MATLAB:

Per le Scienze Comportamentali

Laboratorio Matlab 2012/2013

Matlab `e uno dei programmi scientifici di maggior diffusione, grazie alle sue numerose applicazioni in campi quali l'elettronica, la controllistica, l'analisi dei segnali, l'elaborazione di immagini, la chimica, la statistica e numerosi altri. Viene utilizzato in molti corsi universitari e di ingegneria, e sono ormai numerose le pubblicazioni scientifiche che utilizzano l'ambiente di Matlab quale sostegno matematico della teoria. La primissima versione di Matlab risale alla fine degli anni '70, scritta alla University of New Mexico e alla Stanford University quale pacchetto software di supporto alle lezioni di Algebra lineare e Analisi numerica . Oggi Matlab non si limita pi ` u al solo calcolo matri- ciale e numerico, ma ha sviluppato tutta una serie di funzioni per le applicazioni pi ` u diverse nel campo scientifico. La semplicit`a del linguaggio permette di risolvere problemi molto complessi senza dover sviluppare programmi in C o altri linguaggi di programmazione.

Matlab lavora con alcuni tipi di dati che sono:

• La matrice n-dimensionale di numeri reali, complessi, caratteri o strutture piu' complesse.

• La cella, un contenitore per svariati tipi di dati.

Per la creazione di variabili occorre tenere presente alcune convenzioni

1. Le variabili sono case-sensitive

2. I nomi di variabili possono contenere fino a 19 caratteri 3. I nomi delle variabili devono iniziare con una lettera e possono contenere lettere, numeri e '_'.

Interfaccia

Le finistre laterali.

A sinistra della finistra rincipale ci sono due finistre più piccole.

Finestra superiore :
Si può scegliere fra **WorkSpace** e **Current Directory**.

• **WorkSpace**: Compare il nome delle variabili, con il valore, la dimensione, il numero di Bytes occupati, ed il tipo.
NB Per scegliere quale informazioni vedere cliccate su View e scegliere le colonne.

• **Current Directory** : compare la lista dei file contenuti nella directory corrente con il tipo (file o folder) e la data dell'ultimo aggiornamento.

• **Command History**: contiene tutti i comandi che vengono eseguiti nella finestra pricipale

Menù principale

- **File**: gestisce i file di Matlab.
- **Edit:** Soliti comandi per editare i file.
- **View:** Configurazioni delle finestre
- **Graphics:** Gestisce i grafici
- **Debug:** per trovare gli errori nel programma
- **Desktop:** Configurazione del desktop di Matlab
- **Help** : Aiuto in linea

Arriviamo al dunque…….

Numeri,

Vettori,

Matrici !!!!

Numeri

>> a = 2

a =

2

Se utilizziamo il punto e virgola non permettiamo l'uscita dell'output (ans)

>> a = 2;

Digitando il comando Whos nella command window ci dirà che tipo di variabili sono prensenti nel workspace

>> whos

Name	Size	Bytes	Class	Attributes
a	1x1	8	double	

Per rimuovere la variabile basta digitare

>> clear a

Formato di rappresentazione dei numeri Short

```
>> c=0.456723

c =

   0.4567
```

Il numero è stato rappresentato con 5 numeri

```
>> format short e
>> c

c =

   4.5672e-01
```

Forma esponenziale con 5 cifre

Formato di rappresentazione dei numeri Long

```
>> format long e
>> c

c =

    4.567230000000000e-01
```

Forma esponenziale con 16 cifre

```
>> format long
>> c

c =

   0.456723000000000
```

Il numero è stato rappresentato con 5 numeri

Variabile	Significato
`FORMAT`	Default.
`FORMAT SHORT`	Virgola fissa scalata con 5 cifre.
`FORMAT LONG`	Virgola fissa scalata con 15 cifre.
`FORMAT SHORT E`	Forma esponenziale con 5 cifre di mantissa.
`FORMAT LONG E`	Forma esponenziale con 15 cifre di mantissa.
`FORMAT SHORT G`	Sceglie la rappresentazione migliore con 5 cifre.
`FORMAT LONG G`	Sceglie la rappresentazione migliore con 15 cifre.

$1.5 \Rightarrow$

format short	1.5000
format long	1.50000000000000
format short e	1.5000e+00
format long e	1.500000000000000e+00
format short g	1.5
format long g	1.5

Operazioni aritmetiche

^ potenza
* prodotto
/ divisione
+ somma
- differenza

Es: per calcolare $x = \dfrac{3 + 5^3 - 2/3}{4(5 + 2^4)}$

```
>> x=(3+5^3-2/3)/(4*(5+2^4))
```

- Sono osservate le precedenze classiche dell'artimetica
- Per alterare le precedenze si utilizzano esclusivamente le parentesi **tonde**

Array

Il linguaggio MATLAB lavora con un solo tipo di oggetti: **l'array di MATLAB**.

Tutte le variabili di Matlab, inclusi scalari, vettori, matrici, stringhe, celle (cell arrays), strutture e oggetti sono memorizzati in Matlab come **array**.

Ogni array contiene le seguenti informazioni:

- Il tipo
- La dimensione
- I dati associati all'array
- Se la variabile è reale o complessa, nel caso di array numerico
- Gli indici e gli elementi diversi da zero, nel caso di array sparse
- Il numero di campi e il nome dei campi, nel caso di una struttura o oggetto.

Vettori:

I numeri vanno bene ma il più delle volte abbiamo più di un solo numero:

• Un valore di una funzione f(x) per differenti valori di x

• le risposte dei soggetti a stimoli di differente intensità

•Etc…

Si può utilizzare il simbolo [ ] per unire i signoli numeri in un vettore

>> x= [1 2 3 4 5]

x =

1 2 3 4 5

Vettori riga – colonna

1 riga x 4 colonne (1x4) (M,N):

```
>> a=[0 1 2 3]
a =
   0    1    2    3:
```

4 riga x1 colonne (4x1) (M,N):

```
>> a=[0;1;2;3]
a =

   0
   1
   2
   3
```

Altri metodi per definere un vettore:

```
>> x=[1:5]
x =
   1    2    3    4    5
```

Vettori con salti spazialmente definiti:

```
>> x=[2:2:8]
x =
  2    4    6    8
```

Vettori equamente definiti: (help linspace)

```
>> x=linspace(2,8,4)
x =
 2    4    6    8
```

N Numeri random (Distribuzione uniforme tra 0 e 1)

```
>> rand(1,4)
ans =
 0.2830    0.5383    0.6706    0.5575
```

N Numeri random (Distribuzione normale uniforme media zero ,std=1)

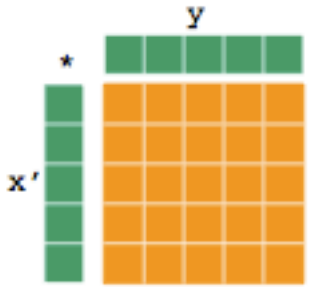
```
>> randn(1,4)
ans =
 -0.4326   -1.6656    0.1253    0.2877
```

Moltiplicazioni tra vettori

```
>> x=[1:4]; y=[10:10:40];
>> x*y
```

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

```
>> x'*y
ans =
   10   20   30   40
   20   40   60   80
   30   60   90  120
   40   80  120  160
```

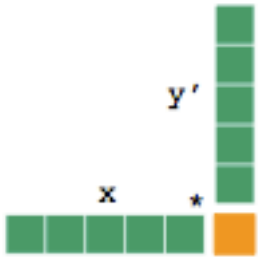


Transposto : trasforma un vettore riga
In un vettore colonna

‘ : segno del transposto

```
>> x*y'
ans =
300
```

Prodotto scalare



Matrici

I vettori sono utili, ma capita avvolte che abbiamo più di un vettore:

• valori differenti funzioni f(x), g(x), h(x) per la stessa x

• risposte di N soggetti all'intensità di uno stimolo.

•Etc..

Matrici

Assegnazione di array - matrici

```
>> c=[5 3 4; 2 4 -2]
c =
     5     3     4
     2     4    -2
```

Per generare un array 2x3, matrice 2 righe e 3 colonne

```
>> d(3,4)=3

d =
     0     0     0     0
     0     0     0     0
     0     0     0     3
```

Genera una matrice 3x4, che ha tutti elementi nulli tranne quello di posto 3,4

Lo spazio o la virgola separano elementi sulla stessa riga.
Il punto e virgola separa le righe.

Funzioni e operazioni con le matrici:

Dimensioni array :

X=[1:4]

>> size(x)

ans =

1 4 (righe,colonne)

>> size(d)

ans =

2 3

Lunghezza :

>> length(x)

ans =

4

Matrici predefinite:

x=zeros(2,2)

x =

0 0

0 0

>> x=ones(2,2)

x =

1 1

1 1

Come estrarre righe e colonne di una matrice

Dato un array A:

- `A(:,j)` è la j-esima colonna di A;
- `A(i,:)` è la i-esima riga di A;
- `A(:,j:k)` è la sottomatrice di A che contiene le colonne di A dalla j-esima alla k-esima;
- `A(:)` fornisce tutti gli elementi di A, vista come una singola colonna

>> d=[1 2 3;4 5 6]

d =

1 2 3
4 5 6

Estrarre elementi matrice

d(1:1)= primo elemento della prima riga prima colonna

>> d(1:1)
ans =
1
>> d(1,:) PRIMA RIGA
ans =
1 2 3
>> d(:,1) PRIMA COLONNA
ans =
1
4

d(:,1:3) dalla prima alla 3 colonna

ans =

1 2 3
4 5 6

d(1:2,:) dalla prima alla 3 riga

Sostituire elementi matrice

d(1:1)=0
d =
0 2 3
4 5 6
d(1,:)=[] Rimuove la prima riga
d =
4 5 6

c=d(2:-1:1,[1:3])
c =
4 5 6
1 2 3

2.4 Manipolazione di matrici

Altre operazioni speciali su una matrice sono descritte nella tabella 2.1

flipud(A)	scambia i valori della matrice A dall'alto in basso
fliplr(A)	scambia i valori della matrice A da sinistra a destra
rot90(A)	ruota la matrice A di 90 gradi in senso antiorario
tril	estrae la parte superiore triangolare
triu	estrae la parte inferiore triangolare
reshape(A,m,n)	ridimensiona la matrice A a m x n. A deve contenere m*n elementi !
end	ultimo indice di una matrice
diag(v)	crea una matrice con sulla diagonale i valori del vettore v
diag(A)	crea un vettore con gli elementi della diagonale della matrice A

Tabella 2.1: Operazioni su matrici

Matrici

Unire due vettori per formare una matrice

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

Oppure il comando Cat

```
>> cat(1,a,b)
ans =
   1   2   3
   4   5   6
>> cat(2,a,b)
ans =

   1   2   3   4   5   6
```

```
>> vertcat(a,b)
ans =

   1   2   3
   4   5   6
>> horzcat(a,b)

ans =

   1   2   3   4   5   6
```

Matrici Multidimensionali 3 o 4 dimensioni:

mat(1, :,:) =[0 1 1 0 ; 0 0 0 0 ; 0 0 0 1 ; 0 1 0 0 ; 0 0 1 0];
mat(2, :, :)=[1 0 1 1 ; 1 0 1 1 ; 0 0 0 0 ; 0 0 0 0 ; 0 1 0 0];
mat(3, :, :)=[0 0 0 0 ; 1 1 0 1 ; 0 0 0 0 ; 0 0 0 0 ; 0 0 0 0];

mat(:,:,1) =

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

mat(:,:,2) =

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

mat(:,:,3) =

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

mat(:,:,4) =

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

Non solo numeri….

Caratteri……

…….stringhe…..

Immettere i valori utilizzando le virgolette permette di creare Elementi stringa.

```
>> a= 'Mi chiamo'
a =
Mi chiamo
>> b= 'Marco'
b =
Marco
>> whos
  Name      Size            Bytes  Class    Attributes

  a         1x9                18  char
  b         1x5                10  char
```

Stringa.

```
>> a='Ciao mi chiamo Marco'

a =

Ciao mi chiamo Marco
```

```
>>str2mat('casa','cassone','nome
')

ans = casa cassone nome

>> size(ans) ans = 3 7
```

Str2mat adatta il numero di colonne di una matrice seconda la stringa più lunga

abs	Converte una stringa nel corrispondente valore numerico
isstr	vero se la variabile è una stringa
setstr	converte valore numerico in stringa
str2mat	Crea una matrice testo
lower	converte in minuscole
upper	converte in maiuscole
strcmp	confronta 2 stringhe
int2str	converte intero in stringa
num2str	converte numero in stringa
sprintf	converte numero in stringa (come C)
str2num	converte stringa in numero
sscanf	converte stringa in numero (come C)
dec2hex	conversione decimale-esadecimale
hex2dec	conversione esadecimale-decimale
hex2num	conversione esadecimale a floating point
base2dec	conversione da base B a decimale
dec2base	conversione da decimale a base B

Tabella 2.3: Operazioni con stringhe

Celle…..

….Strutture….

Con le strutture è possibile accorpare dati di differenti categorie come ad esempio

- **Stringe**
- **Alfabeto**
- **Numeri**
- **Etc…**

```
>> article.title='La visione al contrasto';
>> article.year=2001;
>> article.pages=121:125;

article =
    title: 'La visione al contrasto'
     year: 2001
    pages: [121 122 123 124 125]
```

Con le celle: infatti possiamo costruire una matrice se i suoi elementi hanno la stessa dimensione

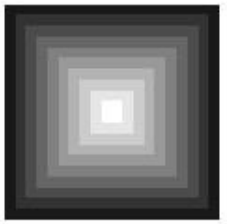
Ma se volessimo immagazzinare immagini di diverse dimensioni??

Useremo le Celle

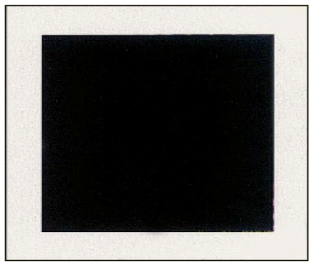
```
% ===  Modern art :-) ===
% K. Malewich, Black Square, 1913. 106.2 x 106.5 cm

I{1} = zeros(100);
% V. Vasarely, Arcturus II, 1969. 160.0 x 159.7 cm
[X Y] = meshgrid(1:10, 1:10);
Q = min(cat(3, X, Y), [], 3)/10;
I{2} = [Q,fliplr(Q);flipud(Q), flipud(fliplr(Q))];
% G. Richter, 4096 Farben, 1974 (WVZ 359). 254 x 254 cm
I{3} = rand(64, 64, 3);

for i=1:length(I)
subplot(1,3,i)
imshow(I{i})
end
```

```
>> I
I =
    'Marco'    [10]    [3x3 double]
```

Matlab Modern Art (Un falso)


"Original" master pieces



Facciamo un esempio pratico

Immaginiamo di presentare un serie di stimoli ad un soggetto sullo schermo. Ai soggetti viene chiesto di premere R quando vedono la faccia di Russel Crowe e E quando vedono la faccia di Leonardo di Caprio. Quando andiamo a plottare i risultati avremo un' alternanza di facce con un vettore risposta del tipo :

resp='rerererererererererererererererererererere';

Immaginiamo ora che per distrazione il soggetto durante la somministrazione risponda K avremo una cosa di questo tipo:

resp='rerekererererererererererererererererere';

Per verificare cosa è avvenuto al 5 Trial dovremmo accedere a questo elemento nel vettore stringa:
>> resp(5)
ans =
K

E poi andare a sostituirlo con il valore R
>> resp(5)='r'
resp =
Rerererererererererererererererererererere

Infine possiamo verificare la presenza delle R e delle E andand a scorrere il vettore
>> resp(2:2:40)
ans =
eeeeeeeeeeeeeeeeeee
>> resp(1:2:40)
Ans
rrrrrrrrrrrrrrrrrrr

Operatori Logici

`&`	And
`|`	Or
`~`	Not
`xor`	XOR

Tabella 5.2: Operazioni logiche

Uguale ==

n1=3.2
n2=3.1
n1==n2
n1 = 3.2000
n2 = 3.1000
ans = 0

Codifica 0= falso 1 = vero

n2=3.2
n1==n2
Ans= 1

Maggiore minore

n1=1; n2=2; n3=3;
n1<n2 & n1>n3
ans = 0

n1<n2 | n1>n3
ans =1

exist	controlla se esiste una variabile o funzione
find	cerca una certa condizione vera in un vettore
finite	controlla se un numero è finito
isempty	controlla se una matrice è vuota
isieee	controlla se il computer usa matematica IEEE
isinf	controlla se un numero è infinito
isnan	controlla se un numero non esiste
issparse	controlla se una matrice è sparsa
isstr	controlla se un vettore rappresenta una stringa

Tabella 5.3: Richieste logiche

Controlli di Flusso

Esistono 3 metodi per controllare il flusso di un programma e prendere decisioni all'interno dello stesso:

- for loop

- While loop

- if – elseif – else

- Switch- case - otherwise

Ciclo for

In Matlab la ripetizione di blocchi di istruzioni per un numero di volte specificato e in modo incondizionato viene eseguita tramite l'istruzione di ciclo FOR ...END la cui sintassi e`:

for indice = espressione
 blocco di istruzioni
end

```
>> for i=1:5
 >>   disp(['numero: ',num2str(i)]);
>> end
numero: 1
numero: 2
numero: 3
numero: 4
numero: 5
```

```
for i=5:-1:1
>>   disp(['numero: ',num2str(i)]);
 >>end
numero: 5
numero: 4
numero: 3
numero: 2
numero: 1
```

N.B i cicli for possono essere annidati l'uno nell' altro

Esempio Pratico di un ciclo for

Immaginiamo di avere creato un ordine di presentazione di un certo stimolo (1 = quadrato 0= cerchio) e vogliamo che ad ogni trial il programma scelga in maniera lo stimolo da presentare:

```
>> A=repmat(0:1,1,5)
A =
   0    1    0    1    0    1    0    1    0    1
```

Presntiamo:

```
>> for i=1:length(A)
A(i)
End

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

Ciclo If

Il ciclo if esegue un comando solo se una relazione o un comando è VERO

```
i=10
>> if i > 10
disp('Number i is > 10.')
End

Nessuna risposta

if i == 10
disp('Number i is = 10.')
end
Number i is = 10.
```

```
if i > 10

disp('Numberi is > 10. ')

Elseif i == 10

disp('Numberi equals 10.')

Else

disp('Numberi is < 10.')
End

Numberi is = 10.
```

Ciclo Switch (molto simile al ciclo if)

```
>> value =10

value =

   10

>>  switch value
case 1,  disp('Valueis 1.')
case 42, disp('Valueis 42.')
case {2, 3}, disp('Numberis 2 or 3')
otherwise
disp('Valueis not in our list.')
end
Valueis not in our list.
```

Break : permette di interrompere immediatamente l’esecuzione di un ciclo for

```
for i = 1 : 10
a = input('Inserisci a ');
if a == 0 disp('attenzione e` un denominatore!');
   break;
end
x(i) = 1/a;
end
```

Ciclo while

Se si ha la necessità di ripetere una o più istruzioni fintanto che una condizione sarà verificata non sapendo a priori il numero di ripetizioni si usa l'istruzione WHILE ...END, la cui sintassi è:

while condizione
blocco istruzioni
end

```
>> i=0
i =
     0
>> while i<5
disp('ciao')
i=i+1
End
ciao
ciao
ciao
ciao
ciao
```

Creare Funzioni in Matlab

Le function sono sottoprogrammi:

- hanno dei parametri di ingresso e di uscita che permettono la comunicazione con la command window.

- utilizzano variabili locali che esistono solo in fase di esecuzione della funzione e sono diverse da quelle della command window

- Le function permettono la riusabilità del codice (la stessa procedura con parametri di input diversi)

Possono chiamare altre function.

Sintassi

La prima riga deve essere del tipo :

Function [output] = nomefunzione (input)

ci possono essere anche più di un output e il nome della funzione durante il salvataggio deve Essere del tipo

nomefunzione.m

```
function [s,d] = sumdiff(x,y)

%% Questa funzione fa visualizzare la somma e la differenza tra due
%% numeri dati in input

% somma e differenza
% deve essere creato il file sumdiff.m
s = x+y;
d = x-y;
```

Grafici e Plottaggio

In Matlab `e possibile utilizzare comandi per grafica 2D e 3D. Esistono inoltre comandi per creare delle animazioni.

t = 0:pi/100:2*pi; y = sin(t);

plot(t,y)

>> **hold on**
>> z=cos(t);
>> plot(t,z)
plot(t,z,t,y)
xlabel('Asse x') ylabel('Asse y')
title('Grafico delle 2 funzioni')

Plot(t,z,'-',t,y,':')

plot	Crea un grafico da vettori di valori
loglog	Grafico con assi logaritmici
semilogx	Grafico con asse x logaritmico e y lineare
semilogy	Grafico con asse x lineare e y logaritmico
title	Aggiunge un titolo al grafico
xlabel	Label asse x
ylabel	Label asse y
text	Mostra un testo ad una posizione generica
gtext	Mostra un testo ad una posizione acquisita con il mouse
grid	Mostra (toglie) la griglia

Plot di più figure (subplot)

```
t = 0:pi/10:2*pi;
y = sin(t);
subplot(2,2,1)
plot(t,y)
y1=cos(t);
subplot(2,2,2)
plot(t,y1)
y2=tan(t);
subplot(2,2,3)
plot(t,y2)
y3=atan(t);
subplot(2,2,4)
plot(t,y3)
```

Codici per i grafici

Simbolo	Colore	Simbolo	Stile linea
y	giallo	.	punto
m	magenta	o	cerchio
c	ciano	x	croce
r	rosso	+	segno +
g	verde	*	stella
b	blu	-	linea continua
w	bianco	:	linea punteggiata
k	nero	-.	Linea-punto
		--	linea tratteggiata

Demo

```
%% Manipolate 3- dimensional Matrix la funzione consente di maniolare una
%% matrice 3-d accendendo o spegnendo i canli RGB (palette colore)
R=ones(100,100,3);
%% Red
R(:,:,1)=1
R(:,:,2)=0
R(:,:,3)=0
I{1}=R
%% Blue
R(:,:,1)=0
R(:,:,2)=1
R(:,:,3)=0
I{2}=R
%% Green
R(:,:,1)=0
R(:,:,2)=0
R(:,:,3)=1
I{3}=R

%% Magenta
R(:,:,1)=1
R(:,:,2)=0
R(:,:,3)=1
I{4}=R

for i=1:length(I)
subplot(1,4,i)
imshow(I{i})
end
```