



Introduzione a Matlab

Informatica B

Che cos'è Matlab?

- ❑ Matlab è uno strumento per il calcolo numerico
- ❑ Facilita lo sviluppo di programmi che eseguono complesse elaborazioni di calcolo numerico grazie a:
 - ▶ un ambiente di sviluppo integrato ed uno specifico linguaggio di programmazione
 - ▶ una ricca libreria di funzioni matematiche
- ❑ È uno strumento commerciale ma ne esiste una alternativa gratuita di nome Octave
 - ▶ molto simile a Matlab in molti aspetti
 - ▶ <http://www.gnu.org/software/octave>
- ❑ Testo
 - ▶ *Introduzione alla programmazione in MATLAB*. Campi, Di Nitto, Loiacono, Morzenti, Spoletini. Esculapio Editrice.



Caratteristiche del linguaggio di Matlab

- ❑ È un linguaggio di alto livello (come il C o il Java)
- ❑ È orientato alle elaborazione numeriche
- ❑ È un linguaggio interpretato
 - ▶ Non richiede la fase di traduzione in codice macchina
 - ▶ Il sorgente viene analizzato da un programma interprete che esegue direttamente tutte i comandi richiesti
- ❑ Non è un linguaggio tipizzato
 - ▶ Non occorre dichiarare le variabili
 - ▶ Non è necessario specificarne il tipo
 - ▶ Alla stessa variabile possono essere assegnati valori di tipo diverso durante l'esecuzione del programma

Array e variabili

- ❑ L'unità fondamentale di dati in MATLAB è l'array: ogni variabile è un array (le variabili scalari sono array con un solo elemento).
- ❑ I nomi di variabili seguono regole simili a quelle del C
- ❑ Il C è un linguaggio a tipizzazione forte
 - ▶ Le variabili vanno dichiarate prima dell'uso
- ❑ Il MATLAB è un linguaggio a tipizzazione debole
 - ▶ Le variabili vengono create assegnando ad esse dei valori
 - ▶ Il loro tipo è determinato dal tipo dei valori assegnati

Creazione ed inizializzazione di una variabile

- ❑ Le variabili sono create al momento dell'inizializzazione
- ❑ Modi di inizializzazione
 - ▶ Assegnamento
 - ▶ Lettura dati da tastiera
 - ▶ Lettura da file
- ❑ Accesso ad un singolo elemento di un array:

nome(pos)

- ▶ dove pos è la posizione dell'elemento (≥ 1)

Assegnamento

□ Scalari

nome = valore

- ▶ $a = 3$
- ▶ $b = 55$

□ Array


nome = $[v_1 \ v_2 \ \dots \ v_n]$

- ▶ $a = [3 \ 4 \ 2 \ 3]$

□ Matrice

nome = $[v_{11} \ v_{12} \ \dots \ v_{1n}; \dots; v_{m1} \ v_{m2} \ \dots \ v_{mn}]$

- ▶ $a = [3 \ 4 \ 2; 4 \ 5 \ 6]$


$$\begin{bmatrix} 3 & 4 & 2 \\ 4 & 5 & 6 \end{bmatrix}$$

Assegnamento (2) – esempi

□ Esempi

▶ $a = [0 \ 7+1]$;  contenuto di a

▶ $b = [a(2) \ 5 \ a]$;

 secondo elemento di a

□ Risultato

▶ $a = [0 \ 8]$

▶ $b = [8 \ 5 \ 0 \ 8]$

□ Non tutti gli elementi devono essere specificati alla creazione...

▶ $c(2, 3) = 5$;

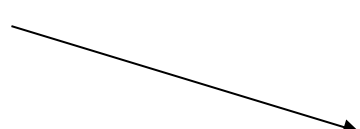
 $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 5 \end{bmatrix}$

Assegnamento (3) – esempi

- ❑ L'array può essere esteso successivamente ...

▶ $d = [2\ 5]; d(4)=2; \longrightarrow d = [2\ 5\ 0\ 2]$

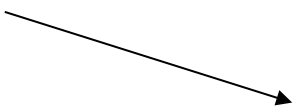
- ❑ Operatore di trasposizione

▶ $g = d';$  $\begin{bmatrix} 2 \\ 5 \\ 0 \\ 2 \end{bmatrix}$

- ❑ Come evitare di enumerare esplicitamente tutti i valori, uso dell'operatore :

▶ $x = 1:2:10; \longrightarrow x = [1\ 3\ 5\ 7\ 9]$

▶ $l = 1:3;$

▶ $m = [l' l'];$  $\begin{bmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \end{bmatrix}$

Assegnamento (4) – funzioni predefinite

Funzione	Significato
<code>zeros (n)</code>	Genera una matrice nxn di zeri
<code>zeros (m,n)</code>	Genera una matrice mxn di zeri
<code>zeros (size(arr))</code>	Genera una matrice di zeri della stessa dimensione di arr
<code>ones(n)</code>	Genera una matrice nxn di uno
<code>ones(m,n)</code>	Genera una matrice mxn di uno
<code>ones(size(arr))</code>	Genera una matrice di uno della stessa dimensione di arr
<code>eye(n)</code>	Genera la matrice identità nxn
<code>eye(m,n)</code>	Genera la matrice identità mxn
<code>length(arr)</code>	Ritorna la dimensione più lunga del vettore
<code>size(arr)</code>	Ritorna il numero di righe e colonne dell'array

Assegnamento (5) – funzioni predefinite

□ Esempi

▶ `a = zeros(2);` → $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

▶ `b = zeros(2,3);` → $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

▶ `c = [1 2; 3 4];`

▶ `d = zeros(size(c));` → $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$


Tipo double

- ❑ Una variabile di tipo double contiene uno **scalare** o un **array** di numeri in doppia precisione (64 bit)
- ❑ Questi numeri possono essere
 - ▶ Reali, es $\text{var1} = -10.7;$
 - ▶ Immaginari, es $\text{var2} = 4i;$ $\text{var3} = 4j;$
 - ▶ Complessi, es $\text{var3} = 10.3 + 10i;$
- ❑ Es: $x = [-1.3 \ 3.1+5.3j \ 0]$
- ❑ Le parti reali e immaginarie possono essere positive e negative nell'intervallo di valori $[10^{-308}, 10^{308}]$, con accuratezza di 15-16 cifre decimali

Tipo char

- Una variabile di tipo char contiene uno **scalare** o un **array** di valori a 16 bit, ciascuno dei quali rappresenta un carattere
 - ▶ Es: commento = 'questa è una stringa';

Nome della variabile Array di 1x21 caratteri



Assegnamento (5) – uso di uno scalare per assegnare valori ad un array

- Esempio

▶ $m(1:4, 1:3) = 3$ \longrightarrow $\begin{bmatrix} 3 & 3 & 3 \\ 3 & 3 & 3 \\ 3 & 3 & 3 \\ 3 & 3 & 3 \end{bmatrix}$

- Regola: il modo con cui uno scalare viene assegnato ad un array dipende dalla forma dell'array che viene specificata a sinistra dell'assegnamento

- Esempio 2

▶ $m(1:2, 1:2) = 4$ \longrightarrow $\begin{bmatrix} 4 & 4 & 3 \\ 4 & 4 & 3 \\ 3 & 3 & 3 \\ 3 & 3 & 3 \end{bmatrix}$

- ... la regola si applica anche ai sottoarray

Variabili predefinite

- ❑ Matlab definisce un insieme di variabili predefinite (e.g., pi)
- ❑ Queste variabili corrispondono in qualche caso a costanti
 - ▶ **Attenzione!** Il valore di queste variabili può essere modificato, per esempio
 - `circ1=2*pi*10;`
 - `pi = 3;`
 - `circ2=2*pi*10;`
 - ▶ Il valore di `circ2` non sarà più la circonferenza di un cerchio
- ❑ E` fortemente sconsigliato modificare il valore di una variabile predefinita

Variabili predefinite più comuni

Variabile	Scopo
pi	contiene 15 cifre significative di π
i, j	contiene il valore i ($\sqrt{-1}$)
inf (o Inf)	rappresentazione dell'infinito (ottenuto di solito come risultato di una divisione per 0)
nan	Not-A-Number è il risultato di una operazione matematica non definita, es 0/0
clock	contiene la data e l'orario corrente. E` un vettore di sei elementi (anno, mese, giorno, ora, minuti, secondi)
date	contiene la data corrente sotto forma di stringa
eps	epsilon: la più piccola differenza rappresentabile tra due numeri
ans	Variabile speciale usata per immagazzinare risultati non assegnati ad altre variabili

Operazioni con scalari e array

- ❑ Operazioni per gli scalari: + - * / ^
- ❑ Operazioni per gli array
 - ▶ Array operation: viene eseguita sugli elementi degli array coinvolti (devono avere lo stesso numero di righe e colonne)

▶ $a = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ $b = \begin{bmatrix} 2 & 3 \\ 5 & 7 \end{bmatrix}$ $a+b = \begin{bmatrix} 3 & 5 \\ 8 & 11 \end{bmatrix}$ $a.*b = \begin{bmatrix} 2 & 6 \\ 15 & 28 \end{bmatrix}$

- ▶ Matrix operation: segue le regole dell'algebra lineare

▶ $a = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ $b = \begin{bmatrix} 2 & 3 \\ 5 & 7 \end{bmatrix}$ $a*b = \begin{bmatrix} 12 & 17 \\ 26 & 37 \end{bmatrix}$ ← $\sum_k a_{ik} * b_{kj}$

Operazioni tipiche per gli array

Operazione	Sintassi Matlab	Commenti
Array addition	$a + b$	Array e matrix addition sono identiche
Array subtraction	$a - b$	Array e matrix subtraction sono identiche
Array multiplication	$a .* b$	Ciascun elemento del risultato è pari al prodotto degli elementi corrispondenti nei due operandi
Matrix multiplication	$a * b$	Prodotto di matrici
Array right division	$a ./ b$	$\text{risultato}(i,j) = a(i,j)/b(i,j)$
Array left division	$a .\ b$	$\text{risultato}(i,j) = b(i,j)/a(i,j)$
Matrix right division	a / b	$a * \text{inversa}(b)$
Matrix left division	$a \ b$	$\text{inversa}(a) * b$
Array exponentiation	$a .^ b$	$\text{risultato}(i,j) = a(i,j)^b(i,j)$

Matrix left division

- Serve per risolvere sistemi di equazioni lineari

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1$$

$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2$$

$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3$$

- può essere espresso come $Ax=B$ con

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad B = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

- di conseguenza, $x = A^{-1}B = A \setminus B$

Altre funzioni

Funzione	Scopo
<code>ceil(x)</code>	approssima x all'intero immediatamente maggiore
<code>floor(x)</code>	approssima x all'intero immediatamente minore
<code>fix(x)</code>	approssima x all'intero più vicino verso lo zero
<code>max(x)</code>	ritorna il valore massimo nel vettore x e, opzionalmente, la collocazione di questo valore in x
<code>min(x)</code>	ritorna il valore minimo nel vettore x e, opzionalmente, la collocazione di questo valore nel vettore
<code>mod(m,n)</code>	$\text{mod}(x,y)$ è $x - n \cdot y$ dove $n = \text{floor}(x./y)$ se $y \neq 0$
<code>round(x)</code>	approssima x all'intero più vicino
<code>rand(N)</code>	genera una matrice di NxN numeri casuali

Strutture di controllo in Matlab



```
if (<expr1>
    statement;
else if (<expr2>)
    statement;
else
    statement;
```

C

```
if <expr1>
    statements
elseif <expr2>
    statements
else
    statements
end
```

MATLAB

Strutture di controllo in Matlab

```
switch (<expr>)  
{  
    case <v1>: statement;  
        ...  
        break;  
    ...  
    case <vN>: statement;  
        ...  
        break;  
    default: statement;  
    ...  
}
```

C

```
switch <expr>  
    case <v1>  
        statement, ..., statement  
    case {<v2>, <v3>, ...}  
        statement, ..., statement  
    otherwise  
        statement, ..., statement  
end
```

MATLAB

Strutture di controllo in Matlab

```
for (<iniz>;<term>;<agg>)  
{  
    statement1;  
    ...  
    statementN;  
}
```

C

```
for variable = init:step:end  
    statement  
    ...  
    statement  
end
```

MATLAB

Strutture di controllo in Matlab



```
while (<expr>)  
{  
    statement1;  
    ...  
    statementN;  
}
```

C

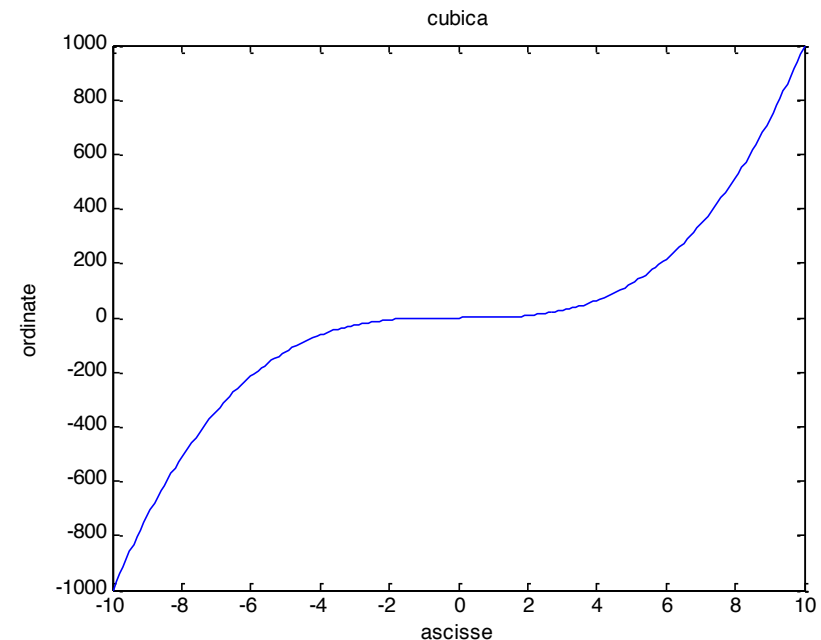
```
while <expr>  
    statement  
    ...  
    statement  
end
```

MATLAB

Diagrammi a due dimensioni

- ❑ Diagramma = insieme di coppie che rappresentano coordinate di punti
- ❑ Si usano vettori per contenere sequenze ordinate dei valori di ognuna delle coordinate
- ❑ `plot(x,y)` disegna digramma cartesiano dei punti che hanno
 - ▶ valori delle ascisse in x , delle ordinate in y
 - ▶ e li congiunge con una linea, per dare continuità al grafico
- ❑ funzioni `xlabel` per visualizzare nome asse ascisse, `ylabel` per ordinate, `title` per il titolo

```
>> x = -10:0.1:10;  
>> y=x.^3;  
>> plot(x,y);  
>> xlabel('ascisse');  
>> ylabel('ordinate');  
>> title('cubica');
```



Un esempio di cinematica

- ❑ Due treni partono da due stazioni adiacenti, che distano 15km, viaggiando a velocità di 50m/s e 30m/s in direzione opposta
- ❑ Costruire un grafico che mostra il loro movimento, fino a quando il più veloce raggiunge la destinazione
 - ▶ Il più veloce impiega $15000/50=300s$
 - $DistanzaTreno1=50 \cdot t$;
 - $DistanzaTreno2=15000-30 \cdot t$; (per mostrare la provenienza dalla direzione opposta)

Soluzione

```
t=0:1:300;
```

```
p1=50 * t;
```

```
p2=15000-30* t;
```

```
plot(t,p1);
```

```
hold on %adesso è possibile inserire nuove curve sul grafico
```

```
plot(t,p2)
```

```
hold off
```

Risultati ottenuti con l'esempio

