Introduzione a MATLAB

2° MODULO

Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Padova

STRUTTURE DI PROGRAMMAZIONE

STRUTTURA IF … (ELSEIF …) ELSE … END

if *espressione*
istruzioni
(**elseif** *espressione*
istruzioni)
else
istruzioni
end

espressione normalmente è costruita con operatori relazionali: : >, >=, ==,…..

elseif è opzionale

Esempio:

```
if rem(x,2)==0          → rem=resto della divisione
        disp(['il numero  ', num2str(x) , ' e ' ' pari '])
else
        disp(['il numero  ', num2str(x) , ' e ' ' dispari '])
end
```

CICLO FOR … END

for *variabile* = *espressione*
    istruzioni
end

espressione tipicamente è:
imin:incremento:imax
(per passo unitario imn:imax)

Esempio

```
somma=0;
v=rand(1,20);
for ii=1:length(v)
 somma=somma+v(ii);
end
disp(['La somma degli elementi di v vale:  ', num2str(somma)])
```

Controllare con sum(v(:))!!

CICLO WHILE … END

while *espressione*
    istruzioni
end

espressione normalmente è costruita con operatori relazionali: : >, >=, ==,…..

Esempio

```
v=rand(1,20)
somma=0;
ii=1;
while ii<=length(v)
    somma=somma+v(ii);
    ii=ii+1;
end
disp(['La somma degli elementi di v vale:  ', num2str(somma)])
```

OSSERVAZIONI

- I cicli possono essere uno dentro l'altro.

  Ad es. per il ciclo for:

```
for  R = 1:M
     for C = 1:N
          B(R,C) = A(R,C)*A(R,C);
     end
end
```

Calcola A.^2

- In questi casi per velocizzare la procedure meglio fare una preallocazione dello spazio per le variabili

```
B=zeros(size(A))
```

- Per bloccare l'esecuzione di un ciclo while o for si può usare l'istruzione

```
break
```

GRAFICI

PLOT: grafici lineari

plot(y)	plotta gli elementi del vettore y rispetto agli indici del vettore stesso
plot(x,y)	plotta gli elementi del vettore y rispetto a quelli del vettore x (cioè x sono i dati delle ascisse e y quelli delle ordinate)

•Si possono usare vari tipi e colori di plot, basta specificarlo fra apici dopo i dati: plot(x,y, 'r+:') → disegna i dati in rosso marchiati con un + e interpolati con linea tratteggiata)

•Si possono plottare più linee nella stessa figura o con un unico plot (plot(x1,y1,x2,y2,….) o usando la funzione hold on (si annulla con hold off)

•Per aprire una finestra grafica si deve usare il comando figure, se tale comando manca la prima volta si apre in automatico dopo si riscrive sulla stessa (cancellando il grafico precedente)

PLOT: grafici lineari (continua)

- Gli assi vengono impostati automaticamente, per modificarli si può usare il comando axis
- Un grafico per essere leggibile deve avere:
  - Un titolo → title
  - Un'etichetta sull'asse x → xlabel
  - Un'etichetta sull'asse y → ylabel
  - Se si devono aggiungere note → text

Esempio:

```
x=[0:0.1:2];
y1=2*exp(x)+3;
y2=3*exp(x)-2;
plot(x,y1,'ro:',x,y2,'c*-')
title('esponenziali')
xlabel('tempo')
ylabel('valori')
text(0.6,6,'prima')
text(0.6,2,'seconda')
```

Diesgno due esponenziali
Nella stessa figura
$y1=2e^{x}+3$
$y2=3e^{x}-2$

PLOT: grafici lineari (continua)

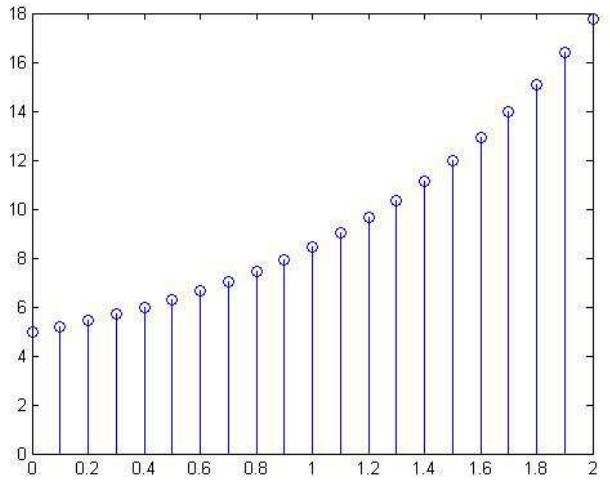
- Se devo mettere apici/pedici nei titoli o nelle label
  - _{a} → a è pedice
  - ^{b} → b è apice
- Se devo scrivere lettere greche basta precedere la scritta da \
  - \beta → β
  - \alpha → α
- Se voglio fare più grafici separati nella stessa figura:
  - subplot(m,n,p) divide la finestra in mxn parti e disegna nella p-esima

Esempio:

```
x=[0:0.1:2];
y1=2*exp(x)+3;
y2=3*exp(x)-2;
subplot(1,2,1)
title(‘prima esponenziale')
plot(x,y1,'ro:’)
subplot(1,2,2)
title(‘seconda esponenziale')
plot(x,y2,'ro:’)
```

STEM PLOT:

stem(x,y): plot per sequenze discrete



semilogx(x,y) : come plot ma l'asse x viene rappresentato in scala log10

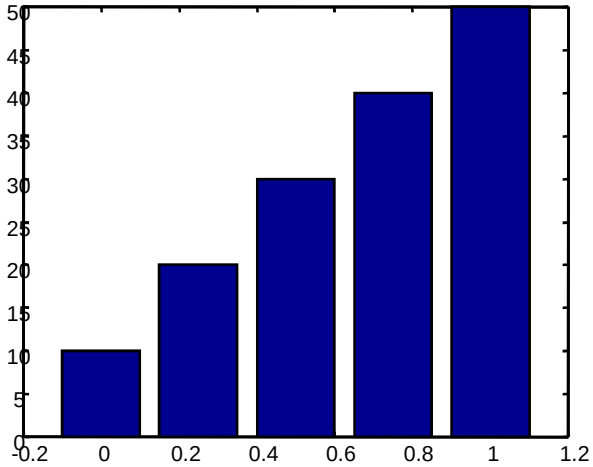
semilogy(x,y) : come plot ma l'asse y viene rappresentato in scala log10

loglog(x,y) : come plot ma con entrambi gli assi in scala log10.

GRAFICI A BARRE

bar(x,y) : produce un diagramma a barre.

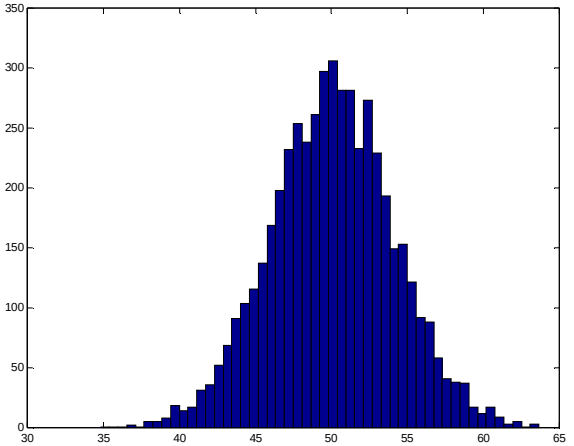
bar([0:0.25:1],[10:10:50])



ISTOGRAMMA

hist(y,m): suddivide l'intervallo dei valori compresi tra il minimo e il massimo di y in m "bin" (=sottointervalli) di egual larghezza e calcola (e poi disegna) il numero di elementi di y compresi in ogni bin

hist(50+4*randn(1,5000),50)



CREAZIONE DI FUNCTION

STRUTTURA DI UNA FUNCTION

- La prima riga di una function DEVE sempre essere:

  function [out1,out2,….] = nomefunction(in1,in2,….)

- La function deve essere salvata in un file con lo stesso nome della function stessa (cioè nomefunzione.m)

- Dopo la prima riga è opportuno inserire dei commenti che faranno parte dell'help on-line

- Seguono le istruzioni con eventuali altri commenti

- Le function vengono poi richiamate dal programma main passando le variabili di ingresso (anche con nome diverso !)

Esempio

```
function a=radice(x)
% RADICE(X) calcola la radice degli elementi di X
% se X>=0, altrimenti stampa un messaggio di errore
if x>=0
  a=sqrt(x)
else
  disp('errore')
end
```

Programma che richiama la function

```
b=input('inserisci un numero: ')
risul=radice(b);
disp(['la radice vale: ', num2str(risul)])
```

Esempio

```
function [xmin,xmax]=minmax(a)
%MINMAX(A) calcola l'elemento  minimo, XMIN, e l'elemento
% massimo, XMAX della matrice A.
xmin=Inf; xmax=-Inf;
[m,n] = size(a);                    %calcola le dimensioni di a
for i=1:m
   for j=1:n
      if a(i,j) > xmax
         xmax = a(i,j);
      end
      if a(i,j) < xmin
         xmin = a(i,j);
      end
   end
end
```

ESEMPIO

```
function y=mia_funz1(x)
%function y=mia_funz1(x)
% questa e' una function che restituisce il
% valore di una_funzione y=mia_funz1(x),
% scalare di variabile scalare,
% che definisco sotto

if length(x)==1
        contributo1=exp(-2*x);
        contributo2=exp(-5*x);
        y=contributo1+contributo2;
else
   error('la funzione mia funz1 accetta solo argomenti scalari')
end
```

Programma "main"

```
% esempio di programma main che richiama
% una function da me costruita di nome mia_funz1
t=(0:0.1:10)';
for k=1:length(t)
   h(k,1)=mia_funz1(t(k));
end
plot(t,h)
```