Introduzione a Matlab 7

File .m

- Un file .m (M-file) è un programma riconoscibile da Matlab
- La scrittura di files .m permette di:
  - Sperimentare un algoritmo senza dover ripetere i comandi
  - Poter riutilizzare il programma
  - Scambiare programmi con altri utenti

Struttura di un file .m

- Due tipi di .m file

- **Script**: contengono comandi, non hanno variabili in entrata e in uscita e operano sulle variabili del workspace

- **Function**: contengono comandi ed hanno argomenti in entrata e in uscita. Le variabili interne a questi programmi non influenzano le variabili del workspace

Commenti

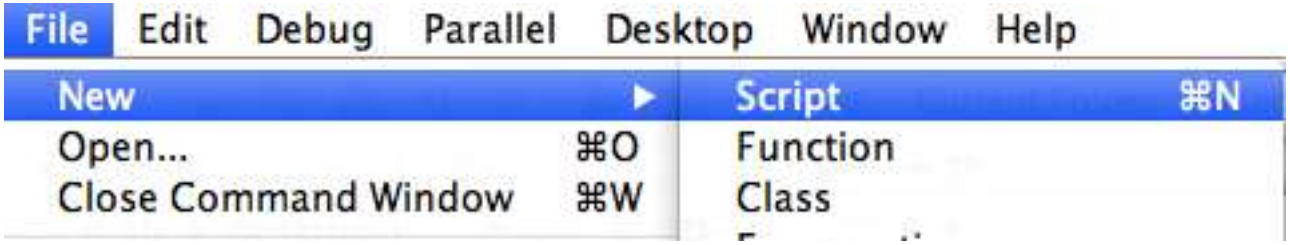
- E’ opportuno che script e function siano opportunamente commentati, ovvero contengano delle spiegazioni di ciò che si sta facendo
- Le righe di commento iniziano con il carattere %: Matlab ignora tutti i caratteri di una riga successivi al %
- Le prime righe di commento inserite in uno script o in una function sono interpretate e visualizzate da Matlab come help online (output del comando help)

Script

- Contengono una sequenza di istruzioni nella forma in cui si scriverebbero nella command window
- Possono utilizzare le variabili già definite nel workspace o definirne altre
- Al termine dell'esecuzione, tutte le modifiche sono visibili all'esterno
  - In questo senso si dice che tutte le variabili sono globali
- Per eseguirli è sufficiente digitare, nella command window, il nome del file (senza l'estensione .m)
  - devono essere salvati nella "Current Directory" (directory corrente)

Creazione script

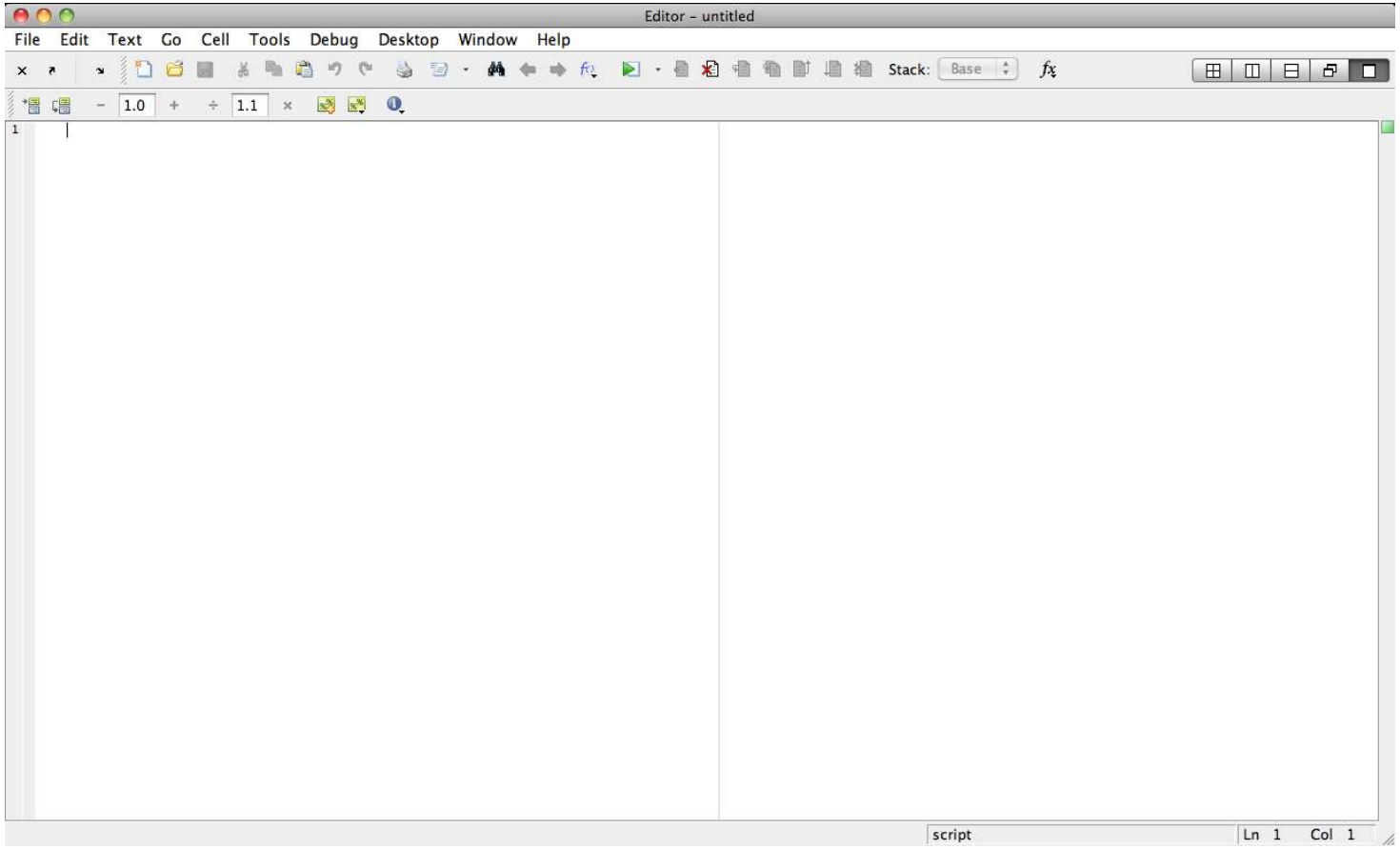
- Per creare uno script si può procedere in due modi
  1. Dal menu File -> Nuovo -> Script



  2. Dalla barra dei menu, mediante l'icona

Creazione script

- Si apre una finestra dell'editor in cui è possibile inserire i comandi

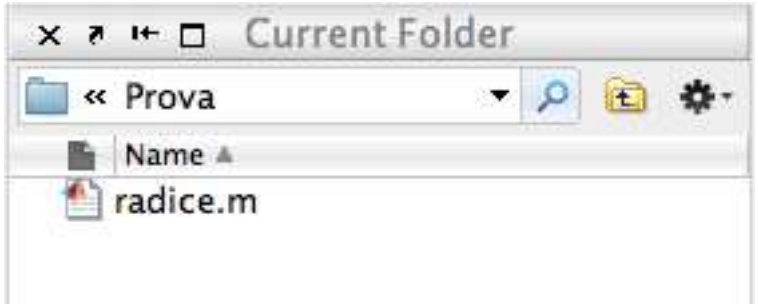


Esempio: radice.m

- Nella finestra dell'editor inserire il codice riportato sotto

```
% Questo file calcola la radice degli elementi di
% una matrice a, se a>0, altrimenti da' un messaggio di errore
if a>=0
  sqrt(a)
else
  display('errore')
end
```

- Salvare il file con il nome radice.m



Esempio: radice.m

- Dichiarare la variabile ‘a’ nel workspace

```
>>a=4;
>>
```

- Lanciare lo script digitando il nome dello script senza .m

```
>>radice
ans =
     2
>>
```

Esempio: minmax.m

- Nella finestra dell'editor inserire il codice riportato sotto

```
% Calcola l'elemento  minimo, XMIN, e l'elemento
% massimo, XMAX della matrice A.
xmin=Inf; xmax=-Inf;
% ricava le dimensioni della matrice A:
[m,n] = size(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
xmin
xmax
```

- Salvare il file con il nome minmax.m

Come funziona minmax.m?

- Le prime due righe sono commenti e poiché sono le prime righe, sono interpretate anche come help

```
% Calcola l'elemento  minimo, XMIN, e l'elemento
% massimo, XMAX della matrice A.
```

- Inizializziamo due variabili xmin e xmax rispettivamente a Inf (infinito) e -Inf (- infinito)

```
xmin=Inf; xmax=-Inf;
```

- Ricaviamo le dimensioni della matrice A utilizzando la funzione size che restituisce il numero di righe (m) ed il numero di colonne (n) di A (nel caso in esame, m=3 e n=3)

```
% ricava le dimensioni della matrice A:
[m,n] = size(A);
```

Come funziona minmax.m?

- Il primo ciclo for serve ad enumerare gli indici di riga (da 1 fino a m)
- Il secondo ciclo for serve ad enumerare gli indici di colonna (da 1 fino a n)

```
for i=1:m
    for j=1:n
        ...
        ...
    end
end
```

- Ogni coppia (i,j) indicizza una cella della matrice A
  - (1,1) (1,2) (1,3)
  - (2,1) (2,2) (2,3)
  - (3,1) (3,2) (3,3)

Come funziona minmax.m?

- Per ogni coppia (i,j), si accede alla cella A(i,j)
- Se il contenuto della cella A(i,j) è maggiore del contenuto della variabile xmax

```
if A(i,j) > xmax
        ...
end
```

- aggiorna il contenuto di xmax (nuovo valore massimo)

```
...
        xmax = A(i,j);
...
```

- Se il contenuto della cella A(i,j) è minore del contenuto della variabile xmin, aggiorna il contenuto di xmin (nuovo valore minimo)

```
if A(i,j) < xmin
        xmin = A(i,j);
end
```

Esecuzione di minmax.m

- Creare una matrice A nel workspace

```
>>A=rand(3,3)
A =
    0.8147 0.9134 0.2785
    0.9058 0.6324 0.5469
    0.1270 0.0975 0.9575
```

- Lanciare lo script digitando il nome dello script senza .m

```
>>minmax
xmin =
     0.0975
xmax =
     0.9575
```

Esempio: sufficienze.m

Dato un vettore contenente una serie di voti da 0 a 30, contare quante sono le sufficienze (cioè i voti maggiori o uguali a 18)

Esempio: sufficienze.m

- Nella finestra dell'editor inserire il codice riportato di seguito

```
voti = [13 19 17 24 30 11 16];
sufficienze = 0;
for i = 1 : length(voti)
        if voti(i) >= 18
                sufficienze = sufficienze + 1;
        end
end
sufficienze
```

- Salvare il file con il nome sufficienze.m

Come funziona sufficienze.m?

- Creiamo un vettore (voti) contenente i voti da analizzare

```
voti = [13 19 17 24 30 11 16];
```

- Inizializziamo una variabile contatore (suff) a 0

```
suff = 0;
```

- Il ciclo for serve ad enumerare gli indici del vettore (da 1 fino alla dimensione di voti). Utilizziamo la funzione length per calcolare la dimensione di voti, ovvero il numero di elementi

```
for i = 1 : length(voti)
        ...
        ...
end
```

Come funziona sufficienze.m?

- Per ogni cella del vettore, voti(i), verifichiamo se il valore contenuto è maggiore o uguale a 18
- In caso affermativo, incrementiamo la variabile contatore di una unità

```
...
        if voti(i) >= 18
                suff = suff + 1;
        end
...
```