

UNIVERSITÀ  
DEGLI STUDI  
DI PADOVA

Laboratorio 1: Introduzione al Matlab

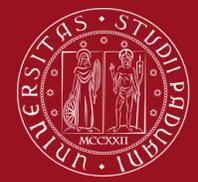


Matlab = Matrix Laboratory

Utilizza le matrici come oggetti fondamentali

Matlab consente:

- L'accesso ad un ambiente di calcolo
- L'utilizzo di funzioni specializzate (librerie)
- Un ambiente di sviluppo integrato
  - Debug
  - Notebook
  - OOP



- Facilità d'uso:
  - Linguaggio interpretato – niente compilazione
  - Ha moltissime funzioni disponibili
  - E' possibile programmare funzioni ad hoc
  - Non ci si deve preoccupare di programmazione a basso livello
    - Niente dichiarazioni di tipi, allocazione di memoria, funzioni virtuali, polimorfismo etc...
    - Tuttavia, se si vuole, si possono usare classi e compilatori (ma noi non lo faremo in questo corso)
- Esportabilità
  - I programmi Matlab sono file di testo
  - Utilizzabili senza modifiche in tutti i sistemi operativi

# Come si svolgono le esercitazioni

Aprirete Matlab in una finestra, e tenete questa presentazione aperta in un'altra. Eseguite le operazioni indicate in questa presentazione.

Le operazioni da eseguire sono quelle negli screenshot di Matlab, oppure quelle evidenziate così:



Non esitate a fare delle prove cambiando i valori dei parametri usati.

The screenshot displays a presentation slide titled "Come si svolgono le esercitazioni" (How exercises are performed) from the University of Padua. The slide contains the following text:

Aprirete Matlab in una finestra, e tenete questa presentazione aperta in un'altra. Eseguite le operazioni indicate in questa presentazione. Le operazioni da eseguire sono quelle negli screenshot di Matlab, oppure quelle evidenziate in blu. Non esitate a fare delle prove cambiando i valori dei parametri usati.

The Matlab Live Editor window shows a script for plotting a discrete sine wave. The script includes the following code:

```
%%  
n = 30:30;  
A = 0.73 ;  
phi = pi* ;  
nu = 0.207 ;  
  
plot(n,A*cos(2*pi*nu*n+phi), 'k-o');  
title(sprintf('cos( 2 \pi %4.3f n  
axis([min(n) max(n) -1 1]); grid
```

The Command Window shows the execution of the script, resulting in a plot of the sine wave. The plot is titled "cos( 2 π 0.207 n +0.00 π)" and shows a discrete sine wave with amplitude 0.73 and frequency 0.207. The plot is displayed in a window titled "mediapeseta.m (Funct...".



# Come eseguire il codice Matlab

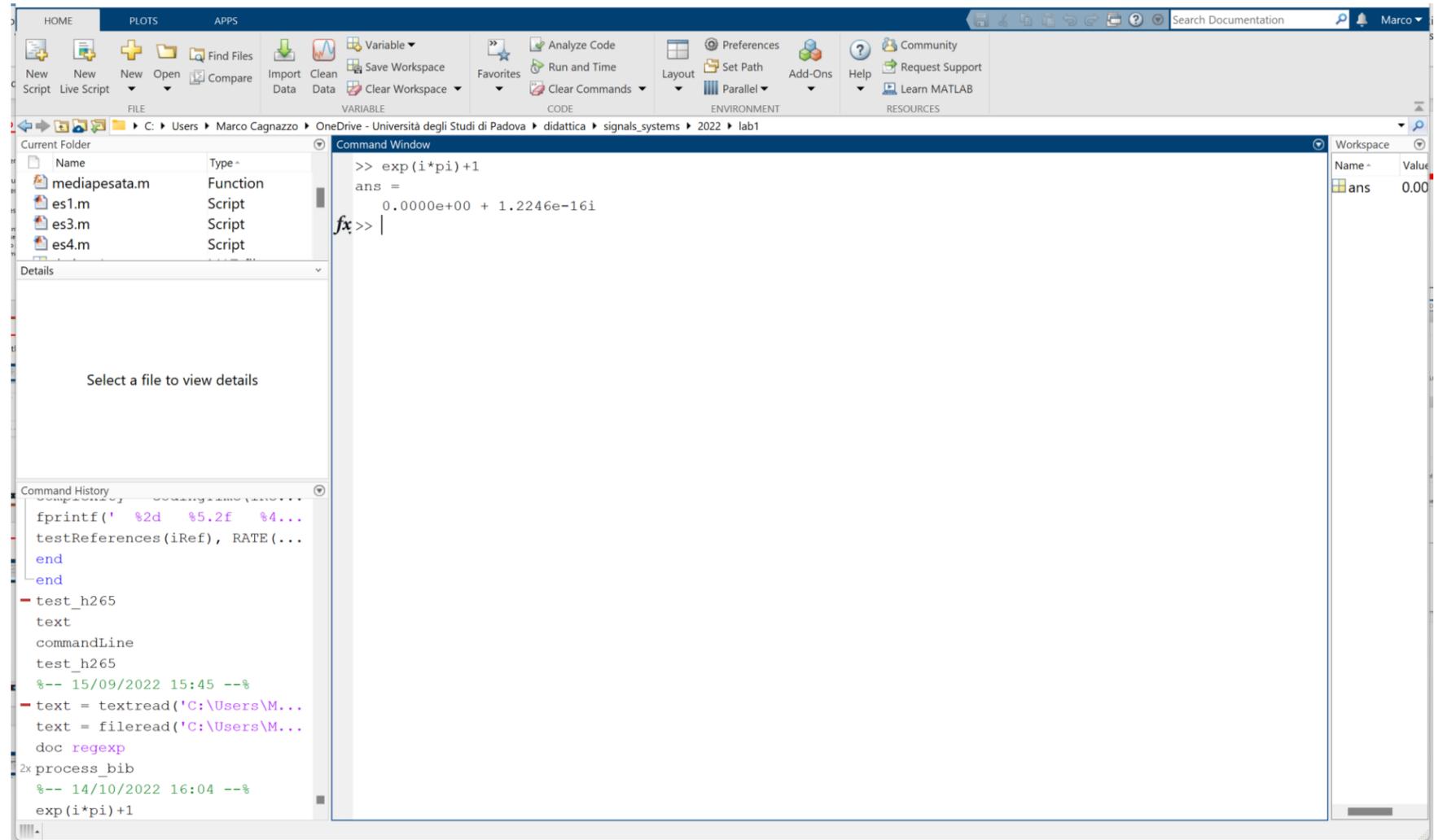
In Matlab ci sono vari modi per scrivere le istruzioni e farle eseguire:

- **Modalità interattiva:** ogni istruzione scritta viene valutata dall'interprete ed immediatamente elaborata nella Command Window (console)
- **Script:** in un file di testo con estensione “.m” si scrive una sequenza di istruzioni.
  - Lo script può essere eseguito interamente (F5)
  - Oppure suddiviso in "sezioni" eseguite individualmente (CTRL+ENTER)
  - Si può anche selezionare una o più istruzioni da uno script, una funzione o dallo storico, ed eseguirle premendo F9
- **Live Script:** in un file con estensione “.mlx”
  - Rispetto allo script, le istruzioni possono essere intervallate da un testo (per esempio, descrizione del codice e dei parametri, formule matematiche, ecc.)
  - Il live script può essere eseguito come uno script, oppure *interattivamente* utilizzando un'interfaccia grafica
  - **I live script sono l'analogo Matlab dei *Notebook* utilizzati in altri linguaggi/ambienti come in Python**



# Il Desktop di Matlab

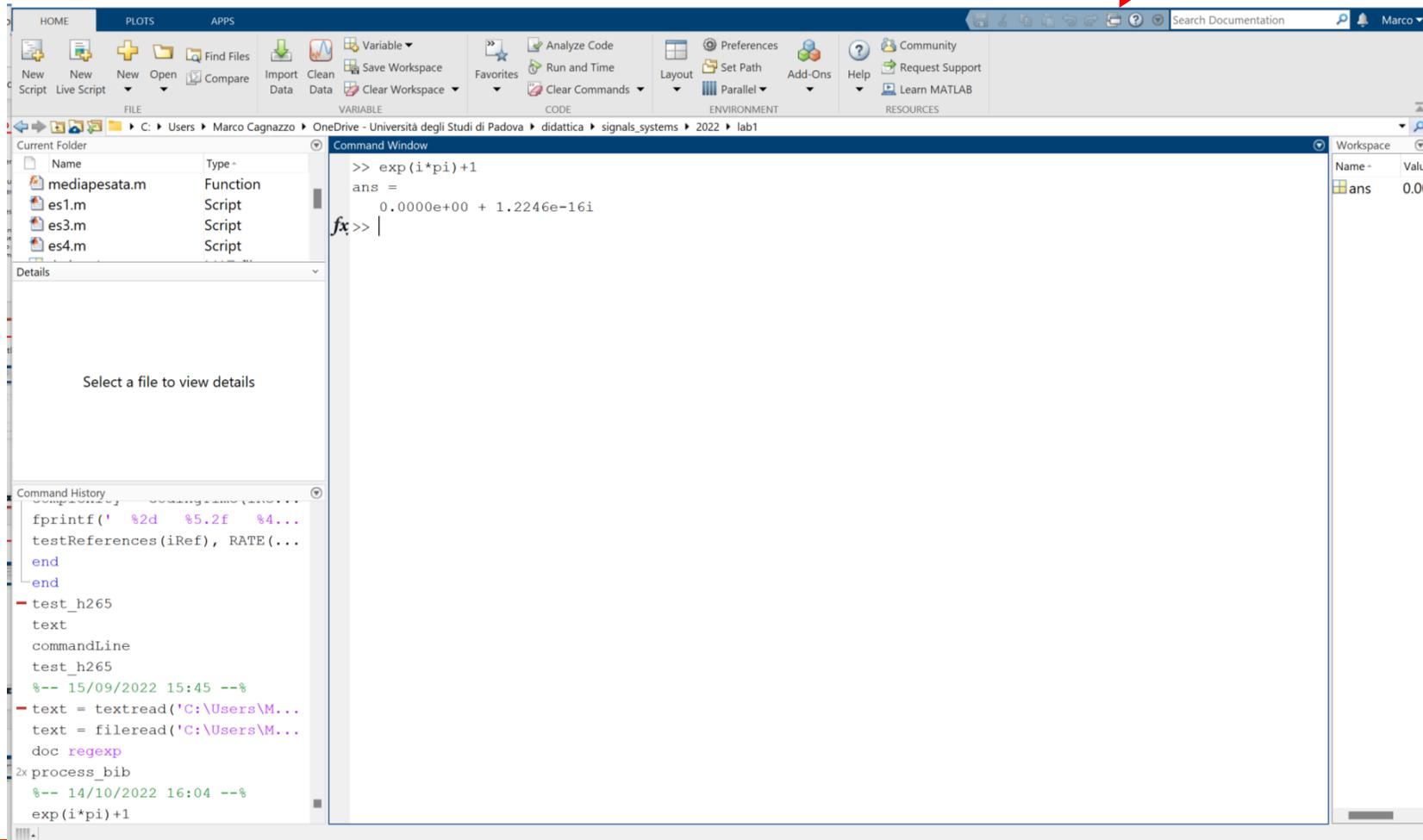
La finestra principale  
(Desktop) di Matlab  
è divisa in vari  
pannelli





# Il Desktop di Matlab

Non tutti i pannelli sono sempre visibili: per attivarli, cliccare su Switch Window





# Command Window

The screenshot shows the MATLAB Command Window interface. The Command Window displays the command `>> exp(i*pi)+1` and the output `ans = 0.0000e+00 + 1.2246e-16i`. A red arrow points from a text box to the Command Window. The Command History window shows a list of previous commands, including `exp(i*pi)+1`.

La *Command Window*: è il prompt interattivo di Matlab, dove possono essere eseguite operazioni e comandi, come il calcolo di  $e^{i\pi} + 1$ . È l'equivalente di una finestra di terminale o console. Si può usare anche per muoversi nel File System e per lanciare eseguibili esterni.



# Current folder

The screenshot shows the MATLAB software interface. The top menu bar includes HOME, PLOTS, and APPS. Below the menu bar is a toolbar with various icons for file operations, workspace management, and analysis. The main workspace is divided into several panes:

- Current Folder:** A file browser showing the contents of the current folder. The path is `C:\Users\Marco Cagnazzo\OneDrive - Università degli Studi di Padova\didattica\signals_systems\2022\lab1`. The files listed are `mediap_data.m` (Function), `es1.m` (Script), `es3.m` (Script), and `es4.m` (Script).
- Command Window:** A window for entering MATLAB commands. The current command is `>> exp(i*pi)+1`, and the output is `ans = 0.0000e+00 + 1.2246e-16i`.
- Command History:** A window showing a list of previously executed commands, including `fprintf`, `testReferences`, `test_h265`, `textread`, `fileread`, `doc regex`, `process_bib`, and `exp(i*pi)+1`.
- Workspace:** A window showing the current workspace variables. The variable `ans` is listed with a value of `0.0000e+00 + 1.2246e-16i`.

Red arrows point from a text box to the Current Folder browser, the Command Window, and the Command History window.

Current Folder mostra il contenuto della cartella corrente, il cui path è in alto. Selezionando un file, i dettagli sono visibili qui

# Navigazione tra cartelle

Usate il pannello Current Folder per navigare nel File System, per creare ed aprire file o cartelle

In alternativa, le stesse operazioni possono essere effettuate nella Command Window:

- Create una cartella di lavoro per il corso di segnali e sistemi, spostatevi in essa e create una sottocartella per la prima esercitazione.

**Scaricare dal moodle tutto il materiale necessario per la prima esercitazione, estrarlo (unzip) se necessario e copiarlo nella sottocartella creata**

**Dovreste vedere questo:**



```
Command Window
>> mkdir segnali_sistemi
>> cd segnali_sistemi\
>> mkdir lab1
>> cd lab1
fx >> |
```



Current Folder	
Name	Type ^
esempio_strutture_pr..	Script
freq_num.m	Script
script_esempio.m	Script
sinusoidi_discrete.mlx	Live Script
Laboratorio_1.pdf	Documento Adob...

# Command History

The screenshot displays the MATLAB environment. The Command Window shows the following interaction:

```
>> exp(i*pi)+1  
ans =  
0.0000e+00 + 1.2246e-16i
```

The Command History window shows a list of commands, with the first command highlighted:

```
fprintf('...')  
testReference(iRef), RATE(...  
end  
end  
- test_h265  
text  
commandLine  
test_h265  
%-- 15/09/2022 15:45 --%  
- text = textread('C:\Users\M...  
text = fileread('C:\Users\M...  
doc regexp  
2x process_bib  
%-- 14/10/2022 16:04 --%  
exp(i*pi)+1
```

A red arrow points from the text box to the first command in the Command History window.

La Command History mostra gli ultimi comandi del prompt. Ci sono molti modi per sfruttare la Command History: Nella Command Window, freccia in alto permette di percorrere la History; se si cominciano a battere dei caratteri, verranno richiamati i comandi della History che cominciano con quei caratteri. I comandi possono essere selezionati dalla Command History ed eseguiti premendo F9. Oppure possono essere copiati e poi incollati nel prompt oppure in un file di comandi (detto script).



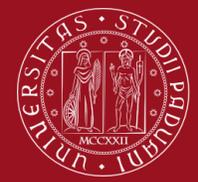
# Il Workspace

The screenshot shows the MATLAB interface with the following components:

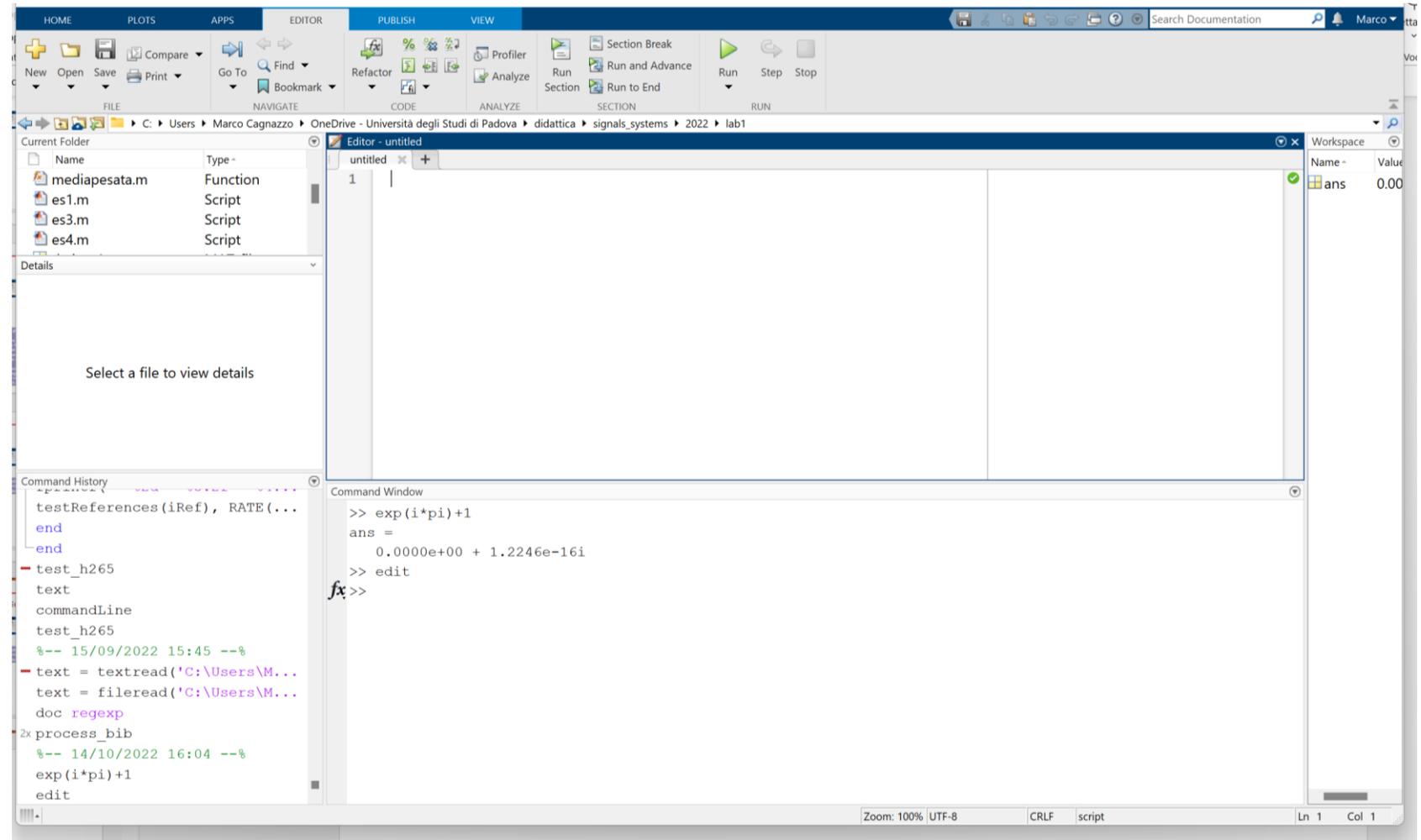
- Command Window:** Contains the command `>> exp(i*pi)+1` and its output: `ans = 0.0000e+00 + 1.2246e-16i`.
- Workspace:** A table with two columns: 'Name' and 'Value'. It contains one entry: 'ans' with the value `0.0000e+00 + 1.2246e-16i`.
- Command History:** Shows a list of previous commands, including `fprintf`, `testReferences`, `test_h265`, `textread`, `fileread`, `doc regexp`, and `process_bib`.

A red arrow points from the text box to the 'ans' variable in the workspace panel.

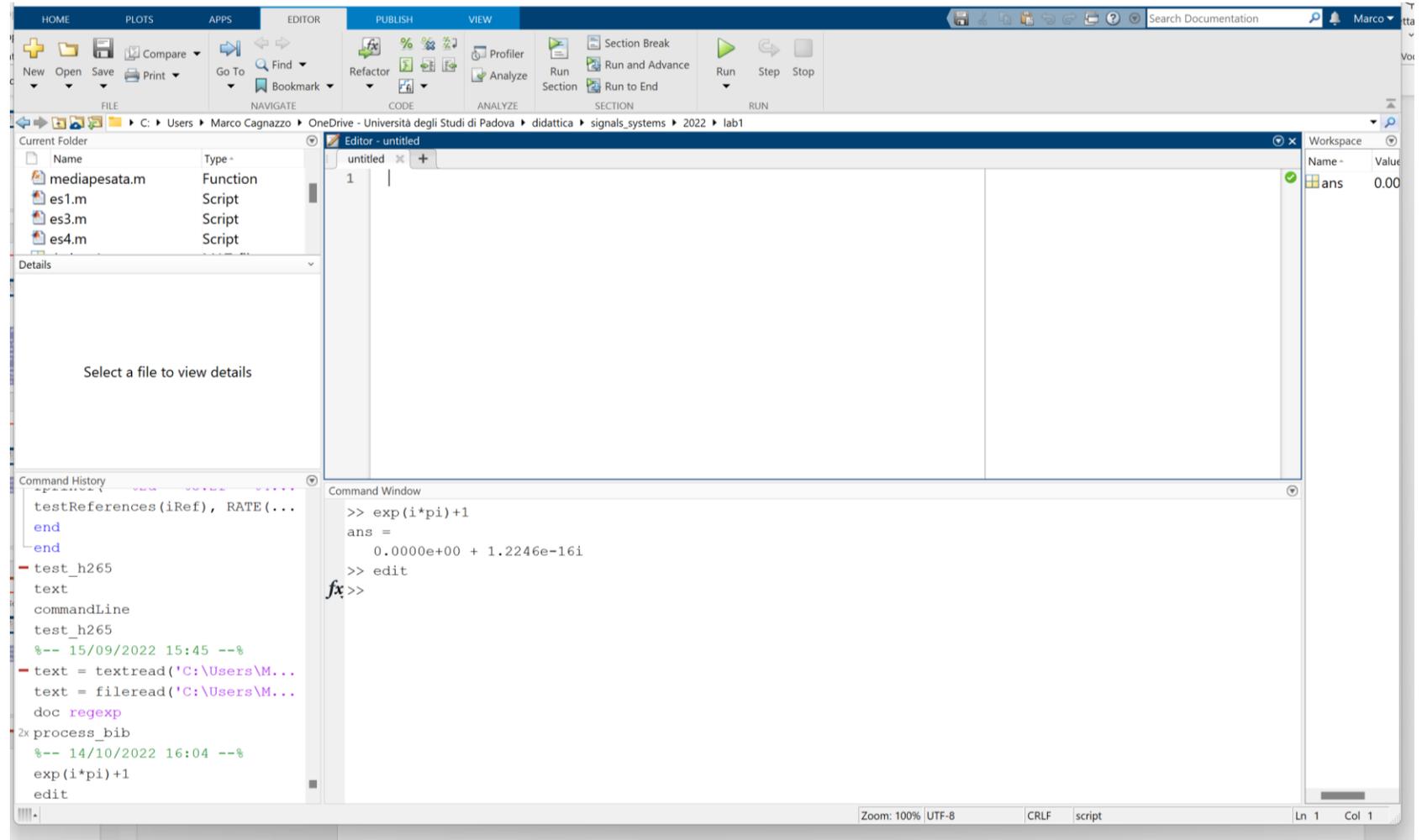
È possibile visualizzare tutte le variabili memorizzate nello spazio di memoria corrente usando il pannello Workspace



Si consiglia di attivare il **pannello Editor** se non è già attivo: scrivere “edit” nella command window



Dal pannello editor si possono modificare e lanciare script e live script





# Apertura ed esecuzione di file

Doppio clic su un file per aprirlo  
Per eseguirlo, click su Run,  
oppure F5, oppure scrivere il  
nome (senza estensione)  
nella CW

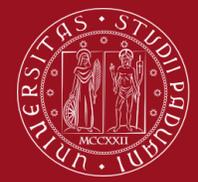
The screenshot shows the MATLAB R2022a interface. The Editor window displays a script file named 'freq\_num.m' with the following code:

```
1 %% Grafici sinusoidi discrete: studio del valore della frequenza numerica
2
3 %% Caso 1
4 A=1; phi = 0; nu = 0.03;
5 n = 0:40;
6 x = A*cos(2*pi*nu*n+phi);
7 % Creazione grafico
8 figure(1);
9 stem(n,x);
10 xlabel('n'); ylabel('x(n)');
11 title(sprintf('Freq: %4.3f',nu));
12
13 %% Caso 2
14 A=1; phi = 0; nu = 0.97;
15 n = 0:40;
16 x = A*cos(2*pi*nu*n+phi);
17 % Creazione grafico
18 figure(2);
19 stem(n,x);
20 xlabel('n'); ylabel('x(n)');
21 title(sprintf('Freq: %4.3f',nu));
22
```

The Command Window (CW) shows the prompt `fx>>`. The Command History window shows the following commands:

```
rmdir tmp
cd student
mkdir student
cd student
ls
clc
```

An orange arrow points from the text 'click su Run' to the Run button in the MATLAB interface.



# Primi passi in Matlab

**Eseguite queste istruzioni nel prompt (Command Window),  
dovreste ottenere una schermata come questa →**

>> **a=10** – crea la variabile a e le assegna il valore 10

>> **b=[]** – crea la variabile b e le assegna il vettore vuoto

>> **size()** – restituisce le dimensioni della variabile (in questo caso: 1x1 per lo scalare a e 0x0 per la variabile “vuota” b)

Notare che il tipo viene assegnato automaticamente al momento della creazione della variabile: per default sono double (guardare nel pannello Workspace)

```
Command Window
>> a=10
a =
    10
>> size(a)
ans =
     1     1
>> b=[]
b =
     []
>> size(b)
ans =
     0     0
fx >> |
```

**Eseguite il codice mostrato**

# Primi passi in Matlab

- + somma
- - sottrazione
- \* moltiplicazione
- / divisione
- ^ potenza
- Notare che l'unità immaginaria si indica con **1i**

Eseguite il codice mostrato

Command Window

```
>> 6*4
ans =
    24
>> 2^10
ans =
    1024
>> sqrt(-1)
ans =
    0.0000 + 1.0000i
>> 1i^2
ans =
    -1
fx >> |
```

# Primi passi in Matlab

Il simbolo % introduce un commento: tutto quello che segue sulla stessa linea è ignorato

Il simbolo ; sopprime l'output e separa le righe delle matrici

Il simbolo . è usato per distinguere le operazioni elemento per elemento da quelle su matrici

Il simbolo , è usato per separare le colonne di vettori e matrici

Il simbolo ' è usato per il trasposto coniugato

Le parentesi **quadre** sono usate per **creare** vettori e matrici, le parentesi **tonde** per **accedere** agli elementi delle matrici e per chiamare le funzioni

**Il primo elemento di un vettore  $x$  è accessibile con  $x(1)$**

Eseguite il codice mostrato

```
>> x = [1, 2, 3]
x =
     1     2     3
>> y = [3, 2, 1]'
y =
     3
     2
     1
>> z=y+[1; 1; 1]
z =
     4
     3
     2
>> w=2*z-x';
>> w(1)
ans =
     7
```

# Creare un vettore

Nell'esempio si mostrano  
diversi modi di creare vettori  
riga e colonna

La virgola separa le colonne, il  
punto e virgola separa le righe

L'apostrofo opera il trasposto  
coniugato (sui reali è  
equivalente al trasposto)

```
Command Window
>> rowVector = [1 2 3]
rowVector =
     1     2     3
>> anotherRow=[1,2,3]
anotherRow =
     1     2     3
>> columnVector=[1;2;3]
columnVector =
     1
     2
     3
>> anotherColumn = rowVector'
anotherColumn =
     1
     2
     3
fx >> |
```

Eseguite il codice mostrato

# Creare una matrice

Nell'esempio si mostrano diversi modi per creare una matrice:

- elencandone i valori
- come trasposta di un'altra matrice
- con delle funzioni che creano matrici speciali



Eseguite il codice mostrato

```
Command Window
>> matrix = [1,2,3; 4 5 6]
matrix =
     1     2     3
     4     5     6
>> matrix2 = matrix'
matrix2 =
     1     4
     2     5
     3     6
>> matrix3 = ones(3)
matrix3 =
     1     1     1
     1     1     1
     1     1     1
>> matrix4 = zeros(2)
matrix4 =
     0     0
     0     0
>> matrix5 = eye(4)
matrix5 =
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
fx >> |
```

# Operazioni su matrici

Nell'esempio si mostra il comando `whos` che mostra lo spazio di memoria corrente, ed una moltiplicazione matriciale

```
Command Window
>> a = [2,3 ; 4, 5];
>> b = [ 2, 5, 1; 4, 6, 7];
>> whos a b
  Name      Size      Bytes  Class  Attributes
  a         2x2         32  double
  b         2x3         48  double

>> a*b
ans =
    16    28    23
    28    50    39
fx >>
```

Eseguite il codice mostrato

# Operazioni tra matrici

Nell'esempio si mostra la somma tra matrici e la differenza tra prodotto matriciale \* e prodotto elemento per elemento .\*  
Si mostra anche la divisione elemento per elemento

```
Command Window
>> a = [2 3 5; 4 1 0; 1 4 -1];
>> b = [1 0 0; 0 1 0; 0 0 1];
>> c=a*b
c =
     2     3     5
     4     1     0
     1     4    -1
>> d=a.*b
d =
     2     0     0
     0     1     0
     0     0    -1
>> [3 4 5]./[6 1 .5]
ans =
     0.5000     4.0000    10.0000
fx >>
```





# Intervalli numerici

In Matlab si possono definire gli intervalli numerici con l'operatore due punti  
Se non si specifica il passo, per default è uno

```
Command Window
>> start = 3; stop = 20;
>> a=start:stop
a =
  Columns 1 through 9
     3     4     5     6     7     8     9    10    11
  Columns 10 through 18
    12    13    14    15    16    17    18    19    20
>> step = 5;
>> b = start:step:stop
b =
     3     8    13    18
fx >> |
```

Eseguite il codice mostrato



# Accesso a vettori e matrici

Cose importanti da ricordare:

**L'indicizzazione di vettori e matrici comincia da 1**

**L'accesso si esegue con le parentesi tonde**

Quindi, il primo elemento di un vettore è:

$x(1)$

Mentre in C sarebbe  $x[0]$

L'accesso a sottomatrici e sottovettori è molto flessibile.

Nell'esempio, la sottomatrice N è il blocco delle righe 1,2 e delle colonne 2,3

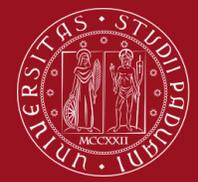
Usando il passo degli intervalli è possibile accedere, **sia in lettura sia in scrittura**, a righe o a colonne regolarmente spaziate

Nell'esempio,  $1:2:end$  sono tutti gli indici dispari della matrice M

Eseguite il codice mostrato

Command Window

```
>> M = [ 1 2 3; 4 5 6; 7 8 9]
M =
     1     2     3
     4     5     6
     7     8     9
>> x=M(1,1)
x =
     1
>> N=M(1:2,2:3)
N =
     2     3
     5     6
>> % Sottomatrice: righe e colonne dispari
>> K = M(1:2:end, 1:2:end)
K =
     1     3
     7     9
>> %Interallacciamento di vettori
>> x1 = zeros(1,3)
x1 =
     0     0     0
>> x2 = ones(1,3)
x2 =
     1     1     1
>> y(1:2:5) = x1; y(2:2:6) = x2
y =
     0     1     0     1     0     1
```



Funzioni su vettori: la maggior parte delle funzioni di Matlab è definita su vettori:

```
Command Window
>> t = -5:5;
>> x = sin(t)
x =
Columns 1 through 10
    0.9589    0.7568   -0.1411   -0.9093   -0.8415         0    0.8415    0.9093    0.1411   -0.7568
Column 11
   -0.9589
>> y = exp(t)
y =
Columns 1 through 10
    0.0067    0.0183    0.0498    0.1353    0.3679    1.0000    2.7183    7.3891   20.0855   54.5982
Column 11
  148.4132
fx>>
```



## Qualche funzione utile:

Command Window

```
>> x = rand % Valori casuali tra 0 e 1
x =
    0.9058
>> X = rand(2,3) % Matrice 2x3 di valori casuali
X =
    0.1270    0.6324    0.2785
    0.9134    0.0975    0.5469
>> mCol = max(X) % massimo colonna per colonna
mCol =
    0.9134    0.6324    0.5469
>> XV = X(:) % "vettorizzazione" di X
XV =
    0.1270
    0.9134
    0.6324
    0.0975
    0.2785
    0.5469
>> max(XV)
ans =
    0.9134
fx >> |
```

Command Window

```
>> X = rand(2,3) % Matrice 2x3 di valori casuali
X =
    0.9575    0.1576    0.9572
    0.9649    0.9706    0.4854
>> mean(X) % media per colonne
ans =
    0.9612    0.5641    0.7213
>> mean(X,2) % media per righe
ans =
    0.6908
    0.8070
>> mean(X(:)) % media globale
ans =
    0.7489
>> sum(X) % somma per colonne
ans =
    1.9224    1.1282    1.4425
>> X > 0.5 % operatore logico
ans =
    2x3 logical array
    1    0    1
    1    1    0
```



- Nella Command Window
  - » **who** – mostra tutte le variabili
  - » **whos** – mostra il nome delle variabili, il tipo e la dimensione
  - » **clear var1 var2...** - cancella le variabili riportate
  - » **clearvars** – cancella tutte le variabili -
  - » **close all** – chiude tutte le figure
  - » **save** – salva le variabili in un file .mat
  - » **load** - carica le variabili da un file .mat



## • Nella Command Window

>> **lookfor**  
**keyword**

Ricerca la keyword in tutta la documentazione (può essere lento)

>> **help**  
**command\_name**  
descrizione sintetica del comando o della libreria (toolbox)

>> **doc**  
**command\_name**  
Versione navigabile e interattiva della documentazione

Command Window

```
>> lookfor Toeplitz
toeplitz           - Toeplitz matrix.
chow              - Chow matrix (singular Toeplitz lower Hessenberg matrix).
kms              - Kac-Murdock-Szego Toeplitz matrix.
parter           - Parter matrix (Toeplitz with singular values near pi).
prolate          - Prolate matrix (symmetric, ill-conditioned Toeplitz matrix).
toeppd           - Symmetric positive definite Toeplitz matrix.
toeppen          - Pentadiagonal Toeplitz matrix (sparse).
>> help toeplitz
toeplitz Toeplitz matrix.
    toeplitz(C,R) is a non-symmetric Toeplitz matrix having C as its
    first column and R as its first row.

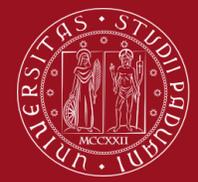
    toeplitz(R) is a symmetric Toeplitz matrix for real R.
    For a complex vector R with a real first element, T = toeplitz(r)
    returns the Hermitian Toeplitz matrix formed from R. When the
    first element of R is not real, the resulting matrix is Hermitian
    off the main diagonal, i.e.,  $T_{i,j} = \text{conj}(T_{j,i})$  for  $i \neq j$ .

    Class support for inputs C,R:
        float: double, single
        integer: uint8, int8, uint16, int16, uint32, int32, uint64, int64

    See also hankel.

    Documentation for toeplitz
    Other functions named toeplitz

>> doc toeplitz
fx>>
```



Risultato di

>> **doc toeplitz**

The screenshot shows the MATLAB Help Center interface. The browser title is "Help". The page title is "Help Center" with a search bar. The main content area is titled "toeplitz" and "Toeplitz matrix". It includes sections for "Syntax", "Description", and "Examples".

**Syntax**

```
T = toeplitz(c,r)
T = toeplitz(r)
```

**Description**

$T = \text{toeplitz}(c, r)$  returns a nonsymmetric Toeplitz matrix with  $c$  as its first column and  $r$  as its first row. If the first elements of  $c$  and  $r$  differ, `toeplitz` issues a warning and uses the column element for the diagonal.

$T = \text{toeplitz}(r)$  returns the symmetric Toeplitz matrix where:

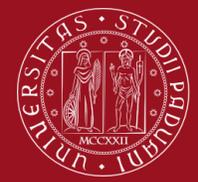
- If  $r$  is a real vector, then  $r$  defines the first row of the matrix.
- If  $r$  is a complex vector with a real first element, then  $r$  defines the first row and  $r'$  defines the first column.
- If the first element of  $r$  is complex, the Toeplitz matrix is Hermitian off the main diagonal, which means  $T_{i,j} = \text{conj}(T_{j,i})$  for  $i \neq j$ . The elements of the main diagonal are set to  $r(1)$ .

**Examples**

▼ Create Symmetric Toeplitz Matrix

```
r = [1 2 3];
toeplitz(r)
```

```
ans = 3x3
     1     2     3
     2     1     2
```



# Stampa di stringhe a schermo: disp

>> **disp('text')** – mostra il testo tra apici (stringa)

Es: `disp('have a nice day')`

`disp(['I am ', num2str(18), ' year old'])`

N.B. **num2str** – converte il numero in una stringa



# fprintf e sprintf

fprintf e sprintf sono molto più potenti di display

Esempi:

```
Command Window
>> number = 10; name = 'John';
>> fprintf('%s has %d apples\nWriting Pi with 4 decimal digits gives %1.4f\n', name, number, pi)
John has 10 apples
Writing Pi with 4 decimal digits gives 3.1416
```

Il simbolo % introduce uno specificatore di formato

Il primo %s significa: stampa la prima variabile (`name`) come stringa

Il secondo %d: stampa la seconda (`number`) come numero intero

La terza %1.4f: stampa la terza variabile (`pi`) come numero decimale con 4 cifre dopo la virgola

Differenza tra **fprintf** e **sprintf**: il primo stampa a schermo (o su file) il secondo crea una stringa





I grafici di Matlab sono mostrati in finestre dette figure

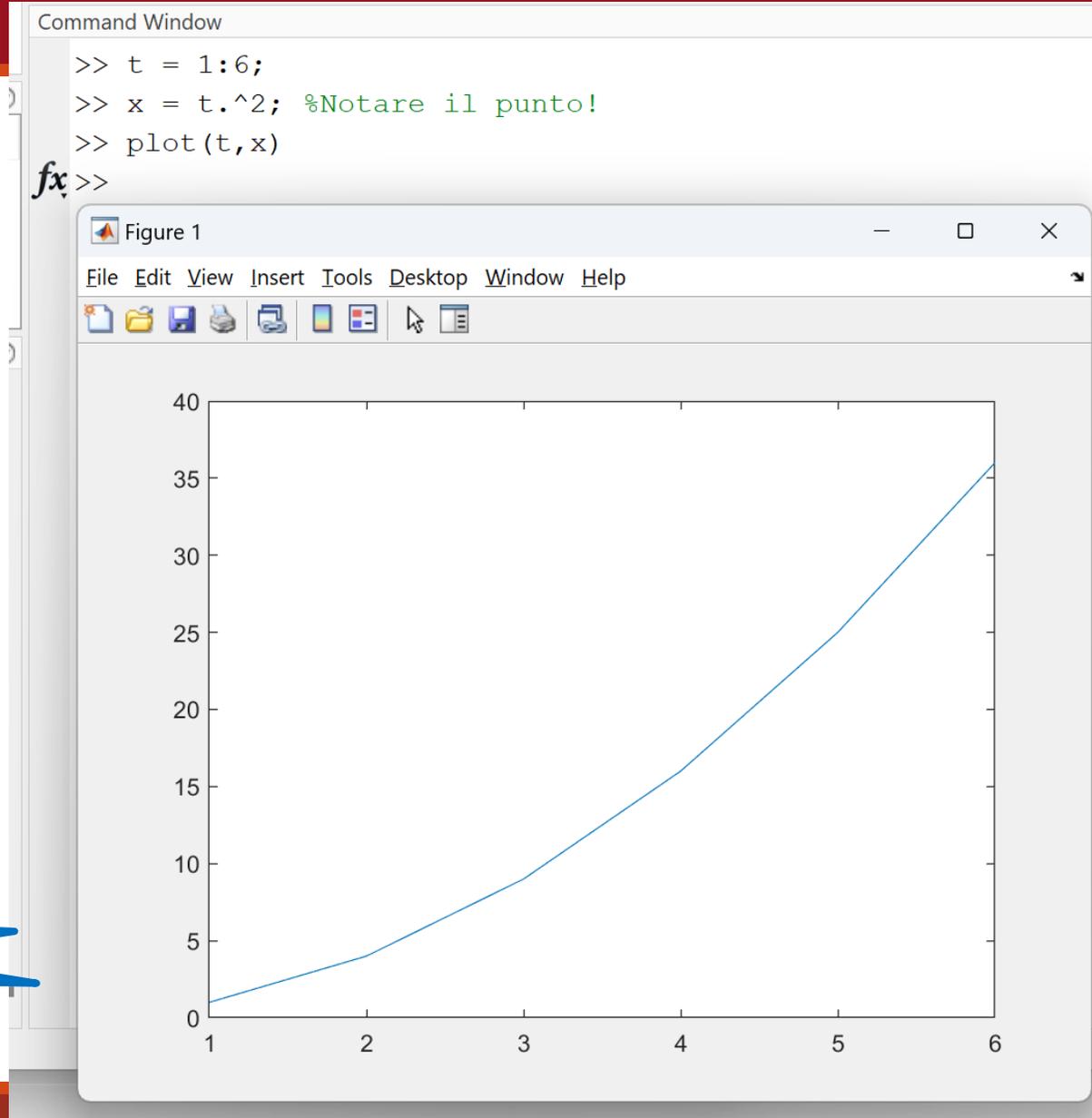
Il comando principale per creare un grafico è `plot`

`plot(t,x)` crea i punti di coordinate specificate dai parametri d'ingresso

C'è grande flessibilità nella parametrizzazione del plot, vedere gli esempi seguenti

**Domanda:** perché si usa `.`^ e non semplicemente ^ ?

Eseguite il codice mostrato





Dopo le ascisse  $t$  e le ordinate  $x$ , si possono specificare i parametri di formato in modo implicito o esplicito

La stringa 'ro--' determina:

Colore rosso del plot

Marker dei punti (cerchio)

Linea tratteggiata

La coppia di parametri 'Linewidth', 2  
specifica lo spessore della linea

Si nota che Matlab traccia il grafico unendo con  
segmenti di retta i punti specificati in  $t$  e  $x$

Per approfondire (in un secondo momento) riferirsi alla  
documentazione di plot con il comando  
doc plot

**Eseguite il codice mostrato**

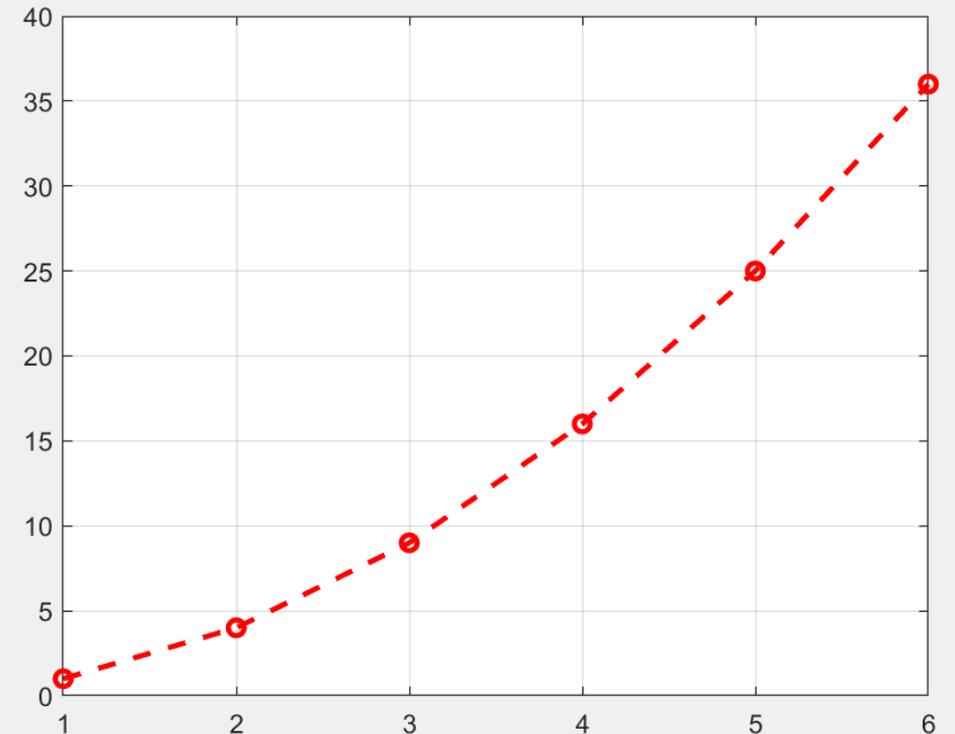
Command Window

```
>> t = 1:6;  
>> plot(t,x, 'ro--', 'LineWidth', 2);  
>> grid;
```

*fx* >>

Figure 1

File Edit View Insert Tools Desktop Window Help



## Matlab lavora a tempo discreto

I grafici 'continui' sono in realtà sempre discreti con un'opportuna scelta del passo dei campioni

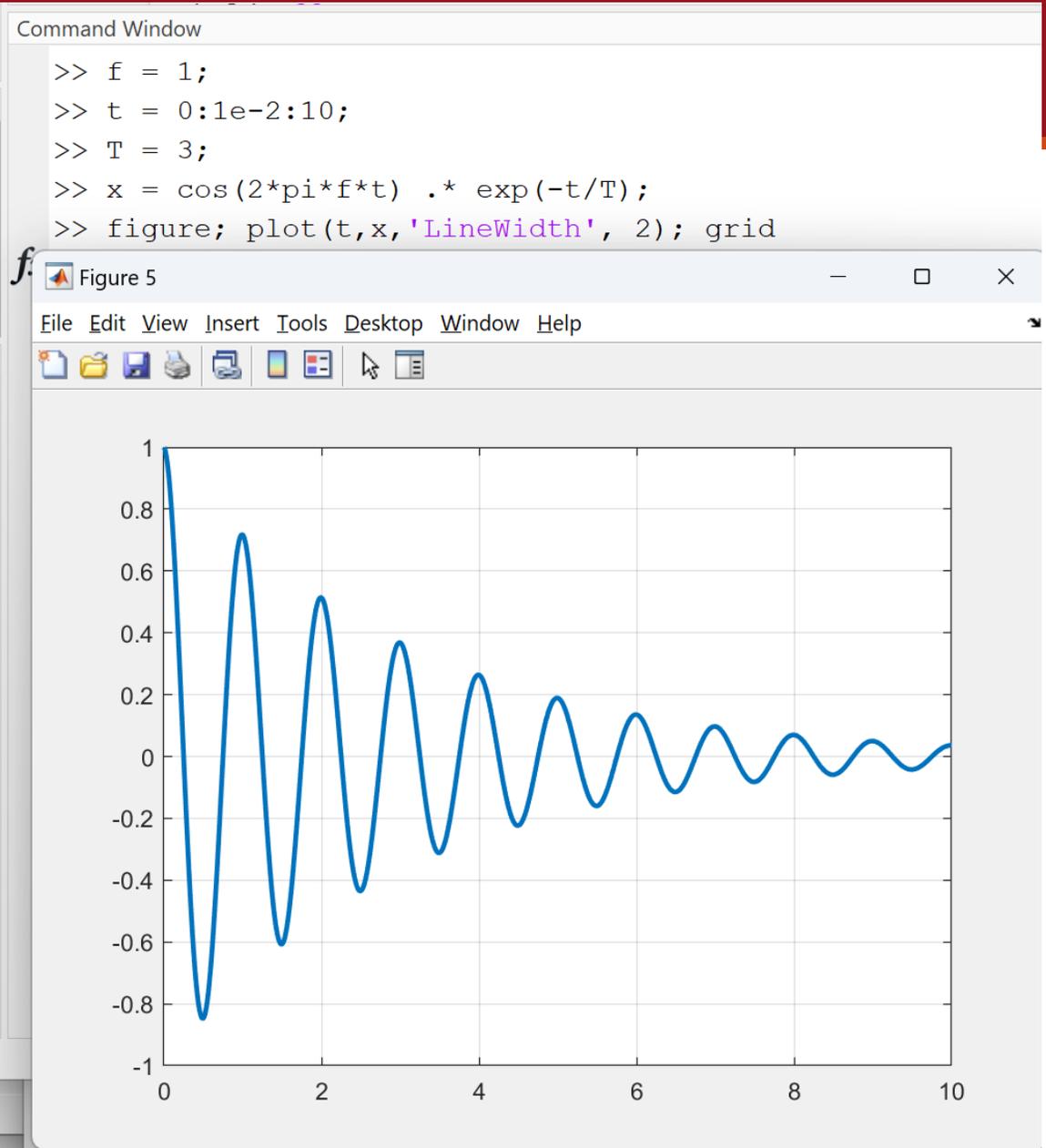
Per esempio, per disegnare una senoide smorzata con frequenza 1, scegliamo un passo di campionamento 1/100, che ci permette di disegnare 100 punti per periodo. Matlab disegnerà una linea retta tra punti consecutivi

`plot` per default disegna nell'ultima figura attiva e, se non ce ne sono, ne crea una nuova

Per forzare la creazione di una nuova figura, usare il comando `figure`

Per forzare `plot` ad usare una figura specifica, per esempio la figura 2, usare `figure(2)`

Eseguite il codice mostrato



# Grafici

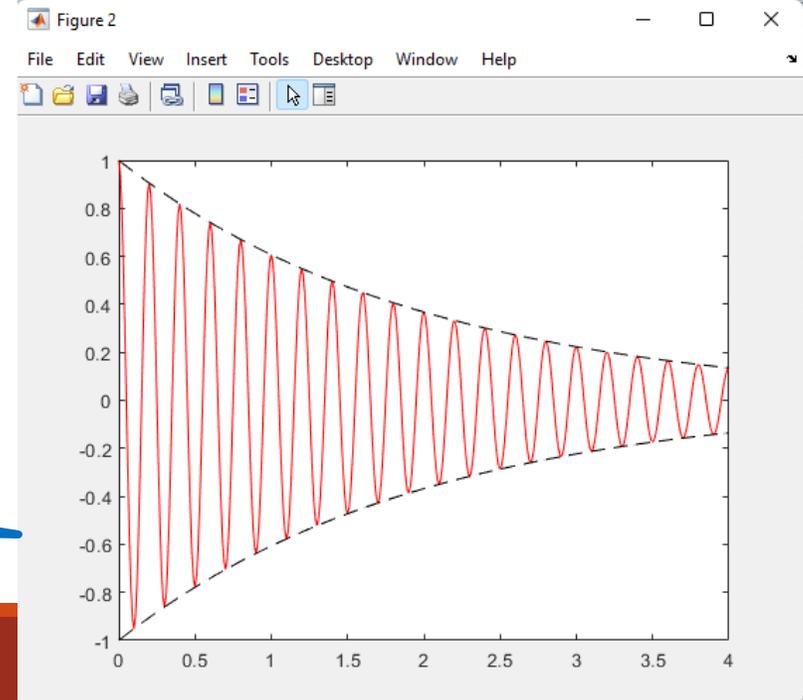
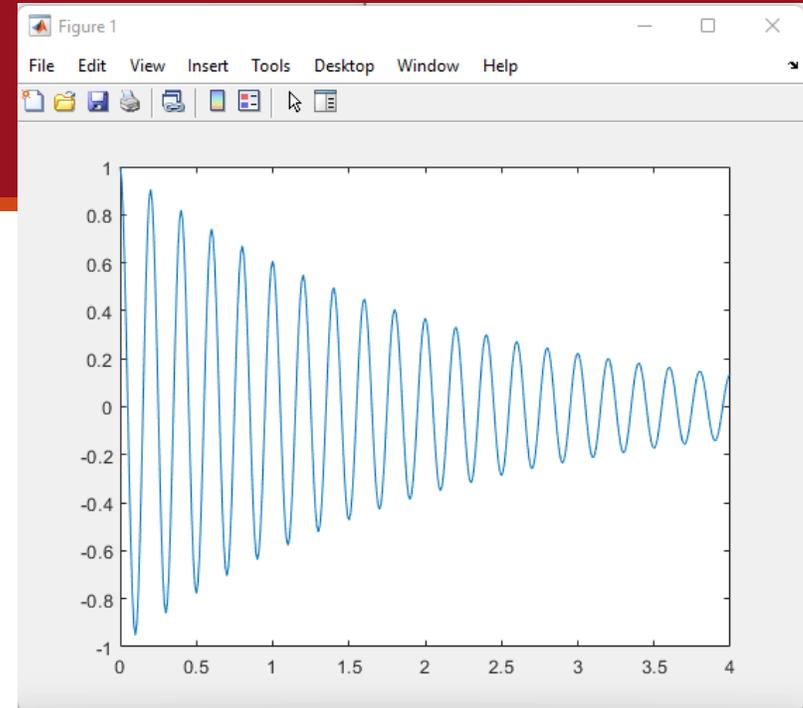
## Command Window

```
>> t = 0:1e-2:4;  
>> v1 = (cos(2*pi*f*t));  
>> v2 = exp(-t/2);  
>> x = v1.*v2;  
>> figure(1);  
>> plot(t,x);  
>> figure(2);  
>> plot(t,x,'r',t,v2,'--k',t,-v2,'--k');  
fx >> |
```

Nella figura 2, si disegnano tre curve, ognuna rappresentata da una terna: ascissa, ordinata, formato

formato è una stringa, dove si indica colore e/o formato della linea

Eseguite il codice mostrato

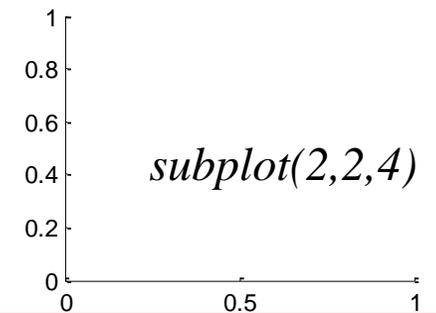
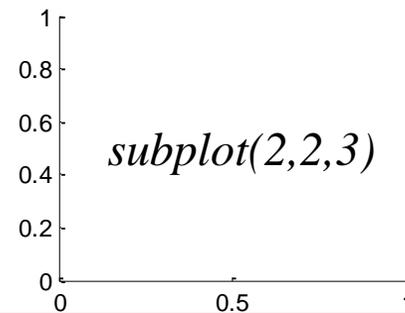
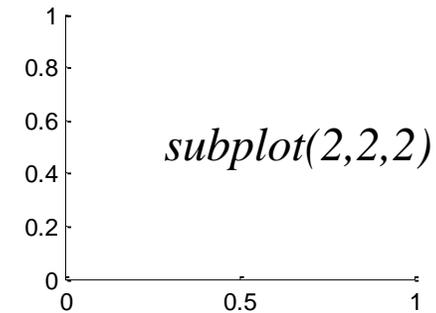
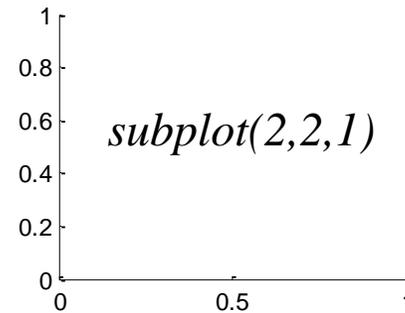




**subplot (m, n, k)** divide la figura in  $m \times n$  sottofinestre e mette il plot seguente nella finestra numero k

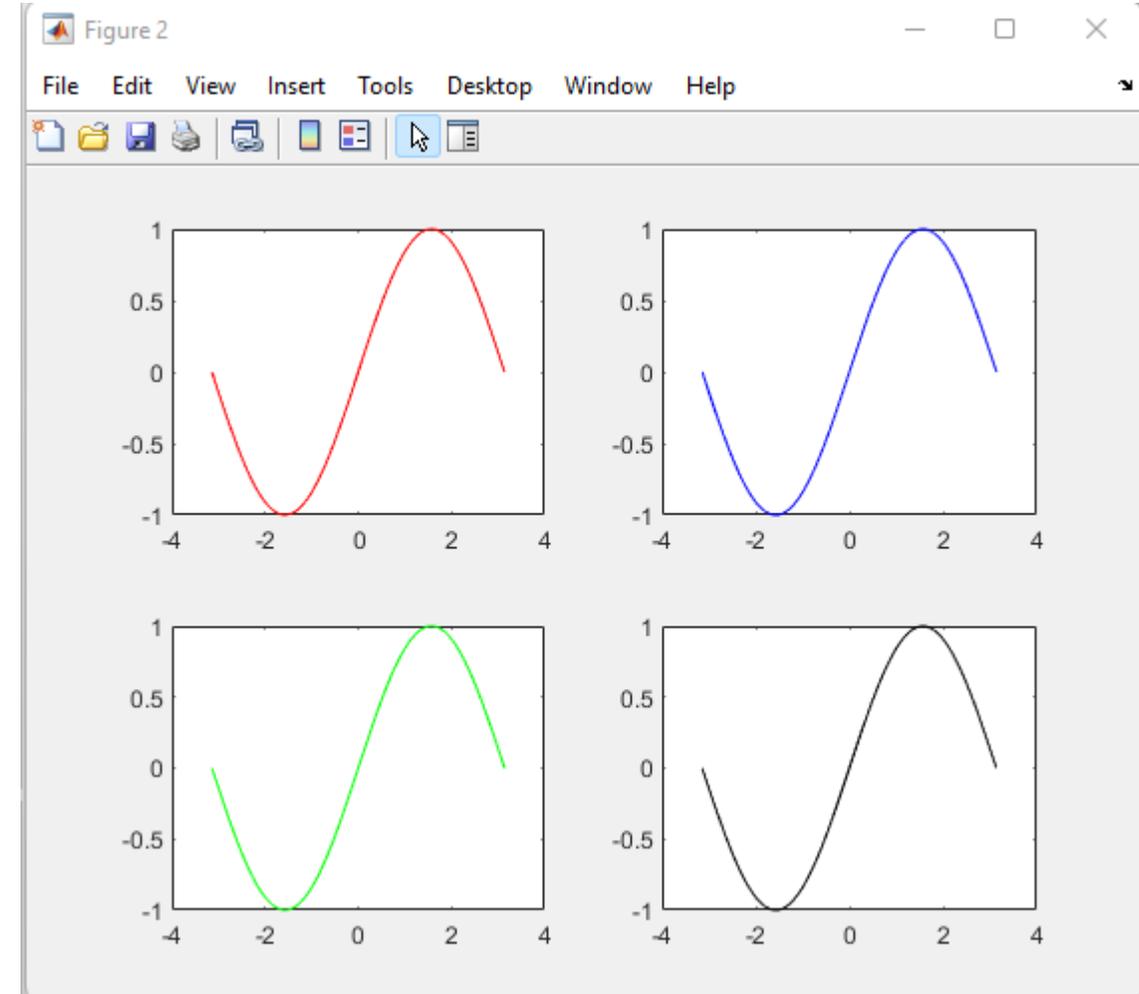
## Esempio

```
subplot(2,2,1)  
subplot(2,2,2)  
subplot(2,2,3)  
subplot(2,2,4)
```

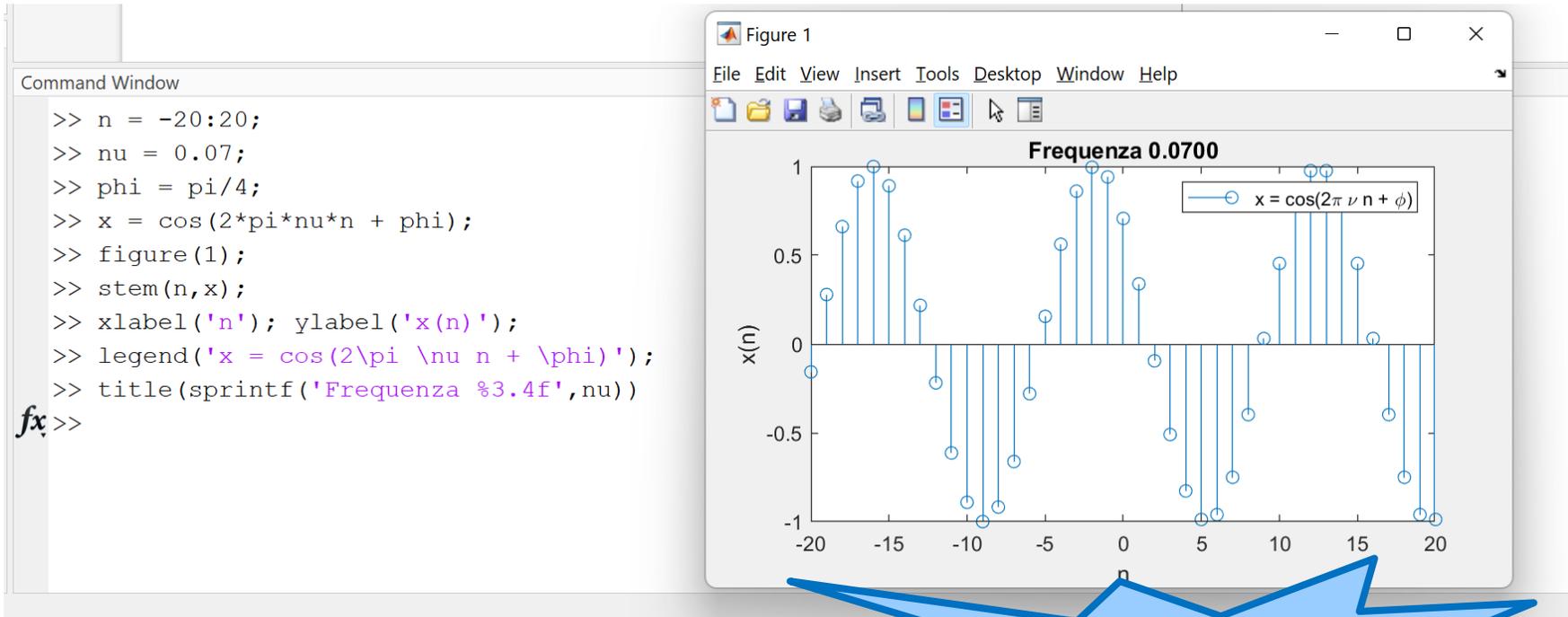


In ogni figura si può  
mettere un plot diverso

```
>> x = -pi:pi/100:pi;  
y = sin(x);  
subplot(2, 2, 1);  
plot(x, y, 'r')  
subplot(2, 2, 2);  
plot(x, y, 'b')  
subplot(2, 2, 3);  
plot(x, y, 'g')  
subplot(2, 2, 4);  
plot(x, y, 'k')
```



**stem** crea grafici adeguati ai segnali discreti



Cosa fanno i comandi xlabel, legend, title?

Qual è il periodo di questo segnale?

Ripetere i comandi, creando ogni volta una nuova figura, con  $\nu = 1.07$ ,  $\nu = 0.93$  e  $\nu = -0.07$

Che cosa osservate?

Eseguite il codice mostrato



Le strutture di programmazione si usano all'interno di un file (funzione o script) e si usano un modo molto simile a tutti gli altri linguaggi di programmazione

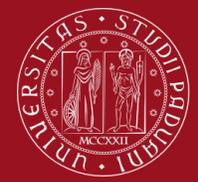
A differenza di altri linguaggi, il carattere usato per la negazione è la **tilde**

A destra il contenuto del file esempio\_strutture\_prog.m  
Apritelo ed eseguitelo

```
Editor - C:\Users\Marco Cagnazzo\OneDrive - Università degli Studi di Padova\didattica\corsi\signals_systems\23-24\la
freq_num.m x freq_num.m x sinusoidi_discrete.mlx x script_esempio.m x sinusoidi_discrete.mlx
1      %% Controllo dei parametri
2      t= 0:1e-2:10; f = -3; phi = 1; A=10;
3      x = A*cos(2*pi*f*t+phi);
4      if f<0
5          disp('La sinusoida NON è in forma canonica');
6      elseif f==0
7          disp('Segnale costante')
8      else
9          disp('Sinusoide in forma canonica');
10     end
11     if phi ~= 0
12         disp('Fase non nulla');
13     end
14     %% Stampa i quadrati
15
16     for number = 1:3
17         fprintf('Il quadrato di %d è %d\n',number, number*number);
18     end
19
20     %% ricerca fattori primi di un numero
21     N = 371;
22     number = 2;
23     while number<=N
24         if ~mod(N,number)
25             fprintf('%d diviso %d fa %d\n',N,number, N/number);
26             N= N/number;
27         else
28             number = number +1;
29         end
30     end
```



- Script e funzioni: .m
- Live script: .mlx
- Dati: .mat
- Figure: .fig



- Gli **script** sono file di testo che contengono una sequenza di comandi MATLAB
- Non accettano in ingresso degli argomenti, né restituiscono variabili in uscita. Lavorano sui dati del workspace (condivisione spazio di memoria)
- Sono comodi quando si devono ripetere lunghe sequenze di istruzioni, cambiando ogni volta qualche parametro
- Uno script, **una volta salvato**, può essere eseguito in vari modi:
  - Dalla command window, scrivere il nome del file (senza .m!)
    - Attenzione, il file deve essere nel Path oppure nella cartella corrente!
  - Dalla finestra dell'editor, premere F5
  - Dalla finestra dell'editor, cliccare su Run

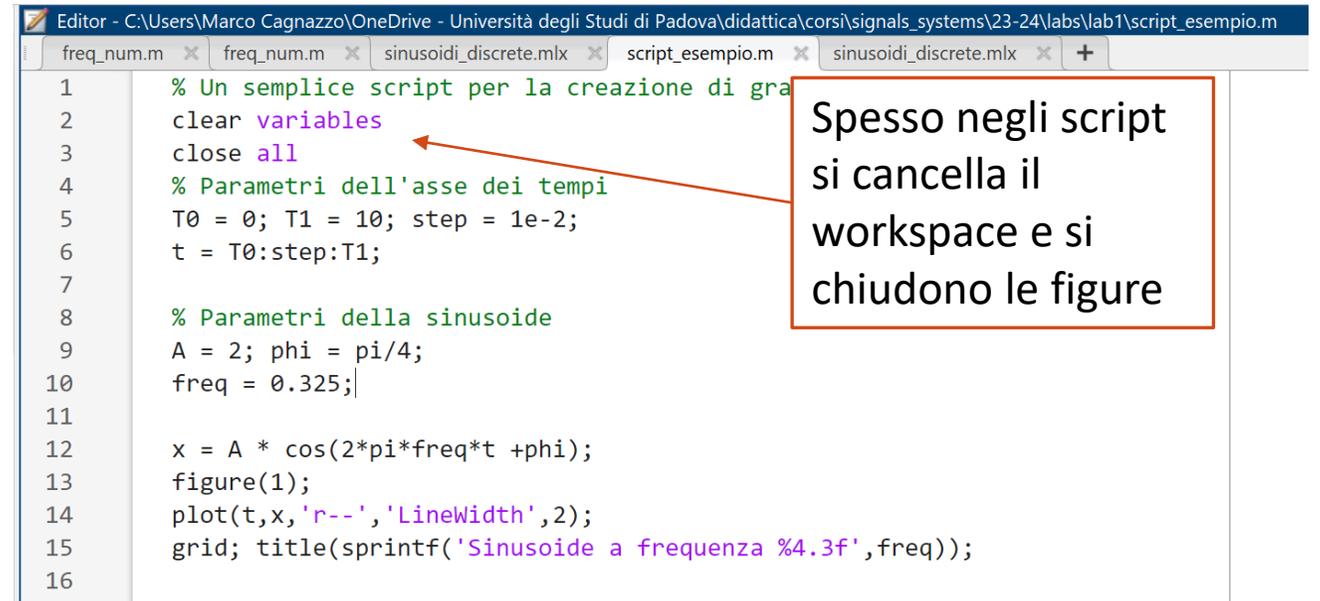
## Esercizio

Aprire nell'editor il file `script_esempio.m`

Eeguire il file dalla command window o con F5

Cambiare i parametri della sinusoide ed osservare il risultato

Osservare che le variabili create appaiono nel Workspace corrente



```
Editor - C:\Users\Marco Cagnazzo\OneDrive - Università degli Studi di Padova\didattica\corsi\signals_systems\23-24\labs\lab1\script_esempio.m
freq_num.m x freq_num.m x sinusoidi_discrete.mlx x script_esempio.m x sinusoidi_discrete.mlx x +
1 % Un semplice script per la creazione di gra
2 clear variables
3 close all
4 % Parametri dell'asse dei tempi
5 T0 = 0; T1 = 10; step = 1e-2;
6 t = T0:step:T1;
7
8 % Parametri della sinusoide
9 A = 2; phi = pi/4;
10 freq = 0.325;
11
12 x = A * cos(2*pi*freq*t +phi);
13 figure(1);
14 plot(t,x,'r--','LineWidth',2);
15 grid; title(sprintf('Sinusoide a frequenza %4.3f',freq));
16
```

Spesso negli script si cancella il workspace e si chiudono le figure



# Esempio di script con sezioni

Aprire il file `freq_num.m`  
(doppio click in Current  
Folder o scrivere `edit`  
`freq_num` nella command  
window  
Eeguire le sezioni del file  
come indicato; modificare i  
parametri ed eseguire di  
nuovo

```
Editor - C:\Users\cagnazzo\OneDrive - Università degli Studi di Padova\didattica\corsi\signals_systems\24-25\labs\lab1
+4  freq_num.m  convoluzione_tc.mlx  lti_denoising.mlx  script.mlx  script_serie_fourier.mlx
1  %% Grafici sinusoidi discrete: studio del valore della frequenza numerica
2
3  close all; % Chiude figure eventualmente aperte
4
5  %% Caso 1
6  A=1; phi = 0; nu = 0.03;
7  n = 0:40;
8  x = A*cos(2*pi*nu*n+phi);
9  % Creazione grafico
10 figure(1);
11 stem(n,x);
12 xlabel('n'); ylabel('x(n)');
13 title(sprintf('Freq: %4.3f',nu));
14
15 %% Caso 2
16 A=1; phi = 0; nu = 0.97;
17 n = 0:40;
18 x = A*cos(2*pi*nu*n+phi);
19 % Creazione grafico
20 figure(2);
21 stem(n,x);
22 xlabel('n'); ylabel('x(n)');
23 title(sprintf('Freq: %4.3f',nu));
24
```

Le sezioni sono delimitate da righe che cominciano con `%%` (due simboli di percento deguiti da almeno uno spazio)  
Nell'Editor, premendo `CTRL+ENTER` si esegue unicamente la Sezione corrente  
Con `CTRL+SHIFT+ENTER` si esegue la sezione corrente e si passa alla seguente  
Nei due casi precedenti il file NON viene salvato: utile per delle prove!  
Infine, con `F5` o `Run`, il file viene salvato ed eseguito interamente

# Esempio di funzione

## Esercizio

Aprire nell'editor il file `script_esempio.m`

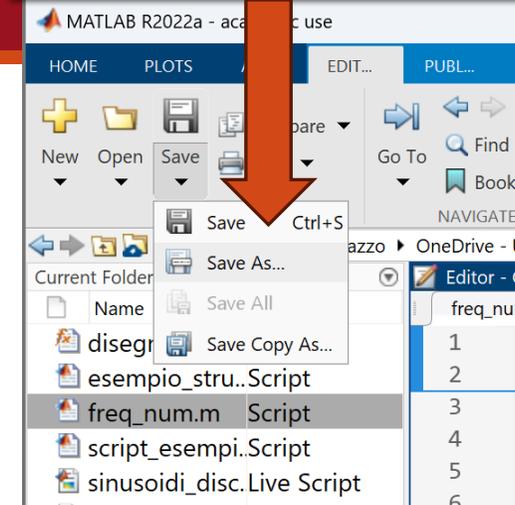
Eseguire **Save as** e salvarlo come `disegnaCos.m`

Trasformarlo in funzione cambiando la prima riga del file  
**ed eliminando le assegnazioni dei parametri d'ingresso, come  
mostrato**

Lanciare la funzione dalla command window, esempio:

```
>> x=disegnaCos(0:1e-2:5, 1, 0.2, 0);
```

Dove trovare Save as...



```
Editor - C:\Users\Marco Cagnazzo\OneDrive - Università degli Studi di Padova\didattica\corsi\signals_systems\23-24\labs\lab1\student
freq_num.m x sinusoidi_discrete.mlx * x esempio_strutture_progr.m x disegnaCos.m x script_esempio.m x +
1 function x = disegnaCos(t,A,freq,phi)
2 % disegnaCos calcola e disegna una sinusoide in forma canonica
3 %x = disegnaCos(t,A,freq,phi) calcola la sinusoide nei punti indicati dal
4 %parametro t. Gli altri parametri sono quelli della forma canonica:
5 %ampiezza, frequenza, fase iniziale
6
7 x = A * cos(2*pi*freq*t +phi);
8 figure(1);
9 plot(t,x,'r--','LineWidth',2);
10 grid; title(sprintf('Sinusoide a frequenza %4.3f',freq));
```

# Esempio di funzione

Un file 'funzione' deve cominciare con la keyword **function**

Dopo **function** c'è la lista dei parametri di uscita tra parentesi quadre, opzionali nel caso di un solo parametro

Segue il nome della funzione: **usare lo stesso nome del file**

Infine la lista di parametri d'ingresso tra parentesi tonde e separati dalla virgola

## Osservazioni importanti

```
1 function x = disegnaCos(t,A,freq,phi)
2 % disegnaCos calcola e disegna una senoide in forma canonica
3 %x = disegnaCos(t,A,freq,phi) calcola la senoide nei punti indicati dal
4 %parametro t. Gli altri parametri sono
5 %ampiezza, frequenza, fase iniziale
6
7 x = A * cos(2*pi*freq*t +phi);
8 figure(1);
9 plot(t,x,'r--','LineWidth',2);
10 grid; title(sprintf('Senoide a frequenza %4.3f',freq));
```

Il primo blocco di commenti è il testo che appare se si digita **help disegnaCos** nella command window (provare!)

**Lo spazio di memoria delle funzioni non è condiviso con la command window**  
**All'uscita dalla funzione, tutte le variabili interne non saranno più accessibili**



I **live script** sono la versione Matlab dei notebook

Si tratta di script con caratteristiche aggiuntive:

- Si può inframezzare del testo (CTRL+E) e delle formule (CTRL+SHIFT+E) al codice Matlab
- Si possono inserire slider, menù a tendina, ecc. per assegnare i valore dei parametri
- L'esecuzione è interattiva
- Si possono facilmente generare dei report

Aprirete il live script `sinusoidi_discrete.mlx` e provate ad utilizzarlo ed a modificarlo.