Fondamenti di Automatica

Introduzione a Matlab

Indice del materiale

- Descrizione generale di Matlab
- Alcune funzioni predefinite
- Definizione di matrici e vettori
- Definizione di polinomi
- Rappresentazione grafica dei dati
- Rappresentazione di sistemi dinamici lineari

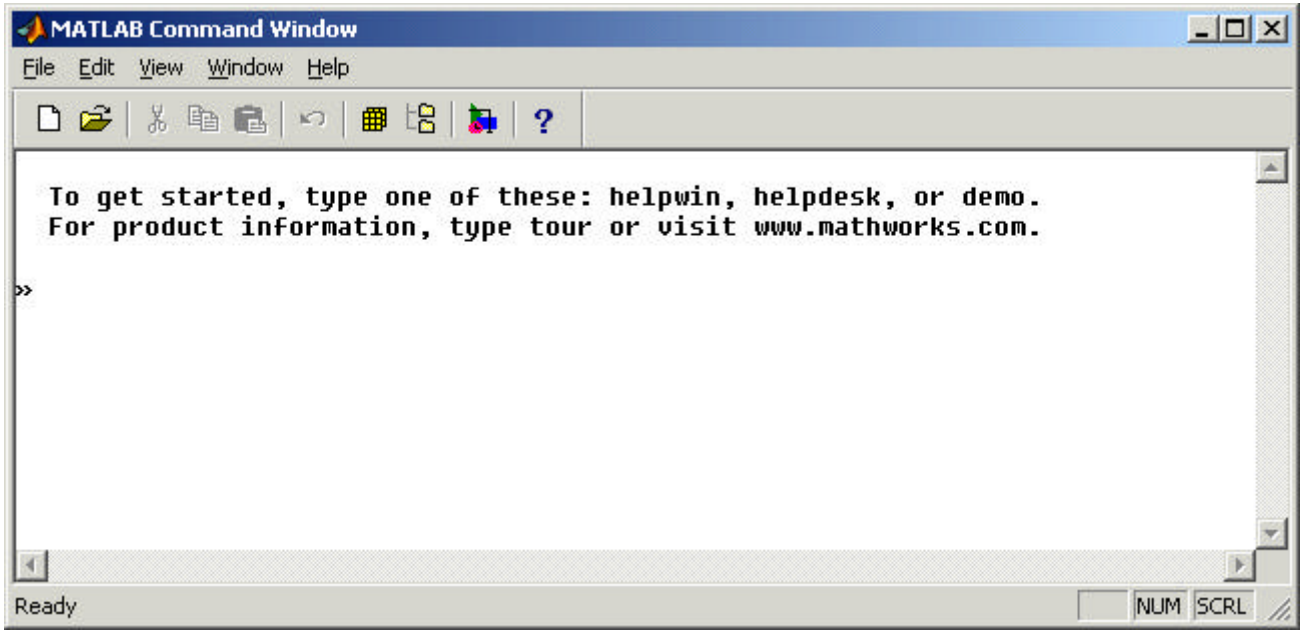
Descrizione generale di Matlab

MATLAB (= MATrix LABoratory)

- un linguaggio di programmazione per applicazioni scientifiche e numeriche
- vasto insieme di funzioni predefinite
- interprete di comandi
- possibilità di definire nuove funzioni
- libreria di TOOLBOX per svariate applicazioni (ad es. analisi dei sistemi e dei segnali, analisi e sintesi di sistemi di controllo, ecc.)

L’interfaccia di Matlab

Interfaccia utente: la Command Window dà accesso all’interprete mediante scrittura diretta dei comandi al prompt »



Matlab come calcolatrice

- La modalità di impiego più semplice è quella per valutare espressioni numeriche
- Esempio: per calcolare $4+\sqrt{2}-\sin(0.2p)^2+e^2$

```
» x=4 + sqrt(2) - sin(0.2*pi)^2 + exp(2)

    x =

       12.4578
```

- Il risultato viene scritto nella variabile x
- Aggiungendo ; (punto e virgola) alla fine del comando si evita la visualizzazione del risultato

Definizione di variabili

- E’ possibile calcolare espressioni algebriche in cui appaiono variabili già definite
- Esempio
  - » a=4; b=2;
  - » c=a*b

```
c =
     8
```

- Per cancellare una variabile (ad esempio a)
  - » clear a
- Per cancellare tutte le variabili
  - » clear all

Il Workspace

- Ogni variabile definita in questo modo viene conservata in memoria, nel Workspace
- Il comando whos mostra una lista delle variabili definite, con dimensioni e tipo

```
» whos
  Name        Size              Bytes  Class

  a           1x1                   8  double array
  x           1x1                   8  double array
  b           1x1                   8  double array

Grand total is 3 elements using 24 bytes
```

Funzioni e variabili

- Esiste un insieme molto vasto di funzioni predefinite (come sin, exp e sqrt nell'esempio precedente)

- A differenza di altri linguaggi (C, Pascal, …) non occorre dichiarare le variabili (l'assegnazione coincide con la dichiarazione)

Esempi di funzioni predefinite

- Funzioni trigonometriche (sin, cos, tan, acos, asin, atan, …)
- Esponenziale e logaritmo (exp, log, log10, …)
- Radice quadrata (sqrt)
- Numeri complessi (abs ➡ modulo, angle ➡ fase, real ➡ parte reale, imag ➡ parte immaginaria, …)
- L'unità immaginaria è indicata con i oppure j

Esempi di uso di funzioni predefinite

```
» x=abs(2+3*i)
x =
       3.6056

» y=20*log10(200)
y =
      46.0206

» z=sqrt(-4)
z =
      0 + 2.0000i
```

Un comando molto utile

help

- seguito dal nome di una funzione restituisce una sintetica descrizione e la sintassi d’uso

- “da solo” restituisce l’elenco di TUTTE le funzioni di Matlab, ordinate per categorie

Definizione di matrici

- Come si definisce una matrice in Matlab?

  Esempio

  » A=[1 , 2 ; 3 , 4]

  A =

  1 2

  3 4

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

- Come si accede agli elementi di una matrice?

  » x=A(2,1)

  x =

  3

indici (riga e colonna) dell'elemento di interesse

La wildcard :

- Per accedere a intere righe o colonne di una matrice, si usa la wildcard :
- Es.: selezionare la prima riga di A

```
» x=A(1,:)
x =
     1     2
```

- Es.: selezionare la seconda colonna di A

```
» y=A(:,2)
y =
     2
     4
```

Operazioni elementari sulle matrici (1)

- Sono definiti gli operatori +,-,*,^
- Esempio: prodotto di matrici

```
» A=[1 , 2 ; 3 , 4];
» B=[1 , 2 , 3 ; 4 , 5 , 6];
» C=A*B
C =
     9    12    15
    19    26    33
```

- Esempio: potenza di matrice

```
» A2=A^2
A2 =
     7    10
    15    22
```

Operazioni elementari sulle matrici (2)

- Matrice trasposta
  - » Atrasp=A’

```
Atrasp =
     1     3
     2     4
```

- Matrice inversa
  - » Ainv=inv(A)

```
Ainv =
   -2.0000    1.0000
    1.5000   -0.5000
```

Operazioni elementari sulle matrici (3)

- Dimensioni di una matrice
  - » dimA=size(A)

```
dimA =
     2     2
```

- Matrice con elementi unitari
  - » A1=ones(size(A))

```
A1 =
     1     1
     1     1
```

Operazioni elementari sulle matrici (4)

- Matrice identità
  - » Id=eye(size(A))

```
Id =
     1     0
     0     1
```

- Matrice nulla
  - » N=zeros(size(A))

```
N =
     0     0
     0     0
```

Operazioni elementari sulle matrici (5)

- Determinante
  - » dA=det(A)

    dA =

    -2
- Traccia
  - » tA=trace(A)

    tA =

    5

Operazioni elementari sulle matrici (6)

- Autovalori

```
» lambda=eig(A)
lambda =
          -0.3723
           5.3723
```

- Autovalori e autovettori

```
» [M,D]=eig(A)
M =
          -0.8246   -0.4160
           0.5658   -0.9094
D =
          -0.3723         0
                0    5.3723
```

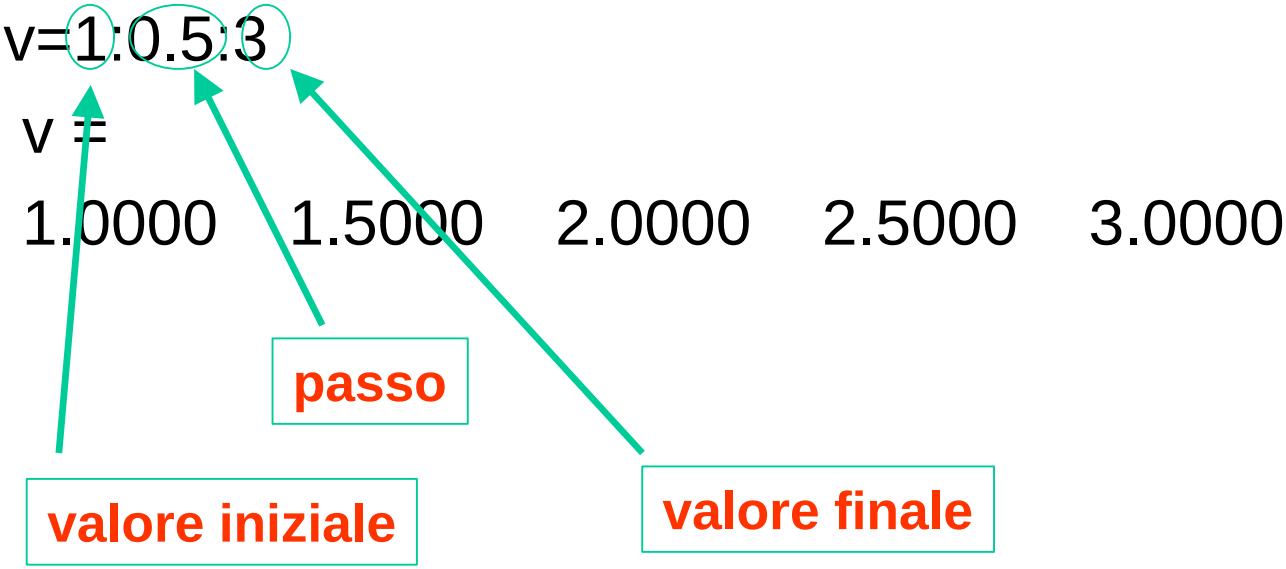
Vettori

- I vettori hanno due funzioni fondamentali in Matlab:
  - rappresentazione dei polinomi (un polinomio è descritto dal vettore dei suoi coefficienti)
  - rappresentazione di segnali (un segnale è rappresentato mediante la sequenza dei valori che assume in un insieme di istanti di tempo)

Definizione di vettori (1)

```
» v=0:10
  v =
        0     1     2     3     4     5     6     7     8     9    10

» v=1:0.5:3
   v =
   1.0000    1.5000    2.0000    2.5000    3.0000
```



Definizione di vettori (2)

- Come matrici riga o colonna
  - » v=[3 6 1 7]

```
v =
     3     6     1     7
```

  - » w=[3 6 1 7]’

```
w =
     3
     6
     1
     7
```

Polinomi

- I polinomi sono rappresentati come vettori

  » p=[3 2 1]

  $3s^2 + 2s + 1$

  p =

  3 2 1

- Operazioni con polinomi

  » y=polyval(p,x); valutazione in x

  » r=roots(p); radici

  » pc=poly(A); pol. caratteristico di A

Rappresentazioni grafiche

plot(x,y) traccia il grafico dei punti che hanno come ascisse e come ordinate gli elementi dei vettori x e y

- Esempio

```
» x=0:0.01:5;
» y=sin(2*pi*x);
» plot(x,y);
» z=sqrt(x);
» plot(x,y,'r',x,z,'b'); grid;
```

griglia

colore (r=rosso, b=blu)

Sistemi dinamici lineari

- Un sistema dinamico lineare invariante può essere descritto in forma di variabili di stato mediante il comando ss
  - a tempo continuo
    - » sistc=ss(A,B,C,D);
  - a tempo discreto
    - » sistd=ss(F,G,H,L,-1);

Esempi (1)

- Definizione del sistema

$$\dot{x}(t) = -x(t) + 3u(t)$$

$$y(t) = 4x(t) + 2u(t)$$

```
» A=-1;B=3;C=4;D=2;
» sistema=ss(A,B,C,D)
a =
           x1
    x1     -1
b =
           u1
    x1      3
c =
           x1
    y1      4
d =
           u1
    y1      2

Continuous-time model.
```

Esempi (2)

- Definizione del sistema

$$\dot{x}(t) = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} 1 & 1 \end{bmatrix} x(t)$$

```
» A=[1 2 ; 3 4];
» B=[1 0]';
» C=[1 1];
» sistema=ss(A,B,C,0);
```

Esempi (3)

- Definizione del sistema

$$x_{k+1} = 0.5x_k + 2u_k$$

$$y_k = -x_k + 4u_k$$

```
» F=0.5;G=2;H=-1;L=4;
» sistema=ss(F,G,H,L,-1)
```

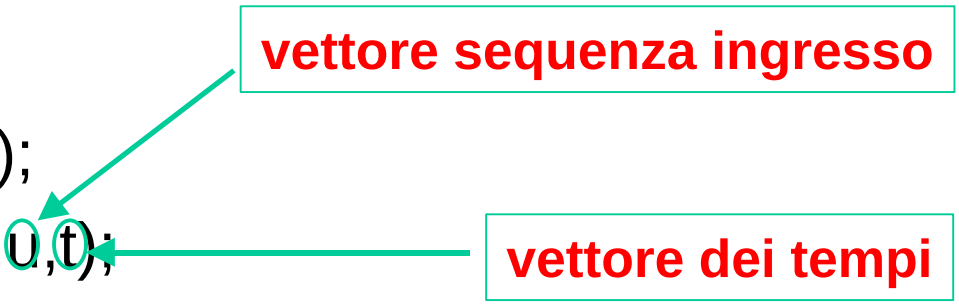
per sistemi a tempo discreto

Simulazione di sistemi lineari

- Funzioni disponibili per la simulazione
  - impulse -> simulazione risposta all'impulso
  - step -> simulazione risposta allo scalino
  - initial -> simulazione movimento libero
  - lsim -> simulazione con ingresso qualsiasi e stato iniziale qualsiasi
- Sintassi

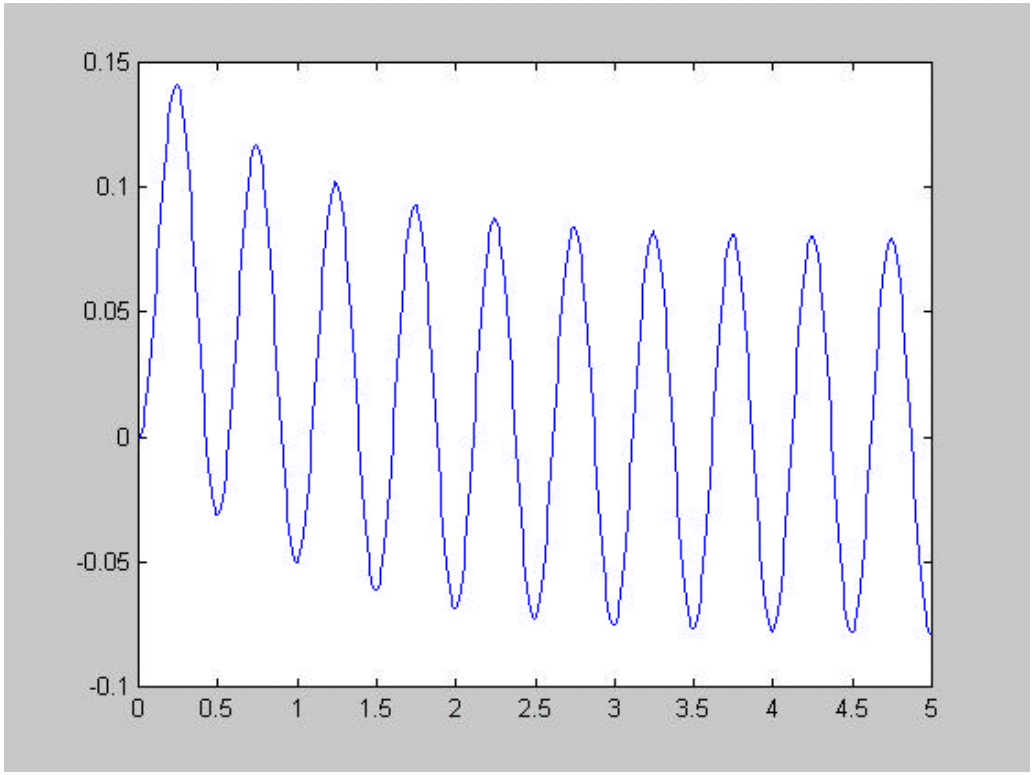
» [y,t,x]=step(sistema);

» [y,t,x]=lsim(sistema,u,t);



Esempio

```
» sistema=ss(-1,1,1,0);
» t=(0:0.01:5);
» u=sin(4*pi*t);
» y=lsim(sistema,u,t);
» plot(t,y)
```



- Chiamando le funzioni di simulazione senza output si ottiene direttamente il grafico