



Introduzione a MATLAB

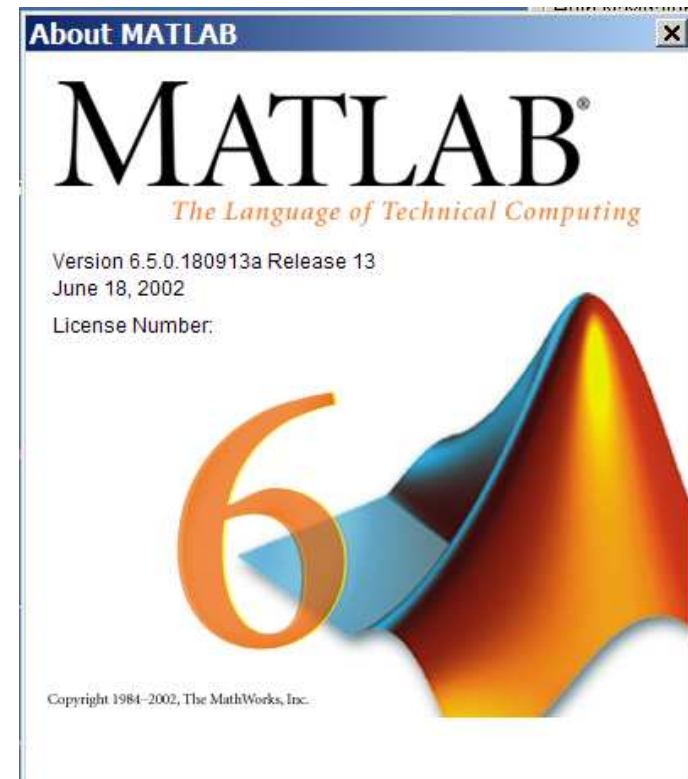
Elaborazione Numerica dei Segnali

a.a. 2008/2009

Simone Bianco

Introduzione

- Ambiente avanzato di calcolo numerico
 - Linguaggio di programmazione
 - Supporto a GUI
- Ampiamente utilizzato in
 - Ambito scientifico
 - Ambito industriale



 The MathWorks <http://www.mathworks.com/>

Introduzione

- **Caratteristiche**
 - Semplice da imparare e utilizzare
 - Notazione matematica
 - Approccio procedurale
 - Linguaggio interpretato
 - Assenza esplicita di tipi
 - Incoraggia a trovare soluzioni matriciali
 - Tipo base è la matrice
 - Operazioni base su matrici
 - Fornisce molte funzionalità
 - numeriche
 - grafiche
 - Possibilità di interfacciamento ad altri linguaggi

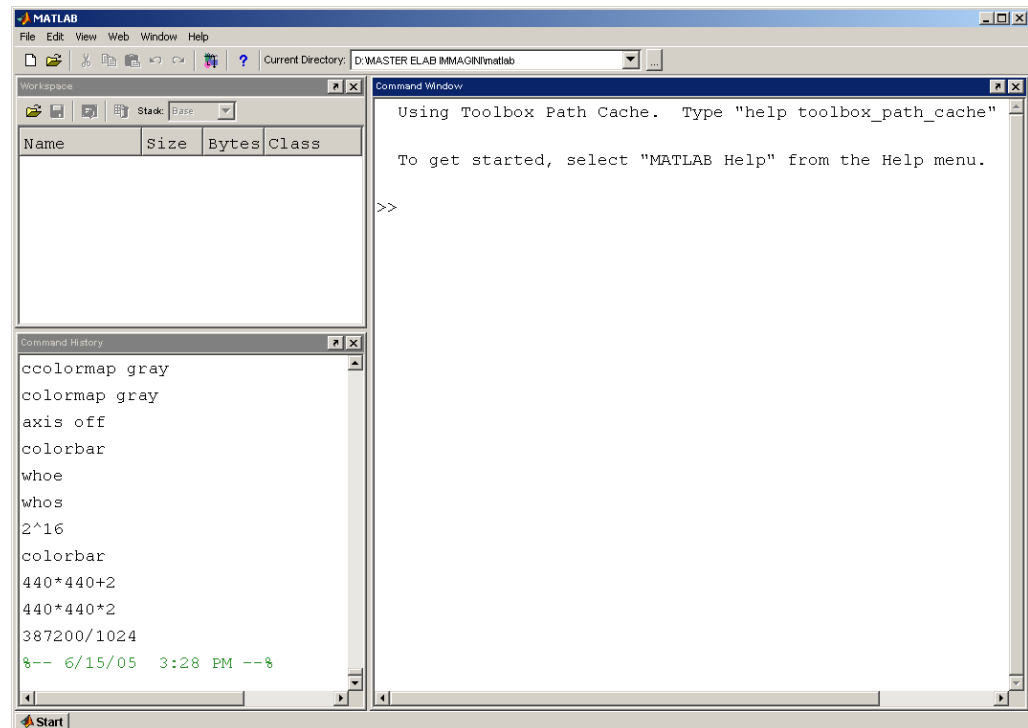
Introduzione

- MATLAB è sostanzialmente un
- PSE : Problem Solving Environment
 - Ambiente che offre funzionalità per risolvere problemi in una data area
 - Image Processing Toolbox
 - Neural Network Toolbox
 - Statistics Toolbox
 - Financial Modeling
 - Signal processing
 - ...
- RPE : Rapid Prototyping Environment
 - Sistema per sviluppare e testare idee od algoritmi rapidamente

Outline

- MATLAB
 - Introduzione
 - Operazioni
 - Visualizzazione scientifica
 - Programmazione

- Esempi



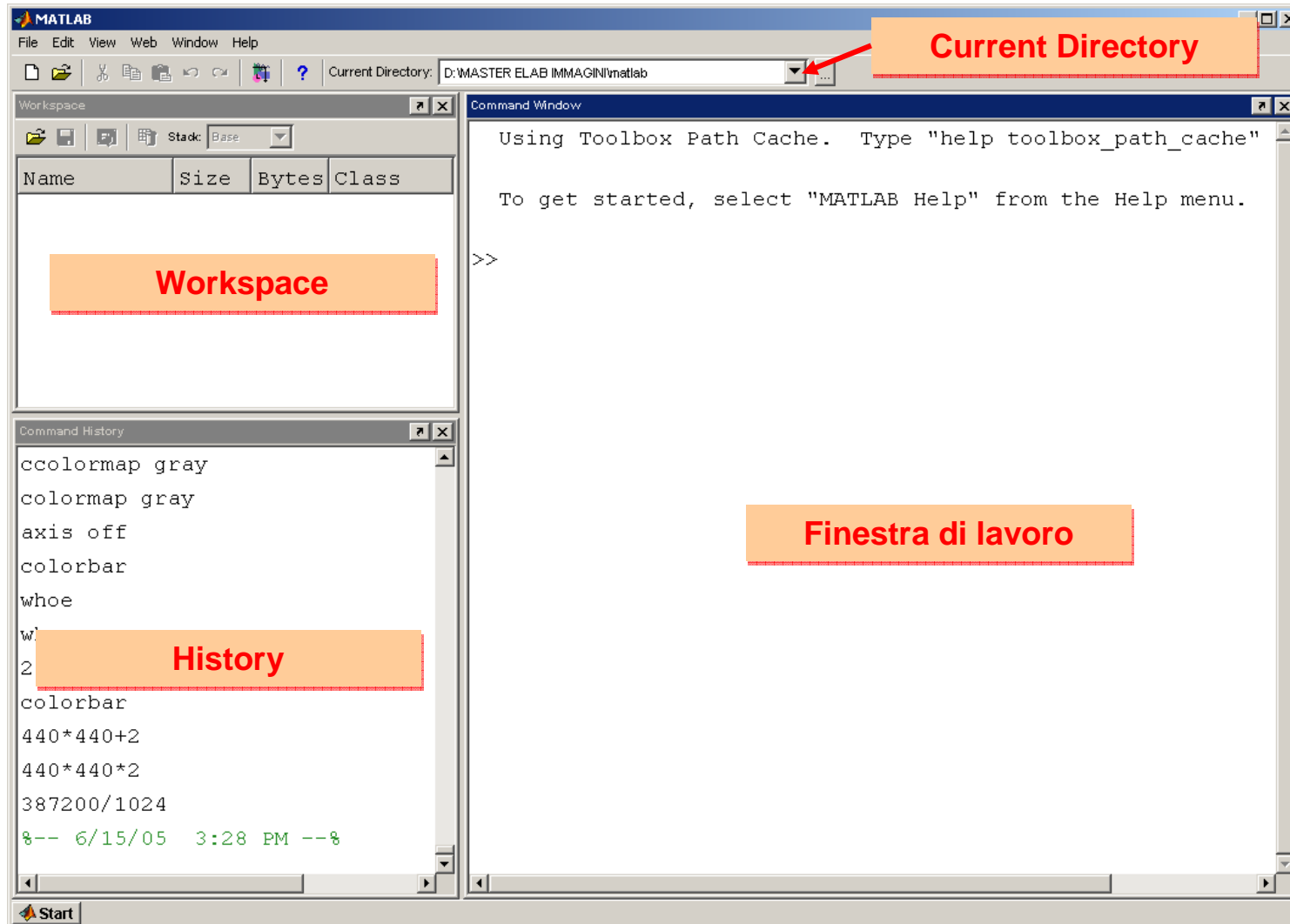
The screenshot displays the MATLAB environment. The Command Window on the right contains the following text:

```
Using Toolbox Path Cache. Type "help toolbox_path_cache"  
To get started, select "MATLAB Help" from the Help menu.  
  
>>
```

The Command History window on the left shows the following commands and their outputs:

```
ccolormap gray  
colormap gray  
axis off  
colorbar  
whoe  
whos  
2^16  
colorbar  
440*440+2  
440*440*2  
387200/1024  
%-- 6/15/05 3:28 PM --%
```

MATLAB



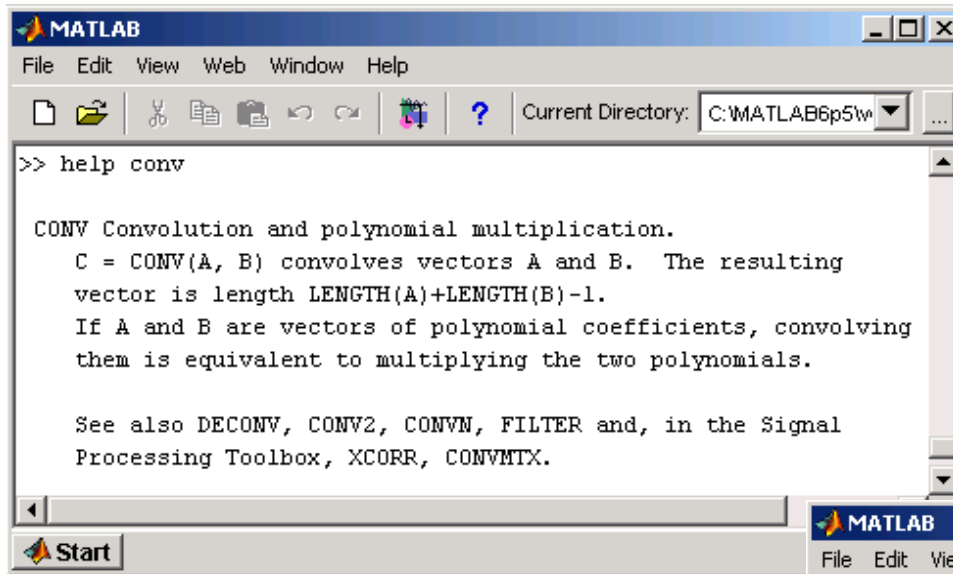
MATLAB

- Current Directory
 - Directory di lavoro
 - Eventuali file vengono cercati/eseguiti in questa cartella e nel path definito (**File->Set Path**)
- Workspace
 - Mostra informazioni relative a tutte le variabili
 - attualmente in memoria
 - che possono essere utilizzate nella finestra deio comandi
- History
 - Mostra tutti i comandi eseguiti fino a quel momento
 - Se doppio click sul comando, viene eseguito
 - Possibile copia/incolla

MATLAB

- Finestra di lavoro
 - Consente di eseguire direttamente dei comandi MATLAB
 - Interni o esterni (programmi/script MATLAB)
 - Forniti comandi generici di sistema
 - ls, dir, cd, delete, ...
 - Programmi esterni possono essere eseguiti con “!”
`!notepad`
 - Permette di richiamare l’help in linea e di fare ricerche
`help comando`
`lookfor testo`

MATLAB

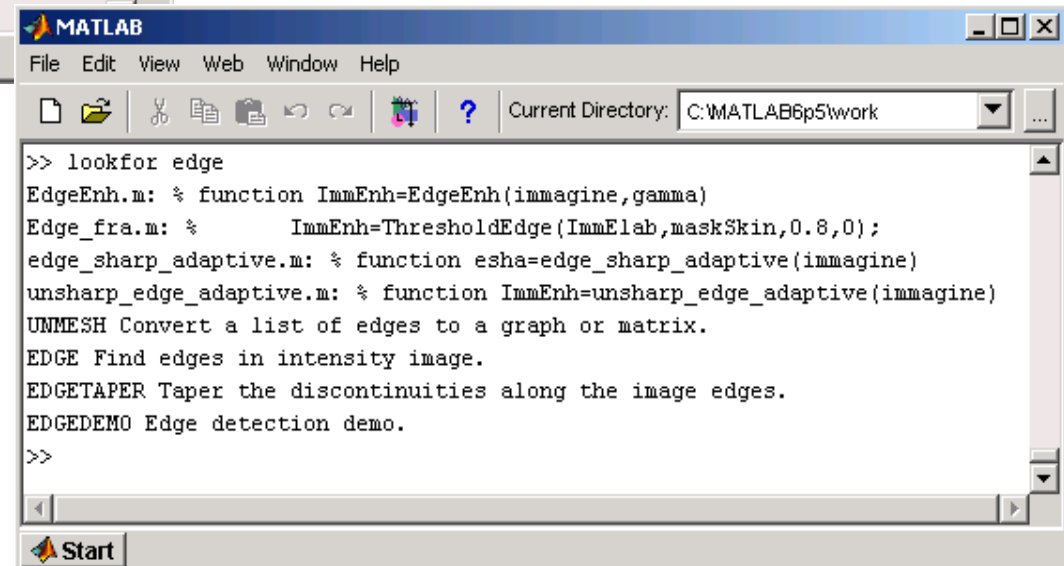


A screenshot of the MATLAB software interface. The window title is "MATLAB". The menu bar includes "File", "Edit", "View", "Web", "Window", and "Help". The toolbar contains icons for file operations and a question mark. The "Current Directory" is set to "C:\MATLAB6p5w". The command window shows the command `>> help conv` and its output:

```
>> help conv

CONV Convolution and polynomial multiplication.
  C = CONV(A, B) convolves vectors A and B.  The resulting
  vector is length LENGTH(A)+LENGTH(B)-1.
  If A and B are vectors of polynomial coefficients, convolving
  them is equivalent to multiplying the two polynomials.

  See also DECONV, CONV2, CONVN, FILTER and, in the Signal
  Processing Toolbox, XCORR, CONVMTX.
```

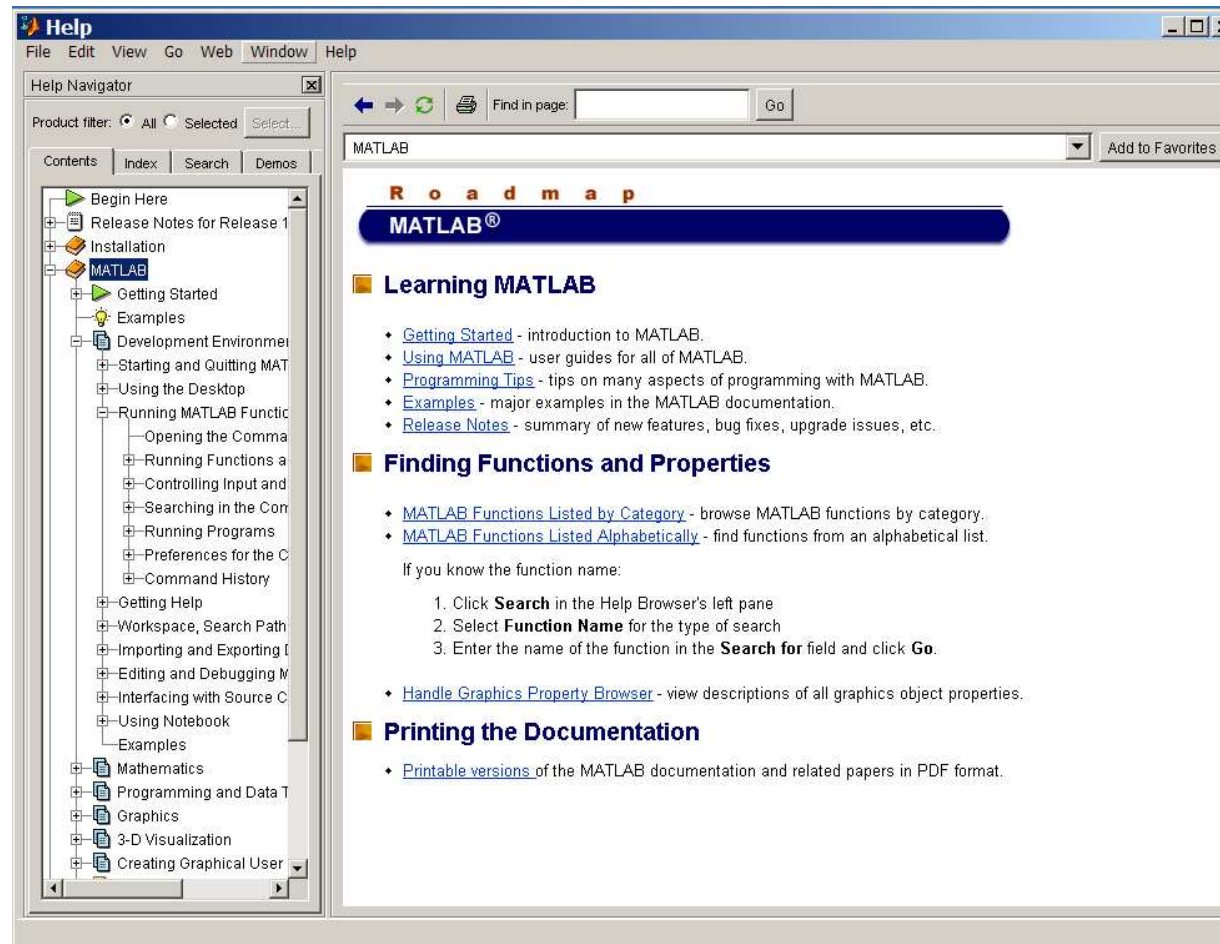


A screenshot of the MATLAB software interface. The window title is "MATLAB". The menu bar includes "File", "Edit", "View", "Web", "Window", and "Help". The toolbar contains icons for file operations and a question mark. The "Current Directory" is set to "C:\MATLAB6p5\work". The command window shows the command `>> lookfor edge` and its output:

```
>> lookfor edge
EdgeEnh.m: % function ImmEnh=EdgeEnh(immimage, gamma)
Edge_fra.m: %      ImmEnh=ThresholdEdge(ImmElab,maskSkin,0.8,0);
edge_sharp_adaptive.m: % function esha=edge_sharp_adaptive(immimage)
unsharp_edge_adaptive.m: % function ImmEnh=unsharp_edge_adaptive(immimage)
UNMESH Convert a list of edges to a graph or matrix.
EDGE Find edges in intensity image.
EDGETAPER Taper the discontinuities along the image edges.
EDGEDEMO Edge detection demo.
>>
```

MATLAB

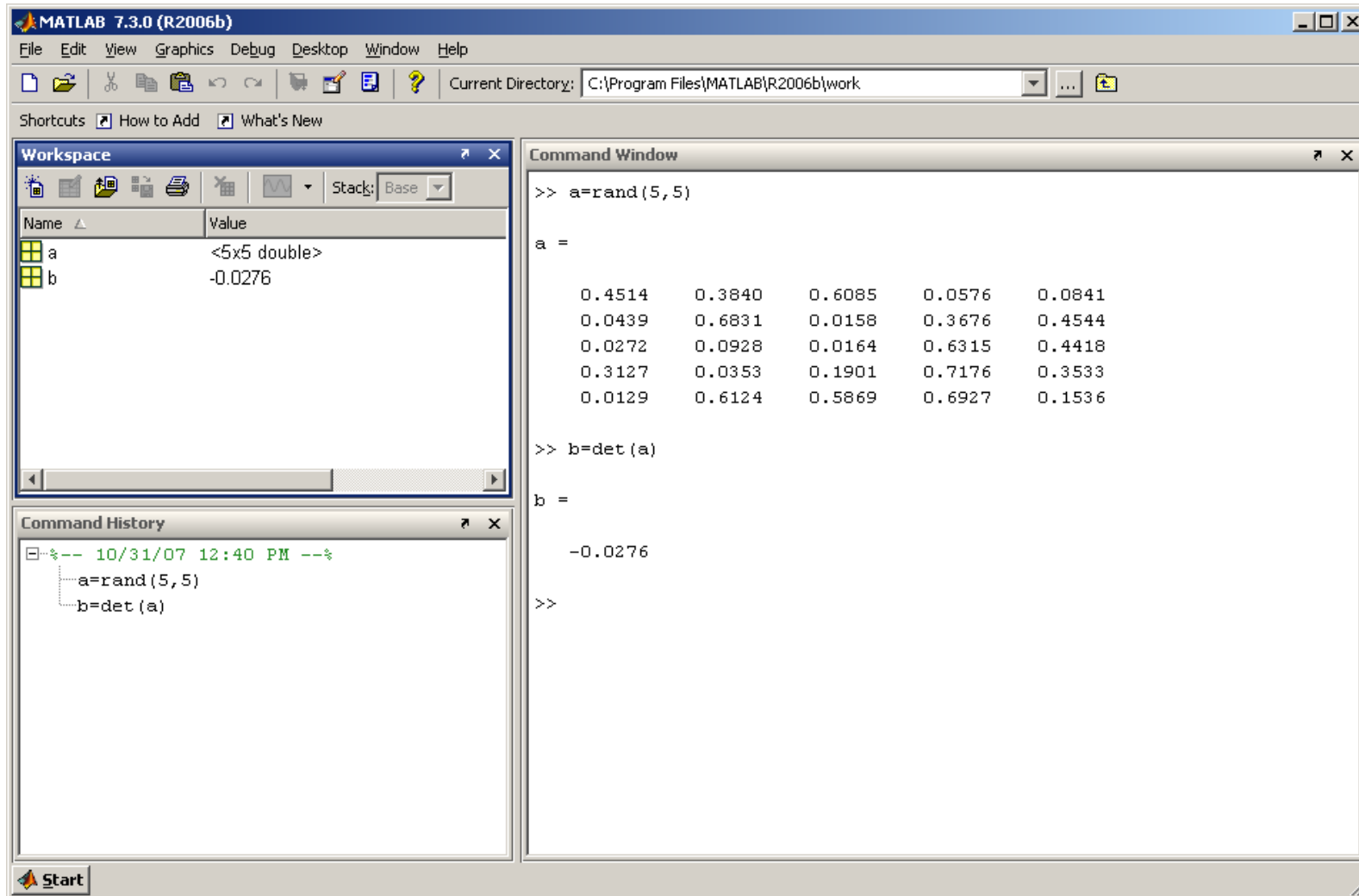
- Help→MATLAB Help



MATLAB

- Comandi MATLAB
 - Possono fare parte del linguaggio
 - Funzioni matematiche, algebriche
 - Risoluzione di sistemi di equazioni
 - Analisi di dati
 - ...
 - Possono essere funzioni (script) MATLAB esterni
 - File testuali con estensione .m
 - Contengono uno o più comandi
 - Toolbox

MATLAB



MATLAB

- Caratteri speciali

- **%** commento (da usarsi nei file .m)
- **...** continuazione sulla riga successiva (da usarsi nei file .m)
- **~=** operatore disuguaglianza
- **==** operatore di uguaglianza
- **;** impedisce l'echo del comando
- **,** separa argomenti o comandi

- Alcuni comandi

- **clc** cancella il contenuto della finestra dei comandi
- **clear n** cancella la variabile n dal workspace
- **clear** cancella tutto il contenuto del workspace
- **close all** chiude tutte le finestre secondarie di MATLAB aperte

MATLAB

The screenshot displays the MATLAB 7.3.0 (R2006b) environment. The **Workspace** window shows variables `a`, `ans`, `b`, and `c`. The **Command Window** shows the execution of `b=det(a)` and `c=inv(a), det(c)`, with the output for `ans` being `-36.2436`. The **Command History** window shows the sequence of commands: `a=rand(5,5)`, `b=det(a)`, `b=det(a);`, and `c=inv(a), det(c)`. A red circle highlights the `ans =` output in the Command Window, with the text `variabile di lavoro` next to it.

Workspace

Name	Value
a	<5x5 double>
ans	-36.2436
b	-0.0276
c	<5x5 double>

Command Window

```
>> b=det(a)
b =
-0.0276
>> b=det(a);
>> c=inv(a), det(c)
c =
-0.1872    1.2665   -3.8779    3.6655   -0.9189
-0.7682    1.7818   -2.7112    1.0751    0.4761
 2.0870   -1.9861    3.9529   -3.0979    0.4864
-1.6205    0.2460   -2.2387    2.4610    0.9390
 2.4117   -0.7304    6.1248   -3.8540   -1.4035
ans =
-36.2436
>>
>>
```

Command History

```
10/31/07 12:40 PM --%
a=rand(5,5)
b=det(a)
b=det(a);
c=inv(a), det(c)
```

MATLAB

- MATLAB non usa dichiarazione di variabili o dichiarazione di dimensioni
 - Variabili automaticamente create all'uso
- I nomi delle variabili sono case sensitive
- Non esistono dichiarazione di tipi di dati
 - Le variabili assumono automaticamente il tipo corretto
 - Il dato fondamentale è la matrice rettangolare (di double)
 - Variabile semplice = matrice 1x1
 - Vettore riga = matrice 1xN
 - Vettore colonna = matrice Nx1

MATLAB: Operazioni

- Creare Matrici

- `ones(r,c)`
- `zeros(r,c)`
- `eye(r,c)`

– `a=[a11 a12 ; a21 a22]`

a=

a11	a12
a21	a22

Riga
successiva

```
Command Window
File Edit View Web Window Help
>> m=zeros(4,4)
m =
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
>> m=ones(4,4)
m =
     1     1     1     1
     1     1     1     1
     1     1     1     1
     1     1     1     1
>> m=eye(4,4)
m =
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
>> a=[1,2;3,4]
a =
     1     2
     3     4
>> |
```

MATLAB: Operazioni

- Matrici

- Gli indici degli elementi partono da 1 !!!

- `m(2,3)`

Terzo elemento della seconda riga

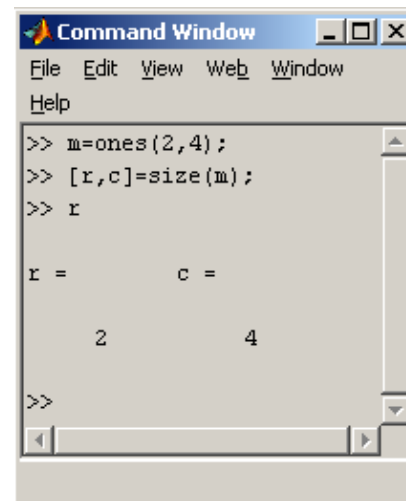
- `sz=size(v)`

sz è un vettore 2D con le dimensioni di v

- `[r,c]=size(v)`

'r' e 'c' conterranno separatamente le dimensioni di v

Notazione per gestire più valori di ritorno di una funzione



```
Command Window
File Edit View Web Window
Help
>> m=ones(2,4);
>> [r,c]=size(m);
>> r

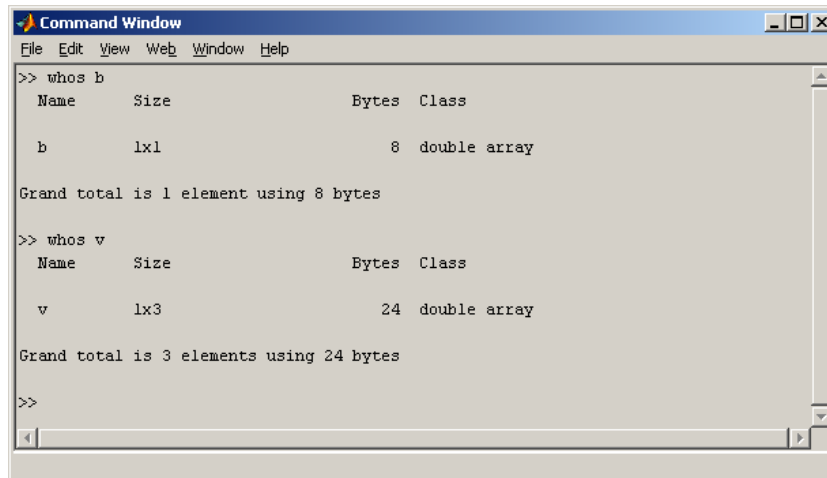
r =          c =

     2         4

>>
```

MATLAB: Operazioni

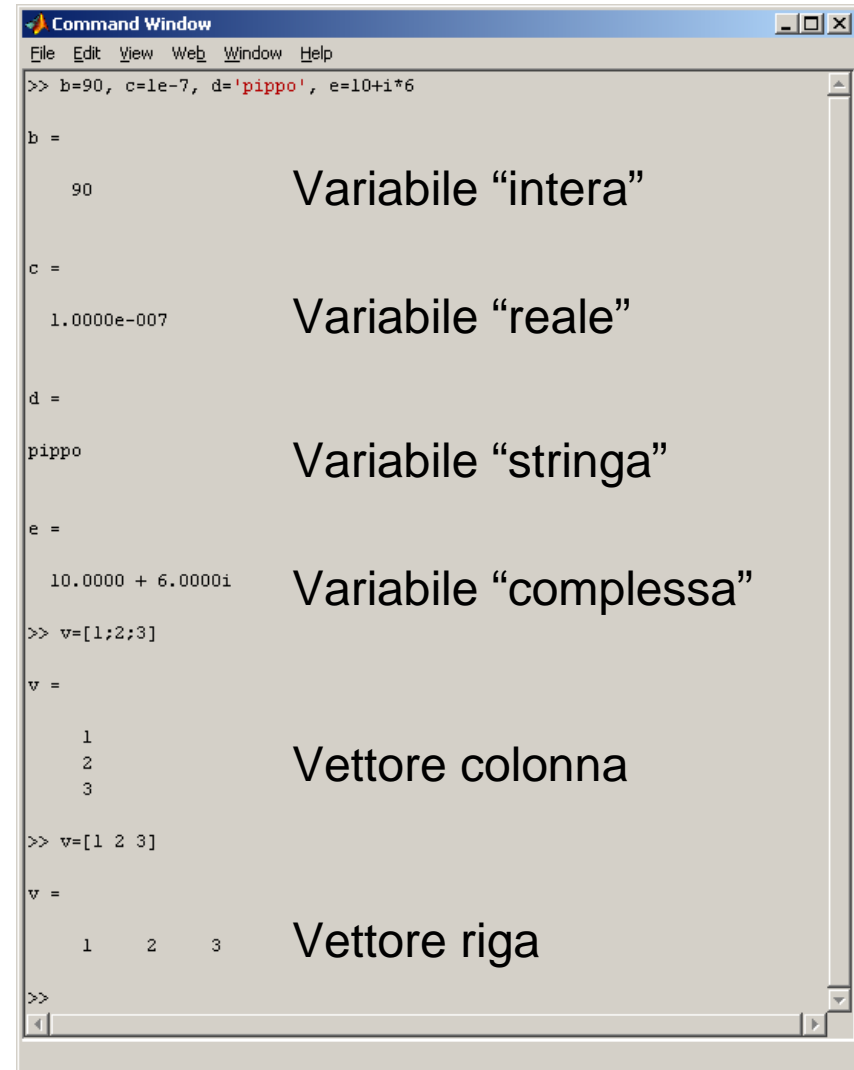
- Creare Vettori e Variabili
 - Basta assegnarle
 - Il tipo concreto della variabile si può conoscere con il comando
 - `whos nomevar`



```
Command Window
File Edit View Web Window Help
>> whos b
  Name      Size      Bytes  Class
  b         1x1         8   double array
Grand total is 1 element using 8 bytes

>> whos v
  Name      Size      Bytes  Class
  v         1x3        24   double array
Grand total is 3 elements using 24 bytes

>>
```



```
Command Window
File Edit View Web Window Help
>> b=90, c=1e-7, d='pippo', e=10+i*6

b =
    90
Variable "intera"

c =
 1.0000e-007
Variable "reale"

d =
pippo
Variable "stringa"

e =
 10.0000 + 6.0000i
Variable "complessa"

>> v=[1;2;3]

v =
     1
     2
     3
Vettore colonna

>> v=[1 2 3]

v =
     1     2     3
Vettore riga

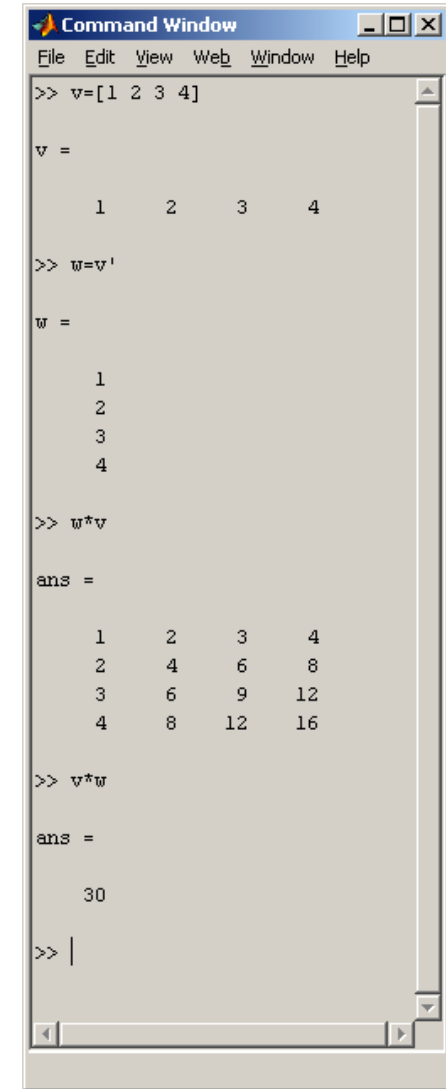
>>
```

MATLAB: Operazioni

- Operazioni su matrici
 - $A \pm B$ Somma / differenza di matrici
 - $A * B$ Prodotto di matrici
 - A' Trasposta di A
 - $A .* B$ Esegue l'operazione * punto a punto

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} .* \begin{bmatrix} 7 & 6 \\ 2 & 4 \end{bmatrix} = \begin{bmatrix} 7 & 12 \\ 6 & 16 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} .^ \wedge \begin{bmatrix} 7 & 6 \\ 2 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 64 \\ 9 & 256 \end{bmatrix}$$



```
Command Window
File Edit View Web Window Help
>> v=[1 2 3 4]
v =
     1     2     3     4
>> w=v'
w =
     1
     2
     3
     4
>> w*v
ans =
     1     2     3     4
     2     4     6     8
     3     6     9    12
     4     8    12    16
>> v*w
ans =
    30
```

MATLAB: Operazioni

- Operazioni su matrici: alcune funzioni
 - `det(A)` Determinante
 - `inv(A)` Inversa
 - `sum(A)` Somma delle colonne (risultato è un vettore)
(`max`, `min`, `mean`, `std`, ...)
 - `max(max(A))` Somma degli elementi di A
 - `magic(N)` Genera un quadrato magico NxN
 - ...

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

MATLAB: Operazioni

- Operazioni su matrici: accesso a righe e colonne

M=

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

V=

1	8	14
---	---	----

$$M(1,:) = \begin{bmatrix} 16 & 2 & 3 & 13 \end{bmatrix}$$

“Tutto”

$$M(2:3,:) = \begin{bmatrix} 5 & 11 & 10 & 8 \\ 9 & 7 & 6 & 12 \end{bmatrix}$$

da - a

$$M(2:3,1:2) = \begin{bmatrix} 5 & 11 \\ 9 & 7 \end{bmatrix}$$

$$M(V) = \begin{bmatrix} 16 & 14 & 8 \end{bmatrix}$$

V è usato come serie di
indici nella matrice

$$M(:,4) = \begin{bmatrix} 13 \\ 8 \\ 12 \\ 1 \end{bmatrix}$$

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

MATLAB: Operazioni

- Operazioni su matrici: eliminazione di righe o colonne

M=

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

$M(2:3,:)=[]$

16	2	3	13
4	14	15	1

$M(:,3)=[]$

16	2	13
5	11	8
9	7	12
4	14	1

- Svuotare (non rimuovere) una variabile: `var=[]`

MATLAB: Operazioni

- Operazioni su matrici: esempi

`M=ceil(rand(4)*100)`

M=

47	61	88	100
57	6	2	79
80	42	77	44
6	31	98	50

`M(find(M>=50))=0`

M=

47	0	0	0
0	6	2	0
0	42	0	44
6	31	0	0

`N=M+10*(not(M==0))+5.*(M==0)`

N=

57	5	5	5
5	16	12	5
5	52	5	54
16	41	5	5

0	1	1	1
1	0	0	1
1	0	1	0
0	0	1	1

MATLAB: Operazioni

- Operazioni su matrici: composizione

 $A =$

1	2
3	4

 $B =$

10	20
30	40

$M = [A \ B] =$

1	2	10	20
3	4	30	40

$M = [A; B] =$

1	2
3	4
10	20
30	40

$M = [A(1, :); [100 \ 200]; B(2, :)] =$

1	2
100	200
30	40

MATLAB: Operazioni

- Serie di valori
 - Si usa l'operatore ":"

`a=1:10`

Passo 1 implicito

a=	1	2	3	4	5	6	7	8	9	10
----	---	---	---	---	---	---	---	---	---	----

`a=-3:2:3`

a=	-3	-1	1	3
----	----	----	---	---

`a=0:pi/4:pi`

a=	0	0.7854	1.5708	2.3562	3.1416
----	---	--------	--------	--------	--------

- Fondamentale per creare serie del tipo $y(i)=f(x(i))$

MATLAB: Operazioni

- Serie di valori
 - Si usa l'operatore ":"

`x=0:pi/4:pi`

x=	0	0.7854	1.5708	2.3562	3.1416
----	---	--------	--------	--------	--------

`y=sin(x)`

y=	0	0.7071	1.0000	0.7071	0
----	---	--------	--------	--------	---

`y=exp(x)`

y=	1.0000	2.1923	4.8105	10.5507	23.1407
----	--------	--------	--------	---------	---------

`y=exp(-(pi/2-x).^2)`

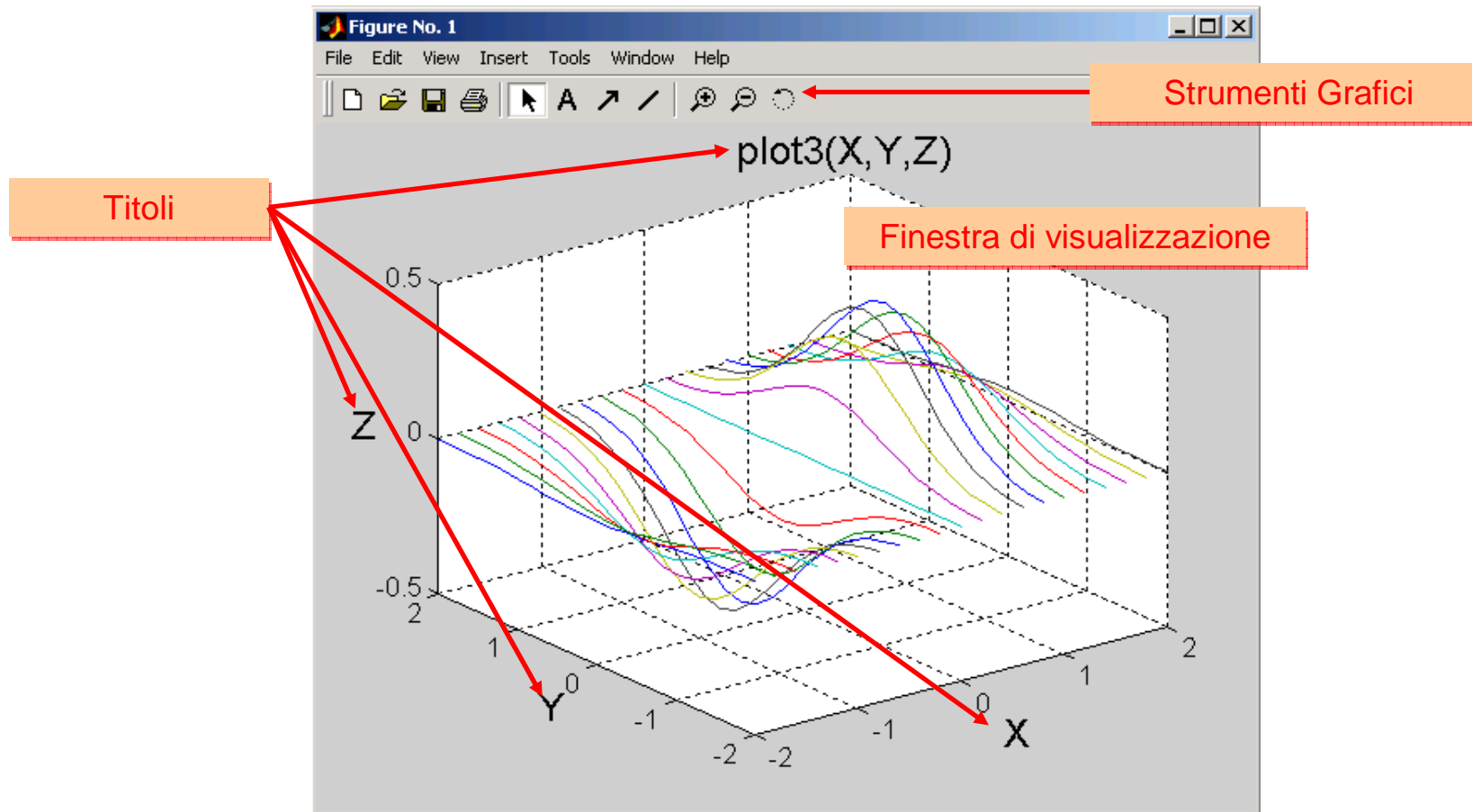
y=	0.0848	0.5396	1.0000	0.5396	0.0848
----	--------	--------	--------	--------	--------

MATLAB: Visualizzazione

- Uno dei punti di forza di MATLAB
- Diverse modalità di visualizzazione dei dati
 - 2D, 3D, Movie,...
- Utili per
 - Trovare modelli
 - Identificare tendenze
 - Comparare informazioni complesse
 - Esaminare dati in modo più “visibile”
- Diverse funzionalità per manipolare gli oggetti grafici

MATLAB: Visualizzazione

- Finestra di visualizzazione grafica



MATLAB: Visualizzazione

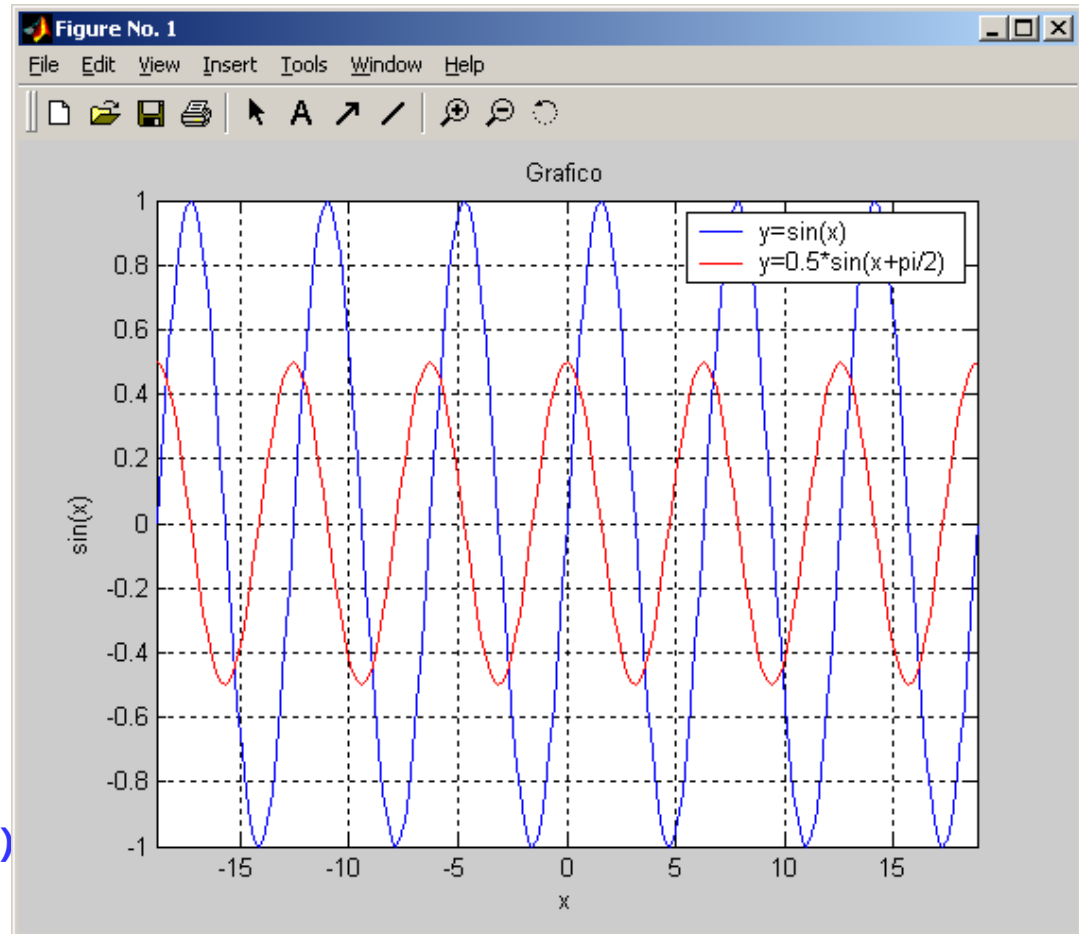
- Le finestre grafiche vengono create
 - Automaticamente da funzioni specifiche
 - `plot` disegna una serie di dati in 2D
 - `plot3` disegna una serie di dati in 3D
 - `figure` apre una finestra grafica vuota e la attiva
 - ...
 - `figure`
 - `figure(n)` rende corrente la finestra grafica numero n
 - `close(n)` chiude la finestra grafica numero n
 - `hold on` continua a disegnare sulla finestra grafica corrente

MATLAB: Visualizzazione

```
x=-6*pi:pi/16:6*pi
y=sin(x)
plot(x,y)

title('Grafico')
xlabel('x')
ylabel('sin(x)')
legend('y=sin(x)')
axis([-6*pi 6*pi -1 1])

hold on
y2=0.5*sin(x+pi/2)
plot(x,y2,'r')
legend('y=sin(x)',...
       'y=0.5*sin(x+pi/2)')
grid on
```



MATLAB: Visualizzazione

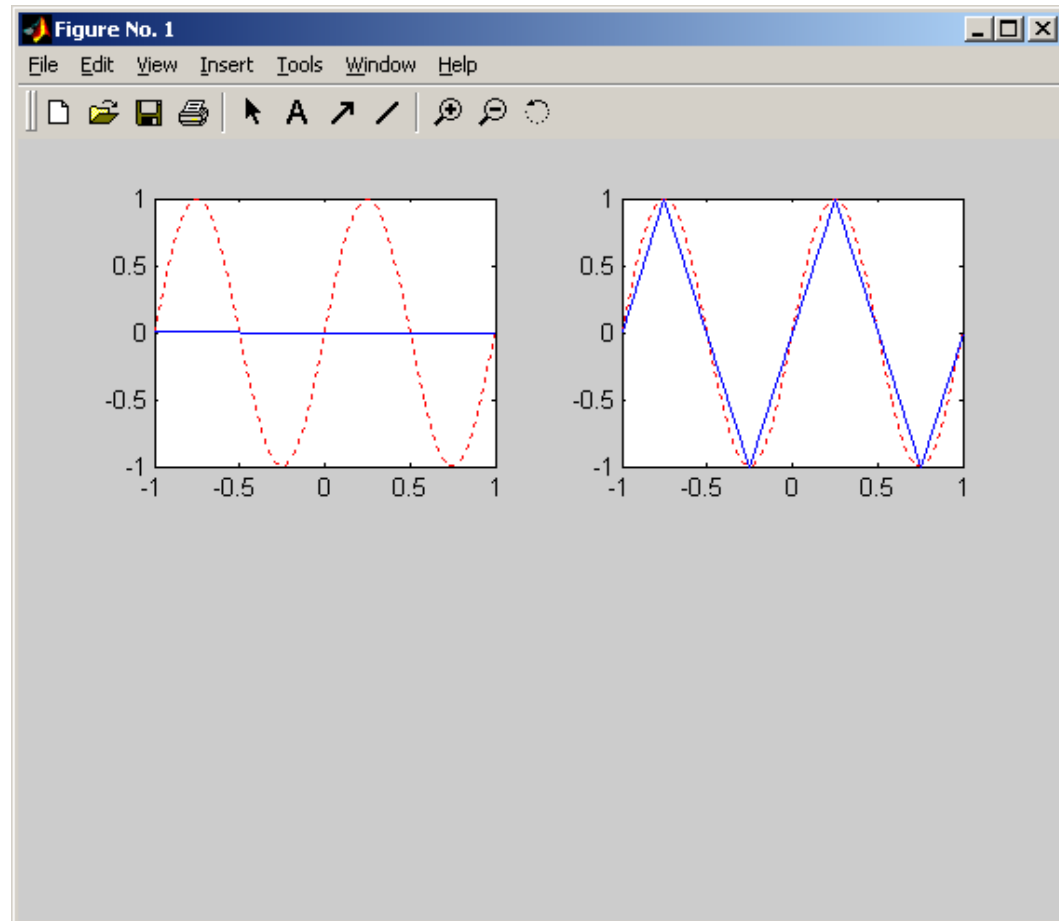
```
xr=linspace(-1,1,360)  
yr=sin(2*pi*xr)
```

```
x=linspace(-1,1,3)  
y=sin(2*pi*x)
```

```
subplot(2,2,1)  
plot(xr,yr,'r:',x,y,'b')
```

```
x=linspace(-1,1,9)  
y=sin(2*pi*x)
```

```
subplot(2,2,2)  
plot(xr,yr,'r:',x,y,'b')
```



MATLAB: Visualizzazione

```
x=linspace(-1,1,25)
```

```
y=sin(2*pi*x)
```

```
subplot(2,2,3)
```

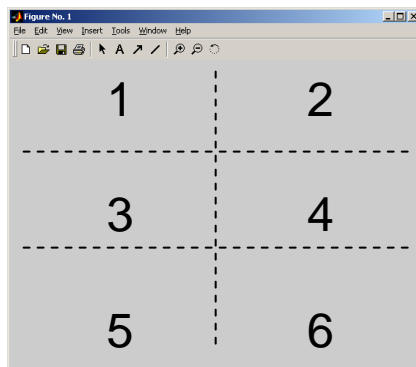
```
plot(xr,yr,'r:',x,y,'b')
```

```
x=linspace(-1,1,100)
```

```
y=sin(2*pi*x)
```

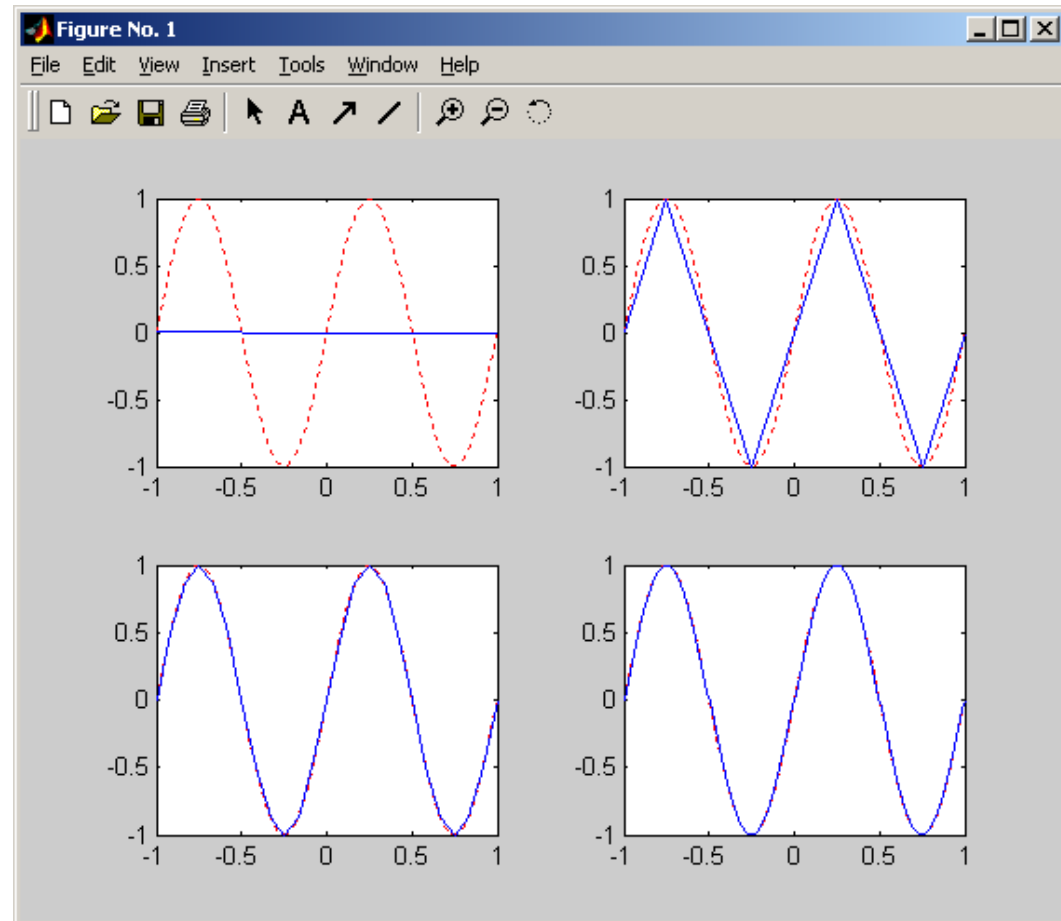
```
subplot(2,2,4)
```

```
plot(xr,yr,'r:',x,y,'b')
```



```
subplot(3,2,5) =
```

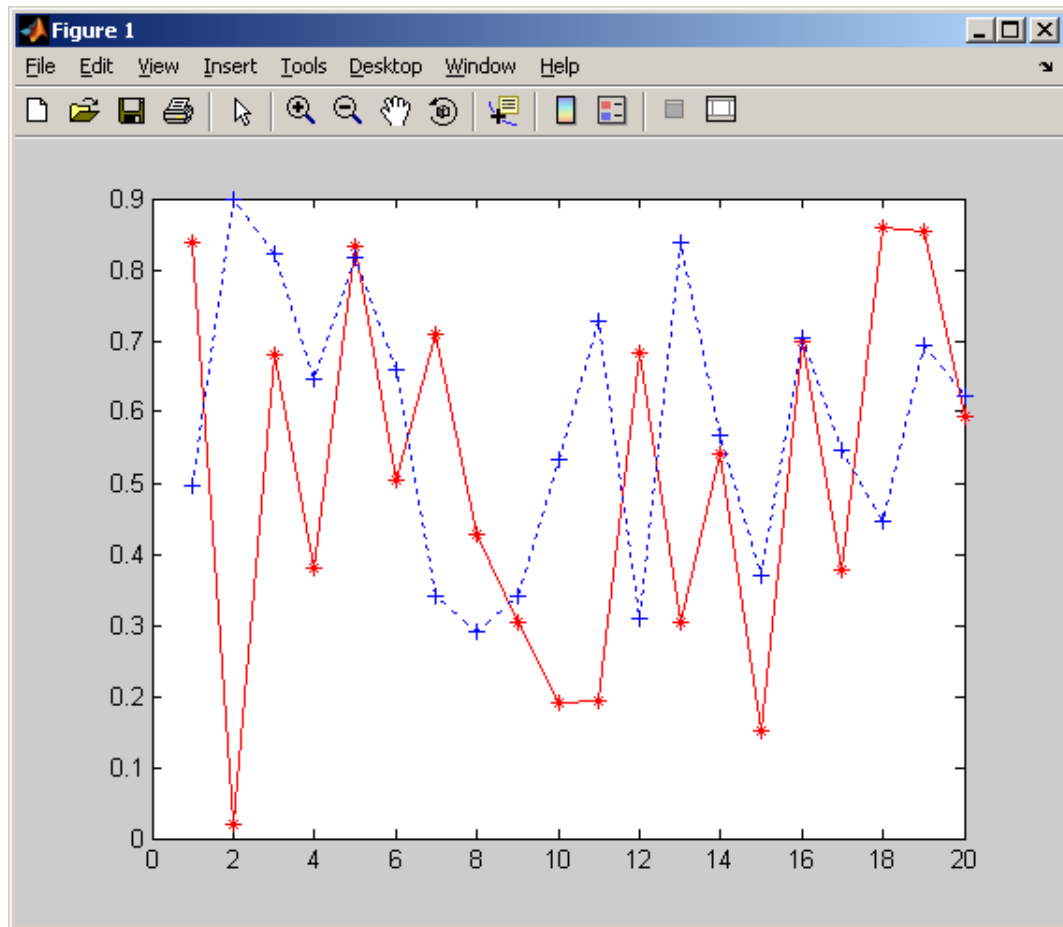
Attivo quinto settore della finestra suddivisa in 3 righe e 2 colonne



MATLAB: Visualizzazione

- Modificare simboli dei grafici

```
x=1:20  
y1=rand(1,20)  
y2=rand(1,20)  
  
plot(x,y1,'*r-')  
hold on  
plot(x,y2,'+b:')
```



MATLAB: Visualizzazione

- Grafici a barre

```
y=rand(5,3)
```

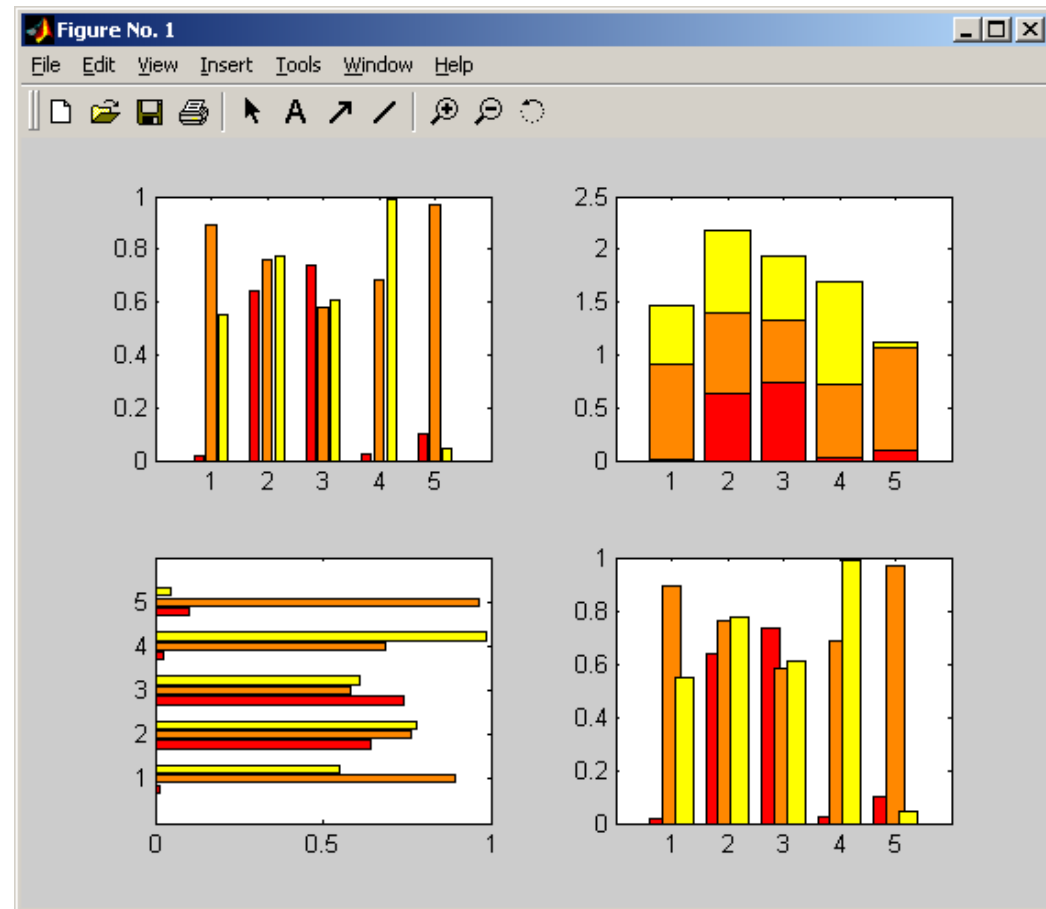
```
subplot(2,2,1)  
bar(y,'grouped')
```

```
subplot(2,2,2)  
bar(y,'stacked')
```

```
subplot(2,2,3)  
barh(y)
```

```
subplot(2,2,4)  
bar(y,1.5)
```

```
colormap autumn
```



MATLAB: Visualizzazione

- Il plot di una matrice viene fatto per colonna

```
x=linspace(-1,1,100)
```

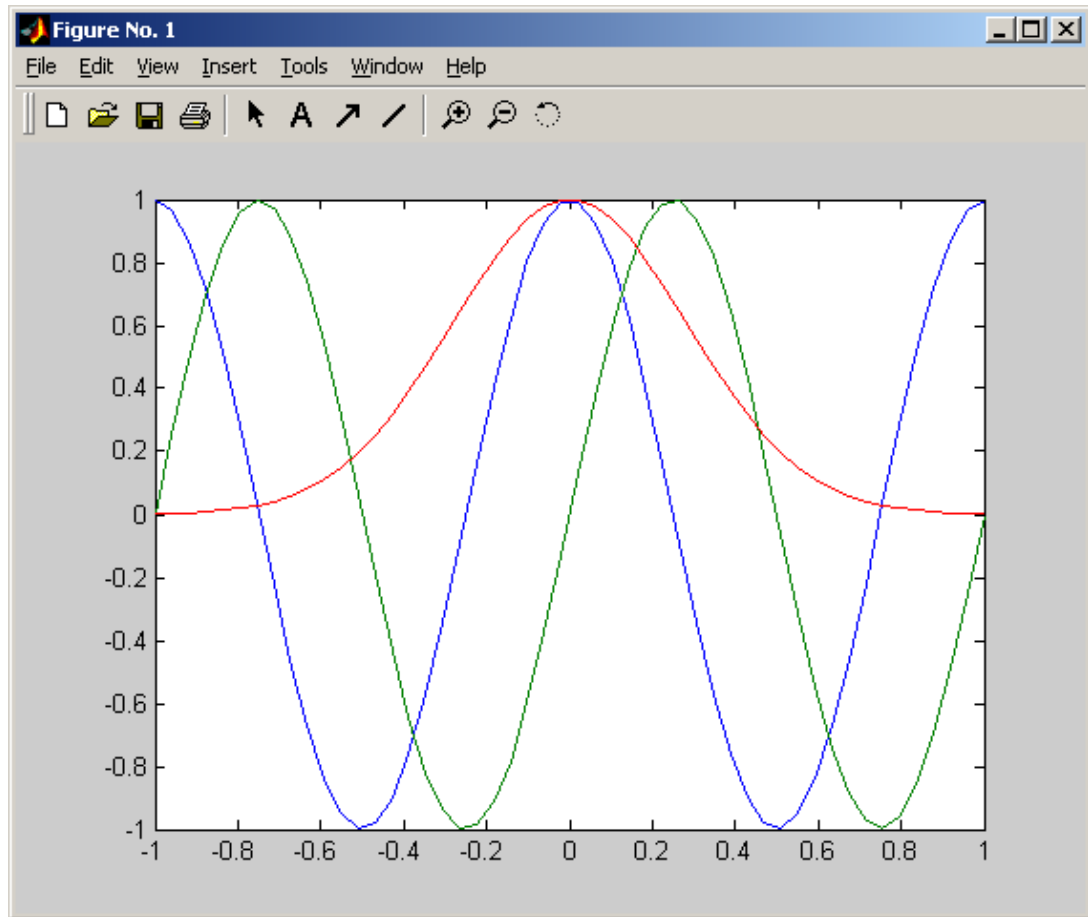
```
s=sin(2*pi*x)
```

```
c=cos(2*pi*x)
```

```
g=exp(-(x./0.3).^2)
```

```
M=[s' c' g']
```

```
plot(x,M);
```



MATLAB: Visualizzazione

- Disegno di una traiettoria in 3D

```
t=0:pi/50:20*pi
```

```
plot3(sin(t),cos(t),'r')
```

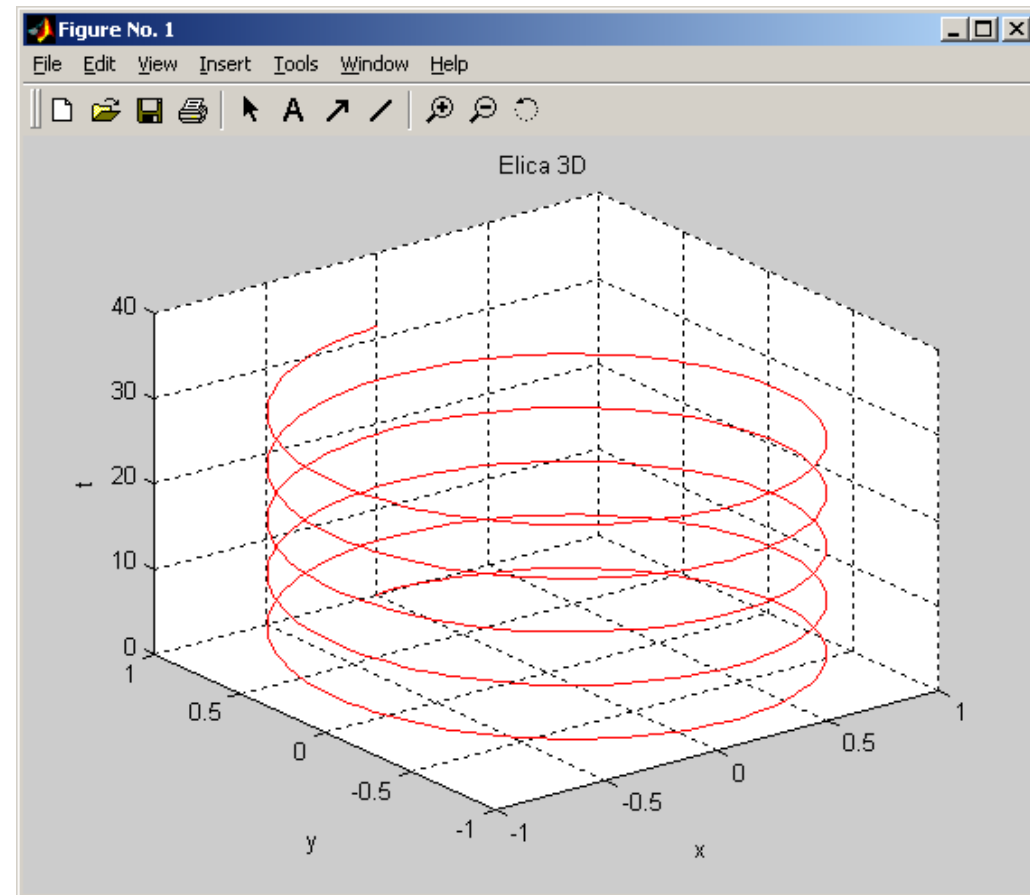
```
grid on
```

```
title('Elica 3D')
```

```
xlabel('x')
```

```
ylabel('y')
```

```
zlabel('t')
```



MATLAB: Visualizzazione

- Se si plottano in 3D matrici, si ottengono le linee corrispondenti alle colonne delle matrici

```
[X Y]=meshgrid(-2:0.2:2)
```

```
Z=X.*exp(-X.^2-Y.^2)
```

```
plot3(X,Y,Z)
```

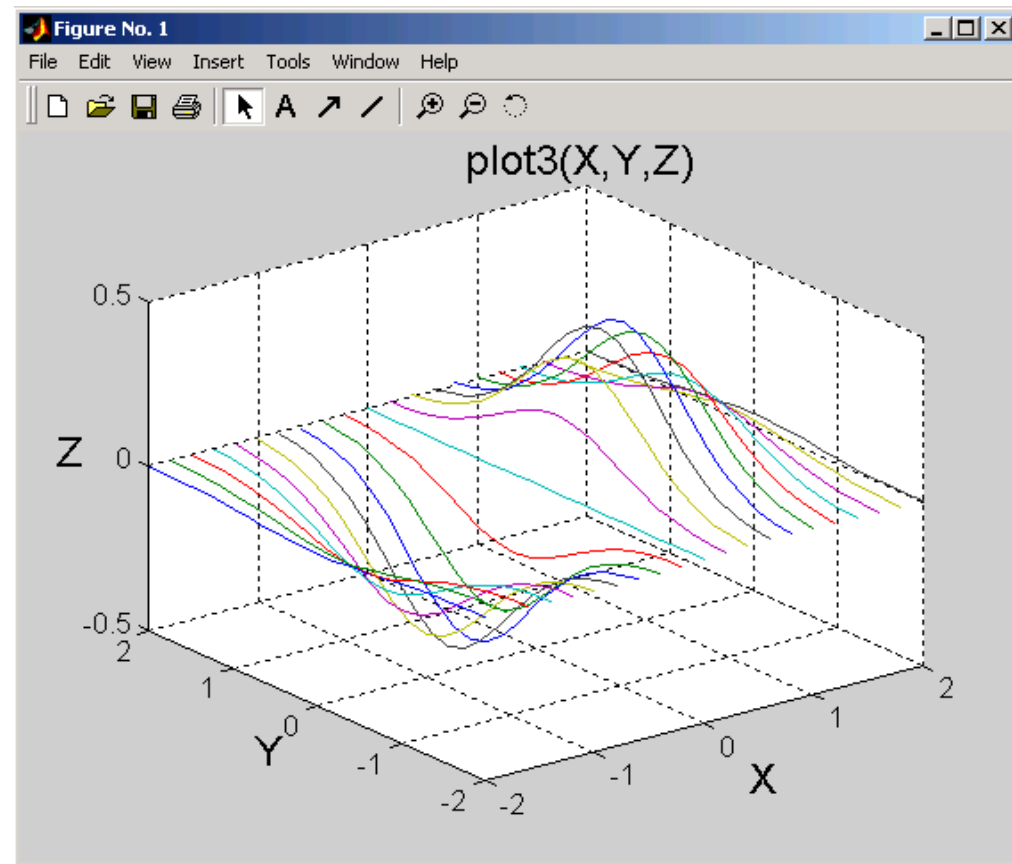
```
title('plot3(X,Y,Z)');
```

```
xlabel('X')
```

```
ylabel('Y')
```

```
zlabel('Z')
```

Meshgrid: NxM coppie di coordinate (x,y)



MATLAB: Visualizzazione

- E' possibile rappresentare delle superfici wireframe

```
[X Y]=meshgrid(-2:0.2:2)
```

```
Z=X.*exp(-X.^2-Y.^2)
```

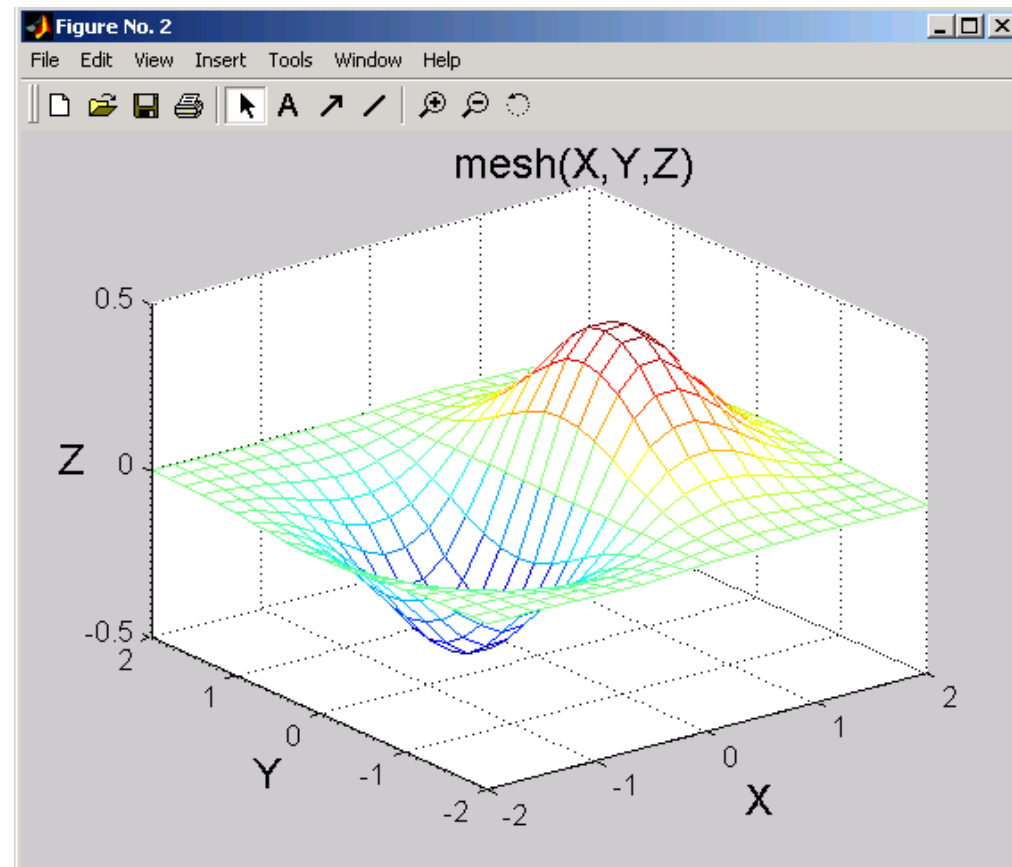
```
mesh(X,Y,Z)
```

```
title('mesh(X,Y,Z)');
```

```
xlabel('X')
```

```
ylabel('Y')
```

```
zlabel('Z')
```



MATLAB: Visualizzazione

- E' possibile rappresentare delle superfici solide

```
[X Y]=meshgrid(-2:0.2:2)
```

```
Z=X.*exp(-X.^2-Y.^2)
```

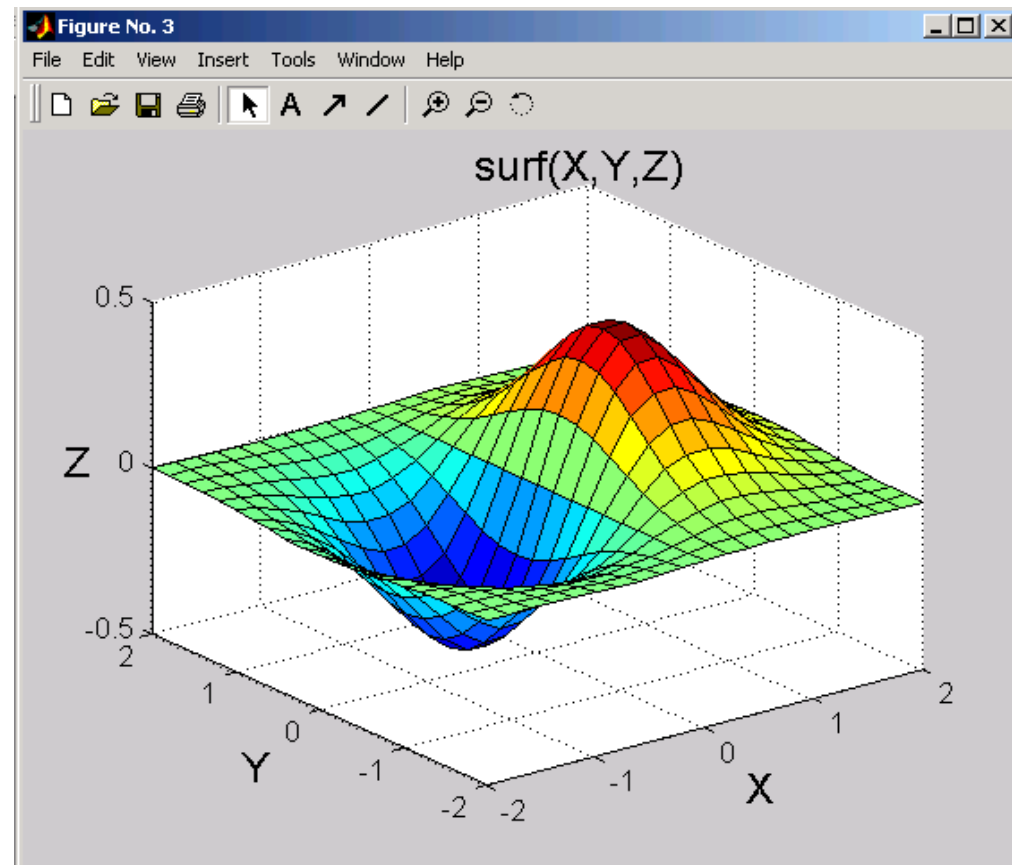
```
surf(X,Y,Z)
```

```
title('surf(X,Y,Z)');
```

```
xlabel('X')
```

```
ylabel('Y')
```

```
zlabel('Z')
```



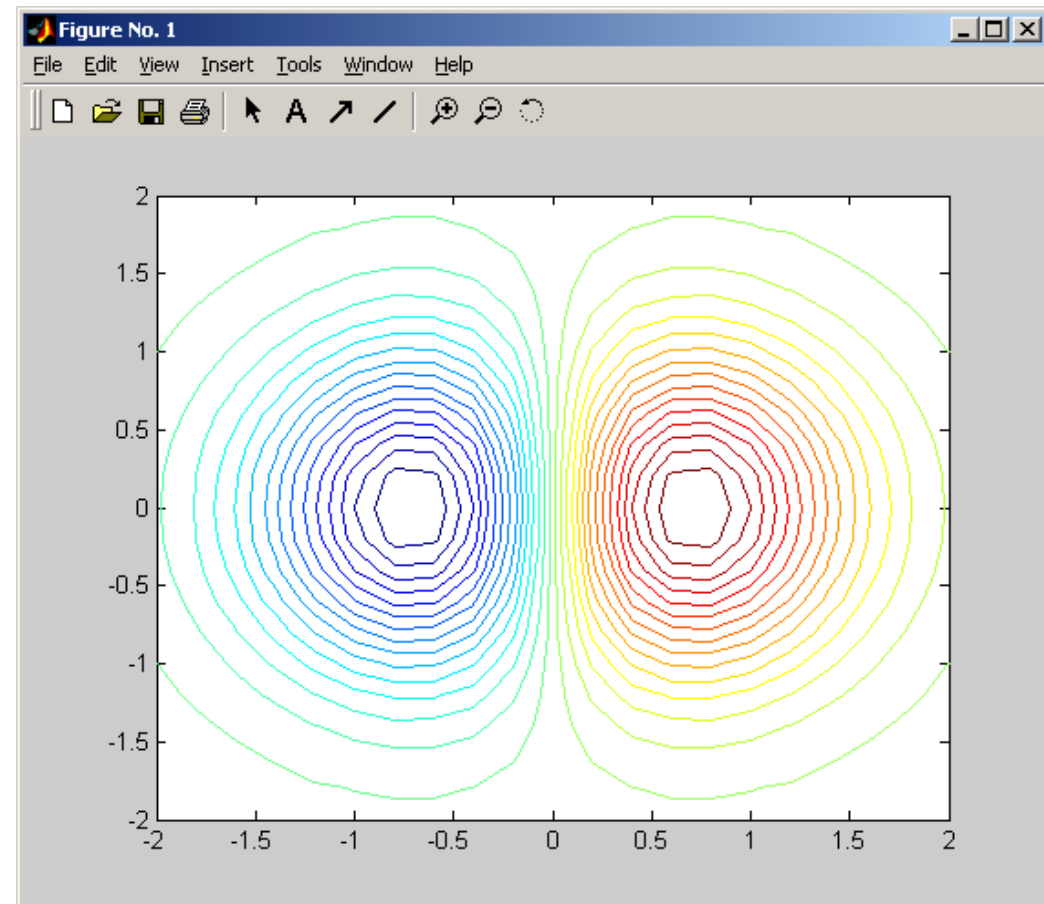
MATLAB: Visualizzazione

- Curve di livello

```
[X Y]=meshgrid(-2:0.2:2)
```

```
Z=X.*exp(-X.^2-Y.^2)
```

```
contour(X,Y,Z);
```



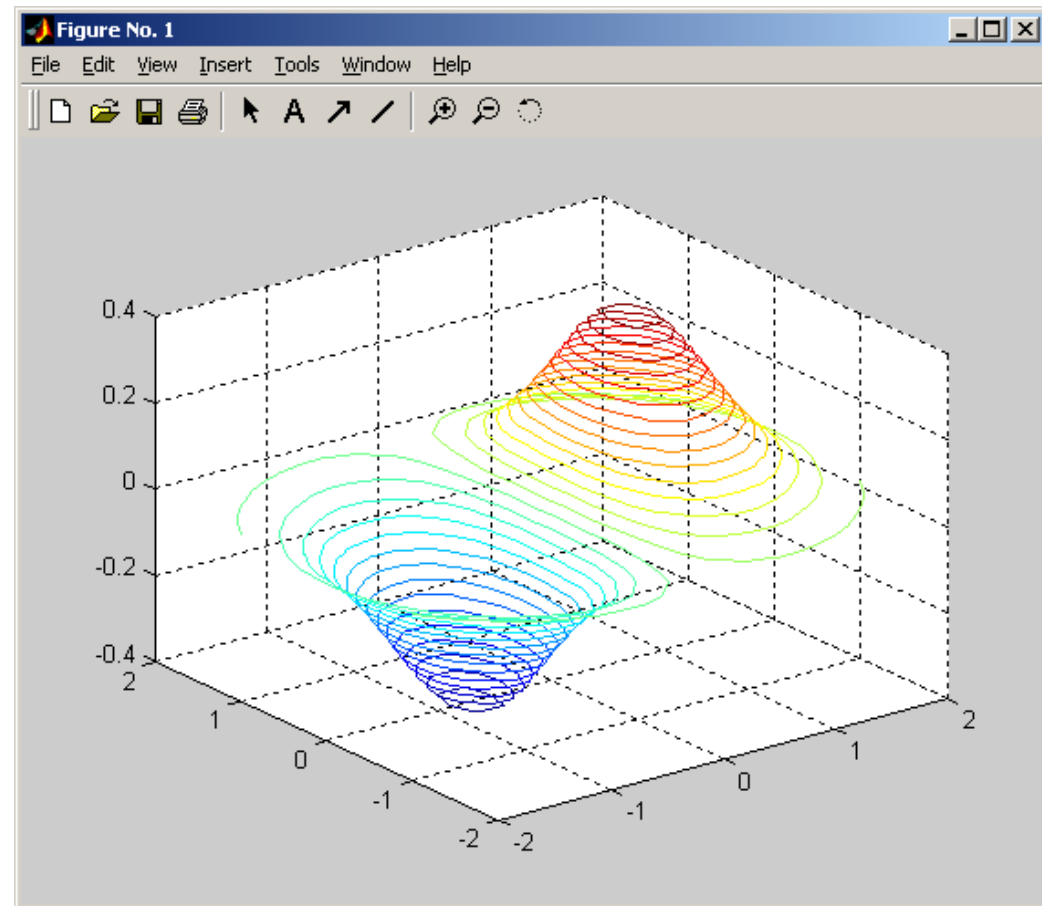
MATLAB: Visualizzazione

- Curve di livello

```
[X Y]=meshgrid(-2:0.2:2)
```

```
Z=X.*exp(-X.^2-Y.^2)
```

```
contour3(X,Y,Z);
```



MATLAB: Visualizzazione

- Rendering

```
[X Y]=meshgrid(-2:0.2:2)
```

```
Z=X.*exp(-X.^2-Y.^2)
```

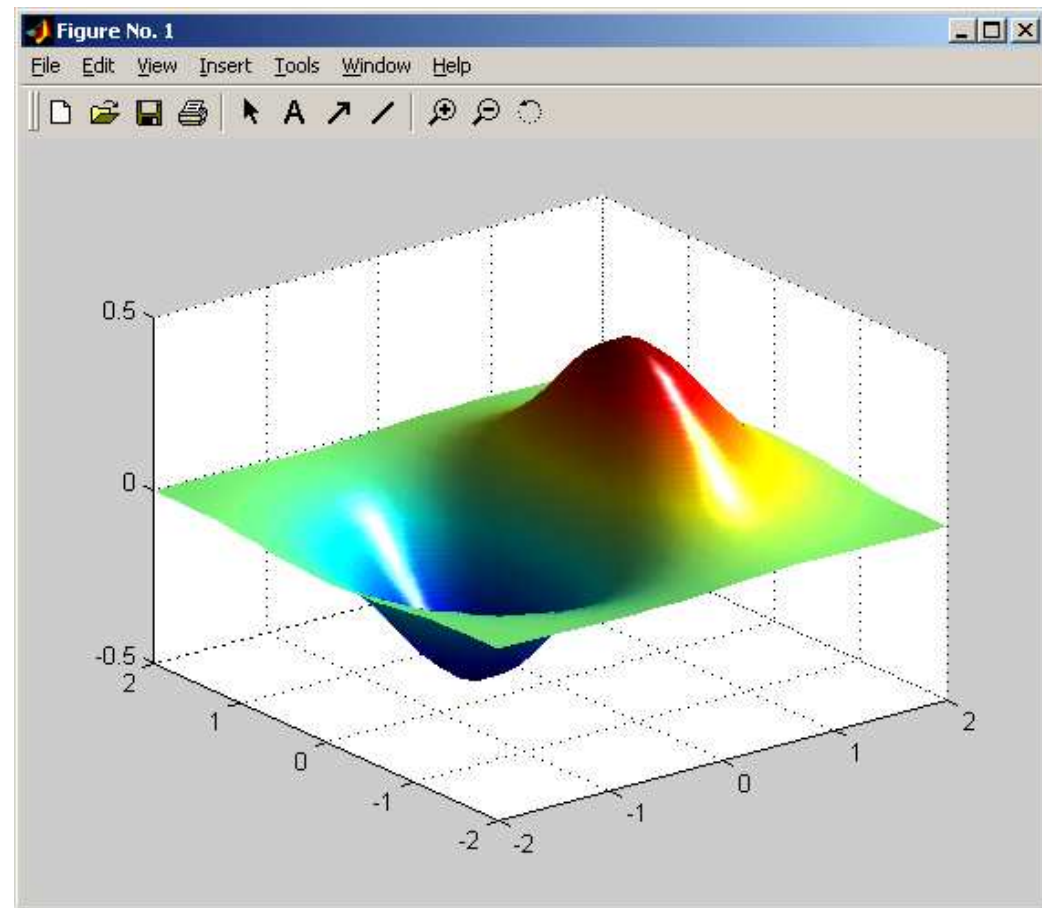
```
surf(X,Y,Z)
```

```
shading interp
```

```
material metal
```

```
lightangle(45,45)
```

```
lighting phong
```



MATLAB: Visualizzazione

- Visualizzare il contenuto di una matrice come una immagine

```
[X Y]=meshgrid(-2:0.2:2)
```

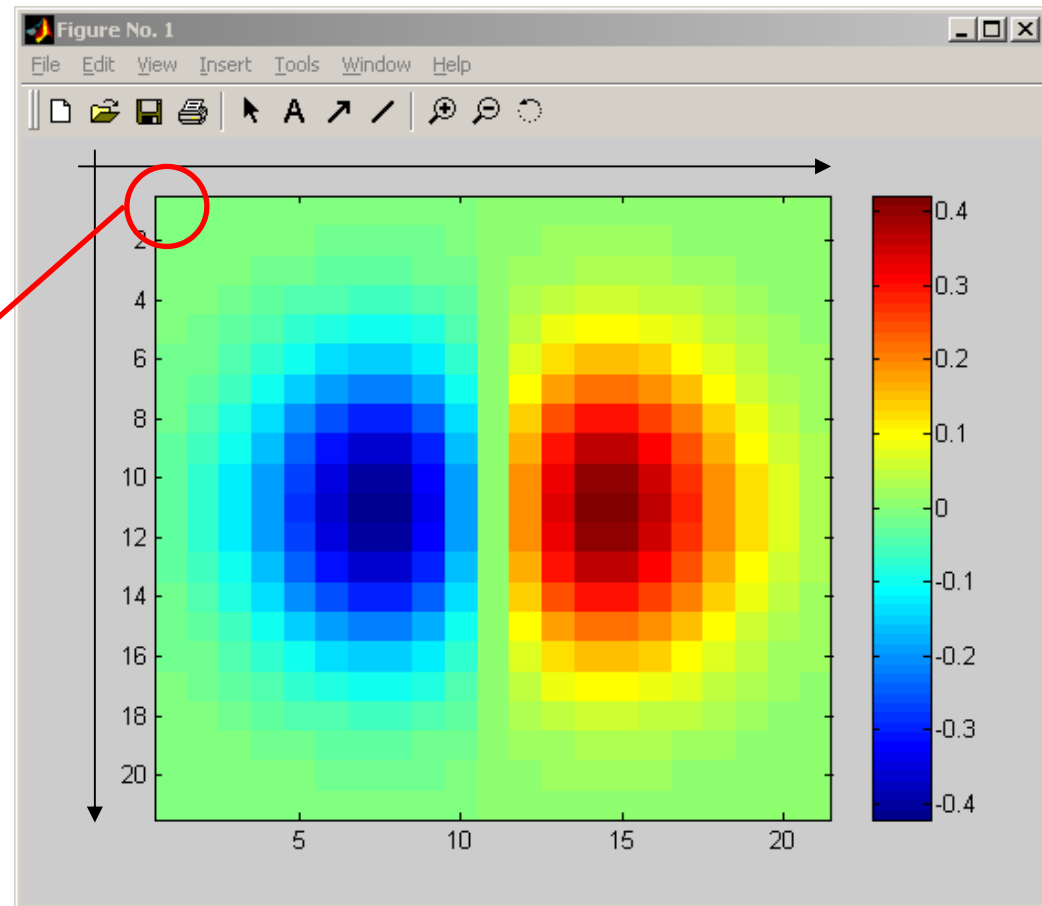
```
Z=X.*exp(-X.^2-Y.^2)
```

```
imagesc(Z)
```

```
colorbar
```

Riscalda i valori della matrice
per rendere le differenze più
visibili

punto (-2,-2)
della funzione



Programmazione

- M-File

- Permettono di definire una serie di operazioni che possono essere eseguite in sequenza tramite chiamata
- Si possono utilizzare tutte le funzioni ed operatori
- E' possibile creare nuove funzioni
- Possibile utilizzare diversi costrutti di controllo

```
if espressione  
    istruzioni  
elseif espressione  
    istruzioni  
end
```

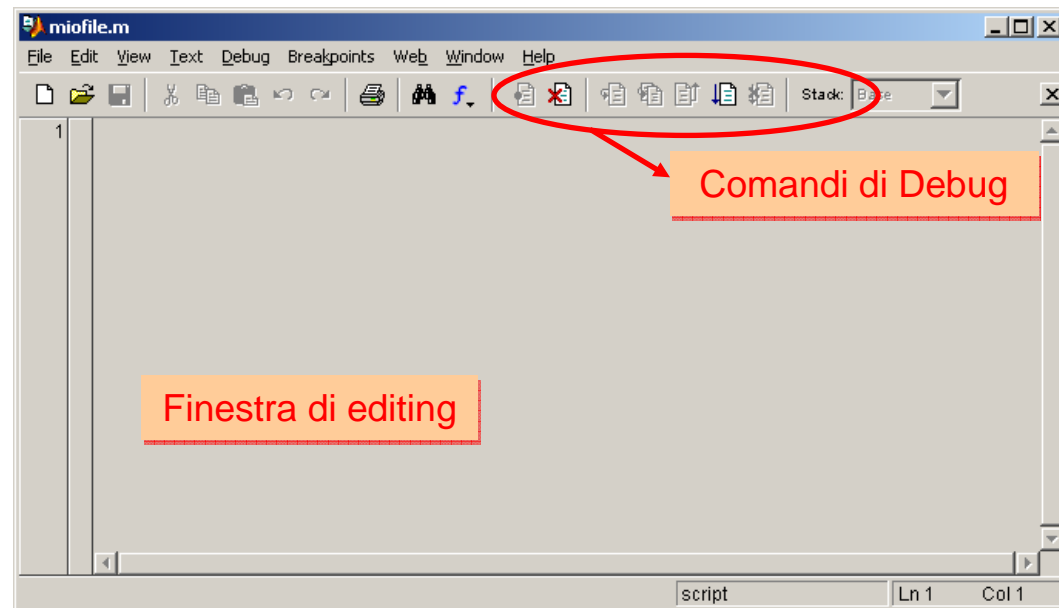
```
while espressione  
    istruzioni  
end
```

```
for variabile=inizio:passo:fine  
    istruzioni  
end
```

Programmazione

- M-File
 - Per editare un m-file di può eseguire il comando

```
edit miofile.m
```



- Alcuni comandi MATLAB sono M-File
- I commenti (non eseguiti) sono preceduti con %

Programmazione: M-File

- Esistono due tipologie di M-File

script	function
Non ha argomenti di input o di output	Ha argomenti di input e di output (keyword function)
Le variabili sono nel workspace globale	Le variabili interne sono locali
Utile per automatizzare una serie di passi ripetitivi	Utile per estendere le funzionalità di MATLAB

Programmazione: M-File

- Script

```
1 % Esempio di script
2
3 x=-pi:pi/100:pi;
4 s=sin(x);
5 c=cos(x);
6
7 if(stato==0)
8     plot(x,s,x,c)
9 else
10    plot(c,s)
11 end
```

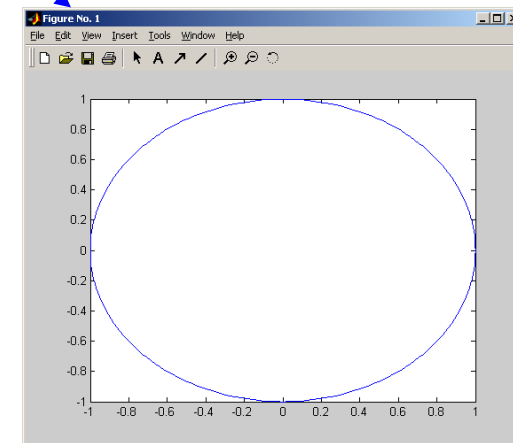
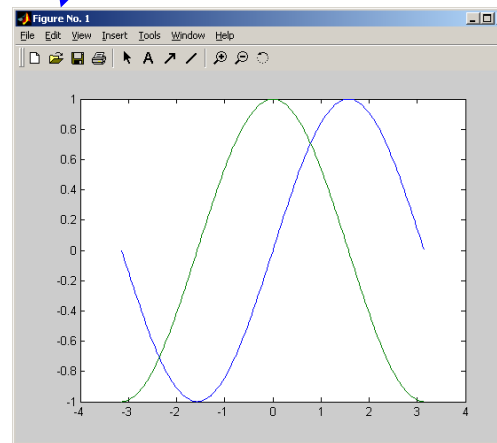
NB: sono nel workspace globale

```
>> mioscript
??? Undefined function or variable 'stato'.

Error in ==> C:\programmi\MATLAB6p5\work\mioscript.m
On line 7 ==> if(stato==0)

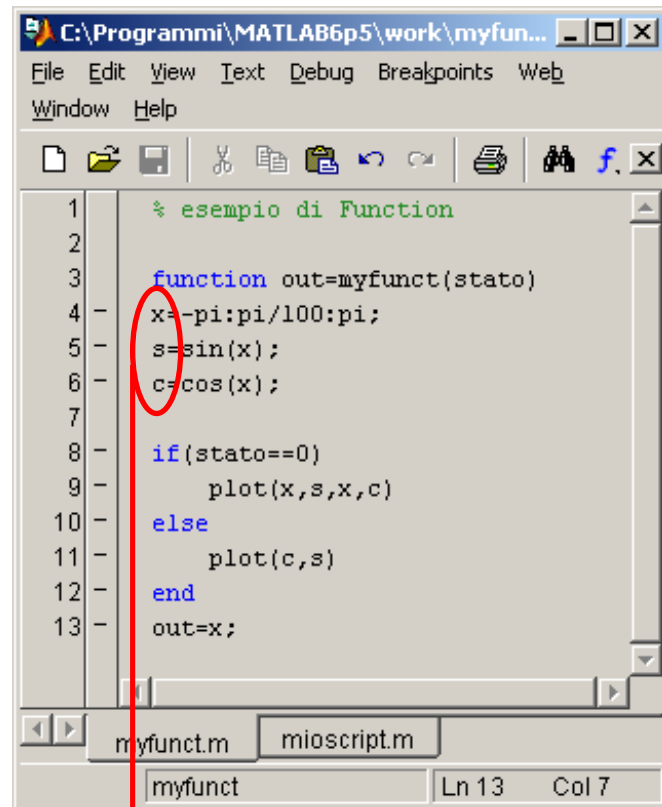
>> stato=0;
>> mioscript
>> stato=1;
>> mioscript
>>
```

Definisco la variabile nel workspace



Programmazione: M-File

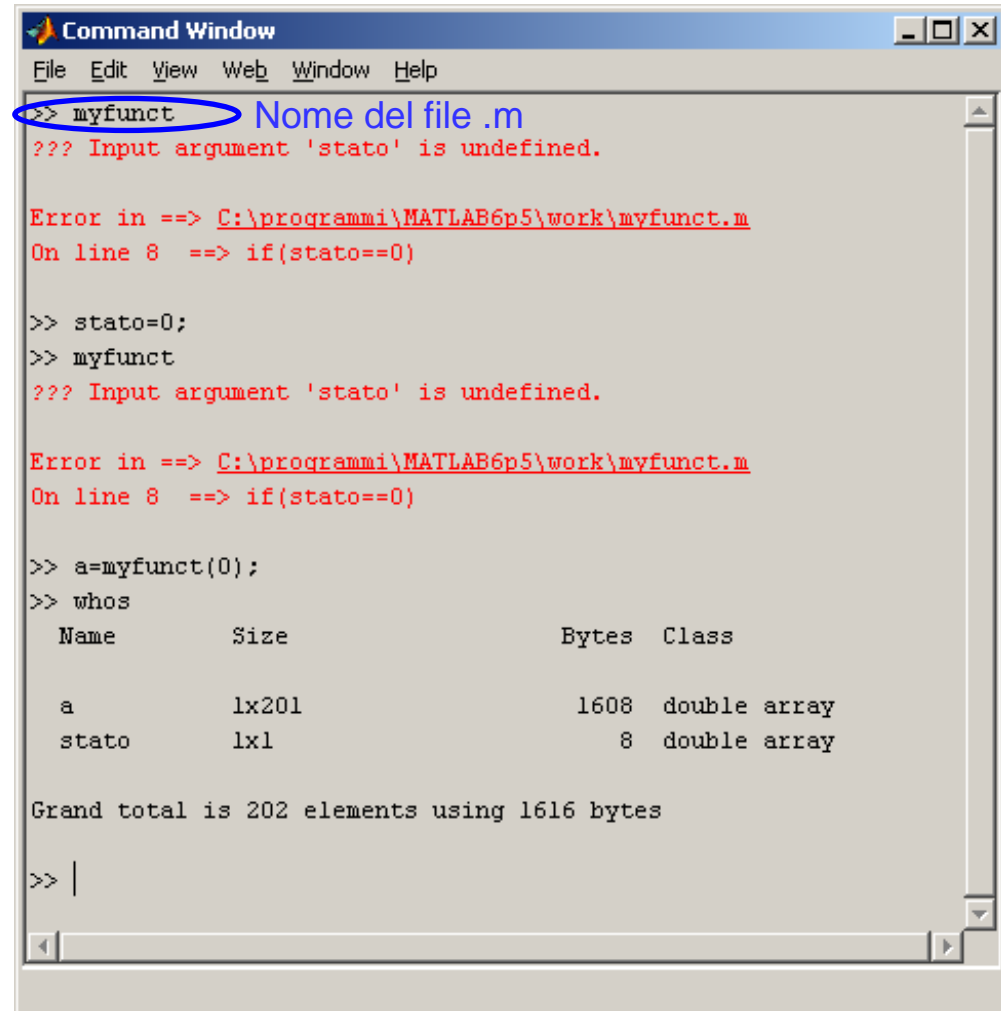
- Function



```
1 % esempio di Function
2
3 function out=myfunct(stato)
4 x=-pi:pi/100:pi;
5 s=sin(x);
6 c=cos(x);
7
8 if(stato==0)
9     plot(x,s,x,c)
10 else
11     plot(c,s)
12 end
13 out=x;
```

The image shows a MATLAB Editor window with the file name 'myfunct.m' and the current cursor position at 'Ln 13 Col 7'. A red circle highlights the function definition line 'function out=myfunct(stato)'. A red arrow points from this circle to the text 'NB: sono locali' below the window.

NB: sono locali



```
Command Window
File Edit View Web Window Help
>> myfunct Nome del file .m
??? Input argument 'stato' is undefined.

Error in ==> C:\programmi\MATLAB6p5\work\myfunct.m
On line 8 ==> if(stato==0)

>> stato=0;
>> myfunct
??? Input argument 'stato' is undefined.

Error in ==> C:\programmi\MATLAB6p5\work\myfunct.m
On line 8 ==> if(stato==0)

>> a=myfunct(0);
>> whos
Name          Size          Bytes  Class
a              1x201         1608  double array
stato          1x1            8   double array

Grand total is 202 elements using 1616 bytes

>> |
```

The image shows the MATLAB Command Window. The first command is '>> myfunct', which results in an error: '??? Input argument 'stato' is undefined.' The second command is '>> stato=0; >> myfunct', which also results in the same error. The third command is '>> a=myfunct(0);', which executes successfully. The fourth command is '>> whos', which displays the following table:

Name	Size	Bytes	Class
a	1x201	1608	double array
stato	1x1	8	double array

The 'Grand total' is 202 elements using 1616 bytes. The Command Window prompt '>>' is shown at the bottom.

Programmazione: M-File

- Function

```
% funzione con un solo valore di output
```

```
function res=nome_funzione(parametri)
```

```
% funzione con diversi valori di output
```

```
function [res1,res2,res3]=nome_funzione(parametri)
```

```
% funzione senza valori di output
```

```
function nome_funzione(parametri)
```

- Possono essere definite variabili globali

```
global nome_variabile
```

- Possono essere definite più funzioni ma

- La funzione associata al file .m è la prima

Programmazione: M-File

- Input da tastiera

```
% input generico
res=input('Inserisci un numero ');

% input di una stringa
res=input('Nome del file: ','s');
```

- Performance
 - Evitare il più possibile cicli for
 - MATLAB è ottimizzato per operazioni matriciali
 - Fare molte operazioni matriciali è più veloce che fare un solo ciclo

```
N=M+10.*(not(M==0))+5.*(M==0)
```



Segnali elementari a tempo discreto ed esercizi

Elaborazione Numerica dei Segnali

a.a. 2008/2009

Simone Bianco

Elaborazione numerica dei segnali

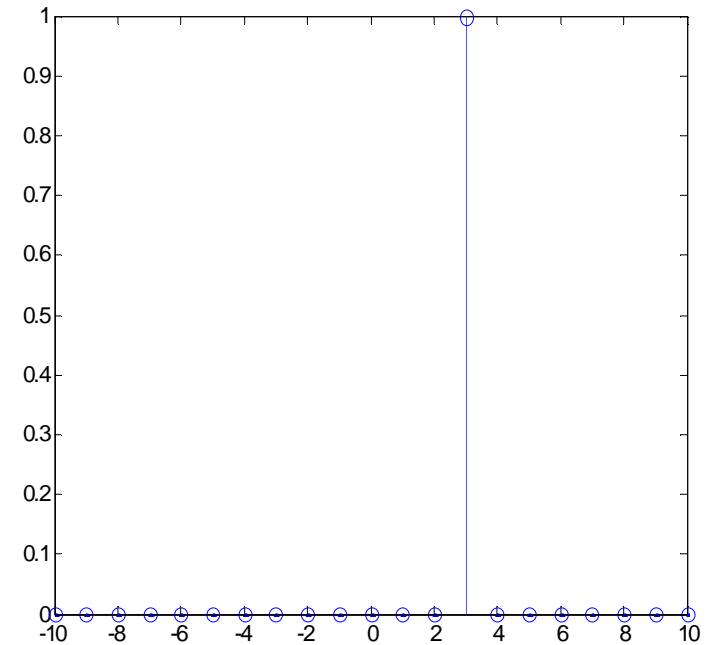
- Creazione di funzioni all'interno di m-files per la costruzione di alcuni tipi di segnali elementari a tempo discreto
- Esercizi

Segnali elementari a tempo discreto

- IMPULSO

```
function [x,n]=impulso(a,b,n0)
% genera x(n)=delta(n-n0), a<=n<=b
n=a:b;
x=double([ (n-n0)==0 ]);
```

```
>>[x,n]=impulso(-10,10,3);
>>stem(n,x)
```

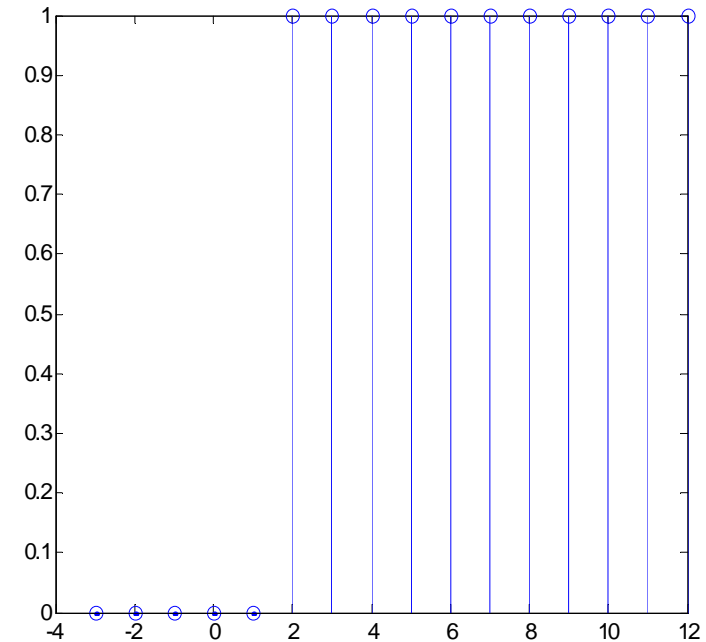


Segnali elementari a tempo discreto

- GRADINO

```
function [x,n]=gradino(a,b,n0)
% genera x(n)=u(n-n0), a<=n<=b
n=a:b;
x=double([ (n-n0)>=0 ]);
```

```
>>[x,n]=gradino(-3,12,2);
>>stem(n,x)
```

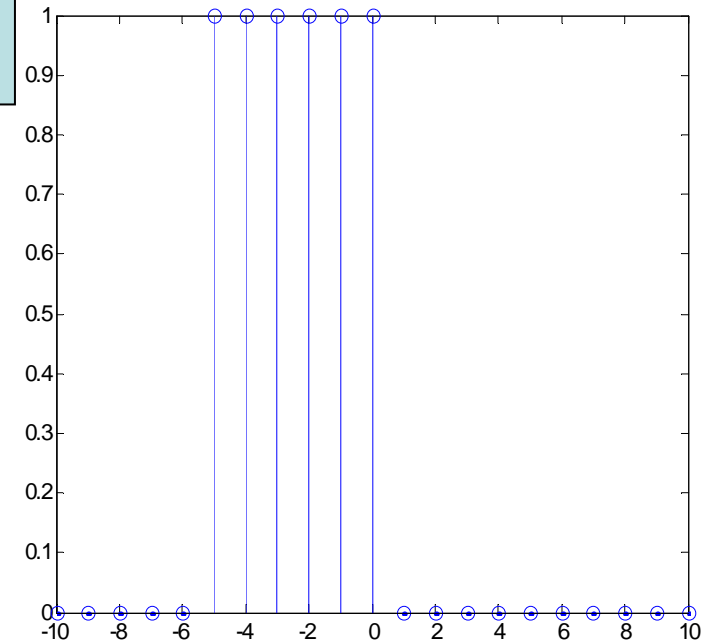


Segnali elementari a tempo discreto

- **FINESTRA**

```
function [x,n]=finestra(a,b,n0,n1)
% genera  $x(n)=u(n-n_0)-u(n-n_1)$ ,  $a \leq n \leq b$ 
n=a:b;
x=double((n-n0)>=0) - double((n-n1)>0);
```

```
>>[x,n]=finestra(-10,10,-5,0);
>>stem(n,x)
```

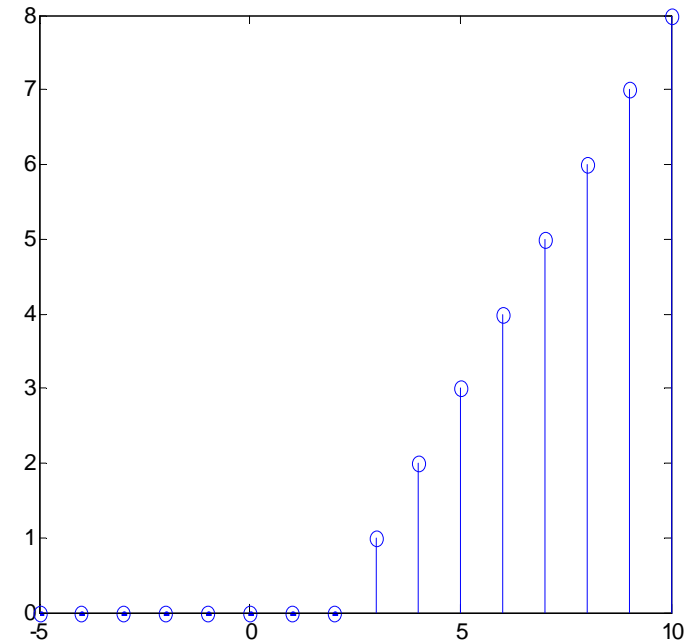


Segnali elementari a tempo discreto

- **RAMPA**

```
function [x,n]=rampa(a,b,n0)
% genera x(n)=n, a<=n<=b
n=a:b;
x=(n-n0).*double((n-n0)>=0);
```

```
>>[x,n]=rampa(-5,10,2);
>>stem(n,x)
```

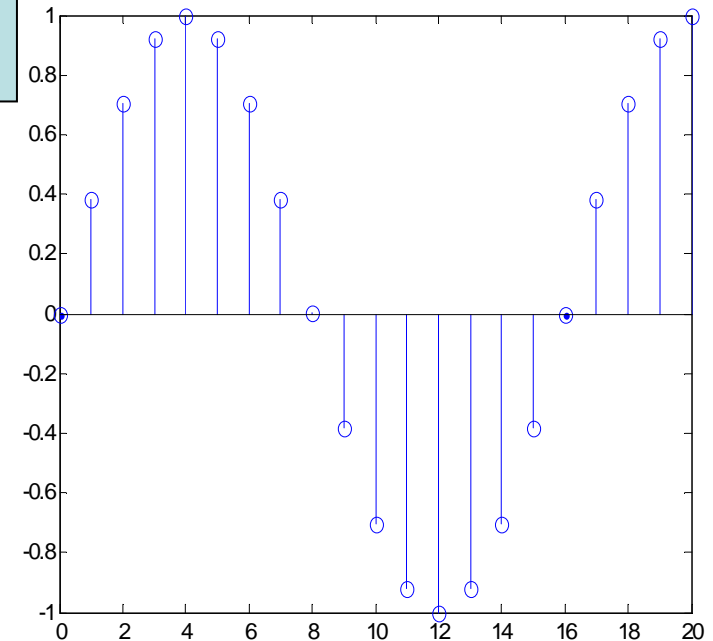


Segnali elementari a tempo discreto

- SINUSOIDE

```
function [x,n]=sinusoide(A,f0,phi,a,b)
% genera  $x(n)=A*\sin(\pi*f0*n+phi)$ 
n=a:b;
x=A*sin(pi*f0*n+phi);
```

```
>>[x,n]=sinusoide(1,1/8,0,0,20);
>>stem(n,x)
```



Esercizio #1

- Dati i segnali analogici

$$x_{a1}(t) = \cos(20\pi t)$$

$$x_{a2}(t) = \cos(100\pi t)$$

campionati con frequenza di campionamento $F_s=40\text{Hz}$, determinare i corrispondenti segnali campionati e disegnarli. Determinare infine i corrispondenti segnali ricostruiti.

Esercizio #1

- Soluzione

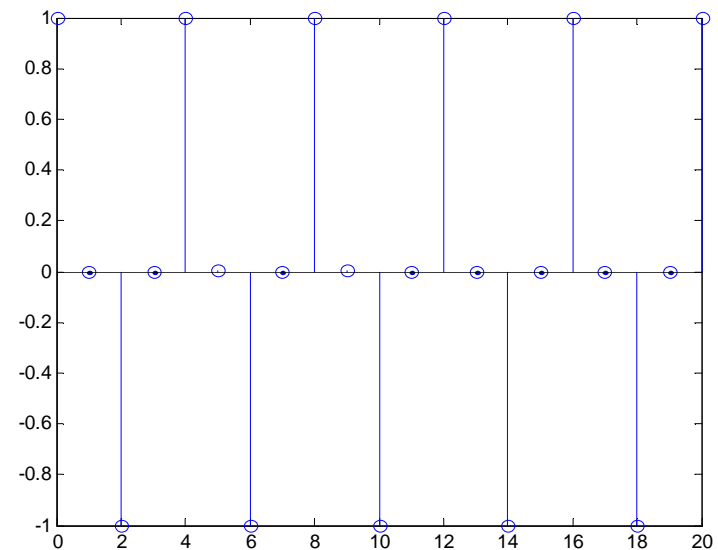
$$x_1(n) = \cos\left(\frac{20}{40}\pi n\right) = \cos\left(\frac{1}{2}\pi n\right) = \cos\left(\frac{1}{4}2\pi n\right)$$

$$x_{a_2}(n) = \cos(100\pi t) = \cos\left(\frac{100}{40}\pi n\right) = \cos\left(\frac{5}{2}\pi n\right) = \cos\left(\left[2\pi + \frac{1}{2}\pi\right]n\right) = \cos\left(\frac{1}{2}\pi n\right) = \cos\left(\frac{1}{4}2\pi n\right)$$

Frequenza normalizzata = $\frac{1}{4}$ cycles/sample,
ovvero 4 samples/cycle

- Segnali ricostruiti

$$x_{a_1}(t) = x_{a_2}(t) = \cos\left(\frac{1}{2}40\pi t\right) = \cos(20\pi t)$$



Esercizio #2

- Dati i segnali analogici

$$x_{a1}(t) = \cos(20\pi t)$$

$$x_{a2}(t) = \cos(100\pi t)$$

campionati con frequenza di campionamento $F_s=50\text{Hz}$, determinare i corrispondenti segnali campionati e disegnarli. Determinare infine i corrispondenti segnali ricostruiti.

Esercizio #2

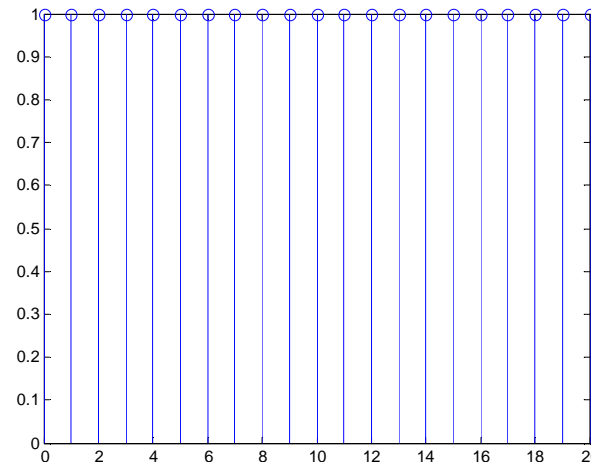
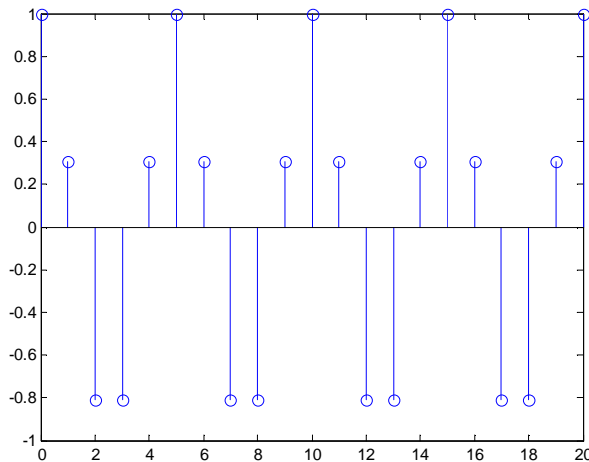
- Soluzione

$$x_1(n) = \cos\left(\frac{20}{50}\pi n\right) = \cos\left(\frac{2}{5}\pi n\right) = \cos\left(\frac{1}{5}2\pi n\right)$$

Frequenza normalizzata = 1/5 cycles/sample, ovvero 5 samples/cycle

$$x_{a2}(n) = \cos(100\pi t) = \cos\left(\frac{100}{50}\pi n\right) = \cos(2\pi n) = \cos(0n) = 1$$

Frequenza normalizzata = 0 cycles/sample



- Segnali ricostruiti

$$x_{a1}(t) = \cos\left(\frac{2}{5}50\pi t\right) = \cos(20\pi t)$$

$$x_{a2}(t) = \cos(0 \cdot 50\pi t) = \cos(0\pi t) = 1$$