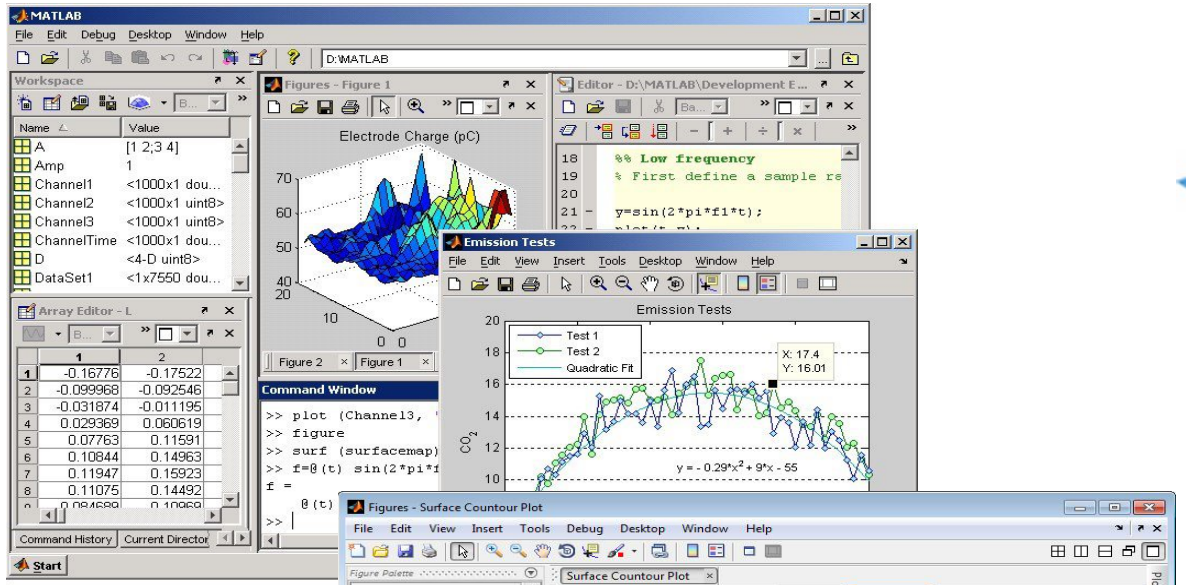
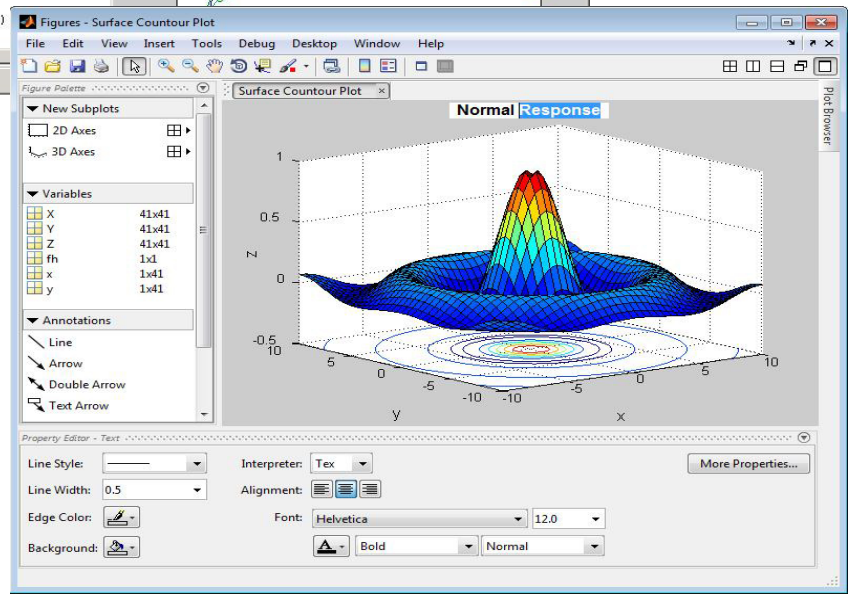
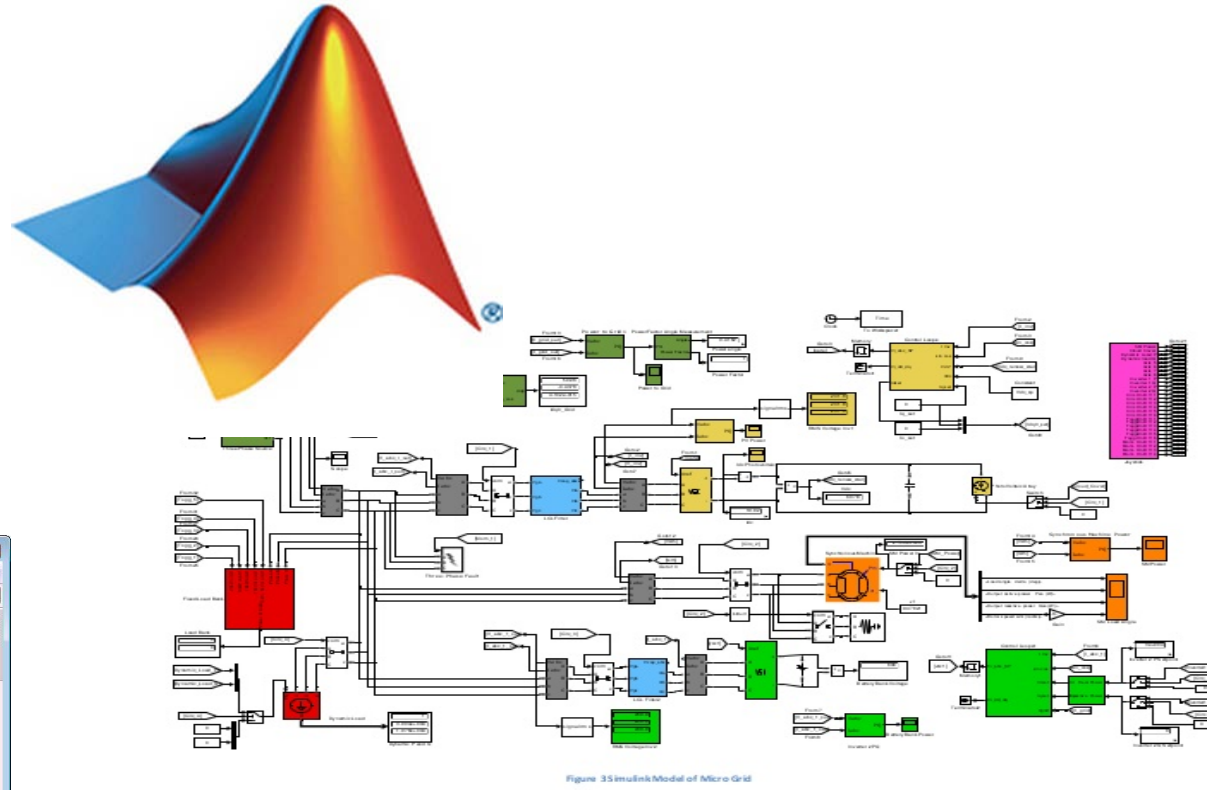
Introduzione a *Matlab* e *Simulink*







Autore: Simone Ciotti, Centro di Ricerca "E.Piaggio", UNIPI
e-mail: simone.ciotti@centropiaggio.unipi.it

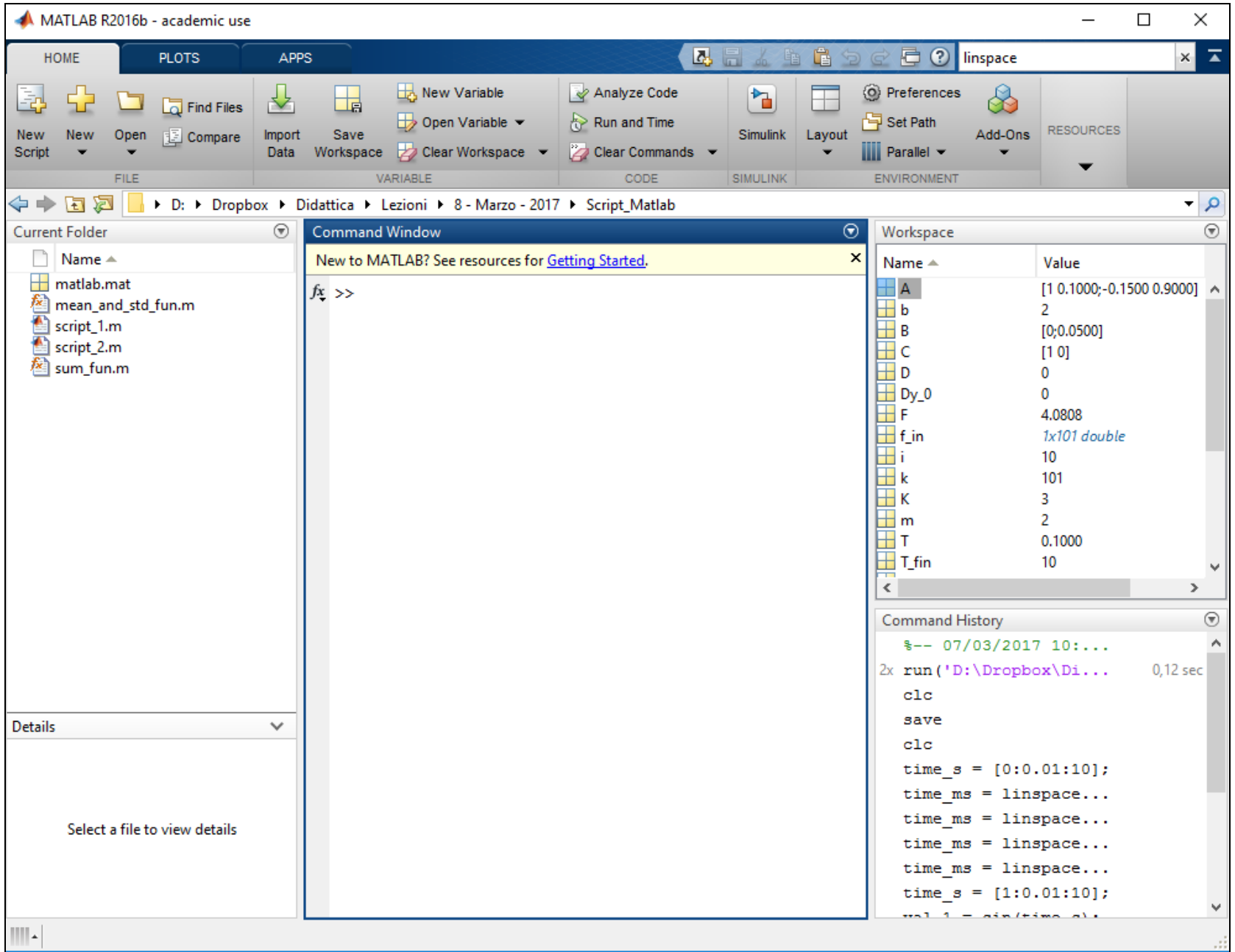
Cosa è *Matlab*?

- Matlab (MATrix LABoratory) è un ambiente di programmazione per applicazioni scientifiche, di analisi numerica, per la simulazione di sistemi dinamici e per la realizzazione di controllori
- L'elemento base di Matlab è la **matrice**
- Matlab contiene:
  - Un vasto set di funzioni di base general purpose e la possibilità di definire nuove funzioni
  - Estensioni application oriented (toolboxes), ad esempio il **Control System Toolbox**
  - Un'interfaccia grafica interattiva per la modellazione e la simulazione di sistemi dinamici → **Simulink**

Perché *Matlab*?

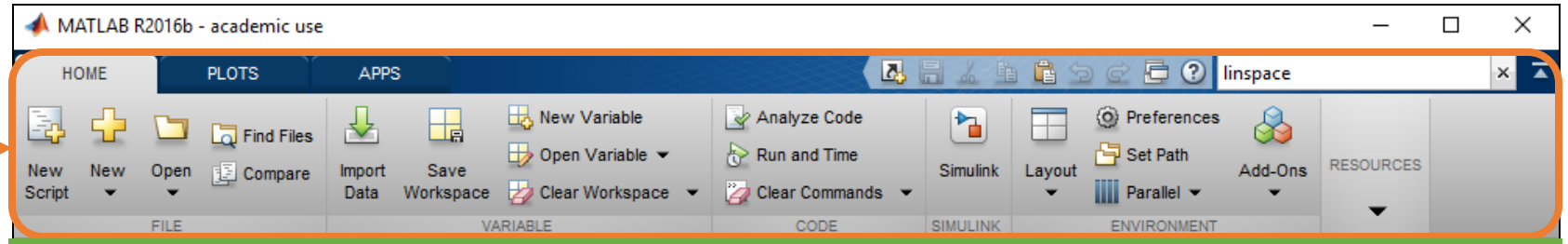
- Per il corso:
  - strumento utile per la verifica personale dei concetti appresi, per la verifica degli esercizi e come approfondimento
  - programma utilizzato in fase di esame per l'analisi del sistema da controllare e la sintesi di un controllore che riesca a soddisfare le richieste di progetto
- Come ingegneri:
  - ambiente di sviluppo software utilizzato nelle aziende e nei centri di ricerca per il progetto e sviluppo di sistemi di controllo, per l'analisi dei dati catturati tramite degli esperimenti, e molto altro …

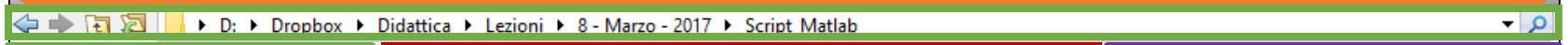
L’interfaccia Grafica di *Matlab*

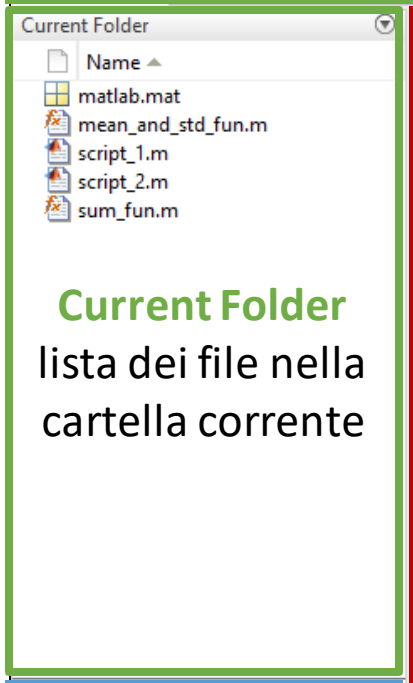


L'interfaccia Grafica di *Matlab*

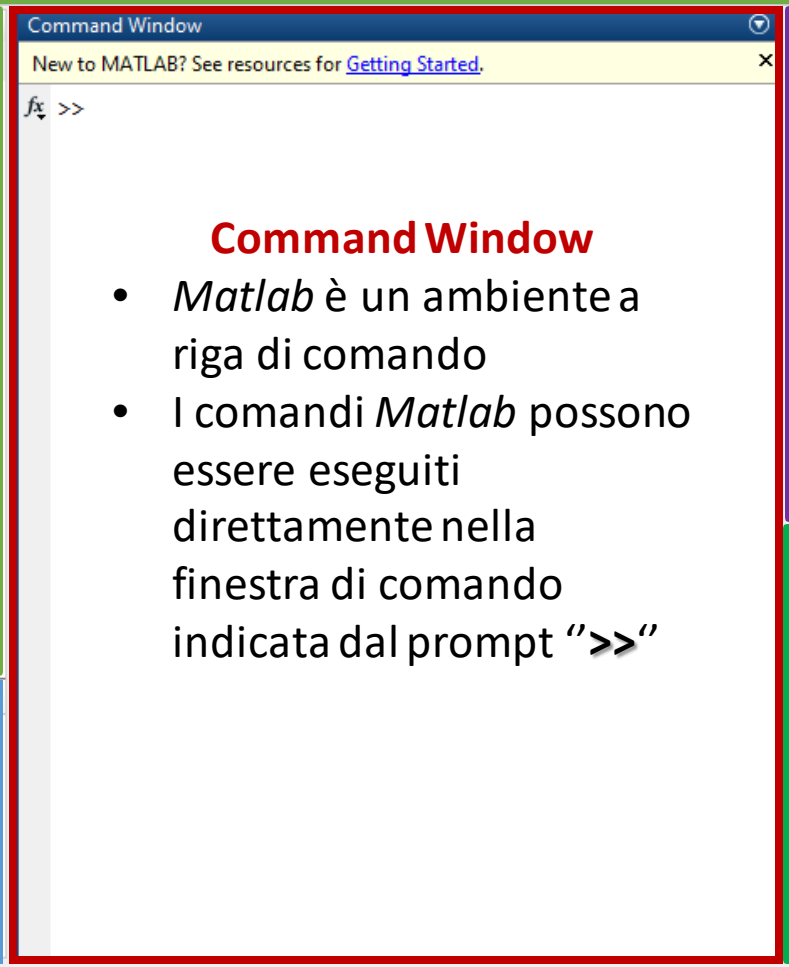
Accesso rapido ad equivalenti comandi eseguibili tramite la Command Window





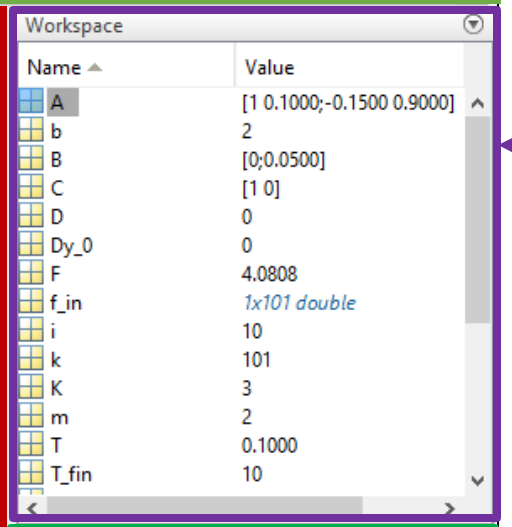


Current Folder
lista dei file nella cartella corrente

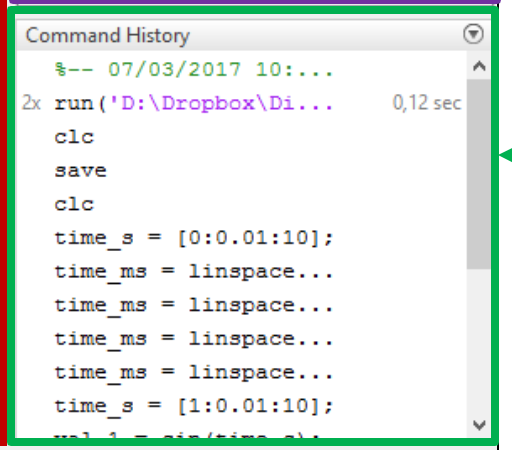


Command Window

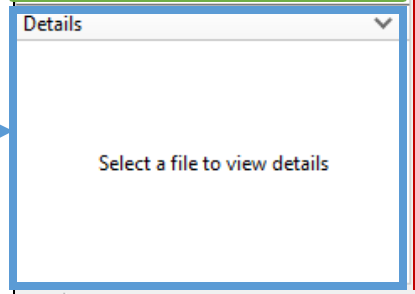
- *Matlab* è un ambiente a riga di comando
- I comandi *Matlab* possono essere eseguiti direttamente nella finestra di comando indicata dal prompt "**>>**"



Workspace
lista delle variabili create o importate da file



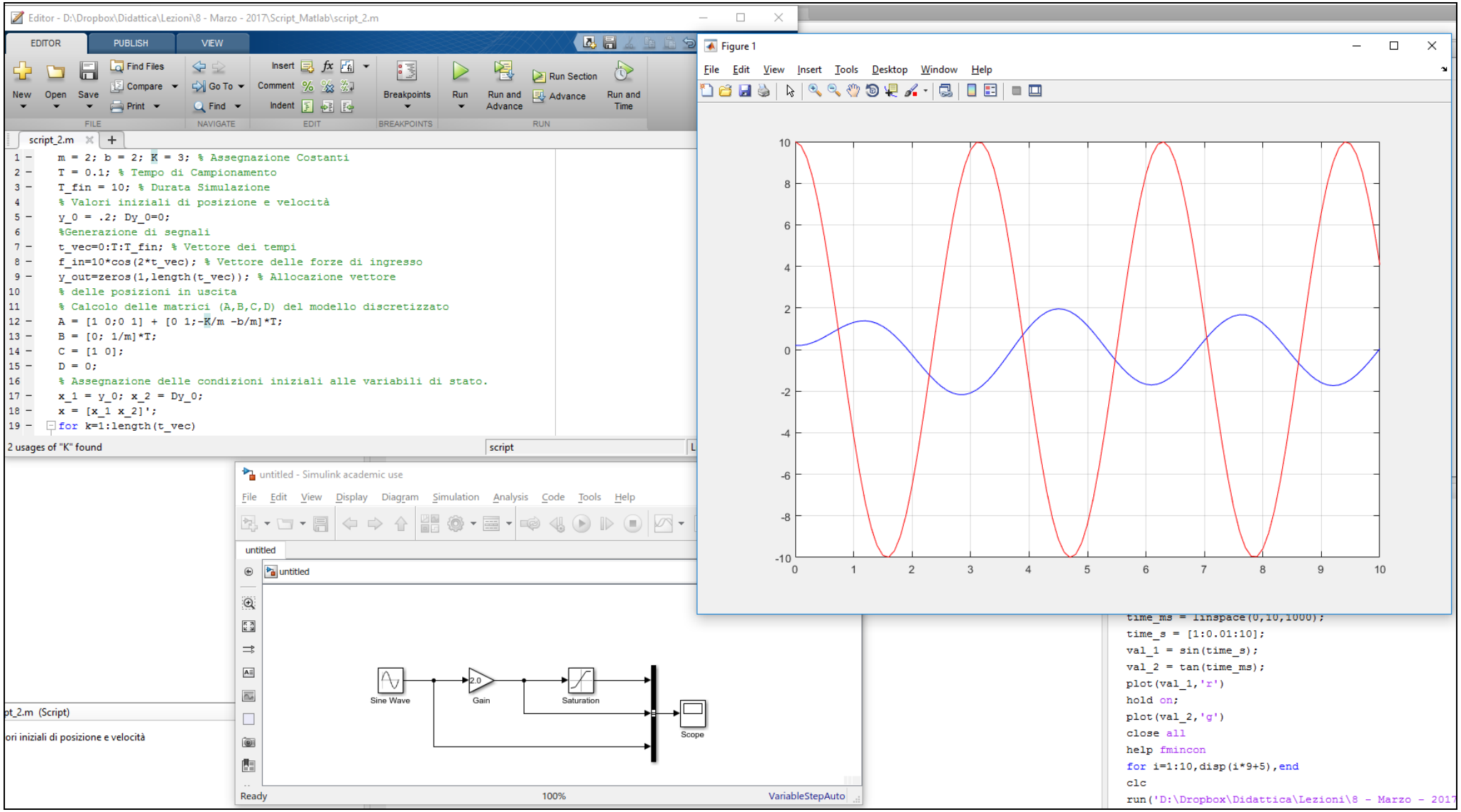
Command History
storico dei comandi eseguiti tramite la Command Window



Details
dettagli relativi al file selezionato tramite il pannello Current Folder

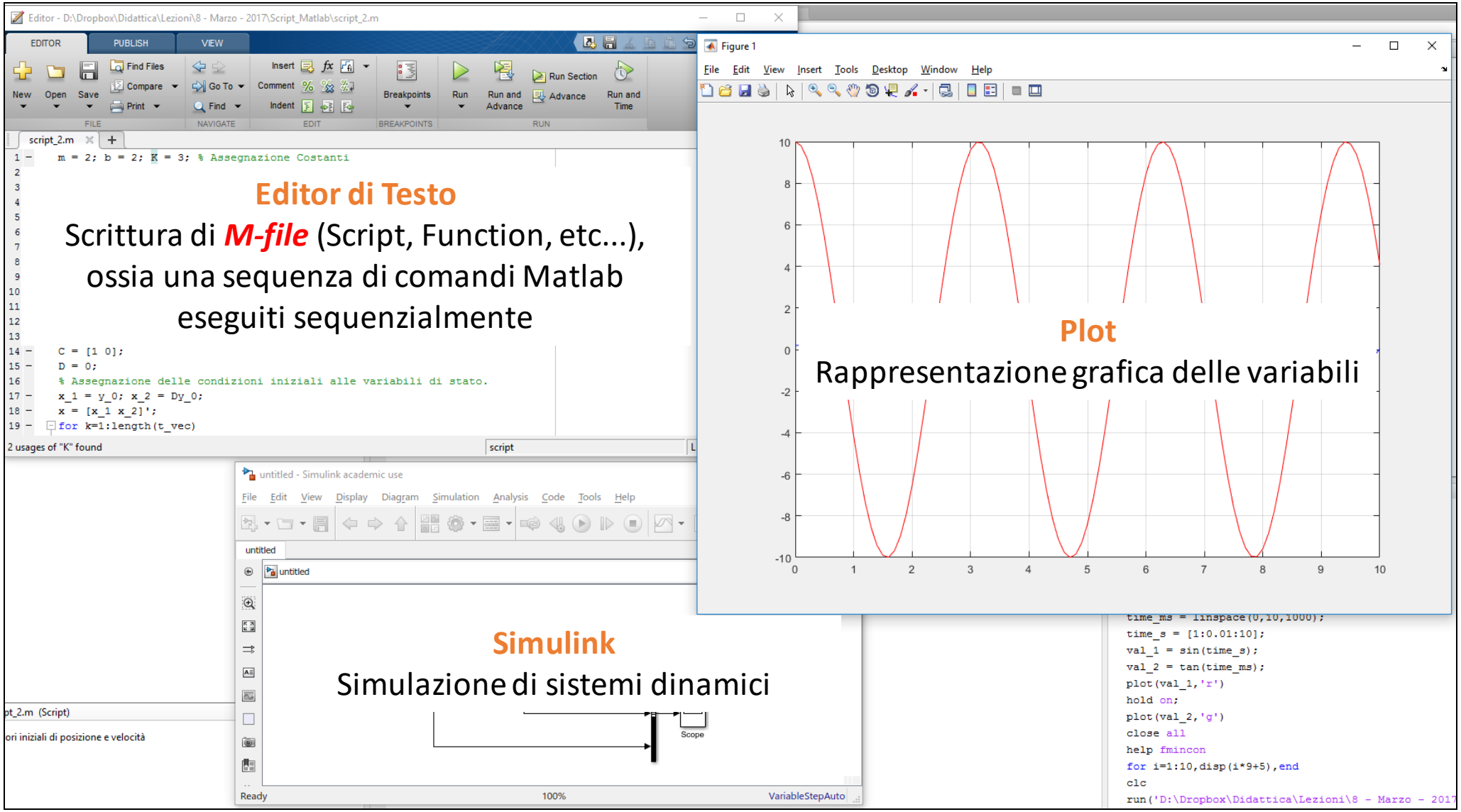
L'interfaccia Grafica di *Matlab*

Interfaccia a finestre



L'interfaccia Grafica di *Matlab*

Interfaccia a finestre



L'*help* di *Matlab*

- **Help in linea**: è sufficiente scrivere nella *Command Window* ''***help <nome_funzione>***'' per avere informazioni dettagliate sul funzionamento della funzione

- **Matlab Help Window**: si esegue dal menu Help e contiene informazioni su tutte le funzionalità di Matlab, Simulink e i Toolbox. L'help dettagliato è disponibile in formato Html e Pdf.

Matlab come calcolatrice

- Valutare espressioni aritmetiche (tramite il prompt dei comandi)
  - Valutare $\sqrt{2} + 4 + \sin(0.2\pi) + e^2$

```
>> sqrt(2) + 4 + sin(0.2*pi) + exp(2)
ans =
   13.3911
```

- ***>> help elfun*** : lista delle funzioni matematiche elementari
- ***>> help elmat*** : lista delle funzioni elementari per la manipolazioni di matrici

N.B. Matlab è case-sensitive!!!

Definizioni di Variabili

- E’ possibile definire variabili ed espressioni complesse:

```
>> a = 4; b = 2;
>> a*b
ans =
     8
```

- Per cancellare una variabile (es. a):

```
>> clear a
```

Per cancellare TUTTE le VARIABILI:

```
>> clearvars
```

Per cancellare TUTTE le STRUTTURE presenti nell’attuale workspace:

```
>> clear all
```

Definizioni di Variabili

Le *matrici* sono l'*elemento* di *base* in *Matlab*

- *Ogni oggetto* in *Matlab* è trattato come una matrice (*matrix*)
- Gli *scalari* sono considerati *casi particolari di matrici* (matrici 1x1)
- Le *matrici colonna* (o *riga*) sono dette *vettori* (vectors)
- E' possibile definire, modificare, visualizzare e eseguire operazioni e funzioni su matrici

Il Workspace

- Ogni variabile viene conservata in memoria nel *workspace*
- Per workspace si intende l'insieme di tutte le variabili definite

- >> who : lista di tutte le variabili del workspace

- >> whos <nome_variabile> : fornisce informazioni (tipo, dimensione, nome, numero di byte occupati in memoria) per la variabile specificata

- >> whos : fornisce informazioni per TUTTE le variabili del workspace

Il Workspace

- >> save <file_name> <var1> <var2> … <varN> : salva le variabili specificate nel file <file_name>.mat
- >> save <file_name> : salva TUTTE le variabili nel file <file_name.mat>
- >> save <var1> <var2> … <varN> : salva le variabili specificate nel file matlab.mat
- >> save : salva TUTTE le variabili nel file matlab.mat

N.B. Usando il formato riportato il file viene salvato nella cartella corrente

Il Workspace

- >> load <file_name> : carica, nel workspace, TUTTE le variabili contenute nel file <file_name>.mat

- >> load <file_name> <var1> <var2> … <varN> : carica, nel workspace, dal file <file_name.mat> le variabili specificate

N.B. Usando il formato riportato il file deve essere presente nella cartella corrente

Inserimento di Matrici

- Inserire una matrice 4x4

```
>> A = [16, 3, 2, 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
A =
    16     3     2    13
     5    10    11     8
     9     6     7    12
     4    15    14     1
```

- *Matlab* mostra la matrice appena inserita
- Inserendo ‘’;’’ alla fine dell’istruzione, non viene visualizzato il risultato dell’istruzione a schermo
  - Utile nel caso di risultati ‘’lunghi’’ (es. matrici di dimensione elevata)

Inserimento di Matrici

- Vettori riga:

```
>> riga1 = [1 2 3 4]
riga1 =
     1     2     3     4
>> riga2 = [1, 2, 3, 4]
riga2 =
     1     2     3     4
```

- Vettori colonna:

```
>> colonna = [1; 2; 3; 4]
```

Operazioni Matriciali

In *Matlab* TUTTE le operazioni sono relative a matrici

- Determinante: det(A)
- Autovalori: eig(A)
- Inversa: inv(A)
- Pseudo-Inversa: pinv(A)
- Diagonale: diag(A)
- Dimensioni di A
  - Numero righe e colonne: size(A)
  - Numero righe: size(A,1)
  - Numero colonne: size(A,2)
  - Lunghezza di un VETTORE: length(b)

Alcune matrici utili…

- Matrice di uno: ones(r,c)
- Matrice di zero: zeros(r,c)
- Matrice identità: eye(n)
- Matrice vuota: X = []

Operazioni Matriciali

- Operatori fondamentali:
  - + : addizione
  - - : sottrazione
  - * : moltiplicazione
  - / : divisione a *destra*
  - \ : divisione a *sinistra*
  - ^ : elevamento a potenza
  - ‘ : trasposizione

Accedere agli Elementi di una Matrice

- Singolo elemento

```
>> A(1,2)
3
>> A(end,end)
1
```

- Sottomatrice

```
>> A(1:3 , 2:4)
ans =
3     2     13
10    11    8
6     7     12
```

- Selezionare la prima riga

```
>> A(1, :)
ans =
16    3    2    13
```

- Selezionare la prima colonna

```
>> A(:,1)'
ans =
16    5    9    4
```

```
A =
    16    3    2    13
    5     10   11   8
    9     6    7    12
    4     15   14   1
```

Sottomatrici (es. 2° e 4° riga)

```
>> A([2,4],:)
ans =
5     10    11    8
4     15    14    1
```

- Per accedere a intere righe (colonne) di una matrice, si usa il wildcard ‘’**:**’’
- Per accedere all’ultimo elemento di una matrice si usa il wildcard ‘’**end**’’

Operazioni Matriciali: *Addizione* e *Sottrazione*

- Per sommare due matrici di ***uguali dimensioni*** A e B e mettere il risultato in una terza matrice C basta scrivere
  >> C = A + B;
- ***Non è possibile sommare matrici di dimensioni diverse.***
- Quanto detto vale anche per l'operazione di ***sottrazione***
- Le operazioni di addizione e sottrazione sono eseguite ***elemento per elemento***

$$\begin{bmatrix} 2 & 5 \\ 7 & 9 \end{bmatrix} + \begin{bmatrix} 1 & 6 \\ 2 & 3 \end{bmatrix} = \begin{bmatrix} 3 & 11 \\ 9 & 12 \end{bmatrix}$$

Operazioni Matriciali: *Moltiplicazione*

- La moltiplicazione tra due matrici è intesa ***righe per colonne***.

$$A[m * n] * B[n * p] = C[m * p]$$

$$A[m * n] * B[q * p] = error$$

- La ***moltiplicazione*** di una ***matrice*** per uno ***scalare*** è ***sempre possibile*** ed è commutativa

A*3 == 3*A

Operazioni Matriciali: *Divisione*

- L'operatore "/" consente di moltiplicare a *destra* una matrice ***per la matrice inversa*** di un'altra matrice

A/B == A*inv(B)

- L'operatore "\" consente di moltiplicare a *sinistra* una matrice ***per la matrice inversa*** di un'altra matrice

A\B == inv(A)*B

- La ***divisione*** di una ***matrice*** per uno ***scalare*** è ***possibile solo tramite*** l'uso dell'operatore "/" e ponendo lo ***scalare*** come ***elemento a destra*** dell'operatore "/". Tale operazione ***corrisponde*** alla ***moltiplicazione*** della ***matrice per*** l'***inverso*** dello ***scalare***

$$A/3 == A * \frac{1}{3}$$

Operazioni Matriciali: *Elevamento a Potenza*

- L'operatore di elevamento a potenza "^" può essere utilizzato in due modi
  - La matrice A è ***quadrata***

B = A^n

ottenendo così la moltiplicazione di A per se stessa n volte

  - A è una matrice ***qualsiasi*** di dimensione m*n

B = A.^n

ottenendo l'elevazione a potenza n-essima dei singoli elementi di A

$$b_{ij} = a_{ij}^{n}$$

Operazioni Matriciali: *Trasposizione*

- L'operatore "'" consente di calcolare la trasposta di una generica matrice A di dimensione m*n

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

$$B = A' = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

Operazioni Matriciali
vs
Operazioni Elemento per Elemento

Anteponendo il punto "**.**" agli operatori "*", "/" e "^" l'operazione viene eseguita fra i singoli elementi delle matrici.

N.B. Ci deve essere coerenza fra le dimensioni delle matrici

A[m*n] .* B[m*n] = C[m*n] dove $c_{ij} = a_{ij} * b_{ij}$

A[m*n] .* B[q*p] = error

Operazioni Matriciali
vs
Operazioni Elemento per Elemento

```
>> A = [1 , 2 ; 3 , 4];
```

Operazione Matriciale

```
>> A*A
ans =
7     10
15    22
```

Operazione Elemento per Elemento

```
>> A.*A
ans =
1    4
9    16
```

I Vettori

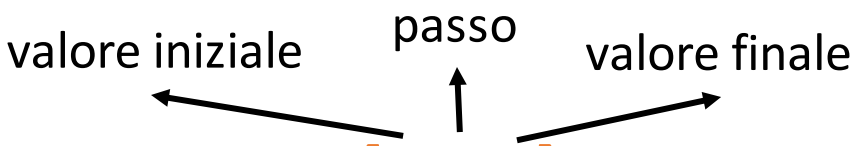
I vettori hanno due funzioni fondamentali in *Matlab:*

- Rappresentazione dei polinomi (un polinomio è descritto dal vettore dei suoi elementi)

- Rappresentazione di segnali (un segnale è rappresentato mediante la sequenza dei valori che assume in un insieme di istanti di tempo, quindi mediante vettore)

I Vettori

E’ possibile definire dei vettori con numeri a intervalli regolari




```
>> c = [1:2:8]

c =

1    3    5    7
```

```
>> d = [1:5 , 2:2:10 , 2]
d =
1   2   3   4   5   2   4   6   8   10   2
```

```
>> e = [3.2:-.1:2.5]
e =
3.2    3.1    3.0   2.9   2.8   2.7   2.6   2.5
```

Utile per

- la selezione degli elementi di una matrice
- la generazione di vettori ‘’temporali’’ equispaziati

M-file

- *Matlab* è un linguaggio di programmazione e un ambiente di calcolo interattivo
- M-file: file contenente codice Matlab
- Vengono scritti mediante un qualsiasi editor di testo ed eseguiti chiamandoli dalla linea di comando. In *Matlab* è disponibile un editor di testo interno (***Edit***)
- 2 tipi di M-file: ***script*** e ***function***

Script e Function

- Gli ***script*** vengono utilizzati per automatizzare le sequenze di comandi
  - Quando viene eseguito uno script, l'esecuzione dei comandi è del tutto equivalente alla scrittura del codice con la tastiera nella *Command Window*
    - Vengono utilizzati per evitare di riscrivere la stessa sequenza di comandi ripetutamene
  - Non hanno argomenti di input e output, tutte le variabili sono *globali* (salvate nel workspace)
- Le ***function*** si usano per estendere le funzionalità di *Matlab*
  - Normalmente generano una o più uscite (matriciali) dipendenti dai parametri in ingresso
  - Le variabili sono locali alla funzione

```
function [output] = nome_function(input)
    istruzioni;
```

- ***Il nome del file in cui viene salvata la funzione deve essere uguale a <nome_function>***

Programmare in *Matlab*

Esistono molti comandi che consentono di controllare il flusso di esecuzione del nostro script (function)

```
for index = values
    statements
end
```

```
for i = [1:10]
    a = i + 10;
    b = i – 10;
    disp(a);
    disp (b);
end
```

Programmare in *Matlab*

Esistono molti comandi che consentono di controllare il flusso di esecuzione del nostro script (function)

```
while expression
    statements
end
```

```
n = 10;
while n>0
    disp(n);
    n = n – 1;
end
```

Programmare in *Matlab*

Esistono molti comandi che consentono di controllare il flusso di esecuzione del nostro script (function)

```
if expression
    statements
elseif expression
    statements
else
    statements
end
```

```
a = 10;
if a > 0
    disp(a);
elseif a < 0
    disp(a*10);
else
    disp(0/a);
end
```

Programmare in *Matlab*

Esistono molti comandi che consentono di controllare il flusso di esecuzione del nostro script (function)

```
switch switch_expression
    case case_expression
        statements
    case case_expression
        statements
    ...
    otherwise
        statements
end
```

```
n = input('Enter a number: ');

switch n
    case -1
        disp('negative one');
    case 0
        disp('zero');
    case 1
        disp('positive one');
    otherwise
        disp('other value');
end
```

La Grafica in *Matlab*

- I grafici vengono visualizzati in ***figure***

- Il comando di base per la grafica è
  plot(y) - visualizza gli elementi del vettore y rispetto agli indici del vettore stesso
  plot(x,y) – visualizza il vettore y vs. il vettore x

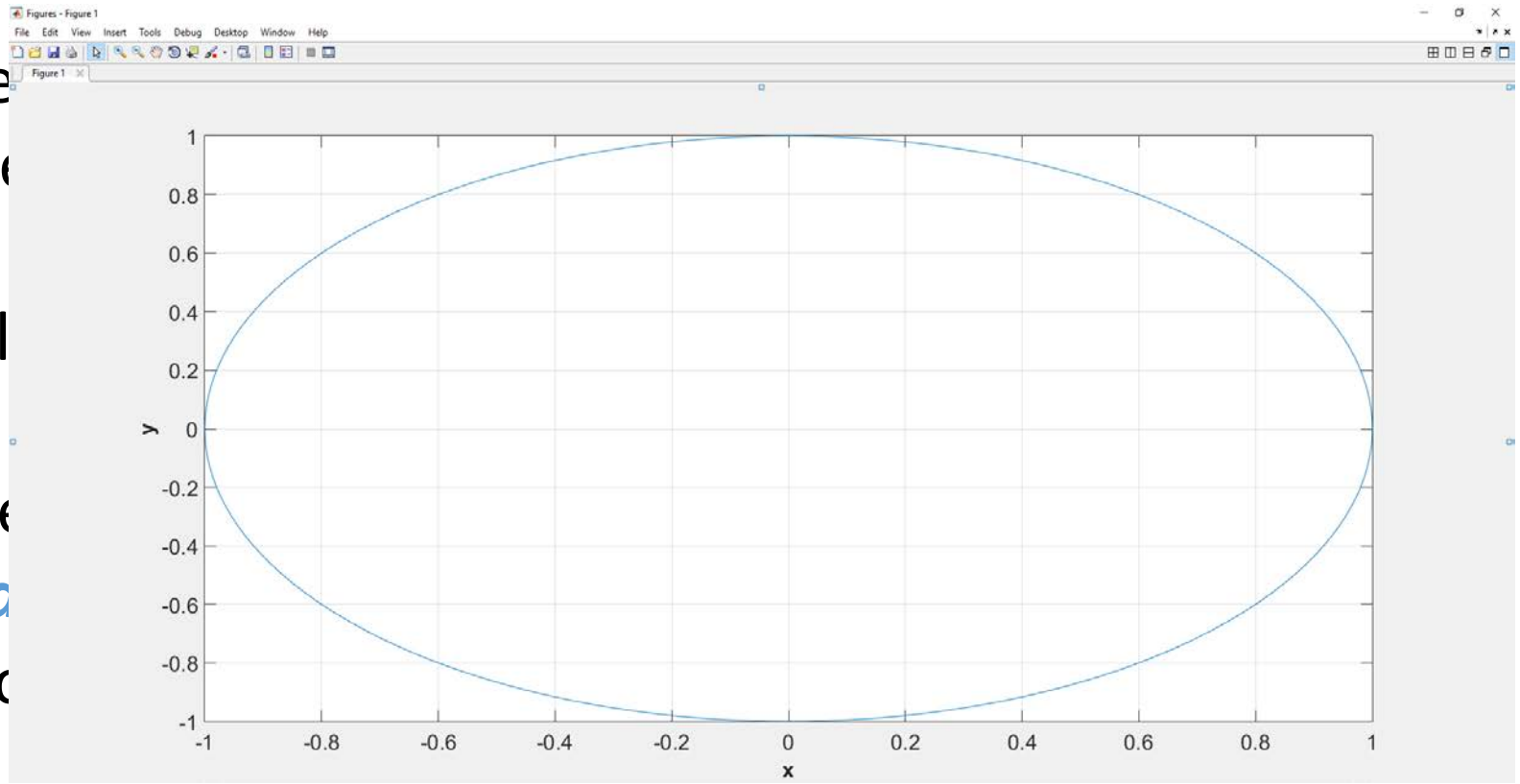
- E' possibile modificare in modo interattivo l'aspetto dei grafici
  - mediante il ***Plot Editing Mode***
  - mediante riga di comando con opportune istruzioni

La Grafica in *Matlab*

- I grafici vengono visualizzati in ***figure***

- Il comando di base pe
  plot(y) - visualizza gli e
  vettore stesso
  plot(x,y) – visualizza il

- E' possibile modificare
  - mediante il ***Plot Ed***
  - mediante riga di co



La Grafica in *Matlab*

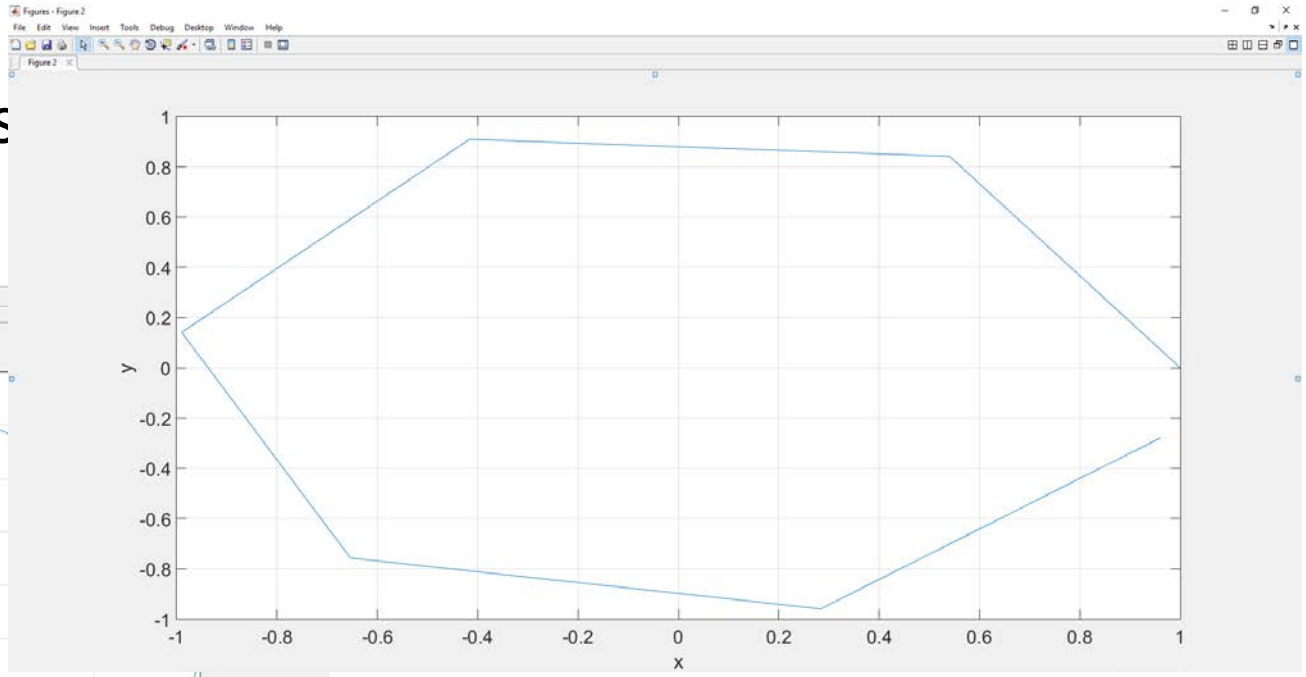
- Per visualizzare una qualsiasi funzione y = f(x) in *Matlab*, è ***SEMPRE*** necessario creare i vettori x e y nel dominio di interesse

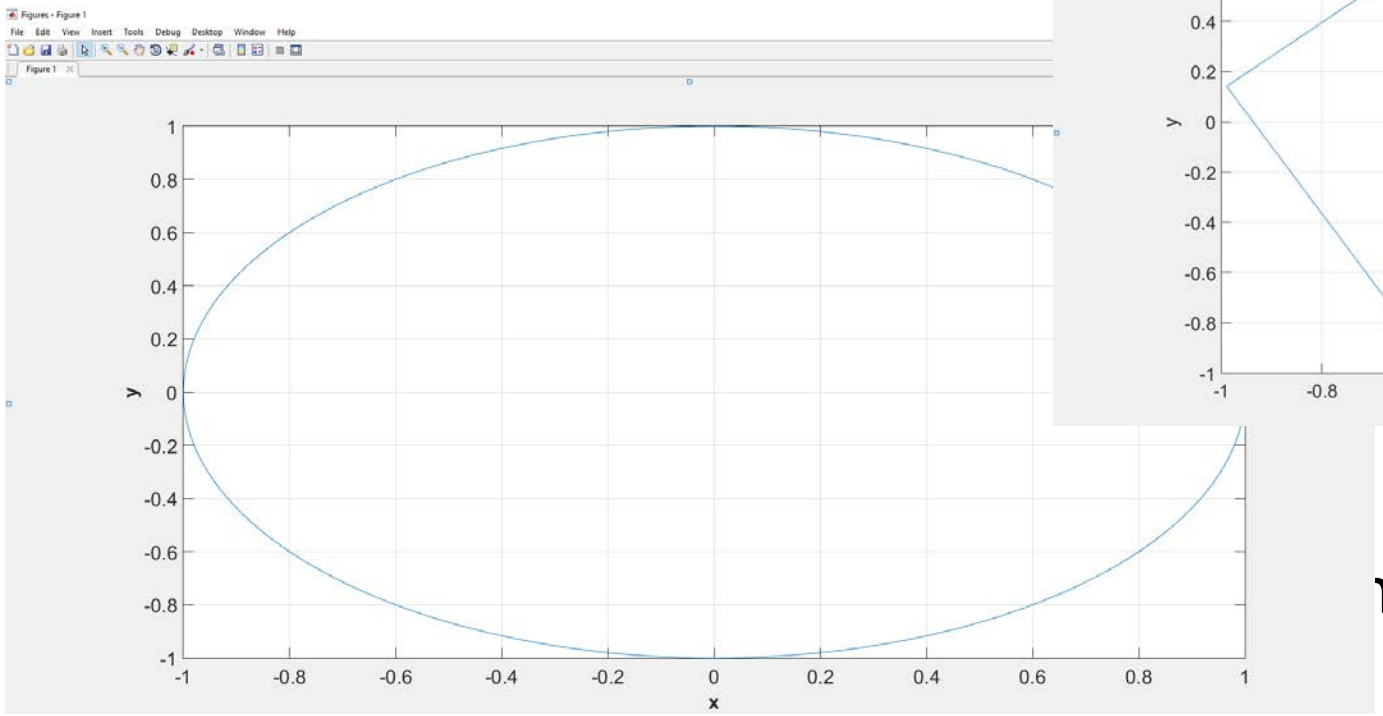
```
% un oscillatore armonico
t=[0:pi/100:2*pi]; % t=[0:2*pi];
x=cos(t);
y=sin(t);
plot(x,y);
```

- E' importante selezionare la "risoluzione" lungo l'asse x sufficientemente elevata

La Grafica in *Matlab*

- Per visualizzare una qualsias
  necessario creare i vettori x





ne" lungo l'asse x sufficientemente

La Grafica in *Matlab*

- figure(n) specifica su quale figura lavorare

```
t = [0:pi/100:2*pi];
x1 = cos(t);
y1 = sin(t);
t = [0:2*pi];
x2 = cos(t);
y2 = sin(t);

figure(1)
plot(x1,y1)
figure(2)
plot(x2,y2)
```

```
t = [0:pi/100:2*pi];
x1 = cos(t);
y1 = sin(t);
t = [0:2*pi];
x2 = cos(t);
y2 = sin(t);

figure(1)
plot(x1,y1)
plot(x2,y2)
```

```
t = [0:pi/100:2*pi];
x1 = cos(t);
y1 = sin(t);
t = [0:2*pi];
x2 = cos(t);
y2 = sin(t);

figure(1)
hold on
plot(x1,y1)
plot(x2,y2)
```

La Grafica in *Matlab*

-

t =

x1 =

y1 =

t =

x2 =

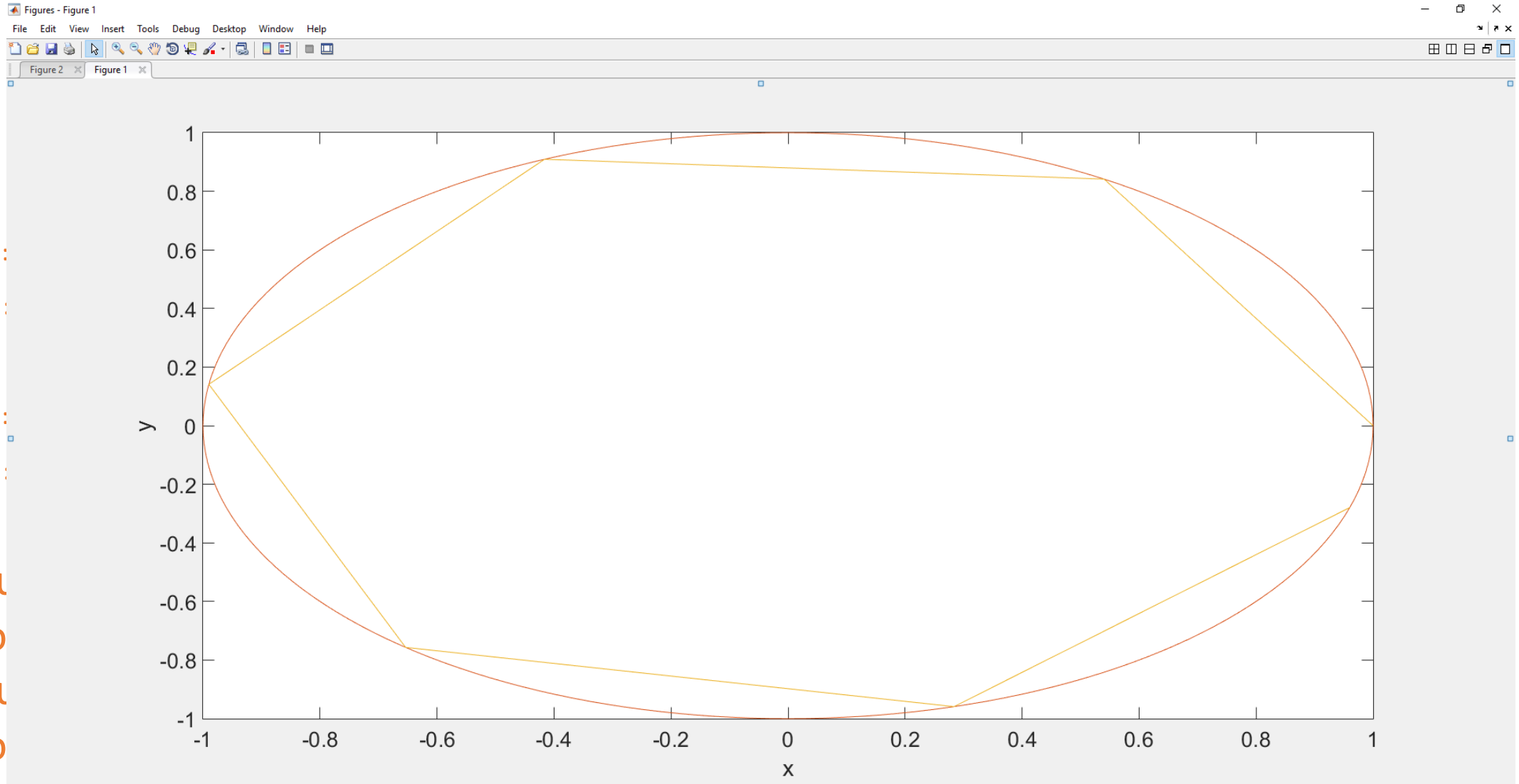
y2 =

figu

plo

figu

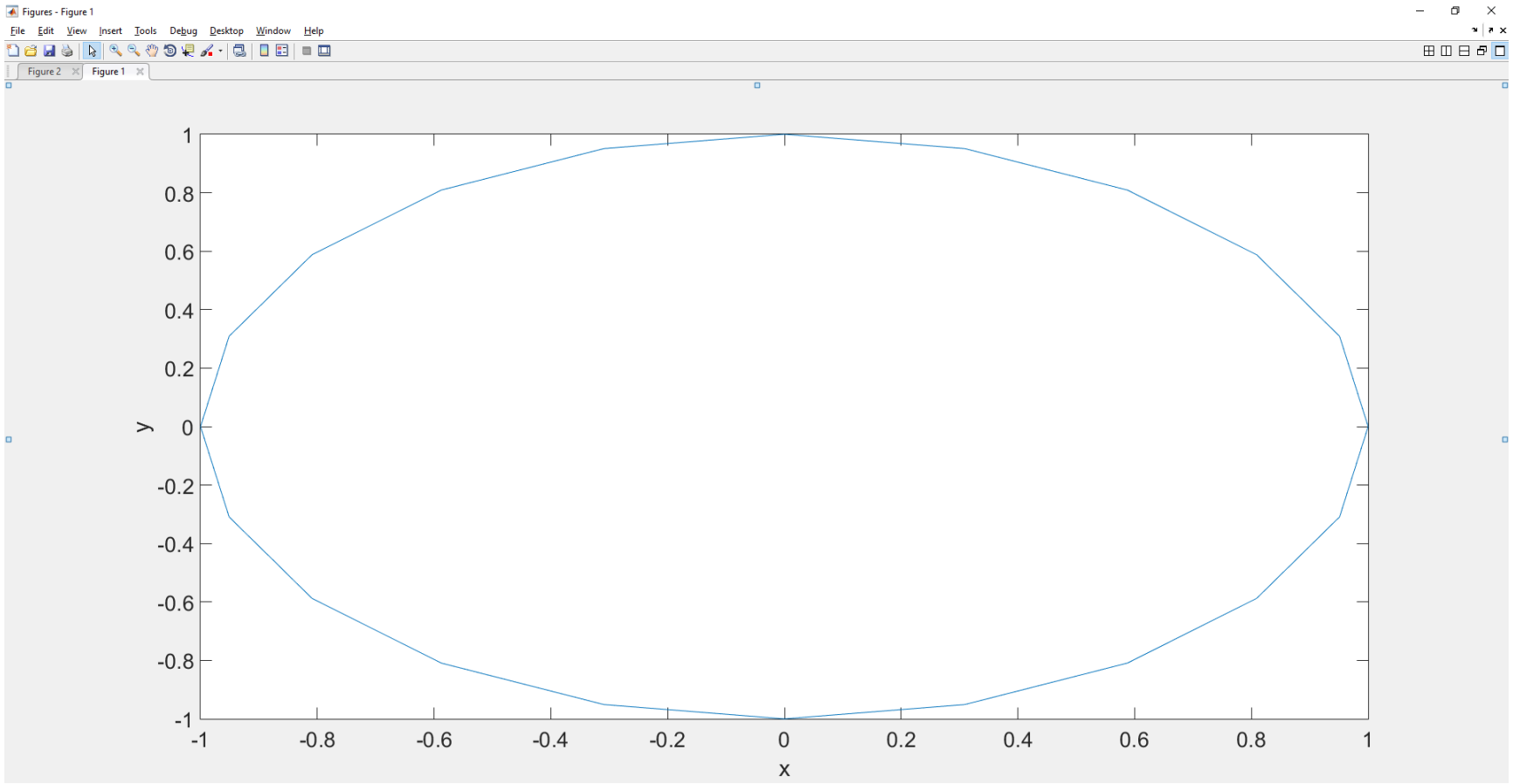
plo



La Grafica in *Matlab*

```
t = [0:pi/100:2*pi];
x1 = cos(t);
y1 = sin(t);
t = [0:2*pi];
x2 = cos(t);
y2 = sin(t);
t = [0:pi/10:2*pi];
x3 = cos(t);
y3 = sin(t);

figure(1)
hold on
plot(x1,y1)
plot(x2,y2)
hold off
plot(x3,y3)
```

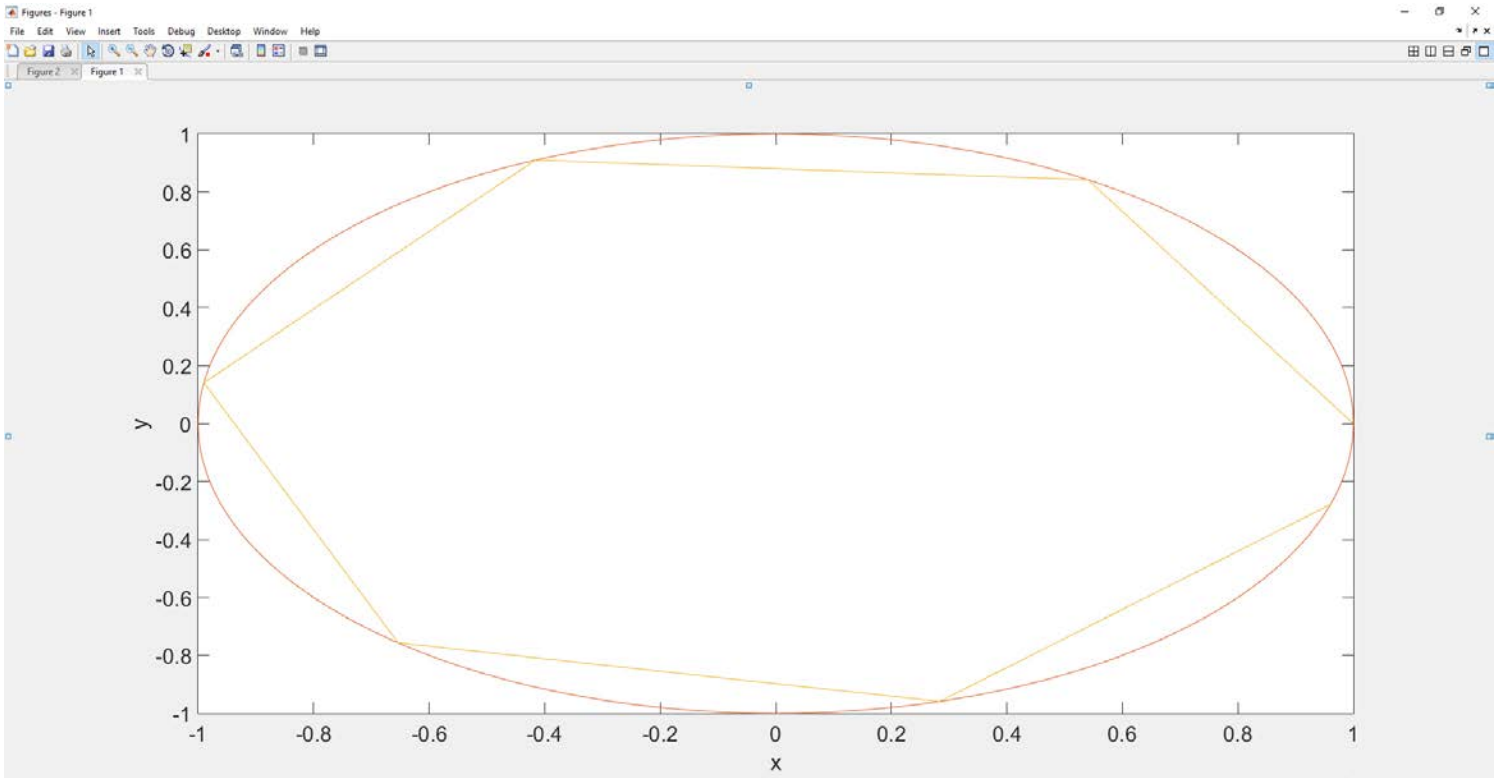


La Grafica in *Matlab*

Una variante all'utilizzo del comando hold on è l'utilizzo del comando plot secondo la struttura ***plot(x1,y1,x2,y2,…)***

```
t = [0:pi/100:2*pi];
x1 = cos(t);
y1 = sin(t);
t = [0:2*pi];
x2 = cos(t);
y2 = sin(t);

figure(1)
plot(x1,y1,x2,y2)
```



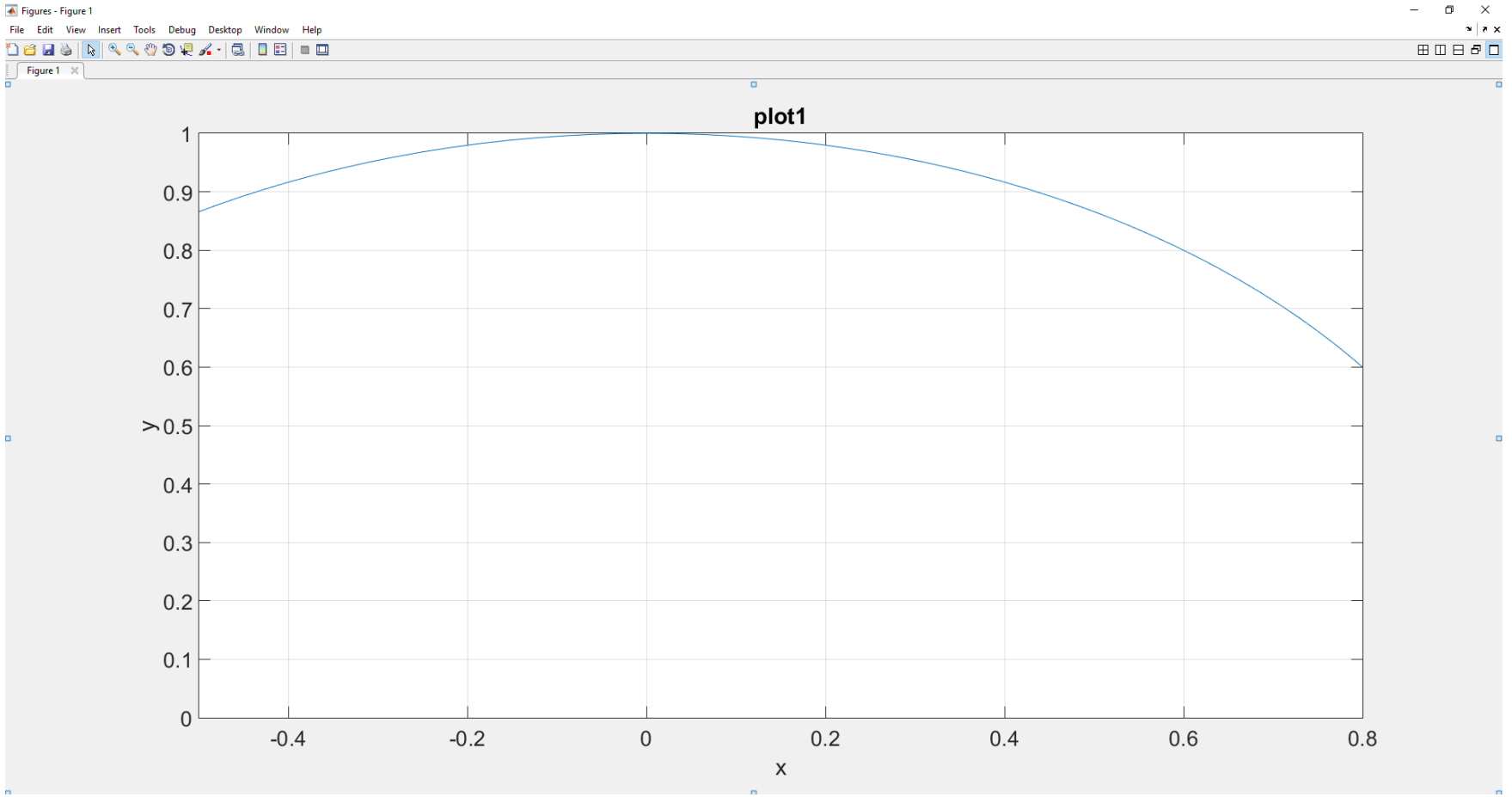
La Grafica in *Matlab*

- axis ([XMIN XMAX YMIN YMAX]) imposta la scala degli assi
- grid on abilita la griglia
- title inserisce il titolo
- xlabel, ylabel inserisce le etichette negli assi, ad esempio per specificare le unità di misura e/o il nome della variabile utilizzata
- legend inserisce la legenda

La Grafica in *Matlab*

```
t = [0:pi/100:2*pi];
x1 = cos(t);
y1 = sin(t);

figure(1)
plot(x1,y1)
axis([-0.5 0.8 0 1])
grid on
title plot1
xlabel x
ylabel y
legend
```



La Grafica in *Matlab*

Tramite il comando plot è possibile modificare vari aspetti del nostro grafico, quali:

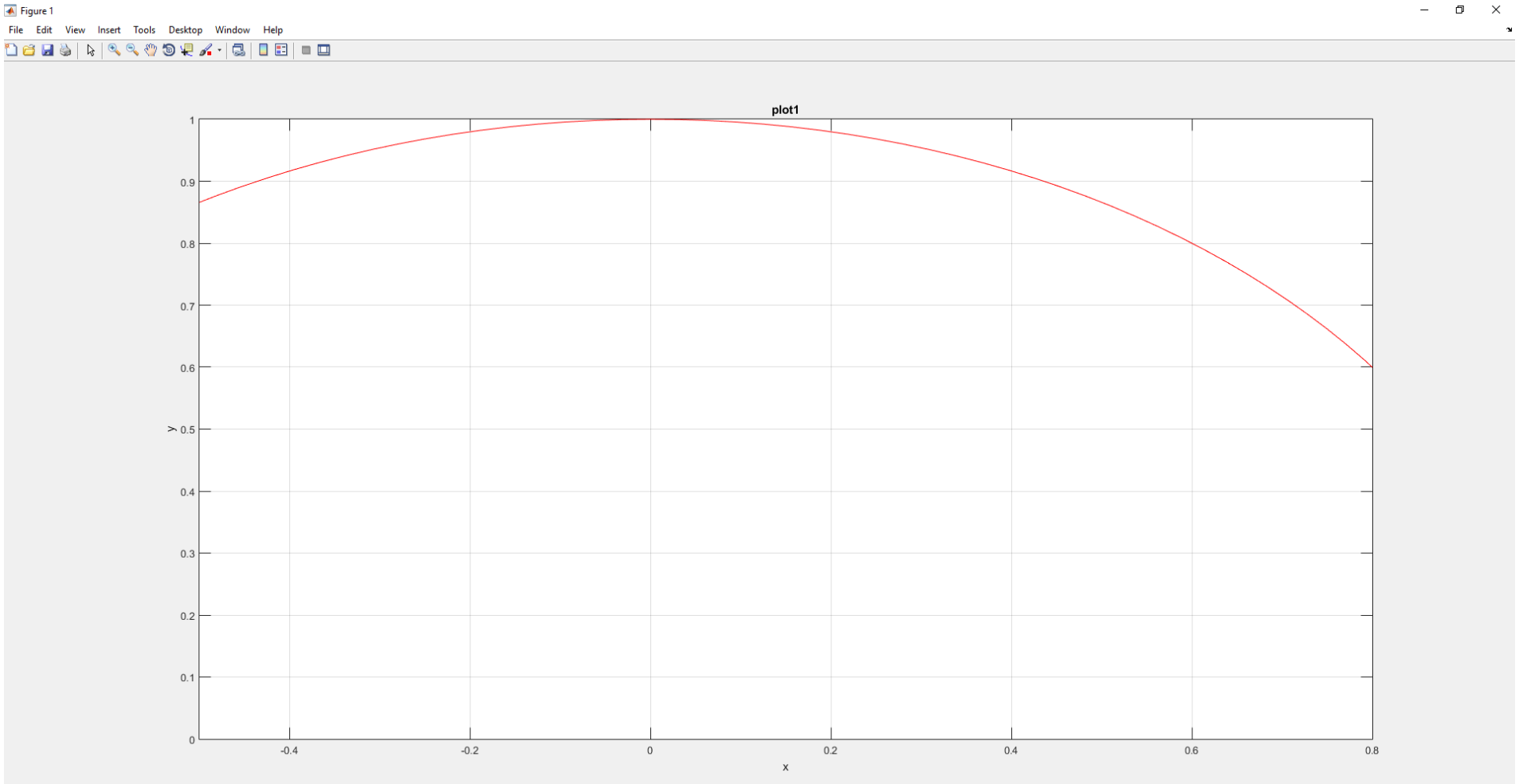
- tipo di linea
- spessore della linea
- colore della linea
- tipo di marker
- dimensione dei marker
- colore dei marker

La Grafica in *Matlab*

- plot(x1,y1,'r')
- plot(x1,y1,'g','LineWidth',5)
- plot(x1,y1,'r--')
- plot(x1,y1,'b*')
- plot(x1,y1,'b*','MarkerSize',10)

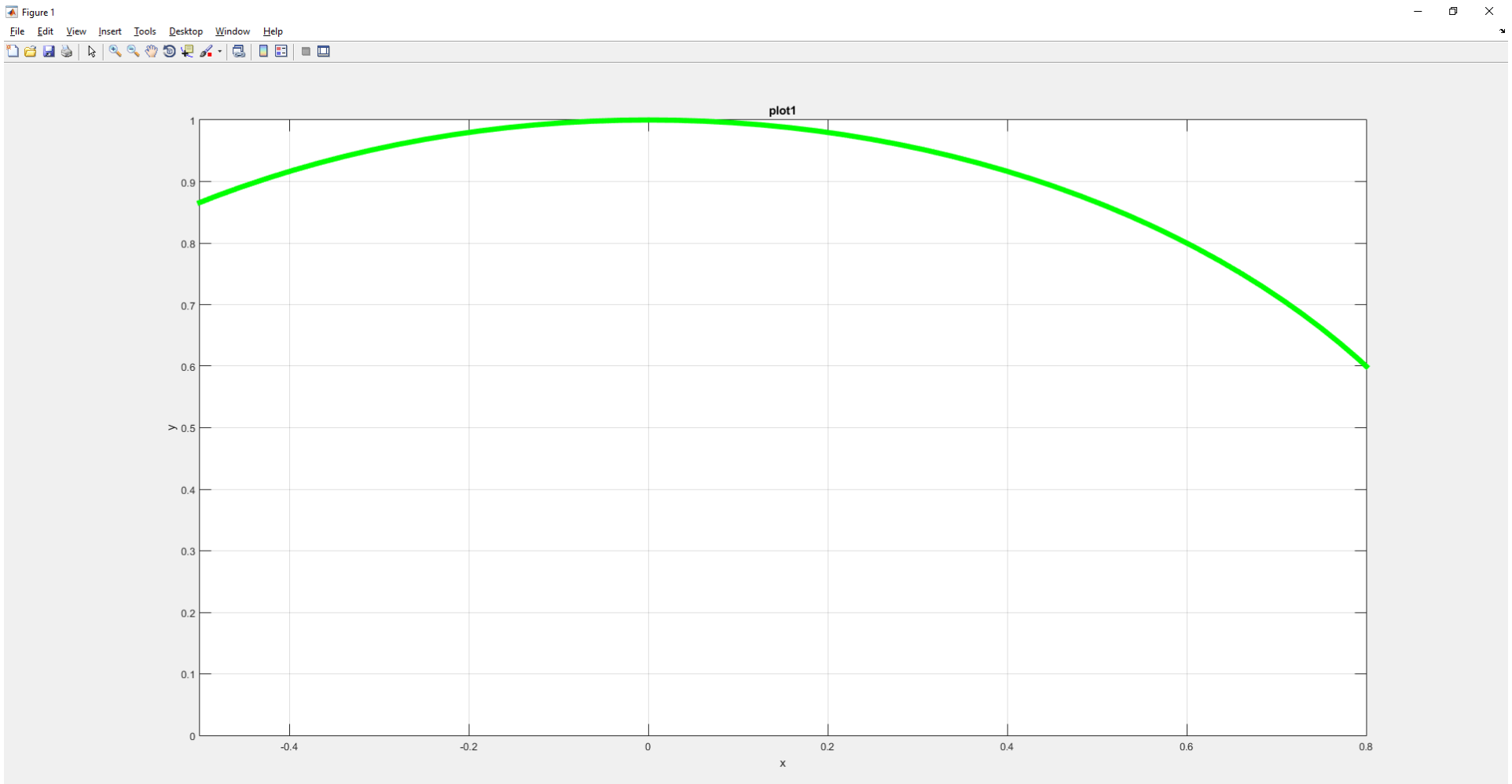
La Grafica in *Matlab*

plot(x1,y1,'r')



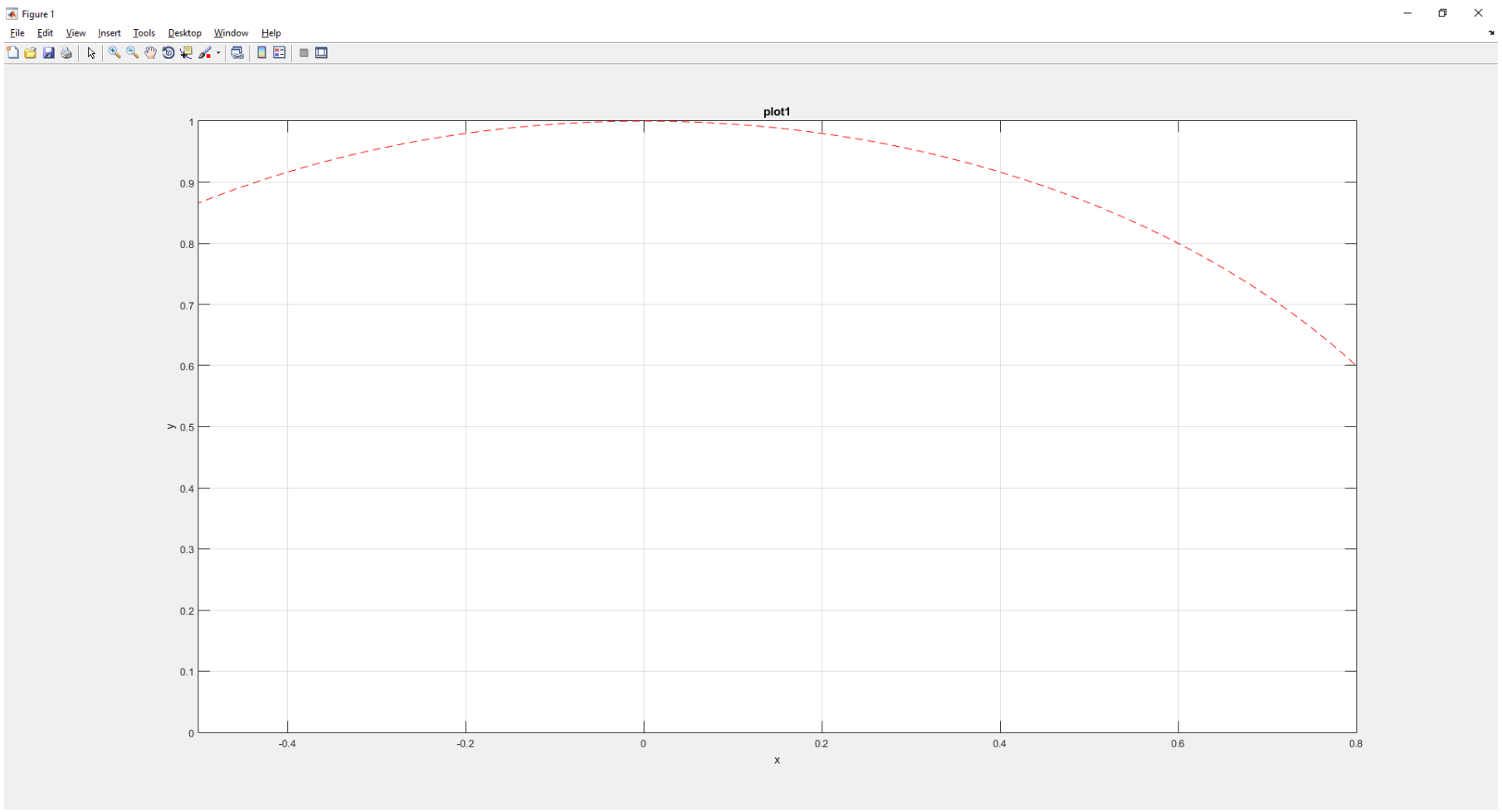
La Grafica in *Matlab*

plot(x1,y1,'g','LineWidth',5)



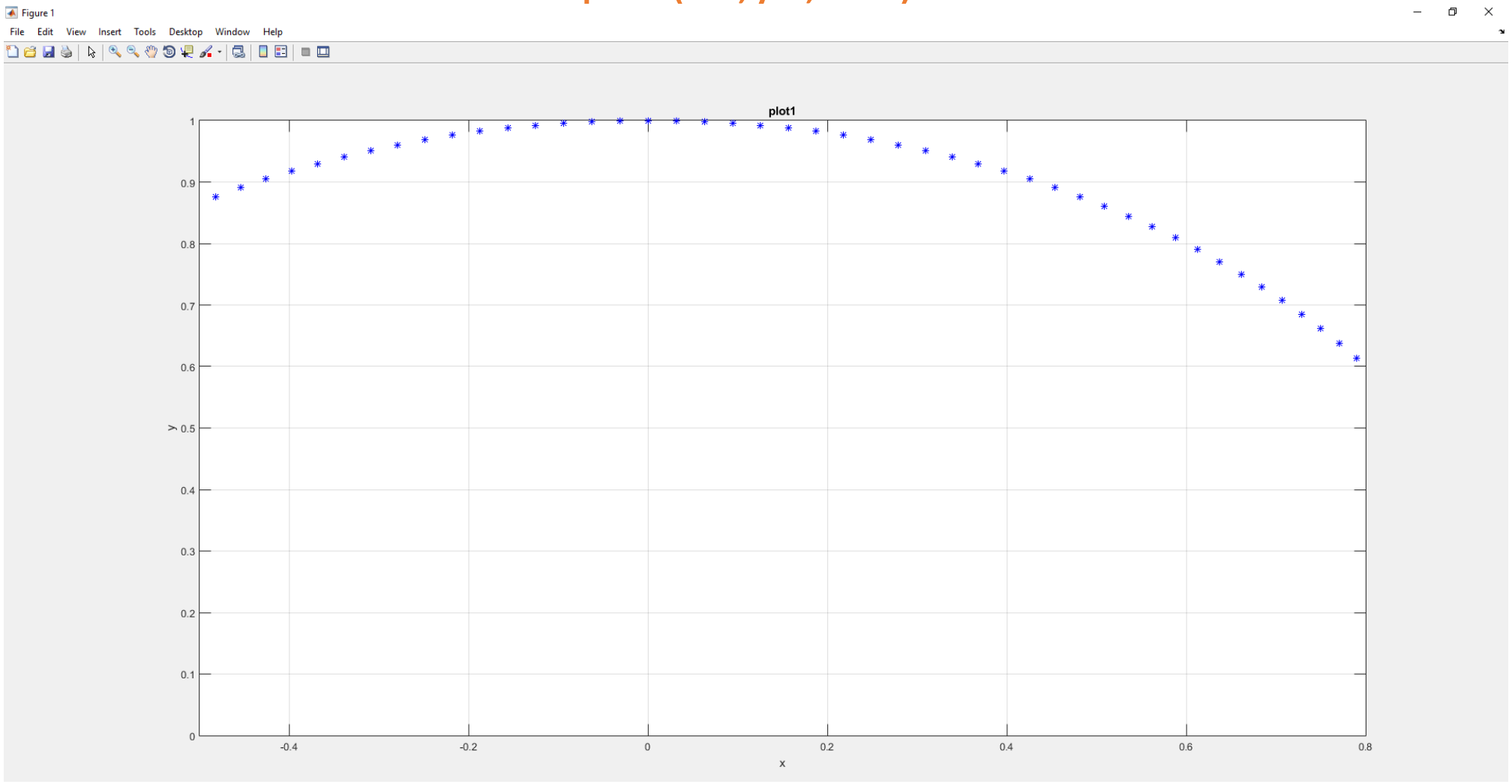
La Grafica in *Matlab*

plot(x1,y1,'r--')



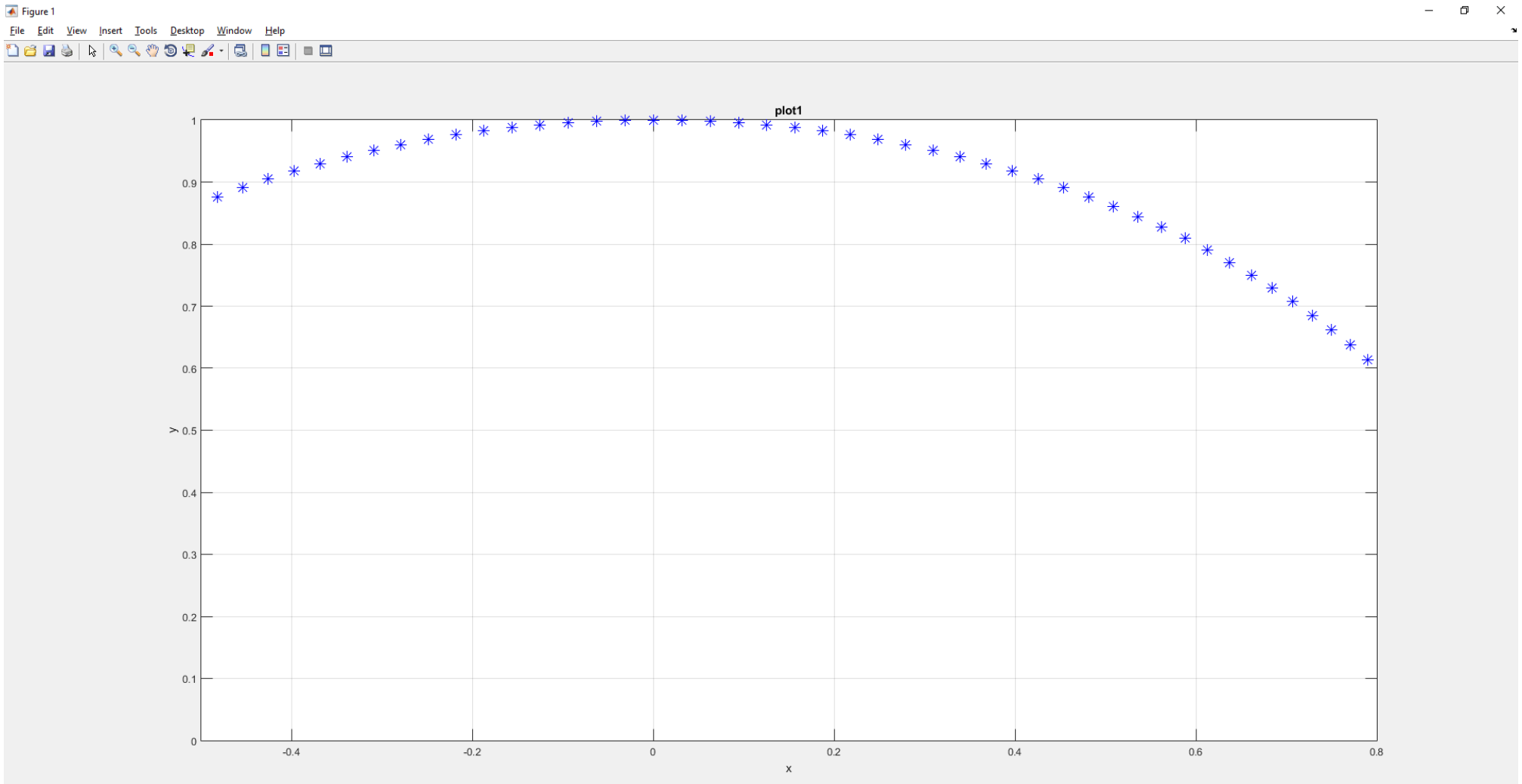
La Grafica in *Matlab*

plot(x1,y1,'b*')



La Grafica in *Matlab*

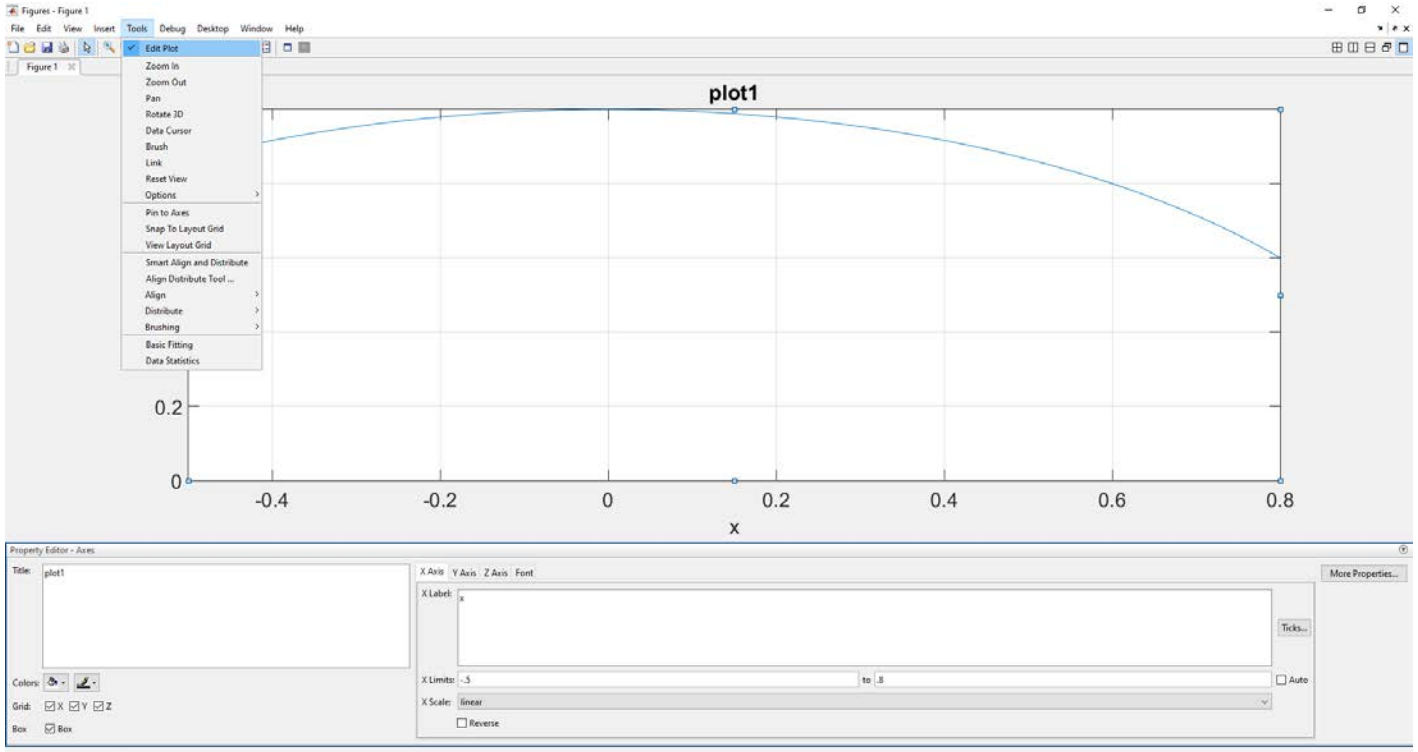
plot(x1,y1,'b*','MarkerSize',10)



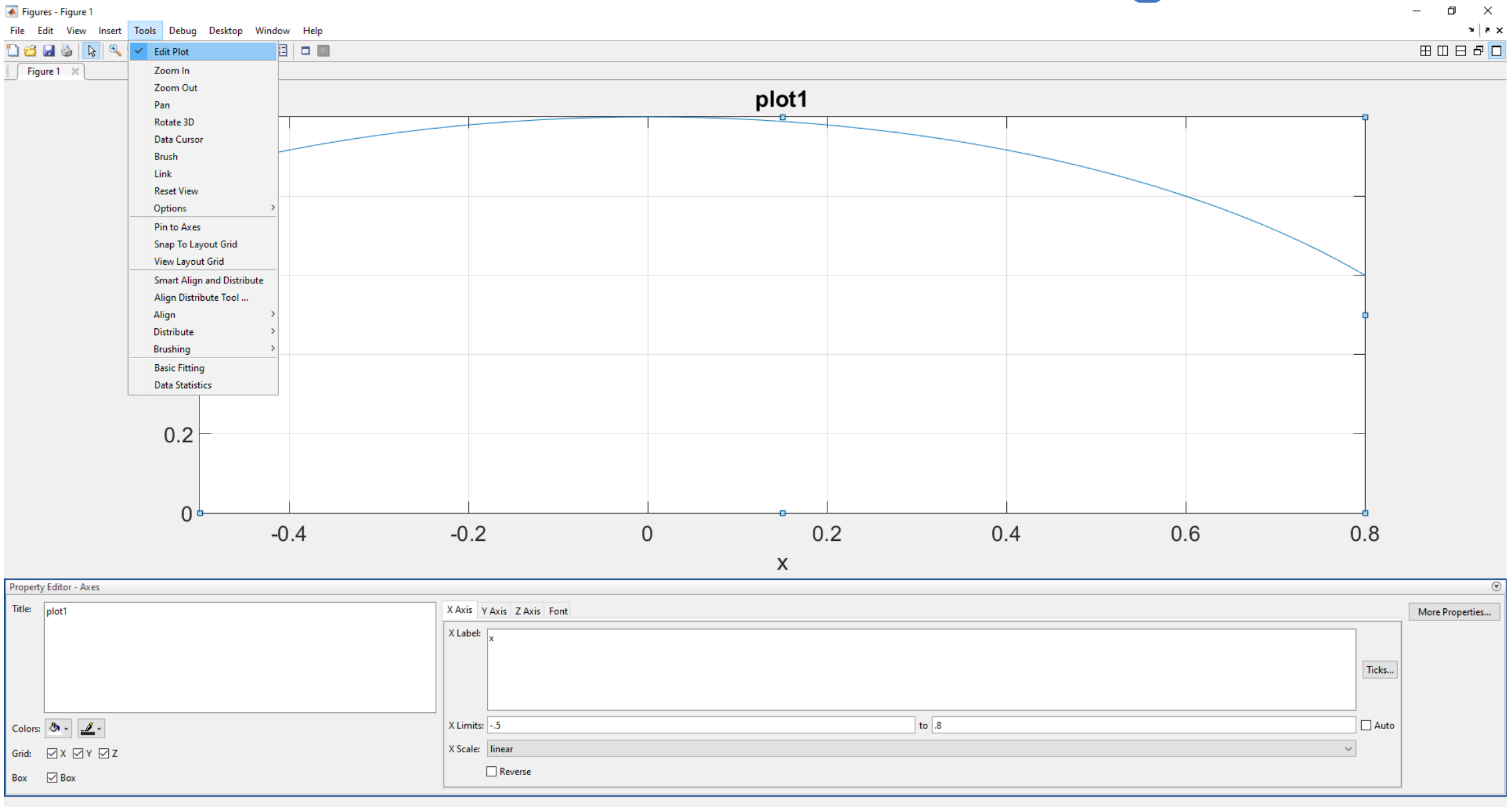
La Grafica in *Matlab*: Plot Editing Mode

Interfaccia user-friendly interattiva per modificare le proprietà delle figure, delle linee, etc...

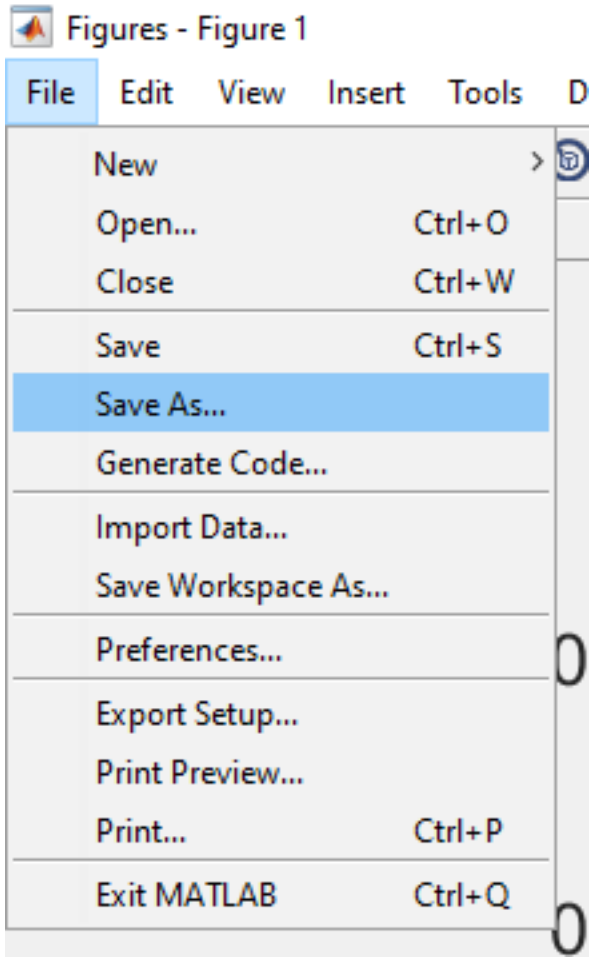
Menu “Tools → Edit Plot → Doppio click sulla figura”

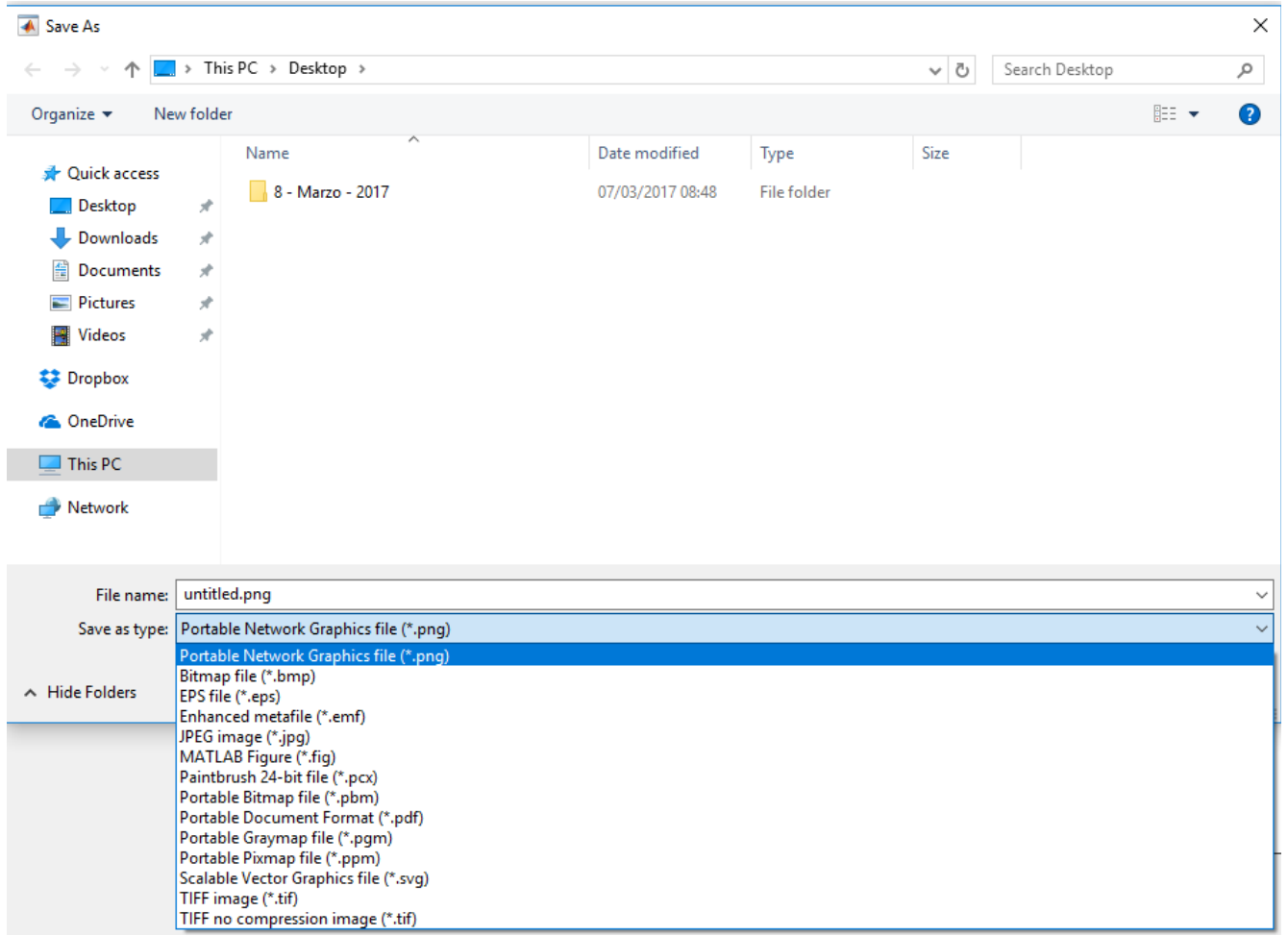


La Grafica in *Matlab*: Plot Editing Mode



La Grafica in *Matlab*: *Esportare i Grafici*





La Grafica in *Matlab*

Esistono diversi comandi per rappresentare i dati:

- plot grafico 2-D con *scala lineare lungo entrambi gli assi*
- loglog grafico con *scale logaritmiche per entrambi gli assi*
- semilogx grafico con *scala logaritmica per l'asse* x e *lineare per l'asse* y
- semilogy grafico con *scala logaritmica per l'asse* y e *lineare per l'asse* x

Riferimenti

- *Introduzione a Matlab e Simulink* di Matteo Sartini, http://www-lar.deis.unibo.it/people/msartini
- *Mini Manuale Matlab* di Antonio Salvato
- ONLINE HELP di Matlab R2016b, https://it.mathworks.com/help/matlab/

COME OTTENERE *MATLAB R2016b*

Seguire la guida http://doc.sid.unipi.it/images/1/15/Istruzioni_Installazione_MatLab_Student_2017.pdf, la quale riporta passo dopo passo tutte le operazioni da svolgere al fine di ottenere la versione *Matlab R2016b* (ed annessi toolbox, quali ad esempio *Simulink*) con licenza *Campus* valida fino al 1 Dicembre 2017.

Alla scadenza della licenza la facoltà provvede ad aggiornare le istruzioni riportate al link sopra riportato per consentire l'aggiornamento della licenza acquisita in precedenza.