

---

**Laboratorio di Immagini**

**Esercitazione 1:**

**Introduzione a MATLAB**

---

Mauro Zucchelli  
09/03/2016

---

# MATLAB

---

**Cos'è MATLAB?**

---

# MATLAB

---

## Cos'è MATLAB?

- MATLAB **non** è un linguaggio di programmazione

# MATLAB

---

## Cos'è MATLAB?

- MATLAB **non** è un linguaggio di programmazione
  - MATLAB è un ambiente per il calcolo numerico, al cui interno contiene un linguaggio di programmazione proprietario
-

# MATLAB

---

## Cos'è MATLAB?

- MATLAB **non** è un linguaggio di programmazione
  - MATLAB è un ambiente per il calcolo numerico, al cui interno contiene un linguaggio di programmazione proprietario
  - MATLAB è disponibile sia per sistemi Unix (Mac Os e Linux) sia per Windows
-

# MATLAB

---

- **MATLAB** è la contrazione delle parole ***MA**Trix*  
*LAB*oratory

# MATLAB

---

- **MATLAB** è la contrazione delle parole ***MATrix***  
***LABoratory***
  - Come si evince dal nome, MATLAB è stato progettato per il calcolo matriciale ed è particolarmente efficiente nello svolgimento di questi calcoli
-

# L'interfaccia di MATLAB

The screenshot displays the MATLAB R2013a user interface. The top menu bar includes HOME, PLOTS, and APPS. Below it, a ribbon contains various toolbars for file operations, variable management, code execution, and environment settings. The main workspace is divided into three panes:

- Current Folder:** Shows a file explorer view of the current directory, listing folders like Devel, Documenti, Dropbox, and files like startx\_root.errors.
- Command Window:** Contains the MATLAB command prompt. The user has entered the command `x = 5`, and the output shows `x = 5`.
- Workspace:** A table showing the current workspace variables. It contains one variable, `x`, with a value of 5, a minimum of 5, and a maximum of 5.

Below the Workspace pane is the **Command History** pane, which records the commands entered in the Command Window. The history shows the command `x = 5` being executed at 14:59:27 on 09/03/2016.

| Name | Value | Min | Max |
|------|-------|-----|-----|
| x    | 5     | 5   | 5   |

```
-- Unknown date --%
-- 09/03/2016 14:20:27 --%
-- 09/03/2016 14:22:58 --%
-- 09/03/2016 14:23:50 --%
-- 09/03/2016 14:59:27 --%
x = 5
```

# L'interfaccia di MATLAB

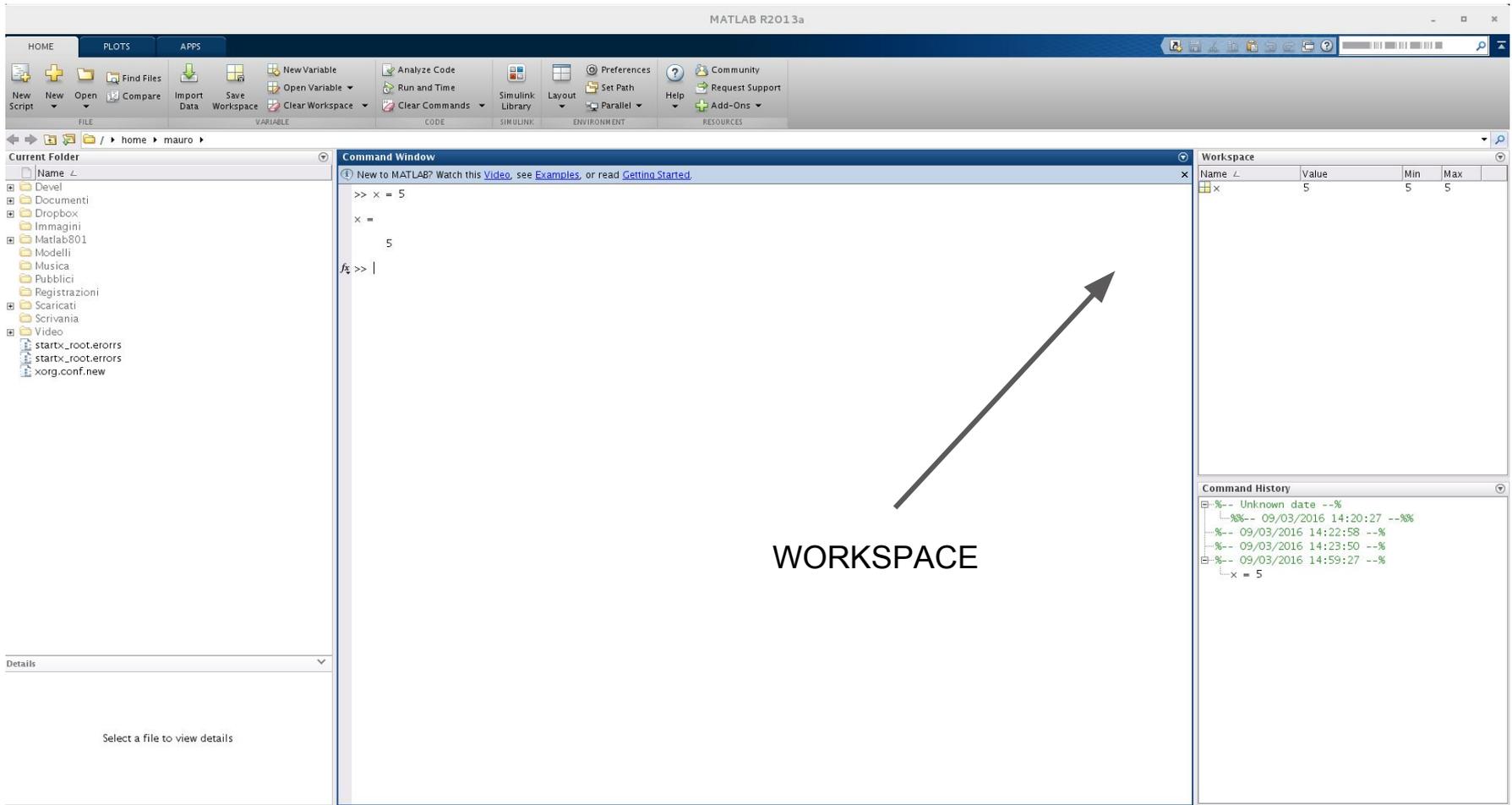
The screenshot displays the MATLAB R2013a software interface. The top menu bar includes options like HOME, PLOTS, and APPS. Below it, a ribbon contains various toolbars for file operations, variable management, code execution, and preferences. The main workspace is divided into several panes:

- Current Folder:** Shows a file explorer view of the current directory, listing folders like Devel, Documenti, and files like startx\_root.errors.
- Command Window:** Contains the MATLAB command prompt. The user has entered the command `x = 5`, which has been executed, resulting in `x = 5` being displayed. A black arrow points to the prompt area with the text "COMMAND WINDOW" below it.
- Workspace:** A table showing the current variables in the workspace. It has columns for Name, Value, Min, and Max. The variable `x` is listed with a value of 5, a minimum of 5, and a maximum of 5.
- Command History:** A list of previously executed commands, including the current command `x = 5`.

| Name | Value | Min | Max |
|------|-------|-----|-----|
| x    | 5     | 5   | 5   |

```
-- Unknown date --%
-- 09/03/2016 14:20:27 --%
-- 09/03/2016 14:22:58 --%
-- 09/03/2016 14:23:50 --%
-- 09/03/2016 14:59:27 --%
x = 5
```

# L'interfaccia di MATLAB



The screenshot displays the MATLAB R2013a software interface. The top menu bar includes options like HOME, PLOTS, and APPS. Below it, a ribbon contains various toolbars for file operations, variable management, code execution, and preferences. The main workspace is divided into several panes:

- Current Folder:** Shows a file explorer view of the current directory, listing folders like Devel, Documents, and files like startx\_root.errors.
- Command Window:** Contains the MATLAB command prompt. The user has entered the command `>> x = 5`, and the output shows `x =` followed by `5`. A large black arrow points from the text "WORKSPACE" below towards the Command Window.
- Workspace:** A table showing the current variables in the workspace. It has columns for Name, Value, Min, and Max. The variable `x` is listed with a value of 5, a minimum of 5, and a maximum of 5.
- Command History:** A list of previously executed commands, including the current command `x = 5`.

At the bottom left, there is a "Details" pane with the text "Select a file to view details".

# L'interfaccia di MATLAB

The screenshot displays the MATLAB R2013a software interface. The top menu bar includes options like HOME, PLOTS, and APPS. Below it is a ribbon with various toolbars for file operations, variable management, code execution, and preferences. The main workspace is divided into several panes:

- Current Folder:** Shows a file explorer view of the current directory, listing folders like Devel, Documents, and files like startx\_root.errors.
- Command Window:** Contains the MATLAB command prompt. The current session shows the command `>> x = 5` and the output `x = 5`. Below the prompt is a cursor and the text `fx >> |`.
- Workspace:** A table showing the current state of variables in memory. It has columns for Name, Value, Min, and Max. The variable `x` is listed with a value of 5, a minimum of 5, and a maximum of 5.
- Command History:** A list of previously executed commands, including the current one. It shows the command `x = 5` was executed on 09/03/2016 at 14:59:27.

An arrow points from the text "COMMAND HISTORY" to the Command History pane.

```
>> x = 5
x =
     5
fx >> |
```

| Name | Value | Min | Max |
|------|-------|-----|-----|
| x    | 5     | 5   | 5   |

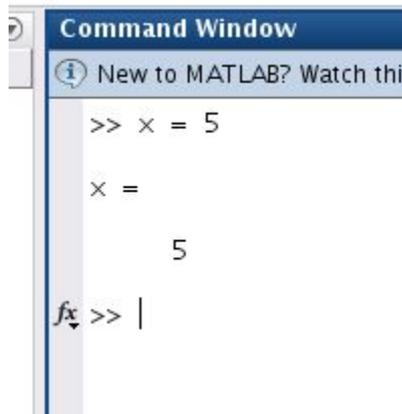
COMMAND HISTORY

```
-- Unknown date --%
-- 09/03/2016 14:20:27 --%
-- 09/03/2016 14:22:58 --%
-- 09/03/2016 14:23:50 --%
-- 09/03/2016 14:59:27 --%
x = 5
```

# MATLAB: comandi base

---

Dichiarazione di una variabile:



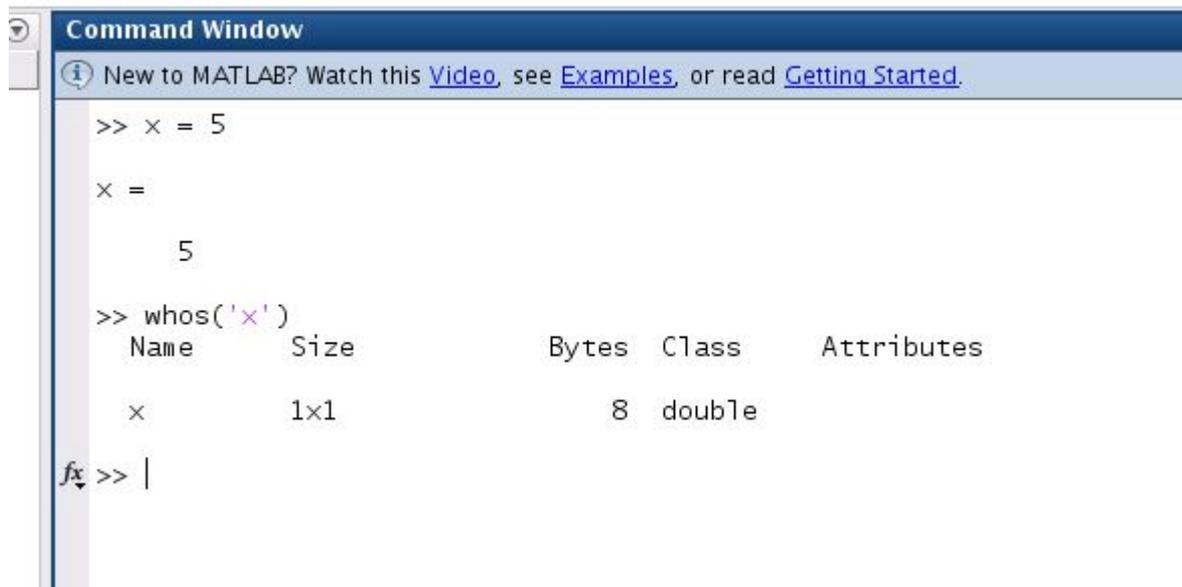
```
Command Window
New to MATLAB? Watch thi
>> x = 5
x =
    5
fx >> |
```

Le variabili in MATLAB non vanno dichiarate con il “tipo”

# MATLAB: comandi base

---

MATLAB è tipizzato implicitamente (a volte è più un problema che un vantaggio)



```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

>> x = 5

x =

    5

>> whos('x')
Name      Size      Bytes  Class  Attributes
x         1x1         8  double
```

*fx* >> |

# MATLAB: comandi base

---

La stessa variabile può essere riciclata

```
 ⓘ New to MATLAB? Watch this Video, see Examples, or read Getting Started.  
  
>> x = 5  
x =  
    5  
  
>> whos('x')  
  Name      Size      Bytes  Class  Attributes  
  x         1x1         8  double  
  
>> x = 'ciao'  
x =  
ciao  
  
>> whos('x')  
  Name      Size      Bytes  Class  Attributes  
  x         1x4         8  char  
  
fx >> |
```

# MATLAB: comandi base

---

## Operazioni base:

```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

>> x = 5;
>> y = 2;
>> x + y

ans =

    7

>> x - y

ans =

    3

>> x * y

ans =

   10

>> x / y

ans =

   2.5000

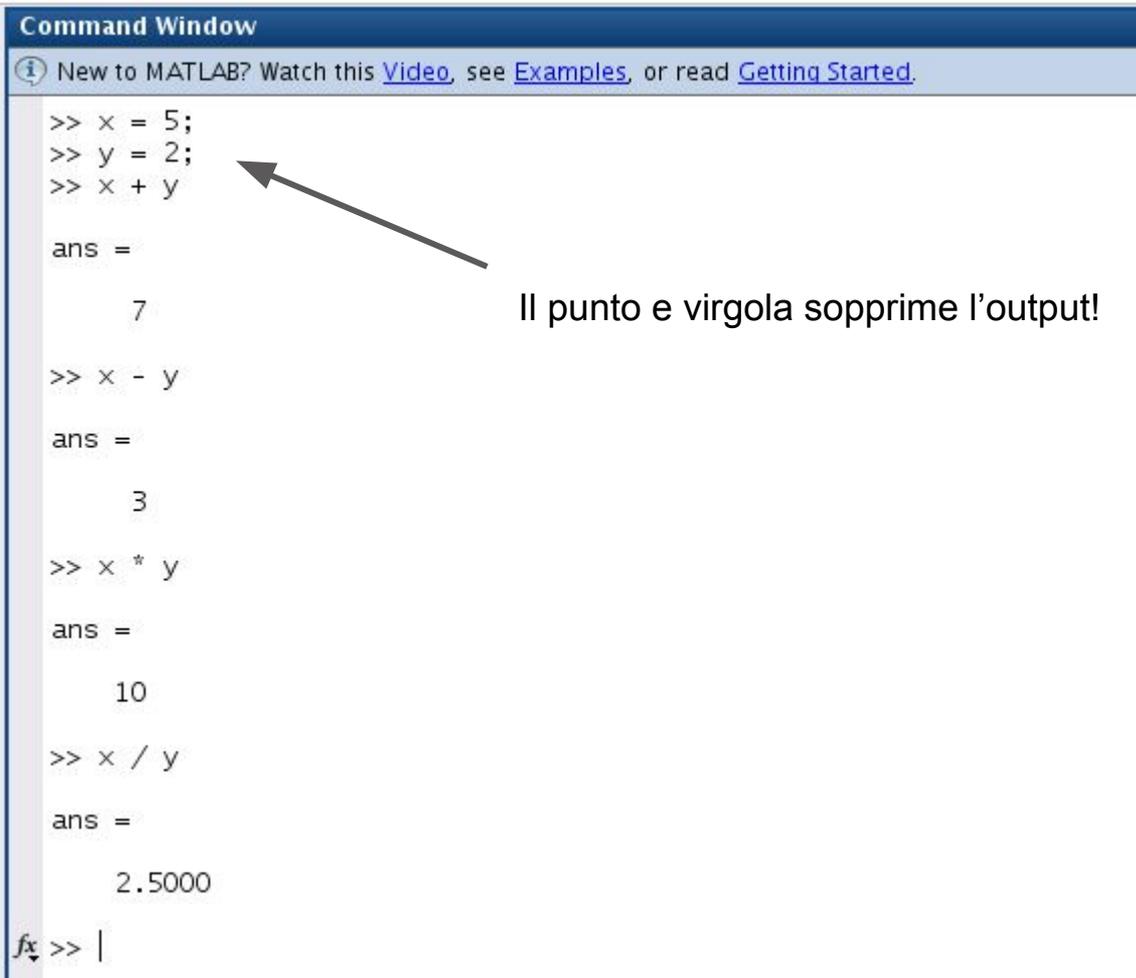
fx >> |
```

# MATLAB: comandi base

---

## Operazioni base:

```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.
>> x = 5;
>> y = 2;
>> x + y
ans =
    7
>> x - y
ans =
    3
>> x * y
ans =
   10
>> x / y
ans =
  2.5000
fx >> |
```



Il punto e virgola sopprime l'output!

# MATLAB: comandi base

---

## Costrutto "if"

```
>> if x == 5
y = 1
else
y = 0
end

y =

     1

fx >> |
```

# MATLAB: comandi base

---

## Costrutto “for”

```
>> x = 0;  
>> for i = 1:5  
x = x + 1  
end  
  
x =  
    1  
  
x =  
    2  
  
x =  
    3  
  
x =  
    4  
  
x =  
    5  
  
fx >> |
```

# MATLAB: array

---

In MATLAB ci sono vari modi per dichiarare un array

```
New to MATLAB? Watch this Video, see Examples, or read Getting Started

>> x = 1:5

x =

     1     2     3     4     5

>> x = [1, 2, 3, 4, 5]

x =

     1     2     3     4     5

>> x = [1; 2; 3; 4; 5]

x =

     1
     2
     3
     4
     5

fx >> |
```

# MATLAB: array

---

- In MATLAB gli array partono da 1
- Per accedere si usano le parentesi tonde

```
Command Window
(i) New to MATLAB? Watch this Video, see Examples, or read Getting Started.

>> x(1)

ans =

     1

>> x(5)

ans =

     5

>> x(1) = 0

x =

     0
     2
     3
     4
     5

fx >> |
```

# MATLAB: array

---

- L'operatore ":" permette di accedere a più valori

```
>> x(2:4)

ans =

     2
     3
     4

>> x(:)

ans =

     0
     2
     3
     4
     5

fx >> |
```

# MATLAB: array

---

- Operazioni tra array (controllate la dimensione)

```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

>> x = [1,2,3]

x =

     1     2     3

>> y = [1; 2; 3]

y =

     1
     2
     3

>> x + y
Error using +
Matrix dimensions must agree.
```

# MATLAB: array

---

- L'apice "traspone" gli array (e le matrici)

```
>> x + y'
```

```
ans =
```

```
     2     4     6
```

```
>> |
```

---

# MATLAB: array

---

- L'operatore '\*' di default indica il prodotto vettoriale!

```
Command Window
> New to MATLAB? Watch this Video, see Examples, or read Getting Started.
>> x * y
ans =
    14
>> x(1)*y(1) + x(2)*y(2) + x(3)*y(3)
ans =
    14
>> |
```

$$1 \times 3 * 3 \times 1 = 1 \times 1$$

```
>> y * x
ans =
     1     2     3
     2     4     6
     3     6     9
>> |
```

$$3 \times 1 * 1 \times 3 = 3 \times 3$$

# MATLAB: array

---

- Usate “ .\* ” per il prodotto tra elementi

```
>> x.*y'  
ans =  
     1     4     9  
* >> |
```

# MATLAB: matrici

```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

>> X = [1, 2, 3; 4, 5, 6]

X =

     1     2     3
     4     5     6

>> X = ones(2, 3)

X =

     1     1     1
     1     1     1

>> X = zeros(2, 4)

X =

     0     0     0     0
     0     0     0     0

>> X = rand(2,2)

X =

    0.8147    0.1270
    0.9058    0.9134

fx >> |
```

# MATLAB: matrici

---

- Per MATLAB le matrici funzionano esattamente come gli array
- Di fatto gli array sono matrici  $1 \times N$  (o  $N \times 1$ )

```
>> X = rand(2,2)
X =
    0.8147    0.1270
    0.9058    0.9134
>> X(1,2)
ans =
    0.1270
>> X(2,1)
ans =
    0.9058
fx >> |
```

```
>> X = [1,2; 3,4]
X =
     1     2
     3     4
>> X'
ans =
     1     3
     2     4
>> X + X
ans =
     2     4
     6     8
```

# MATLAB: matrici

---

Vale sempre la regola del prodotto: l'operatore “ \* “ indica il prodotto di matrici.

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

>> X.*X
ans =
     1     4
     9    16

fx >> |
```

# MATLAB: comandi utili

---

```
>> x = [1,2,3,4];  
>> length(x)
```

```
ans =  
     4
```

```
>> size(x)
```

```
ans =  
     1     4
```

```
>> x = rand(3,2)
```

```
x =  
    0.9575    0.9706  
    0.9649    0.9572  
    0.1576    0.4854
```

```
>> length(x)
```

```
ans =  
     3
```

```
>> size(x)
```

```
ans =  
     3     2
```

- il comando “*length*” ritorna la lunghezza di un vettore o il numero di righe di una matrice
- Il comando “*size*” ritorna il numero di righe e colonne

# MATLAB: comandi utili

---

```
>> % somma degli elementi di una matrice
>> X = [1,2; 3,4]

X =

     1     2
     3     4

>> somma = 0

somma =

     0

>> [x,y] = size(X);
>> for i=1:x
for j=1:y
somma = somma + X(i,j);
end
end
>> somma

somma =

    10

>> |
```

# MATLAB: comandi utili

---

MATLAB ha già implementato praticamente tutte le funzioni matematiche

```
>> sum(X(:))
```

```
ans =
```

```
10
```

```
>> mean(X(:))
```

```
ans =
```

```
2.5000
```

```
>> median(X(:))
```

```
ans =
```

```
2.5000
```

```
fx >> |
```

# MATLAB: comandi utili

---

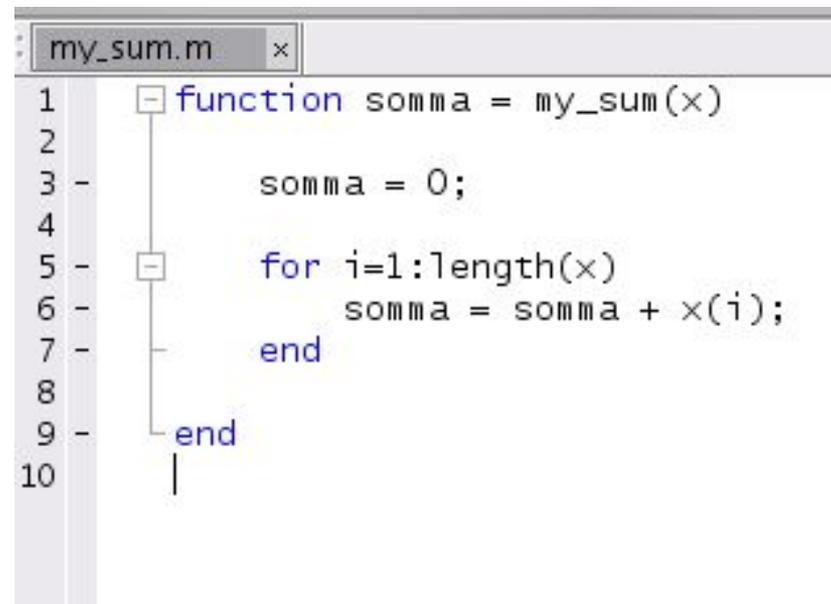
MATLAB ha già implementato praticamente tutte le funzioni matematiche

```
>> sum(X(:))  
ans =  
    10  
  
>> mean(X(:))  
ans =  
    2.5000  
  
>> median(X(:))  
ans =  
    2.5000  
  
fx >> |
```

# MATLAB: funzioni

---

In MATLAB possiamo definire le nostre funzioni in file con l'estensione **.m**



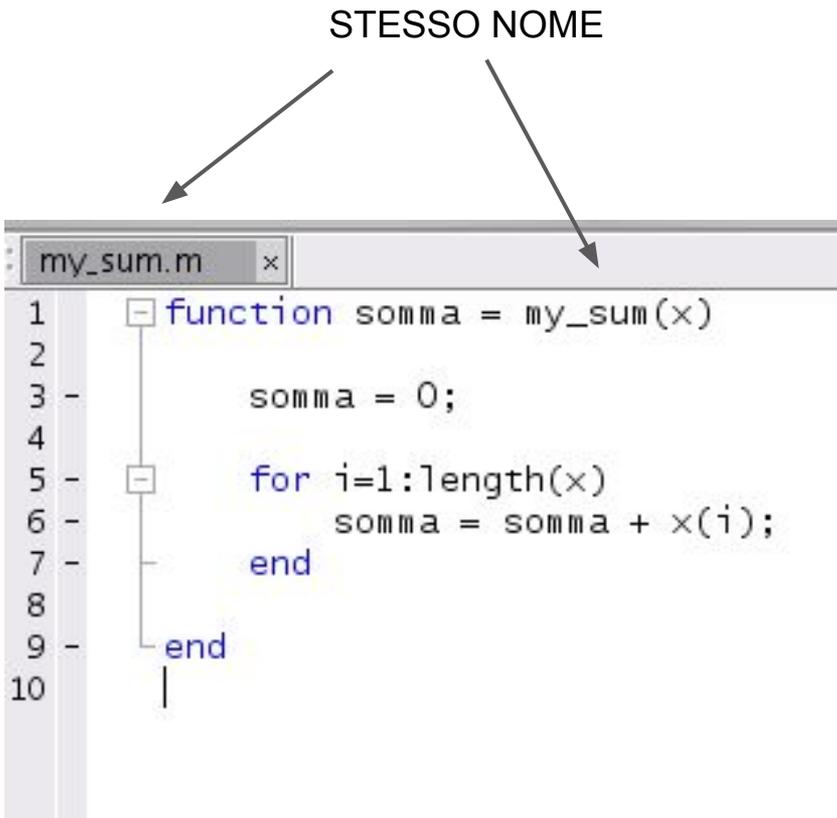
```
my_sum.m x
1  function somma = my_sum(x)
2
3      somma = 0;
4
5      for i=1:length(x)
6          somma = somma + x(i);
7      end
8
9  end
10
```

# MATLAB: funzioni

---

In MATLAB possiamo definire le nostre funzioni in file con l'estensione **.m**

STESSO NOME



```
my_sum.m x
1  function somma = my_sum(x)
2
3      somma = 0;
4
5      for i=1:length(x)
6          somma = somma + x(i);
7      end
8
9  end
10 |
```

# MATLAB: funzioni

---

In MATLAB possiamo definire le nostre funzioni in file con l'estensione **.m**

RETURN  
IMPLICITO

```
my_sum.m x
1  function somma = my_sum(x)
2
3      somma = 0;
4
5      for i=1:length(x)
6          somma = somma + x(i);
7      end
8
9  end
10
```

# MATLAB: attenzione

---

MATLAB è super-ottimizzato per le operazioni matriciali, non usiamo i cicli se non strettamente necessario!

```
>> x = ones(10000,1);  
>> tic; x_somma = sum(x); toc  
Elapsed time is 0.000041 seconds.  
>> x_somma
```

```
x_somma =  
  
    10000
```

```
fx >> |
```

```
>> x = ones(10000,1);  
>> tic; x_somma = my_sum(x); toc  
Elapsed time is 0.002982 seconds.  
>> x_somma
```

```
x_somma =  
  
    10000
```

```
fx >> |
```

# MATLAB: attenzione

---

MATLAB è super-ottimizzato per le operazioni matriciali, non usiamo i cicli se non strettamente necessario!

```
>> x = ones(10000,1);  
>> tic; x_somma = sum(x); toc  
Elapsed time is 0.000041 seconds.  
>> x_somma
```

```
x_somma =  
  
    10000
```

```
fx >> |
```

```
>> x = ones(10000,1);  
>> tic; x_somma = my_sum(x); toc  
Elapsed time is 0.002982 seconds.  
>> x_somma
```

```
x_somma =  
  
    10000
```

```
fx >> |
```

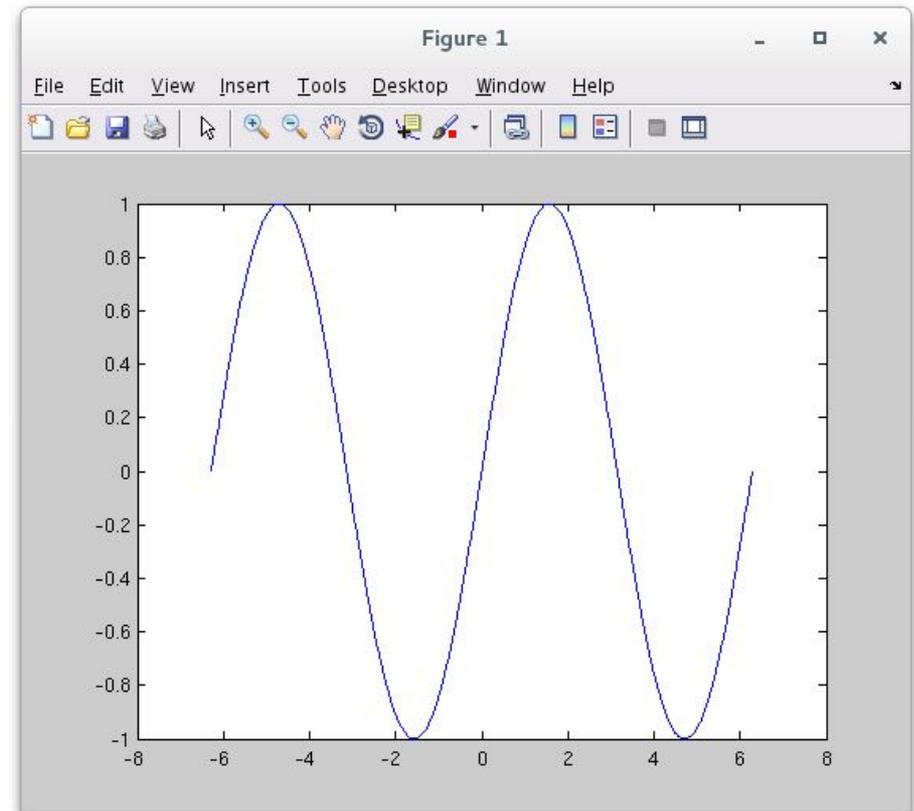
La funzione **sum** di MATLAB è 72 volte più veloce di **my\_sum**

---

# MATLAB: plot

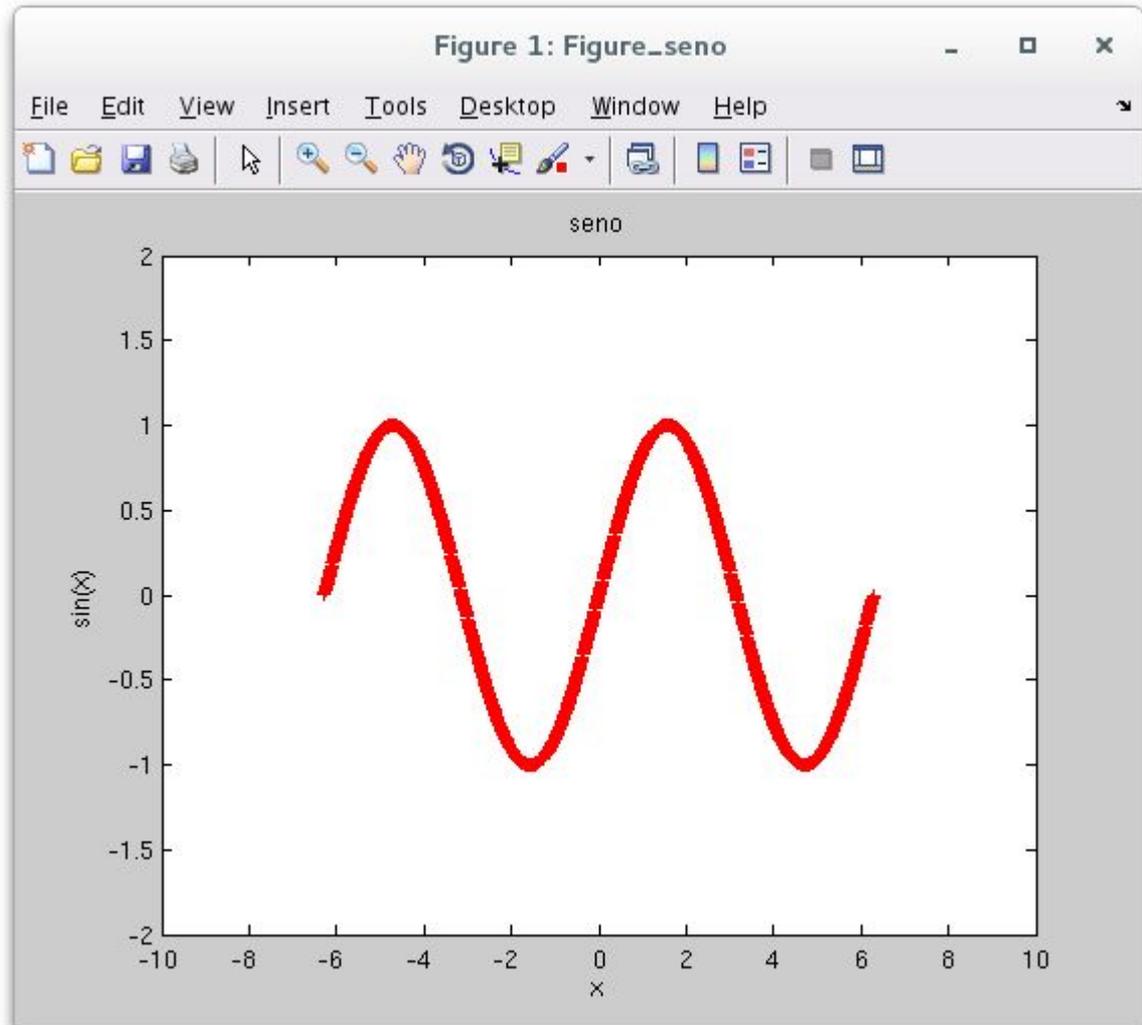
---

```
>> x = linspace(-2*pi,2*pi,1000);  
>> y = sin(x);  
>> plot(x,y)  
>>
```



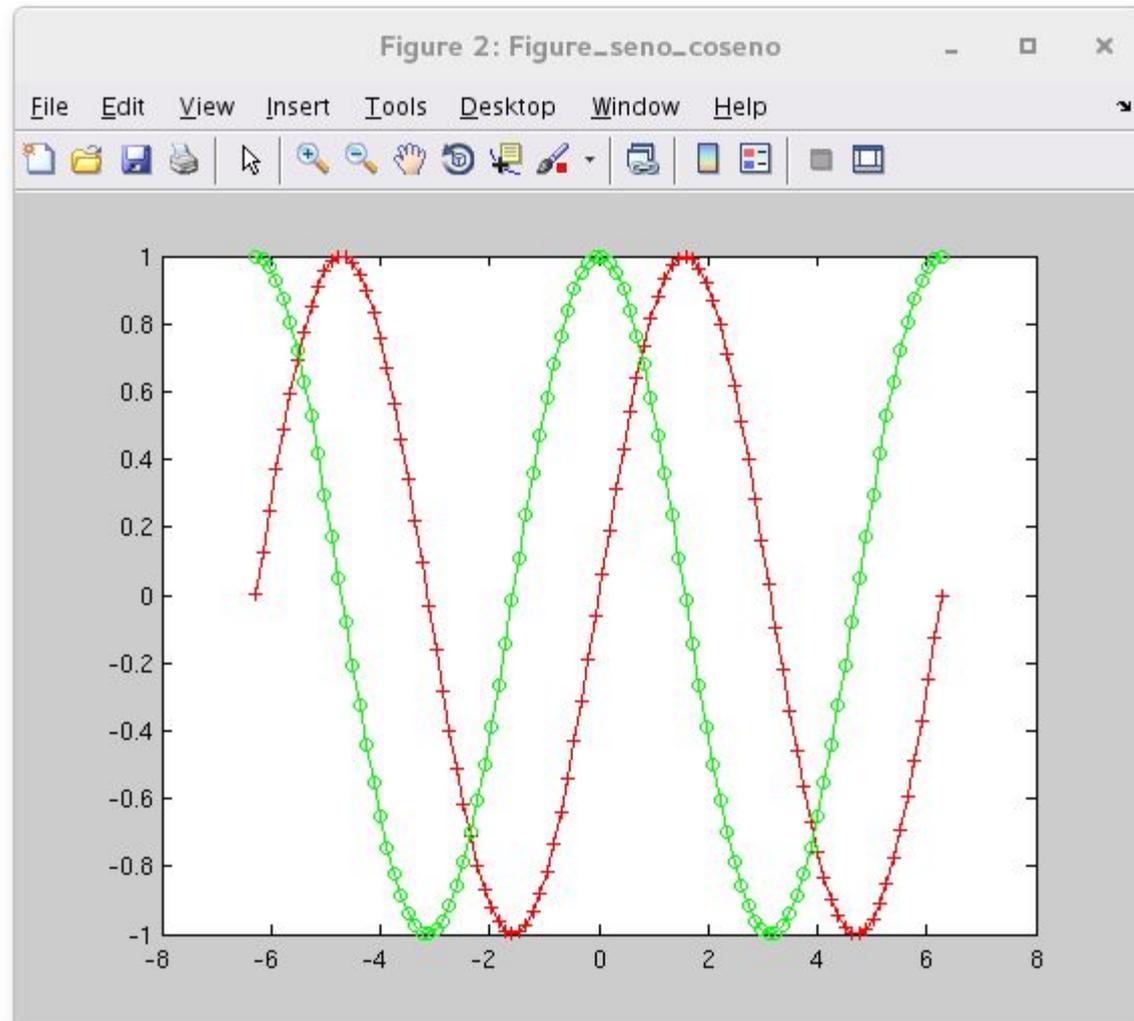
# MATLAB: plot

```
>> x = linspace(-2*pi,2*pi,1000);  
>> y = sin(x);  
>> figure('Name','Figure_seno');  
>> plot(x,y,'r+-')  
>> xlabel('x');  
>> ylabel('sin(x)');  
>> title('seno');  
>> axis([-10,10,-2,2]);  
>> |
```



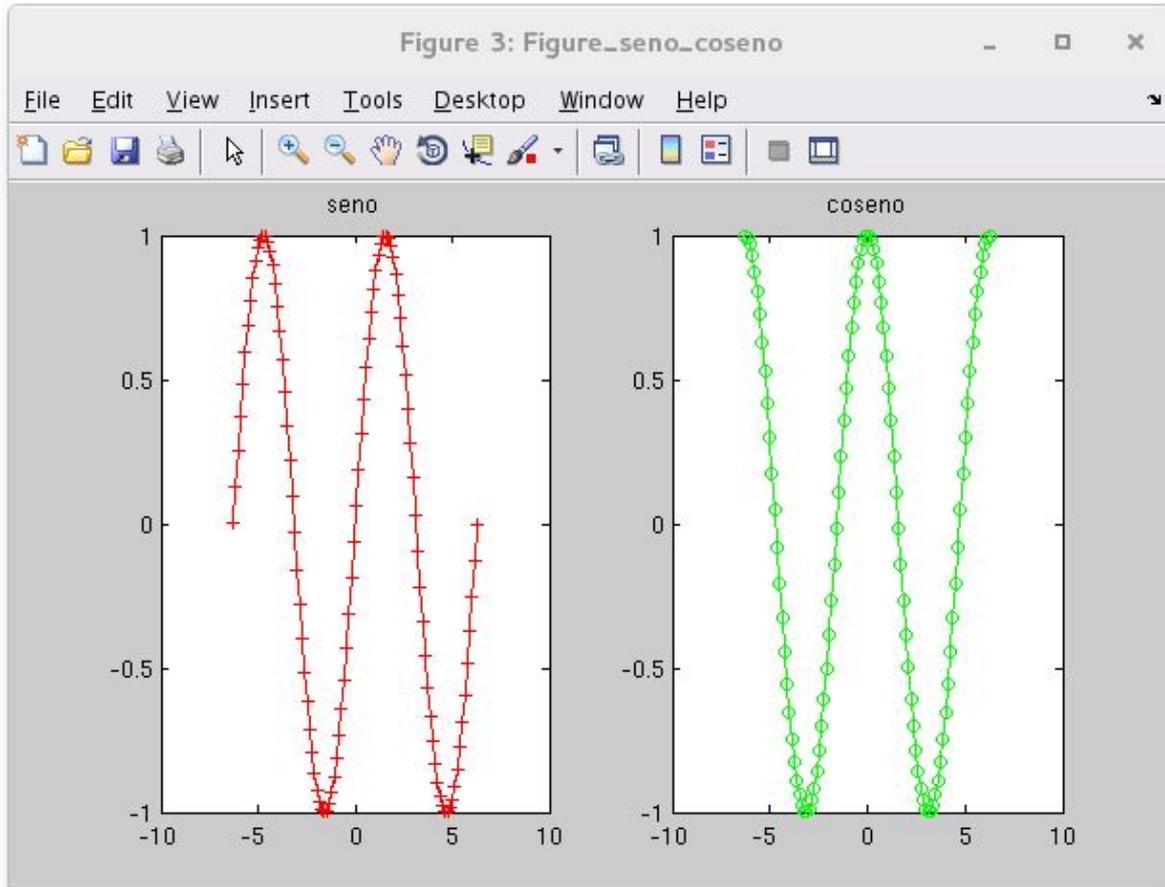
# MATLAB: multiple plots: “hold on”

```
>> x = linspace(-2*pi,2*pi,100);  
>> y = sin(x);  
>> z = cos(x);  
>> figure('Name','Figure_seno_coseno');  
>> plot(x,y,'r+-')  
>> hold on;  
>> plot(x,z,'go-')  
>> hold off;  
>> |
```



# MATLAB: multiple plots: “subplot”

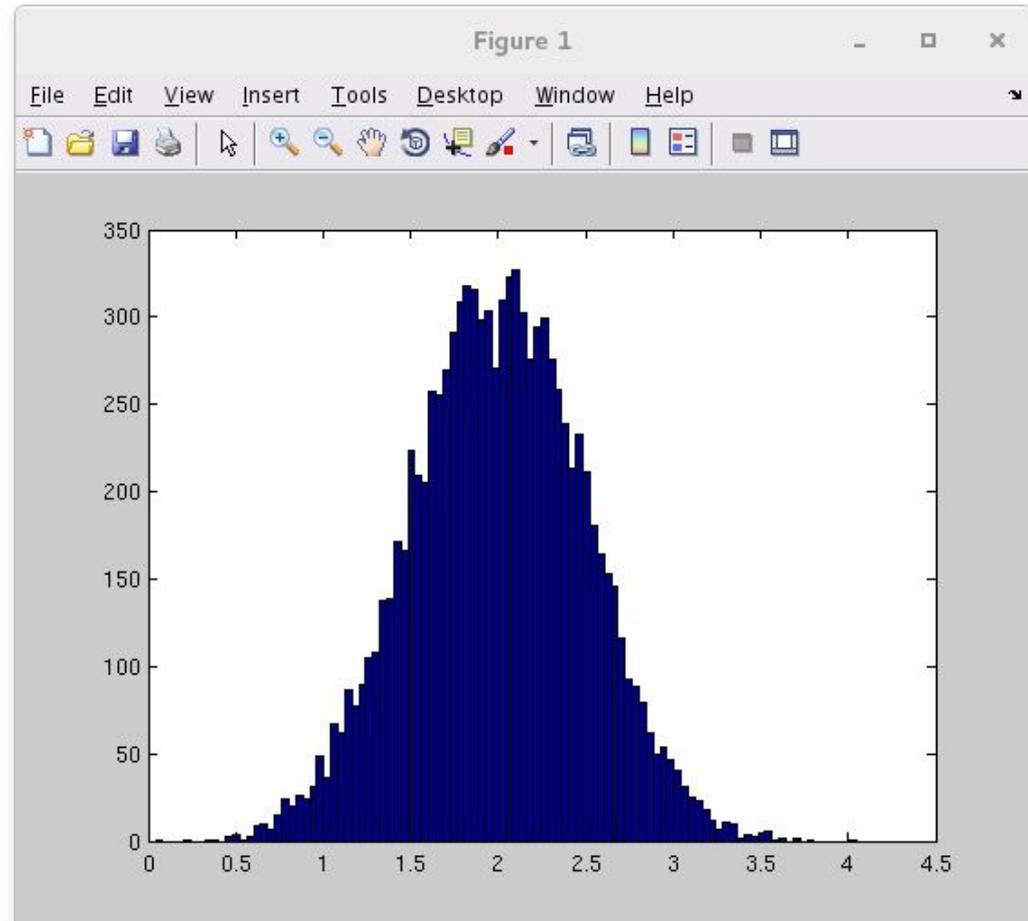
```
>> x = linspace(-2*pi,2*pi,100);  
>> y = sin(x);  
>> z = cos(x);  
>> figure('Name','Figure_seno_coseno');  
>> subplot(1,2,1)  
>> plot(x,y,'r+-')  
>> title('seno')  
>> subplot(1,2,2)  
>> plot(x,z,'go-')  
>> title('coseno')  
>> |
```



# MATLAB: histograms

---

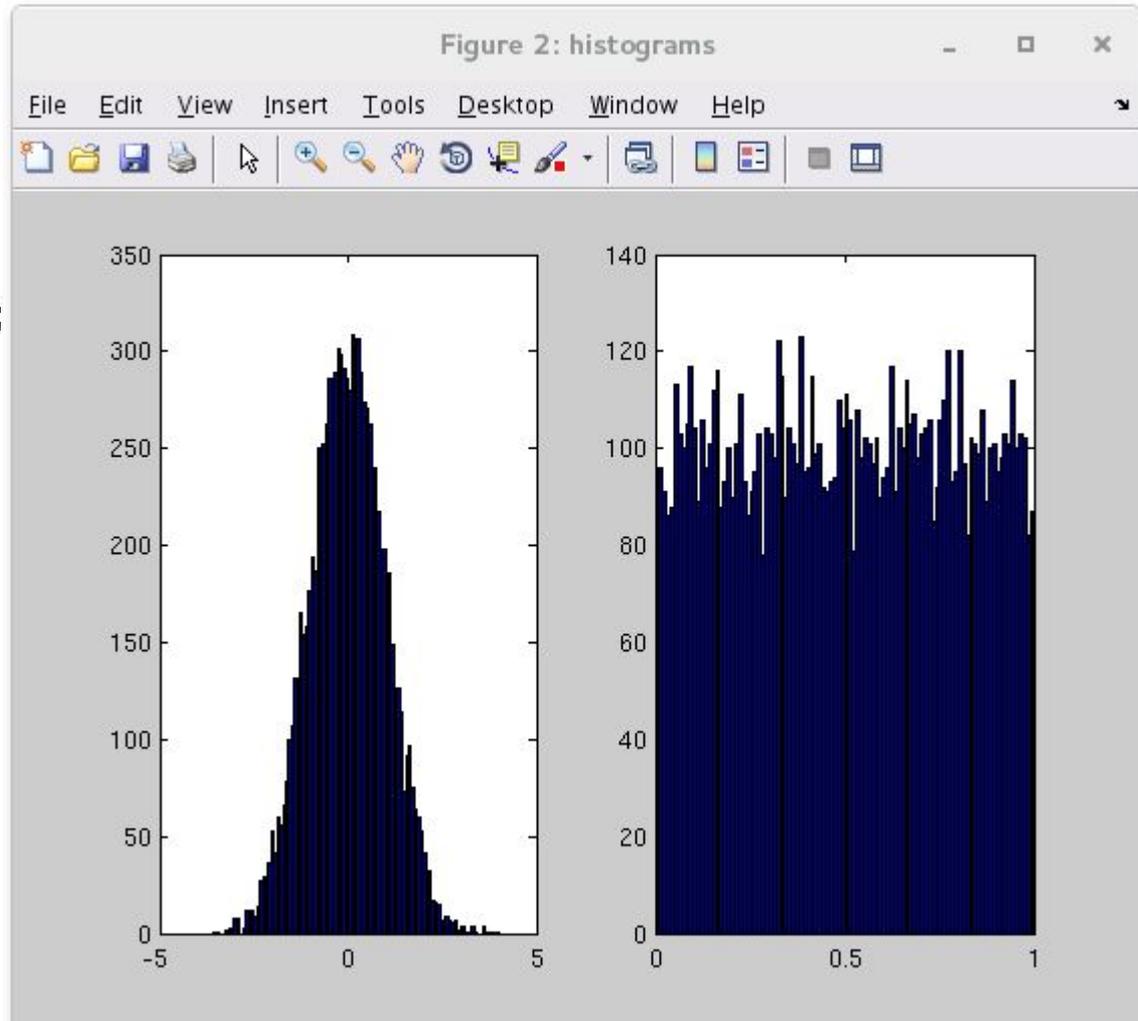
```
>> x = randn(1,10000);  
>> hist(x,100)  
>> x = randn(1,10000)*0.5 + 2;  
>> hist(x,100)  
>> |
```



# MATLAB: histograms

---

```
>> figure('Name','histograms');  
>> subplot(1,2,1)  
>> x = randn(1,10000);  
>> hist(x,100)  
>> subplot(1,2,2)  
>> x = rand(1,10000);  
>> hist(x,100)  
>> |
```



**MATLAB**

---

**LE IMMAGINI**

---

# MATLAB: le immagini

---

- MATLAB vede le immagini come matrici:
  - Immagini in bianco e nero (scala di grigi) come matrici  $N \times M$ ;
  - Immagini a colori come matrici  $N \times M \times 3$  (RGB);

# MATLAB: le immagini

---

- MATLAB vede le immagini come matrici:
  - Immagini in bianco e nero (scala di grigi) come matrici  $N \times M$ ;
  - Immagini a colori come matrici  $N \times M \times 3$  (RGB);

# MATLAB: le immagini

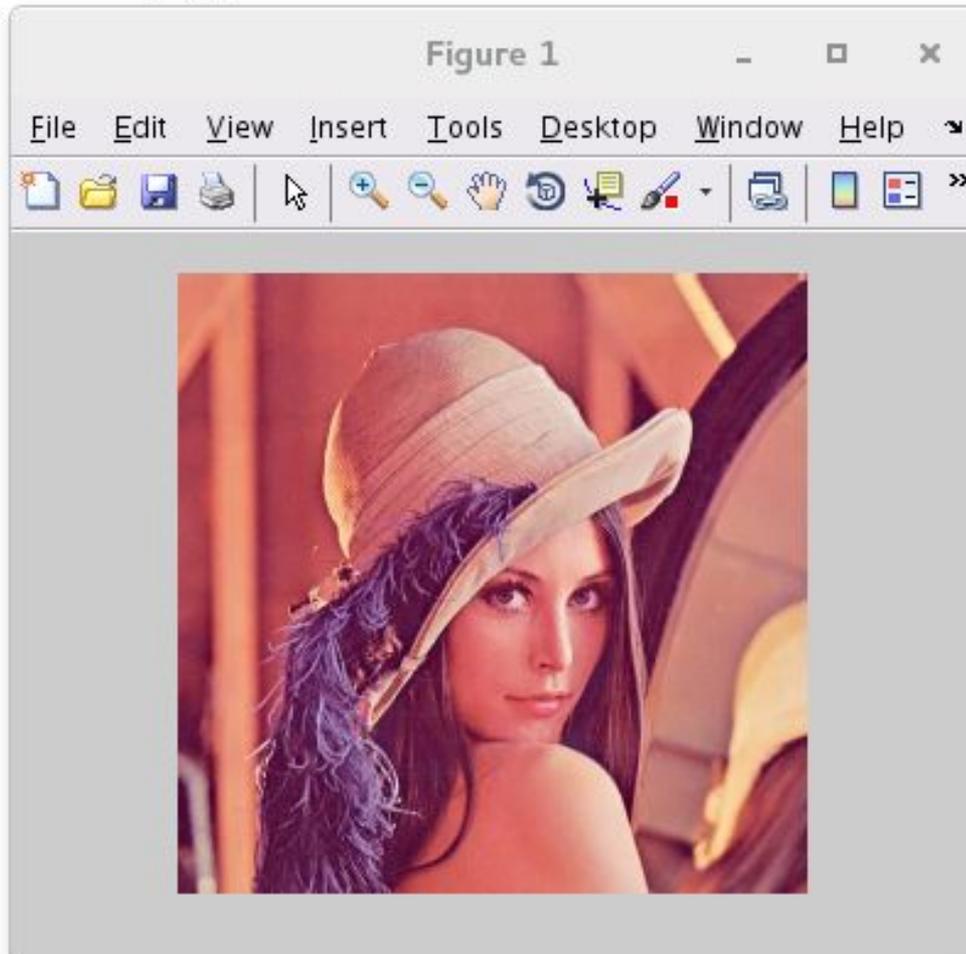
---

```
>> img = imread('lena.jpg');  
>> whos img  
  Name      Size           Bytes  Class  Attributes  
  
  img      512x512x3       786432  uint8  
  
>> 512*512*3  
  
ans =  
  
    786432  
  
>> |
```

# MATLAB: le immagini

---

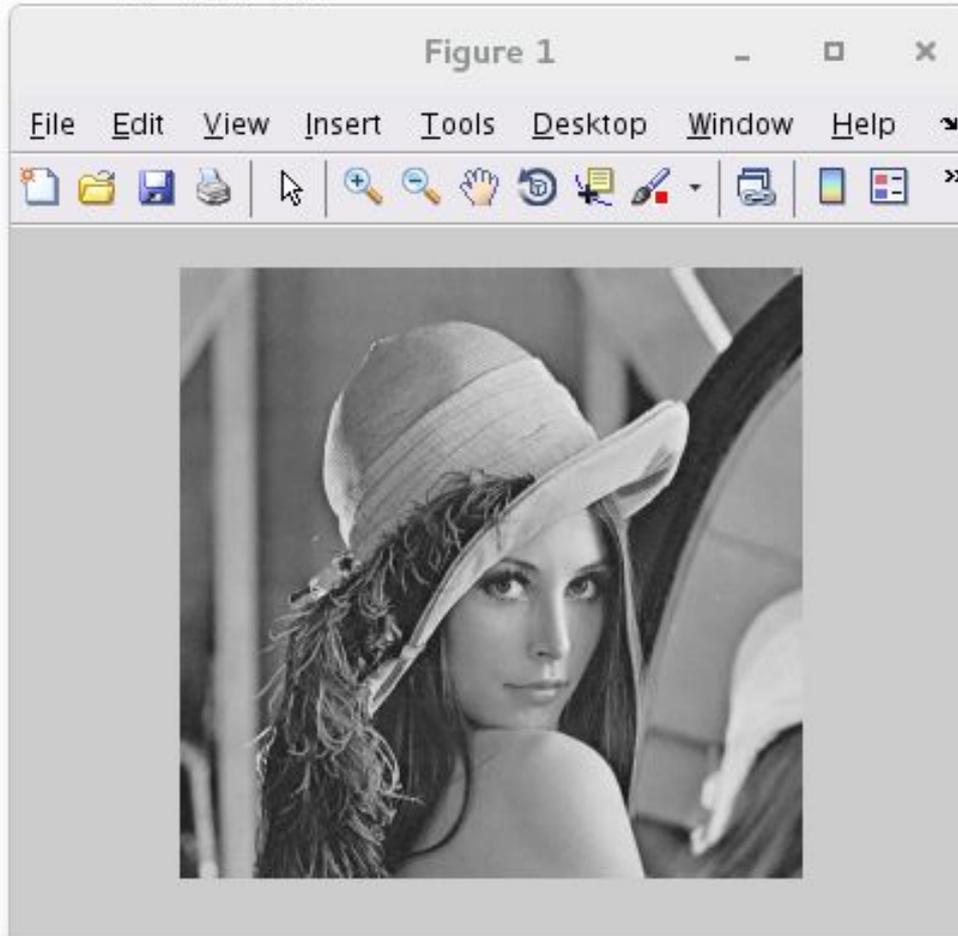
```
>> imshow(img)  
>>
```



# MATLAB: le immagini

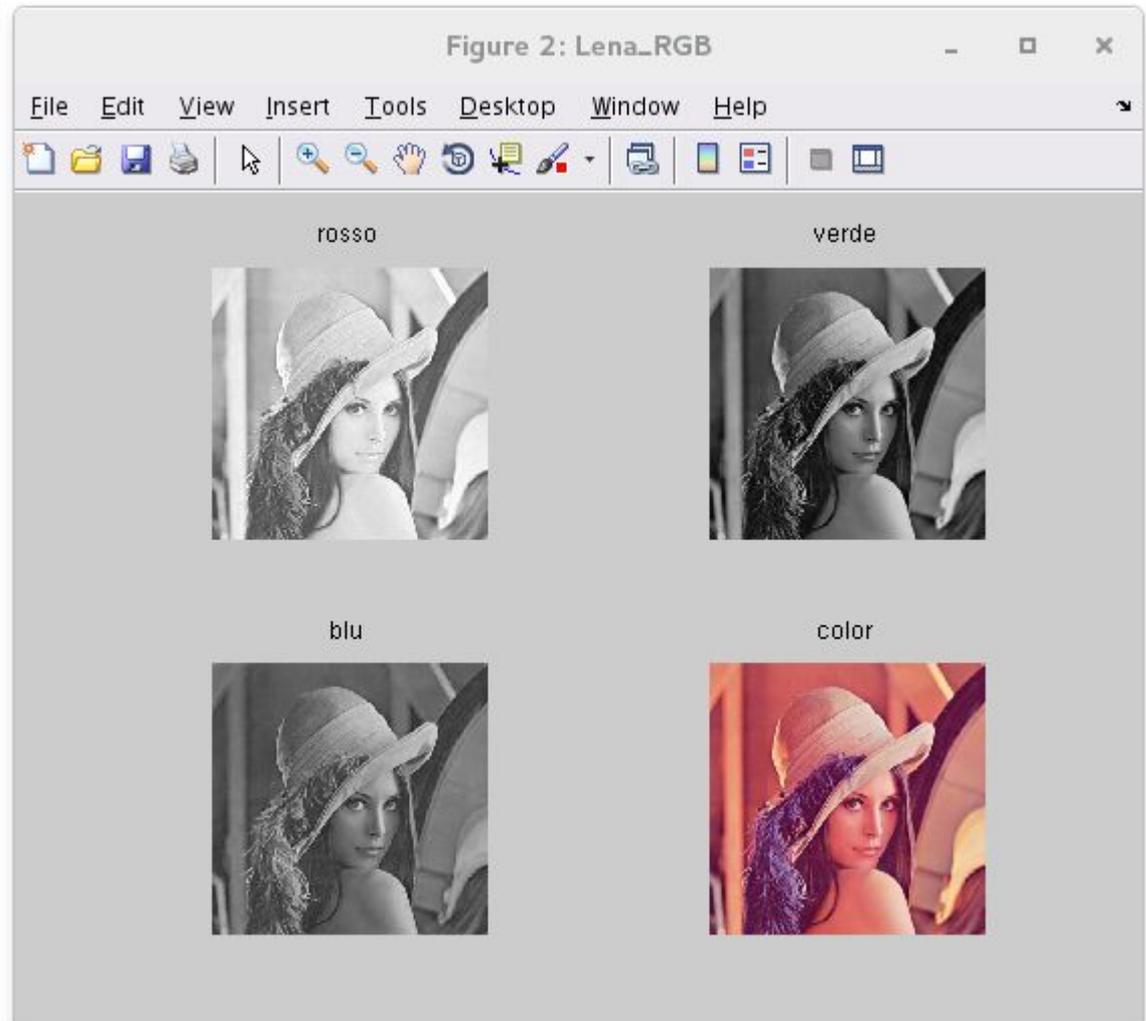
---

```
>> img_gray = rgb2gray(img);  
>> imshow(img_gray)  
>>
```



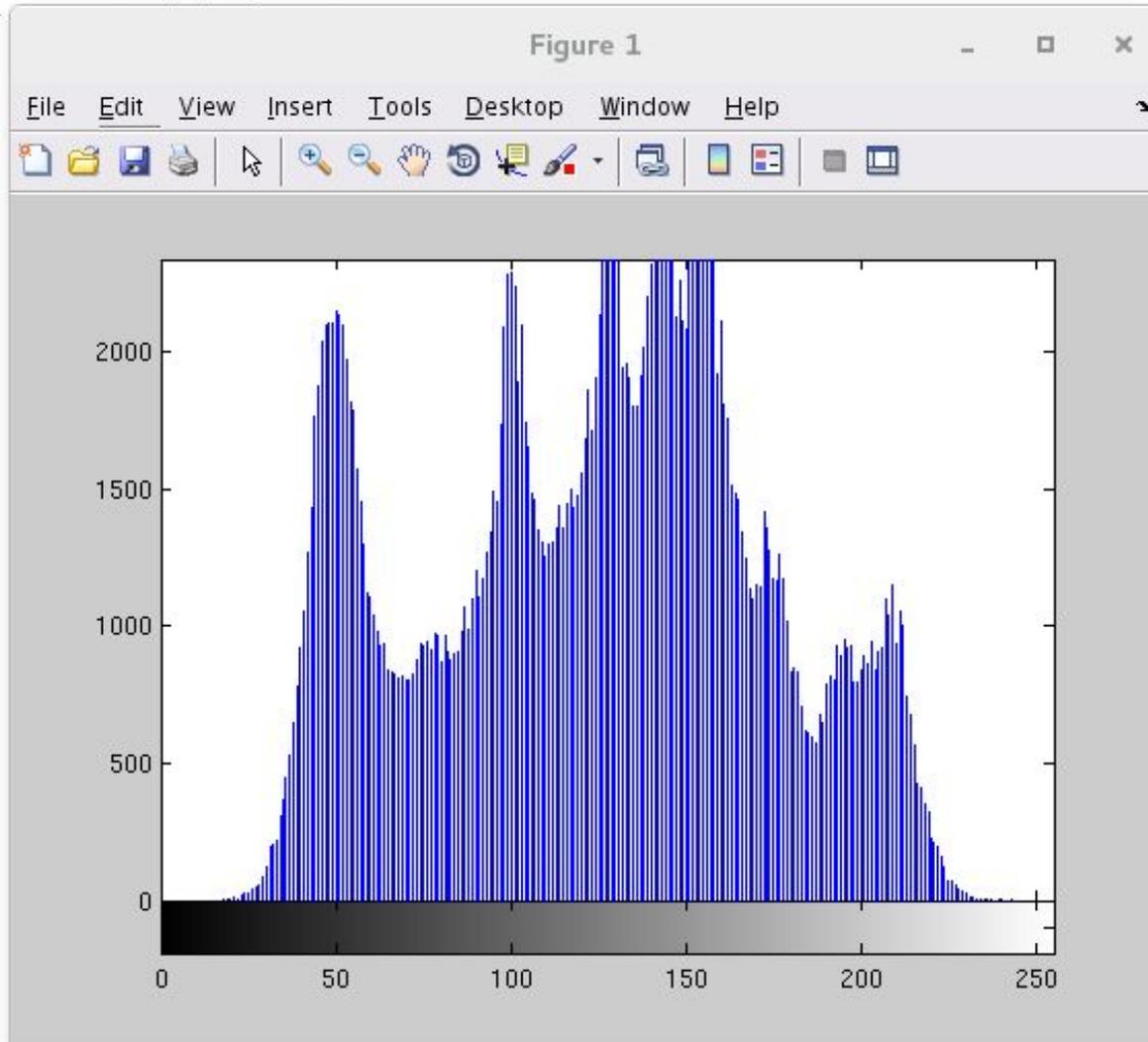
# MATLAB: le immagini

```
rosso = img(:,:,1);  
verde = img(:,:,2);  
blu = img(:,:,3);  
figure('Name', 'Lena_RGB');  
subplot(2,2,1);  
imshow(rosso);  
title('rosso')  
subplot(2,2,2);  
imshow(verde);  
title('verde');  
subplot(2,2,3);  
imshow(blu);  
title('blu')  
subplot(2,2,4);  
imshow(img);  
title('color')
```



# MATLAB: le immagini

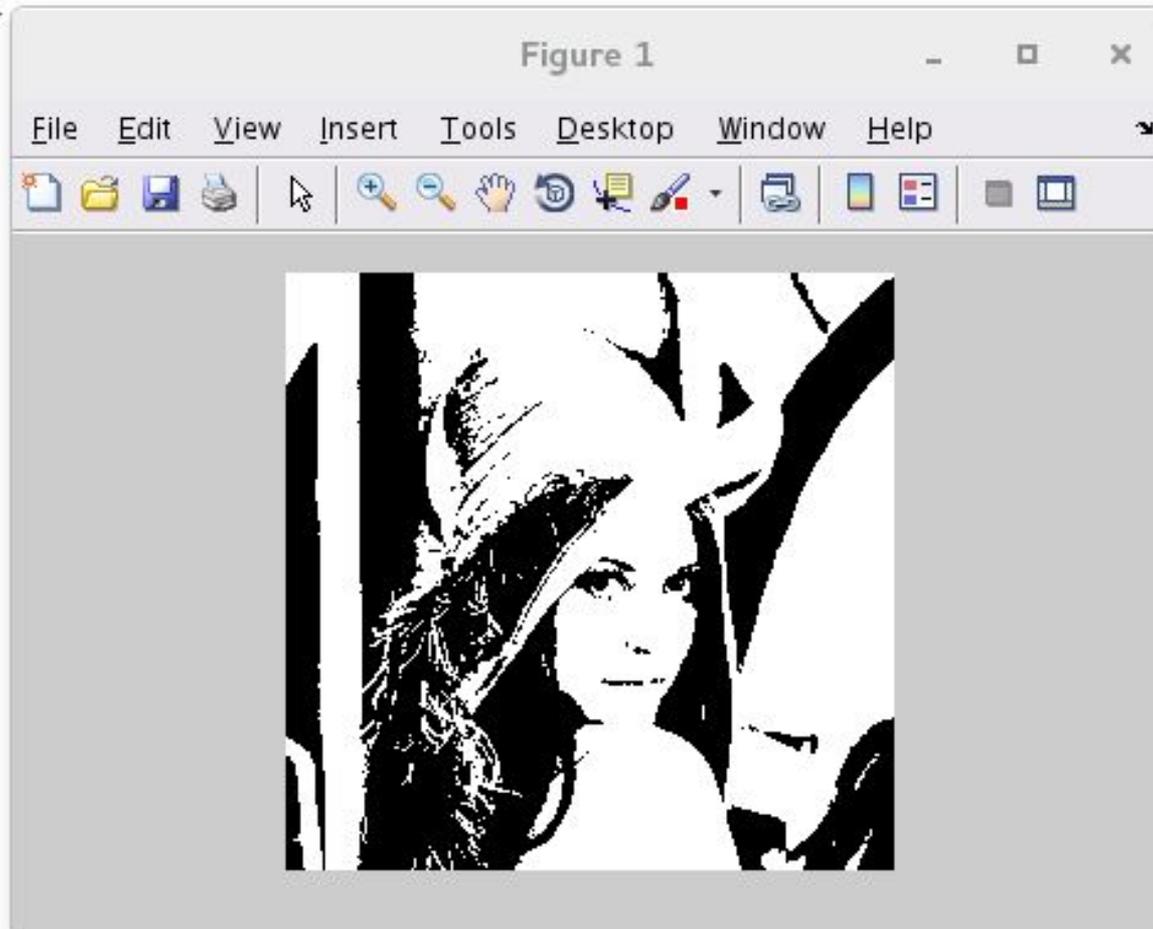
```
>> imhist(img_gray,500)  
>>
```



# MATLAB: le immagini

---

```
>> img_bw = img_gray > 112;  
>> imshow(img_bw)  
>>
```



# MATLAB: compiti per oggi

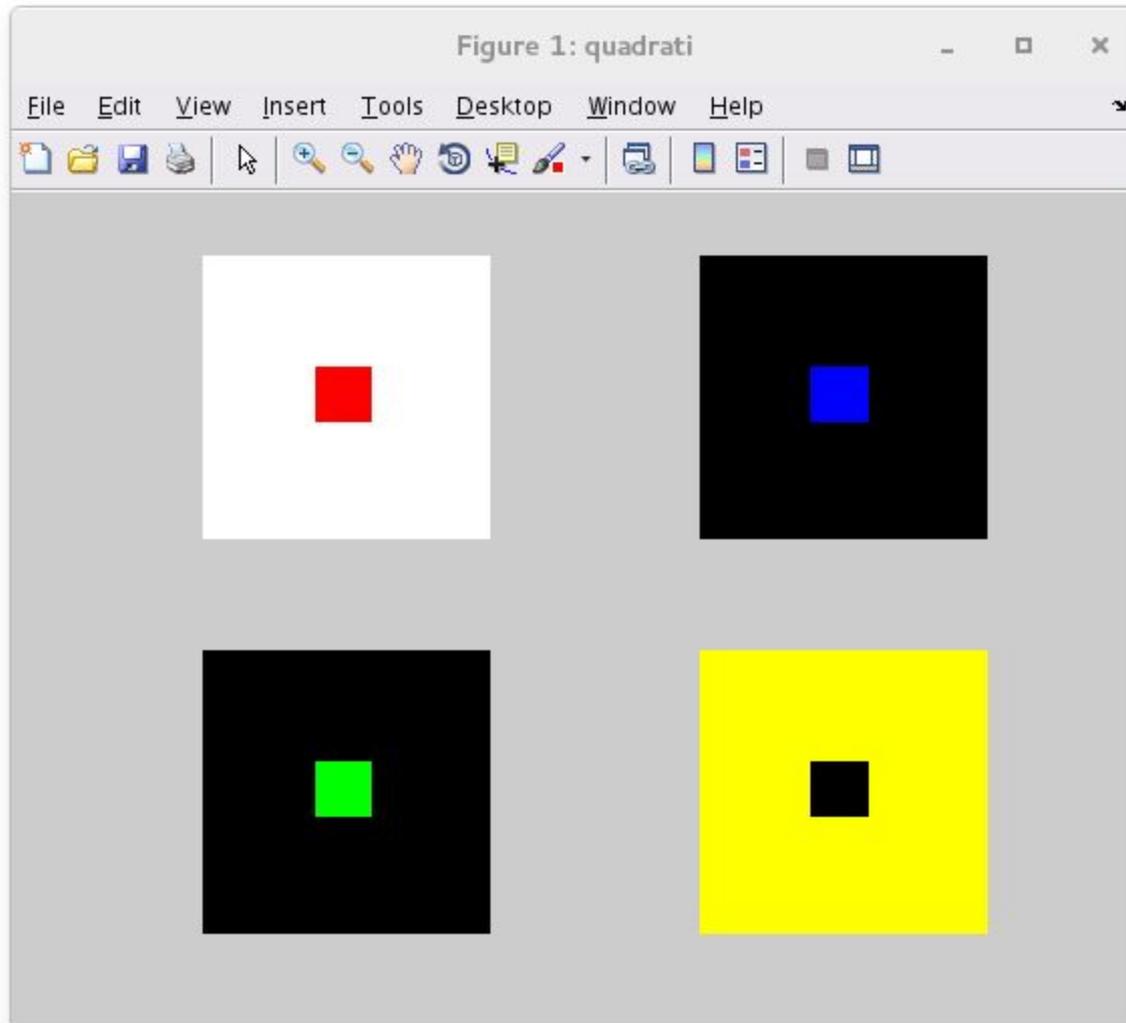
---

Disegnare in una figure 4 immagini 512X512  
rappresentanti:

- Un quadrato rosso su sfondo bianco
  - Un quadrato blu su sfondo nero
  - Un quadrato verde su sfondo nero
  - Un quadrato nero su sfondo giallo
-

# MATLAB: compiti per oggi

---



# MATLAB: compiti per oggi

---

```
rosso = ones(512,512,3) *255;  
rosso(200:300,200:300,2:3) = 0;
```

```
blu = zeros(512,512,3);  
blu(200:300,200:300,3) = 255;
```

```
verde = zeros(512,512,3);  
verde(200:300,200:300,2) = 255;
```

```
giallo = ones(512,512,3) *255;  
giallo(:, :, 3) = 0;
```

```
giallo(200:300,200:300,:) = 0;
```

```
figure('Name','quadrati');  
subplot(2,2,1);  
imshow(rosso);  
subplot(2,2,2);  
imshow(blu);  
subplot(2,2,3);  
imshow(verde);  
subplot(2,2,4);  
imshow(giallo);
```