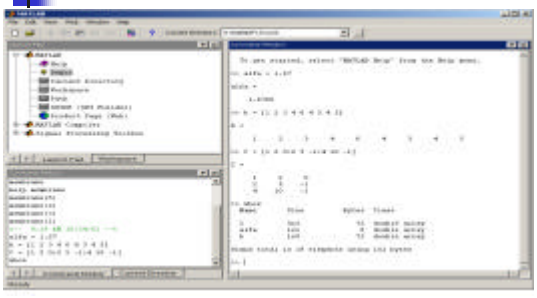
## Introduzione a MATLAB

## MATLAB=MATrix LABoratory

- È un sistema in cui ogni dato è rappresentato sotto forma di una matrice di numeri.
- Permette di:
  - Effettuare operazioni tra matrici.
  - Rappresentare in forma grafica il contenuto delle matrici.

## Interfaccia utente



## Linea di comando

**La linea di comando di MATLAB è indicata da un prompt come in DOS: >> .**

**Accetta dichiarazioni di variabili, espressioni e chiamate a tutte le funzioni disponibili nel programma.**

**Tutte le funzioni di MATLAB non sono altro che files di testo, simili a quelli che l'utente può generare con un text editor, e vengono eseguite semplicemente digitandone il nome sulla linea di comando.**

**MATLAB permette inoltre di richiamare le ultime righe di comandi inseriti usando le frecce in alto e in basso.**

## Help di Matlab

MATLAB presenta un help in linea con informazioni sulla sintassi di tutte le funzioni disponibili.

Per accedere a queste informazioni, basta digitare:

**help nome_funzione**

È anche possibile avere un help di tutte le funzioni di una certa categoria; ad esempio per sapere quali sono le funzioni specifiche per l'analisi ed il controllo di sistemi dinamici, basta digitare:

**help control**

Per sapere quali sono le varie categorie di funzioni disponibili (i cosiddetti *toolbox*), basta digitare:

**help**

## I files di Matlab

I files interpretati dal programma sono file di testo ASCII con estensione .m ; sono generati con un text editor e sono eseguiti in MATLAB semplicemente digitandone il nome sulla linea di comando (senza estensione!).

È possibile inserire dei commenti al loro interno precedendo ogni linea di commento col percento %

*Attenzione!* Può essere molto utile andare nelle directories dove si trova il programma ed analizzare come le varie funzioni sono state implementate.

Ciò è possibile poichè ogni funzione ed ogni comando MATLAB richiama un file .m

## Punteggiatura

Le istruzioni (siano esse contenute in un file .m lanciato da MATLAB, oppure digitate direttamente dalla linea di comando) vanno normalmente terminate con un punto e virgola, altrimenti è visualizzato il risultato dell'applicazione dell'istruzione.

```
Es: var1=6;
Es: var2=linspace(-10,10,10000);
```

## Nomi delle variabili

Le variabili seguono le regole dei linguaggi di programmazione come il Pascal o il C. MATLAB è

*case-sensitive*

e accetta nomi di variabili lunghi fino ad un massimo di 19 caratteri alfanumerici, con il primo obbligatoriamente alfabetico. Sono nomi leciti

x , x1, X1, ossido, silice

NON sono leciti :

1X, nome_troppo_lungo_per_essere_valido

## Visualizzazione risultato

Per visualizzare il contenuto di una variabile è sufficiente digitarne il nome senza punto e virgola sulla linea di comando.

Tutti i calcoli effettuati in MATLAB sono eseguiti in doppia precisione.

Il risultato dell'ultima operazione è memorizzato nella variabile ans.

## Esempio 1

- Matrice 1x1, è composta da 1 riga e da 1 colonna:

```
>> alfa = 1.67
alfa =
        1.6700
```

## Esempio 2

- Matrice 1x9, è composta da 1 riga e da 9 colonne:

```
>> b = [1 2 3 4 6 4 3 4 5]
b =
    1 2 3 4 6 4 3 4 5
```

## Esempio 3

- Matrice 3x3, è composta da 3 righe e da 3 colonne:

```
>> C = [1 2 0; 2 5 –1;4 10 -1]
C =
    1   2   0
    2   5   –1
    4   10  -1
```

## Comando **whos**

- Digitando **whos** si ottiene il riepilogo dei dati introdotti dall'utente.

```
>> whos
  Name    Size     Bytes    Class
  C        3x3        72    double array
  alfa     1x1         8    double array
  b        1x9        72    double array
```

## Comando **whos**

- Digitando **whos** si ottiene il riepilogo dei dati introdotti dall'utente.

```
>> whos
  Name    Size     Bytes    Class
  C        3x3        72    double array
  alfa     1x1         8    double array
  b        1x9        72    double array
```

Ogni numero è memorizzato in floating point in doppia precisione ed occupa 8 bytes

## Operazioni aritmetiche

- Potenza: **^**
- Prodotto: *****
- Divisone: **/**
- Somma: **+**
- Sottrazione: **-**

## Esempio

```
>> c = [ 1 2 3]
>> d = c+2
d =
     3 4 5
>> e = d*1.5
e =
    4.5000 6.0000 7.5000
```

## Operazioni su matrici

- Somma di matrici: **+**
- Sottrazione tra matrici: **-**
- Le due matrici devono avere le stesse dimensioni:
  - Due matrici 2x2 possono essere sommate
  - Una matrice 3x3 non può essere sommata ad una matrice 2x2.

## Esempio 1

```
>> a=[1 2;3 4]
a =
     1     2
     3     4
>> b = [3 4;5 6]
b =
     3     4
     5     6
```

```
>> c = a+b
c =
     4     6
     8    10
```

## Esempio 2

```
>> a=[1 2;3 4]
a =
     1     2
     3     4
>> b = [3 4 5;5 6 7;0 0 0]
b =
     3     4     5
     5     6     7
     0     0     0
```

```
>> c = a+b
??? Error using ==> +
Matrix dimensions
  must agree.
```

## Prodotto di due matrici

- Si effettua utilizzando *
- È necessario che le dimensioni delle due matrici siano concordi:
  - È possibile moltiplicare una matrice 2x1 per una matrice 1x2, ottenendo una matrice 2x2.
  - Non è possibile moltiplicare una matrice 1x2 per una matrice 1x2.

## Esempio 1

```
>> a=[1;2]
a =
     1
     2
>> b =[3 4]
b =
     3     4
```

```
>> c = a*b
c =
     3     4
     6     8
```

## Esempio 2

```
>> c=a*a
??? Error using ==> *
Inner matrix dimensions must agree.
```

## Operazioni elemento per elemento

- MATLAB permette di eseguire operazioni tra gli elementi di due matrici che abbiamo identica dimensione:
  - Prodotto elemento per elemento: `.*`
  - Divisione elemento per elemento: `./`
  - Potenza elemento per elemento: `.^`

## Esempio

```
>> a=[4 16 256]
a =
     4    16   256
>> b=[0.5 0.5 0.25]
b =
    0.5000    0.5000    0.2500
>> c = a.^b
c =
     2     4     4
```

## Funzioni predefinite

- Radice quadrata: **sqrt(x)**
- Arrotondamento all'intero più vicino: **round(x)**
- Parte intera di un numero: **fix(x)**
- Segno di un numero: **sign(x)**
- Funzioni trigonometriche: **sin(x), cos(x),…**
- Funzioni iperboliche: **sinh(x), cosh(x),…**
- Funzioni trigonometriche inverse: **asin(x),…**
- Esponenziale: **exp(x)**
- Logaritmo naturale: **log(x)**
- Logarimo in base 10: **log10(x).**

## Valori predefiniti

- PI greco: **pi**
- Unità immaginaria: **i**, **j**
- Infinito: **Inf**
- Base del logaritmo naturale: **eps**

## Grafica

- MATLAB permette di rappresentare graficamente i dati contenuti in una matrice.
- Esempio:

```
>> plot(e)
>> hold on
>> plot(d)
>> grid on
```

## Grafica

- MATLAB permette di rappresentare graficamente i dati contenuti in una matrice.
- Esempio:

```
>> plot(e)
>> hold on
>> plot(d)
>> grid on
```

Disegna su di un piano cartesiano i segmenti che uniscono tutti i punti contenuti nella matrice e.

## Grafica

- MATLAB permette di rappresentare graficamente i dati contenuti in una matrice.
- Esempio:

```
>> plot(e)
>> hold on
>> plot(d)
>> grid on
```

Fa si che al prossimo disegno il contenuto del piano cartesiano non venga cancellato.

## Grafica

- MATLAB permette di rappresentare graficamente i dati contenuti in una matrice.
- Esempio:

```
>> plot(e)
>> hold on
>> plot(d)
>> grid on
```

Disegna su di un piano cartesiano i segmenti che uniscono tuttii punti contenuti nella matrice d.

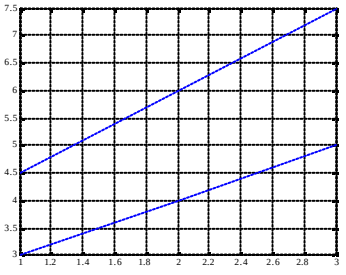
## Grafica

- MATLAB permette di rappresentare graficamente i dati contenuti in una matrice.
- Esempio:

```
>> plot(e)
>> hold on
>> plot(d)
>> grid on
```

Visuallizza sul piano cartesiano una griglia per i valori sull'asse x e sull'asse y.

## Risultato



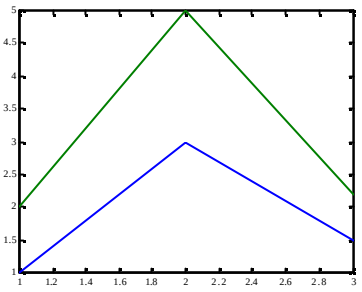
## Tipi di grafico

- MATLAB offre diverse possibilità:
  - Grafici a linea
  - Grafici a barre
  - Grafici con simboli definiti dall'utente
- Esempio:

```
f=[1 2; 3 5; 1.5 2.2]
```

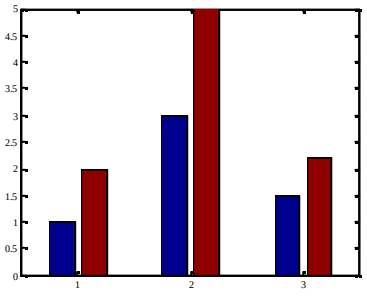
## Grafici a linea

```
plot(f)
```



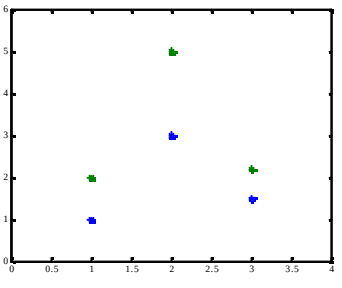
## Grafici a barre

```
bar(f)
```



## Grafici con simboli definiti dall'utente

```
plot(f,'*')
```



## Disegno di funzioni 2D

- MATLAB permette di disegnare funzioni 2D su di un intervallo definito dall'utente.
- È necessario:
  - Definire la funzione da disegnare.
  - Definire l'interevallo su cui effettuare il disegno.

## Esempio

- Si rappresenti la seguente funzione sull'intervallo I=[-1,2]:

$$f(x) = (2x - \sqrt{2})^2 \sin(2x)$$

## Definizione della funzione

- Si utilizza la seguente sintassi:

```
Funzione = 'espressione'
```

- Nel nostro caso si ha:

```
>> f='(2*x-sqrt(2))^2*sin(2*x)'
```

## Definizione dell'intervallo

- L'intervallo viene definito al momento del disegno della funzione utilizzando la seguente sintassi:

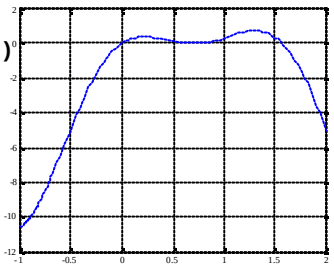
```
fplot(funzione, intervallo)
```

- Nel nostro caso si ha:

```
>> fplot(f, [-1,2])
```

## Risultato

```
fplot(f, [-1,2])
```




## Disegno di funzioni 3D

- MATLAB permette di disegnare funzioni 3D su di un dominio definito dall'utente.
- È necessario:
  - Definire il dominio su cui effettuare il disegno.
  - Definire la funzione da disegnare.

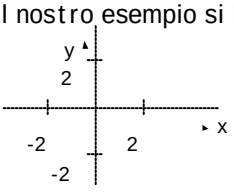
## Esempio

- Si rappresenti la funzione f(x,y) sul dominio D={-2 ≤ x ≤ 2; -2 ≤ y ≤ 2} dove:

$$f(x,y) = xe^{-(x^2-y^2)}$$

## Dominio

- È l'insieme di valori (x,y) per cui si vuole disegnare la funzione.
- Nel nostro esempio si ha:



## Definizione del dominio

- Si utilizza la seguente sintassi:

```
[x,y]= meshgrid(Xmin:Xstep:Xmax,Ymin:Ystep:Ymax);
```

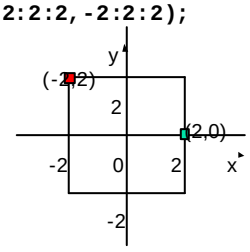
- $X_{min}$ è il più piccolo valore di x per cui si vuole disegnare la funzione.
- $X_{max}$ è il più grande valore di x per cui si vuole disegnare la funzione.
- $X_{step}$ è la distanza tra due punti adiacenti sull'asse x.

## Definizione del dominio

- Un punto $P_{ij}$ appartenente al dominio è definito dalla coppia $(x_{ij}, y_{ij})$ dove:
  - $x_{ij}$ è l'emento memorizzato in x alla riga i, colonna j.
  - $y_{ij}$ è l'emento memorizzato in y alla riga i, colonna j.

## Esempio

```
>>[x,y] = meshgrid(-2:2:2,-2:2:2);
x =
    -2     0     2
    -2     0     2
    -2     0     2
y =
    -2    -2    -2
     0     0     0
     2     2     2
```




## Definizione della funzione

- Si utilizza la stessa sintassi usata per le funzioni 2D.
- Nel nostro esempio si ha:

```
>> f='x.*exp(-x.^2-y.^2)'
```

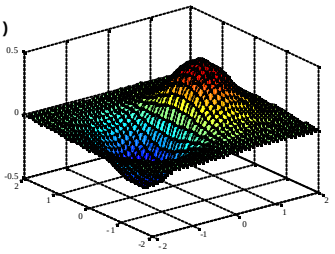
- **x** e **y** fanno riferimento alle matrici utilizzate per definire il dominio.
- Le operazioni algebriche tra **x** e **y** devono essere eseguire elemento per elemento.
- È necessario usare: `.*  ./  .^`

## Disegno della funzione

- Si effettua in due passi:
  - **`z=eval(f)`**: calcola il valore della funzione per tutte le coppie di punti del dominio.
  - **`surf(x,y,z)`**: disegna la funzione.

## Risultato

```
>> [x,y]=meshgrid(-2:.1:2,-2:.1:2);
>> z=eval(f);
>> surf(x,y,z)
```



## Polinomi

- MATLAB permette:
  - Definire polinomi
  - Eseguire operazioni tra polinomi dello stesso ordine.

## Definizione di un polinomio

- Si effettua definendo un vettore i cui elementi siano i monomi del polinomio scritti in ordine di potenza decrescente
- Esempio:

$$p = x^4 + 111 \cdot x^3 + 1100 \cdot x^2 + 1000 \cdot x + 4$$

p=[1 111 1100 1000 4]

## Operazioni tra polinomi

- p1=[1 1 1] $p1 = x^2 + x + 1$
- p2=[1 111 1000] $p2 = x^2 + 111 \cdot x + 1000$
- Somma=p1+p2
- Differenza=p1-p2
- Prodotto=conv(p1, p2)

## Operazioni tra polinomi

- Calcolo delle radici:
  - P=[1 3 5 2]
  - R=roots(P)
- Funzione inversa:
  - PP=poly(R)

## IF-THEN-ELSE

```
if condizione1,
      operazioni1;
elseif condizione2,
      operazioni2;
else
      operazioni3;
end;
```

## Condizioni

- Maggiore: >
- Minore: <
- Maggiore-uguale: >=
- Minore-uguale: <=
- Uguale: ==
- Diverso: ~=
- AND logico: &
- OR logico: |
- NOT logico: ~

## Esempio

```
if n==10,
        a=b*c,
        d=e/f;
elseif n~=20,
        a=e*f,
        d=a/b;
else
        Disp('Errore !!!');
end;
```

## Cicli

| Matlab | C/C++ |
| --- | --- |
| for k=1:s:N,<br>Operazioni;<br>end; | for (k=1;k<=N;k+=step){<br>blocco istruzioni<br>} |
| while condizione,<br>Operazioni;<br>end; | while (condizione){<br>blocco istruzioni<br>} |

## Stringhe input e output

Il testo in MATLAB viene inserito sempre tra apici:
Es.: string='Ciao';

Per visualizzare stringhe o messaggi si adopera la funzione disp.
Es.: disp('Premere un tasto');

## Messaggi di errore e Input

La funzione error mostra un messaggio di errore ed interrompe l'esecuzione di un file .m
Es.: error('A deve essere simmetrica');

La funzione input mostra un messaggio e permette l'inserimento di dati.
Es.: num_di_iter=input('Inserire il numero di iterazioni: ');