipud(c),exp(i*x));
90 plot(xk,yk,'o',t,y,'r',t,real(Q),'b');
91 axis tight
92 end
93 legend('punti di interpolazione','funzione interpolata',...
94 'real(Q(x))','Location','NorthEast');

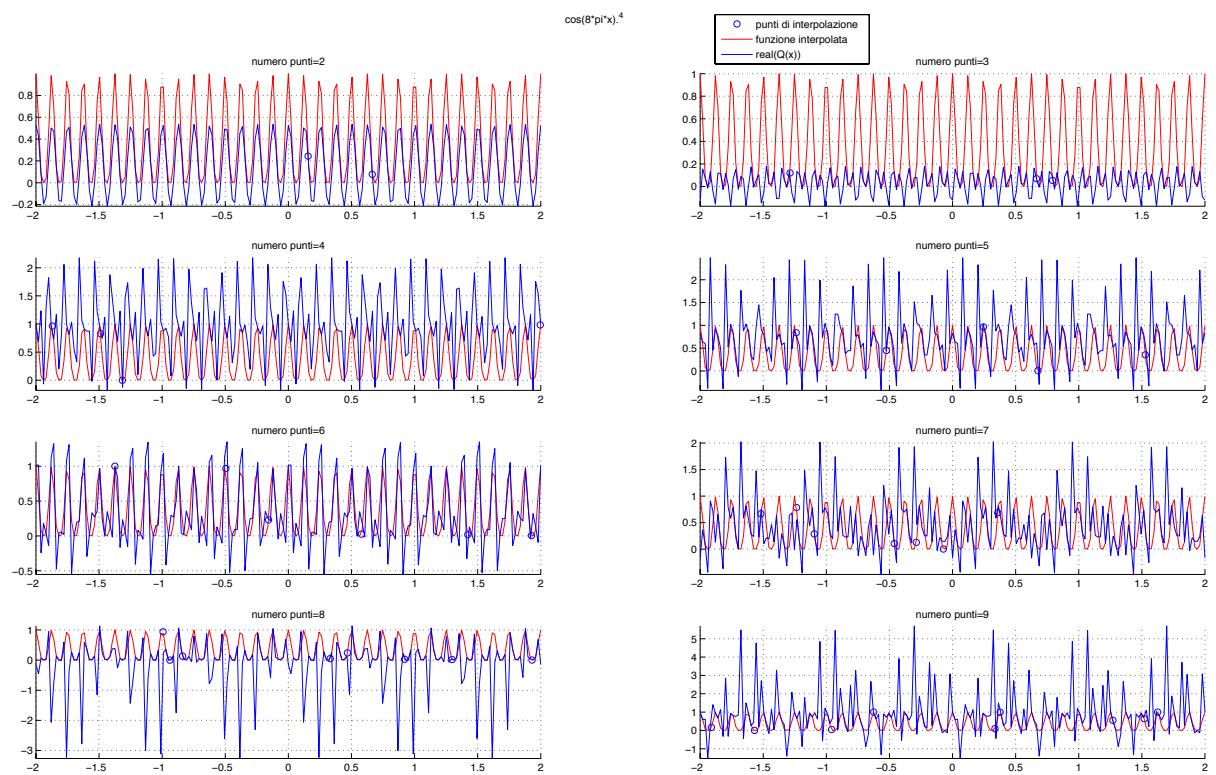
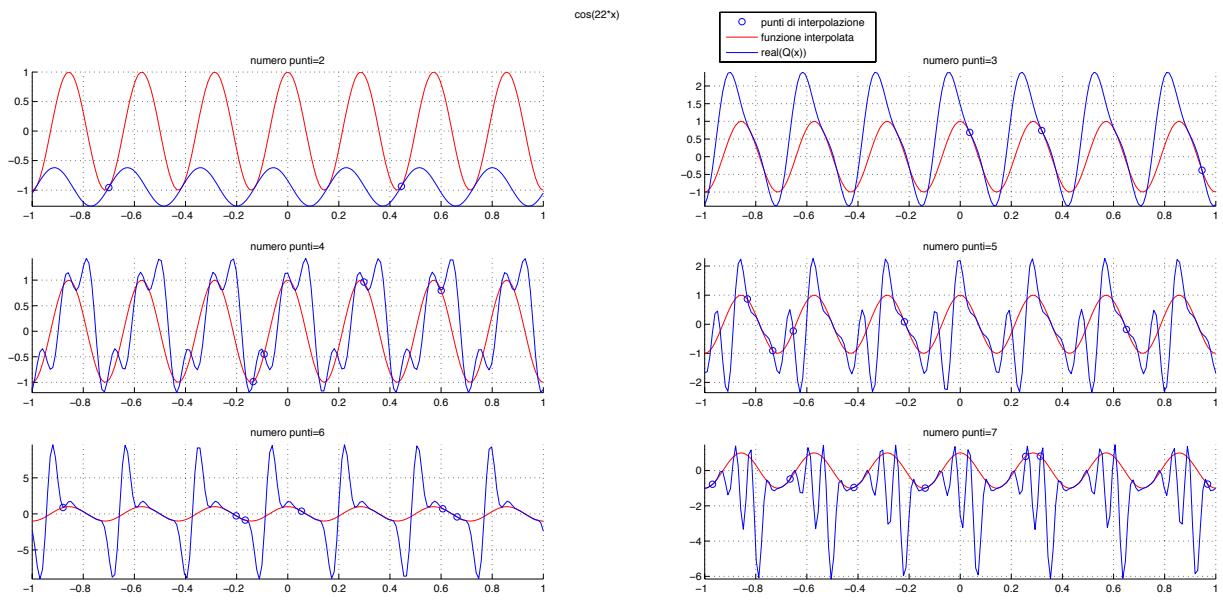
```

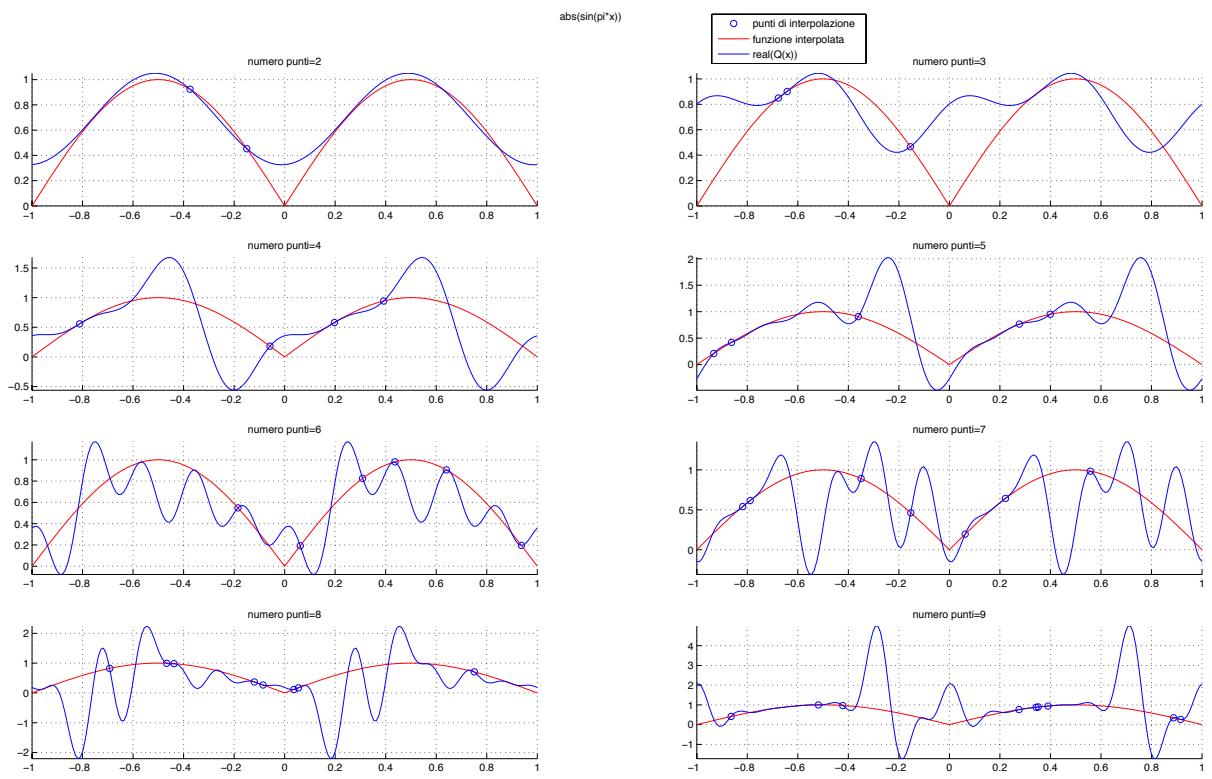
ESECUZIONE DEL PROGRAMMA



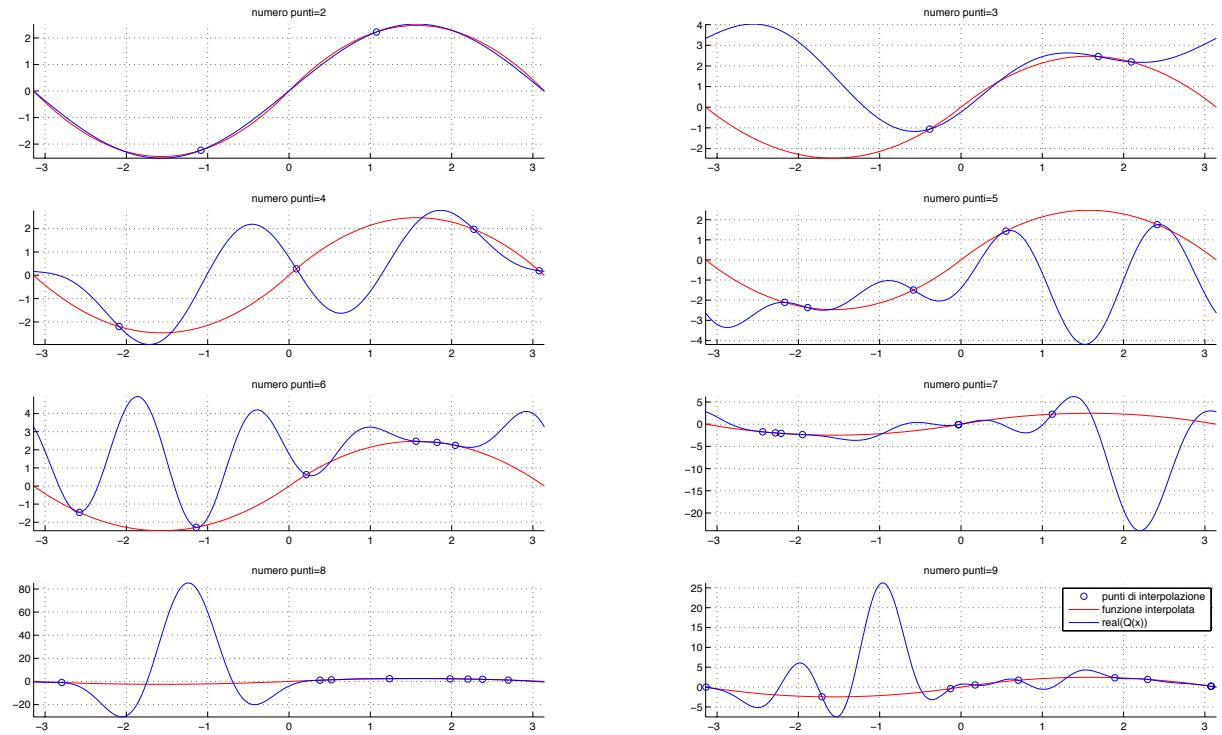








Funzione a tratti



4.2 Per le funzioni e gli intervalli di seguito specificati, costruire efficientemente, al variare del numero M di nodi, i polinomi trigonometrici interpolanti Q e T generando i nodi di interpolazione come segue: $x_i = \text{linspace}(\dots)$; $x_i = \text{rand}(\dots)$.

Confrontare graficamente tra loro i polinomi interpolanti quando vengano considerati approssimazioni delle funzioni dei dati:

$$\begin{aligned} f(x) &= \sin(2x), x \in [0, 2\pi] & f(x) &= \sin(8x)^2, x \in [0, \pi] \\ f(x) &= |x|, x \in [-1, 1] & f(x) &= x^2, x \in [-1, 1] \\ f(x) &= \frac{1}{1+x^2}, x \in [-5, 5] & f(x) &= \sin(2x) - \sin(8x), x \in [-\pi, \pi] \\ f(x) &= \cos(22x), x \in [-1, 1] & f(x) &= \cos(8\pi x)^4, x \in [-2, 2] \\ f(x) &= |\sin(\pi x)|, x \in [-1, 1] & f(x) &= \{x(\pi + x), x \in [-\pi, 0] \cup (\pi - x), x \in [0, \pi]\} \end{aligned}$$

[liv.1]

```

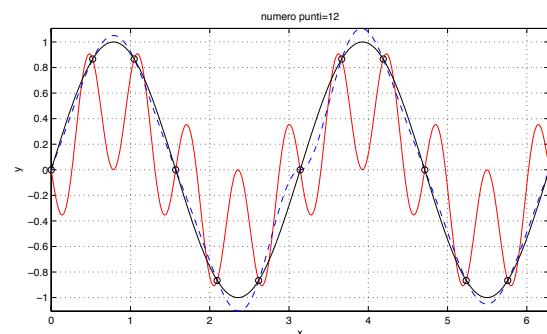
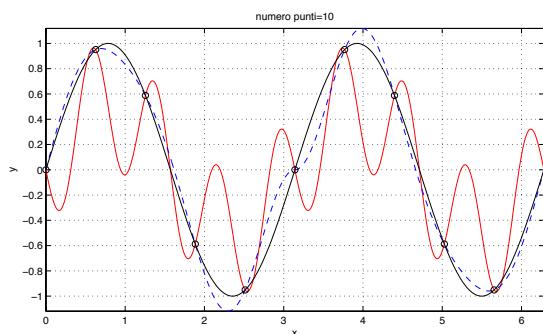
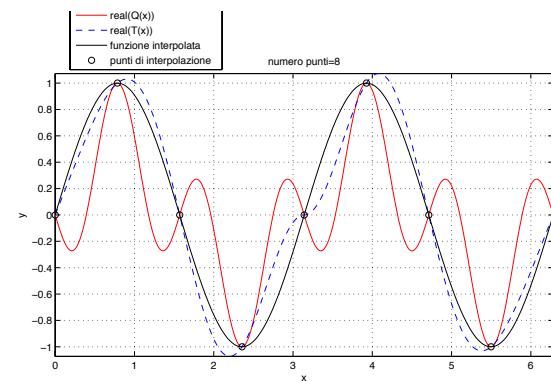
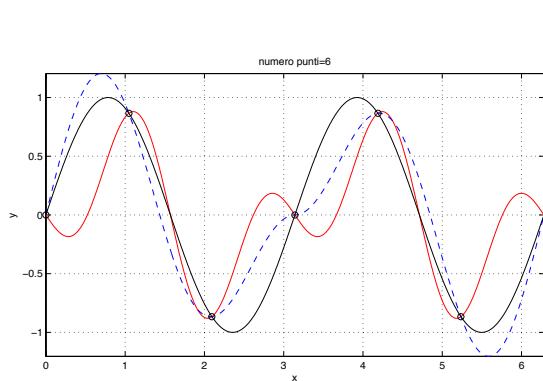
1 %INTERPOLAZIONE TRIGONOMETRICA DI NODI EQUISPAZIATI MEDIANTE FFT
2 %L'interpolazione di nodi non equispaziati (rand) e' stata eseguita
3 %nell'esercizio precedente.
4 if ~exist('f', 'var')
5     f=input('Scegliere la funzione:');
6 end
7 switch f
8     case 1
9         fx=inline('sin(2*x)');a=0;b=2*pi;
10        %a e b sono gli estremi dell'intervallo di interpolazione
11    case 2
12        fx=inline('(sin(8*x)).^2');a=0;b=pi;
13    case 3
14        fx=inline('abs(x)');a=-1;b=1;
15    case 4
16        fx=inline('x.^2');a=-1;b=1;
17    case 5
18        fx=inline('1./(1+x.^2)');a=-5;b=5;
19    case 6
20        fx=inline('sin(2*x)-sin(8*x)');a=-pi;b=pi;
21    case 7
22        fx=inline('cos(22*x)');a=-1;b=1;
23    case 8
24        fx=inline('cos(8*pi*x).^4');a=-2;b=2;
25    case 9
26        fx=inline('abs(sin(pi*x))');a=-1;b=1;
27    otherwise
28        disp('Inserimento errato');
29        exit
30    end
31    ind=1;
32    for M=6:2:12 %M indica il numero dei punti da interpolare
33        k=0:M-1;
34        x=linspace(a,b,401);
35        y=fx(x);
36        N=M-1; %grado del polinomio interpolante
37        xk=linspace(a,b,M+1)';xk(end)=[]; %genero M punti equispaziati in [a b]
38        yk=fx(xk);
```

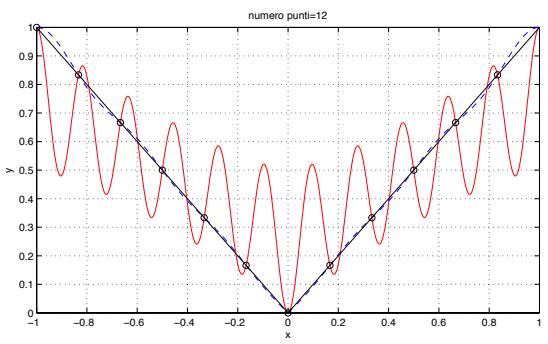
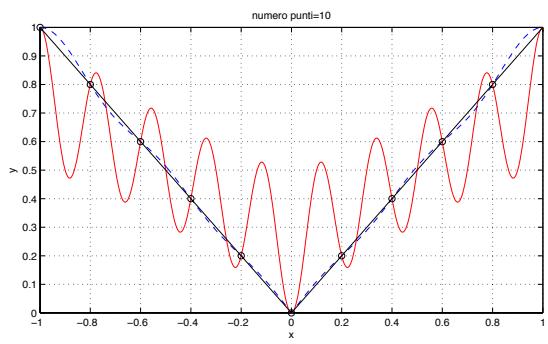
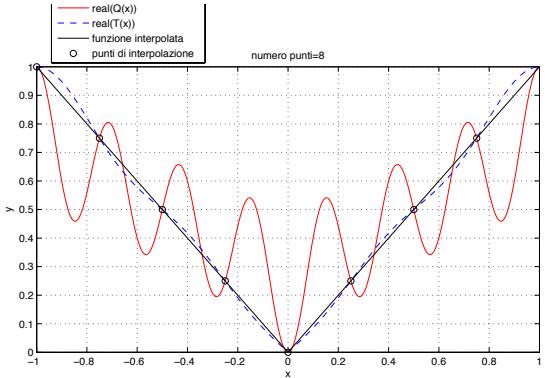
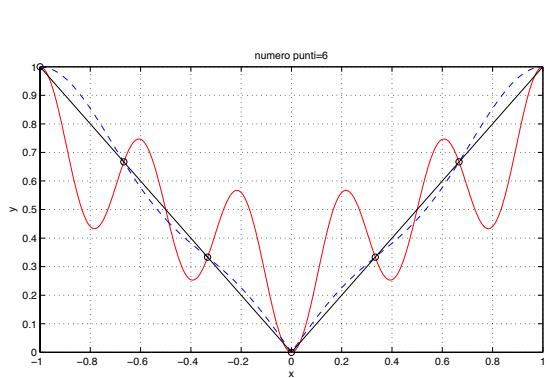
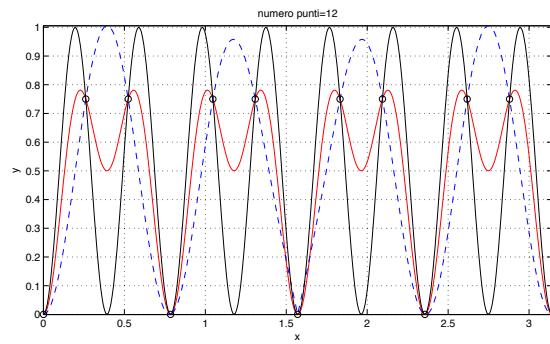
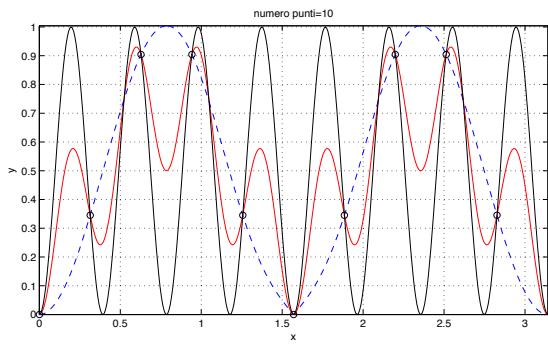
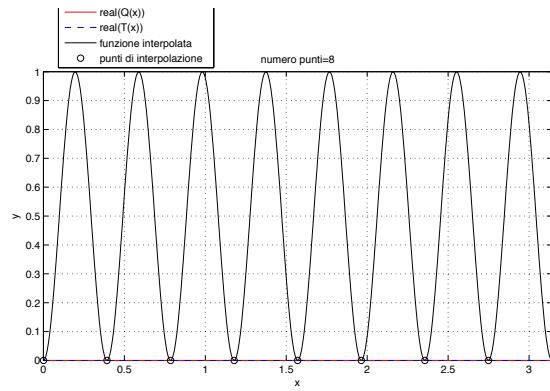
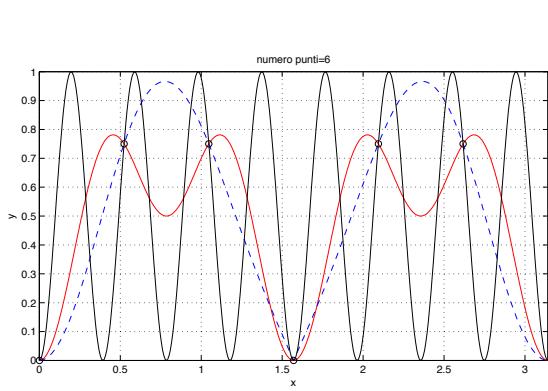
```

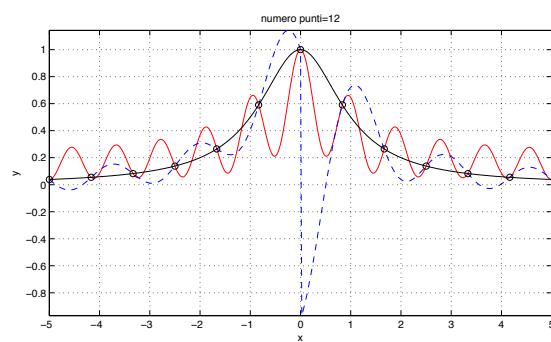
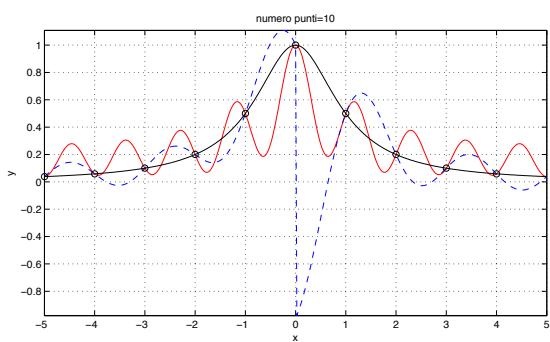
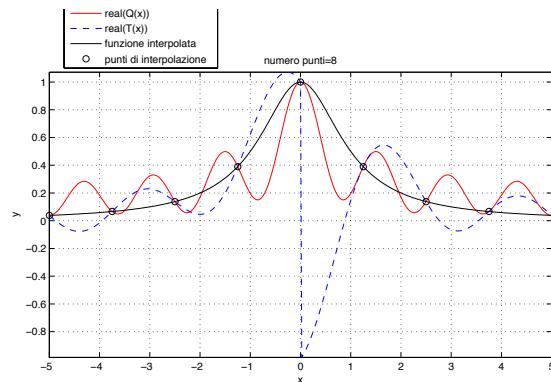
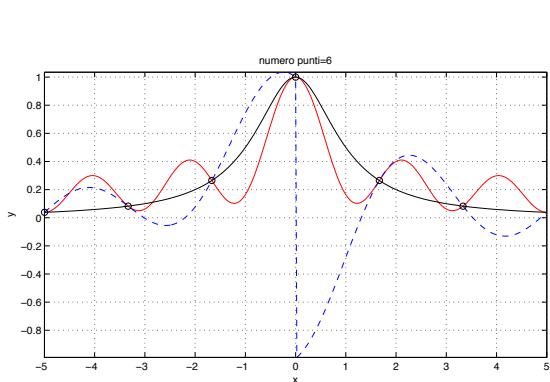
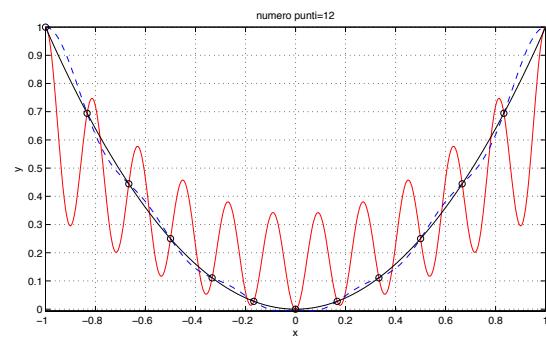
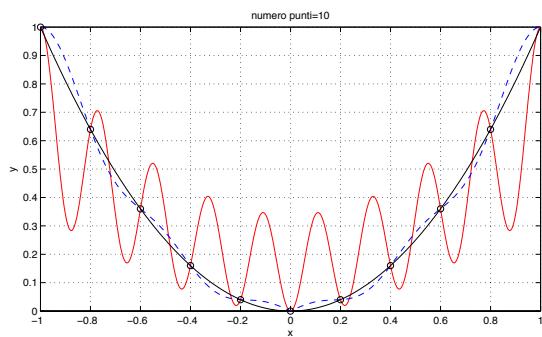
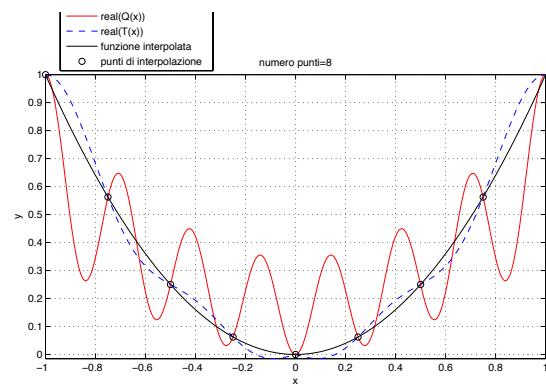
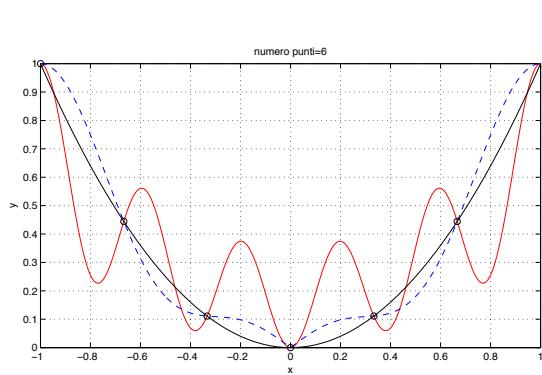
39 zk=exp(i*2*pi*(xk-a)/(b-a));
40 z=exp(i*2*pi*(x-a)/(b-a));
41 c3=fft(yk)/M;
42 c4=fft(yk.*zk.^^(N/2))/M;
43 Q3=polyval(flipud(c3),z);
44 T=z.^(-N/2).*polyval(flipud(c4),z);
45 subplot(2,2,ind);
46 ind=ind+1;
47 plot(x,real(Q3),'r',x,real(T),'b--',x,y,'k',xk,yk,'ok');
48 xlabel('x');ylabel('y');
49 s=sprintf('numero punti=%d',M);
50 title(s);grid on;axis tight;
51 end
52 h1=legend('real(Q(x))','real(T(x))','funzione interpolata',...
53 'punti di interpolazione');
54 p = get(h1,'Position');
55 p(1)=0.5;p(2)=0.895;
56 set(h1,'Position',p);
57 ha = axes('Position',[0 0 1 1],'Xlim',[0 1],'Ylim',[0 1],'Box',...
58 'off','Visible','off','Units','normalized','clipping','off');
59 s=char(fx);
60 text(0.5, 1,s,'HorizontalAlignment','center',...
61 'VerticalAlignment','top');

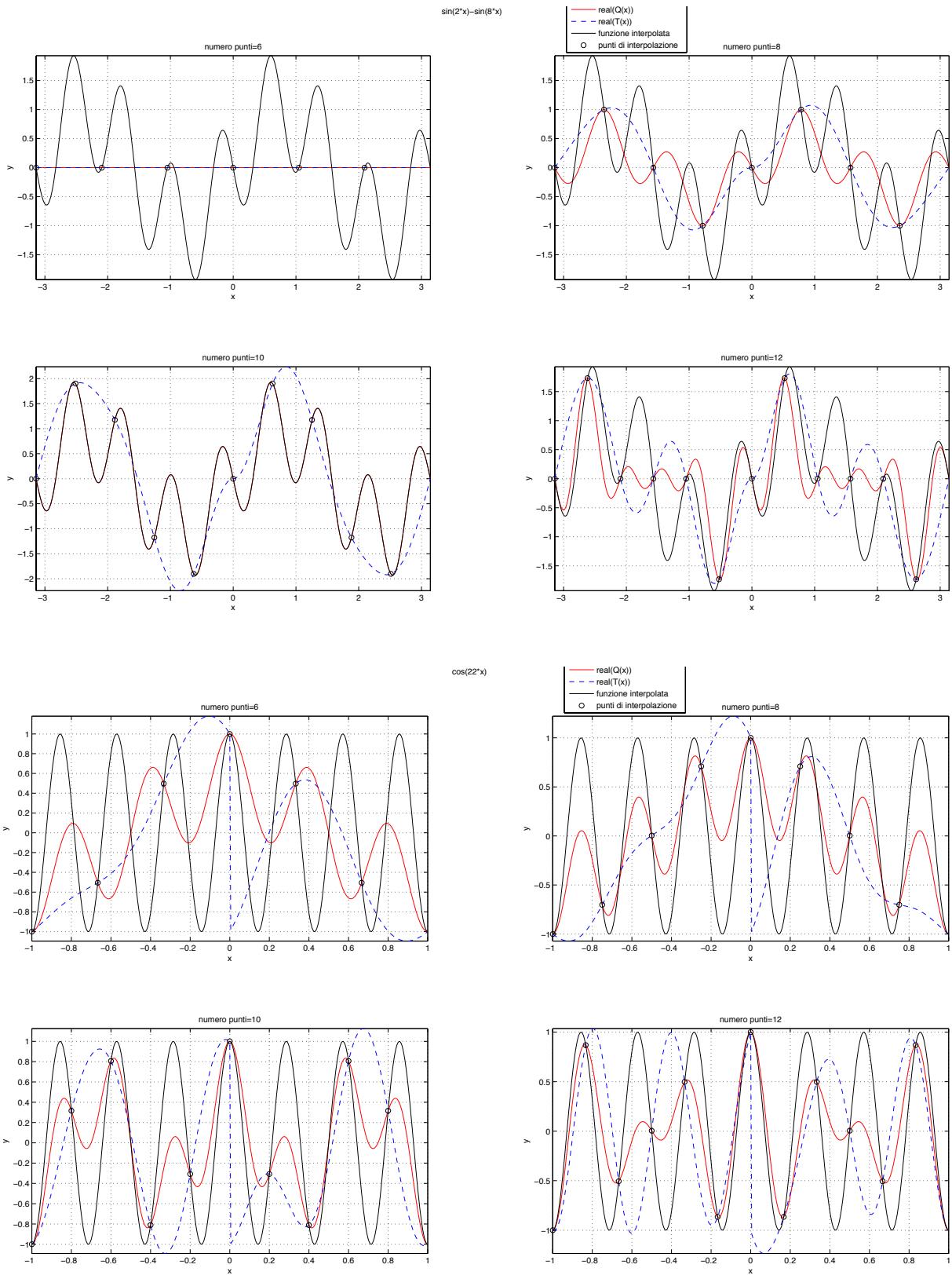
```

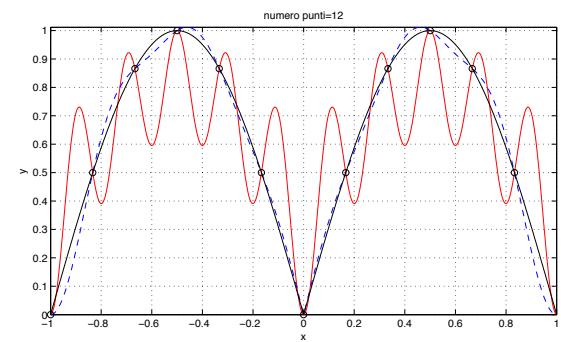
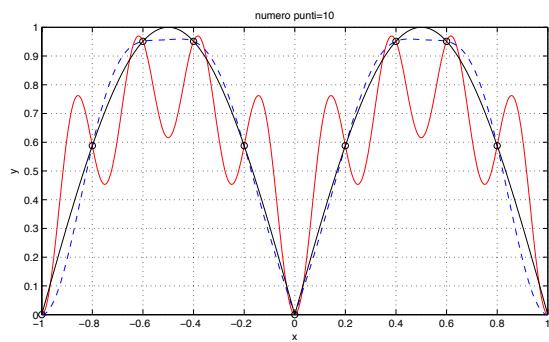
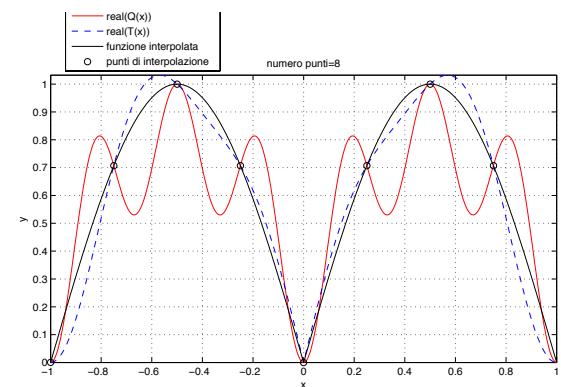
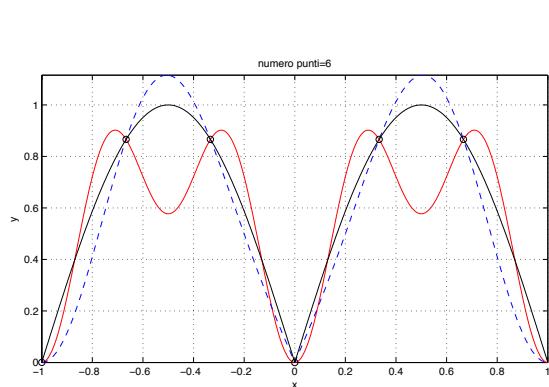
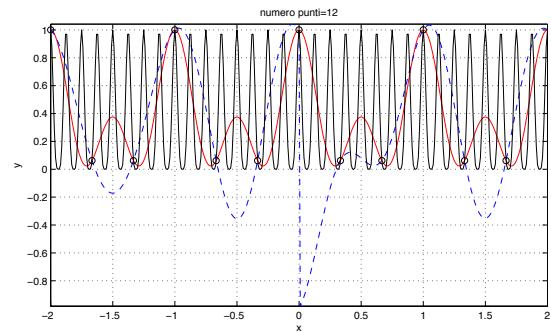
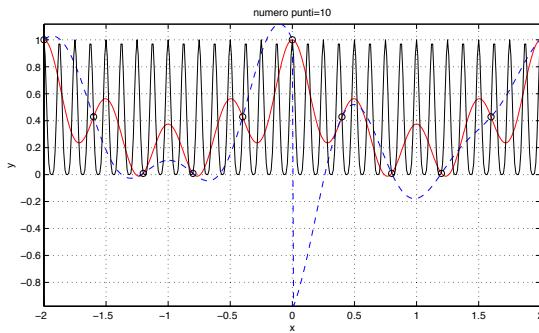
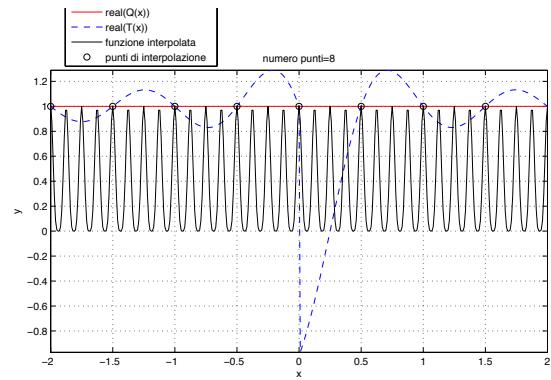
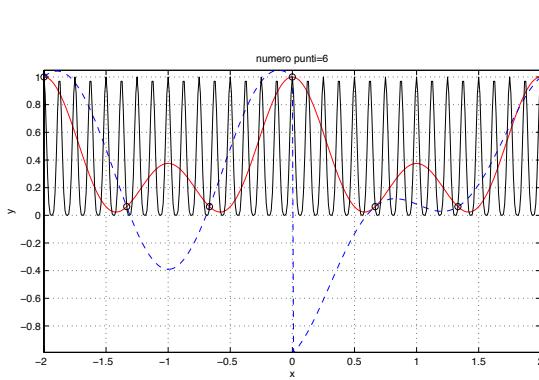
ESECUZIONE DEL PROGRAMMA











4.3 Ripetere l'esercizio precedente solo per 'molti' nodi equispaziati costruendo i coefficienti del polinomio interpolante mediante i 3 metodi. Confrontare i relativi tempi di esecuzione.[liv.3]

```

1 if ~exist('f', 'var')
2 f=input('Scegliere la funzione:');
3 end
4 switch f
5   case 1
6     fx=inline('sin(2*x)');a=0;b=2*pi;
7     %a e b sono gli estremi dell'intervallo di interpolazione
8   case 2
9     fx=inline('(sin(8*x)).^2');a=0;b=pi;
10  case 3
11    fx=inline('abs(x)');a=-1;b=1;
12  case 4
13    fx=inline('x.^2');a=-1;b=1;
14  case 5
15    fx=inline('1./(1+x.^2)');a=-5;b=5;
16  case 6
17    fx=inline('sin(2*x)-sin(8*x)');a=-pi;b=pi;
18  case 7
19    fx=inline('cos(22*x)');a=-1;b=1;
20  case 8
21    fx=inline('cos(8*pi*x).^4');a=-2;b=2;
22  case 9
23    fx=inline('abs(sin(pi*x))');a=-1;b=1;
24 otherwise
25   disp('Inserimento errato');
26   exit
27 end
28 for M=[100,1000,10000];
29   k=0:M-1;
30 x=linspace(a,b,401);
31 y=fx(x);
32 N=M-1;                                %grado del polinomio interpolante
33 xk=linspace(a,b,M+1)';xk(end)=[];    %genero M punti equispaziati in [a b]
34 yk=fx(xk);
35 zk=exp(i*2*pi*(xk-a)/(b-a));
36 z=exp(i*2*pi*(x-a)/(b-a));
37 str=char(fx);
38 fprintf('Funzione scelta:%s\n',str);
39 %METODO 1 (GAUSS)
40 tic
41 A=exp(i*(xk-a)*((2*pi)/(b-a))*k);
42 c1=A\yk;
43 fprintf('Tempo Gauss [%d nodi]\n',M);
44 toc
45 Q1=polyval(flipud(c1),z);
46 %METODO 2 (MATRICE-VETTORE)
47 tic
48 A=exp(i*(xk-a)*((2*pi)/(b-a))*k);
49 c2=A'/M*yk;
50 fprintf('Tempo Matrice*vettore [%d nodi]\n',M);
51 toc
52 Q2=polyval(flipud(c2),z);
53 %METODO 3 (FFT)

```

```

54 tic
55 c3=fft(yk)/M;
56 fprintf('Tempo FFT [%d nodi]\n',M);
57 toc
58 c4=fft(yk.*zk.^ (N/2))/M;
59 Q3=polyval(flipud(c3),z);
60 T=z.^ (-N/2).*polyval(flipud(c4),z);
61 end

```

ESECUZIONE DEL PROGRAMMA

```

Funzione scelta:1./(1+x.^2)
Tempo Gauss [100 nodi]
Elapsed time is 0.076196 seconds.
Tempo Matrice*vettore [100 nodi]
Elapsed time is 0.000509 seconds.
Tempo FFT [100 nodi]
Elapsed time is 0.000072 seconds.
Funzione scelta:1./(1+x.^2)
Tempo Gauss [1000 nodi]
Elapsed time is 0.118030 seconds.
Tempo Matrice*vettore [1000 nodi]
Elapsed time is 0.039184 seconds.
Tempo FFT [1000 nodi]
Elapsed time is 0.000092 seconds.
Funzione scelta:1./(1+x.^2)
Tempo Gauss [10000 nodi]
Elapsed time is 38.111186 seconds.
Tempo Matrice*vettore [10000 nodi]
Elapsed time is 2.966025 seconds.
Tempo FFT [10000 nodi]
Elapsed time is 0.000261 seconds.

```

4.4 Mediante circshift() simulare una stringa animata che scorre da destra a sinistra.[liv.1]

```

1 hold on
2 axis off
3 str='...Mia stringa...';
4 h=text(0.2,0.5,str,'FontSize',25);
5 pause(0.5);
6 while(1)
7     str=circshift(str',-1)';
8     set(h,'string',str);
9     pause(2);
10 end

```

4.5 Visualizzare in MATLAB le proprieta' della DFT mediante Symbolic Math Toolbox.[liv.3]

```

1 %PRIMA PROPRIETA':SE v E' REALE==>DFT(v) E' HERMITIANO SIMMETRICO.
2 syms v1 v2 v3 v4 a2 b2 h real;
3 disp('Prima proprie.:se v reale==>DFT(v) HERMITIANO SIMMETRICO');
4 disp('Vettore v');
5 v=[v1 v2 v3 v4];
6 disp(v);
7 N=numel(v);
8 w=exp(sym(-i*2*pi/N));
9 k=0:N-1;
10 W=w.^ (k'*k);
11 disp('DFT(v)');
12 dft=W*v'; %DFT del vettore v
13 pretty(dft);
14 %ESSENDO IL VETTORE DI QUATTRO COMPONENTI LA PRIMA DELLA DFT
15 %DOVRA' ESSERE REALE, LA SECONDA DOVRA' ESSERE UGUALE AL COMPLESSO
16 %CONIUGATO DELLA QUARTA COMPONENTE E LA TERZA DOVRA' ESSERE REALE
17 %IN QUANTO IL VETTORE HA UN NUMERO PARI DI ELEMENTI
18 disp('imag(dft(1),imag(dft(3))');
19 disp([imag(dft(1)) imag(dft(3))]);
20 disp('dft(2),conj(dft(4))');
21 disp([dft(2) conj(dft(4))]);
22 %-----
23 %SECONDA PROPRIETA':SE v E' HERMITIANO SIMMETRICO==>DFT(v) E' REALE.
24 disp('Seconda proprie.:se v HERMITIANO SIMMETRICO==>DFT(v) REALE');
25 disp('Vettore v');
26 v=[v1,a2+b2*i,v3,a2-b2*i];
27 disp(v);
28 disp('DFT(v)');
29 dft=W*v';
30 pretty(dft);
31 disp('imag(dft)');
32 disp(imag(dft));
33 %-----
34 %TERZA PROPRIETA':SE v E' IMMAGINARIO ALLORA DFT(v) E' HERMITIANO.
35 %ANTISIMMETRICO
36 disp('Terza proprie.:se v IMMAGINARIO==>DFT(v) HERMITIANO ANTISIMMETRICO');
37 disp('Vettore v');
38 v=[v1*i v2*i v3*i v4*i];
39 disp(v);
40 disp('DFT(v)');
41 dft=W*v';
42 pretty(dft);
43 disp('real(dft(1)),real(dft(3))');
44 disp([real(dft(1)) real(dft(3))]);
45 disp('dft(2),conj(-dft(4))');
46 disp([dft(2) conj(-dft(4))]);
47 %-----
48 %QUARTA PROPRIETA':SE v E' HERMITIANO ANTISIMMETRICO==>DFT(v) E'
49 %IMMAGINARIO.
50 disp('Quarta proprie.:se v HERMITIANO ANTISIMMETRICO==>DFT(v) IMMAGINARIO');
51 disp('Vettore v');
52 v=[v1*i,a2+b2*i,v3*i,-a2+b2*i];
53 disp(v);
54 disp('DFT(v)');

```

```

55 dft=W*v';
56 pretty(dft);
57 disp('real(dft)');
58 disp(real(dft));
59 %-----
60 %QUINTA PROPRIETA' :PROPRIETA' DELLO SHIFT.
61 disp('Quinta proprie:propriet della SHIFT');
62 disp('Vettore v');
63 v=[v1 v2 v3 v4];
64 disp(v);
65 disp('DFT(v)');
66 dft=W*v';
67 pretty(dft);
68 disp('Vettore V ottenuto ruotando di 2 posizioni v');
69 V=circshift(v',2)';
70 disp(V);
71 disp('DFT(V)');
72 dft2=W*V';
73 pretty(dft2);
74 F=exp(sym(i*2*pi/N)*h*(0:N-1));
75 F=subs(F,'h','2');
76 disp('dft(v).*F,dft(V)');
77 pretty([dft.*F dft2]);
78 %-----
79 %SESTA PROPRIETA':PRODOTTO DI CONVOLUZIONE.
80 %IMMAGINO DI VOLER ESEGUIRE IL PRODOTTO DI DUE POLINOMI P E Q
81 %ENTRAMBI DI GRADO 2.IL RISULTATO SARA' UN POLINOMIO DI GRADO 4, AVENTE
82 %I COEFFICIENTI DATI DAL PRODOTTO DI CONVOLUZIONE DEI COEFFICIENTI
83 %DI P E Q (P1,P2,P3 0 0)X(Q1 Q2 Q3 0 0).
84 disp('Quinta proprie:PRODOTTO DI CONVOLUZIONE');
85 syms x p1 p2 p3 q1 q2 q3 real
86 disp('p(x)');
87 p=p1+p2*x+p3*x^2;
88 pretty(p);
89 disp('q(x)');
90 q=q1+q2*x+q3*x^2;
91 pretty(q);
92 disp('Vettore dei coefficienti v');
93 v=[p1 p2 p3 0 0];
94 disp(v);
95 disp('Vettore dei coefficienti V');
96 V=[q1 q2 q3 0 0];
97 disp(V);
98 %v E V SONO I DUE VETTORI DEI COEFFICIENTI DI P E Q
99 N=numel(v);
100 w=exp(sym(-i*2*pi/N));
101 k=0:N-1;
102 W=w.^ (k'*k);
103 disp('DFT(v)');
104 dft=W*v';
105 pretty(dft);
106 %CALCOLO LA DFT DI v
107 disp('DFT(V)');
108 dft2=W*V';
109 pretty(dft2);
110 %CALCOLO LA DFT DI V
111 disp('DFT(v).*DFT(V)');
112 dft3=dft.*dft2;
113 pretty(dft3);

```

```

114 %CALCOLO LA MATRICE PRODOTTO DELLE DUE DFT
115 Winv=1/N*W';
116 disp('IDFT(DFT(v).*DFT(V))');
117 idft=Winv*dft3;
118 %CALCOLO LA IDFT DELLA MATRICE PRODOTTO
119 pretty(simplify(idft));
120 disp('PRODOTTO p*q');
121 pretty(collect(p*q));
122 %COME SI PUO' NOTARE I COEFFICIENTI SONO UGUALI
123 %-----
124 %SESTA PROPRIETA':UGUAGLIANZA DI PARSEVAL.
125 disp('Sesta prop.:UGUAGLIANZA DI PARSEVAL');
126 disp('Vettore v');
127 v=[v1 v2 v3 v4];
128 disp(v);
129 N=numel(v);
130 w=exp(sym(-i*2*pi/N));
131 k=0:N-1;
132 W=w.^ (k'*k);
133 disp('DFT(v)');
134 dft=W*v';
135 pretty(dft);
136 norm2v=v*v';
137 normdft=dft'*dft;
138 disp('Norma2^2 DFT(v),N*Norma2^2 v');
139 pretty([simplify(normdft) simplify(N*norm2v)]);

```

ESECUZIONE DEL PROGRAMMA

Prima prop.: se v reale $\Rightarrow DFT(v)$ HERMITIANO SIMMETRICO
 Vettore v
 $[v_1, v_2, v_3, v_4]$

$DFT(v)$

$$\begin{array}{c} + - \\ | \quad v_1 + v_2 + v_3 + v_4 \quad | \\ | \\ | \quad v_1 - v_2 i - v_3 + v_4 i \quad | \\ | \\ | \quad v_1 - v_2 + v_3 - v_4 \quad | \\ | \\ | \quad v_1 + v_2 i - v_3 - v_4 i \quad | \\ + - \end{array}$$

$imag(dft(1), imag(dft(3)))$
 $[0, 0]$

$dft(2), conj(dft(4))$
 $[v_1 - v_2*i - v_3 + v_4*i, v_1 - v_2*i - v_3 + v_4*i]$

Seconda prop.: se v HERMITIANO SIMMETRICO $\Rightarrow DFT(v)$ REALE
 Vettore v
 $[v_1, a_2 + b_2*i, v_3, a_2 - b_2*i]$

$DFT(v)$

$$\begin{array}{c} + - \end{array}$$

```

| 2 a2 + v1 + v3 |
|                   |
| v1 - 2 b2 - v3 |
|                   |
| v1 - 2 a2 + v3 |
|                   |
| 2 b2 + v1 - v3 |
+-                  +-+
imag(dft)
0
0
0
0

Terza propr.: se v IMMAGINARIO==>DFT(v) HERMITIANO ANTISIMMETRICO
Vettore v
[ v1*i, v2*i, v3*i, v4*i]

DFT(v)

+-                  +-+
| - v1 i - v2 i - v3 i - v4 i |
|                   |
| v4 - v2 + v3 i - v1 i |
|                   |
| v2 i - v1 i - v3 i + v4 i |
|                   |
| v2 - v4 + v3 i - v1 i |
+-                  +-+
real(dft(1)), real(dft(3))
[ 0, 0]

dft(2), conj(-dft(4))
[ v4 - v2 + v3*i - v1*i, v4 - v2 + v3*i - v1*i]

Quarta propr.: se v HERMITIANO ANTISIMMETRICO==>DFT(v) IMMAGINARIO
Vettore v
[ v1*i, a2 + b2*i, v3*i, - a2 + b2*i]

DFT(v)

+-                  +-+
| - 2 b2 i - v1 i - v3 i |
|                   |
| v3 i - v1 i - 2 a2 i |
|                   |
| 2 b2 i - v1 i - v3 i |
|                   |
| 2 a2 i - v1 i + v3 i |
+-                  +-+
real(dft)
0
0
0
0

Quinta propr.: proprietà dello SHIFT
Vettore v
[ v1, v2, v3, v4]

```

DFT (v)

$$\begin{array}{c} +- \\ | \quad v1 + v2 + v3 + v4 \\ | \\ | \quad v1 - v2 i - v3 + v4 i \\ | \\ | \quad v1 - v2 + v3 - v4 \\ | \\ | \quad v1 + v2 i - v3 - v4 i \\ +- \end{array}$$

Vettore V ottenuto ruotando di 2 posizioni v
[v3, v4, v1, v2]

DFT (V)

$$\begin{array}{c} +- \\ | \quad v1 + v2 + v3 + v4 \\ | \\ | \quad v3 + v2 i - v1 - v4 i \\ | \\ | \quad v1 - v2 + v3 - v4 \\ | \\ | \quad v3 - v2 i - v1 + v4 i \\ +- \end{array}$$

dft (v) .*F, dft (V)

$$\begin{array}{c} +- \\ | \quad v1 + v2 + v3 + v4, \quad v1 + v2 + v3 + v4 \\ | \\ | \quad v3 + v2 i - v1 - v4 i, \quad v3 + v2 i - v1 - v4 i \\ | \\ | \quad v1 - v2 + v3 - v4, \quad v1 - v2 + v3 - v4 \\ | \\ | \quad v3 - v2 i - v1 + v4 i, \quad v3 - v2 i - v1 + v4 i \\ +- \end{array}$$

Quinta prop.: PRODOTTO DI CONVOLUZIONE
p(x)

$$p_3 x^2 + p_2 x + p_1$$

q(x)

$$q_3 x^2 + q_2 x + q_1$$

Vettore dei coefficienti v
[p1, p2, p3, 0, 0]

Vettore dei coefficienti V
[q1, q2, q3, 0, 0]

DFT (v)

$$\begin{array}{c} +- \\ | \quad p1 + p2 + p3 \\ | \\ | \quad p1 + p3 \#1 - p2 \#1 \\ | \end{array}$$

```

|           2           4   |
| p1 + p2 #1 + p3 #1   |
|           |           |
|           3           6   |
| p1 - p2 #1 + p3 #1   |
|           |           |
|           4           8   |
| p1 + p2 #1 + p3 #1   |
+-                   +-+

```

where

$$\#1 = \frac{1/4 - \frac{5}{4} + \frac{1/2(5 + 5)}{4}}{i}$$

DFT (V)

```

+-               +-+
| q1 + q2 + q3   |
|           |           |
|           2           |
| q1 + q3 #1 - q2 #1   |
|           |           |
|           2           4   |
| q1 + q2 #1 + q3 #1   |
|           |           |
|           3           6   |
| q1 - q2 #1 + q3 #1   |
|           |           |
|           4           8   |
| q1 + q2 #1 + q3 #1   |
+-                   +-+

```

where

$$\#1 = \frac{1/4 - \frac{5}{4} + \frac{1/2(5 + 5)}{4}}{i}$$

DFT (v) .*DFT (V)

```

+-               +-+
| (p1 + p2 + p3) (q1 + q2 + q3)   |
|           |           |
|           2           2   |
| (p1 + p3 #1 - p2 #1) (q1 + q3 #1 - q2 #1)   |
|           |           |
|           2           4           2           4   |
| (p1 + p2 #1 + p3 #1) (q1 + q2 #1 + q3 #1)   |
|           |           |
|           3           6           3           6   |
| (p1 - p2 #1 + p3 #1) (q1 - q2 #1 + q3 #1)   |
|           |           |
|           4           8           4           8   |
| (p1 + p2 #1 + p3 #1) (q1 + q2 #1 + q3 #1)   |
+-                   +-+

```

where

$$\#1 == 1/4 - \frac{5}{4} + \frac{2}{4} + \frac{(5 + 5)}{4} i$$

$$\text{IDFT}(\text{DFT}(v) \cdot \star \text{DFT}(V))$$

PRODOTTO p*q

$$p_3^4 + (p_2 q_3 + p_3 q_2) x^3 + (p_1 q_3 + p_2 q_2 + p_3 q_1) x^2 +$$

$$(p_1 \ q_2 + p_2 \ q_1) \ x + p_1 \ q_1$$

Sesta propr.: UGUAGLIANZA DI PARSEVAL

Vettore v

[v1, v2, v3, v4]

DFT (v)

```

+-+-----+-----+
| v1 + v2 + v3 + v4 | |
| | |
| v1 - v2 i - v3 + v4 i | |
| | |
| v1 - v2 + v3 - v4 | |
| | |
| v1 + v2 i - v3 - v4 i | |
+-+-----+-----+

```

Norma2^2 DFT(v), N*Norma2^2 v

4.6 Quale relazione lega i coefficienti del polinomio trigonometrico interpolante ed una DFT? [liv.1]

Se i punti da interpolare sono equispaziati, i coefficienti del polinomio interpolante possono essere trovati in diverso modo. La matrice dei coefficienti del sistema di equazioni con $N+1$ punti di interpolazione è la seguente:

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & e^{(i\frac{2\pi}{N+1})^1} & e^{(i\frac{2\pi}{N+1})^2} & \dots & e^{(i\frac{2\pi}{N+1})^N} \\ 1 & e^{(i\frac{2\pi}{N+1})^2} & e^{(i\frac{2\pi}{N+1})^4} & \dots & e^{(i\frac{2\pi}{N+1})^{2N}} \\ 1 & e^{(i\frac{2\pi}{N+1})^3} & e^{(i\frac{2\pi}{N+1})^6} & \dots & e^{(i\frac{2\pi}{N+1})^{3N}} \end{pmatrix}$$

Tale matrice è la complessa coniugata della matrice Ω della DFT la cui principale proprietà è che il calcolo della matrice inversa è immediato.

Per una proprietà di Ω si ha:

$$\Omega_N^{-1} = \frac{1}{N} \cdot (\Omega_N)^H$$

dove $(\Omega_N)^H$ indica l'Hermitiana di Ω_N e N indica il numero totale dei punti. Siccome la matrice è simmetrica, l'hermitiana si calcola semplicemente come $\overline{\Omega_N}$.

Quindi:

$$\begin{aligned} \Omega_N^{-1} &= \frac{1}{N} \cdot \overline{\Omega_N} \\ \overline{\Omega_N} &= \Omega_N^{-1} \cdot N \end{aligned}$$

Il sistema di equazioni da risolvere è:

$$\begin{aligned} A \cdot c &= y \\ A &= \overline{\Omega_N} \\ A^{-1} &= \frac{1}{N} \cdot \Omega_N \end{aligned}$$

Infine:

$$c = \frac{1}{N} \cdot \Omega_N \cdot y = \frac{1}{N} \cdot DFT(y)$$

L'ultima equazione indica la relazione tra i coefficienti del polinomio interpolante e la DFT.

4.7 Ricostruire nell'intervallo $[0,3\pi]$ mediante interpolazione opportuna su 10,20 dati, la curva 3d di equazioni parametriche:

$$x = \cos(t), y = \sin(t), z = t$$

[liv.2]

```

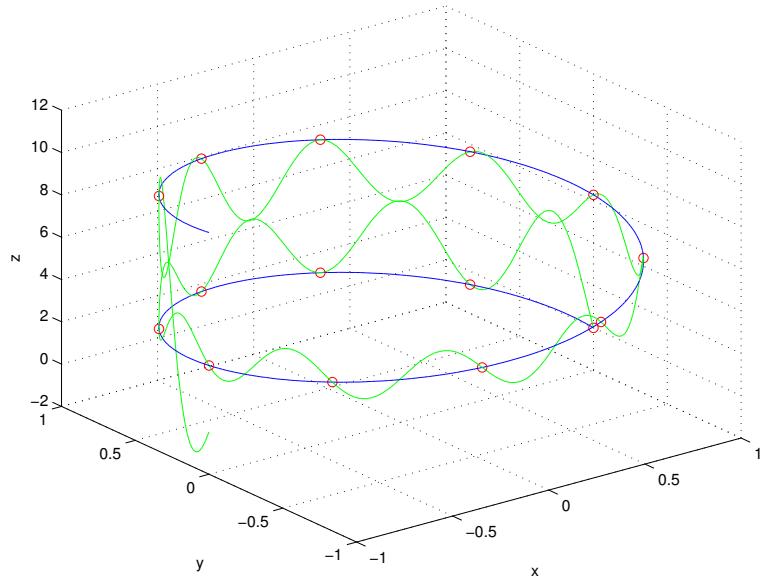
1 a=0;b=3*pi;
2 f=inline('unwrap(angle(z))');hold on;grid on;
3 t=linspace(a,b,401);
4 z=exp(i*t);
5 fz=f(z);
6 plot3(real(z),imag(z),fz);
7 N=15;
8 tk=linspace(a,b,N+1)'; %genero N punti equispaziati in [a b]
9 tk(end)=[];
10 zk=exp(i*tk);
11 fz_k=f(zk);
12 plot3(real(zk),imag(zk),fz_k,'or');
```

```

13 c=fft(fzk)/N;
14 Q=polyval(flipud(c),exp(i*2*pi*(t-a)/(b-a)));
15 plot3(real(z),imag(z),real(Q),'g');
16 view(3);
17 xlabel('x'); ylabel('y'); zlabel('z');

```

ESECUZIONE DEL PROGRAMMA



5 Serie e coefficienti di Fourier

5.1 Mediante il Symbolic Math Toolbox di Matlab costruire i coefficienti di Fourier in forma complessa

$$\gamma_k = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(t) e^{-ikt} dt$$

per le funzioni $t/2, t^2, \sin(t), \cos(t)$ nell'intervallo $[-\pi, \pi]$ e per ognuna visualizzare il grafico della funzione e:

- di tutte le ridotte fino all'ordine 41;
- delle ultime 6 ridotte.

$$S_{N+1}(x) = \sum_{k=-\frac{N}{2}}^{+\frac{N}{2}} \gamma_k e^{ikx}$$

[liv.2]

```

1 syms t real;
2 if ~exist('scelta', 'var')
3 scelta=input('Scegliere la funzione:');
4 end
5 switch scelta
6 case 1
7     f=t/2;
8 case 2
9     f=t.^2;
10 case 3
11     f=sin(t);
12 case 4
13     f=cos(t);
14 otherwise
15     disp('Inserimento errato');
16     exit
17 end
18 h=ezplot(f, [-pi pi]); %VISUALIZZO LA FUNZIONE IN [-PI,PI]
19 set(h,'LineWidth',4,'Color','k');
20 hold on;
21 grid on;
22 axis tight;
23 N=20;
24 k=(-N:N)';
25 cf=1/(2*pi)*int(f.*exp(-i*k*t),'t',-pi,pi);
26 %CALCOLO I COEFFICIENTI DI FOURIER IN FORMA COMPLESSA
27 S=sum(subs(cf).*exp(i*k*t));
28 %S E' LA RIDOTTA N-SIMA
29 ezplot(real(S), [-pi pi])
30 vett=subs(cf).*exp(i*k*t);
31 %IL VETTORE VETT CONTIENE TUTTE LE COMPONENTI DEL POLINOMIO TRIGONOMETRICO
32 %DI GRADO N. LA COMPONENTE CENTRALE DI TALE ARRAY(VETT(N+1)) E' GAMMA(0)
33 vetcum(1)=sum(vett(N:N+2));
34 %VETCUM(1)==VETT(N+1-1)+VETT(N+1)+VETT(N+2)==RIDOTTA ORD 2
35 for j=2:N
36     vetcum(j)=vetcum(j-1)+vett(N+1-j)+vett(N+1+j);
37 %IL VETTORE VETCUM E' IL VETTORE DELLE RIDOTTE.

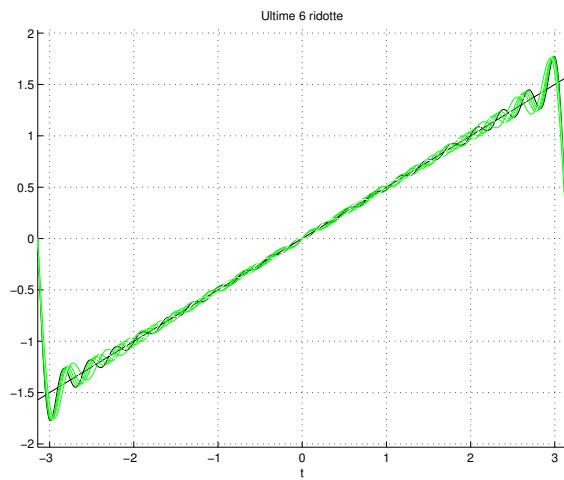
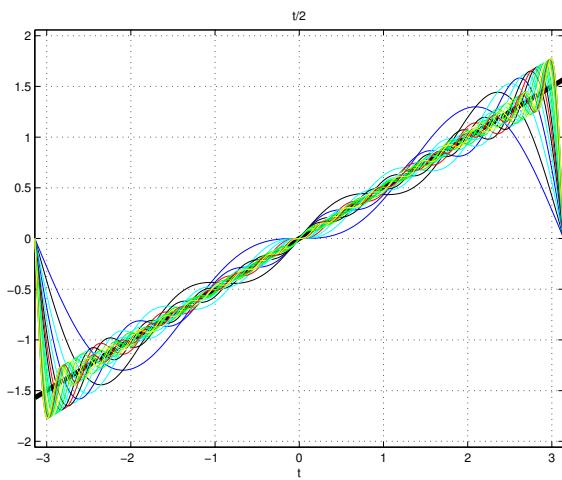
```

```

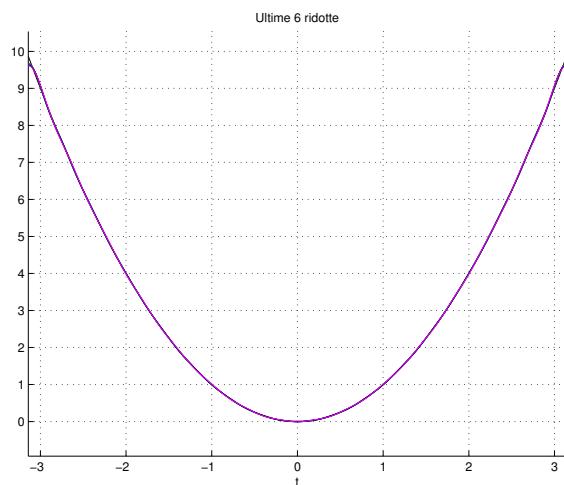
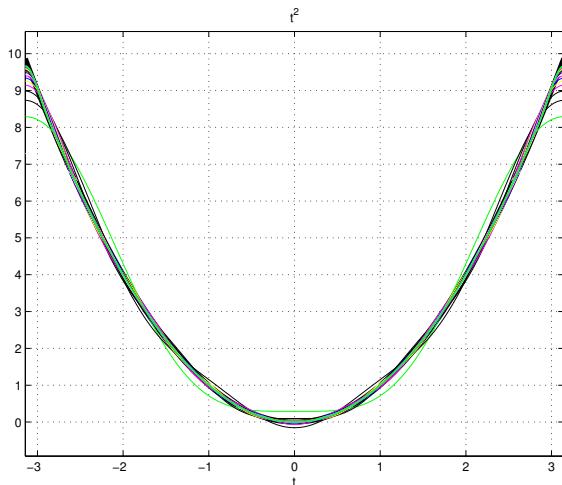
38 %VETCUM(2)==VETCUM(1)+VETT(N+1-2)+VETT(N+1+2)==RIDOTTA ORD 4...
39 h=ezplot(vetcum(j),[-pi,pi]);
40 set(h,'Color',randi(2,1,3)-1);%PRODUCO COLORI CASUALI
41 end
42 title(char(f));
43 figure;grid on;axis tight;hold on;
44 h=ezplot(f,[-pi pi]);
45 set(h,'Color','k');
46 for j=0:5
47 h=ezplot(vetcum(numel(vetcum)-j),[-pi,pi]);
48 set(h,'Color',randi(2,1,3)-1);%PRODUKO COLORI CASUALI
49 end
50 title('Ultime 6 ridotte');

```

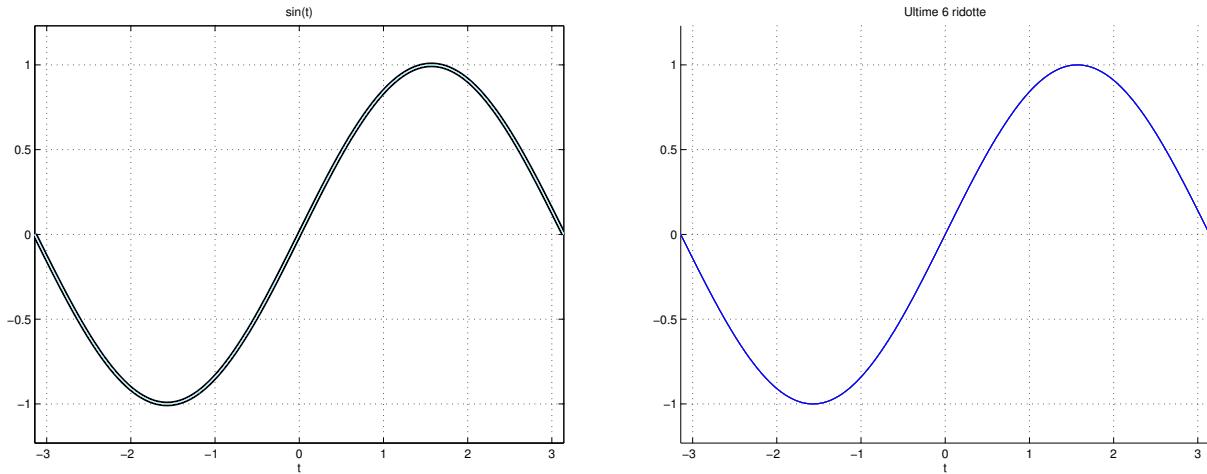
ESECUZIONE DEL PROGRAMMA(SCELTA=1, $f(t) = t/2$)



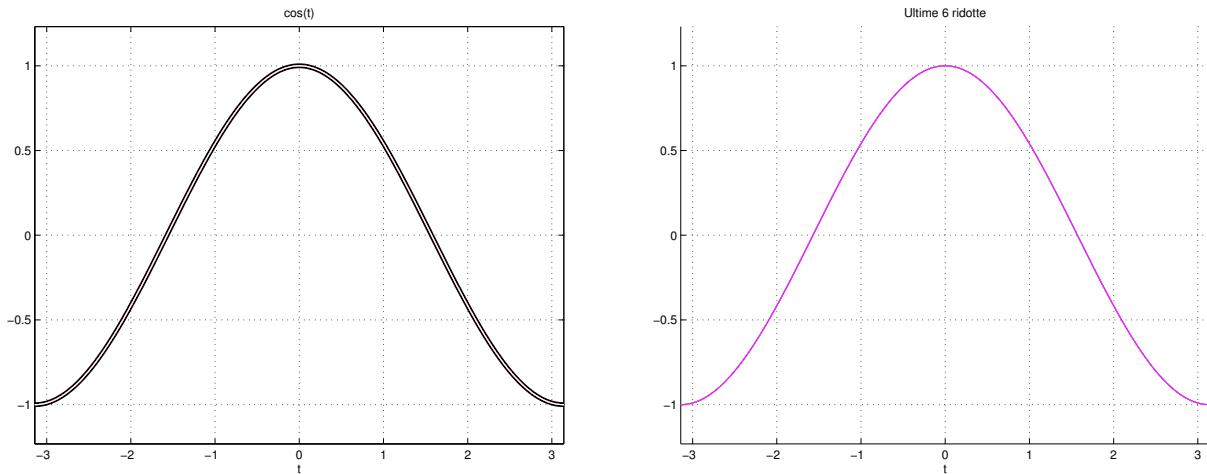
ESECUZIONE DEL PROGRAMMA(SCELTA=2, $f(t) = t^2$)



ESECUZIONE DEL PROGRAMMA(SCELTA=3, $f(t) = \sin(t)$)



ESECUZIONE DEL PROGRAMMA(SCELTA=4, $f(t) = \cos(t)$)



5.2 Unire i due riquadri MATLAB per costruire una function

function[y] = period_repeat(fun, a, b, x)

che restituisca il valore in x della ripetizione periodica della restrizione all'intervallo $]a, b[$ di fun.**[liv.2]**

```

1 function y = period_repeat(fun,a,b,x)
2 fprintf('fun=%s,a=%d,b=%d,x=%d',char(fun),a,b,x);
3 while (x>b)
4     x=x-(b-a);
5 end
6 while (x<a)
7     x=x+(b-a);
```

```

8 end
9 y=fun(x);
10 end

```

ESECUZIONE DEL PROGRAMMA

```

fun=t/2, a=-1, b=1, x=3.600000e+00
ans =
-0.2000

```

- 5.3 Mediante il Symbolic Math Toolbox di Matlab costruire i coefficienti di Fourier in forma complessa per le funzioni $t/2, t^2, \sin(t), \cos(t)$ in un intervallo generico $[a, b]$, stabilito in input, e per ognuna visualizzare, nell'intervallo $[a - 2(b - a), b + 2(b - a)]$ il grafico della funzione e delle ridotte della sua Serie di Fourier fino all'ordine max 40. [liv.2]**

```

1 syms t real;
2 if ~exist('scelta', 'var')
3 scelta=input('Scegliere la funzione:');
4 end
5 switch scelta
6 case 1
7 f=t/2;
8 case 2
9 f=t.^2;
10 case 3
11 f=sin(t);
12 case 4
13 f=cos(t);
14 otherwise
15 disp('Inserimento errato');
16 exit
17 end
18 if ~exist('a', 'var')
19 a=input('Inserire il valore di a:');
20 end
21 if ~exist('b', 'var')
22 b=input('Inserire il valore di b:');
23 end
24 h=ezplot(f, [a-2*(b-a), b+2*(b-a)]);
25 %VISUALIZZO LA FUNZIONE NELL'INTERVALLO RICHIESTO
26 set(h,'LineWidth',4,'Color','k');
27 hold on;
28 grid on;
29 axis tight;
30 N=20;
31 k=(-N:N)';
32 cf=1/(b-a)*int(f.*exp(-i*k*((t-a)/(b-a)*2*pi-pi)), 't', a, b);
33 %CALCOLO I COEFFICIENTI DI FOURIER IN FORMA COMPLESSA
34 %PROIETTANDO L'INTERVALLO [A,B] IN [-PI,PI]
35 vett=subs(cf).*exp(i*k*((t-a)/(b-a)*2*pi-pi));
36 %IL VETTORE VETT CONTIENE TUTTE LE COMPONENTI DEL POLINOMIO TRIGONOMETRICO
37 %DI GRADO N. LA COMPONENTE CENTRALE DI TALE ARRAY (VETT(N+1) E' GAMMA(0)

```

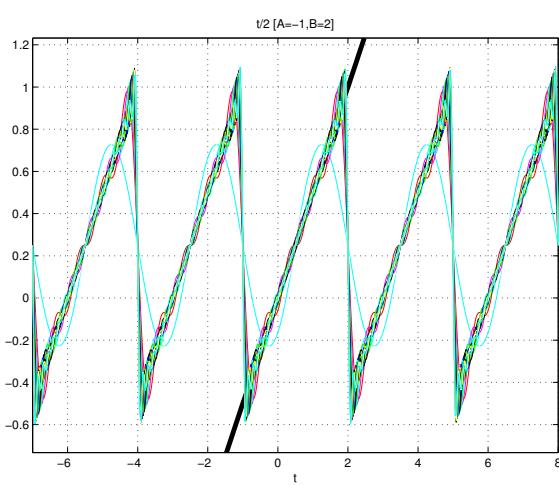
```

38 vetcum(1)=sum(vett(N:N+2));
39 %VETCUM(1)==VETT(N+1-1)+VETT(N+1)+VETT(N+2)==RIDOTTA ORD 2
40 for j=2:N
41     vetcum(j)=vetcum(j-1)+vett(N+1-j)+vett(N+1+j);
42     %IL VETTORE VETCUM E' IL VETTORE DELLE RIDOTTE.
43     %VETCUM(2)==VETCUM(1)+VETT(N+1-2)+VETT(N+1+2)==RIDOTTA ORD 4...
44     h=ezplot(vetcum(j),[a-2*(b-a),b+2*(b-a)]);
45     set(h,'Color',randi(2,1,3)-1);%PRODUCO COLORI CASUALI
46 end
47 s=sprintf(' %s [A=%d,B=%d]',char(f),a,b);
48 title(s);

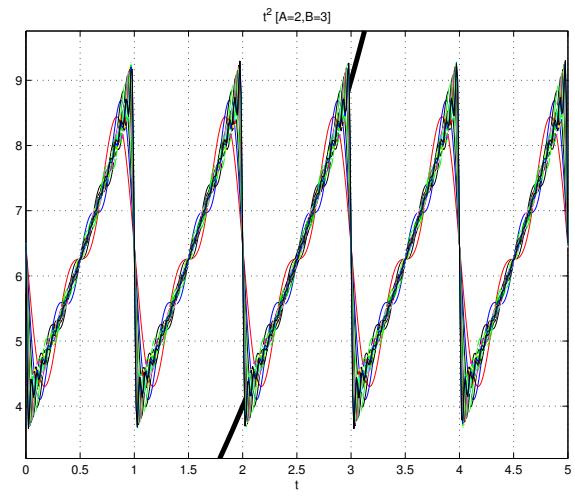
```

ESECUZIONE DEL PROGRAMMA

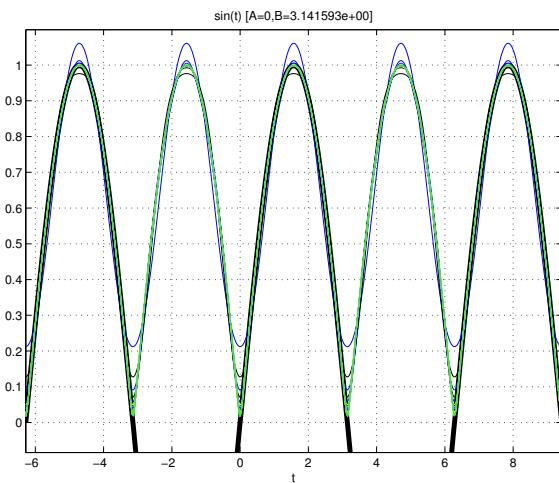
(SCELTA=1, $f(t) = t/2$)



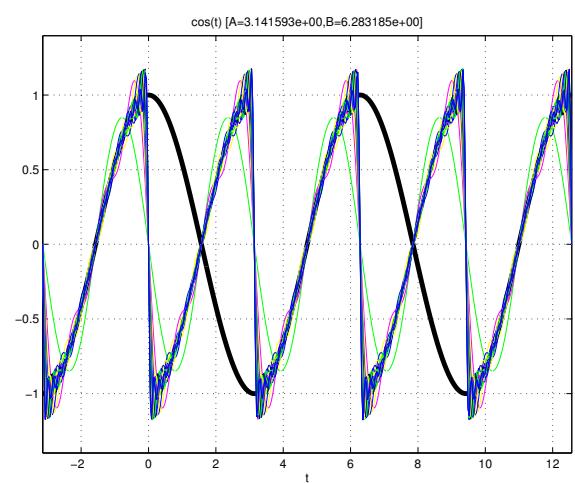
(SCELTA=2, $f(t) = t^2$)



(SCELTA=3, $f(t) = \sin(t)$)



(SCELTA=4, $f(t) = \cos(t)$)



- 5.4 Approssimare numericamente nell'intervallo $[-T/2, T/2]$ alcune funzioni $f(x)$ mediante ridotta di ordine N della sua serie di Fourier costruendo i coefficienti di Fourier sia mediante integrazione numerica (function quad()) sia mediante DFT (functions fft() ed fftshift()) e confrontare i risultati ottenuti al variare di T e N.[liv.1]

```

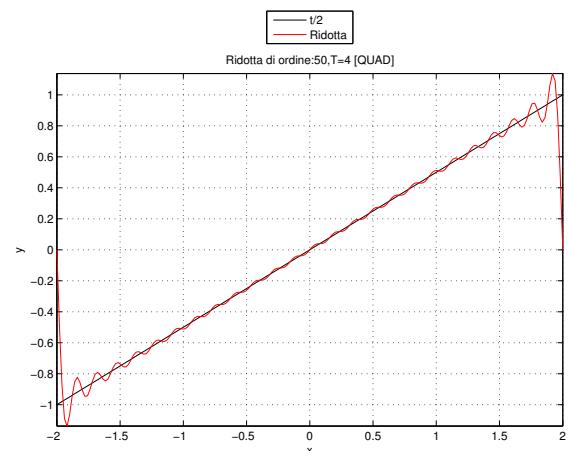
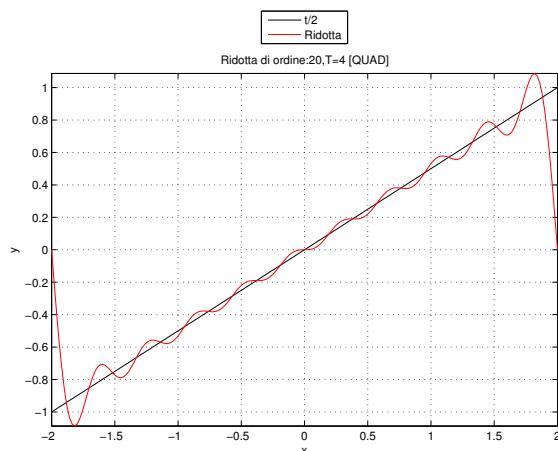
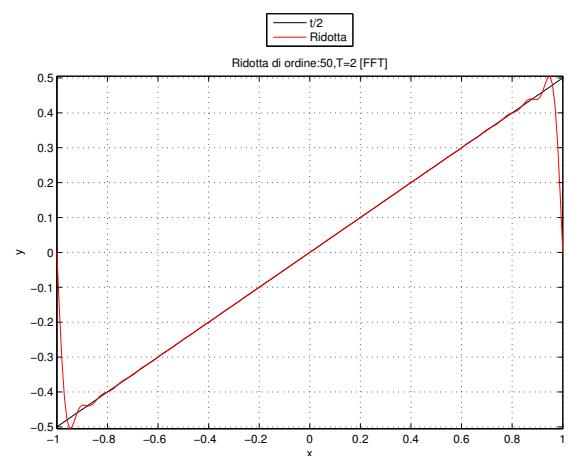
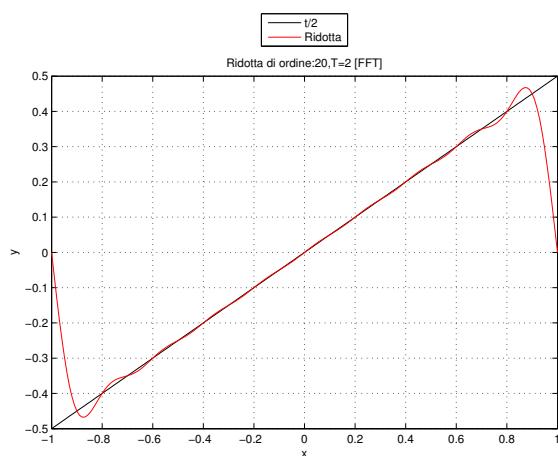
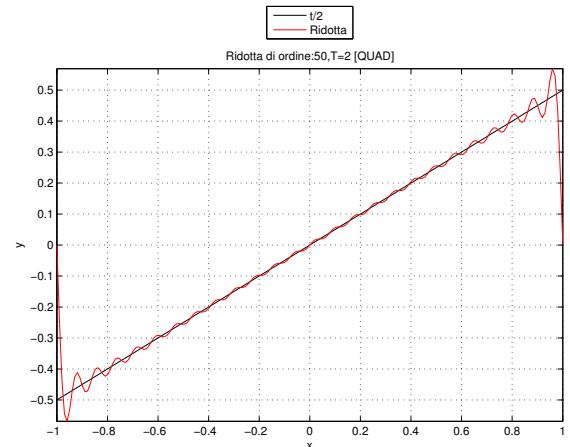
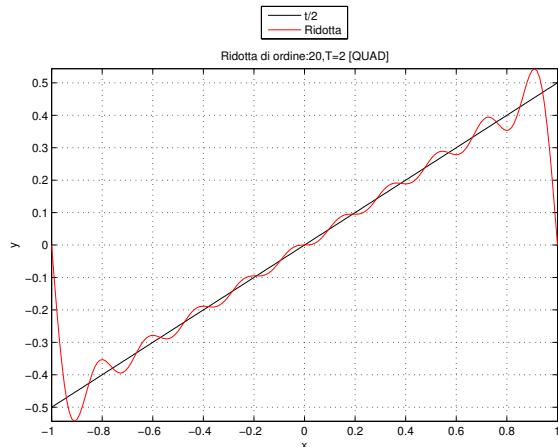
1 f=inline('t/2');
2 for T=[2,4]
3     x=linspace(-T/2,T/2,199);
4     y=f(x);
5     f2=inline('x./2.*exp(-i*2*k*pi/T*x)', 'x', 'T', 'k');
6     for N=[20 50]
7         c=[];
8         val_f=0;
9         for k=-N/2:N/2
10             [Q,count]=quad(f2,-T/2,T/2,[],[],T,k);
11             val_f=val_f+count;
12             c=[c;Q/T];
13         end
14         S=exp(-i*N*pi/T*x).*polyval(flipud(c),exp(i*2*pi/T*x));
15         fprintf('Num. di valutazioni della funzione integranda:
16 %d [N=%d, T=%d]\n',val_f,N,T);
17         figure
18         plot(x,y,'k',x,real(S),'r');
19         xlabel('x');ylabel('y');grid on;axis tight;
20         str=sprintf('Ridotta di ordine:%d, T=%d [QUAD]',N,T);
21         legend(char(f),'Ridotta','Location','NorthOutside');
22         title(str);
23     end
24 %-----%
25 %APPROSSIMAZIONE MEDIANTE DFT
26 for N=[20 50]
27     c=[];
28     tt=linspace(-T/2,T/2,N+1)';
29     ftt=f(tt);
30     v=[(ftt(1)+ftt(end))/2;ftt(2:end-1)];
31     c=fftshift(fft(v));
32     c=[c;c(1)]/N;
33     if mod(N/2,2)==0
34         %se N/2 e' pari parto dal secondo elem a cambiare segno
35         c(2:2:end)=-c(2:2:end);
36     else
37         c(1:2:end)=-c(1:2:end);
38     end
39     S=exp(-i*N*pi/T*x).*polyval(flipud(c),exp(i*2*pi/T*x));
40     figure
41     plot(x,y,'k',x,real(S),'r');
42     xlabel('x');ylabel('y');grid on;axis tight;
43     str=sprintf('Ridotta di ordine:%d, T=%d [FFT]',N,T);
44     legend(char(f),'Ridotta','Location','NorthOutside');
45     title(str);
46 end
end

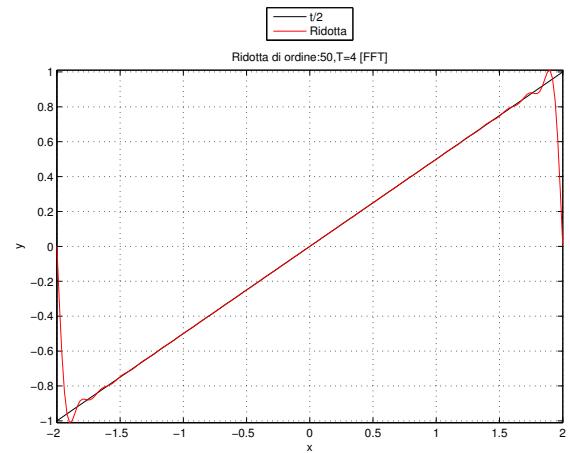
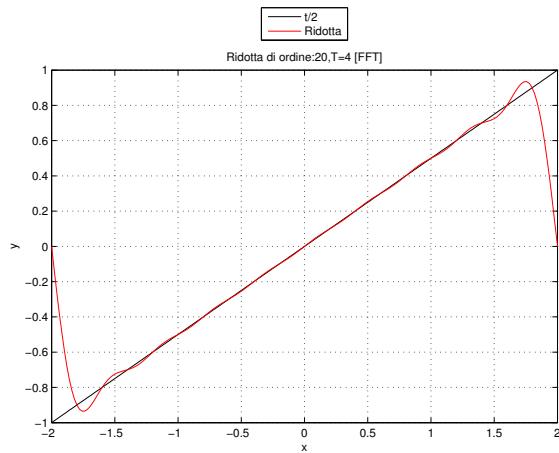
```

Num. di valutazioni della funzione integranda:3073 [N=20, T=2]
Num. di valutazioni della funzione integranda:14815 [N=50, T=2]

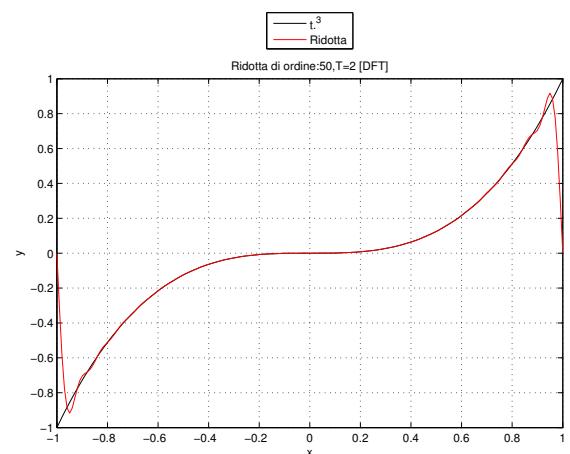
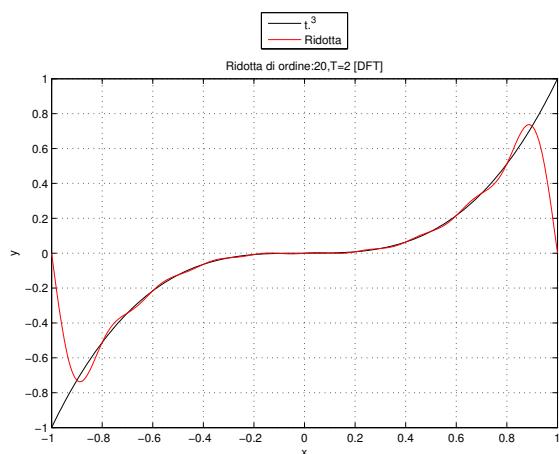
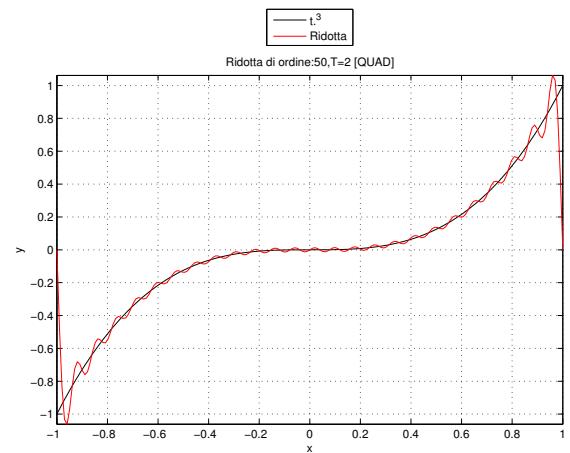
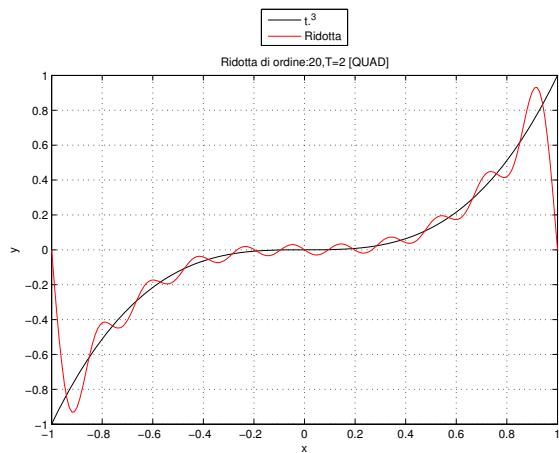
Num. di valutazioni della funzione integranda: 4161 [N=20, T=4]
 Num. di valutazioni della funzione integranda: 19887 [N=50, T=4]

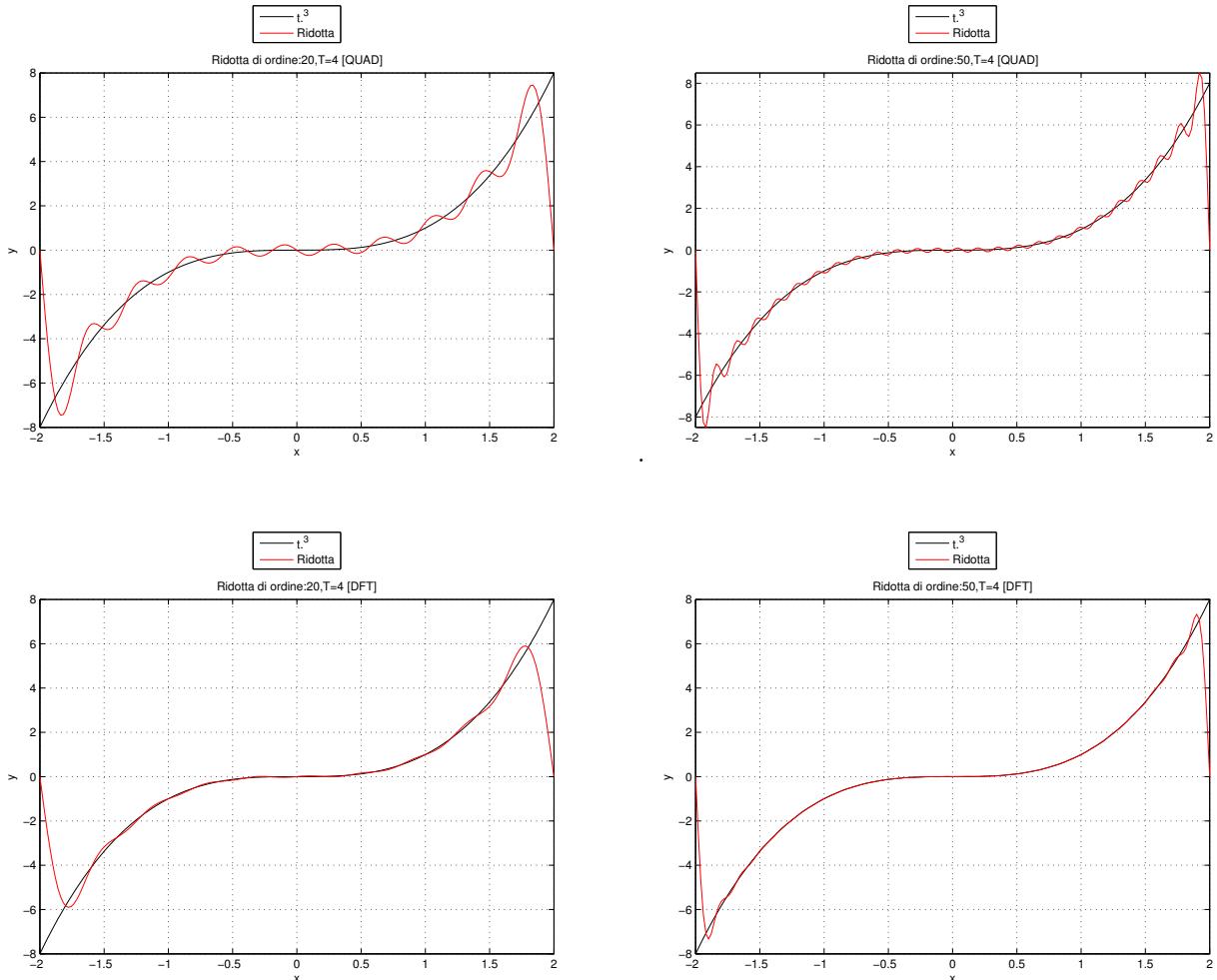
ESECUZIONE DEL PROGRAMMA (SCELTA=1, $f(t) = t/2$)





ESECUZIONE DEL PROGRAMMA (SCELTA=2, $f(t) = t^3$)





5.5 Ripetere l'esercizio precedente supponendo di avere a disposizione solo n campioni della funzione da approssimare. [liv.2]

```

1 f=inline('t/2');
2 for T=[2,4]
3 x=linspace(-T/2,T/2,199);
4 %Siccome non possiedo l'espressione della funzione,interpolo i
5 %campioni con una spline cubica,in modo da poter calcolare
6 %successivamente l'integrale richiesto per i coefficienti di Fourier
7 for N=[20 50]
8 tt=linspace(-T/2,T/2,N+1)';%campioni della funzione
9 ftt=f(tt);
10 yspline=spline(tt,ftt);
11 c=[];
12 val_f=0;
13 for k=-N/2:N/2
14 [Q,count]=quad(@(x)ppval(yspline,x).*exp(-i*2*k*pi/T*x),-T/2,T/2);
15 c=[c;Q/T];val_f=val_f+count;
16 end
17 S=exp(-i*N*pi/T*x).*polyval(flipud(c),exp(i*2*pi/T*x));
18 figure
19 plot(x,ppval(yspline,x),'k',x,real(S),'r');
```

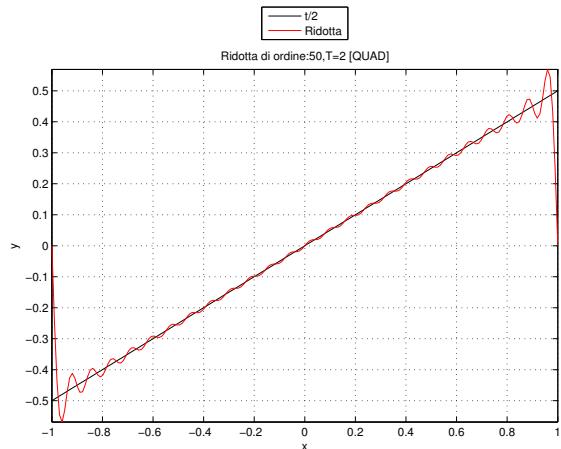
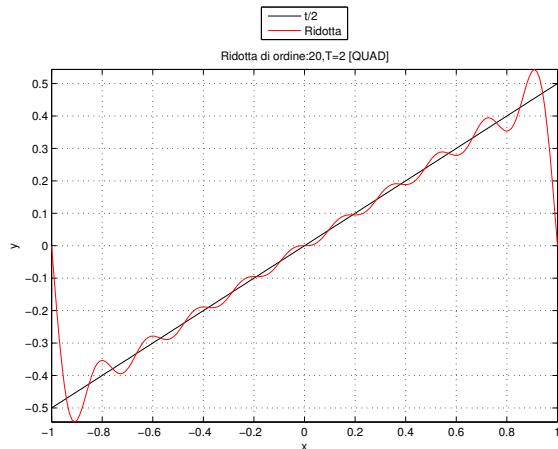
```

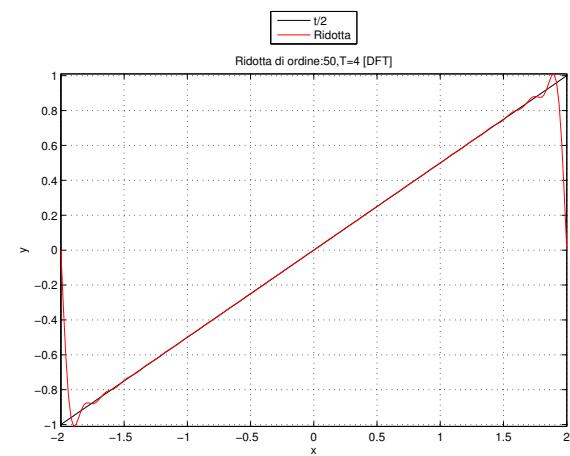
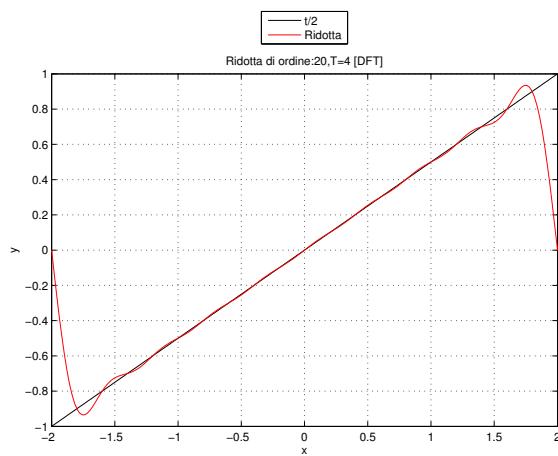
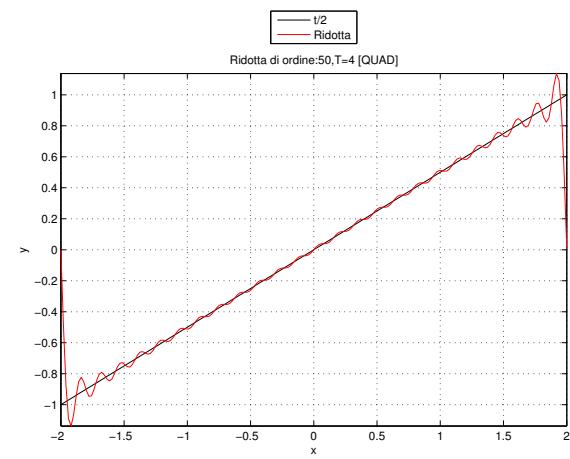
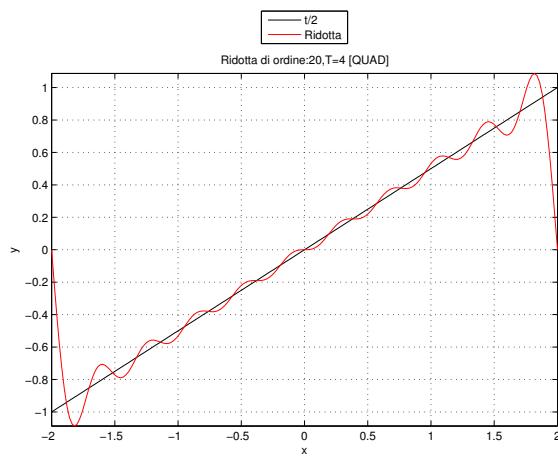
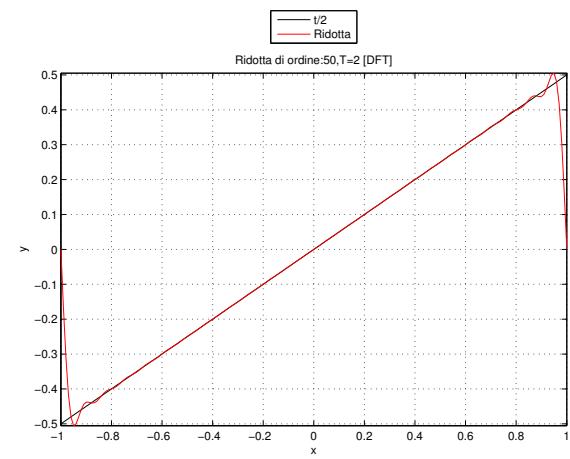
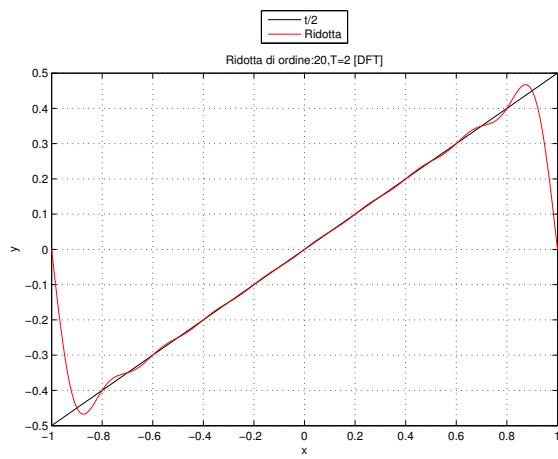
20 xlabel('x'); ylabel('y'); grid on; axis tight;
21 str=sprintf('Ridotta di ordine:%d,T=%d [QUAD]',N,T);
22 title(str);
23 legend(char(f),'Ridotta','Location','NorthOutside');
24 fprintf('Numero di valutazioni della funzione integranda:%d\n',val_f);
25 end
%-----
26 %APPROXIMAZIONE MEDIANTE DFT
27 for N=[20 50]
28 c=[];
29 tt=linspace(-T/2,T/2,N+1)';
30 ftt=f(tt);
31 v=[(ftt(1)+ftt(end))/2;ftt(2:end-1)];
32 c=fftshift(fft(v));
33 c=[c;c(1)]/N;
34 if mod(N/2,2)==0
35 %se N/2 e' pari parto dal secondo elem a cambiare segno
36 c(2:2:end)=-c(2:2:end);
37 else
38 c(1:2:end)=-c(1:2:end);
39 end
40 S=exp(-i*N*pi/T*x).*polyval(flipud(c),exp(i*2*pi/T*x));
41 figure
42 plot(x,ppval(yspline,x),'k',x,real(S),'r');
43 xlabel('x'); ylabel('y'); grid on; axis tight;
44 str=sprintf('Ridotta di ordine:%d,T=%d [DFT]',N,T);
45 title(str);
46 legend(char(f),'Ridotta','Location','NorthOutside');
47 end
48 end
49 end

```

Numero di valutazioni della funzione integranda:3073
 Numero di valutazioni della funzione integranda:14815
 Numero di valutazioni della funzione integranda:4161
 Numero di valutazioni della funzione integranda:19887

ESECUZIONE DEL PROGRAMMA(SCELTA=1, $f(t) = t/2$)





- 5.6 Approssimare numericamente nell'intervallo $[-T/2, T/2]$ alcune funzioni $f(x)$ mediante ridotta di ordine N della sua serie di Fourier. Per l'errore di discretizzazione in funzione di N usare le seguenti misure:

$$E_1(N) = \max |f(x) - FS_n(x)|, x \in \left[-\frac{T}{2}, +\frac{T}{2}\right]$$

$$E_2(N) = \int_{-T/2}^{+T/2} |f(x) - FS_n(x)|^2 dx$$

Visualizzare al variare di $N, E_1(N)$ e $E_2(N)$, costruendo i coefficienti di Fourier sia mediante integrazione numerica (function quad()) sia mediante DFT (function fft() ed fftshift()) e confrontare i risultati ottenuti al variare di T e N . [liv.3]

```

1 f=inline('t/2');
2 for T=[2,4]
3     x=linspace(-T/2,T/2,199);
4     y=f(x);
5     f2=inline('x./2.*exp(-i*2*k*pi/T*x)', 'x', 'T', 'k');
6     err1=[];
7     err2=[];
8     for N=2:2:80
9         c=[];
10        for k=-N/2:N/2
11            Q=quad(f2,-T/2,T/2,[],[],T,k);
12            c=[c;Q/T];
13        end
14        S=exp(-i*N*pi/T*x).*polyval(flipud(c),exp(i*2*pi/T*x));
15        err1=[err1;N,max(abs(y-S))];%NORMA DEL MASSIMO
16        %ERR1 SARA' UNA MATRICE CHE CONTERRA' NELLA PRIMA COLONNA IL VALORE DI
17        %N E NELLA SECONDA COLONNA IL MASSIMO IN VALORE ASSOLUTO TRA F(X)-S(X)
18        %DOVE S(X) E' LA RIDOTTA DELLA SERIE DI FOURIER.
19        e=(y-S).^2;
20        %NORMA EUCLIDEA, VOGLIO CALCOLARE L' INTEGRALE DA -T/2 A T/2 DELLA
21        %FUNZIONE DESCRITTA DA e. INTERPOLO I PUNTI (X,e) CON UNA SPLINE
22        %CUBICA E NE CALCOLO L' INTEGRALE DA -T/2 A T/2
23        yspline=spline(x,e);
24        Q=quad(@(x)ppval(yspline,x),-T/2,T/2);
25        err2=[err2;N,Q];
26        %SALVO IN Err2 LE COPPIE [N,ERRORE]
27    end
28    %VISUALIZZO L'ULTIMA RIDOTTA
29    figure;
30    plot(x,y,'k',x,real(S),'r');
31    xlabel('x'); ylabel('y'); grid on; axis tight;
32    str=sprintf('Ridotta di ordine:%d [QUAD], T=%d', N, T);
33    title(str);
34    %VISUALIZZO I GRAFICI DELL'ERRORE IN FUNZIONE DI N
35    figure
36    plot(err1(:,1),real(err1(:,2)));
37    str=sprintf('Errore [norma del massimo, QUAD, N=%d, T=%d]', N, T);
38    title(str);
39    xlabel('N'); ylabel('Err'); grid on; axis tight;
40    figure
41    plot(err2(:,1),real(err2(:,2)));

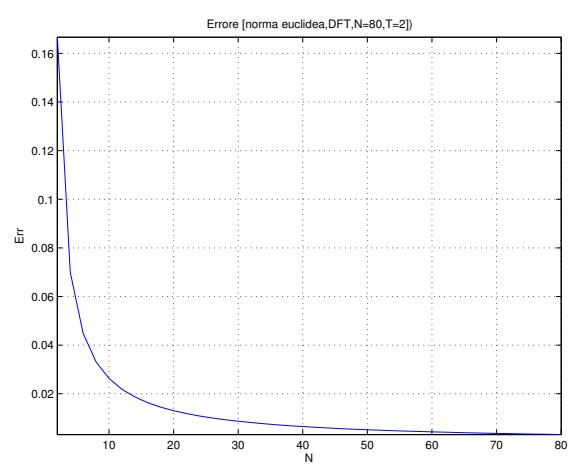
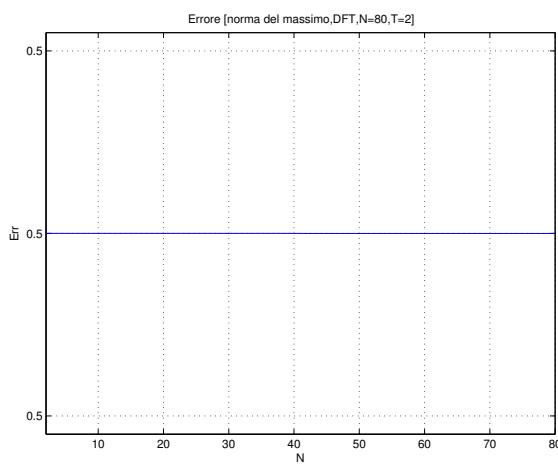
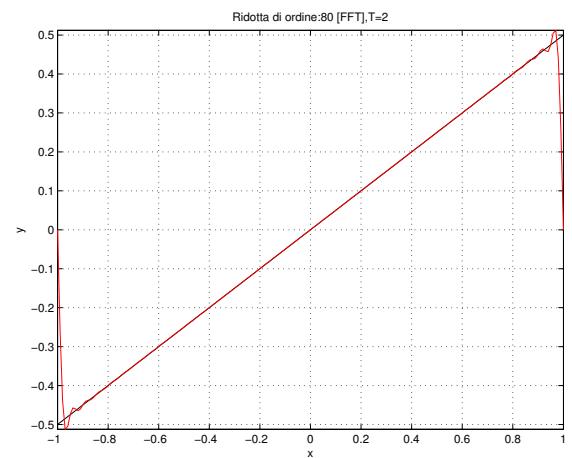
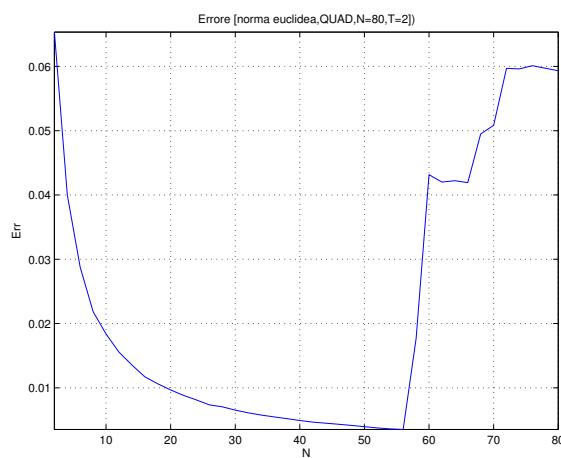
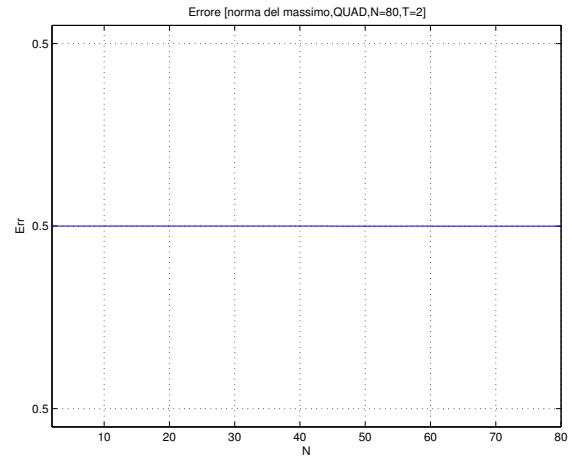
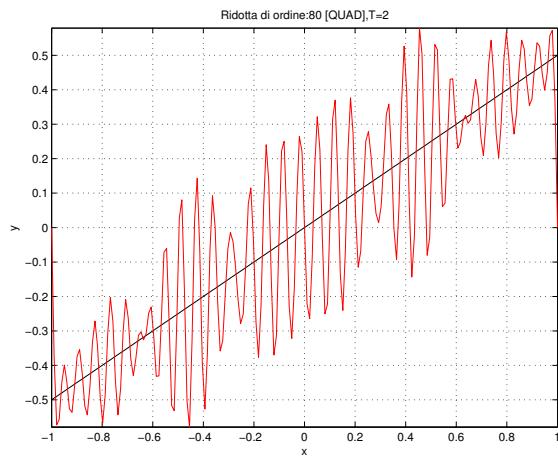
```

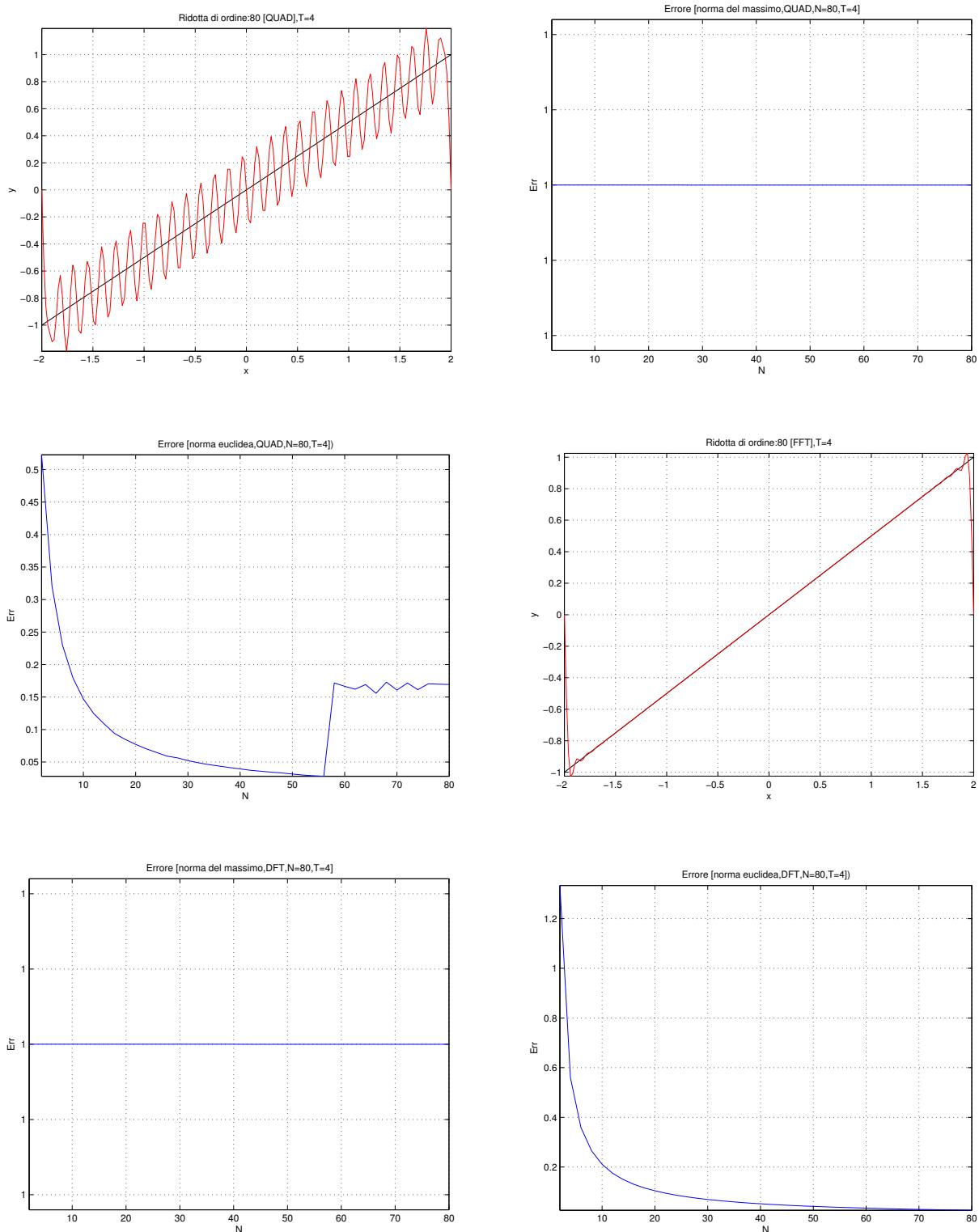
```

42 str=sprintf('Errore [norma euclidea,QUAD,N=%d,T=%d]',N,T);
43 title(str);
44 xlabel('N'); ylabel('Err'); grid on; axis tight;
45 %-----
46 %APPROXIMAZIONE MEDIANTE DFT
47 Err1=[]; Err2=[];
48 for N=2:2:80
49     c=[];
50     tt=linspace(-T/2,T/2,N+1)';
51     ftt=f(tt);
52     v=[(ftt(1)+ftt(end))/2;ftt(2:end-1)];
53     c=fftshift(fft(v));
54     c=[c;c(1)]/N;
55     if mod(N/2,2)==0
56         %se N/2 e' pari parto dal secondo elem a cambiare segno
57         c(2:2:end)=-c(2:2:end);
58     else
59         c(1:2:end)=-c(1:2:end);
60     end
61     S=exp(-i*N*pi/T*x).*polyval(flipud(c),exp(i*2*pi/T*x));
62     Err1=[Err1;N,max(abs(y-S))];
63     e=(y-S).^2;
64     yspline=spline(x,e);
65     Q=quad(@(x)ppval(yspline,x),-T/2,T/2);
66     Err2=[Err2;N,Q];
67 end
68 %VISUALIZZO L'ULTIMA RIDOTTA
69 figure
70 plot(x,y,'k',x,real(S),'r');
71 xlabel('x'); ylabel('y'); grid on; axis tight;
72 str=sprintf('Ridotta di ordine:%d [FFT],T=%d',N,T);
73 title(str);
74 %VISUALIZZO I GRAFICI DELL'ERRORE IN FUNZIONE DI N
75 figure
76 plot(Err1(:,1),real(Err1(:,2)));
77 str=sprintf('Errore [norma del massimo,DFT,N=%d,T=%d]',N,T);
78 title(str);
79 xlabel('N'); ylabel('Err'); grid on; axis tight;
80 figure
81 plot(Err2(:,1),real(Err2(:,2)));
82 str=sprintf('Errore [norma euclidea,DFT,N=%d,T=%d]',N,T);
83 title(str);
84 xlabel('N'); ylabel('Err'); grid on; axis tight;
85 end

```

ESECUZIONE DEL PROGRAMMA ($f(t) = t/2$)

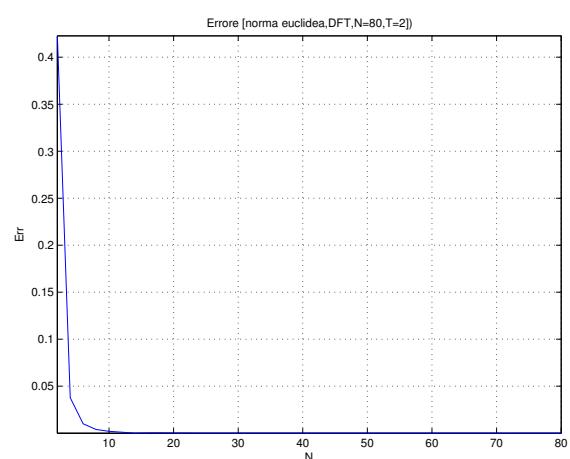
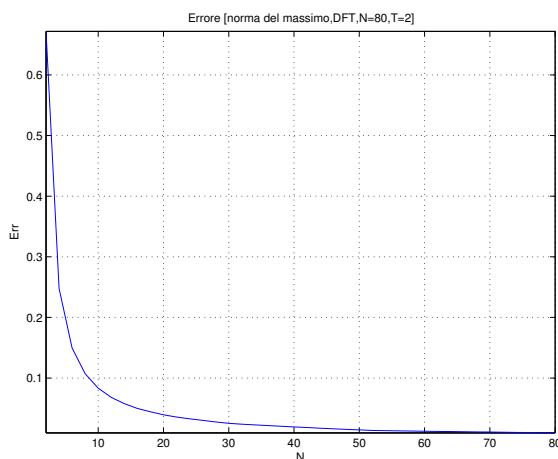
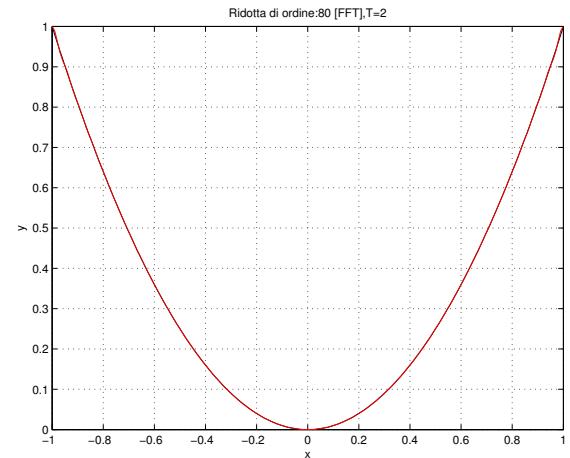
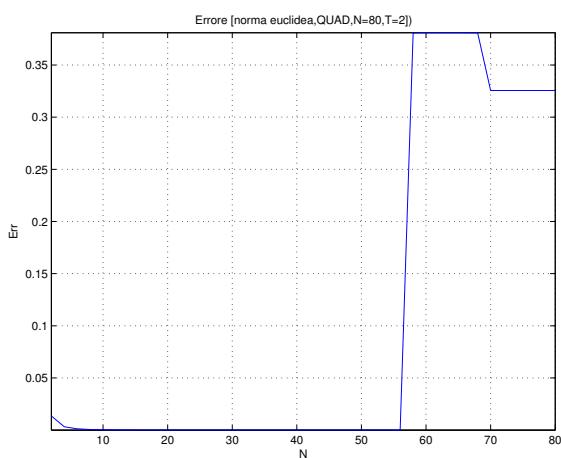
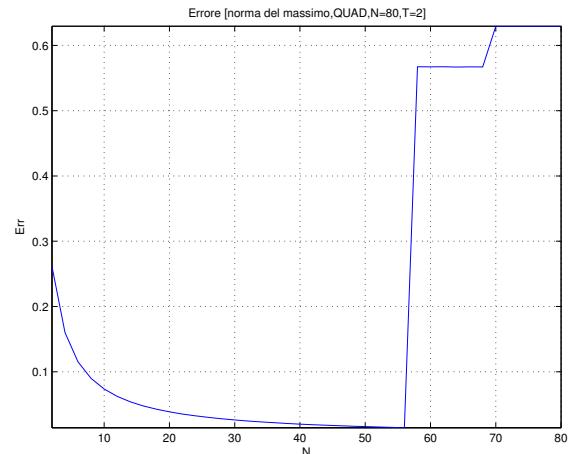
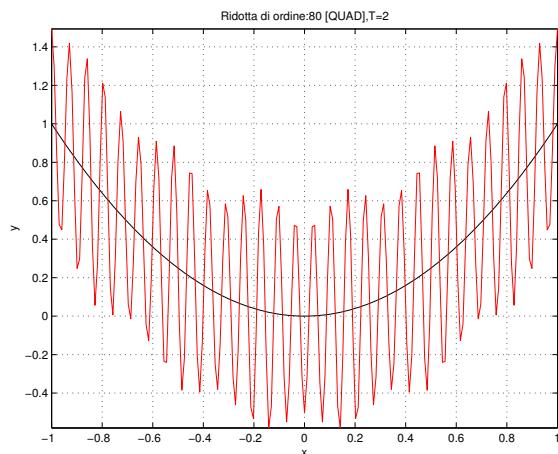


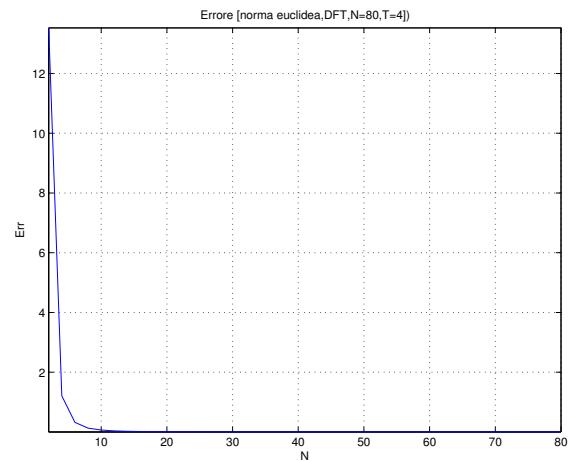
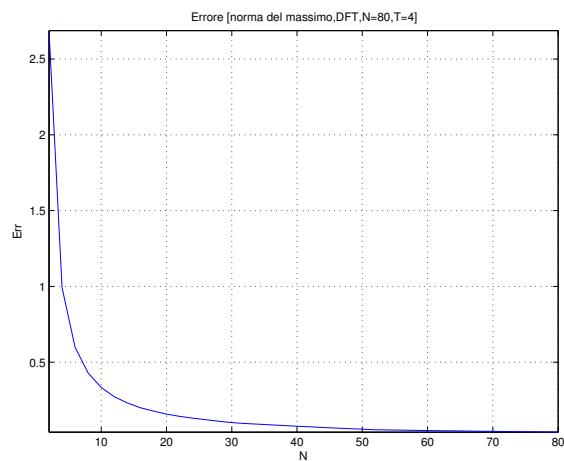
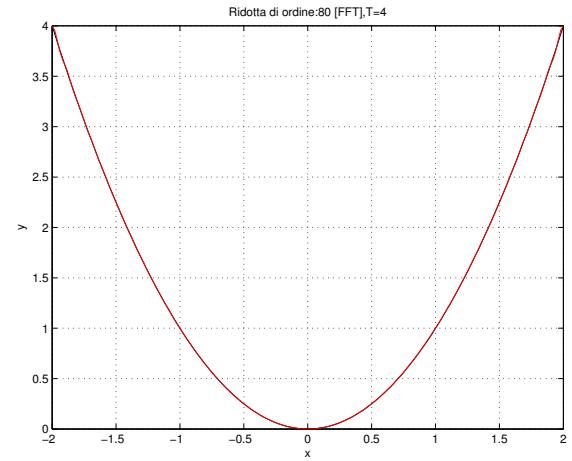
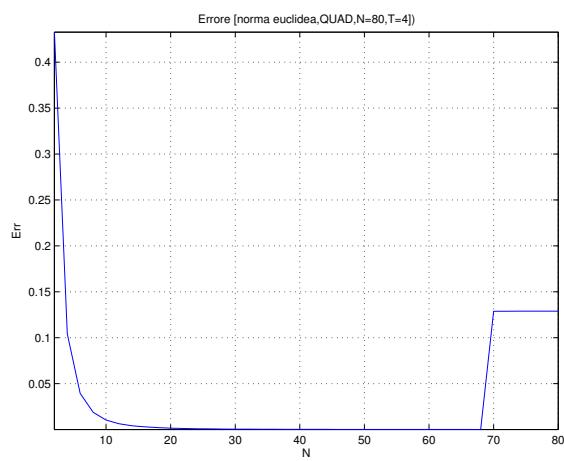
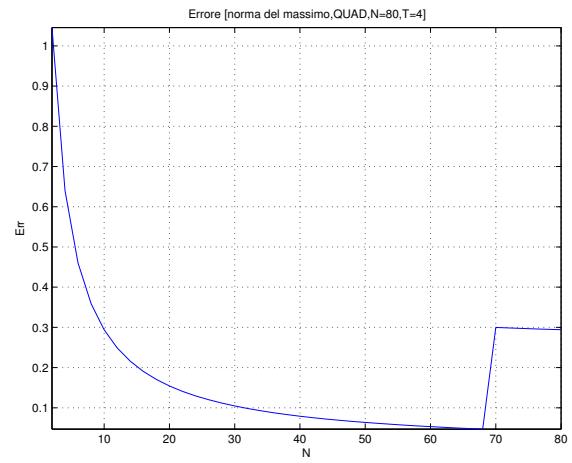
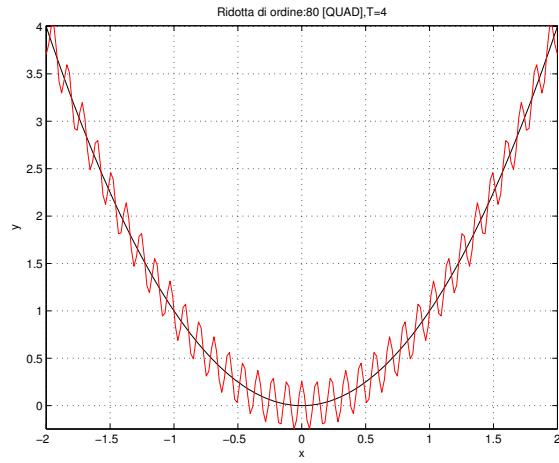


Come si osserva dall'esecuzione del programma $E_1(N)$ e' costante sia nel caso dell'utilizzo di formule di quadratura, sia ricorrendo alla DFT, in quanto agli estremi dell'intervallo $[-T/2, T/2]$ la funzione $T/2$ presenta valori uguali ma di segno opposto, quindi anche al crescere dell'ordine della ridotta della serie di Fourier, in tali estremi le ridotte passeranno sempre per il punto $(f(a) + f(b))/2$, ossia 0. Quindi anche se le ridotte convergono uniformemente alla funzione, in tali estremi l'errore sara' costante e pari a $|f(a)/2|$. Si puo' in generale notare che la qualita' dell'approssimazione di $f(x)$ e' molto piu' accurata utilizzando l'algoritmo della DFT, che non soffre dell'errore di round-off della QUAD come si osserva dai grafici, ed e' in generale molto piu' veloce di quest'ultima. Nelle immagini a seguire verrà eseguito lo stesso programma con

la funzione x^2 .

ESECUZIONE DEL PROGRAMMA ($f(t) = t^2$)





6 Trasformata di Fourier e applicazioni

6.1 Derivare l'algoritmo per l'antitrasformata.[liv.3]

Anche per quanto riguarda l'antitrasformata di Fourier si puo' ripetere lo stesso ragionamento applicato per la trasformata, con alcune differenze. L'antitrasformata di Fourier é definita come:

$$f(t) = \int_{-\infty}^{+\infty} F(\nu) e^{2\pi i \nu t} d\nu$$

Possiamo troncare l'integrale:

$$f(t) = \int_{-\frac{\Omega}{2}}^{+\frac{\Omega}{2}} F(\nu) e^{2\pi i \nu t} d\nu$$

dove:

$$\Omega = \frac{N}{T}$$

e applicare la trapezoidale composita:

$$f(t) = \frac{1}{T} \left\{ \frac{1}{2} \left[F\left(-\frac{\Omega}{2}\right) e^{-2\pi i \frac{\Omega}{2} t} + F\left(\frac{\Omega}{2}\right) e^{2\pi i \frac{\Omega}{2} t} \right] + \sum_{j=1}^{N-1} F(\nu_j) e^{2\pi i t \left(-\frac{\Omega}{2} + \frac{\Omega}{N} j\right)} \right\}$$

Se si valuta f in $t_k = \frac{k}{\Omega}$, $k = -N/2 \dots N/2$, otteniamo:

$$f(t_k) = \frac{1}{T} \left\{ \frac{1}{2} \left[F\left(-\frac{\Omega}{2}\right) e^{-\pi i k} + F\left(\frac{\Omega}{2}\right) e^{\pi i k} \right] + \sum_{j=1}^{N-1} F(\nu_j) e^{-\pi i k + \frac{2\pi i}{N} (kj)} \right\}$$

A questo punto possiamo costruire il vettore \underline{f} in questo modo:

$$\underline{f}(0) = \frac{1}{2} \left[F\left(-\frac{\Omega}{2}\right) + F\left(\frac{\Omega}{2}\right) \right], \underline{f}(j) = F(\nu_j) \rightarrow j = 1, \dots, N-1$$

Quindi:

$$f(t_k) = \frac{(-1)^k}{T} \sum_{j=0}^{N-1} \underline{f}_j \cdot e^{\frac{2\pi i}{N} (kj)} \rightarrow k = -N/2, \dots, N/2 - 1$$

E la componente della sommatoria e' pari a:

$$IDFT(\underline{f}) \cdot N$$

Quindi i passi dell'algoritmo sono:

- Prendo un numero dispari di campioni equispaziati della trasformata di Fourier;
- Costruisco il vettore \underline{f} nel modo indicato precedentemente;
- Calcolo e riordino la $IDFT(\underline{f})$ mediante una $FFTSIFT$;
- Aggiungo l'ultima componente;
- Infine aggiungo il fattore moltiplicativo

$$\frac{(-1)^k}{T} \cdot N$$

- 6.2 -Al variare di $n=16,32,64$ come ottenere gli $N+1$ campioni $F(\nu)$ della trasformata di Fourier tutti nello stesso intervallo $[-\Omega/2, \Omega/2]$? Quali considerazioni possono farsi...[liv.2]
 -Approssimare numericamente in $[-T/2, T/2]$, la trasformata di Fourier di alcune funzioni $f(x)$. Visualizzare i risultati sia rispetto alla frequenza circolare che rispetto a quella angolare, confrontandoli quando possibile con la trasformata analitica e commentandoli al variare dei parametri di discretizzazione T e N .[liv.1]
 -Ripetere l'esercizio con l'antitrasformata.[liv.2]

```

1 syms t v w real;
2 if ~exist('scelta', 'var')
3     scelta=input('Scegliere la funzione:');
4 end
5 switch scelta
6     case 1
7         fun=inline('exp(-2.*abs(t))');
8         fun2=exp(-2*abs(t));
9     case 2
10        fun=@sinc;
11        fun2=sin(pi*t)/(pi*t);
12    case 3
13        fun=@sign;
14        fun2=2*heaviside(t)-1;
15    otherwise
16        disp('Inserimento errato');
17        exit
18 end
19 F=simplify(fourier(fun2));%CALCOLO LA TRASFORMATA DI FOURIER SIMBOLICA
20 for N=[16,32,64]
21     T=N/2;
22     %AL RADDOPPIARE DI N RADDOPPIO ANCHE T IN MODO DA MANTENERE
23     %COSTANTE OMEGA(N/T) E QUINDI OTENGO I CAMPIONI DELLA TRASFORMATA DI
24     %FOURIER TUTTI NELLO STESSO INTERVALLO.
25     tj=T/N*((0:N)-N/2);%nodi di campionamento
26     ftj=fun(tj);
27     x=linspace(-T/2,T/2,299);
28     y=fun(x);
29     figure
30     plot(x,y,tj,ftj,'o');
31     axis tight;grid on;
32     str=char(fun2);
33     title(str);
34     legend('Segnale','Campioni');
35     %COSTRUISCO IL VETTORE DELLA DFT
36     f=[1/2*(ftj(1)+ftj(end));ftj(2:end-1)];
37     Ft=fftshift(fft(f));
38     Ft=[Ft;Ft(1)]*T/N;
39     Ft(2:end)=-Ft(2:end);
40     nu=(-N/2:N/2)'/T;O=N/T;
41     %VISUALIZZO L'APPROXIMAZIONE DELLA FT RISPETTO ALLA FREQUENZA CIRCOLARE
42     figure;hold on;
43     F=simplify(subs(F,'w','2*pi*ni'));
44     %TRASFORMATA RISPETTO ALLA FREQUENZA CIRCOLARE
45     ezplot(abs(F),[-O/2,O/2]);
46     plot(nu,abs(Ft),'or:');

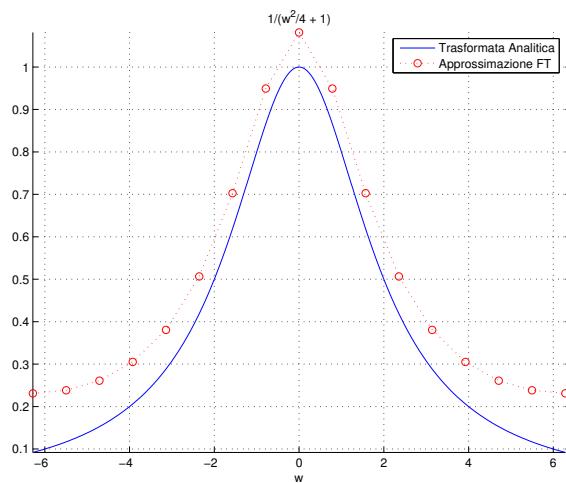
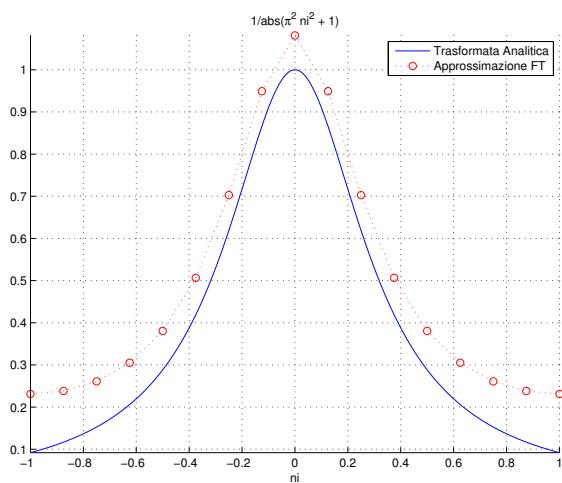
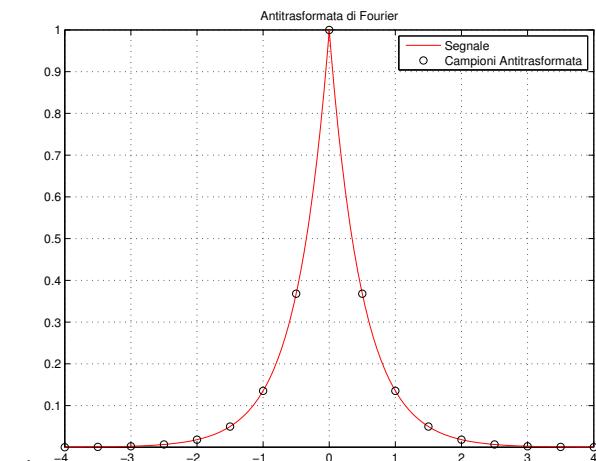
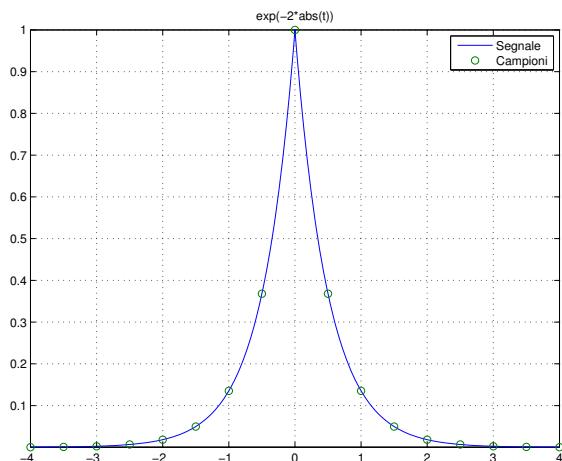
```

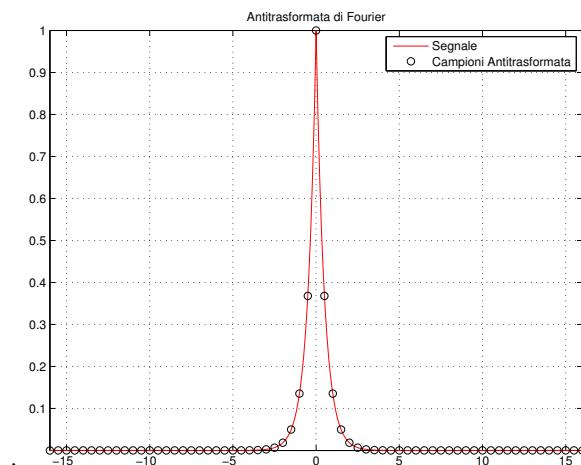
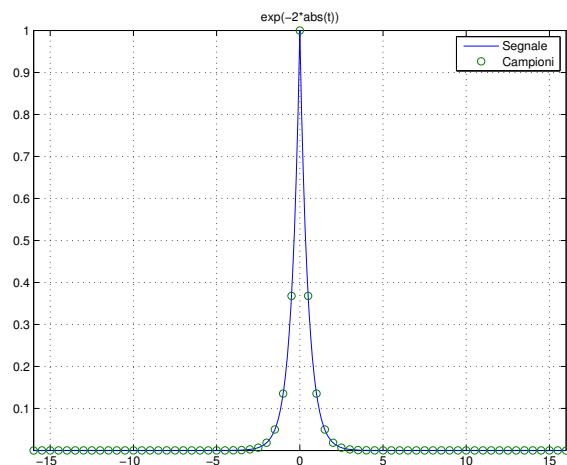
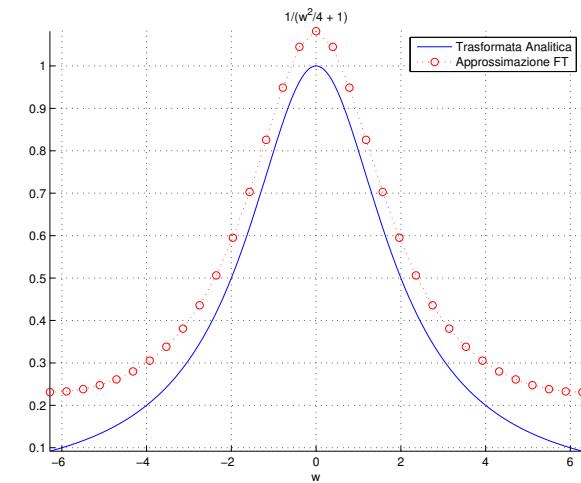
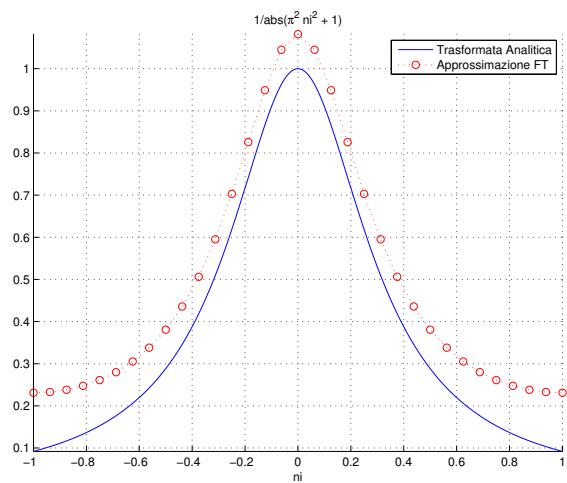
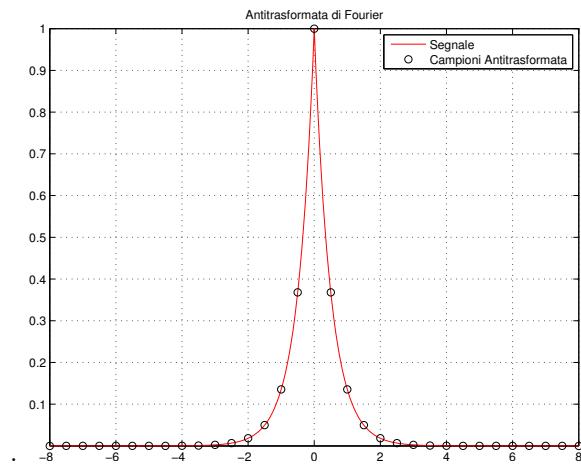
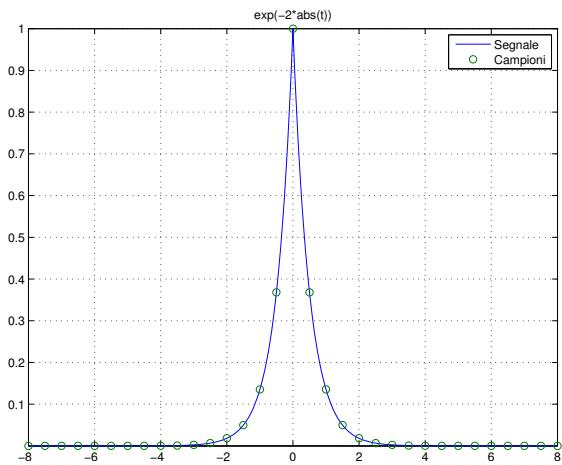
```

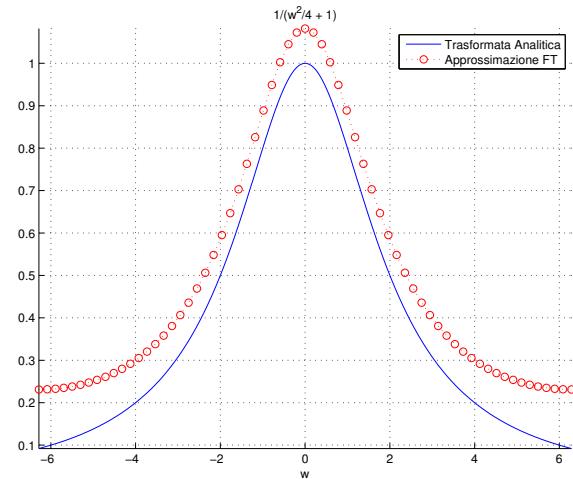
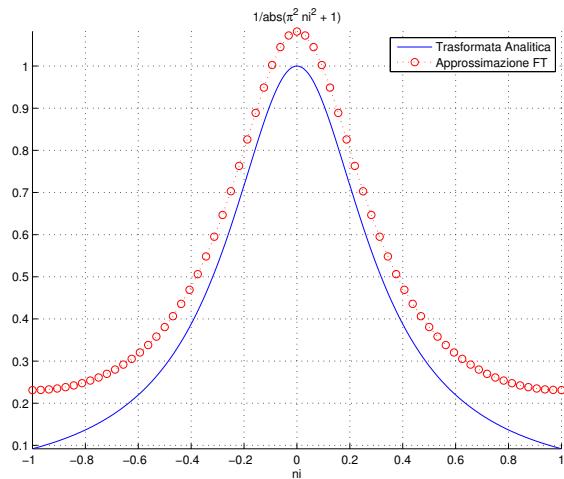
47 axis tight;grid on;
48 legend('Trasformata Analitica','Approssimazione FT');
49 %VISUALIZZO L'APPROXIMAZIONE DELLA FT RISPETTO ALLA FREQUENZA ANGOLARE
50 F=subs(F,'ni','w/(2*pi)');
51 %TRASFORMATA RISPETTO ALLA FREQUENZA ANGOLARE
52 figure;hold on;
53 ezplot(abs(F),[-0*pi,0*pi]);
54 plot(nu*2*pi,abs(Ft),'or:');
55 axis tight;grid on;
56 legend('Trasformata Analitica','Approssimazione FT');
57 %ANTITRASFORMATA DI FOURIER.A PARTIRE DAI CAMPIONI DELLA TRASFORMATA DI
58 %FOURIER APPLICO LO STESSO ALGORITMO PER TROVARE I CAMPIONI DI PARTENZA DEL
59 %SEGNALE.
60 g=[1/2*(Ft(1)+Ft(end));Ft(2:end-1)];
61 IFT=fftshift(ifft(g));
62 IFT=[IFT;IFT(1)]*N/T;
63 IFT(2:2:end)=-IFT(2:2:end);
64 figure
65 plot(x,y,'r',tj,real(IFT),'ok');
66 title('Antitrasformata di Fourier');
67 legend('Segnale','Campioni Antitrasformata');
68 axis tight;grid on;
69 end

```

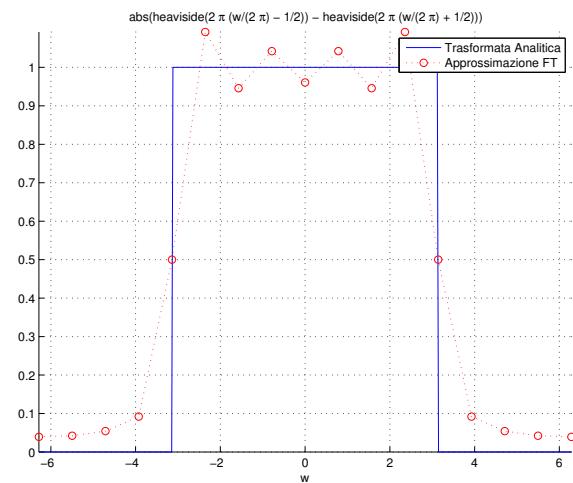
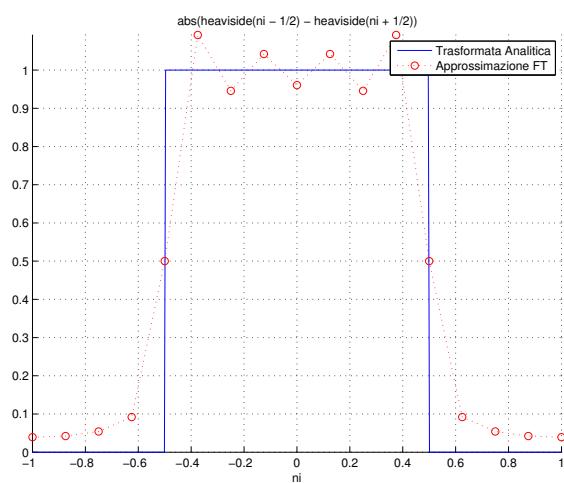
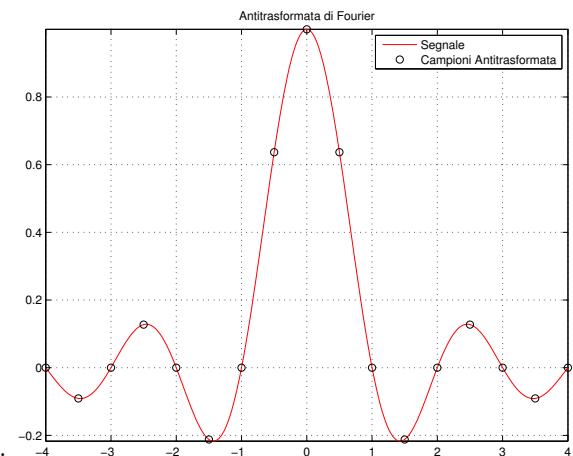
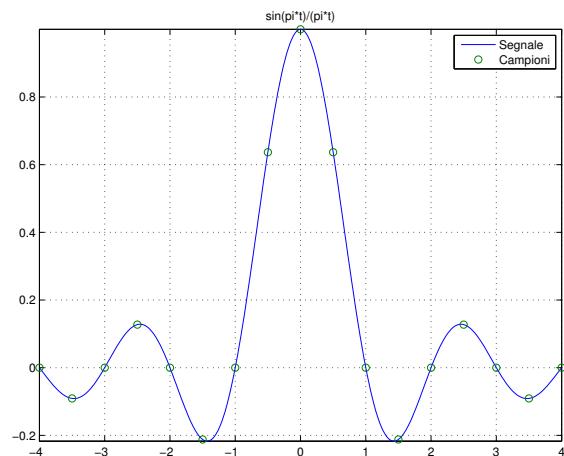
ESECUZIONE DEL PROGRAMMA(SCELTA=1, $f(t) = e^{-2|t|}$)

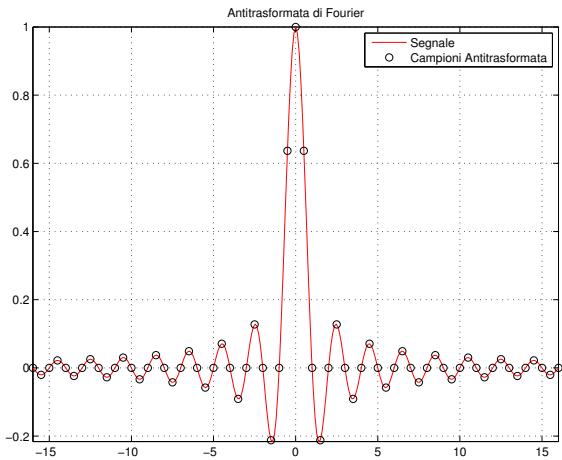
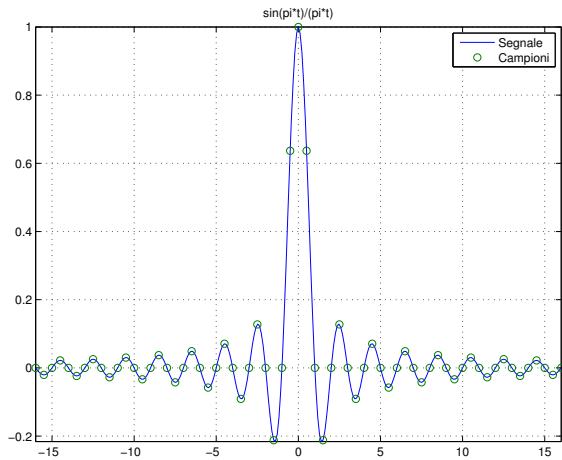
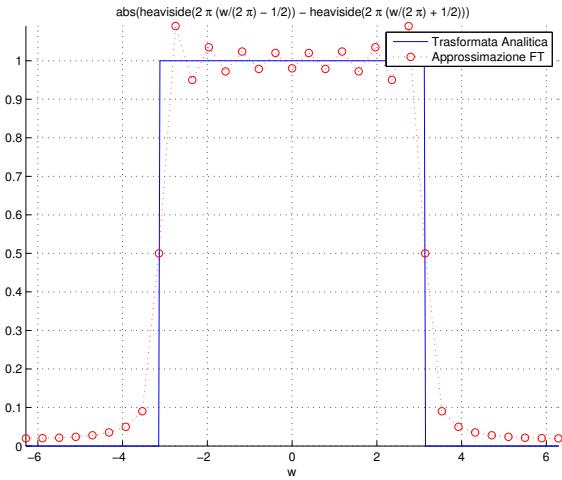
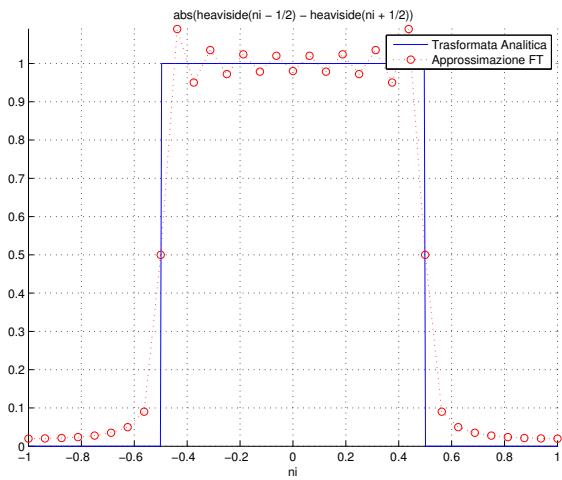
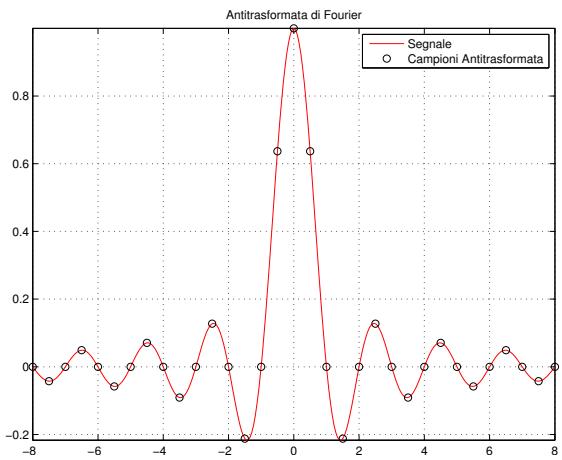
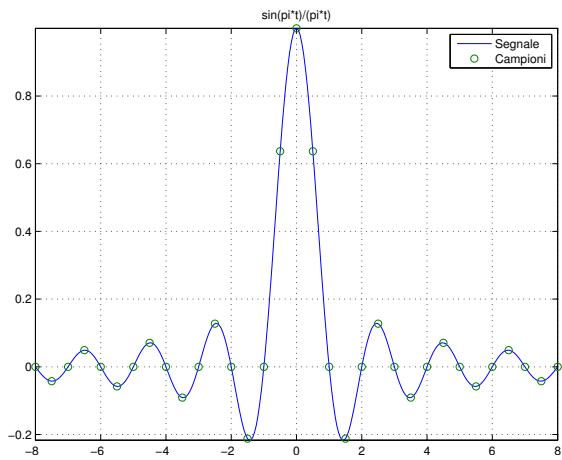


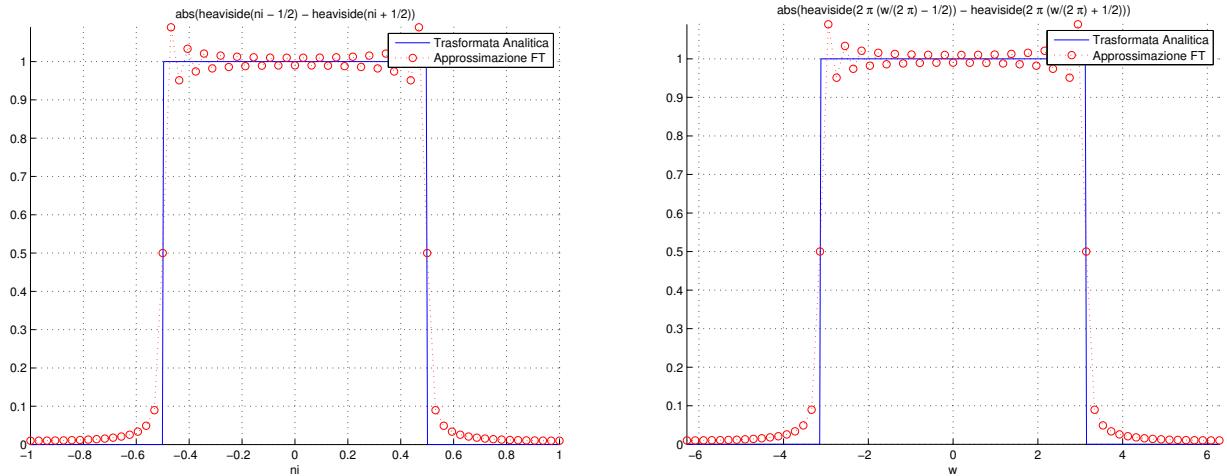




ESECUZIONE DEL PROGRAMMA (SCELTA=2, $f(t) = \text{sinc}$)







In questo programma e' stata approssimata la trasformata di Fourier di alcune funzioni mediante l'algoritmo che ci consente,a partire da N campioni della funzione in $[-T/2, T/2]$,di ottenere N campioni della FT in $[-\Omega/2, \Omega/2]$ dove $\Omega = N/T$.Avendo scelto T pari a $N/2$,al raddoppiare di N raddoppia anche T,quindi Ω rimane costante e i campioni della FT vengono ottenuti tutti nello stesso intervallo.I grafici sono stati prodotti sia rispetto alla frequenza angolare ω ,sia rispetto a quella circolare ν .Infine e' stato applicato l'algoritmo dell'antitrasformata di Fourier che ci consente a partire dai campioni della FT di ottenere i campioni del segnale di partenza.Infatti i campioni ottenuti sono sovrapposti al grafico di tale segnale.

6.3 Determinare,mediante la FT e senza usare spectrogram(),il numero di telefono dal suono registrato nei file:

-NumTel.wav
 -noise1NumTel.wav
 -noise2NumTel.wav
 estraendo le frequenze dei singoli toni (STFT).[liv.3]

```

1 [y,fs,nbits]=wavread('NumTel.wav');%LEGGO IL SEGNALE
2 N=numel(y);%N RAPPRESENTA IL NUMERO DI CAMPIONI
3 Dt=1/fs;%PASSO DI CAMPIONAMENTO
4 t_j=Dt*(0:N-1)';%ASCISSE DEI CAMPIONI
5 T=N*Dt;%DURATA TOTALE DEL SEGNALE
6 %APPLICO L'ALGORITMO PER L'APPROXIMAZIONE DELLA FT DEL SEGNALE
7 nu=(-N/2:N/2)'/T;
8 plot(tj,y);
9 title('Segnale di partenza');axis tight;grid on;
10 F=fftshift(fft(y));
11 F=[F;F(1)]*T/N;
12 F(1:2:end)=-F(1:2:end);
13 figure
14 plot(nu,abs(F));
15 title('Trasformata del segnale completo');axis tight;grid on;
16 M=fix(N/10);
17 figure
18 spectrogram(y,M,0,[],fs);%VISUALIZZO LO SPETTROGRAMMA
19 V=axis;
20 axis([600 1500 0 3]);
21 T=T/10;%AMPIEZZA DELLA FINESTRA STFT
22 for j=[0:9]

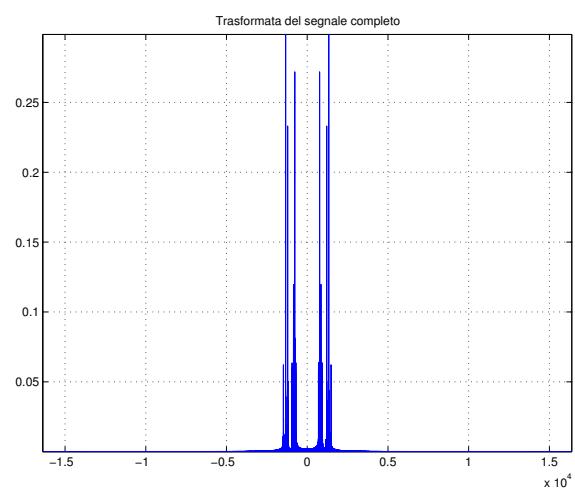
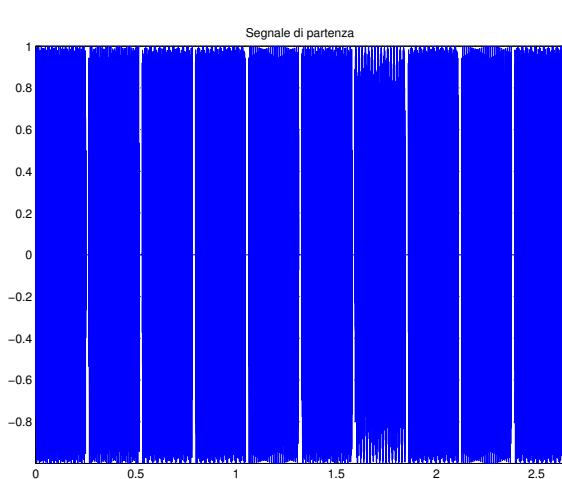
```

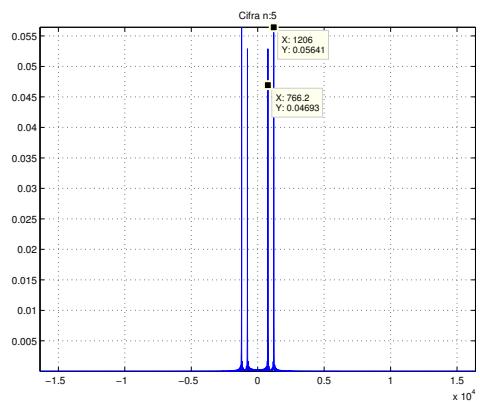
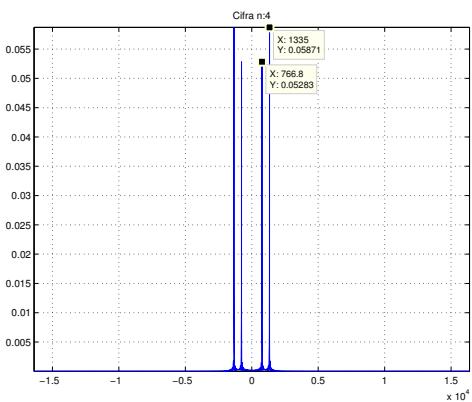
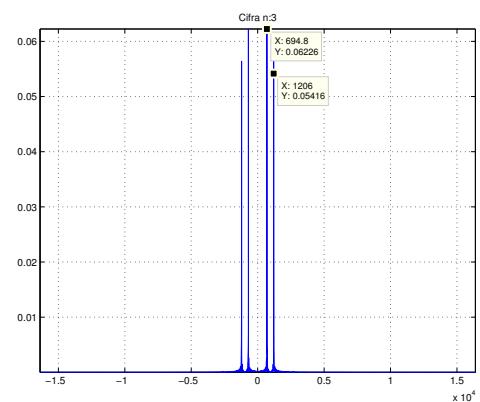
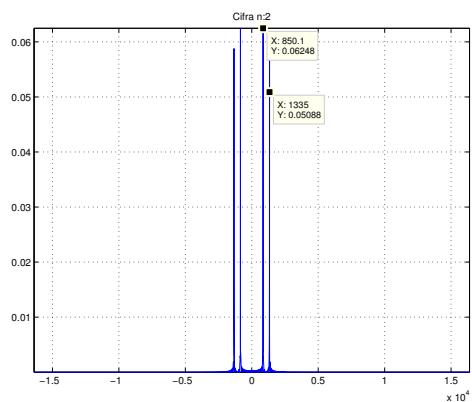
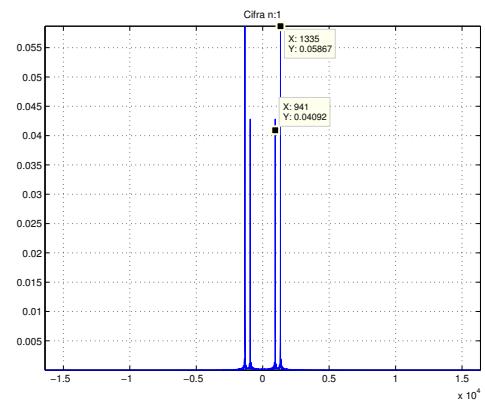
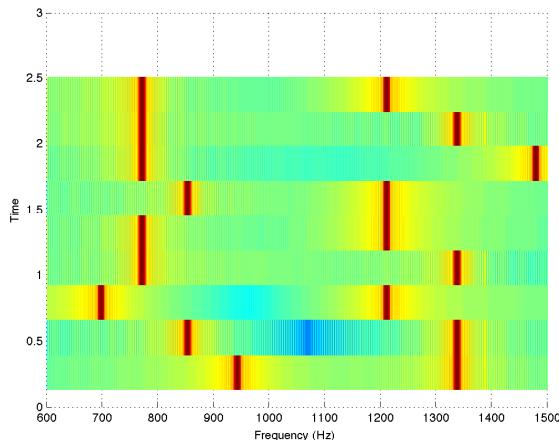
```

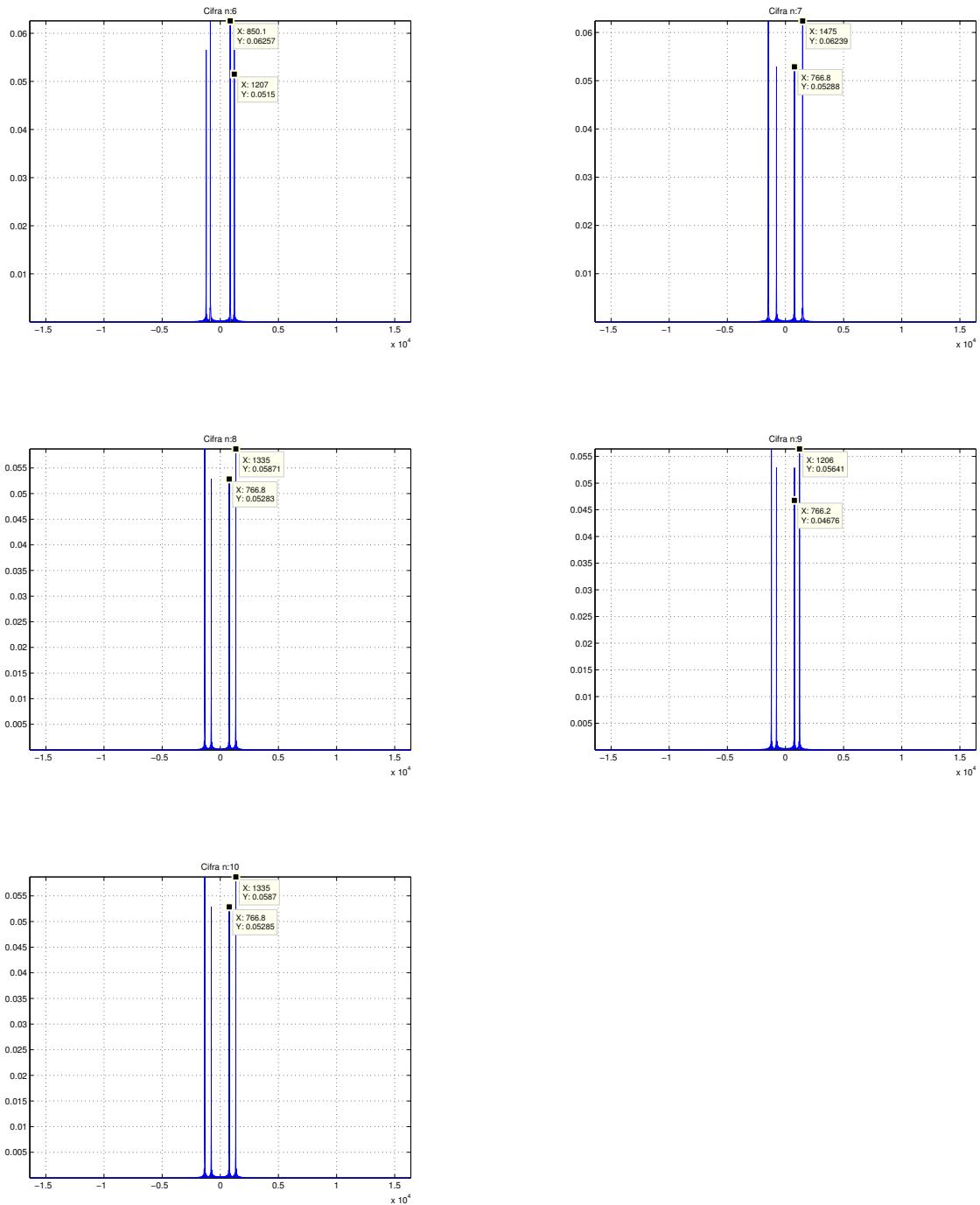
23 %SICCOME IL SEGNALE E' COSTITUITO DA 10 PARTI INTERVALLATE DA GAP DI
24 %SILENZIO FACCIO VARIARE UN' INDICE J TRA 0 E 9.TALE INDICE MI
25 %PERMETTERA' DI SCORRERE CON 10 FINESTRE SEPARATE IL SEGNALE DI
26 %PARTENZA E DI ISOLARE QUINDI LE SINGOLE FREQUENZE COMPONENTI.
27 N=fix(T/Dt);
28 %N INDICA IL NUMERO DI CAMPIONI NELLA FINESTRA DI DIMENSIONE T CON
29 %PASSO DI CAMPIONAMENTO DT.
30 Tj=t(j*(N*j+1:N*j+N));
31 %TJ SONO LE ASCISSE DEI CAMPIONI IN TALE FINESTRA
32 %[J=0 ==> TJ(1:N), J=1 ==> TJ(N+1,2N) ...
33 Yj=y(N*j+1:N*j+N);
34 %YJ SONO LE ORDINATE DEI CAMPIONI
35 %APPLICO QUINDI L'ALGORITMO PER L'APPROXIMAZIONE DELLA FT AI CAMPIONI
36 %FINESTRATI
37 nu=(-N/2:N/2)'/T;
38 F=fftshift(fft(Yj));
39 F=[F;F(1)]*T/N;
40 if mod(N/2,2)==0
41     F(2:2:end)=-F(2:2:end);
42 else
43     F(1:2:end)=-F(1:2:end);
44 end
45 figure
46 plot(nu,abs(F));
47 str=sprintf('Cifra n:%d',j+1);
48 title(str);axis tight;grid on;
49 end

```

ESECUZIONE DEL PROGRAMMA (NumTel.wav)



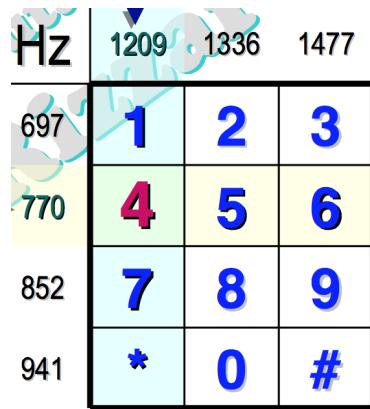




Questo programma prende in input un segnale che rappresenta la composizione di un numero telefonico mediante un tastierino multifrequenza. Il suono di ogni tasto e' la somma di due sinusoidi opportune:

$$y = \frac{\sin(2\pi\Phi_{row}t) + \sin(2\pi\Phi_{col}t)}{2}$$

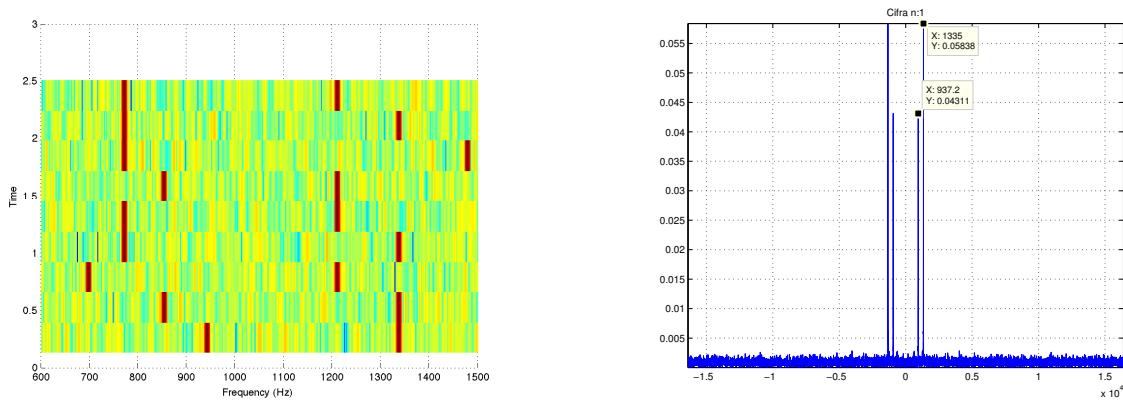
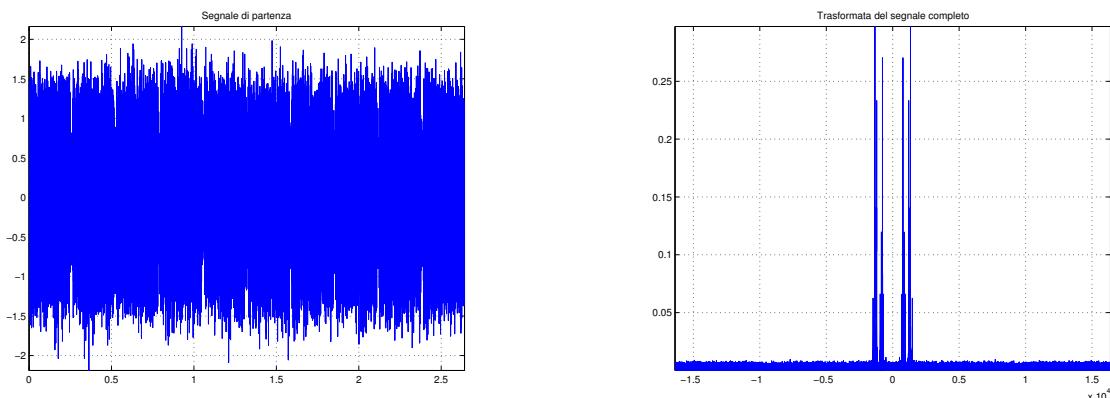
Dove Φ_{row} e Φ_{col} sono le frequenze identificate dalla seguente tabella:

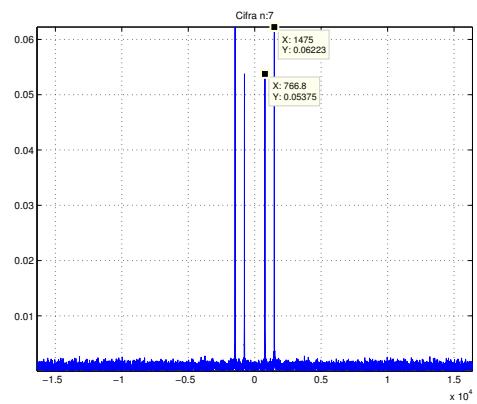
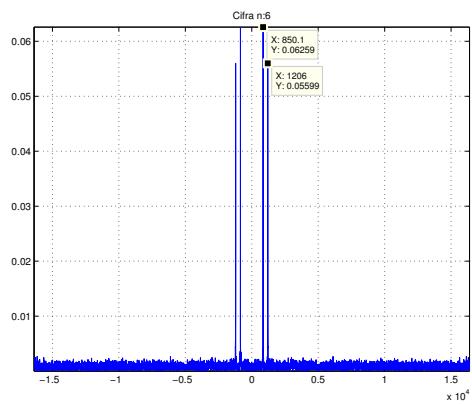
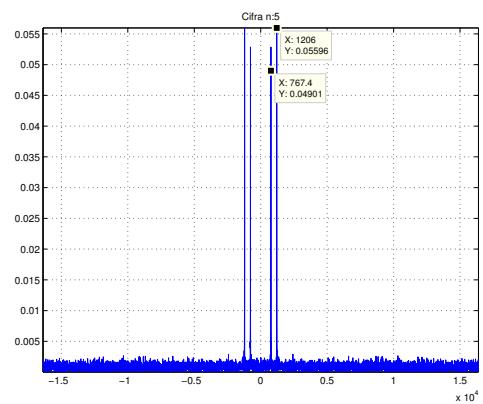
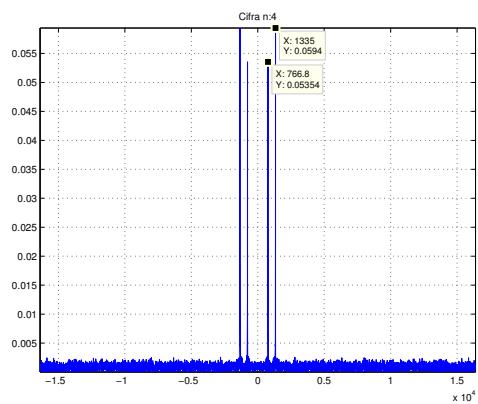
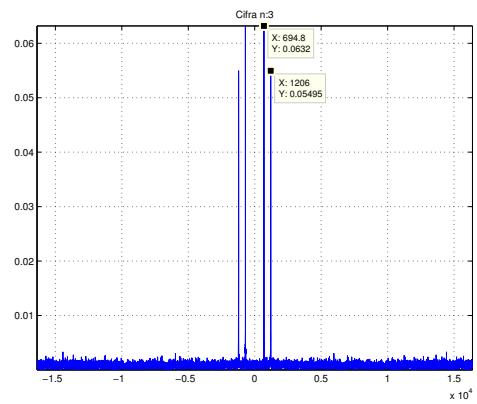
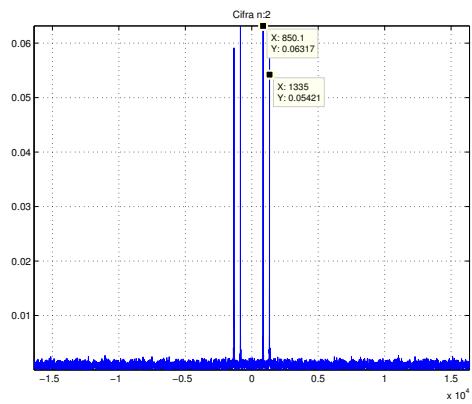


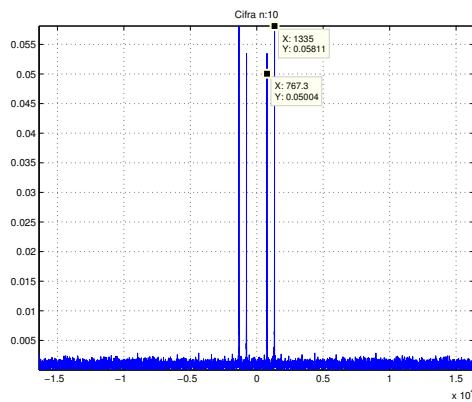
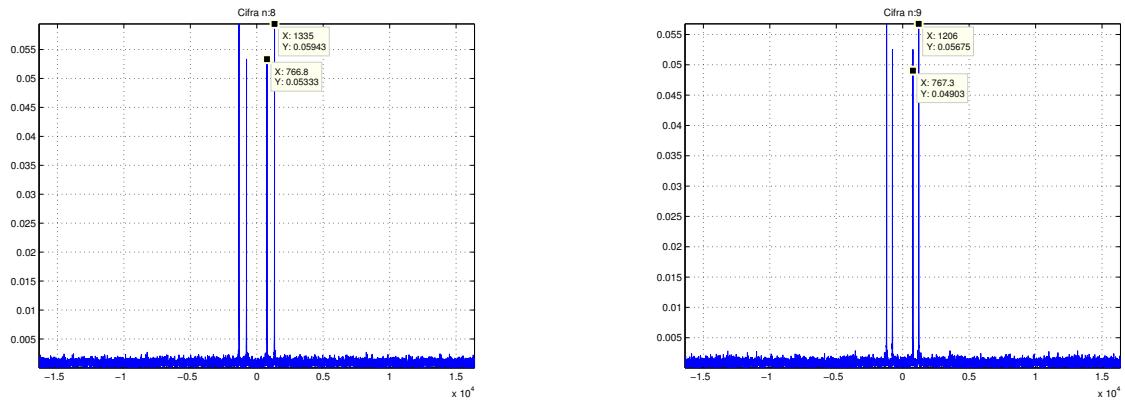
Il programma inizialmente rappresenta il grafico del segnale in funzione del tempo, poi la sua trasformata di Fourier globale, riferita quindi all'intero segnale, dove troviamo tutte le frequenze componenti e infine viene utilizzata una finestra che scorre e inquadra una cifra alla volta (Short Time Fourier Transform) in modo da isolare le singole frequenze che compongono il numero telefonico e risalire quindi al numero digitato. In ordine, come si può osservare dalle figure prodotte in output dal programma, le singole frequenze componenti sono:

(1335,941), (850,1335), (695,1206), (1332,766), (1206,766), (850,1207), (1475,767), (1335,767), (1206,766), (1335,767).
Il numero è quindi: 0815476545.

ESECUZIONE DEL PROGRAMMA (noise1NumTel.wav)



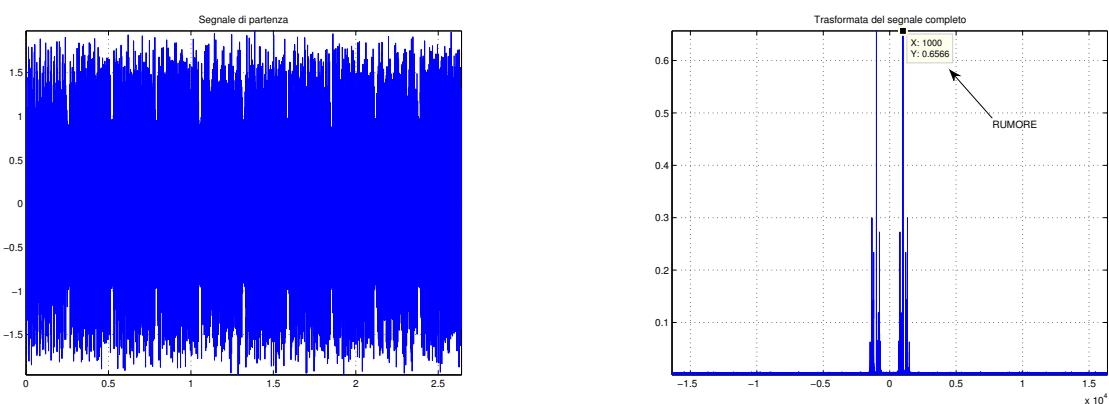


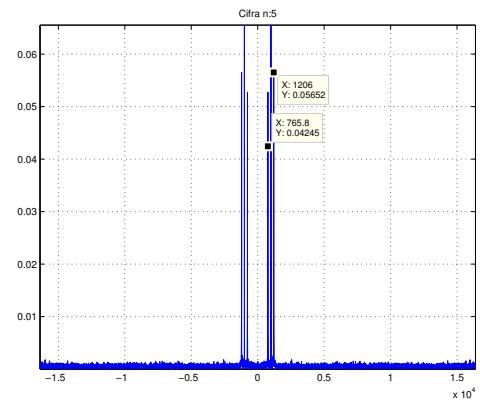
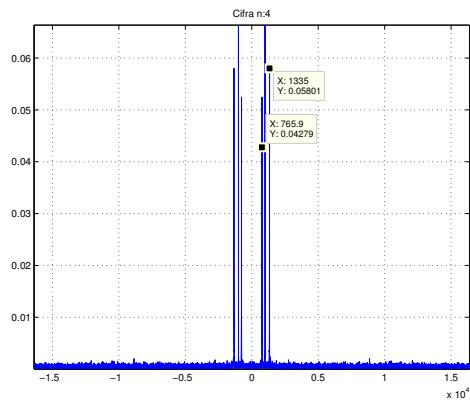
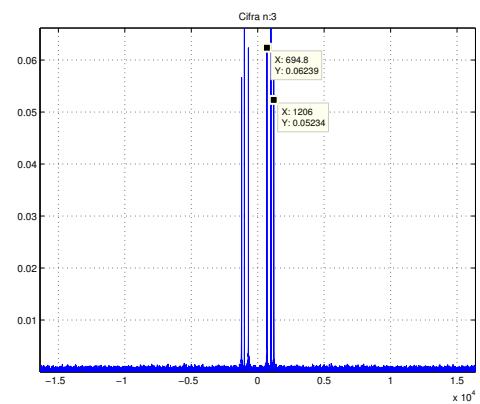
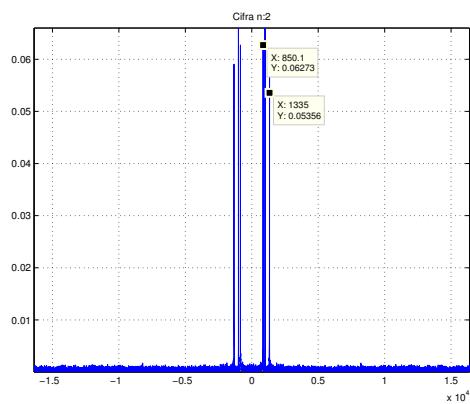
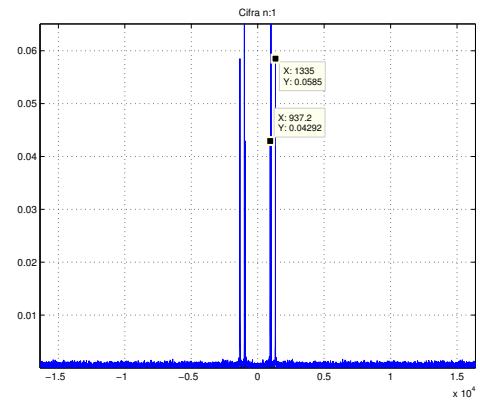
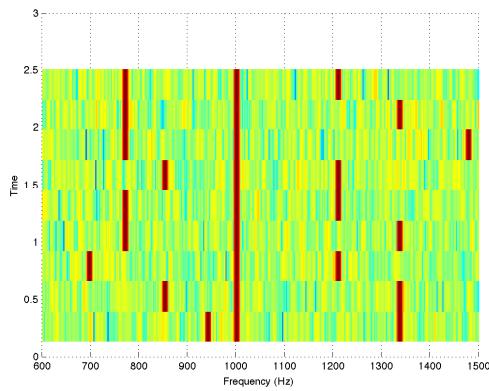


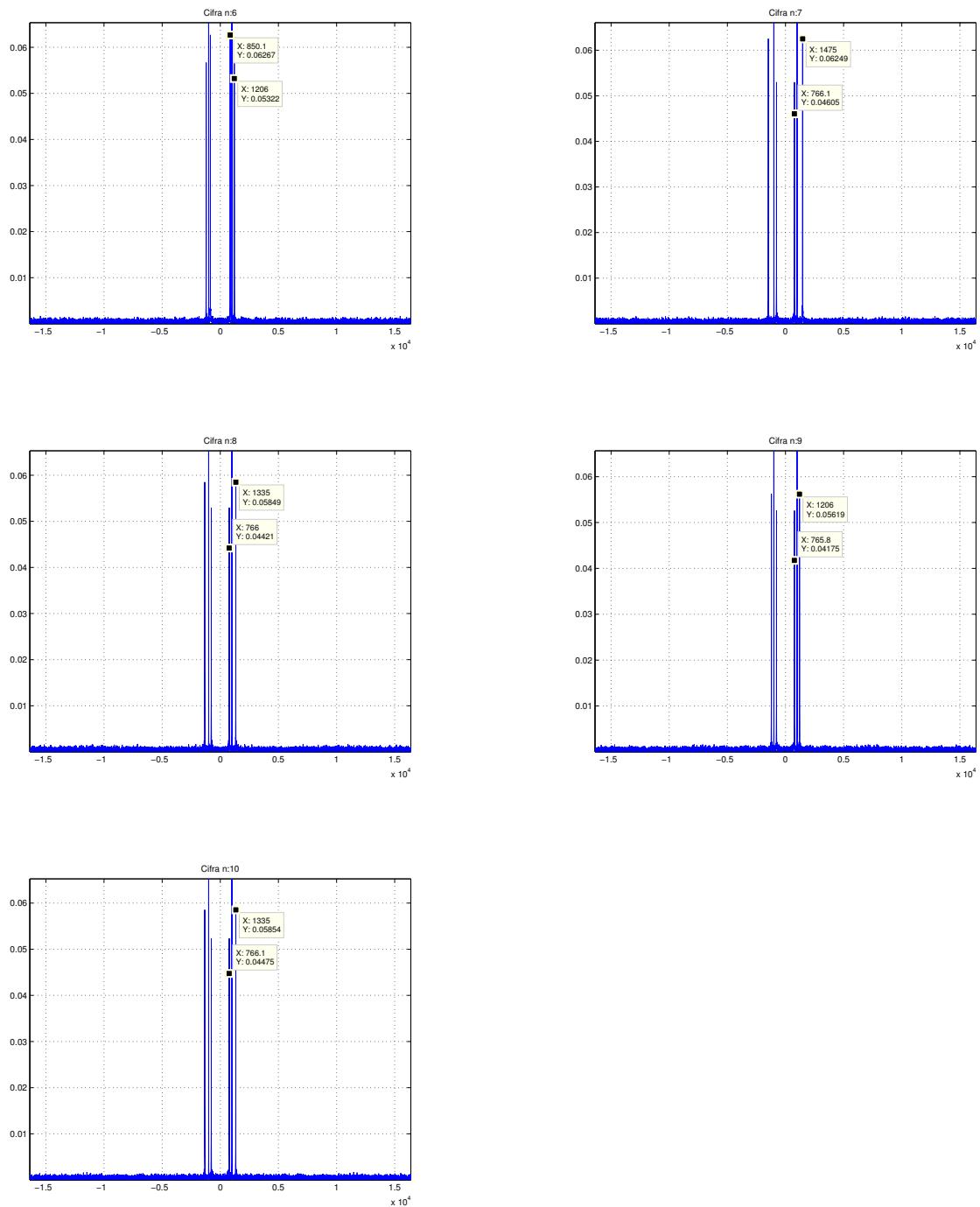
Come si osserva dalle immagini il segnale e' piu' disturbato del precedente. Le singole frequenze componenti in questo caso sono:

(1335,937),(850,1335),(695,1206),(1335,767),(1206,767),(850,1206),(1475,767),(1335,767),(1206,767),(1335,767)
Il numero e' quindi: 0815476545.

ESECUZIONE DEL PROGRAMMA (noise2NumTel.wav)







Come si osserva dalle immagini il segnale e' ancora piu' disturbato del precedente. Come si puo' osservare dallo spettrogramma e' presente un rumore fisso intorno ai 1000 Hz. Le singole frequenze componenti in questo caso sono:
 $(1335, 937), (850, 1335), (695, 1206), (1335, 767), (1206, 767), (850, 1206), (1475, 767), (1335, 767), (1206, 767), (1335, 767)$
Il numero e' quindi: 0815476545.

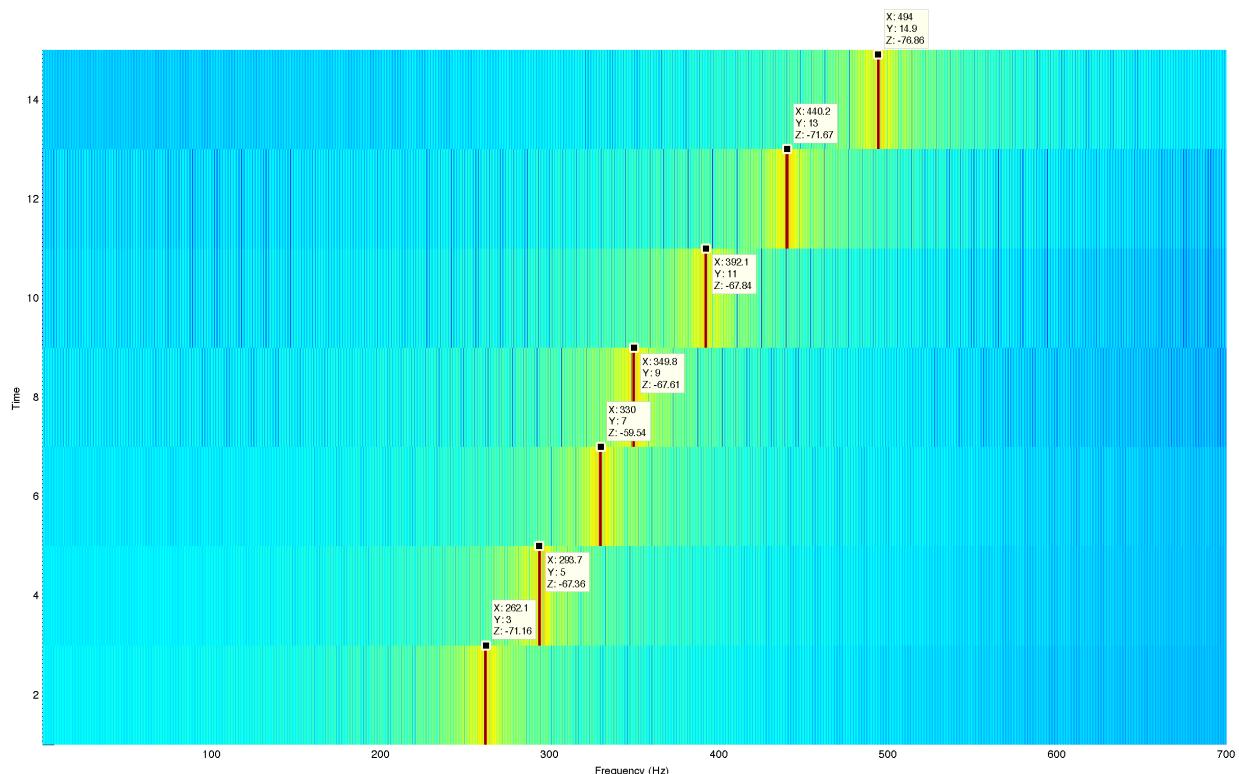
- 6.4 A partire dalle frequenze delle note (nel file frequenze_note_musicali.pdf), creare in Matlab un file audio (function auwrite(), wavwrite()), con al piu' 2 secondi per nota, contenente:
 -le 7 note musicali
 -un accordo (tipo do_maggiore=DO-MI-SOL). [liv.3]

```

1 t=linspace(0,2,100000);
2 dt=t(2)-t(1);%PASSO DI CAMPIONAMENTO
3 fs=1/dt;%FREQUENZA DI CAMPIONAMENTO
4 freq_base=261.6;%FREQUENZA DEL DO
5 i=1;
6 for n=[2,4,5,7,9,11]
7   freq(i)=freq_base*2^(n/12);
8   i=i+1;y=[];
9 end
10 frequenze=[freq_base,freq];
11 %FREQUENZE DELLE 7 NOTE MUSICALI
12 for n=1:7
13   y=[y,sin(2*pi*frequenze(n)*t)];
14   %COSTRUISCO AD OGNI PASSO UN SEGNALE SINUSOIDALE CON LE FREQUENZE DELLE
15   %NOTE MUSICALI.LA DURATA E' PARI A 2 SECONDI PER NOTA (t VA DA 0 A 2)
16 end
17 %INFINE COSTRUISCO L'ACCORDO DO-MI-SOL COME SOMMA DELLE TRE SINUSOIDI CON
18 %LE FREQUENZE DELLE TRE NOTE
19 y=[y,sin(2*pi*frequenze(1)*t)+sin(2*pi*frequenze(3)*t)+...
20 sin(2*pi*frequenze(5)*t)];
21 N=numel(y);
22 M=fix(N/8);
23 figure
24 spectrogram(y,M,0,[],fs);
25 V=axis;
26 axis([0 700 V(3:4)])
27 sound(y,fs);
28 wavwrite(y,fs,'note.wav');
```

Warning: Data clipped during write to file:note.wav

ESECUZIONE DEL PROGRAMMA



DO (DO4)	261.6 Hz
DO# = REb	277.2 Hz
RE	293.7 Hz
RE#	311.1 Hz
MI	329.6 Hz
FA	349.2 Hz
FA#	370.0 Hz
SOL	392.0 Hz
SOL#	415.3 Hz
LA	440.0 Hz
LA#	466.0 Hz
SI	493.9 Hz
DO	523.3 Hz

Accordo

