Funzioni definite dall'utente: sintassi

Definizione di funzione:

In un M-file, il cui nome è (bene che sia…vedi diapo successive): "**funcion_name.m"**

Variabili di uscita.
Se è una sola variabile, le
[, e] non sono necessarie

Nome della funzione

Variabili di ingresso.

```
function [var_out1,var_out2,….,var_outn] = function_name (var_in1,var_in2,….,var_inn)
Command1;
Command2;
….
Commandn;
```

Corpo della funzione

Chiamata di funzione:

Nell'M-file con il programma principale

[v1,v2,….,vn] = function_name (in1, in2,…., inn)

N.B.: Nome dell'M-file in cui è definita la funzione

Funzioni definite dall'utente: esempio

Definizione di funzione:

```
function y =seno_in_gradi(x)
% seno_in_gradi calcola il seno di un angolo in gradi
tmp = x*pi/180;   %N.B.: tmp è una variabile LOCALE
y = sin(tmp);
```

Il nome dell'M-file con la definizione, è (bene che sia… vedi diapo successive):
seno_in_gradi.m

Chiamata di funzione:

```
% script che richiama la funzione seno_in_gradi, per diversi valori
% dell'angolo in gradi
x = input('valore dell''angolo in gradi:');
s = seno_in_gradi(x);
disp(['il seno di ' num2str(x) '° e'': ' num2str(s)])
```

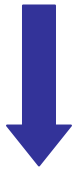
Funzioni definite dall'utente: regole

1. **Chiamata di funzione:**

   Una funzione definita dall'utente è chiamata utilizzando ***il nome dell'M-file in cui questa è definita non il nome della funzione, dato nella prima riga di definizione***

   Se la funzione seno_in_gradi dell'esempio precedente venisse salvata in un M_file con nome "***sin_in_degree.m***", lasciando invariate le linee di codice,(e quindi anche il nome della funzione: ***seno_in_gradi***), nell'M-file principale (per esempio "main.m") dovremo fare la seguente chiamata di funzione:

   y = sin_in_degree(x);



Per comodità, e per evitare errori, è conveniente dare lo stesso nome alla funzione e all'M-file!

2. Commenti:

Nella definizione di funzione, le righe dopo la prima, dovrebbero essere commenti, che spiegano cosa fa la funzione che si sta definendo.

```
function y =seno_in_gradi(x)

% seno_in_gradi calcola il seno di un angolo in gradi

tmp = x*pi/180;   %N.B.: tmp è una variabile LOCALE
y = sin(tmp);
```

Tale commento può essere visto, utilizzando il comando "help nome_funzione":

```
>> help seno_in_gradi

seno_in_gradi calcola il seno di un angolo in gradi
```

nella Command Window

3. Informazioni in uscita:

La sola informazione in uscita alla chiamata di funzione, sono i valori delle variabili di uscita.

Nel corpo della funzione deve sempre esserci *almeno un comando che assegna un valore alle variabili di uscita* specificate nella prima linea di definizione.

Le variabili di uscita sono opzionali: si possono definire funzioni che eseguono operazioni, ma non restituiscono informazioni (per esempio “plot” o “disp”)

4. **“comunicazione”:**

Una funzione comunica con il workspace del Matlab solo attraverso le variabili di ingresso e di uscita.

Le variabili ‘intermedie’, cioè definite ed utilizzate nel corpo della funzione, non appaiono e non interagiscono con il Workspace principale del Matlab:

ogni funzione ha un suo Workspace, separato dal Workspace principale.

Le variabili presenti nel Workspace locale si dicono, appunto **Variabili Locali**

```
function y =seno_in_gradi(x)
% seno_in_gradi calcola il seno di un angolo in gradi
tmp = x*pi/180;   %N.B.: tmp è una variabile LOCALE
y = sin(tmp);
```

N.B.: molti comandi Matlab sono scritti come <u>M-files funzione.</u>

Si può conoscere la cartella in cui tali M-files sono salvati mediante il comando: ***which:***

```
>> which linspace   % comando per conoscere in quale cartella è l'M-file linspace.m
C:\Programmi|MATLAB701|toolbox\matlab\elmat\linspace.m
```

Si può vedere il contenuto dell'M-file funzione con il comando ***type***

```
>>type linspace
function y = linspace(d1, d2, d3,n)
.....
```

Grafici di funzioni y=f(x): *fplot*

- **Una funzione f(x), come per esempio un polinomio, può essere valutata e poi se ne può fare il grafico, utilizzando la funzione Matlab "plot".**
- **<u>In alternativa</u>:**
  la funzione Matlab "*fplot*", permette di eseguire le due operazioni dette sopra (valutazione e grafico), in un unico passo:

fplot(fun,lims) fa il grafico della funzione specificata dalla stringa fun e calcolata per valori di ascissa compresi tra lims = [xmin xmax].

Utilizzando lims = [xmin xmax ymin ymax] si controllano anche i valori delle ordinate.

Esempio

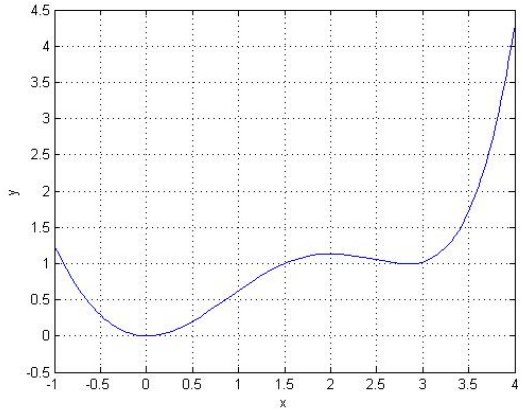
M-file: fp5.m

```
function y = fp5(x)

% FP5 calcola la seguente funzione di 5° grado:
% f = 0.025x^5 -0.0625x^4 -0.333x^3+x^2

y = 0.025*x.^5 -0.0625*x.^4 -0.333*x.^3+x.^2;
```

M-file: main1_fp5.m

```
% script principale per la chiamata e la costruzione del
% grafico della funzione FP5

fplot('fp5',[-1 4 -0.5 4.5]), xlabel('x'), ylabel('y'),grid;
```

Stringa con nome della funzione



Esempio: mostra nel grafico il minimo locale della funzione

M-file:
main2_fp5.m

```
%  script principale per la chiamata e la costruzione del grafico
%  della funzione FP5, e per il calcolo del minimo relativo
%   per -1<=x<=4

figure;
fplot('fp5',[-1 4]), axis([-1 4 -0.5 4.5]), title('grafico di una funzione'),...
   xlabel('x'), ylabel('y'),grid on;

pause;

% calcolo del  minimo locale e visualizzazione sul grafico
xmin = fminbnd('fp5',-1,4);
ymin = fp5(xmin);

hold on;
text(xmin,ymin-.2,'min');
```

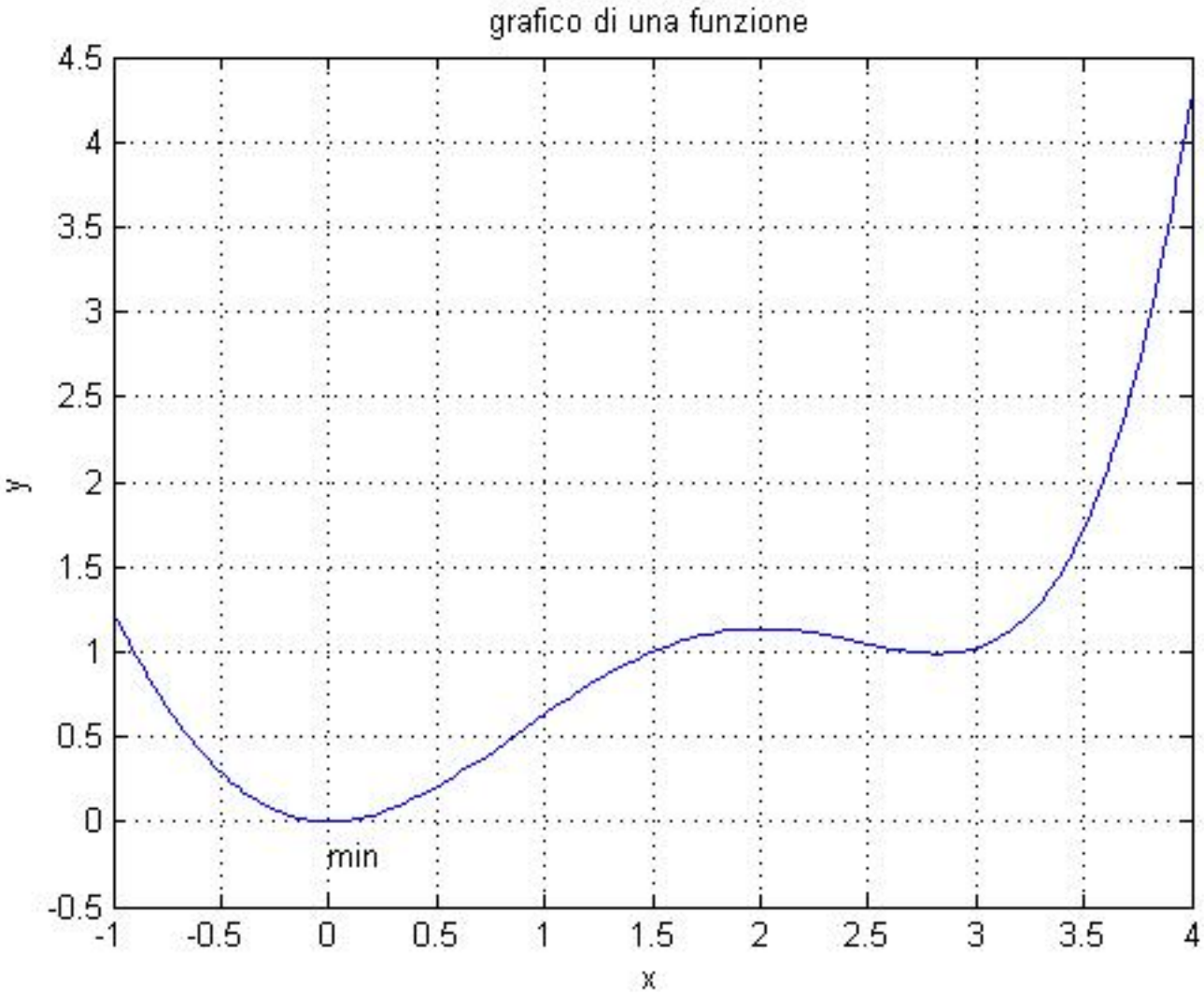


grafico di una funzione
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
y
min
-1
-0.5
0
0.5
1
1.5
2
2.5
3
3.5
4
x