Introduzione al MATLAB ©
Parte 3
Script e function

Lucia Gastaldi

DICATAM - Sezione di Matematica,
http://www.ing.unibs.it/gastaldi/

Indice

Script e Function

Due tipi di file:

Script

- Opera sui dati presenti in Workspace.
- Non accetta variabili in input.
- Non ha variabili di output.
- Utile per automatizzare una serie di istruzioni che si devono eseguire più volte.

Function

- Le variabili interne sono locali.
- Può accettare variabili in input.
- Può avere variabili in output.
- Utile per estendere il linguaggio MATLAB alle applicazioni personali.

Contenuto di un file: **script** o **function**

- Chiamate di un'altra function;
- Cicli `for` oppure `while`;
- `if, elseif, else`;
- Input/Output interattivi;
- Calcoli;
- Assegnazioni;
- Commenti;
- Linee bianche;
- Comandi per la costruzione di grafici.

M-file di tipo **script**

Esempio: `alglin.m`

```
%  Risoluzione di un sistema lineare      % Commento
%    e calcolo dell'errore relativo
%  A  Matrice di Hilbert
%
%      Inizio istruzioni
A=hilb(n);                                % Calcolo
x=[1:n]';
b=A*x;
x1=A\b;
errore=norm(x-x1)
errorerel=errore/norm(x)
```

Caratteristiche di un file di tipo **script**

- È il tipo più semplice di M-file perchè non ha variabili di input e output.
- Serve per automatizzare una serie di comandi MATLAB che devono essere eseguiti più volte.
- Opera sui dati esistenti nell'ambiente di lavoro di base.
- I dati che vengono generati rimangono nell'ambiente di lavoro di base e possono essere riutilizzati per altri calcoli.

Come si usa

Basta scrivere il nome del file sulla riga di comando senza l'estensione `.m`.

```
>> miofile
```
return

M-file di tipo **function**

Esempio: `errsl.m`

```
function [errore,errrel] = errsl(n)  % Riga di definizione
                                     %    della function
%  ERRSL errore per  sistema lineare % Riga H1
%  Risoluzione di un sistema lineare % Testo per help
%    e calcolo dell'errore relativo
%  A  Matrice di Hilbert
%
%  Inizio istruzioni della function
A=hilb(n);                           % Corpo della function
x=[1:n]';
b=A*x;
x1=A\b;
errore=norm(x-x1);
errrel=errore/norm(x);
```

function

Riga di definizione

```
function [output] = nome_ function (input)
```

Output

una sola variabile in uscita x: `[output]` $\longrightarrow$ `x`

più variabili in uscita x, y, z: `[output]` $\longrightarrow$ `[x,y,z]`

nessuna variabile in uscita: `[output]` $\longrightarrow$ `[ ]`

Input

Le variabili in input possono essere `array` (scalari, vettori, matrici) ma anche il nome di altre `function`:

```
function [t,y] = ode23(f,[t0,tf],y0)
function [t,y] = ode23(@f,[t0,tf],y0)
```

Come si usa una **function**

```
>> [lista output]= miofile(dati in input)
```

Nota bene

- I dati **devono** essere passati alla function, qualora richiesti, perchè la function **NON** lavora sulle variabili nel Workspace.
- I valori dei risultati prodotti dalla function **devono** essere assegnati a delle variabili, altrimenti i risultati **NON** sono disponibili nel Workspace.
- Non è necessario assegnare alle variabili in uscita lo stesso nome che hanno nella function.
- I dati possono anche essere assegnati con valori numerici.

Esempi

- Se uso la function come uno script:

```
>> errsl
???  Input argument "n" is undefined.
```

- Se non assegno il risultato alle variabili di output

```
>> errsl(5)

ans =
   3.1881e-11
```

- Uso corretto

```
>> [assoluto,relativo]=errsl(8)

assoluto =
   4.4273e-07

relativo =
   3.0997e-08
```

function

Riga H1

È la prima riga del testo di help.
Siccome è una riga di commento inizia con %

Testo di help.

Si può creare un aiuto in linea per la propria `function` introducendo una o più righe di commento immediatamente dopo la riga H1.
>> `help` *nome_ function*
MATLAB scrive le righe di commento che ci sono fra la riga di definizione della function e la prima riga che non è di commento.

function

Corpo della function

Contiene le istruzioni per il calcolo e l'assegnazione dei valori alle variabili di output.

Commenti

- Le righe di commento iniziano con %
- Si possono inserire righe di commento in qualsiasi punto della function.
- Si possono aggiungere commenti alla fine di una riga del codice.

Esempio

```
% Somma di tutti gli elementi di un vettore.
y = sum(x)          % Usa la function sum
```

input

Il comando `input` serve per assegnare interattivamente un dato dalla tastiera.

```
var=input('Dammi il valore di var ')
```

Durante l'esecuzione di un programma, questa istruzione interrompe l'esecuzione fintanto che non viene introdotto da tastiera il valore che si vuole assegnare alla variabile `var`. Nell'angolo in basso a sinistra della finestra di Matlab appare la scritta Waiting for input.

Comando da usare con parsimonia

Se si hanno tanti dati da assegnare ogni volta è meglio usare una function.

display e error

Il comando display può essere utile per fare comparire sullo schermo un risultato oppure un messaggio di attenzione.

- `display('messaggio')` Per scrivere una stringa
- `display(['valore x= ',num2str(x)])` Per riportare il valore della variabile `x`. Il comando `num2str(x)` converte il valore di `x` in una stringa.

Il comando `error` dà una segnalazione di errore che compare sullo schermo in rosso e interrompe l'esecuzione.

```
error('Hai fatto un errore')
??? Hai fatto un errore
```