MATLAB e Octave

Terza Parte

Grafici in MATLAB/Octave

Come già detto, MATLAB/Octave permette di fare dei grafici/disegni in una finestra definita con il comando figure. Per creare un grafico/disegno è necessario riferirsi ad un sistema di coordinate cartesiane floating point.

Comando	Descrizione
figure(k)	Apre una nuova figura nella finestra num. k
plot(x,y)	Genera un grafico
title('text')	Inserisce il titolo nel grafico
xlabel('text')	Aggiunge un'etichetta all'asse orizzontale
ylabel('text')	Aggiunge un'etichetta all'asse verticale
legend('text')	Inserisce una legenda
hold on	Mantiene la stessa finestra di disegno

Grafici

- Per aprire una finestra grafica si deve dare il comando figure() o figure(id) o id=figure()
- Matlab/Octave aprirà una finestra in cui verrà visualizzato l'output di successive chiamate grafiche come plot(x,y)
- (x,y) rappresentano le coordinate floating point di punti da disegnare
- MATLAB su Windows usa la libreria grafica di Windows, su Mac e Linux usa OpenGL
- Octave usa gnuplot

Grafico di funzioni e plot

Il grafico di una funzione $y=f(x)$ con $x \in [a,b]$ deve essere disegnato per punti:

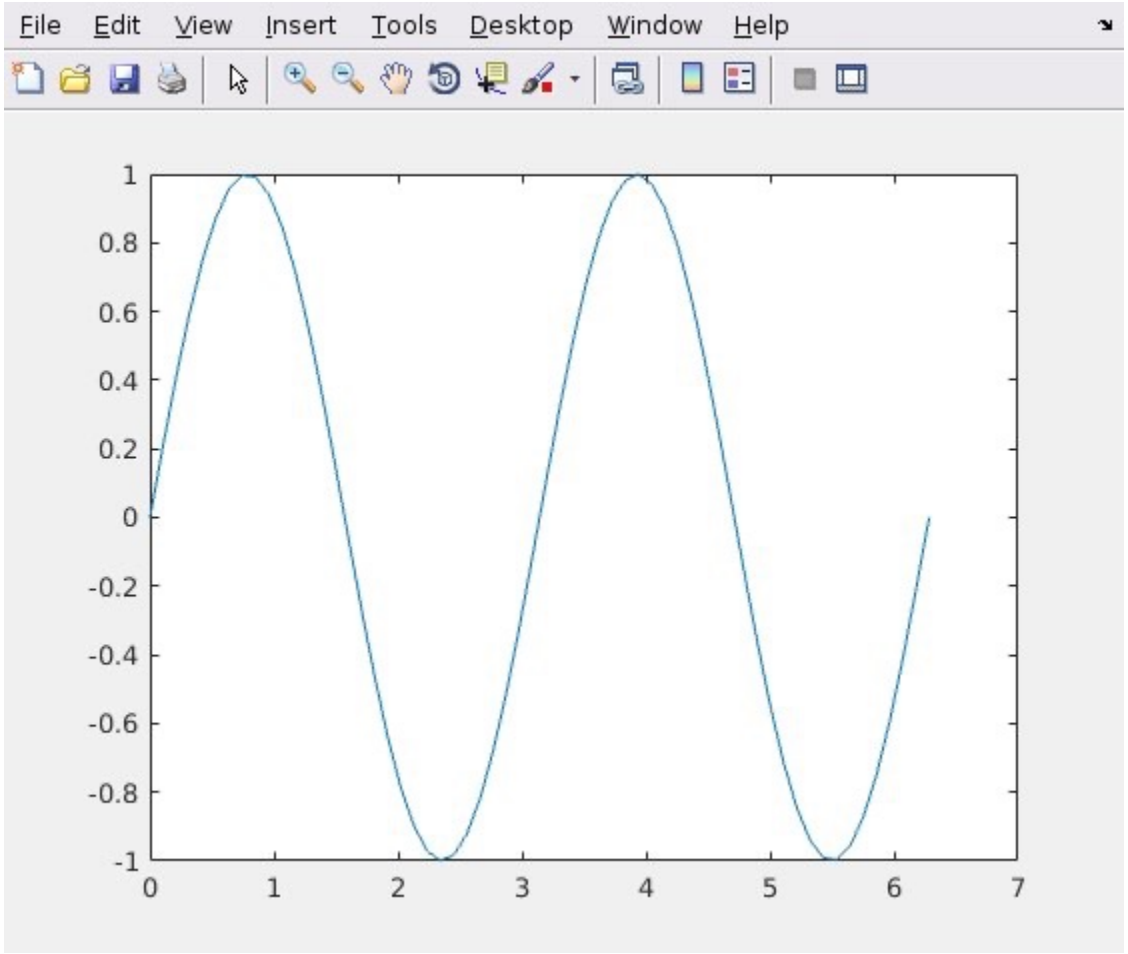
- si definisce un vettore x di ascisse nell'intervallo [a,b] (discretizzazione del dominio [a,b]), quindi si valuta la funzione f(x) su questa discretizzazione, ossia si determina il vettore y valutando i valori y(i) come f(x(i))

- si utilizza la funzione plot(x,y) per disegnare i punti del piano definiti nei vettori x ed y, cioè le coppie (x(i),y(i)) di ascisse e ordinate (per punti o con una spezzata)

Funzione plot

Esempio:

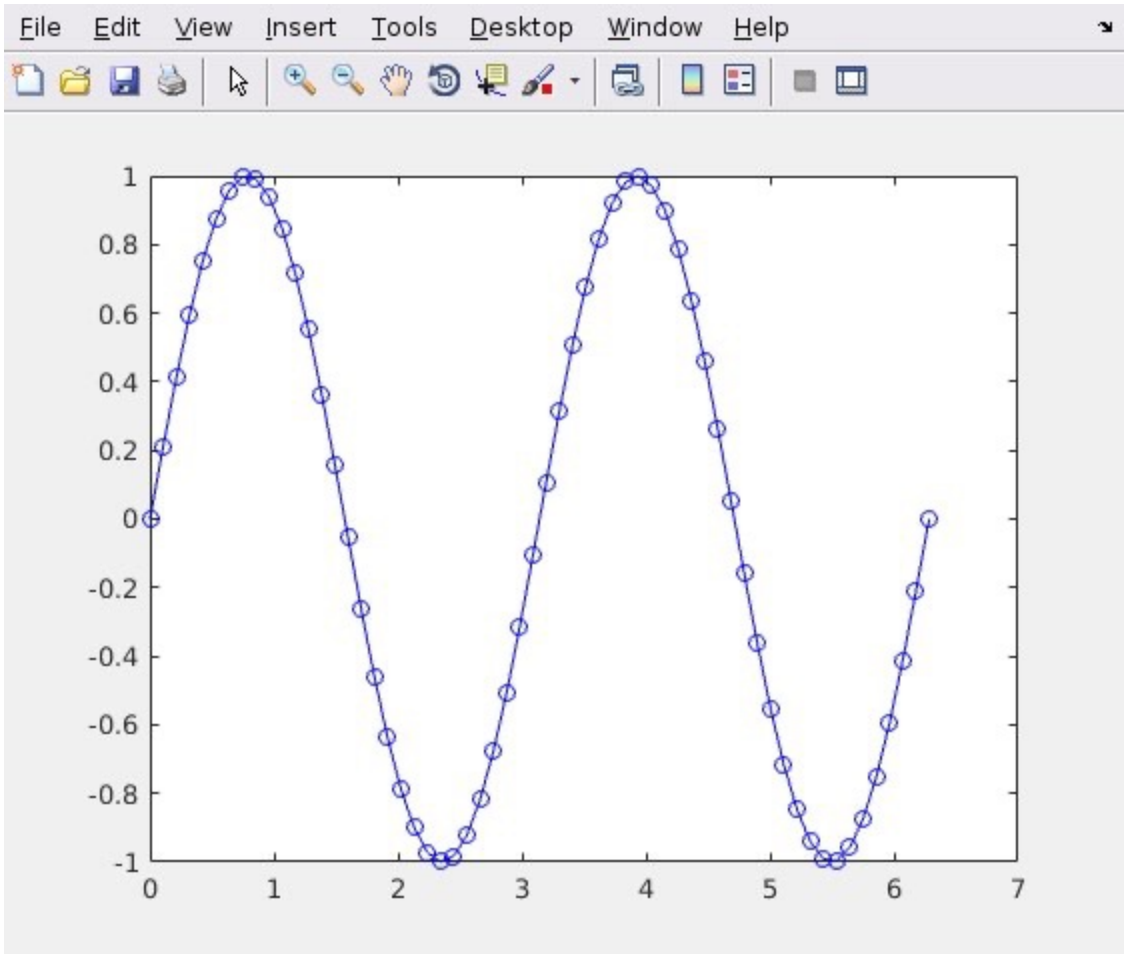
```
x = linspace( 0, 2*pi, 60);
y = sin( 2*x );
plot( x, y )
```



Funzione plot

Esempio:

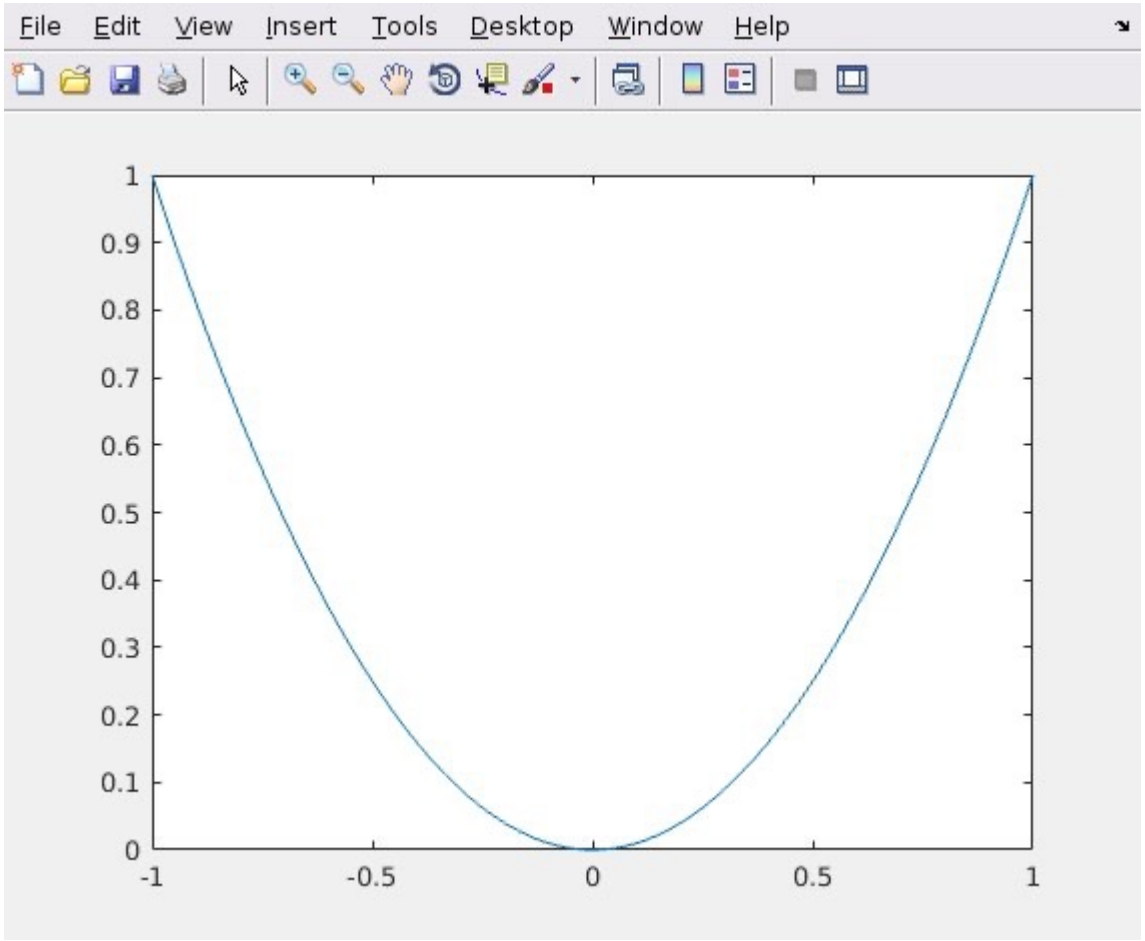
```
x = linspace( 0, 2*pi, 60);
y = sin( 2*x );
plot( x, y, 'b-o' )
```



Funzione plot

Esempio:

```
x = linspace( -1, 1, 100 );
y = x .^ 2;
plot( x, y )
```

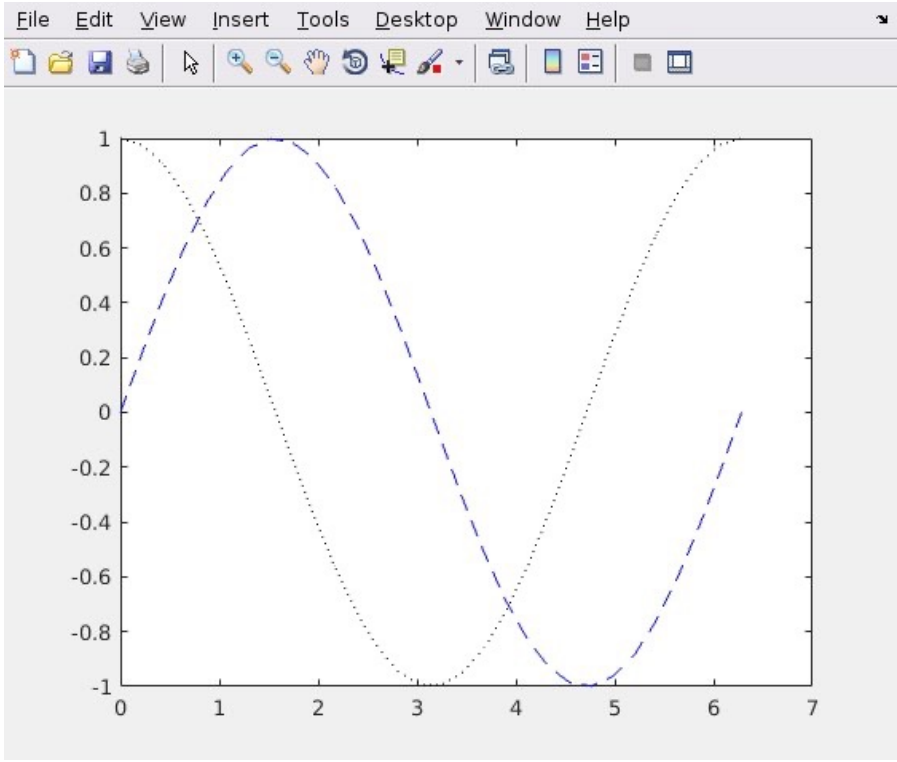


Funzioni hold on/off

hold on: comando che attiva una modalità per realizzare più grafici nella stessa figure.

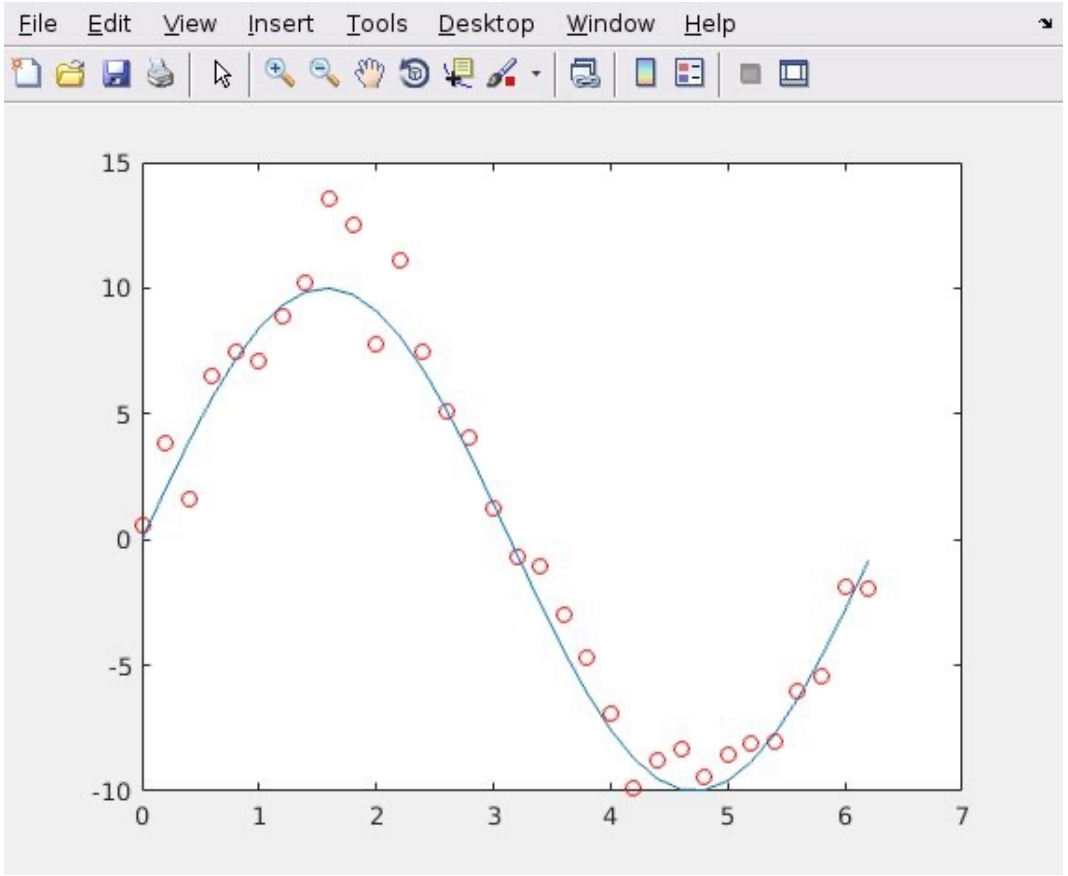
hold off: disattiva la modalità precedente.

```
x = linspace( 0, 2*pi, 30 );
plot( x, cos(x), ‘k:’ )
hold on
plot( x, sin(x), 'b--’ )
hold off
```



Esempi hold on/off

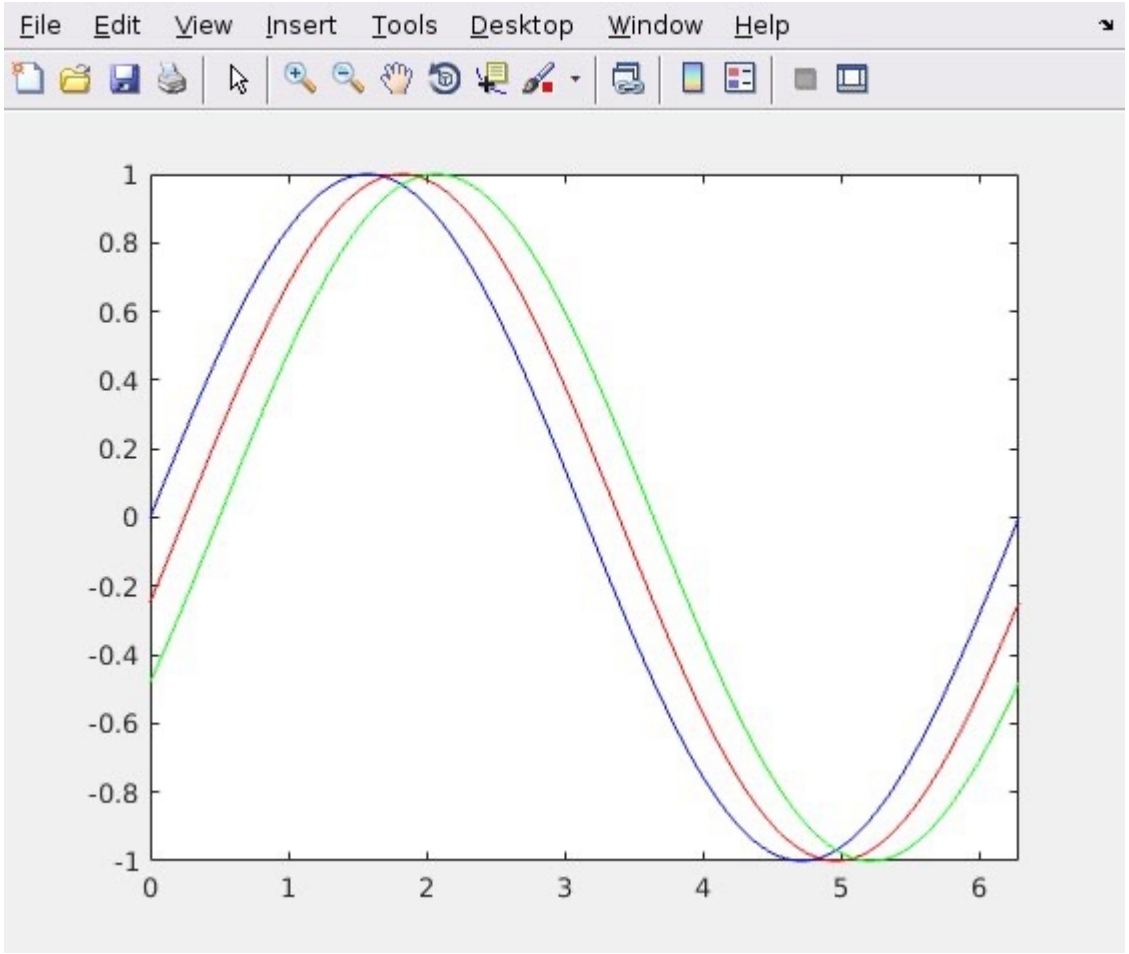
```
x =0 : 0.2 : 2.*pi;
y = sin( x );
yr = randn(size( x ));
plot( x, 10.*y+yr, 'ro')
hold on
plot( x, 10.*y )
```



Comando hold

Quindi usando hold on si possono generare grafici multipli, cioè sovrapposti:

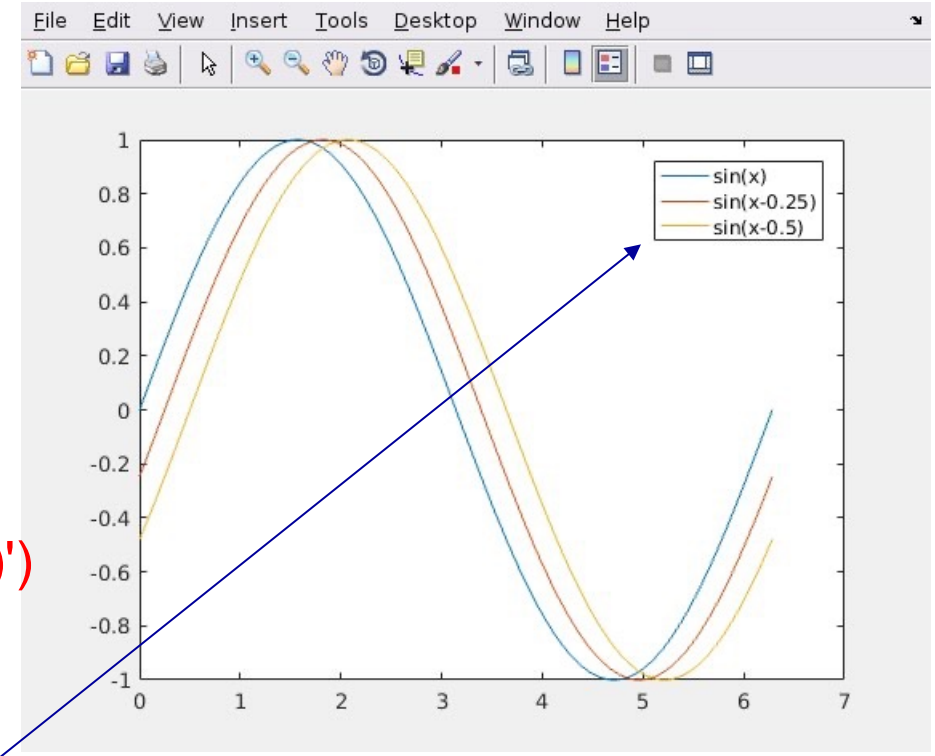
```
x = 0 : pi/100 : 2*pi;
y1 = sin(x);
plot(x,y1,'b');
hold on;
y2 = sin(x - 0.25);
plot(x,y2, 'r');
y3 = sin(x - 0.5);
plot(x,y3, 'g');
axis tight;
hold off;
```



Esempio grafici multipli

Diverse coppie di dati x-y creano grafici multipli con una singola chiamata alla plot. Per esempio, queste istruzioni disegnano 3 funzioni di x, ciascun grafico viene rappresentato con un colore diverso

```
x = 0:pi/100:2*pi;
y = sin(x);
y2 = sin(x-.25);
y3 = sin(x-.5);
plot(x,y,x,y2,x,y3)
legend('sin(x)','sin(x-.25)','sin(x-.5)')
```



L'istruzione legend fornisce un modo semplice per identificare i grafici.

Esempio label e title

```
time=[0:0.01:6];
y=sin(time);
```

Genera dati

```
plot(time,y);
xlabel('time');
ylabel('sin(time) ');
title(' plot of sin(time) ');
grid on;
```

Plot di y versus time
time è in asse x, Y è in asse y

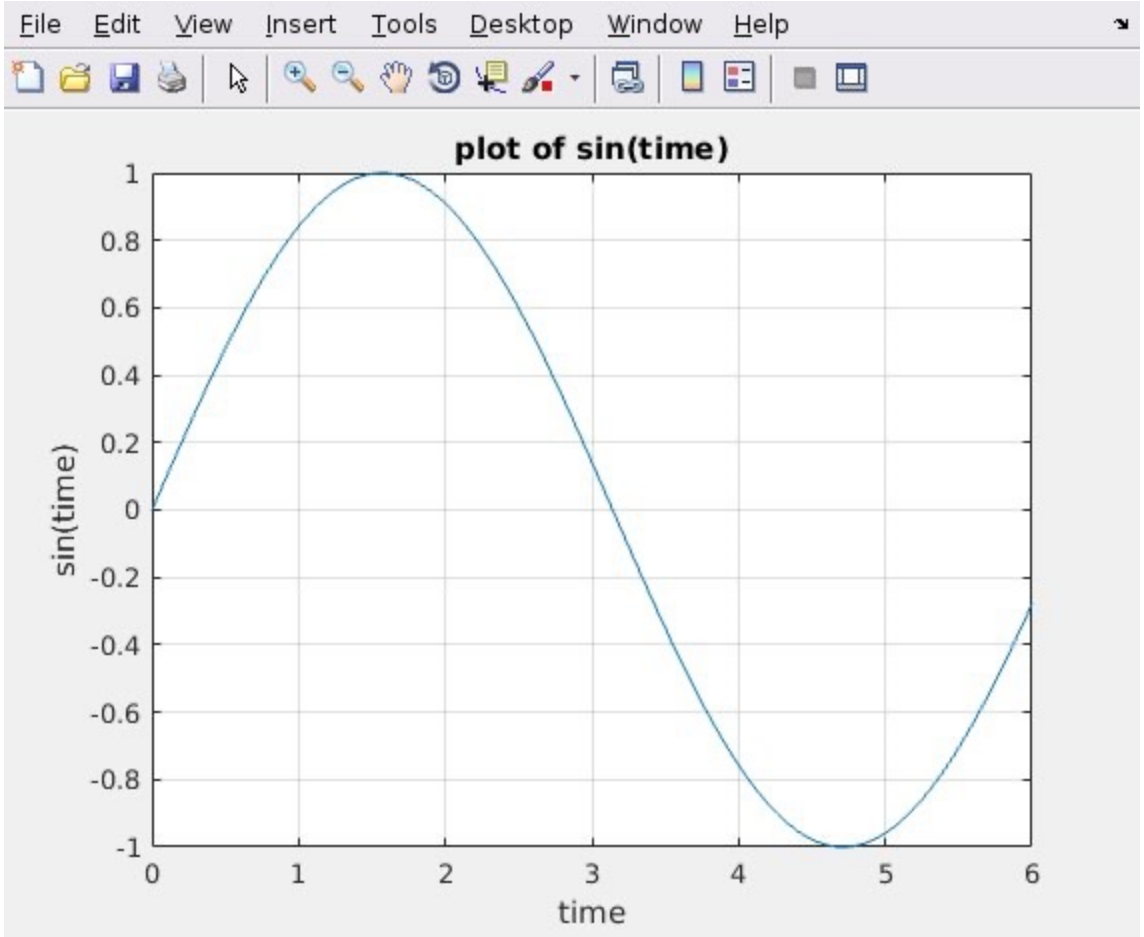
etichetta asse x

etichetta asse y

titolo

Aggiunge una griglia

Esempio


File Edit View Insert Tools Desktop Window Help
plot of sin(time)
sin(time)
time
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
1
2
3
4
5
6

Toolbox anmglib_3.0

In questo corso per fare disegni e grafici utilizzeremo la libreria anmglib_3.0 che usa opportunamente le function grafiche messe a disposizione da Matlab/Octave.

Per usare la libreria, dalla cartella di lavoro matlab_mnc_2324_x dare sempre all'inizio della sessione di lavoro il comando

>>add_path

si tratta di uno script presente nella cartella di lavoro

Toolbox anmglib_3.0

Un Toolbox consiste di funzioni scritte in linguaggio Matlab/Octave che possiamo utilizzare durante le esercitazioni semplicemente richiamandole per nome e con opportuni argomenti, come abbiamo visto per le built-in function di Matlab/Octave

Le funzioni della libreria hanno nomi autoesplicativi su quello che fanno mediante suffissi; per esempio la funzione

point_plot (…) disegna un punto, ma anche un array di punti 2D o 3D

vect2_plot (…) disegna un vettore 2D

Toolbox anmglib_3.0

Utilizzare il comando:

>>help_anmglib3 suffisso

per conoscere le funzioni a partire dai suffissi:

plot --> funzioni che producono un disegno

vect2 --> funzioni che elaborano vettori 2D

trans --> funzioni che gestiscono trasformazioni affini

curv2 --> funzione che elabora curve 2D

bezier, spline, nurbs --> elaborano curve o superfici di Bezier, spline e nurbs

solid --> elaborano solidi

c2, c3, s --> function che definiscono curve 2D, 3D e superfici

cp2, cp3 --> valore e derivata prima di curve 2D e 3D

Esercizio B.1

Definire una lista di punti 2D per rappresentare un quadrato con un vertice nell'origine e lato 2 e disegnarlo insieme agli assi del sistema cartesiano. Lo script si chiami ssquare.m

- Chiamare la funzione per aprire una finestra grafica
- Chiamare la funzione per disegnare gli assi
- Definire una lista di punti 2D vertici del quadrato
- Chiamare la funzione per disegnare la lista dei punti e dei segmenti che li collegano nell'ordine

Creare uno script

Usare l'Editor per scrivere il seguente script:

```
%script di esempio
close all
open_figure(1);
axis_plot(3,0.25);
P=[0 0; 2 0; 2 2; 0 2; 0 0];
point_plot(P,'b-o',1.5,'r');
```

- Salvare il programma usando 'salva con nome' e dando 'ssquare' come nome
- Salvare il file nella cartella mnc_2324_3
- Se la cartella non è riconosciuta da MATLAB, per eseguire lo script bisogna renderla corrente con il Browse for Folder

Eseguire lo script

Dalla Command Window dare il comando:

```
>>ssquare
```

- MATLAB cerca nella cartella corrente se esiste un M-file con questo nome
- Se lo trova lo carica in memoria e lo esegue, cioè esegue istruzione per istruzione il contenuto dello script e riporta il risultato.

Nel caso specifico apre una finestra grafica e disegna gli assi cartesiani e un quadrato.

Attenzione

Se abbiamo scritto male qualche comando Matlab segnalerà degli errori sulla command window; nostro compito è capire il messaggio, correggere lo script, risalvarlo e rieseguirlo, fino a che Matlab esegua senza errori.

Una volta prodotto un risultato o un disegno dovremo valutare se è corretto in base a quello che ci aspettavamo; lo script può eseguire senza segnalare errori, ma produrre risultati errati.

Attenzione

Prima di procedere, usiamo il comando help di Matlab seguito dal nome di una funzione per capire cosa fanno le funzioni della libreria che abbiamo usato nello script:

```
>>help open_figure
>>help axis_plot
>>help point_plot
```

Attenzione

Una volta ottenuto uno script che esegue correttamente e produce risultati corretti, lo si usi per effettuare altre esecuzioni con dati differenti;

Per esempio, sullo script ssquare si possono modificare i parametri delle funzioni chiamate; per esempio nella point_plot si possono dare fino a 5 parametri e per la precision nell'ordine:

LineSpecification, LineWidth, MarkerEdgeColor, MarkerFaceColor e MarkerSize

```
point_plot(P,'b-o',1,'r','r',0.25);
```

Sintassi per LineSpecification

Colori		Marker		Specif. linea	
b	blue	.	point	-	solid
g	green	o	circle	:	dotted
r	red	x	x-mark	-.	dashdot
c	cyan	+	plus	--	dashed
m	magenta	*	star	(none)	no line
y	yellow	s	square		
k	black	d	diamond		
		v	triangle (down)		
		^	triangle (up)		
		<	triangle (left)		
		>	triangle (right)		
		p	pentagram		
		h	hexagram		

Provate da soli

Seguendo il foglio dell'
Esercitazione 3
realizzare gli script richiesti
al punto B

Attenzione

Si suggerisce di iniziare ogni script con i seguenti comandi Matlab:

clear all : pulisce il work space

close all : chiude tutte le finestre grafiche

clc : pulisce la command window