Introduzione all'ambiente MATLAB

Parte I

Matlab - Licenza Campus

In seguito ad un accordo con MathWorks l'Ateneo ha attivato la licenza Campus di MATLAB, grazie alla quale tutti i docenti, i ricercatori, il personale tecnico-amministrativo e gli studenti possono installare gli applicativi MATLAB e Simulink (comprensivi di 50 toolbox/librerie) sui propri computer, oltre che seguire gratuitamente corsi di formazione online.

Codici di attivazione

http://www.unibo.it/it/servizi-e-opportunita/studio-e-non-solo/agevolazioni-per-computer-tablet-e-software/matlab-licenza-campus/matlab-licenza-campus

Matlab per studenti

(accesso con credenziali istituzionali @studio.unibo.it)

Tutorial

Sito ufficiale di MATHWORKS:

http://www.mathworks.it/help/index.html

Tutorial in italiano

http://guide.supereva.it/manuali/matlab

Tutorial in inglese: MATLAB primer

http://math.ucsd.edu/~driver/21d-s99/matlab-primer.html

Overview

- MATLAB e le sue potenzialità
  - Funzionalità di MATLAB
  - Ambiente di sviluppo MATLAB
- Il manuale o Help
- Le variabili e lo spazio di lavoro
- Operatori MATLAB
  - Operatori di base, operatori logici, operatori relazionali
  - Vettori e Matrici

- Operazioni tra matrici
  - Matrici speciali
- Operatori su elementi
- Operazioni aritmetiche su vettori-matrici
  - Variabili complesse
- Grafica in MATLAB

Ci sono software gratuiti open source alternativi a MATLAB, in particulare ***GNU Octave, FreeMat, e Scilab che sono compatibili*** *con MATLAB (ma non nell'ambiente MATLAB desktop).*

MATLAB (MATrix LABoratory) è un ambiente interattivo ad alto livello che consente di costruire e gestire facilmente matrici e, come casi particolari, vettori e scalari.

La struttura dati di base è la matrice: ciò significa che durante l'elaborazione ogni quantità viene trattata dall'ambiente di calcolo come una matrice di dimensione **nxm**.
Un vettore è una matrice **1xn**, uno scalare è gestito come una matrice **1x1**

Il pacchetto però non è utilizzato solo per gestire matrici, ma è un'ottima piattaforma di sperimentazione e verifica per il calcolo numerico in genere.

Il calcolo simbolico in MATLAB è basato sul software Maple.

Funzionalità di MATLAB

MATLAB fornisce un ambiente di calcolo, visualizzazione e programmazione scientifica, in cui è possibile:

- **calcolare direttamente espressioni matematiche;**

```
>> ((tan(pi/5)+2)*exp(2.3)-0.01)/log(2)
ans =
    39.2197
```

- **utilizzare il semplice ambiente di programmazione per costruire i propri algoritmi** (Parte II);

- **sfruttare algoritmi di base già implementati**
  - *built-in function* -

```
>> mean([1.5 2.5 3.5])
ans =
   2.5000
```

Built-in functions

Tabella 5. Alcune funzioni predefinite in MATLAB.

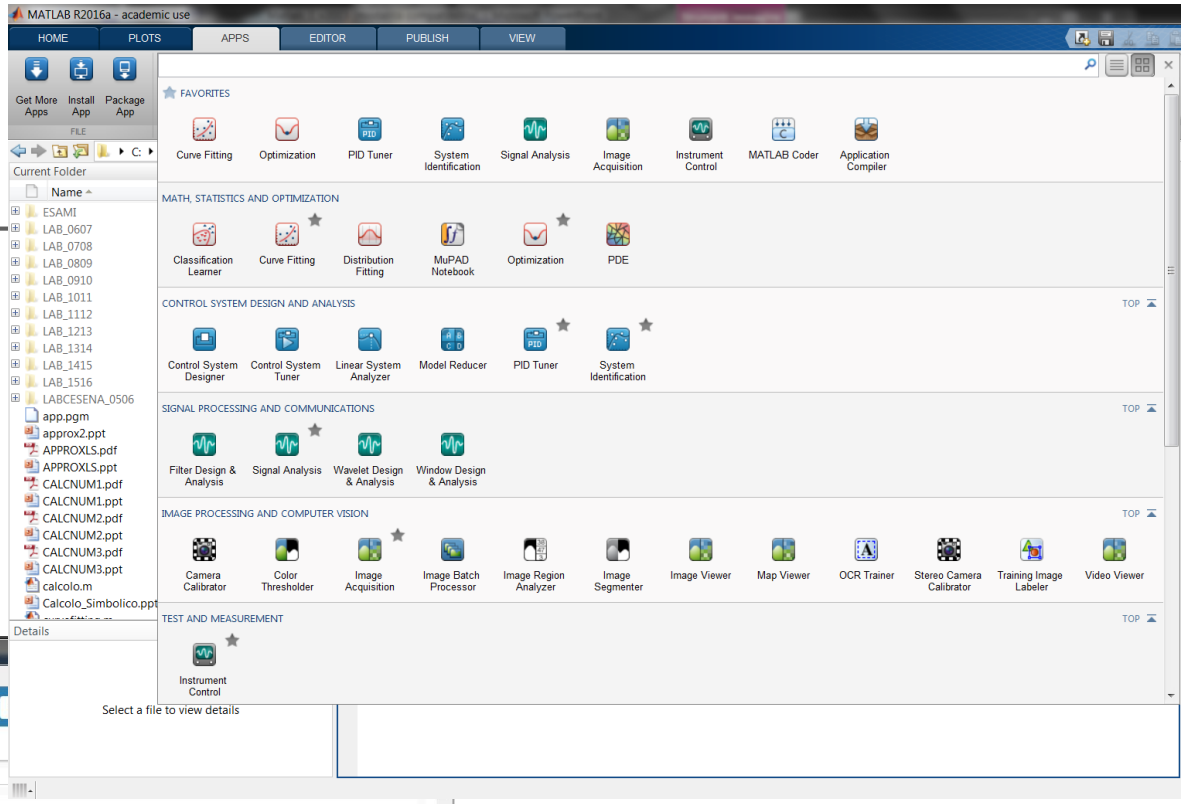
Funzione	Significato
`sin`	seno
`cos`	coseno
`asin`	arcoseno
`acos`	arcocoseno
`tan`	tangente
`atan`	arcotangente
`exp`	esponenziale
`log`	logaritmo naturale
`log2`	logaritmo in base 2
`log10`	logaritmo in base 10
`sqrt`	radice quadrata
`abs`	valore assoluto o modulo
`real`	parte reale
`imag`	parte immaginaria
`sign`	funzione segno
`factorial`	fattoriale
`round`	arrotonda all'intero più vicino
`floor`	arrotonda per difetto all'intero più vicino
`ceil`	arrotonda per eccesso all'intero più vicino
`chop(x,t)`	arrotonda `x` a `t` cifre significative

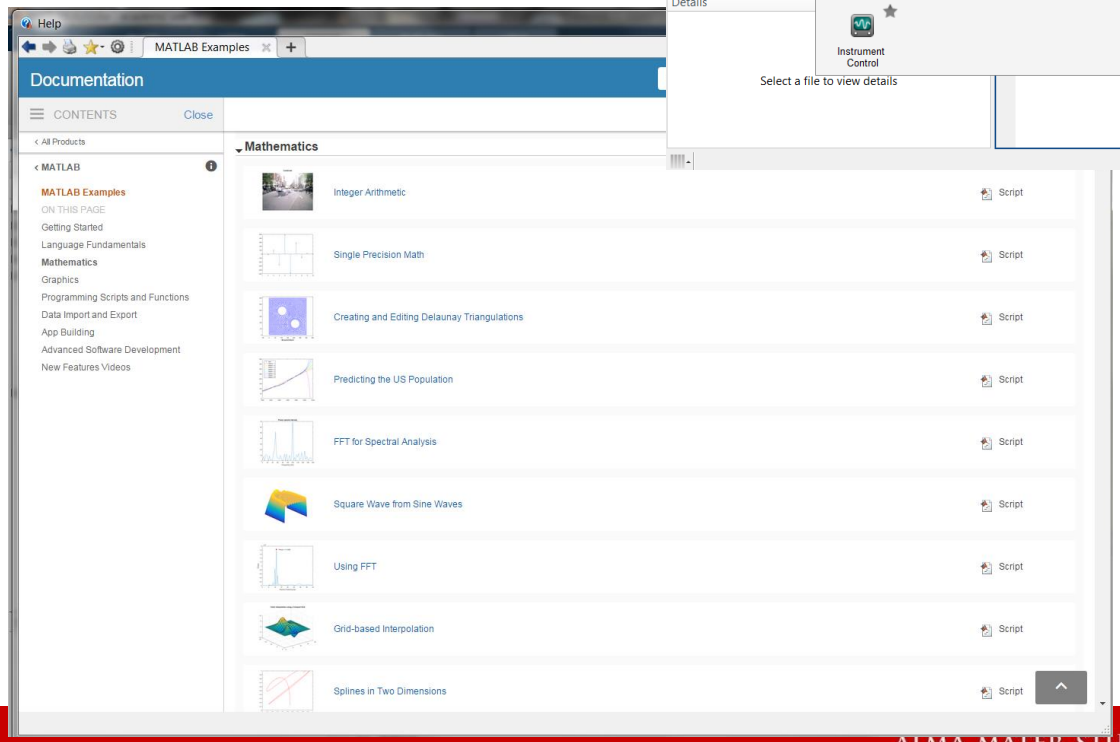
Altre funzionalità di MATLAB

- Matematica e calcolo
- Sviluppo di procedure e applicazioni
- Modellistica, simulazione e prototipizzazione
- Analisi di dati, esplorazione e visualizzazione
- Disegno industriale e scientifico
- Costruzione di interfacce utente
- TOOLBOX vari

Con il comando **demos** è possibile vederne alcuni esempi.

toolbox





Examples: demos

MATLAB ha anche un linguaggio proprio per programmare.

E' un linguaggio interpretato e non compilato:

questo significa che le istruzioni vengono tradotte in linguaggio macchina (il linguaggio "capito" dal processore) e subito eseguite una per volta.

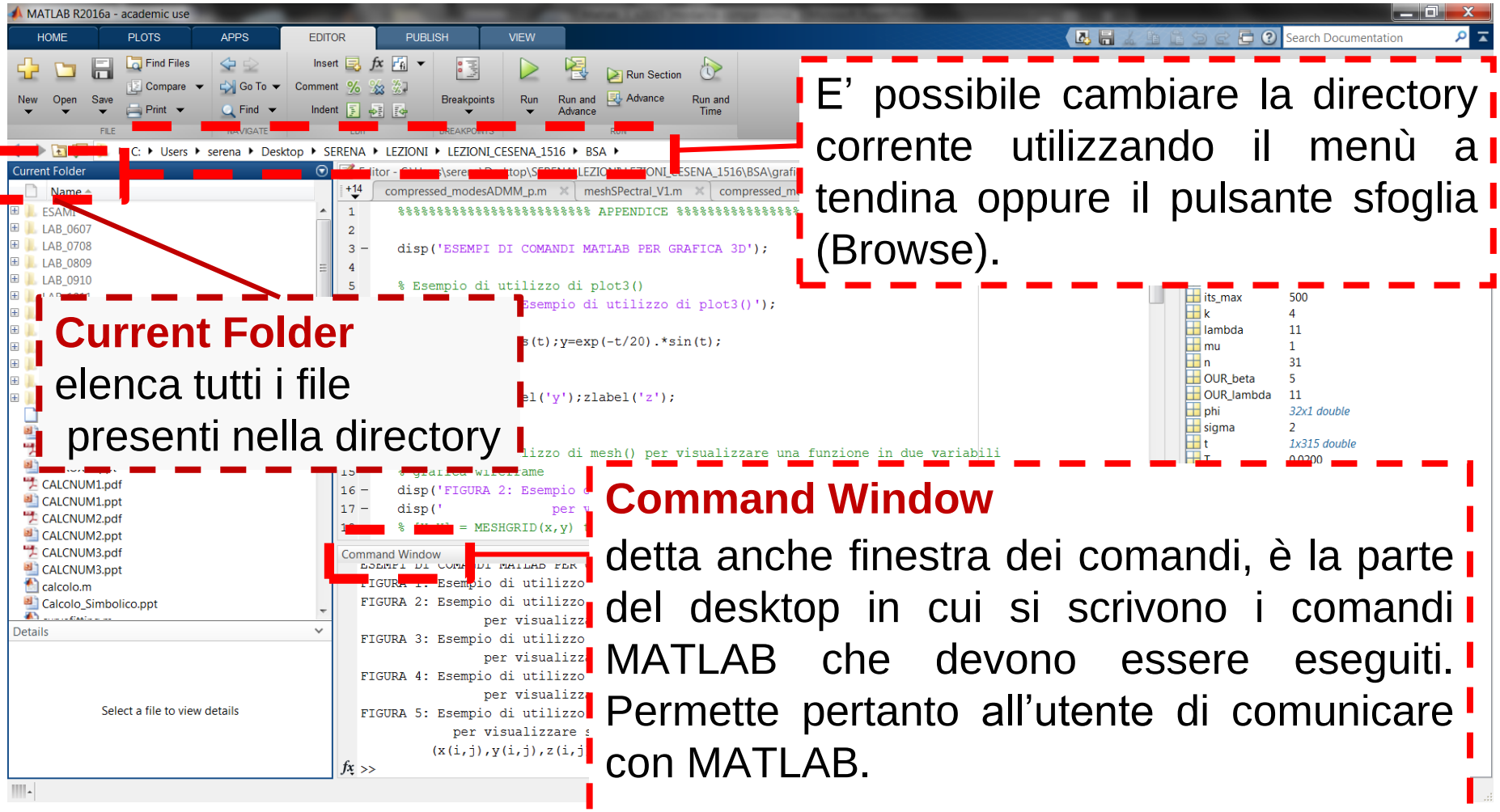
Utilizzando C, Fortran, C++, la traduzione da linguaggio ad alto livello a linguaggio macchina avviene invece nel processo di compilazione, in cui tutto il programma viene tradotto in linguaggio macchina e poi eseguito.

Ambiente di sviluppo MATLAB

- Per lanciare MATLAB da ambiente Windows basta cliccare due volte con il mouse sull'icona corrispondente.
- La finestra che appare quando si esegue MATLAB viene chiamata desktop.
- Il simbolo prompt >> indica che il calcolatore è pronto a ricevere le istruzioni e ad eseguirle.
- Per uscire dall'ambiente basta digitare

>> **quit**

Interfaccia grafica (per gestire files, variabili e programmi)



E' possibile cambiare la directory corrente utilizzando il menù a tendina oppure il pulsante sfoglia (Browse).

Current Folder

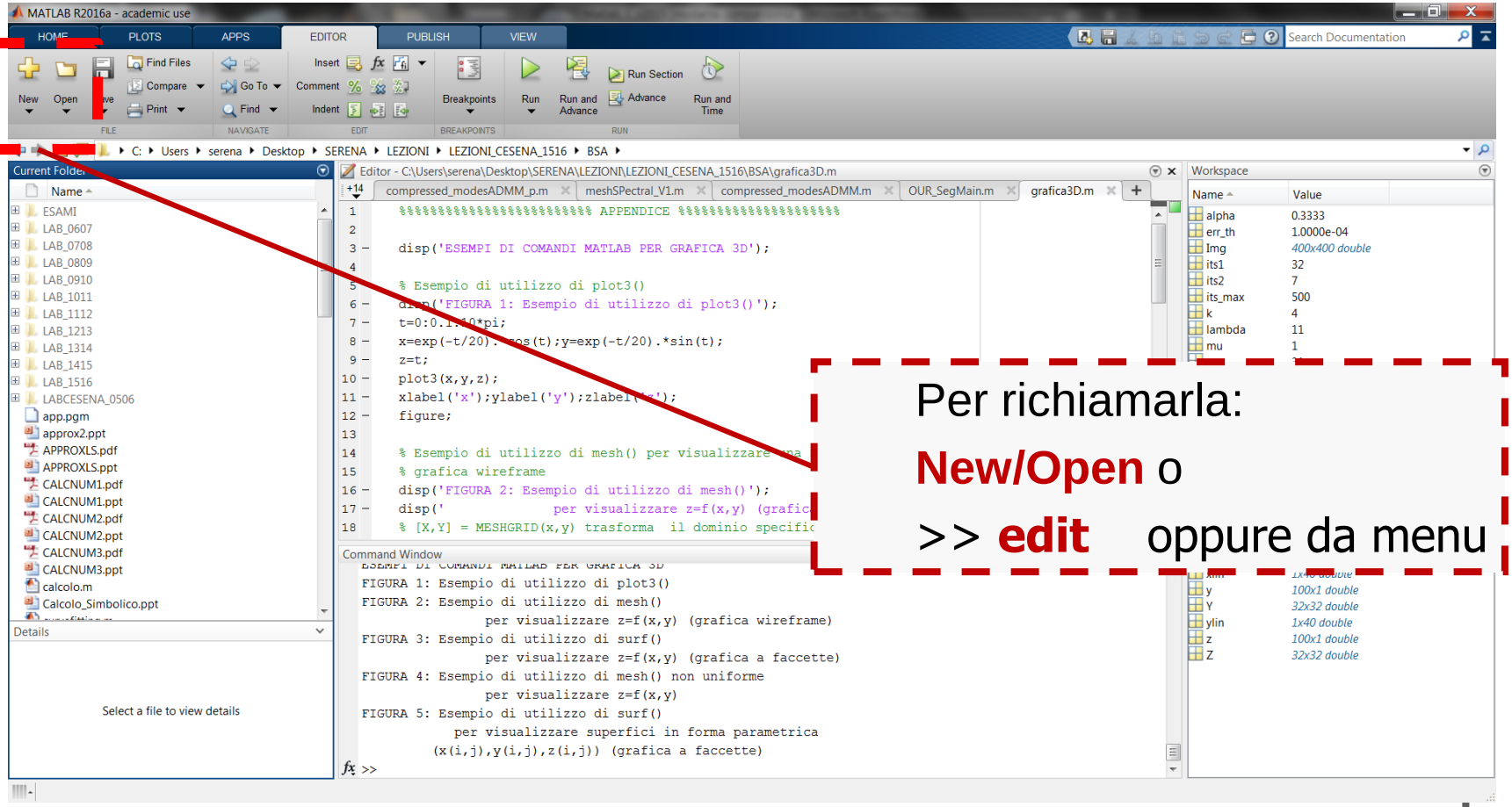
elenca tutti i file presenti nella directory

Command Window

detta anche finestra dei comandi, è la parte del desktop in cui si scrivono i comandi MATLAB che devono essere eseguiti. Permette pertanto all'utente di comunicare con MATLAB.

Editor

L'editor è la finestra in cui si scrivono i programmi MATLAB (M-files, cioè file con estensione ".m").



Per richiamarla:

New/Open o

>> **edit** oppure da menu

Il manuale (o help)

- Si può accedere al manuale o tramite il menu

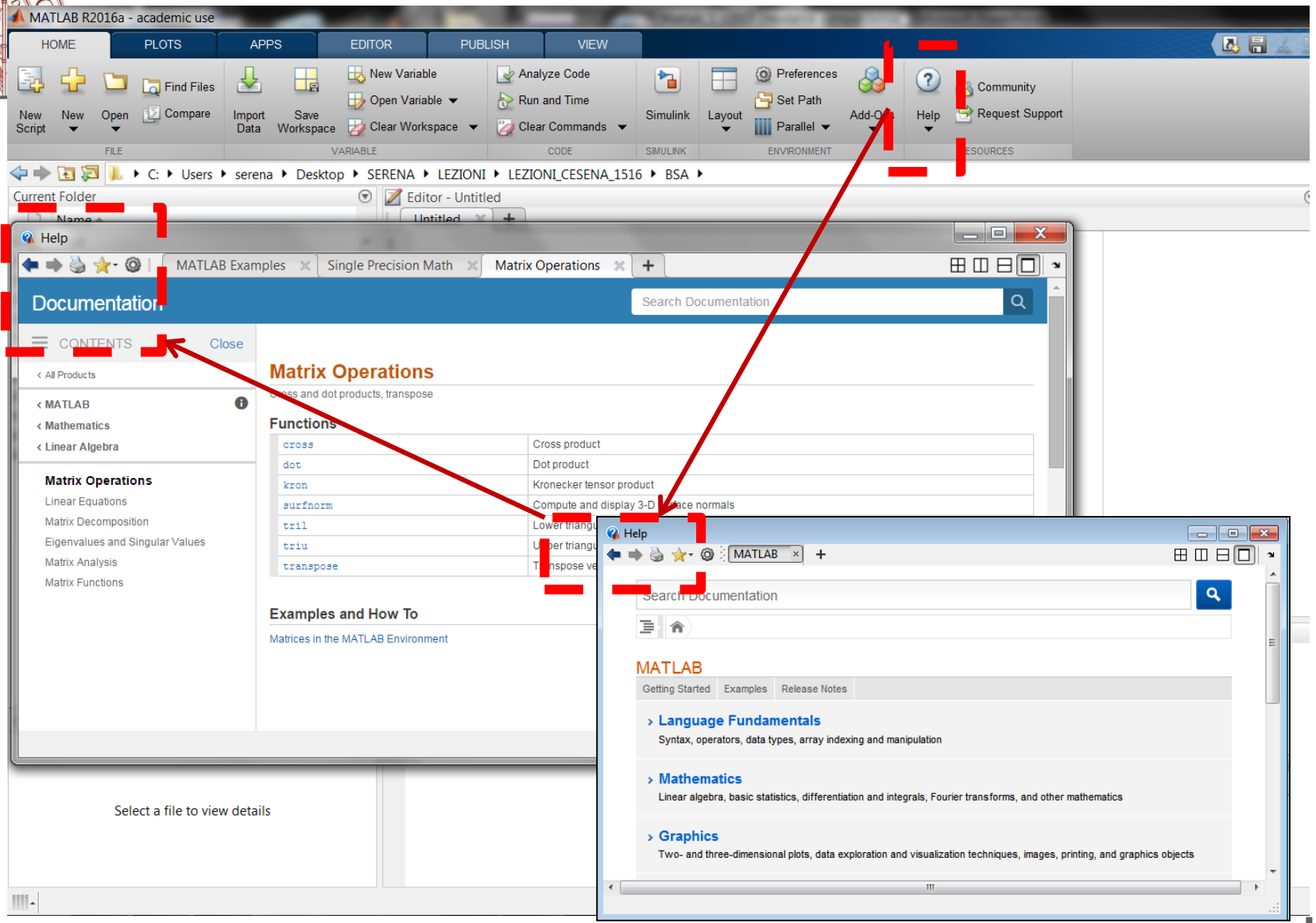
  Help → MATLAB Help (guida in linea)

  o digitando **help** dalla Command Window (help generale di tutte le funzioni).

- Per visualizzare l'help di un singolo comando digitare **help <nomecomando>** dalla Command Window (esempio **help plot**).

- Digitando **lookfor <keyword>** si attiva invece la ricerca di funzioni basate su una parola chiave.

- Attraverso il comando **doc** si accede direttamente alla documentazione online di MATLAB.


MATLAB R2016a - academic use
HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
New Script
New
Open
Find Files
Compare
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
Analyze Code
Run and Time
Clear Commands
Simulink
Layout
Preferences
Set Path
Parallel
Add-Ons
Help
Community
Request Support
FILE
VARIABLE
CODE
SIMULINK
ENVIRONMENT
RESOURCES
C: ▸ Users ▸ serena ▸ Desktop ▸ SERENA ▸ LEZIONI ▸ LEZIONI_CESENA_1516 ▸ BSA ▸
Current Folder
Editor - Untitled
Help
MATLAB Examples
Single Precision Math
Matrix Operations
Documentation
Search Documentation
CONTENTS
Close
All Products
MATLAB
Mathematics
Linear Algebra
Matrix Operations
Linear Equations
Matrix Decomposition
Eigenvalues and Singular Values
Matrix Analysis
Matrix Functions
Matrix Operations
Functions
cross
Cross product
dot
Dot product
kron
Kronecker tensor product
surfnorm
tril
triu
transpose
Examples and How To
Matrices in the MATLAB Environment
Select a file to view details
Help
MATLAB
Search Documentation
MATLAB
Getting Started
Examples
Release Notes
Language Fundamentals
Syntax, operators, data types, array indexing and manipulation
Mathematics
Linear algebra, basic statistics, differentiation and integrals, Fourier transforms, and other mathematics
Graphics
Two- and three-dimensional plots, data exploration and visualization techniques, images, printing, and graphics objects

Le variabili

Variabile: nome associato ad una entità (scalare, vettore, matrice) che contiene dati.

I nomi scelti (meglio se legati all'entità che rappresentano) devono rispettare le seguenti regole di sintassi:

1. possono contenere solo lettere, cifre e il carattere di sottolineatura ("_");
2. non possono iniziare con una cifra;
3. non si possono utilizzare parole riservate di MATLAB.

- Matlab è un linguaggio case sensitive, ossia distingue fra lettere maiuscole e minuscole: la variabile **A** è quindi diversa dalla variabile **a**.

- La variabile utilizzata da MATLAB in default è ***ans***.

Esempio

Introdurre 4 variabili e farne la media.

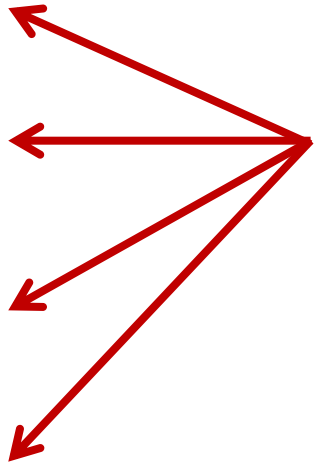
```
>> a=10;

>> b=20;

>> c=30;

>> d=40;

>> media=(a+b+c+d)/4
media =
    25
```



Le variabili **a**,**b**,**c**,**d** contengono rispettivamente i valori 10,20,30,40

Il valore della media aritmetica tra 10,20,30,40 è stato memorizzato nella variabile **media**

Tipi di variabili

Tipo	Tipo dato	Occupazione di memoria
Double	Numeri reali nell'intervallo $[10^{-37},10^{37}]$	8 byte
Complex double	Numeri complessi	16 byte
Logical double	Risultato di una operazione logica (1=vero, 0=falso)	8 byte
Char	Carattere	2 byte

Esempi

a = sqrt(3) → Double

c = -2+i*8.2 → Complex double

t = a<1 → Logical double

q = 'k' → Char

var_testo = 'questa stringa viene assegnata alla variabile var_testo' → **Char array**

Formato output

L'output può essere visualizzato in diversi modi, pur non influendo sul formato che MATLAB usa per memorizzare ed elaborare i dati (double precision).

SHORT	(default) Virgola fissa 5 cifre
SHORT E	Virgola mobile 5 cifre
SHORT G	Meglio tra virgola fissa e mobile 5 cifre
LONG	Virgola fissa 15 cifre
LONG E	Virgola mobile 15 cifre
LONG G	Meglio tra virgola fissa e mobile 15 cifre
RAT	Approssimazione mediante il rapporto ridotto ai minimi termini

Esempio

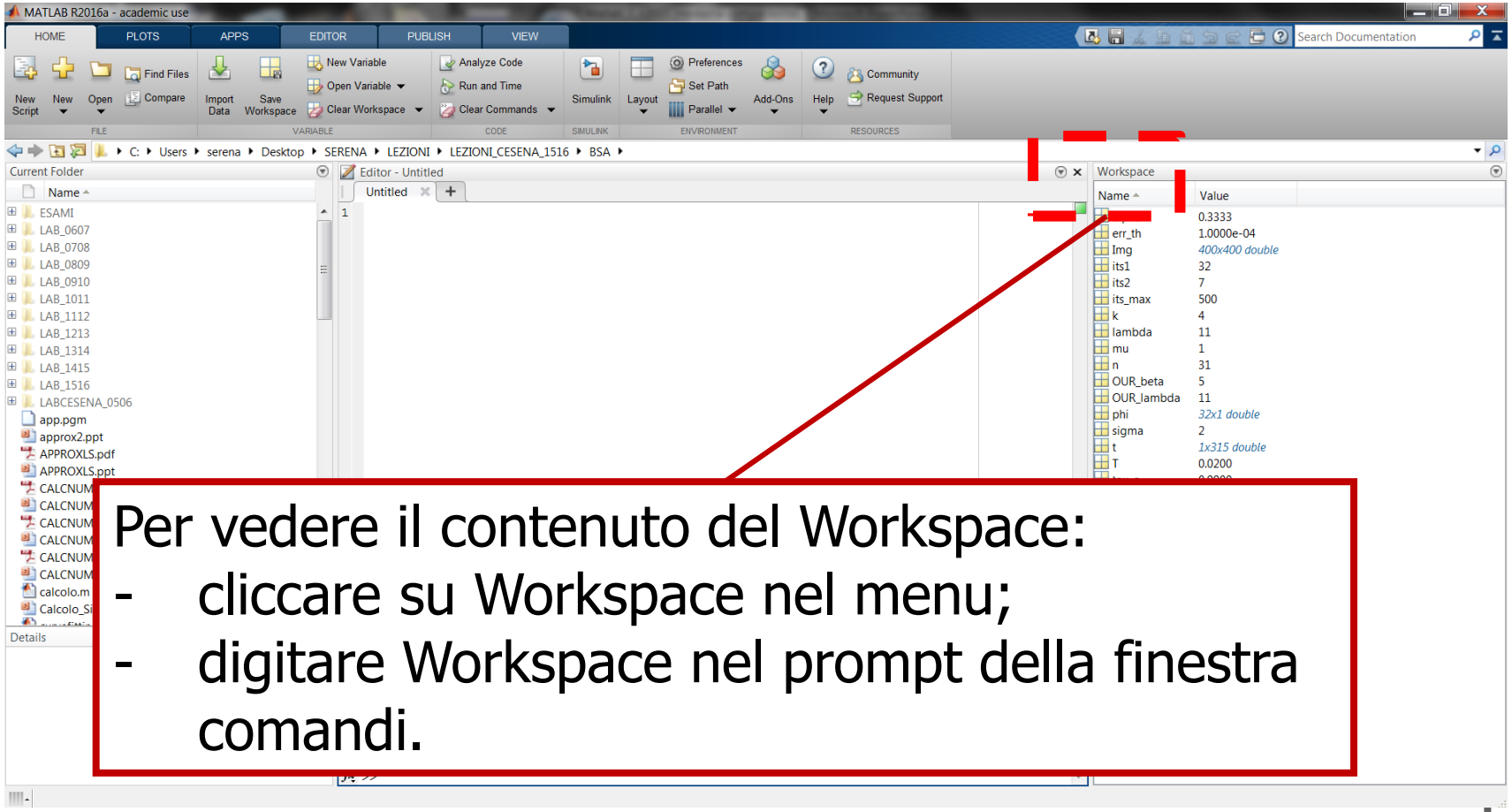
```
>>y = 8/6

>>format short
        1.3333
>>format short e
        1.3333E+000
>>format short g
        1.3333
>>format long
        1.33333333333333
>>format long e
        1.33333333333333E+000
>>format long g
        1.33333333333333
>>format rat
        4/3
```

default : format short

Workspace: lo spazio di lavoro

E' l'insieme delle variabili mantenute in memoria durante una sessione MATLAB.

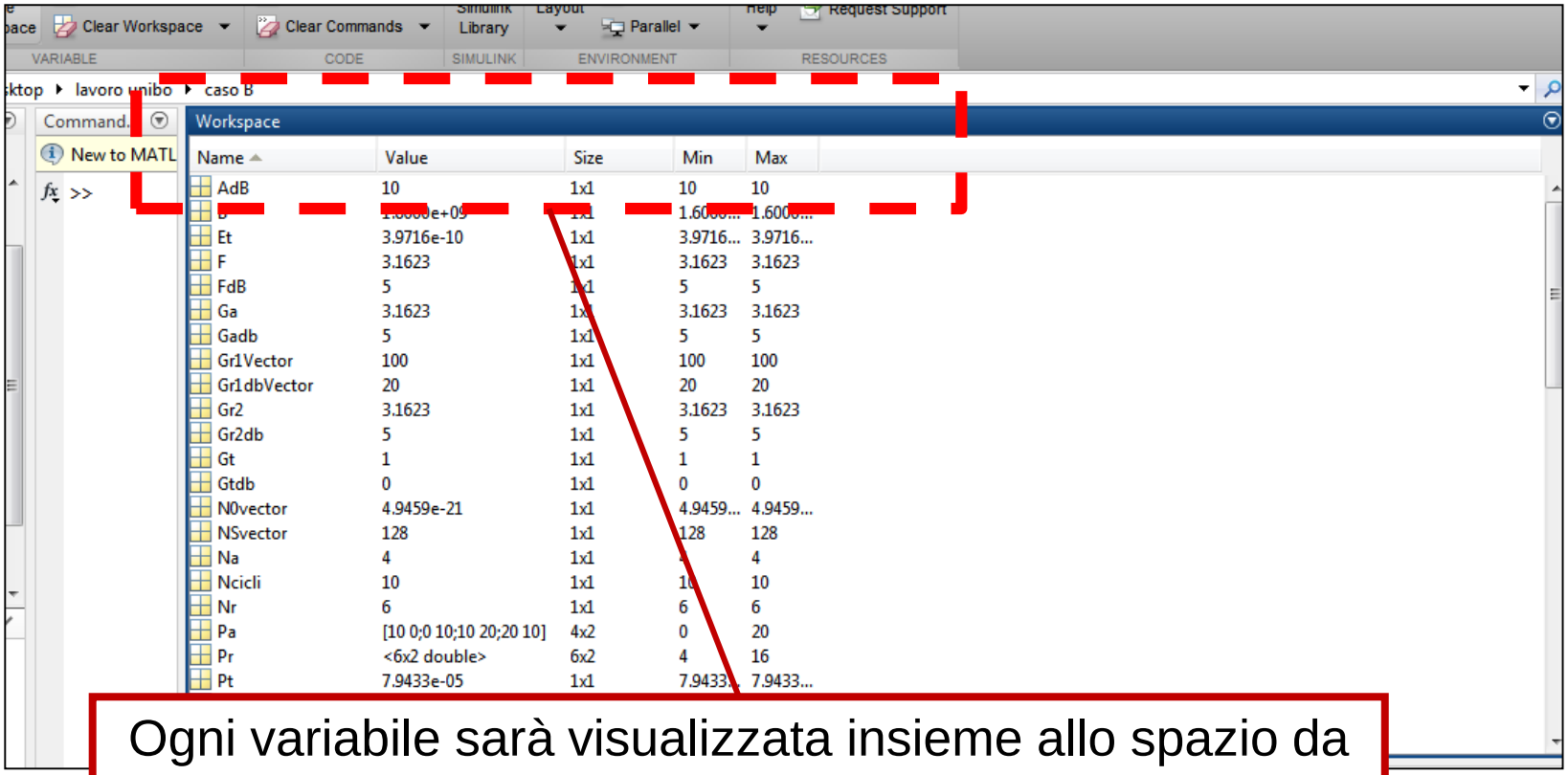


Per vedere il contenuto del Workspace:

- cliccare su Workspace nel menu;
- digitare Workspace nel prompt della finestra comandi.

Workspace

Informazioni per ogni variabile attiva:



Ogni variabile sarà visualizzata insieme allo spazio da essa occupato, al numero di elementi, e al suo tipo.

Con il workspace si può:

1. visualizzare e modificare le variabili dell'area di lavoro (eventualmente cambiandone il formato output);
2. cancellare le variabili dell'area di lavoro;
3. rappresentare graficamente le variabili dell'area di lavoro;
4. salvare l'area di lavoro;
5. caricare un'area di lavoro precedentemente salvata.

La sessione di lavoro

Il comando

diary <nome file>

memorizza nel file ASCII < nomefile > la sessione di lavoro (comandi dati e workspace) da quel punto in poi in modo da poterla poi consultare con un qualsiasi editor (es. WORD).

Non possiamo però utilizzare il file < nomefile > per ricaricare il lavoro fatto e continuare a lavorarci; per questo si devono utilizzare i comandi save e load.

La sessione di lavoro

Ogni sessione di lavoro può essere salvata in un file binario (nomefile.mat) mediante il comando

- **save** <nome file>

e ricaricata in ambiente MATLAB mediante

- **load** <nome file>

Le variabili utilizzate in ogni workspace possono essere consultate semplicemente digitando il nome della variabile stessa.

La sessione di lavoro

- Per cancellare la variabile *nomevariabile*

  clear *<nomevariabile>*

- Per cancellare tutte le variabili di una sessione di lavoro

  clear

- Per pulire il desktop di MATLAB

  clc

- Per pulire la finestra di una figura

  clf

- Per avere un elenco degli m-files memorizzati

  what

La sessione di lavoro

- **pwd** visualizza la directory in cui si sta lavorando
- **dir**, o **ls** elenca i file presenti nella directory
- **cd** modifica la directory attuale
- **path**, o **matlabpath** elenca i possibili percorsi delle directory in MATLAB
- **addpath** aggiunge la directory ai percorsi già esistenti
- **pathtool** permette di accedere alla finestra degli strumenti per la gestione dei percorsi

Operatori Matlab

Operatori di base:

+	addizione
-	sottrazione
/	divisione a destra
\	divisione a sinistra
^	elevamento a potenza
*	moltiplicazione

1/4=4\1=0.25

Operatori logici:

&	and
|	or
~	not

Operatori relazionali:

~ =	diverso da
<=	minore uguale a
<	minore a
>=	maggiore uguale a
>	maggiore a
= =	uguale logico

(per ~ in una tastiera italiana premere ALT e digitare 126 nel tastierino numerico)

Scalari, Vettori e matrici

Una matrice ha dimensione **nxm**
(n righe, m colonne).
Un vettore ha dimensione **nx1**
(vettore colonna) o **1xn** (vettore riga).
Uno scalare ha dimensione **1x1**.

- **a = 1** (scalare, ovvero matrice 1x1)
- **a = [0 1 2 3 4]**, **a = [0,1,2,3,4]** ed anche **a = [0:4]**
- **b = [0, .5, 1, 1.5, 2, 2.5]** ed anche **b = [0:.5:2.5]**
- **c = [.1, .1, .1, .1, .1]** ed anche **c = ones(1,5) * 0.1**

Costruire Vettori

```
» a = [1 2 3 4]
a =
    1    2    3    4
```

Definisce un vettore RIGA "a" (le parentesi quadre indicano un vettore o matrice)

```
» size(a)
ans =
    1    4
```

fornisce la dimensione di "a"

```
» length(a)
ans =
    4
```

usato per i vettori indica la loro lunghezza

Definire un vettore COLONNA

```
» a'
ans =
    1
    2
    3
    4
```

```
» [1;2;3;4]
ans =
    1
    2
    3
    4
```

```
» size(a')
ans =
    4    1
```

```
» length(a')
ans =
    4
```

trasposta di "a" (ha dimensioni "invertite" rispetto ad "a")

Tabella 6. Alcuni comandi per generare e manipolare vettori.

Comando	Azione
`x'`	genera il vettore trasposto di `x`
`x=[]`	genera il vettore vuoto `x`
`sort(x)`	riordina in ordine crescente le componenti del vettore `x`
`x=[a:h:b]`	genera il vettore riga $x = (x_i)_{i=1,\ldots,m+1}$ ove $x_i = a + (i-1)h$ e m è la parte intera di $(b-a)/h$
`x=linspace(a,b,n)`	genera il vettore riga $x = (x_i)_{i=1,\ldots,n}$ ove $x_i = a + (i-1)h$ e $h = (b-a)/(n-1)$
`x=logspace(a,b,n)`	genera il vettore riga $x = (10^{x_i})_{i=1,\ldots,n}$ ove $x_i = a + (i-1)h$ e $h = (b-a)/(n-1)$
`x(r)`	estrae le componenti del vettore `x` i cui indici sono specificati in `r`
`x(r)=z`	assegna alle componenti del vettore `x` (i cui indici sono specificati in `r`) i valori definiti in z rispettivamente
`x(r)=[]`	rimuove le componenti del vettore `x` (i cui indici sono specificati in `r`)
`x([i j])=x([j i])`	scambia le componenti i e j del vettore `x`

Tabella 7. Alcune funzioni predefinite in MATLAB agenti su un vettore x.

Comando	Azione
`a=sum(x)`	genera lo scalare $a = \sum_{i=1}^{n} x_i$
`a=prod(x)`	genera lo scalare $a = \prod_{i=1}^{n} x_i$
`a=max(x)`	genera lo scalare $a = \max_i x_i$
`a=min(x)`	genera lo scalare $a = \min_i x_i$
`a=norm(x)`	genera lo scalare $a = \|\|x\|\|_2$
`a=norm(x,1)`	genera lo scalare $a = \|\|x\|\|_1$
`a=norm(x,inf)`	genera lo scalare $a = \|\|x\|\|_\infty$
`A=diag(x)`	genera la matrice diagonale $A = (a_{ij})_{i,j=1,\ldots,n}$, con $a_{ii} = x_i$

Costruire Matrici

```
» c =[1 2 3 4 ; 5 6 7 8 ]
c =
    1    2    3    4
    5    6    7    8
```

- Per fare riferimento ad un elemento della matrice “c”:

```
» c(1,1)
ans =
1
```

```
» c(2,3)
ans =
7
```

- **Usare “:” per indicare tutte le righe o tutte le colonne, esempio:**
  - c(1,:) indica la prima riga, tutte le colonne
  - c(:,2) indica tutte le righe, la seconda colonna
  - c(:,2:4) indica tutte le righe, dalla seconda alla quarta colonna

```
» c(1,:)
ans =
1    2    3    4
```

```
» c(:,2)
ans =
2
6
```

```
» c(:,2:4)
ans =
2    3    4
6    7    8
```

Funzioni di matrici

```
» c =[1 2 3 4 ; 5 6 7 8 ]
c =
    1    2    3    4
    5    6    7    8
```

```
» sum(c)
ans =
     6     8    10    12
```

```
» sum(c')
ans =
    10    26
```

```
» sum(sum(c))
ans =
    36
```

La funzione *sum* calcola la somma degli elementi di una matrice per colonne; il risultato è un vettore.

Se la matrice è un vettore 1 x m (come sum(c)) , allora la somma è calcolata sugli elementi del vettore.

```
» mean(c)
ans =
     3     4     5     6
```

```
» max(c)
ans =
     5     6     7     8
```

```
» min(c)
ans =
     1     2     3     4
```

mean fornisce la media per colonne; max e min il massimo e il minimo ancora per colonne.

Tabella 9. Alcune funzioni predefinite in MATLAB agenti su una matrice `A`.

Comando	Azione
`a=norm(A)`	genera lo scalare $a = \|\|A\|\|_2$
`a=norm(A,1)`	genera lo scalare $a = \|\|A\|\|_1$
`a=norm(A,inf)`	genera lo scalare $a = \|\|A\|\|_\infty$
`x=sum(A)`	genera il vettore riga $x = (x_j)_{j=1,\ldots,n}$, con $x_j = \sum_{i=1}^{n} a_{ij}$
`x=max(A)`	genera il vettore riga $x = (x_j)_{j=1,\ldots,n}$, con $x_j = \max_i a_{ij}$
`x=min(A)`	genera il vettore riga $x = (x_j)_{j=1,\ldots,n}$, con $x_j = \min_i a_{ij}$
`x=diag(A)`	genera il vettore colonna $x = (x_i)_{i=1,\ldots,n}$, con $x_i = a_{ii}$
`B=abs(A)`	genera la matrice $B = (b_{ij})_{i,j=1,\ldots,n}$, con $b_{ij} = \|a_{ij}\|$
`B=tril(A)`	genera la matrice triangolare inferiore $B = (b_{ij})_{i,j=1,\ldots,n}$, con $b_{ij} = a_{ij}$, $i = 1, \ldots, n$, $1 \le j \le i$
`B=triu(A)`	genera la matrice triangolare superiore $B = (b_{ij})_{i,j=1,\ldots,n}$, con $b_{ij} = a_{ij}$, $i = 1, \ldots, n$, $i \le j \le n$

Concatenazione di matrici

» A = [2,0; 0,1; 3,3]

```
A =
     2     0
     0     1
     3     3
```

» A(1:2,1:2)

```
ans =
     2     0
     0     1
```

» A =[A,[1;2;3]]

```
A =
     2     0     1
     0     1     2
     3     3     3
```

Equivale a **cat(2,A,[1;2;3])**

» A =[A;[1 2]]

```
A=
     2     0
     0     1
     3     3
     1     2
```

Equivale a **cat(1,A,[1 2])**

» **a = [1 2 3 4]; b = [5 6 7 8]; c = [a ; b]**

```
c =
     1     2     3     4
     5     6     7     8
```

Definiamo una matrice “c” come concatenazione dei vettori “a” e “b”.

Nota: l’uso di “;” entro [….] indica la fine della riga.

» **size(c)**

```
ans =
     2     4
```

Definiamo la matrice “d” ponendo “a” e “b” a fianco: notare il risultato diverso da “c”

» **d = [a b]**

```
d =
     1     2     3     4     5     6     7     8
```

Condizioni logiche su matrici

```
» d = c(1,:)
d =
    1    2    3    4
```

Possiamo definire un vettore “e” che è una funzione logica di d

```
» e = d>2
e =
    0    0    1    1
```

Possiamo ora usare “e” per trovare gli elementi di d>2

```
» d(e)
ans =
    3    4
```

Matrici speciali

» ones(2,3)

ans =

```
    1    1    1
    1    1    1
```

» zeros(1,4)

ans =

```
    0    0    0    0
```

» rand(3,3)

ans =

```
    0.2176    0.4909    0.8985
    0.4054    0.1294    0.5943
    0.5699    0.5909    0.3020
```

» ones(2)

ans =

```
    1    1
    1    1
```

» zeros(2,1)

ans =

```
    0
    0
```

» eye(2)

ans =

```
    1    0
    0    1
```

Alcuni comandi per manipolare matrici

Comando	Azione
`A=[]`	genera la matrice vuota A
`A'`	genera la matrice trasposta di A
`A=eye(n)`	genera la matrice identità $A = (a_{ij})_{i,j=1,\ldots,n}$, con $a_{ij} = \delta_{ij}$
`A=zeros(n,m)`	genera la matrice $A = (a_{ij})_{i=1,\ldots,n,j=1,\ldots,m}$, con $a_{ij} = 0$
`A=ones(n,m)`	genera la matrice $A = (a_{ij})_{i=1,\ldots,n,j=1,\ldots,m}$, con $a_{ij} = 1$
`A=rand(n,m)`	genera la matrice $A = (a_{ij})_{i=1,\ldots,n,j=1,\ldots,m}$, con $0 < a_{ij} < 1$ pseudo-casuali
`A=hilb(n)`	genera la matrice di Hilbert $A = (a_{ij})_{i,j=1,\ldots,n}$, con $a_{ij} = 1/(i+j-1)$
`A=vander(x)`	genera la matrice di Vandermonde $A = (a_{ij})_{i,j=1,\ldots,n}$, con $a_{ij} = x_i^{n-j}$
`A(r,c)`	estrae gli elementi di `A` appartenenti all'intersezione delle righe e delle colonne specificate in `r` e in `c` rispettivamente
`A(r,c)=C`	assegna agli elementi di `A` (i cui indici di riga e di colonna sono specificati in `r` e in `c`) i valori definiti in C rispettivamente
`A(r,c)=[]`	rimuove gli elementi di `A` (i cui indici di riga e di colonna sono specificati in `r` e in `c`)
`A([i j],c)=A([j i],c)`	scambia gli elementi delle righe `i` e `j` di `A` appartenenti alle colonne specificate in `c`
`A(r,[i j])=A(r,[j i])`	scambia gli elementi delle colonne `i` e `j` di `A` appartenenti alle righe specificate in `r`

Operazioni aritmetiche su vettori-matrici

```
» a = [1 2 3]
a =
    1    2    3

» b = [4 5 6]
b =
    4    5    6

» a + b
ans =
    5    7    9


» ones(3)+5*ones(3)
ans =
   ????????????
```

Operatori:

\+ -

Somma/sottrazione
tra elementi corrispondenti
purchè le dimensioni siano compatibili

Operazioni aritmetiche su vettori

```
» a = [1 2 3]
a =
    1    2    3

» b = [4 5 6]
b =
    4    5    6

>> a'*b
ans =

    4    5    6
    8   10   12
   12   15   18

» a*b'
ans =
   32
```

Operatori:
* /

Moltiplicazione tra vettori:
-prodotto scalare
-prodotto esterno

moltiplichiamo una matrice 3 x 1 per una 1 x 3 per ottenere una matrice 3 x 3)

moltiplichiamo una matrice 1 x 3 per una 3 x 1 per ottenere una 1 x 1 scalare)

Operazioni aritmetiche su matrici

Moltiplicazione tra matrici

C=A*B

Operatori:
* /

moltiplicare una matrice **A (n x m)** con una matrice **B (m x p)** per ottenere una matrice **(n x p)**

```
>> A= [1 2 ; 0 2]; B= [2 3 ; 1 0]
>> C= A*B

C =
     4     3
     2     0
```

$$\begin{bmatrix} 1 & 2 \\ 0 & 2 \end{bmatrix} * \begin{bmatrix} 2 & 3 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 4 & 3 \\ 2 & 0 \end{bmatrix}$$

Operazioni aritmetiche su vettori-matrici

Operatori:
+ - * /

Attenzione alle dimensioni dei vettori /matrici

??? Error using ==> mtimes
Inner matrix dimensions must agree.

Operatori su elementi di vettori-matrici

Gli operatori su elementi indicano operazioni aritmetiche tra elementi corrispondenti: .* ./ .\ .^

» **f =[1 2 3]; g= [4 5 6];** Definiamo i vettori 1x3 **f** e **g**.

Nota: usando ";" alla fine della linea si elimina la stampa del risultato.

```
» h=f.*g
h = 4  10  18

» h=f.\g
h = 4.0000   2.5000  2.0000

» h=f./g
h = 0.2500  0.4000  0.5000

» h=f.^2
h =  1   4   9
```

Valido per vettori e matrici.

Tabella 10. Operazioni puntuali in MATLAB.

Operazione	Azione
`z=x.*y`	genera il vettore riga (colonna) $z = \{z_i\}_{i=1,\ldots,n}$, con $z_i = x_i * y_i$
`z=x./y`	genera il vettore riga (colonna) $z = \{z_i\}_{i=1,\ldots,n}$, con $z_i = x_i / y_i$
`z=x.^y`	genera il vettore riga (colonna) $z = \{z_i\}_{i=1,\ldots,n}$, con $z_i = x_i^{y_i}$
`z=x.^e`	genera il vettore riga (colonna) $z = \{z_i\}_{i=1,\ldots,n}$, con $z_i = x_i^e$
`C=A.*B`	genera la matrice $C = (c_{ij})_{i,j=1,\ldots,n}$, con $c_{ij} = a_{ij} * b_{ij}$
`C=A./B`	genera la matrice $C = (c_{ij})_{i,j=1,\ldots,n}$, con $c_{ij} = a_{ij} / b_{ij}$
`C=A.^B`	genera la matrice $C = (c_{ij})_{i,j=1,\ldots,n}$, con $c_{ij} = a_{ij}^{b_{ij}}$
`C=A.^e`	genera la matrice $C = (c_{ij})_{i,j=1,\ldots,n}$, con $c_{ij} = a_{ij}^e$

Variabili complesse

Un complex double array è visto come la somma di due double array.

Con le variabili 'i' o 'j' si indica l'unità complessa

$$i = \sqrt{-1}$$

- Numeri complessi

  z = 3 + 4 * i (o **z = 3 + 4 * j**)

Variabili complesse

Array di variabili complesse

B=[1 2 ; 3 4]+i*[5 6 ; 7 8]

o equivalentemente

B=[1+5*j 2+6*j ; 3+7*j 4+8*j]

Non ci devono essere spazi bianchi nell'espressione del numero complesso.