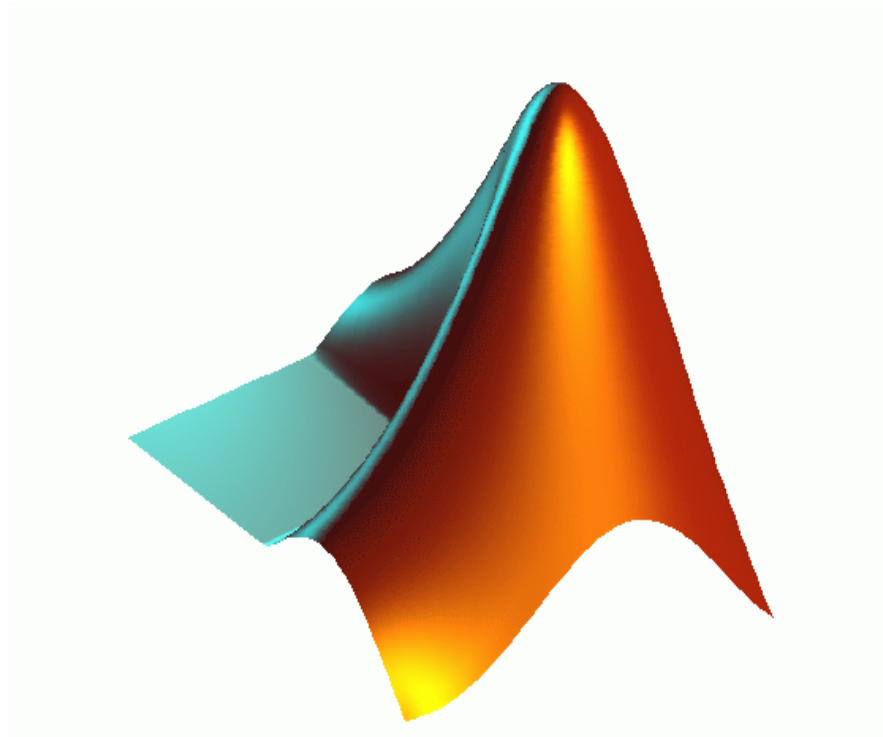


INTRODUZIONE A MATLAB

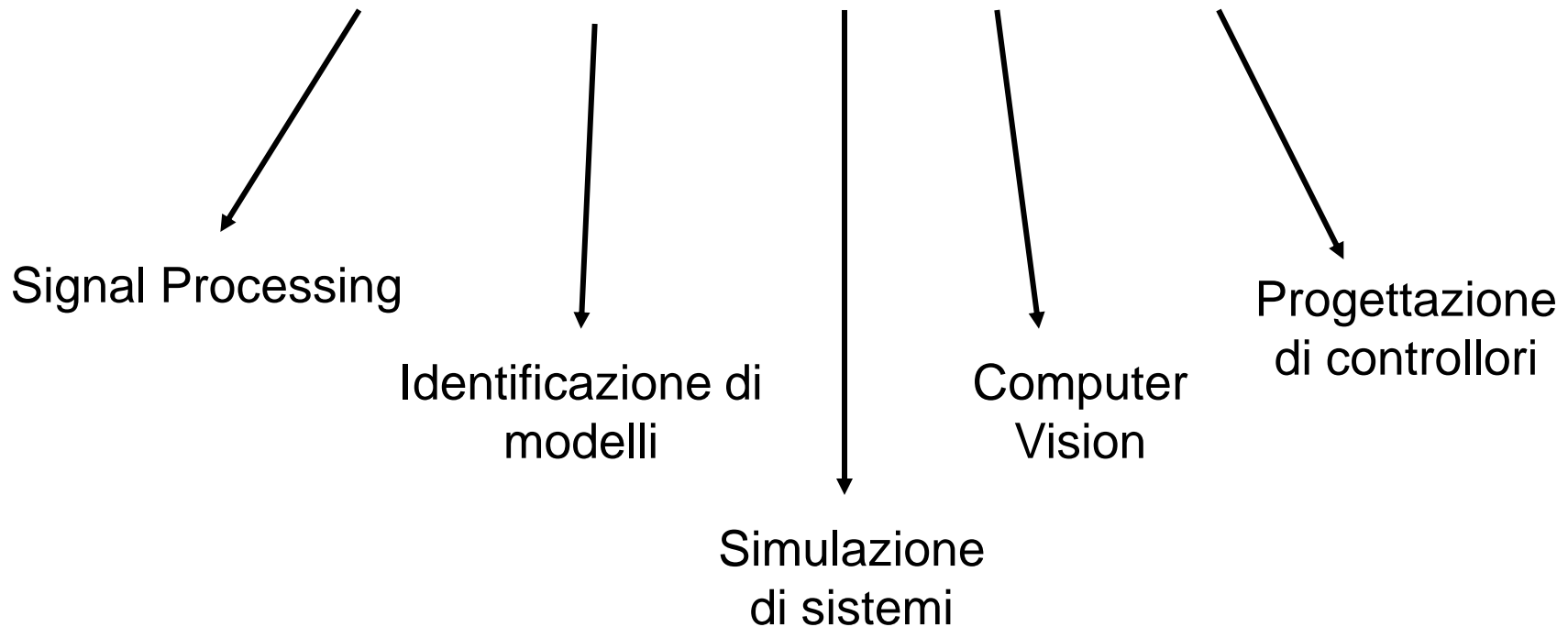


Che cos'è MatLab?

- Abbreviazione di MATrix LABoratory
- Originariamente realizzato esplicitamente per la manipolazione di matrici
- Linguaggio di programmazione ad alto livello

A cosa serve MatLab?

Calcolo matriciale e vettoriale

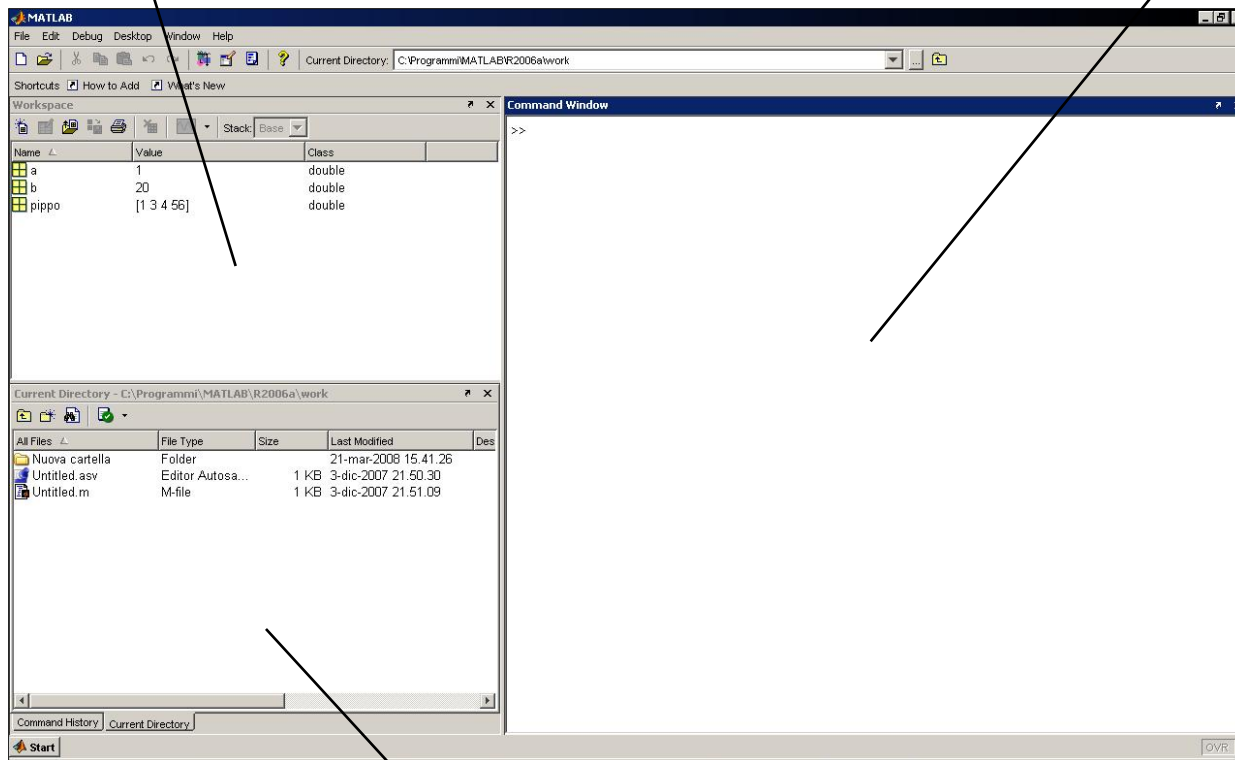


... e molte altre “discipline” ingegneristiche

Come appare MatLab

WorkSpace

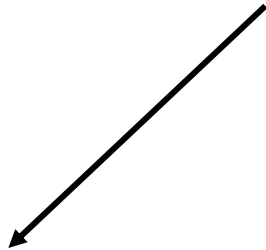
Command Window



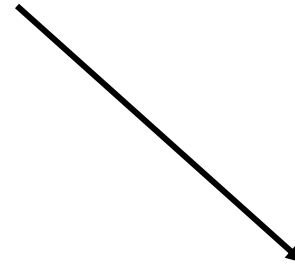
Directory di lavoro

Interagire con MatLab

Generalmente si inseriscono comandi



Sulla **Command Window**: se si chiude l'applicazione si perde ciò che si è scritto!



Su un **m-file**: si può salvare la sequenza di comandi!

Primi comandi

Definizione di una variabile scalare:

```
>> a = 3
```

Sulla Command Window appare

```
a =
```

```
3
```

Se scriviamo:

```
>> a = 3;
```

Non compare nulla sulla Command Window

Primi comandi (2)

Definizione del vettore riga $v = [1\ 2\ 5\ 7]$

```
>> v = [1 2 5 7];
```

Definizione della matrice $A = \begin{bmatrix} 1 & 3 & 6 \\ 4 & 0.4 & 12 \end{bmatrix}$

```
>> A = [1 3 6; 4 0.4 12];
```



Indica la fine di una riga

Primi comandi (3)

MatLab gestisce anche numeri complessi:

```
>> z = 3 + i*5;      o      >> z = 3 + j*5;
```

i e *j* sono già predefinite come $\sqrt{-1}$



Attenzione a non sovrascriverle!

```
>> r = real(z)
```

```
>> w = imag(z)
```



Parte reale e immaginaria
del numero *z*

```
>> m = abs(z)
```

```
>> p = phase(z)
```



Modulo e fase

Operazioni di base

- Somme tra vettori o matrici: $A = \begin{bmatrix} 1 & 2 \\ 5 & 6 \end{bmatrix}$ $B = \begin{bmatrix} 0 & 3 \\ 1 & 1 \end{bmatrix}$

$$>> C = A + B \longrightarrow C = \begin{bmatrix} 1 & 5 \\ 6 & 7 \end{bmatrix}$$

- Prodotto tra vettori e/o matrici:

$$>> C = A * B \longrightarrow C = \begin{bmatrix} 2 & 5 \\ 6 & 21 \end{bmatrix}$$

Attenzione alle dimensioni degli operandi

Operazioni di base (2)

- Prodotto componente per componente:

$$>> C = A .* B \longrightarrow C = \begin{bmatrix} 0 & 6 \\ 5 & 6 \end{bmatrix}$$

Vale anche per la divisione (./) e l'elevamento a potenza (.^)

- Inversione di una matrice (quadrata a rango pieno):

$$>> C = \text{inv}(A) \quad \text{o} \quad >> C = A^{-1}$$

- Potenza di una matrice (p scalare qualsiasi):

$$>> C = A^p$$

Operazioni di base (3)

- Trasposizione: `>> C = A'`

- Accostamento di matrici

$$\text{\>\> } C = [A \ B] \longrightarrow C = \begin{bmatrix} 1 & 2 & 0 & 3 \\ 5 & 6 & 1 & 1 \end{bmatrix}$$

- Se si scrive: `>> C = [A; B]` si ottiene incolonnamento
- Selezione di un elemento (prima riga, seconda col.):

$$\text{\>\> } c = A(1, 2) \longrightarrow c = 2$$

- Selezione di una riga intera

$$\text{\>\> } C = A(2, :)$$

- Selezione di una colonna intera

$$\text{\>\> } C = A(:, 1)$$

Operazioni di base (4)

$$v = [3 \quad 3 \quad 1 \quad 0 \quad 0.5 \quad 6.1 \quad 4 \quad 1]$$

- Selezione di una sottosequenza:

$$>> w = v(2:6) \longrightarrow w = [3 \quad 1 \quad 0 \quad 0.5 \quad 6.1]$$

$$>> w = v(1:2:7) \longrightarrow w = [3 \quad 1 \quad 0.5 \quad 4]$$

- Somma/prodotto degli elementi:

$$>> s = \text{sum}(v) \qquad >> p = \text{prod}(v)$$

- Valore massimo:

$$>> [m, \text{in}] = \text{max}(v) \longrightarrow \begin{array}{l} m = 6.1 \\ \text{in} = 6 \text{ (indice del v. max)} \end{array}$$

- Somma di vettore e scalare

$$>> s = v + 1; \longrightarrow s = [4 \quad 4 \quad 2 \quad 1 \quad 1.5 \quad 7.1 \quad 5 \quad 2]$$

Altri comandi

<code>>> t = 0:5</code>	→	$t = [0 \ 1 \ 2 \ 3 \ 4 \ 5]$
<code>>> t = 0:0.1:5</code>	→	$t = [0 \ 0.1 \ 0.2 \ \dots \ 5]$
<code>>> zeros(m,n)</code>	→	Matrice m x n di zeri
<code>>> ones(m,n)</code>	→	Matrice m x n di 1
<code>>> eye(n)</code>	→	Matrice identità di dim. n
<code>>> rank(A)</code>	→	Rango di una matrice
<code>>> eig(A)</code>	→	Autovalori
<code>>> find(t > n)</code>	→	Restituisce gli indici degli elementi maggiori di n (molto utile!)

Polinomi in MatLab

Un vettore v si può associare ad un polinomio in 2 modi:

- v contiene i coefficienti del polinomio

`>> v = [1 2 3];` $\longrightarrow p(z) = z^2 + 2z + 3$

`>> roots(v)` $\longrightarrow z_i = -1 \pm 1.41$

- v contiene le radici del polinomio

`>> v = [1 2 3];` $\longrightarrow p(z) = (z - 1)(z - 2)(z - 3)$

`>> poly(v)` $\longrightarrow p(z) = z^3 - 6z^2 + 11z - 6$

Gestione variabili

- Ogni variabile è memorizzata nel workspace
- Alla chiusura di MatLab, si perde il lavoro!
- Per ispezionare il workspace:

>> whos →

Lista variabili con descrizione

>> clear v →

Elimina la variabile v

>> clear all →

Elimina tutte le variabili

>> save n_file →

Salva il ws. nel file n_file.mat

>> save n_f v →

Salva in n_f.mat la var. v

m-files

- MatLab gestisce file di estensione *.m
- **Script**: sequenza di comandi, si avviano dal workspace

>> nome_file \longrightarrow Avvia il file nome_file.m

- **Funzione**: riceve un input, restituisce un output

>> [a, b, c] = fn(C,D)

Richiama la funzione fn, che riceve due parametri in ingresso e ne restituisce tre.

Esempio di script

```
A = eye(3);  
B = ones(3,3);  
C = A + B;  
disp('Calcolo autovalori di C');  
lambda = eig(C)  
save autov C lambda
```

Esempio di funzione

← Variabili output

```
[C, lambda] = nome_funzione(A, B)
```

← Variabili in input

```
C = A + B;  
lambda = eig(C);
```

- Il file .m della funzione deve avere il nome della stessa
- Non è necessario dichiarare la tipologia di input/output

Istruzioni per m-files

- Istruzione con condizione

Condizioni “booleane”

```
if [condizione]
    [istruzioni]
else
    [istruzioni]
end
```



Es: $a==4$ (uguaglianza)

$a\neq 2$ (diverso da)

$a\geq 2$ (magg. eguale)

`isempty(a)` (a vett. vuoto)

- Cicli for e while

```
for n=1:100
    [istruzioni]
end
```

```
while [condizione]
    [istruzioni]
end
```

Esempi di funzioni MatLab

<code>>> sin(t)</code>	—————→	Seno nei valori del vettore t
<code>>> cos(t)</code>	—————→	Coseno
<code>>> exp(t)</code>	—————→	Esponenziale
<code>>> log(t)</code>	—————→	Logaritmo

- Consultare l'help di MatLab per scoprire altre funzioni
- Digitare `help nome_funzione` per informazioni dettagliate su una data funzione

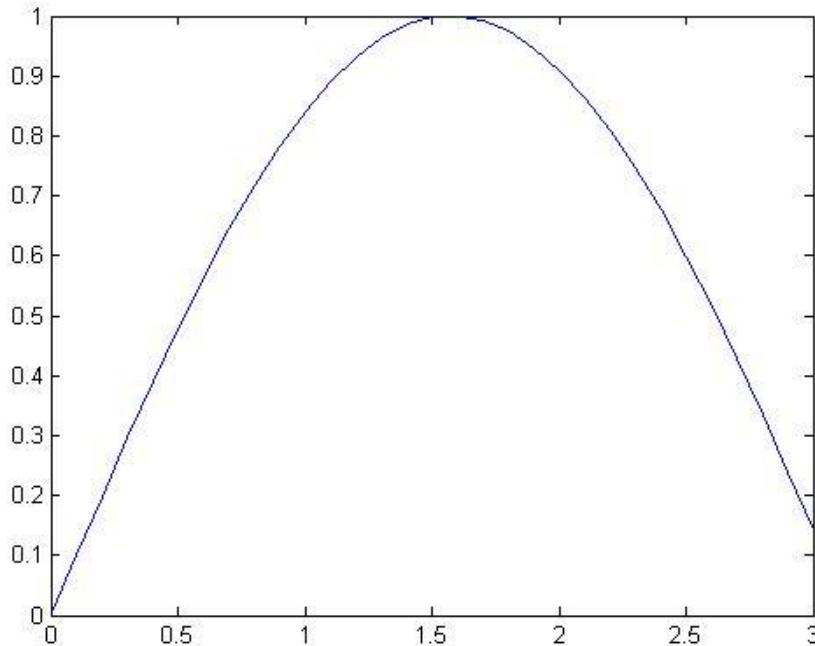
Creazione di grafici

- In MatLab sono implementate potenti funzioni grafiche
- Esempio: funzione `plot`

```
>> t = 0:0.1:3;
```

```
>> plot(t, sin(t))
```

Disegna la funzione seno
nell'intervallo $[0,3]$, con periodo
di campionamento 0.1

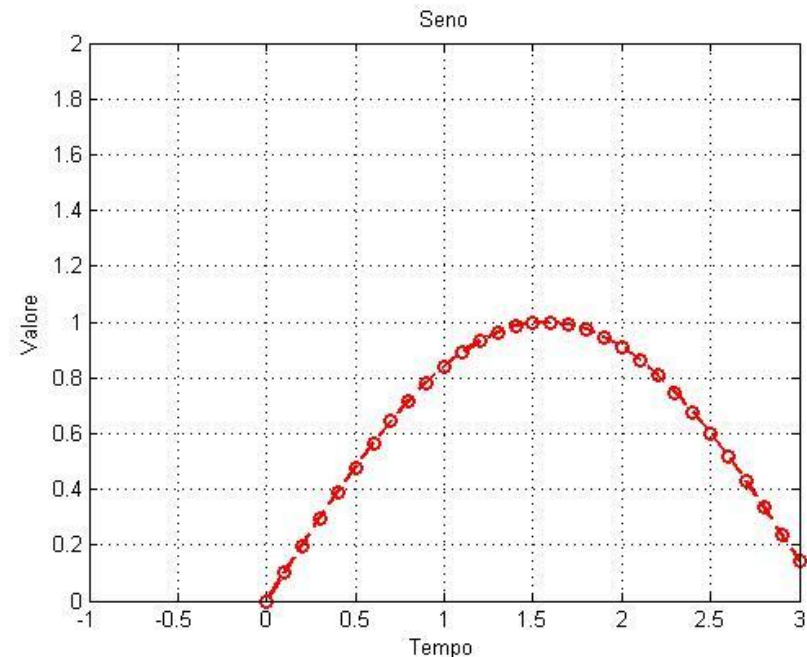


Creazione di grafici (2)

- E' possibile modificare il grafico con semplici comandi

```
>> plot(t,sin(t),'r--o','Linewidth',2)
>> grid on
>> axis([-1 3 0 2])
>> title('Seno')
>> xlabel('Tempo')
>> ylabel('Valore')
```

Output →



Creazione di grafici (3)

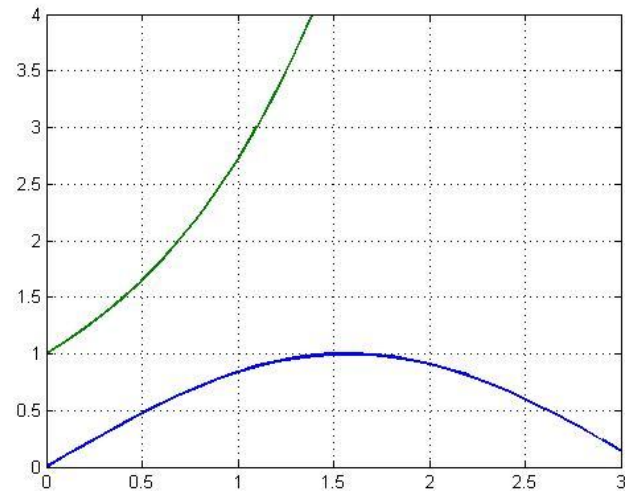
- E' possibile disegnare due grafici in un'unica figura

- Primo metodo

```
>> plot(t,sin(t))
```

```
>> hold on
```

```
>> plot(t,exp(t))
```



- Secondo metodo (cambia i colori automaticamente)

```
>> plot(t,sin(t),t,exp(t))
```