LA GRAFICA IN MATLAB

- I diagrammi più usati in MatLab sono quelli bidimensionali o **xy**. In generale vengono rappresentate funzioni del tipo $y = f(x)$, in cui solitamente sull'asse orizzontale vengono riportati i valori della variabile indipendente **x** e sull'asse verticale quelli della variabile dipendente ***y*.**

Anatomia di un diagramma

L'anatomia e la nomenclatura di un tipico diagramma **xy** sono riportate in figura 1, che rappresenta una serie di dati e la curva generata da una funzione (modello).

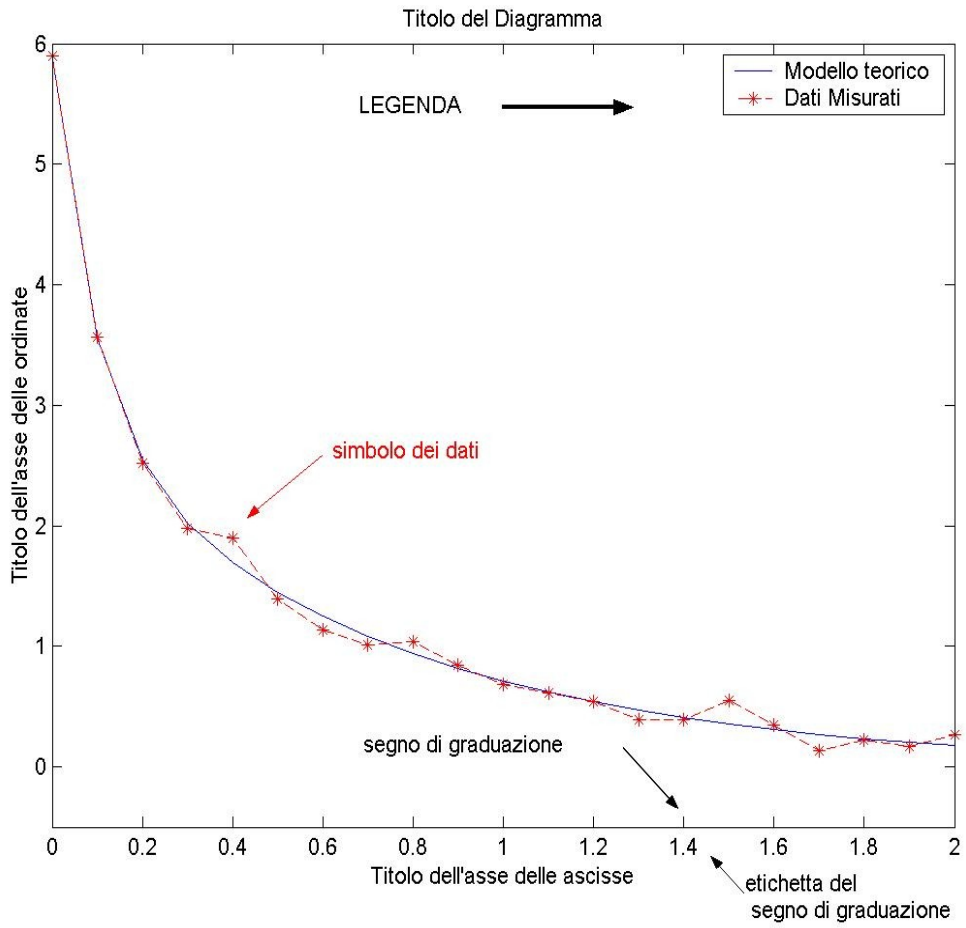


Figura 1

- La scala nei due grafici fa riferimento all'intervallo e alla spaziatura dei numeri rappresentati. Entrambi gli assi di questo diagramma sono lineari, in quanto la spaziatura dei numeri è costante. Un altro tipo di scala è quella logaritmica, che sarà spiegata in seguito.

- I segni di graduazione (**tick mark**) sono posti sull'asse per facilitare la valutazione dei numeri rappresentati nel diagramma. Le etichette dei segni di graduazione (**tick mark label**) sono i numeri che corrispondono alle posizioni dei segni di graduazione.

- Ogni asse deve avere un'etichetta o titolo dell'asse, questo titolo assegna il nome all'asse e descrive la quantità rappresentata lungo l'asse. Solitamente un diagramma ha anche un titolo che ne descrive il contenuto; questo titolo è posto nella parte superiore del diagramma.

- Un diagramma può essere ottenuto da dati sperimentali o da un'equazione. Quando i dati vengono riportati nel diagramma, ogni dato viene rappresentato con un simbolo o marcatore, come ad esempio il piccolo asterisco della figura 1. A volte i simboli dei dati sono collegati da una linea per agevolare l'interpretazione dei dati, specialmente se i dati del diagramma sono pochi.

- Quando vengono rappresentate più curve o serie di dati nello stesso diagramma, deve essere possibile distinguerle. Un metodo per farlo consiste nell'utilizzare una legenda che mette in relazione il simbolo della serie di dati o il tipo di curva con le quantità rappresentate nel diagramma.

Comandi per creare diagrammi, etichette, titoli

- La funzione di base per creare diagrammi **xy** è **plot (x, y)**, se **x** e **y** sono vettori di uguale lunghezza, MatLab genera una sola curva con i valori delle componenti del vettore **x** riportati sull'asse delle ascisse e quelli delle componenti di **y** sull'asse delle ordinate

Per rappresentare il grafico di $f(x)=cos(x)$ con $x \in [-2\pi,\ 2\pi]$ si possono dare i seguenti comandi:

```
>>  x = -2*pi : 0.01 : 2*pi ;
>>  y = cos(x);
>> plot(x,y);
```

Lo stesso grafico si poteva ottenere anche tramite la funzione **eval**, che consente di valutare una funzione definita dall'utente mediante una stringa:

```
>> x = -2*pi : 0.01 : 2*pi ;
>> f= 'cos(x)';
>> y=eval(f);
>>figure,plot(x,y)
```

La prima volta che viene usato il comando **plot** viene aperta una finestra e ogni nuovo comando **plot** sulla stessa finestra cancella il grafico precedente, a meno che non si usi il comando

```
>> hold on
```

che viene annullato con

```
>> hold off
```

Una nuova finestra grafica viene aperta con

```
>> figure
```

e gli viene assegnato un numero progressivo. Una finestra aperta precedentemente viene chiusa con il comando **close** seguito dal numero della finestra che si vuole chiudere. Per far diventare corrente una finestra aperta precedentemente basta *cliccarla*.

I grafici bidimensionali sono ottenuti congiungendo con una spezzata punti del piano le cui coordinate sono le componenti di due vettori: il primo contiene le ascisse e il secondo le ordinate.

```
>> x=linspace(0,10,500);
>> y=cos(x)-sin(x).^2;
>> plot(x,y)
```

- Nel comando **plot** si può aggiungere anche lo stile che può essere:

y	yellow	.	point
m	magenta	o	circle
c	cyan	x	x-mark
r	red	+	plus
g	green	-	solid
b	blue	*	star
w	white	:	dotted
k	black	-.	dashdot
		--	dashed

La lista completa dei parametri da usare con il comando **plot** si ottiene con

>> help plot

Si vede così che si possono ottenere grafici con linee tratteggiate, oppure con punti, simboli ecc.

Ad esempio:

» plot(x,y,'r--')

- I comandi **xlabel** e **ylabel**, creano rispettivamente i titoli o le etichette per i rispettivi assi **x** e **y**. La sintassi per entrambi è:

  xlabel(’testo’) o ylabel(’testo’)

  dove ’testo’ è il titolo da assegnare all’asse corrispondente.

- La funzione **plot(x,y)** di MatLab imposta automaticamente la spaziatura dei segni di graduazione per i due assi cartesiani e pone le etichette appropriate: questa caratteristica si chiama SCALA AUTOMATICA (*autoscaling*).

- Il comando **grid** visualizza le linee di una griglia in corrispondenza delle etichette dei segni di graduazione di un diagramma. Digitando **grid** on si aggiunge la griglia, **grid off** la si esclude.

- Il comando **axis** serve a modificare le impostazioni dei valori limite che MatLab ha scelto per gli assi. La sintassi di base del comando è:

  axis([xmin xmax ymin ymax])

  Esistono diverse varianti di questo comando:

- **axis square** :seleziona i limiti degli assi in modo che il diagramma sia quadrato;
- **axis equal**: seleziona i fattori di scala e la spaziatura dei segni di graduazione in modo che siano uguali nei due assi;
- **axis auto**: attiva le impostazioni di MatLab che calcolano i limiti ideali degli assi in modo automatico.

- Matlab dispone anche del comando **fplot** per creare diagrammi di funzioni,; in particolare questo comando esamina la funzione da rappresentare stabilendo automaticamente il numero di punti da utilizzare, in modo che il diagramma presenti tutte le caratteristiche della funzione. La sua sintassi è:

fplot (‘stringa’, [xmin xmax])

Dove ‘**stringa**’ è un testo che descrive la funzione da rappresentare e **[xmin xmax]** specifica il minimo e il massimo valore della variabile indipendente **x**.

Esempio:

```
>> f = ’cos(tan(x))-tan(sin(x))’;

>> fplot (f,[1,2]);
```

Il simbolo **x** deve essere usato per la variabile indipendente.

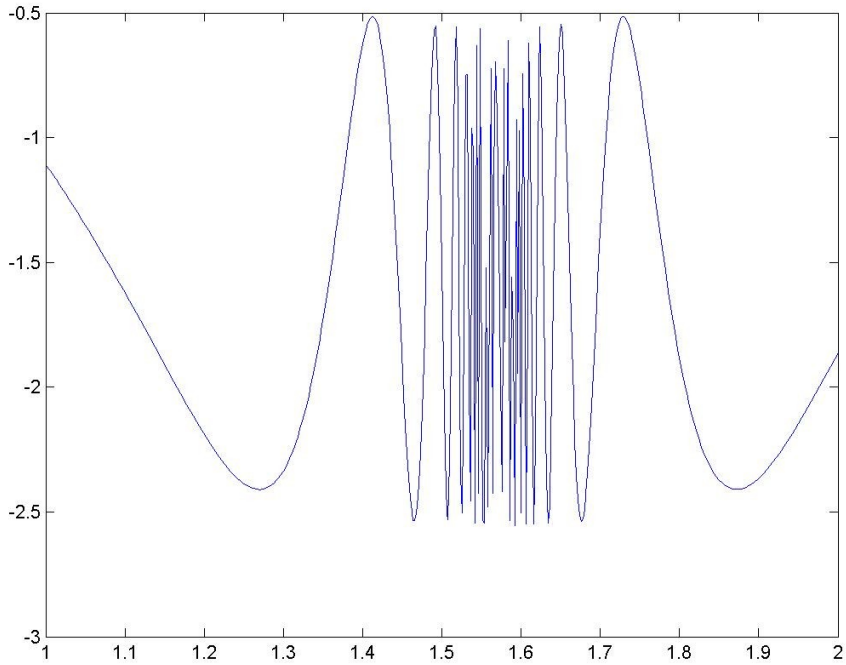


Figura 2

Se confrontiamo questo grafico con quello generato da **plot(x,y)**:

```
>> x = [ 1 : .01 :2 ];
>> y = cos(tan(x))-tan(sin(x)) ;
```

```
>> plot(x,y) ;
```

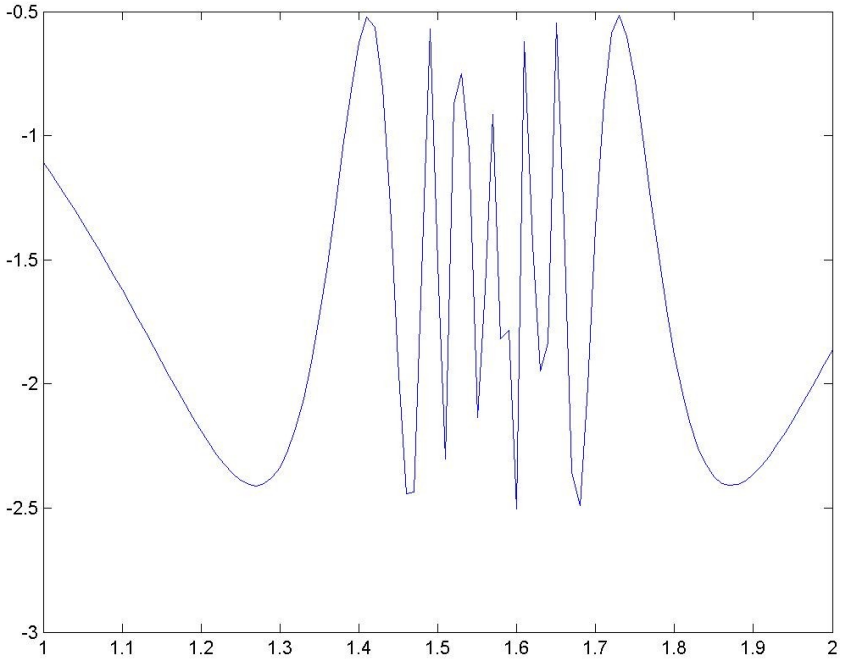


Figura 3

Si può notare come **fplot** abbia scelto automaticamente un numero di punti per visualizzare tutte le variazioni della funzione. Lo stesso diagramma può essere ottenuto con il comando plot, ma è necessario conoscere quanti valori utilizzare per specificare il vettore **x**.

Diagrammi multipli e sovrapposti

- MatLab è in grado di creare grafici che contengono più diagrammi distinti o diagrammi sovrapposti, si tratta di diagrammi particolarmente utili per confrontare gli stessi dati rappresentati in tipi di assi differenti.

- Il comando **subplot(m,n,p)** suddivide la finestra grafica di MatLab in una serie di pannelli rettangolari disposti su **m** righe e **n** colonne. La variabile **p** indica a MatLab di porre l'output del comando plot che segue il comando **subplo**t nel **p**_esimo pannello.

Esempio dell'uso del comando **subplot**.

```
x=[0:.01:5];
y=exp(-1.2*x).*(sin(10*x+5));
subplot(1,2,1);
plot(x,y),xlabel('x'),ylabel('y'),axis([0 5 -1 1])
x=[-6:.01:6];
y=abs(x.^3-100);
subplot(1,2,2);
plot(x,y),xlabel('x'),ylabel('y'),axis([-6 6 0 350])
```

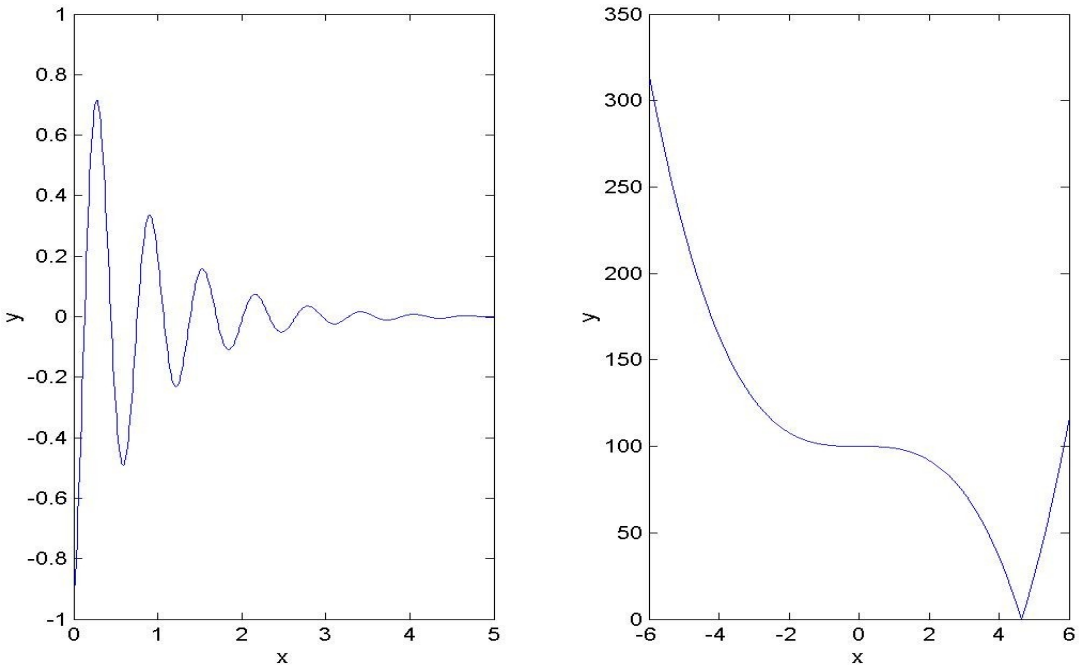

1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
y
0
1
2
3
4
5
x
350
300
250
200
150
100
50
0
y
-6
-4
-2
0
2
4
6
x

Figura 4

Le seguenti varianti del comando plot possono essere utilizzate per creare diagrammi sovrapposti:

- **plot(A)** : rappresenta in diagramma le colonne di **A** in funzione dei loro indici, generando **n** curve; **A** è una matrice con **m** righe e **n** colonne;
- **plot(x,A)** : rappresenta in diagramma la matrice **A** in funzione del vettore **x**; **x** è un vettore riga o un vettore colonna e **A** è una matrice **m** *x* **n**. Se la dimensione di **x** è pari a **m**, ogni colonna di **A** viene rappresentata in funzione del vettore **x**; se la dimensione di **x** è pari a **n**, ogni riga di **A** viene rappresentata in funzione del vettore **x**.
- **plot(A,x)** : rappresenta in diagramma il vettore **x** in funzione della matrice **A** ; Se la dimensione di **x** è pari a **m**, **x** sarà rappresentato in funzione delle colonne di **A**; se la dimensione di **x** è pari a **n**, **x** sarà rappresentato in funzione delle righe di **A**.
- **plot (A,B)** : rappresenta in diagramma le colonne della matrice **B** in funzione delle colonne della matrice **A**;

- Il comando **hold** crea un diagramma che richiede due o più comandi plot. Il seguente file script consente di creare questo diagramma con il comando **hold**. La figura 5 illustra il risultato di questo file script.

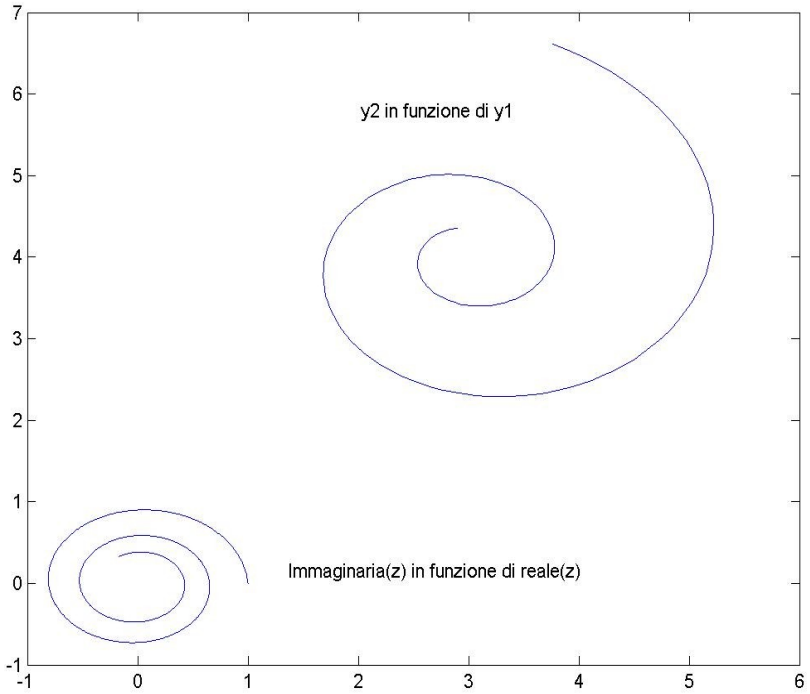


Figura 5

- È possibile creare diagrammi in scala logaritmica, che per semplicità vengono chiamati *diagrammi logaritmici*. L'utilizzo di diagrammi logaritmici permette di rappresentare facilmente un insieme di dati che si estende per un intervallo molto vasto di valori e di mettere in evidenza particolari andamenti nella variazione dei dati. Le figure 6 e 7 due diagrammi della seguente funzione:

Il primo diagramma utilizza assi in scala aritmetica, mentre il secondo è un diagramma logaritmico. A causa dell'ampio intervallo di valori che le variabili x e y

possono assumere, il diagramma in scala aritmetica non riesce a mettere in evidenza importanti caratteristiche della funzione.

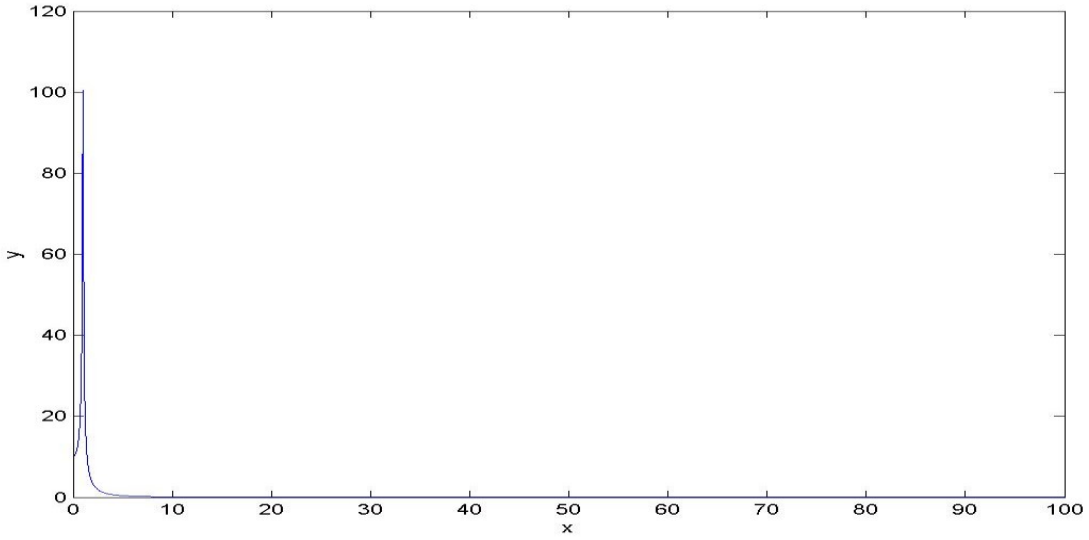


Figura 6

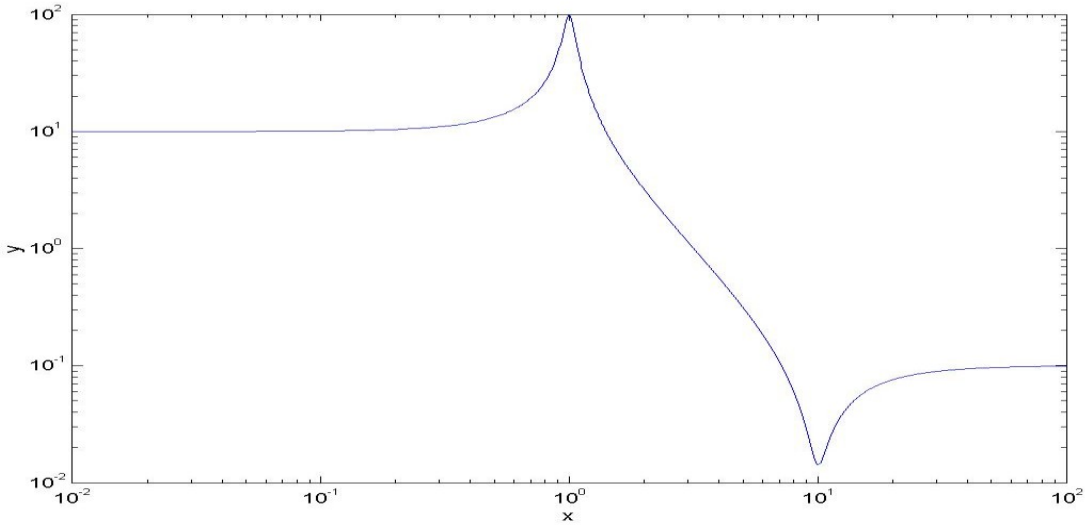
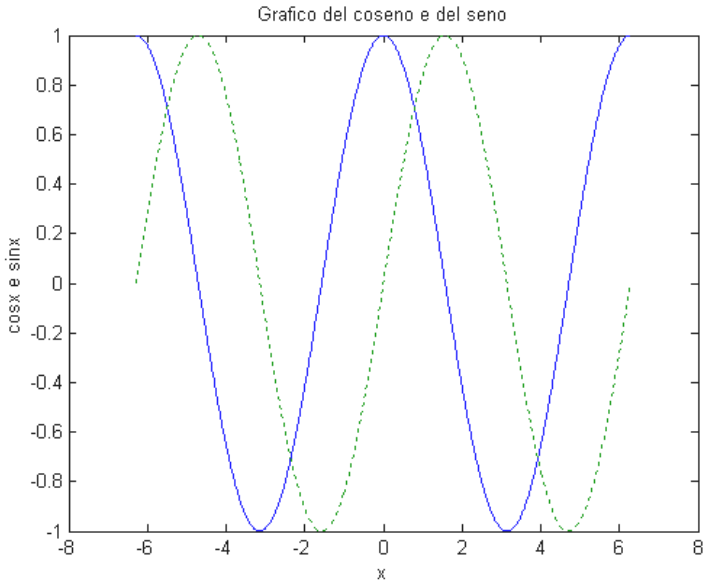


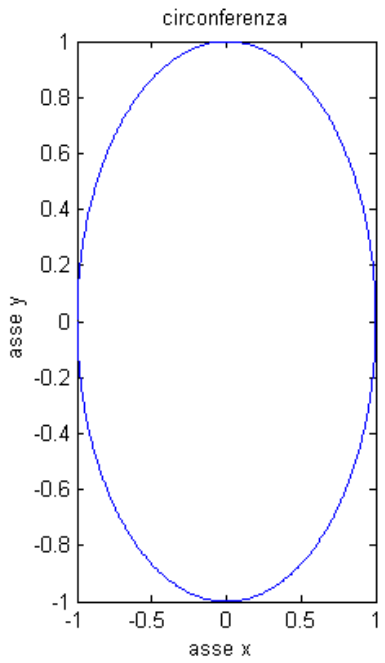
Figura 7

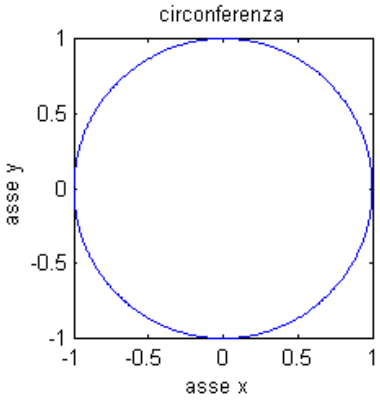
Esempi di creazione di diagrammi

```
x = -2*pi:0.01:2*pi;
y = cos(x);
y2 = sin(x);
figure,plot(x,y,'-',x,y2,':');
ylabel('cosx e sinx');
xlabel('x');
title('Grafico del coseno e del seno');
```



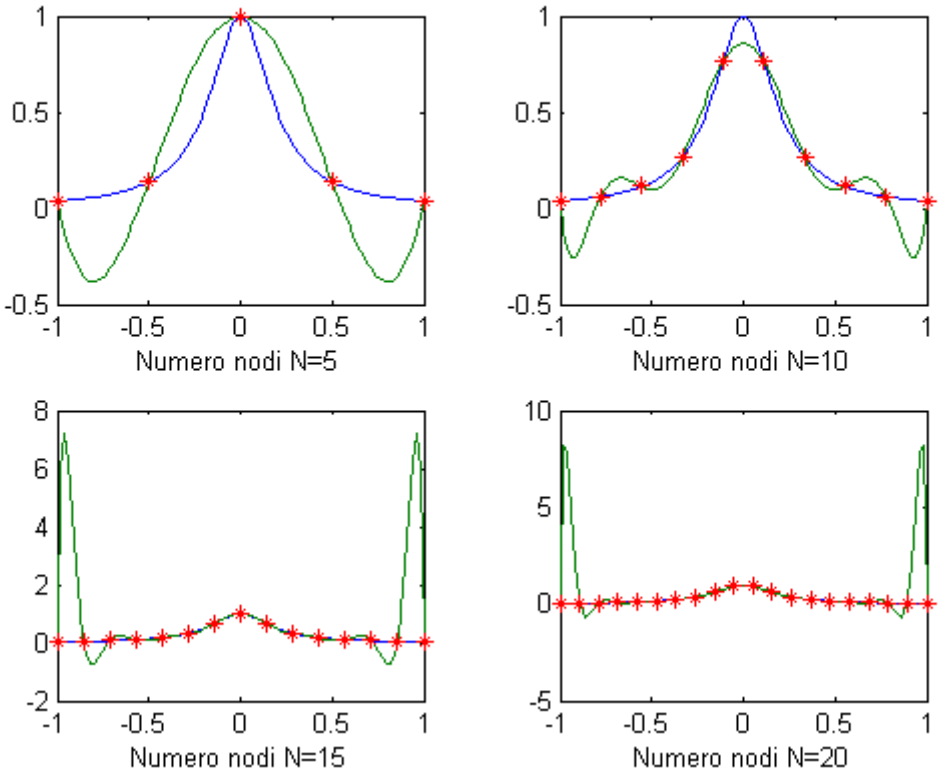

```
t=[-pi:0.01:pi];
x=cos(t);
y=sin(t);
subplot(1,2,1);
plot(x,y)
title('circonferenza');
xlabel('asse x ');
ylabel('asse y ');
subplot(1,2,2);
plot(x,y)
axis('square');
title('circonferenza');
xlabel('asse x ');
ylabel('asse y ');
```



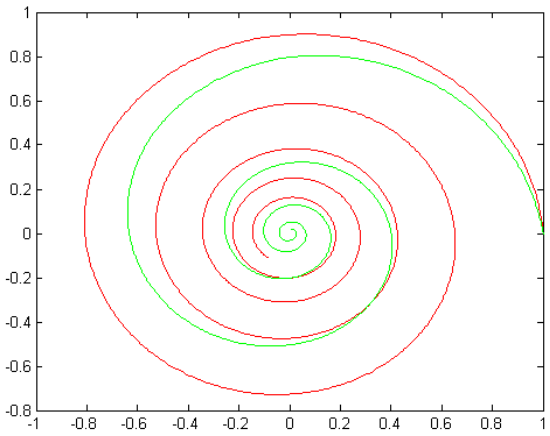



```
%runge.m
% interpolazione della funzione di Runge
% sull'intervallo [-1,1] con nodi equidistanti
%---------------------------------------
N=4;
x=linspace(-1,1,N+1);
y=1./(1+25*x.^2);
xx=linspace(-1,1);
ye=1./(1+25*xx.^2);
p=polyfit(x,y,N)
yy=polyval(p,xx);
subplot(2,2,1);
plot(xx,ye,xx,yy,'-',x,y,'*')
xlabel('N=4');
clear
N=9
x=linspace(-1,1,N+1);
y=1./(1+25*x.^2);
xx=linspace(-1,1);
ye=1./(1+25*xx.^2);
p=polyfit(x,y,N);
yy=polyval(p,xx);
subplot(2,2,2);
plot(xx,ye,xx,yy,'-',x,y,'*')
xlabel('N=9');
clear
N=14
x=linspace(-1,1,N+1);
y=1./(1+25*x.^2);
xx=linspace(-1,1);
ye=1./(1+25*xx.^2);
```

```
p=polyfit(x,y,N);
yy=polyval(p,xx);
subplot(2,2,3);
plot(xx,ye,xx,yy,'-',x,y,'*')
xlabel('N=14');
clear
N=19
x=linspace(-1,1,N+1);
y=1./(1+25*x.^2);
xx=linspace(-1,1);
ye=1./(1+25*xx.^2);
p=polyfit(x,y,N);
yy=polyval(p,xx);
subplot(2,2,4);
plot(xx,ye,xx,yy,'-',x,y,'*')
xlabel('N=19')
```



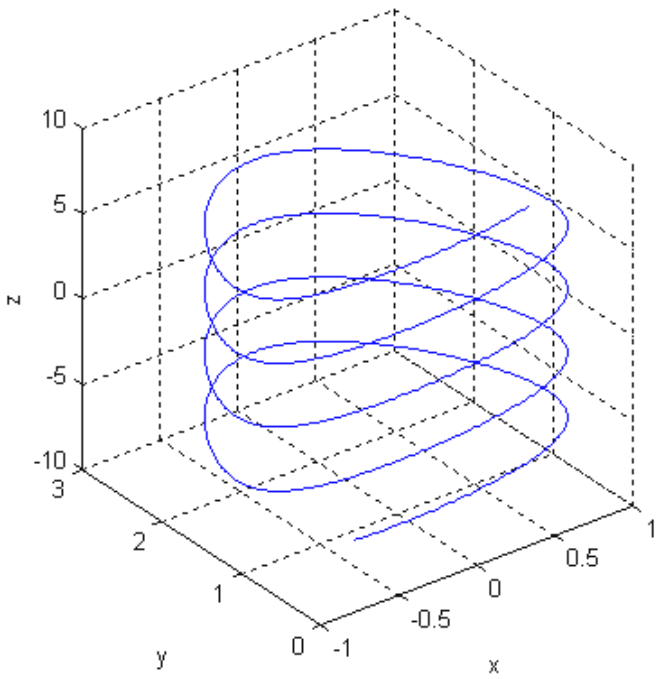

```
w1=0.2+0.8i;
y=0.1+0.9i;
m=0:0.01:20;
plot(y.^m,'r'), hold, plot(w1.^m,'g')
```



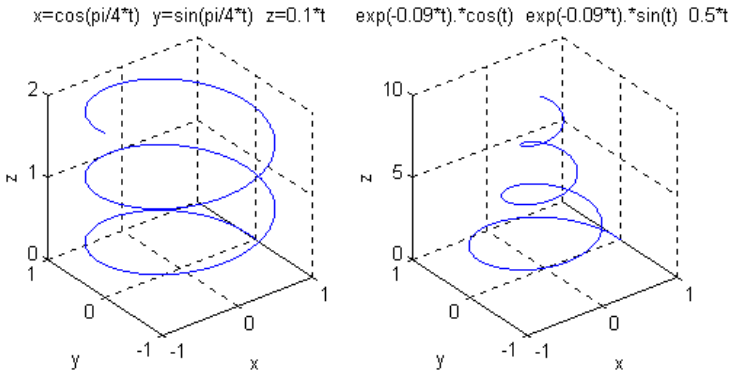
GRAFICA 3D - Comandi principali

Funzione	**Descrizione**
countour(x,y,z)	Disegna le curve di livello
mesh(x,y,z)	Disegna una superficie
meshc(x,y,z)	Come mesh, in più disegna le curve di livello sotto la superficie
meshz(x,y,z)	Come mesh, in più disegna linee di riferimento verticali sotto la superficie
surf(x,y,z)	Disegna una superficie con pannelli opachi
surfc(x,y,z)	Come surf, in più disegna le curve di livello sotto la superficie
[X,Y]=meshgrid(x,y)	Crea le matrici X e Y dai vettori x e y per definire una griglia rettangolare
[x,y]=meshgrid(x)	Equivale a [X,Y]=meshgrid(x,y)
waterfall(x,y,z)	Come mesh, ma genera linee di griglia in una sola direzione

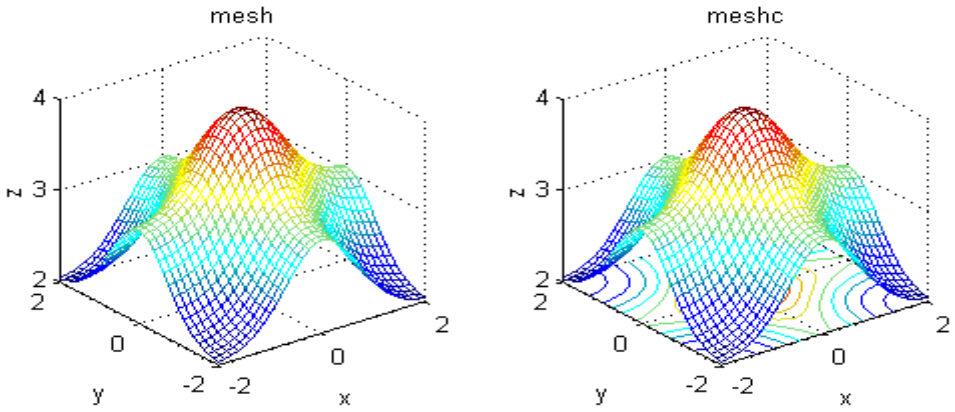
```
t=[-2*pi:0.01:2*pi];
x=sin(2*pi/3*t);
y=exp(-cos(2*pi/3*t));
z=1.25.*t;
plot3(x,y,z);
axis('square')
xlabel('x'),ylabel('y'),zlabel('z')
grid
```



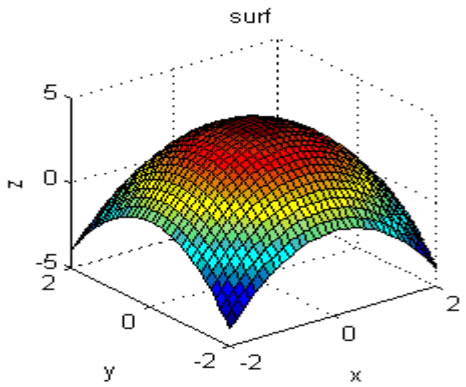

```
t=[0:0.01:20];
x=cos(pi/4*t);    y=sin(pi/4*t);    z=0.1*t;
subplot(1,2,1);    plot3(x,y,z);
title('x=cos(pi/4*t)  y=sin(pi/4*t)  z=0.1*t')
xlabel('x'),ylabel('y'),zlabel('z')
axis('square'),   grid
t=[0:0.01:20];
x=exp(-0.09*t).*cos(t);        y=exp(-0.09*t).*sin(t);      z=0.5*t;
subplot(1,2,2);    plot3(x,y,z);
title('exp(-0.09*t).*cos(t)  exp(-0.09*t).*sin(t)  0.5*t')
xlabel('x'),ylabel('y'),zlabel('z')
axis('square'),   grid
```

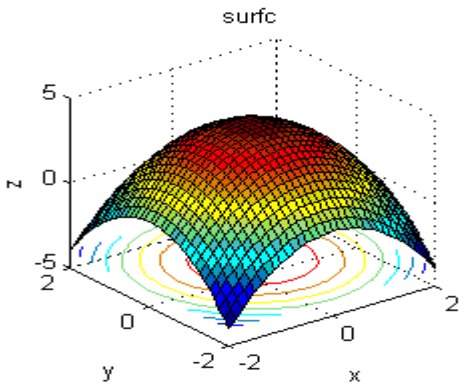


```
[x,y]=meshgrid(-2:0.125:2);
z=exp(-x.^2)+exp(-y.^2)+2;
subplot(1,2,1)
mesh(x,y,z)
axis('square')
xlabel('x'),ylabel('y'),zlabel('z')
title('mesh')
subplot(1,2,2)
meshc(x,y,z)
axis('square')
xlabel('x'),ylabel('y'),zlabel('z')
title('meshc')
```

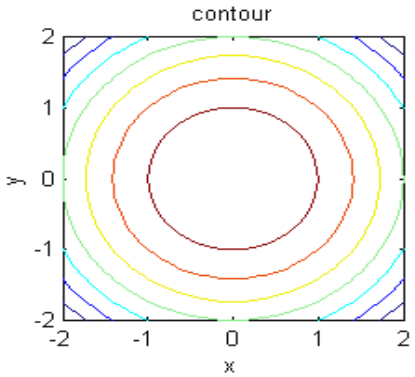


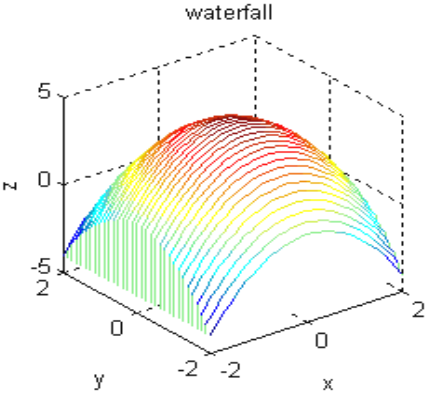

```
[x,y]=meshgrid(-2:0.125:2);
z=-x.^2-y.^2+4;
subplot(1,2,1)
surf(x,y,z)
axis('square')
xlabel('x'),ylabel('y'),zlabel('z')
title('surf')
subplot(1,2,2)
surfc(x,y,z)
axis('square')
xlabel('x'),ylabel('y'),zlabel('z')
title('surfc')
```






```
[x,y]=meshgrid(-2:0.125:2);
z=-x.^2-y.^2+4;
subplot(1,2,1)
contour(x,y,z)
axis('square')
xlabel('x'),ylabel('y'),zlabel('z')
title('contour')
subplot(1,2,2)
waterfall(x,y,z)
axis('square')
xlabel('x'),ylabel('y'),zlabel('z')
title('waterfall')
```





sommario dei comandi

help	guida in linea
lookfor	lista di tutte le voci della guida in linea che contengono una data parola chiave
doc	stampa la documentazione html
whos	elenca tutte le variabili correnti
what	elenca gli m-file
save	salva in un file su disco i valori delle variabili dello spazio di lavoro
load	Recupera da file su disco i valori di variabili salvate
clear	Cancella i valori delle variabili dello spazio di lavoro
diary	Salva su disco una versione in formato testo della sessione di Matlab in corso
clc	Cancella il contenuto della finestra dei comandi
type	Lista dei comandi di un m-file
disp	Scrive un testo o una matrice
format	Definisce il formato di uscita
tic	Accende il cronometro
toc	Spegne il cronometro e dà il tempo trascorso in secondi

caratteri speciali

%	commenti
=	assegnazione
.	punto decimale
+	addizione
-	sottrazione
*****	moltiplicazione
/	divisione
^	elevamento a potenza
.*	moltiplicazione vettorizzata
./	divisione vettorizzata
( )	indica la precedenza delle operazioni aritmetiche o racchiude gli argomenti di una funzione
[ ]	racchiude gli elementi di un vettore o di una matrice
,	separa gli elementi di un vettore o gli argomenti di una funzione
...	caratteri di continuazione di un comando su più righe
;	separa le righe di una matrice, al termine di un comando sopprime la stampa
:	colon notation
>	maggiore di
>=	maggiore o uguale a
<	minore di
<=	minore o uguale a
=	uguale a
&	and
|	or
'	trasposto
****	operatore di backslash, per risolvere i sistemi lineari
~	not

costanti predefinite

pi	pi-greco π
i, j	unità immaginaria
eps	precisione di macchina
realmax	massimo numero di macchina
realmin	minimo numero di macchina
inf	speciale costante di macchina che rappresenta ∞
NaN	Speciale configurazione di macchina che segnala una situazione anomala

matrici speciali

zeros	**Matrice nulla**
ones	Matrice di elementi uguali ad 1
eye	Matrice identica (identità)
diag	Estrae la diagonale di una matrice o costruisce una matrice diagonale
linspace	Vettore di elementi equispaziati
logspace	Vettore di elementi spaziati in scala logaritmica
rand	Matrice di elemnti random
hilb	Matrice di Hilbert
invhilb	Inversa della matrice di Hilbert
vander	Matrice di Vandermonde
pascal	Matrice di Pascal

funzioni scalari predefinite

abs	Valore assoluto o modulo
sqrt	Radice quadrata
gcd	Massimo comun divisore
lcm	Minimo comun divisore
rem	Resto della divisione fra interi
round	Arrotonda all'intero più vicino
floor	Arrotonda all'intero immrdiatamente inferiore
ceil	Arrotonda all'intero immrdiatamente superiore
factorial	Fattoriale di un intero
sign	Segno
exp	Esponenziale
log	Logaritmo naturale
log10	Logaritmo in base 10
log2	Logaritmo in base 2
sin , asin	Seno, arcoseno
cos, acos	Coseno, arcocoseno
tan , atan	Tangente, arcotangente
conj	Coniugato di un numero complesso
real	Parte reale di un numero complesso
imag	Parte immaginaria di un numero complesso

Risoluzione di equazioni

Matlab mette a disposizione delle funzioni predefinite per il calcolo delle soluzioni di equazioni

1. la funzione **fzero** calcola una soluzione di *f(x)=0*, dove *f(x)* è una funzione di una variabile. Come parametri di input si devono fornire la funzione e un'approssimazione della soluzione cercata (vedi grafico della funzione).
Ad esempio l'equazione

$$f(x)=\sin x+x^2-5x=0$$

ha una radice prossima a 5. Vogliamo approssimare tale radice

```
>> f=inline('sin(x)+x.^2-5.*x')

f =

     Inline function:
     f(x) = sin(x)+x.^2-5.*x

>> y=fzero(f,5)

y =

    5.1731
```

2. la funzione **roots** calcola tutte le radici, reali e complesse, di una equazione algebrica di grado *n* della forma

$$f(x)=a_1 x^n+a_2 x^{n-1} + ... + a_n x + a_{n+1} = 0$$

Basta fornire il vettore dei coefficienti. Toviamo come esempio le radici dell'equazione

$$x^4 - x^3 - x^2 - 2 =0$$

```
>> a=[1 -1 -1 -1 -2];
>> radici=roots(a)

radici =

   2.0000
  -1.0000
   0.0000 + 1.0000i
   0.0000 - 1.0000i
```

Abbiamo quindi due radici reali (2 e -1) e due complesse coniugate (**i** e **-i**).