Michele Ventrone

INTRODUZIONE
A
GNU Octave

Versione 2.9.13

GUIDA RAPIDA

luglio 2008

rev. novembre 2011

INDICE

mike_ven_s.maria.c.v@libero.it

9. - Alcune funzioni

mike_ven_s.maria.c.v@libero.it

mike_ven_s.maria.c.v@libero.it

1. - Per iniziare

1.1 - Introduzione

In questa guida sono discussi solo alcuni comandi e alcune funzioni utili agli studenti degli ultimi anni di corso delle scuole medie superiori e agli studenti universitari del primo anno di facoltà scientifiche e si rimanda, per ogni approfondimento, al testo in formato *pdf* del Prof. John W. Eaton, che si trova nella sezione "*documentazione*" di Octave. Queste pagine possono costituire un ausilio didattico nei corsi introduttivi di algebra lineare, di analisi, di calcolo numerico, di programmazione e statistica. Chi ha già affrontato questi corsi impiegherà pochissimo tempo ad usare Octave.

1.2 - Che cos'è GNU Octave

Octave è un software open-source scritto da John W. Eaton e molti altri. Octave non è di pubblico dominio perché esistono delle restrizioni sulla sua ridistribuzione. Queste restrizioni sono pensate per assicurare a tutti di avere gli stessi diritti di usare e ridistribuire Octave. Ciò significa che chiunque può ottenere e usare il codice sorgente, può modificarlo e ridistribuirlo nei termini della licenza GNU General Public Licence come descritta e pubblicata dalla Free Software Foundation in cui sono espresse le precise condizioni sul copyright.

Octave è un linguaggio interpretato ad alto livello orientato al calcolo numerico: moltiplica e inverte matrici, determina radici di equazioni lineari e non lineari, manipola polinomi, integra, differenzia e disegna grafici.

Il suo insieme di funzioni ne comprende molte orientate all'ottimizzazione, alla statistica, alla matematica finanziaria, all'interpolazione, alla teoria del controllo e allo studio dell'audio e delle immagini.

Octave, in linea di massima, è compatibile con Matlab di Mathworks (laboratorio di matrici) ma non permette la manipolazione simbolica degli oggetti matematici.

1.3 - Installiamo Octave

All'indirizzo http://www.octave.org potrete trovare una versione di Octave per il vostro sistema operativo. L'autore di questo manuale ha provato Octave con Windows XP. Quando il file eseguibile sarà sulla vostra macchina fatelo partire. Non variate le eventuali opzioni proposte, lo potrete fare dopo, quando sarete più esperti. L'installazione si completerà in pochi minuti.

1.4 - Lanciamo Octave

Dopo l'installazione cliccate sull'icona di Octave (fig.1) che si trova sul desktop o cercatela nell'elenco *GNU OCTAVE* raggiungibile da *start* e *Tutti i programmi*.

figura 1 - Icona del programma Octave, versione 2.9.13.

All'avvio apparirà in una finestra la versione di Octave, il nome dell'autore, l'avviso che il software è free e senza garanzia, l'indirizzo del sito ove cercare altre informazioni e il prompt *octave.exe:1>* che attende l'immissione del primo comando (fig.2).

mike_ven_s.maria.c.v@libero.it

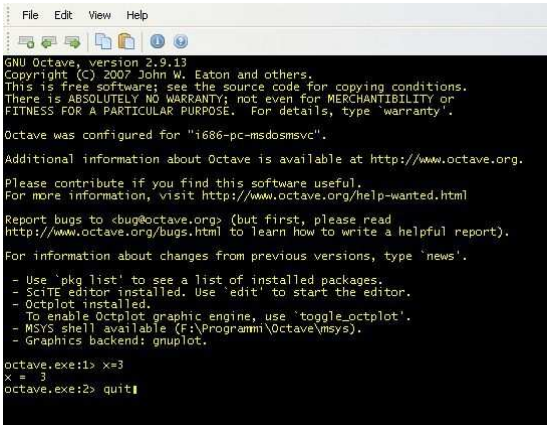


figura 2 - Octave al primo avvio.

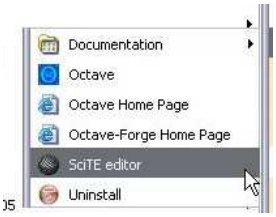
Digitate *x=3* e terminate l'immissione con ***INVIO*** (o ***ENTER***). Il sistema risponderà con *x=3* (fig.2). Ecco la successione delle operazioni illustrate:

```
octave.exe:1> x=3
x  =  3
octave.exe:2>
```

Con il prompt *octave.exe:2*> il sistema attende il secondo comando. Per terminare la sessione di lavoro digitare ***quit*** oppure ***exit*** e poi ***INVIO*** (fig.2). E' consigliabile chiudere Octave sempre in questo modo per evitare eventuali segnalazioni di errore da parte di Windows.

1.5 - SciTe

Octave ha un editor potente che si adatta al formato di moltissimi linguaggi di programmazione.



Vedere la sezione "Scrivere una funzione".

2. - Il Menu di Octave

Nella barra del menu (fig.1) di Octave si trovano le voci *File*, *Edit*, *View* e *Help*, e nella *Toolbar* sottostante trovano posto le icone *NewTab*, *Previouse tab*, *Next tab*, *Copy Paste*, *Abaut*, *Help*.

2.1 - File

Questa voce del menu permette di creare una nuova finestra di lavoro, di chiudere la finestra attuale di lavoro, aprire un prompt di comando e di uscire da Octave. Con la successione *File-New Tab-Octave* (fig.2) si crea una nuova finestra di lavoro di Octave. Lo stesso risultato lo si ottiene cliccando nella *Toolbar* sulla prima icona a

mike_ven_s.maria.c.v@libero.it

sinistra *New tab* (fig.3). Nelle figg.3 e 4 sono in evidenza le linguette di 4 e 5 aree di lavoro di Octave poste in successione di creazione, sotto la *Toolbar*. Quando un'area di lavoro viene creata, Octave le assegna per default il nome "Octave".

Si passa da un'area all'altra sia cliccando sull'icona che la individua sia cliccando sulle icone *freccia sinistra verde* o *freccia destra verde* della *Toolbar*. Con queste frecce il passaggio da un'area all'altra è ciclico; ad esempio se si arriva all'ultima usando l'icona *freccia destra verde*, cliccando ancora sulla *freccia destra verde* della *Toolbar* si ricomincia dalla prima. Per chiudere definitivamente un'area di lavoro si può procedere in tre modi:

a) dare nell'area di lavoro il comando quit;
b) scegliere *Close Tab* nel menu File (fig. 5);
c) cliccare sul simbolo X su sfondo rosso in alto a destra della finestra.

E' consigliabile il primo modo in quanto la chiusura di un'area di lavoro nel secondo e terzo modo genera una condizione di errore quando Octave gira sotto Windows XP, con la successiva richiesta di Windows di inviare una segnalazione di errore a Microsoft.

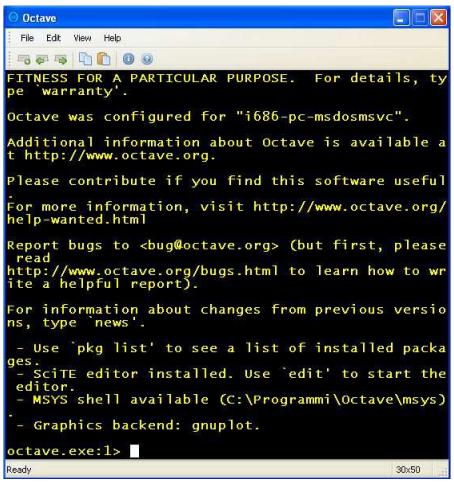


fig.1

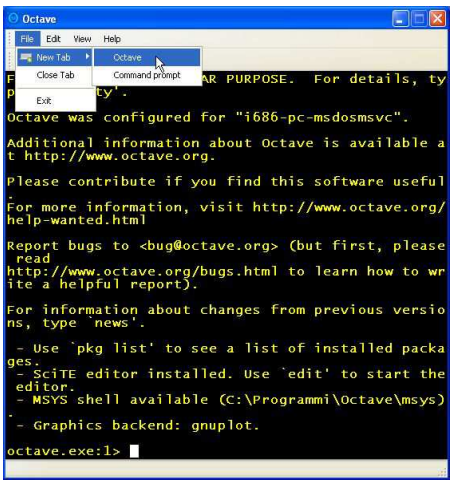


fig. 2

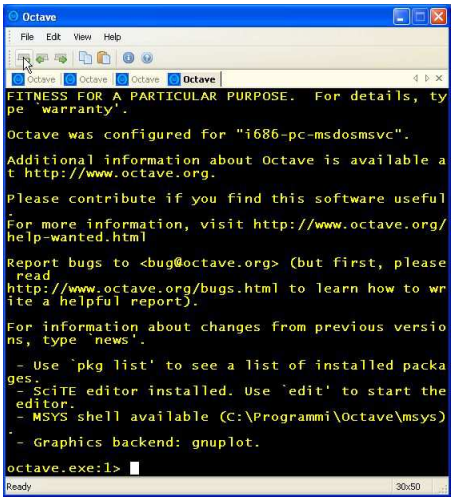


fig.3

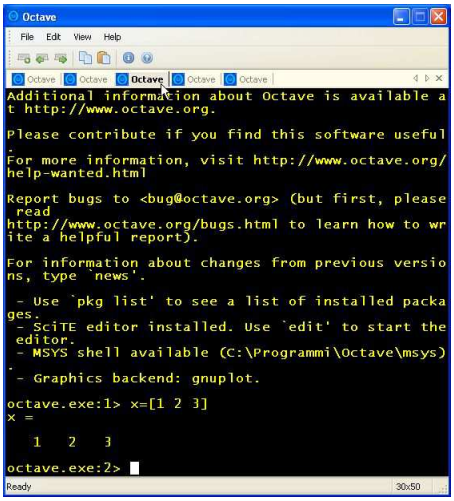


fig.4

mike_ven_s.maria.c.v@libero.it

2.2 - Edit

Con *Edit* si può copiare, incollare, rinominare una finestra di lavoro e accedere alla *Console Settings* (si veda il capitolo XY: Octave Settings). Con *Edit-Rename Tab* si apre una piccola finestra in cui cambiare il nome corrente dell'area di lavoro che è di default sempre "Octave" (fig.5 e fig.6).

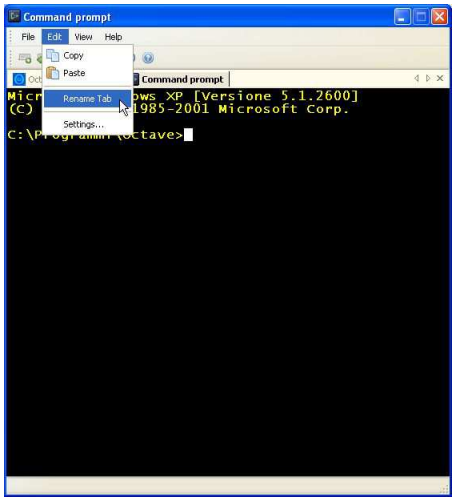


Fig.5

Fig.6

2.3 - View

Nella figura 7 sono evidenziate le voci del menu View. Il segno di spunta a sinistra di una voce attiva la funzione. Se si toglie il segno di spunta alla voce *Menu*, scompare la barra dei menu, allo stesso modo, se si toglie il segno di spunta alla voce *Toolbar*, scompare la *Toolbar* e così per le altre voci. Il *Menu* e la *Toolbar* ricompaiono se si spunta la voce corrispondente.

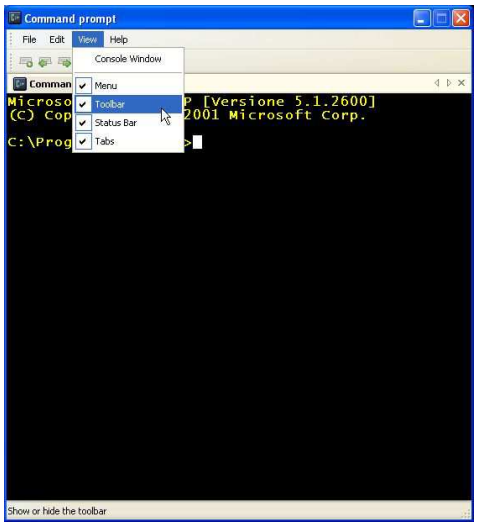


fig.7

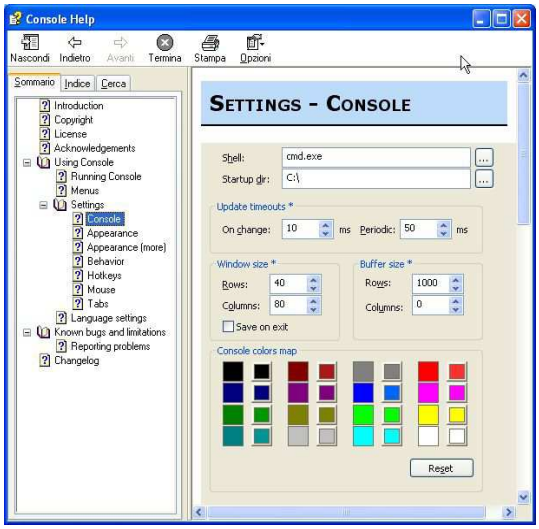


fig.8

mike_ven_s.maria.c.v@libero.it

2.4 - Help

Con l'Help (fig.8) si ottengono oltre alle notizie sul Copyright, sulla licenza GNU (General Public License) anche quelle relative alle impostazioni della finestra di lavoro di Octave (§ o cap. XX).

3. - Cambiare l'aspetto della finestra di Octave

3.1 - La finestra *Console Settings*

Nella riga del menu di Octave selezionare *Edit* e poi *Settings* (fig.1). Nella nuova finestra *Console Settings* (fig.2) si potrà agire per cambiare l'aspetto della finestra principale di lavoro di Octave. E' possibile ad esempio cambiare:

- Il numero di righe
- Il numero di colonne
- La grandezza del buffer delle righe dei risultati
- Il tipo del carattere
- Lo stile del carattere
- La dimensione del carattere
- Il colore del carattere

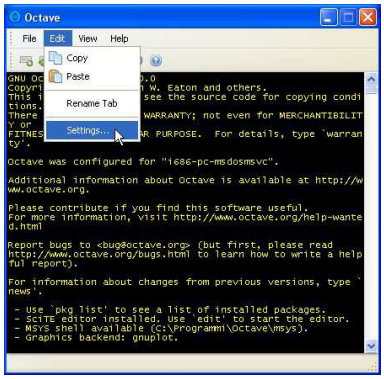


fig.1

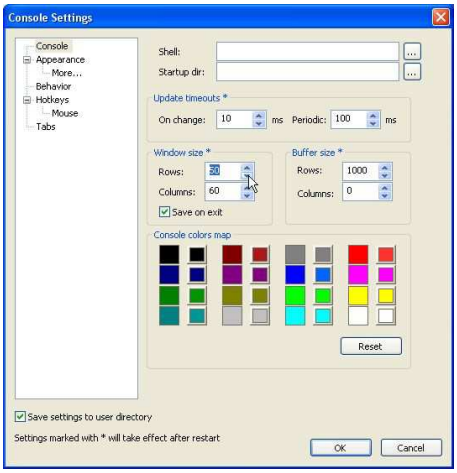


fig.2

Per quanto riguarda la grandezza del buffer delle linee dei risultati è bene che la sua dimensione sia maggiore della dimensione delle righe di visualizzazione dell'area di lavoro di Octave, in questo caso si vedrà a destra una barra di scorrimento verticale che permetterà di visualizzare le righe che sono scorse verso l'alto e che risultano nascoste se eccedono il numero di quelle da visualizzare. Se la dimensione del buffer è minore o uguale del numero di linee di visualizzazione nell'area di lavoro di Octave, la barra di scorrimento verticale non apparirà.

Le impostazioni contrassegnate da un asterisco diventano effettive al successivo avvio di Octave. Un segno di spunta in basso a sinistra di questa finestra permette di salvare le impostazioni nella cartella di lavoro dell'utente.

3.2 - Cambiare le righe e le colonne

Per cambiare il numero di righe e di colonne delle finestra di Octave scegliere *Settings* dal menu Edit (fig.1),

mike_ven_s.maria.c.v@libero.it

impostare poi nella finestra *Console Settings* il numero di righe e di colonne desiderato (fig.2).

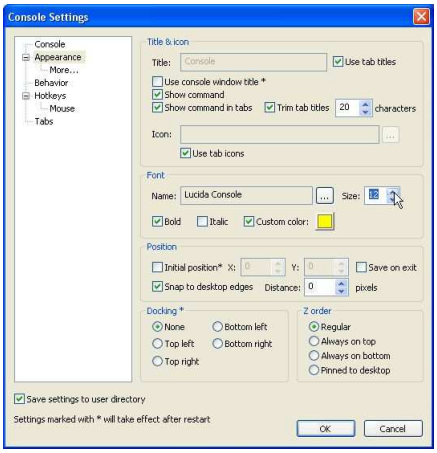


fig.3

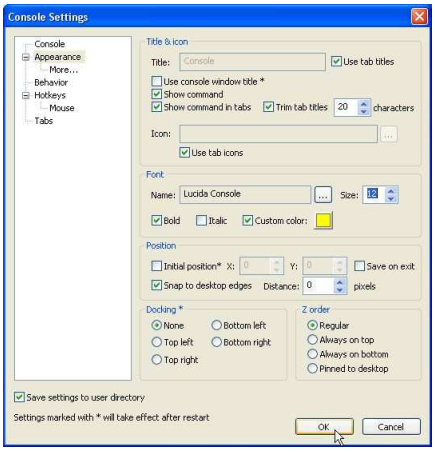


fig.4

Cliccare poi il bottone OK, uscire da Octave con il comando *quit* e riavviarlo. A questo punto le impostazioni sono diventate effettive. Nell'angolo in basso a destra della finestra di lavoro, se è attiva la *Status Bar* (menu *View*), Octave visualizza il numero di righe e di colonne dell'impostazione corrente.

3.3 - Cambiare il font del carattere

Per cambiare il font del carattere che Octave dovrà usare nella finestra di lavoro, scegliere *Settings* dal menu *Edit* (fig.1), selezionare *Appareance* nella finestra *Console Settings* e impostare il tipo di carattere *Lucida Console* o altro, lo stile *Bold* (grassetto), normale o *Italic* (corsivo), il colore e la dimensione (fig.3). Per scegliere il colore occorre mettere il segno di spunta a *Custom Color* e poi cliccare sul quadratino a destra per selezionarlo fra quelli proposti. Al termine cliccare su *OK* per rendere effettive da subito le impostazioni (fig.4).

4. - Numeri e operatori

4.1 - Tipi di dati numerici

Una costante numerica può essere uno scalare, un vettore o una matrice. Un numero, una frazione decimale, un numero in notazione scientifica o un numero complesso costituiscono ognuno un semplice esempio di costante numerica scalare. In Octave le costanti numeriche sono rappresentate in un formato in virgola mobile a doppia precisione per default. I numeri complessi *a+bi* sono memorizzati come coppie di numeri in virgola mobile a doppia precisione. La lettera *i* ha come valore √-1. In luogo di *i* sono riconosciute con lo stesso significato anche le lettere *I, J, j*:

```
octave.exe:1> 3+2i
ans  =  3+2i
octave.exe:2>3+2I
ans  =  3+2i

octave.exe:3> 3+2j
ans  =  3+2i

octave.exe:4> 3+2J
ans  =  3+2i
```


mike_ven_s.maria.c.v@libero.it

Quando si inserisce un numero complesso occorre fare attenzione a non lasciare uno spazio fra il coefficiente *b* dell'immaginario e l'immaginario *i*.

4.2 - Operazioni aritmetiche elementari

Nella seguente tabella sono definite le operazioni tra numeri reali e tra numeri complessi. Per maggiori dettagli sulle operazioni sulle matrici si veda più avanti il paragrafo 'Operatori matriciali'.

Simbolo	Uso	Operazione
+	a+b	Addizione
-	a-b	Sottrazione
*	a*b	Moltiplicazione
/	a/b	Divisione
^	a^b	Elevamento a potenza
--	x--	Stampa il vecchio valore di x e poi decrementa di uno x Se x è complesso l'azione avviene sulla parte reale
	--x	Prima decrementa di uno x poi stampa il nuovo valore di x Se x è complesso l'azione avviene sulla parte reale
++	x++	Stampa il vecchio valore di x e poi incrementa di uno x Se x è complesso l'azione avviene sulla parte reale
	++x	Prima incrementa di uno x poi stampa il nuovo valore di x Se x è complesso l'azione avviene sulla parte reale

La potenza si ottiene con il simbolo ^, ad esempio 2^3 = 2x2x2 = 8. Octave può essere usato come una calcolatrice, ad esempio:

```
octave.exe:1>3^2+7-3*(6-1)/5
ans  = 13
```

La risposta è visualizzata immediatamente dopo l'immissione del comando con il tasto ***INVIO***. La parola *ans* è la variabile del sistema in cui è conservato il valore dell'ultimo calcolo effettuato. Se si vuole il doppio di quanto è stato appena calcolato digitare *2*ans*.

```
octave.exe:2>2*ans
ans  =  26
```

L'operatore di divisione / opera anche tra numeri complessi:

```
octave.exe:3> 5/(2-i)
ans  =  2  +  1i
```

L'operatore - - scritto a destra dell'operando *x*, stampa prima il valore attuale di *x* e poi decrementa *x* di uno:

```
octave.exe:3> x=10
x  =  10
octave.exe:3> x--
ans  =  10
octave.exe:3> x
ans  =  9
```

L'operatore - - scritto a sinistra dell'operando *x*, prima decrementa di uno il valore attuale di *x* e poi stampa il nuovo valore di x:

```
octave.exe:4> x=10
x  =  10
octave.exe:5> --x
ans  =  9
octave.exe:6> x
ans  =  9
```


mike_ven_s.maria.c.v@libero.it

L'operatore ++ scritto a destra dell'operando *x*, stampa prima il valore attuale di *x* e poi incrementa *x* di uno:

```
octave.exe:5> x=10
x  =  10
octave.exe:3> x++
ans  = 10
octave.exe:3> x
ans  = 11
```

L'operatore ++ scritto a sinistra dell'operando *x*, prima incrementa di uno il valore attuale di *x* e poi stampa il nuovo valore di *x*:

```
octave.exe:6> x=10
x  =  10
octave.exe:5> ++x
ans  =  11
octave.exe:6> x
ans  = 11
```

4.3 - Variabili

Si possono conservare dati in variabili il cui nome è scelto dall'utente. Ogni volta che si pone un dato in una variabile Octave risponde per comunicare che l'operazione è stata eseguita. Il nome di una variabile non deve superare la lunghezza di 30 caratteri. Non occorre definire il tipo di una variabile prima del suo uso, Octave lo fa run-time. L'identificatore di una variabile è il nome che la distingue dalle altre, ad esempio *altezza* è l'identificatore di una variabile. Si assegna il valore *p* ad una variabile di nome *altezza* con la scrittura *altezza* = *p*.

Octave è *case sensitive*, cioè è sensibile ai caratteri maiuscoli e minuscoli. In altre parole le variabili *ALTEZZA* e *altezza* per Octave sono due variabili distinte e il valore di una di esse non altera il valore dell'altra.

```
octave.exe:1> X = 100
X  =  100

octave.exe:2>  x=5
x  =  5
```

Per conoscere il valore di una variabile se ne scrive l'identificatore seguito da INVIO e Octave ritorna il valore della variabile.

```
octave.exe:3> X
X  =  100

octave.exe:4>  x=5
x  =  5
```

Alcune variabili sono predefinite, cioè hanno già un valore interno assegnato come π che viene indicato con *pi* e il numero di Nepero che viene indicato con *e*:

```
octave.exe:6>  pi
ans  =  3.1416

octave.exe:7>  e
ans   =  2.7183
```

Le variabili che iniziano e terminano con una coppia di *underscore* sono usate internamente da Octave e non devono essere usate dall'utente nella definizione di proprie variabili. Le variabili sono fondamentalmente di tipo matrice, numero e stringa.

mike_ven_s.maria.c.v@libero.it

```
octave.exe:3>  altezza=5
altezza  = 5

octave.exe:4>  base=8
base  =  8
octave.exe:5>  area=base*altezza
area  = 40
```

Dopo il calcolo dell'area il sistema ha ancora in uso le variabili *altezza* e *base*. Per conoscerne il contenuto basta digitarne il nome

```
octave6:> altezza
altezza = 5
```

4.4 - Costruzione dei vettori

Un vettore è una lista di numeri che possono essere scritti orizzontalmente o verticalmente. Nel primo caso si ha un vettore riga, nel secondo caso un vettore colonna. Gli elementi di un vettore vengono chiamati anche componenti del vettore. L'inserimento degli elementi in un vettore riga avviene separando con uno spazio o con una virgola i numeri in parentesi quadre. L'inserimento degli elementi in un vettore colonna avviene separando con un punto e virgola i numeri in parentesi quadre. Si veda il seguente esempio per la creazione del vettore riga *x* con cinque componenti.

```
octave.exe:##>  x=[2  4  5  0  23]
x  =
      2   4   5   0   23
```

oppure

```
octave.exe:##>  x=[2, 4, 5, 0, 23]
x  =
      2   4   5   0   23
```

Si veda il seguente esempio per la creazione del vettore colonna *y* con tre componenti.

Esempio

```
octave.exe:##>  y=[7; 9; -1]
y  =
       7
       9
      -1
```

Si può usare un vettore già definito per costruire un nuovo vettore con più elementi. Nel seguente esempio il vettore riga *z* viene costruito mediante il vettore riga *x*.

```
octave.exe:##>  x=[2  4  5 ]
x  =
       2   4   5

octave.exe:##>  z=[x 100]
x  =
      2   4   5   100
```

Nel seguente esempio il vettore colonna *t* viene costruito mediante il vettore colonna *y*.

```
octave.exe:##>  y=[7; 9; -1]
y  =
         7
         9
        -1
```


mike_ven_s.maria.c.v@libero.it

```
octave.exe:##>  t=[y; 1000]
t  =
         7
         9
        -1
      1000
```

4.5 - Operatori matriciali

Simbolo	Uso	Operazione
+	a+b	Addizione elemento per elemento
-	a-b	Sottrazione elemento per elemento
*	a*b	Moltiplicazione righe per colonne su matrici quadrate
.*	a.*b	Moltiplicazione elemento per elemento
\	a\b	Divisione sinistra
./	a./b	Divisione elemento per elemento
^	a^b	Elevamento a potenza su matrici quadrate
.^	a.^b	Elevamento a potenza elemento per elemento
‘	a’	Trasposta di una matrice
--	--x	Stampa il vecchio valore di x e poi decrementa di uno x, l’azione avviene su ogni elemento della matrice
	x--	Prima decrementa di uno x poi stampa il nuovo valore di x, l’azione avviene su ogni elemento della matrice
++	x++	Stampa il vecchio valore di x e poi incrementa di uno x, l’azione avviene su ogni elemento della matrice
	++x	Prima incrementa di uno x poi stampa il nuovo valore di x, l’azione avviene su ogni elemento della matrice

Gli operatori + - ./ .* .^ sono usati su due vettori che abbiano la stessa dimensione e la stessa forma. Gli stessi operatori operano elemento per elemento pure se applicati a due matrici aventi la stessa forma e le stesse dimensioni. Per esempio

4.6 - Operatori + e –

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \pm \begin{pmatrix} y_1 \\ y_2 \\ 3 \end{pmatrix} = \begin{pmatrix} x_1 \pm y_1 \\ x_2 \pm y_2 \\ x_3 \pm y_3 \end{pmatrix}$$

```
octave.exe:##> x=[1;2;3];
octave.exe:##> y=[3;4;5];
octave.exe:##> x+y
ans  =
       4
       6
       8
```


mike_ven_s.maria.c.v@libero.it

$$(x_1 \quad x_2 \quad x_3) \pm (y_1 \quad y_2 \quad y_3) = (x_1 \pm y_2 \quad x_1 \pm y_2 \quad x_3 \pm y_3)$$

```
octave.exe:##> x=[1  2  3];
octave.exe:##> y=[3  4  5];
octave.exe:##> x+y
ans  =
       4    6    8
```

4.7 - L' operatore \

Vedere nella sezione "*Calcolo numerico*": Risoluzione di sistemi lineari (ultimo esempio).

4.8 - L' operatore ./

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} ./ \begin{pmatrix} y_1 \\ y_2 \\ 3 \end{pmatrix} = \begin{pmatrix} \frac{x_1}{y_1} \\ \frac{x_2}{y_2} \\ \frac{x_3}{y_3} \end{pmatrix}$$

```
octave.exe:##> x=[4  24  32];
octave.exe:##> y=[2  4  8];
octave.exe:##> x./y
ans  =
       2    6    4
```

4.9 - L' operatore .*

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} .* \begin{pmatrix} y_1 \\ y_2 \\ 3 \end{pmatrix} = \begin{pmatrix} x_1 y_1 \\ x_2 y_2 \\ x_3 y_3 \end{pmatrix}$$

```
octave.exe:##> x=[1;2;3];
octave.exe:##> y=[3;4;5];
octave.exe:##> x.*y
ans =
       3
       8
      15
```

$$(x_1 \quad x_2 \quad x_3) .* (y_1 \quad y_2 \quad y_3) = (x_1 y_2 \quad x_1 y_2 \quad x_3 y_3)$$

```
octave.exe:##> x=[1  2  3];
octave.exe:##> y=[3  4  5];
octave.exe:##> x.*y
ans  =
       3    8    15
```


mike_ven_s.maria.c.v@libero.it

4.10 - L' operatore ^

In Octave si usa il simbolo ^ per moltiplicare riga per colonne più volte una matrice quadrata A per se stessa. Per esempio, il prodotto riga per colonna della una matrice quadrata di ordine 3

$$A=\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & {}_{33} \end{pmatrix}$$

effettuato n volte con se stessa è

$$A^n = \underbrace{A \cdot A \cdot \ldots \cdot A}_{n\ volte}$$

ed equivale a

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & {}_{33} \end{pmatrix}^n = \underbrace{\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & {}_{33} \end{pmatrix} * \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & {}_{33} \end{pmatrix} * \ldots * \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & {}_{33} \end{pmatrix}}_{n\ volte}$$

Eseguiamo con Octave la seguente potenza

$$\begin{pmatrix} 1 & 2 \\ 0 & -1 \end{pmatrix}^2$$

```
octave.exe:##> x=[1  2;0 -1];
octave.exe:##>x^2
ans =
       1   0
       0   1
```

e poi

$$\begin{pmatrix} 1 & 2 \\ 0 & -1 \end{pmatrix}^3$$

```
octave.exe:##>x^3
ans =
       1    2
       0   -1
```


mike_ven_s.maria.c.v@libero.it

4.11 - L' operatore .^

$$\begin{pmatrix} x & y & z \end{pmatrix}.\wedge\begin{pmatrix} a & b & c \end{pmatrix} = \begin{pmatrix} x^a & y^b & z^c \end{pmatrix}$$

```
octave.exe:##> x=[1  2  3];
octave.exe:##> y=[3  4  5];
octave.exe:##> x.*y
ans  =
       1   16   243
```

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix}.\wedge\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} x^a \\ y^b \\ z^c \end{pmatrix}$$

```
octave.exe:##> x=[1; 2; 3];
octave.exe:##> y=[3; 4; 5];
octave.exe:##> x.^y
ans  =
       1
       16
       243
```

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix}^{.\wedge n} = \begin{pmatrix} a^n \\ b^n \\ c^n \end{pmatrix}$$

```
octave.exe:##> a=[1; 2; 3];
octave.exe:##> a.^3
ans  =
       1
       8
       27
```

$$\begin{pmatrix} a & b & c \end{pmatrix}^{\wedge n} = \begin{pmatrix} a^n & b^n & c^n \end{pmatrix}$$

```
octave.exe:##> a=[1  2  3];
octave.exe:##> a.^3
ans  =
       1    8    27
```

4.12 - L' operatore *

L' operatore * moltiplica due matrici *A* e *B* riga per colonna. Il prodotto è una matrice *C=A*B*. Se *A* è del tipo *m x n* e *B* è del tipo *n x p* allora la matrice prodotto *C* è del tipo *m x p*. Si osservi che le colonne di *A* devono essere pari alle righe di *B* altrimenti si ottiene un messaggio d'errore. E' noto che la moltiplicazione tra matrici non è commutativa cioè non sempre risulta *A*B=B*A*.

mike_ven_s.maria.c.v@libero.it

La moltiplicazione di una matrice *3x2* per una matrice *2x2* dà una matrice *2x2*:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{pmatrix} * \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \\ a_{31}b_{11} + a_{31}b_{21} & a_{31}b_{12} + a_{32}b_{22} \end{pmatrix}$$

```
octave.exe:##> a=[1  2;0 1;1 0];
octave.exe:##> b=[0  1;1 2];
octave.exe:##> a*b
ans =
       2   5
       1   2
       0   1
```

La moltiplicazione di due matrici quadrate ha come risultato una matrice quadrata dello stesso ordine:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} * \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}$$

```
octave.exe:##> a=[0 1;2 3];
octave.exe:##> b=[1  2;3  4];
octave.exe:##> a*b
ans =
       3   4
      11   16
```

4.13 - L' operatore ‘

L'apice è l'operatore usato per ottenere la trasposta di un vettore o di una matrice. Questo operatore unario esegue la trasposta di una matrice, cioè ne scambia le righe con le colonne.

```
octave.exe:##> a=[1   2   3;4   5   6;7   8   9]
a  =
       1   2   3
       4   5   6
       7   8   9

octave.exe:##> a'
ans =
       1   4   7
       2   5   8
       3   6   9
```

5. - I/O

5.1 - Input da tastiera

Per inserire dei dati da tastiera si utilizzano le istruzioni *input* e *menu* le quali danno la possibilità di stampare anche un messaggio perché l'utente capisca cosa è richiesto. La loro forma è

mike_ven_s.maria.c.v@libero.it

- input(“stringa di messaggio”)
- input(“stringa di messaggio”, “s”).
- menu(“messaggio”,”opzione 1”,”opzione 2”,”opzione 3”,”opzione 4”)

Nella prima forma *input* attende l’immissione di un dato numerico e nella seconda forma un dato stringa; l’istruzione *menu* attende un inserimento numerico.

Vediamo con un esempio come vanno usate le prime due istruzioni. Vogliamo chiedere in ingresso un numero con la contemporanea stampa del messaggio che informa l’utente sull’operazione che deve compiere

```
octave.exe:1> input(“inserisci un numero x = “)
inserisci un numero x =
```

Octave stampa il messaggio e attende l’immissione del dato che voi digiterete, per esempio 1000,

```
inserisci un numero x = 1000
x =  1000
```

Octave vi informa che il dato è stato conservato nella variabile x. Per convincervi di ciò chiedete subito ad Octave di calcolare il doppio del valore di x:

```
octave.exe:2>  2*x
ans =  2000
```

Allo stesso modo se vogliamo chiedere in ingresso una stringa con la contemporanea stampa del messaggio che informi l’utente sull’operazione che deve compiere scriveremo

```
octave.exe:1> input(“inserisci una  sequenza di caratteri  b = “)
inserisci una  sequenza di caratteri  b = abc123
b  =  abc123
octave.exe:2>
```

Chiedete adesso ad Octave il contenuto della variabile b:

```
octave.exe:2> b
```

vi risponderà con:

```
b  =  abc123
```

Illustriamo il funzionamento dell’istruzione *menu*.

```
octave.exe:3>  scelta=menu(“Scegli”,”Scelta 1”,”Scelta 2”,”Scelta 3”)
Scegli
       [1]  Scelta 1
       [2]  Scelta 2
       [3]  Scelta 3

pick a number, any namber :  2
scelta  =  2
```

Alla richiesta di scegliere abbiamo introdotto 2 da tastiera e Octave risponde ponendo il valore 2 nella variabile scelta.

5.2 - L’istruzione disp

Per visualizzare messaggi sul video o valori di variabili si può usare *disp(x)*. Vediamo un esempio.

```
octave.exe:##> x=100;

octave.exe:##> disp(“Il valore di x è: “) , disp(x)
```


mike_ven_s.maria.c.v@libero.it

```
Il valore di x è:
100
```

Creiamo il vettore di 6 elementi x=[1,2,3,4,5,6]

```
octave.exe:##> x=1:6;
```

e usiamo di nuovo *disp*

```
octave.exe:##> disp("Il valore di x è: ") , disp(x)
Il valore di x è:
1   2   3   4   5   6
```

5.3 - Salvataggio e caricamento dei dati

Con il comando *save* seguito dal *nome_file.mat* si salvano tutte le variabili create nella cartella corrente. Octave provvede ad aggiungere automaticamente l'estensione *.mat* se viene omessa. Per avere disponibili tutte le variabili di una precedente sessione di lavoro usate il comando *load* seguito dal *nome_file*. Lanciate Octave e provate di nuovo a calcolare l'area di un rettangolo come fatto poco sopra e poi salvate tutto nel file dal nome *mie_variabili*. Poi uscite da Octave, riapritelo e caricate con load il file *mie_variabili*. Digitate altezza e base, vedrete che sono ancora disponibili. Si veda il seguente esempio.

```
octave.exe:1>  altezza=10
altezza = 10
octave.exe:2>  base=3
base  =  3

octave.exe:3>  area=base*altezza
area  = 30

octave.exe:4>  save mie_variabili

octave.exe:5>  quit
```

Riavviate Octave e poi caricate il file mie_varaibili:

```
octave.exe:1>  load mie_variabili
```

Vengono rese disponibili di nuovo tutte le variabili della precedente sessione di lavoro. Verificatelo digitando

```
octave.exe:2>  altezza
```

il sistema risponde con

```
altezza  = 10.
```

5.4 - Eliminazione delle variabili

Ogni variabile viene messa da Octave nello spazio di lavoro. Se si vuole eliminare una variabile usare il comando *clear nome_variabile*, cioè *clear* seguito dal nome della variabile.

```
octave.exe:##> y=2
y  =  2
octave.exe:##> clear y
```

Digitate *y* per vederne il valore:

```
octave.exe:##>  y
```

il sistema avverte che la variabile *y* non è definita e attende un nuovo comando:

mike_ven_s.maria.c.v@libero.it

```
error: ‘y’ undefined near line 5 column 1
octave.exe:##>
```

Per eliminare tutte le variabili usare solo *clear*.

5.5 - Vedere le variabili

Con il comando *who* il sistema risponde con l’elenco di tutti i nomi delle funzioni e delle variabili create o usate.

```
octave.exe:##> who

*** dynamically linked function:
__COM__ builtin: find  dispatch       getpwuid

*** currently compiled functions:
__default_graphics__          findstr        ispc           pkg            strcat
              edit            fullfile               isunix         rem
              strrep
              fileparts       index

*** local user variabiles:
__nargin__                    ans            home_path      x              y
```

6. - Numeri e matrici

6.1 - Costruire una successione di numeri

Con la scrittura *a:passo:b* si chiede a Octave di partire dal numero *a*, di incrementarlo con il *passo* fino a *b*. L’ultimo numero della successione che supera il limite destro *b* non viene considerato. Questa tecnica è usata per creare vettori con componenti equispaziati. Il passo può essere sia intero che decimale. Se il passo manca l’incremento è di una unità.

```
octave.exe:1>  x = 1:5
x  =
       1   2   3  4  5

octave.exe:2>  y = 1:2:6
y  =
       1   3   5

octave.exe:3>  z = 3:0.2:4
z  =
      3.2000   3.4000  3.6000  3.8000  4.0000
```

Si può creare una successione di numeri con centinaia o migliaia di elementi. Se non è specificato diversamente, Octave visualizza sul monitor tutti gli elementi. L’operazione porta via molto tempo e ad ogni schermata si deve scegliere tra (f)orward, (b)ack, (q)uit (avanti, indietro, uscire). Per evitare la stampa sul monitor, alla fine del comando aggiungete un punto e virgola, come nei seguenti esempi:

```
octave.exe:4>  t = 2:2:10;
octave.exe:5>
```

oppure

```
octave.exe:5>  50:5:100;
octave.exe:6>
```

Nel primo caso si noti che la successione dei pari da 2 a 10 è disponibile per successivi usi e si trova tutta nella variabile *t*. Per convincerci di ciò digitiamo la variabile *t* nella linea di comando:

mike_ven_s.maria.c.v@libero.it

```
octave.exe:6>  t
t  =
      2   4   6   8   10
```

Nel secondo caso la successione è disponibile nella variabile *ans* però solo immediatamente dopo la generazione. La successione viene persa se è seguita subito da un'altra operazione perché la variabile *ans* prende il nuovo valore.

6.2 - Costruzione di matrici

Una matrice *X* è una tabella di numeri scritta secondo righe e colonne. Le righe sono numerate da sinistra a destra iniziando da 1. Le colonne sono numerate dall'alto verso il basso iniziando da 1. In questo modo ogni elemento x_{ij} di una matrice può essere individuato da una coppia di numeri *i* e *j*, detti indici. Il primo indica la posizione dell'elemento x_{ij} nella riga, il secondo è la posizione dell'elemento x_{ij} nella colonna. Una matrice quadrata ha tante righe quante colonne. Una matrice è rettangolare se il numero delle righe è diverso dal numero delle colonne.

Scriviamo una matrice quadrata:

$$X = \begin{pmatrix} 1 & -2 & 3 \\ -1 & 4 & -6 \\ 7 & 5 & 8 \end{pmatrix}$$

Di seguito gli elementi della matrice X contraddistinti dagli indici:

$x_{1,1}=1$ $x_{1,2}=-2$ $x_{1,3}=3$

$x_{2,1}=-1$ $x_{2,2}=4$ $x_{2,3}=-6$

$x_{3,1}=7$ $x_{3,2}=5$ $x_{3,3}=8$

L'inserimento degli elementi di una matrice avviene scrivendo gli elementi di ogni riga separati tra loro da uno spazio, scrivendo un punto e virgola alla fine della riga e racchiudendo tutto tra parentesi quadre. Non occorre scrivere il punto e virgola dopo l'ultimo elemento dell'ultima riga. Diamo la matrice X a Octave.

```
octave.exe:##>  [1  -2   3; -1  4  -6; 7  5  8]
ans  =
       1  -2    3
      -1   4   -6
       7   5    8
```

Se la matrice non ha un nome si può comunque estrarre ogni suo elemento mediante la variabile *ans*. Per esempio l'elemento nell'incrocio della prima riga e della prima colonna è *ans(1,1)*:

```
octave.exe:##>  ans(1,1)
ans  =  1
```

Adesso però tutta la matrice con i suoi elementi è stata persa. Per evitare questa situazione di perdita di dati, si può o inserire la matrice senza nome ma porre subito dopo *x=ans*, oppure inserirla con un nome.

Inseriamo prima una matrice senza nome e poi assegneremo *ans* alla variabile *x*.

```
octave.exe:##>  [1  2   3; 4  5  6; 7  8  9]
ans  =
       1   2    3
       4   5    6
       7   8    9
```


mike_ven_s.maria.c.v@libero.it

```
octave.exe:##>  x=ans
x  =
       1   2    3
       4   5    6
       7   8    9
```

Adesso inseriamo la matrice *z*.

```
octave.exe:##>  z=[0  1   2; 3  4  5; 6  7  8]
z  =
       0   1    2
       3   4    5
       6   7    8
```

Per estrarre l'elemento centrale, quello appartenente alla seconda riga e alla seconda colonna scriveremo *z(2,2)*.

```
octave.exe:##>  z(2,2)
ans  =  4
```

Poniamo in *v* il doppio dell'elemento che si trova nell'incrocio della seconda riga con la terza colonna.

```
octave.exe:##> v=2*z(2,3)
v  =  10
```

Creiamo una matrice *y* uguale a *z*.

```
octave.exe:##>  y=z
z  =
      0   1    2
      3   4    5
      6   7    8
```

Osserviamo che *b=[7]* equivale a *b=7*. In altre parole una matrice con un unico elemento può essere pensata come una variabile ordinaria e trattata come tale.

```
octave.exe:##> b=[3]
b  =  3

octave.exe:##> b=(1,1)
ans  =  3

octave.exe:##> 2*b
ans  =  6
```

6.3 - Formato dei numeri

Octave visualizza i numeri con cinque cifre significative ma è possibile avere una visualizzazione con 15 cifre significative con il comando

```
octave.exe:##>  format long
```

Per ritornare al precedente formato usate il comando

```
octave.exe:##>  format short
```

La notazione usata è quella esponenziale, per esempio 3673.2=3.6732x10^3. Questo numero è visualizzato da Octave nella forma 3.6732e+3 e i numeri possono essere digitati nel sistema nella stessa forma.

Vi sono delle varianti ai comando format. Se ad esempio x=5/7 e y=2.345e-7, si veda di seguito come possono essere visualizzati:

mike_ven_s.maria.c.v@libero.it

1)	format short	x=0.71429	y=2.3450e-007
2)	format short e	x=7.1429e-001	y=2.3450e-007
3)	format short g	x=0.71429	y=2.345e-007
4)	format long	x=0.714285714285714	y=2.34500000000000e-007
5)	format long e	x=7.14285714285714e-001	y=2.34500000000000e-007
6)	format long g	x=0.714285714285714	y=2.3450-007
7)	format bank	x=0.71	y=0.00
8)	format rat	x=5/7	y=3/12793177
9)	format hex	x=3fe6db6db6db6db7	y=3e8f795bd04f271e

6.4 - Numeri complessi: vengono introdotti nel formato *a+ib* ad esempio inseriamo *2-3i*.

```
octave.exe:##> 2-3i
ans = 2 - 3i
```

6.5 - Infinito: quando si divide per zero Octave visualizza ***inf*** con un messaggio.

```
octave.exe:##> 5/0
warning: division by zero
ans = Inf
```

6.6 - Non è un numero: il sistema risponde con ***NaN*** (Not a Number) quando si divide zero per zero e in altri casi che danno un risultato indefinito.

```
octave.exe:##> 0/0
warning: division by zero
ans = NaN
```

7. - Manuale in linea

7.1 - Ripetizione dei comandi

I tasti freccia su ↑ e freccia giù ↓ servono per ripetere il precedente o il successivo comando nella riga d'immissione. Con i tasti freccia destra → e freccia sinistra ← si procede lungo la riga d'immissione.

7.2 - Aiuto in linea

È possibile avere un aiuto in linea digitando semplicemente *help*. Sul monitor compare un elenco di operatori, di parole riservate, di funzioni , ecc. di Octave. Nell'ultima riga in basso vengono visualizzati i comandi (f)orward , (b)ack, (q)uit, con i quali si procede nell'elenco in avanti, indietro o si esce. Basta digitare la lettera racchiusa fra parentesi tonde.
Per ottenere un aiuto su un comando o una funzione bisogna scrivere *help nome_comando*, oppure *help nome_funzione* cioè *help* seguito dal nome del comando o della funzione. In questo caso Octave dopo aver portato sul monitor le informazioni ritorna nello stato di attesa con il prompt.
Ad esempio, per chiedere informazioni sulla funzione *abs* scrivere

```
octave.exe:##> help  abs
```

7.3 - Manuale in linea

Con il comando *help –i* il sistema visualizza un menu. Con i tasti freccia su ↑ e freccia giù ↓ si porta il cursore sull'argomento di interesse. Con ***INVIO*** si visualizza il contenuto dell'argomento scelto. Con ***Pag-su*** e ***Pag-giù*** si va da una pagina all'altra più velocemente. Per navigare dopo la scelta digitare le lettere ***U*** (***U****p*), ***N*** (***N****ext*), ***P*** (***P****revious*). Per ritornare al menu iniziale digitare ***ret***. Per uscire dall'help digitare la lettera ***q***. La sintassi della

mike_ven_s.maria.c.v@libero.it

richiesta del manuale in linea è

```
octave.exe:##> help  -i
```

8. - Grafici

8.1 - Definire una successione di numeri con dato passo

Una sequenza di numeri viene definita come matrice avente una sola riga e più colonne, cioè come vettore. La scrittura 1:5 è interpretata come la sequenza degli interi da 1 a 5 con passo (sottinteso) di 1.

```
octave.exe:##> a=0:6
a =
   0  1  2  3  4  5  6
```

E' possibile indicare il *passo* (*step*) inserendolo fra gli estremi dell'intervallo, usando come separatore i due punti.

```
octave.exe:##> a=0:2:6
a =
   0  2  4  6

octave.exe:##> a=1:2:8
a =
   1  3  5  7

octave.exe:##> a=2:5:15
a =
   2  7  12
```

Per tabulare una funzione oppure per disegnarne il grafico occorre usare una sequenza di punti scelti in qualche modo in un determinato intervallo. Per esempio creiamo una sequenza di numeri nell'intervallo $[-\pi; \pi]$ con passo $\pi/4$.

```
octave.exe:##>  x= -pi:pi/4:pi
x =
-3.1416      -2.3562     -1.5708     -0.7854     0.0000     0.7854     1.5708     2.3562
3.1416
```

8.2 - Tabulare una funzione

Per tabulare una funzione *F* occorre scegliere in qualche modo dei valori x_i nel dominio di *F* e in questi calcolare $F(x_i)$. Supponiamo che i valori x_i nel dominio di *F* siano:

$x_1, x_2, x_3, \ldots, x_n$

allora i valori di *F* in tali punti sono:

$F(x_1), F(x_2), F(x_3), \ldots, F(x_n)$.

Definiamo la successione dei punti nell'intervallo $[-\pi; \pi]$, con passo di $\pi/4$, in cui calcolare il valore di y=sin(x).

```
octave.exe:##>  x= -pi:pi/4:pi
```


mike_ven_s.maria.c.v@libero.it

```
x =
 -3.1416  -2.3562  -1.5708  -0.7854  0.0000  0.7854   1.5708   2.3562  3.1416
```

Calcoliamo i valori di y=sin(x) nei punti sopra definiti.

```
octave.exe:##>  y=sin(x)
y=
-0.00000       -0.70711       -1.00000       -0.70711       0.00000    0.70711
 1.00000        0.70711        0.00000
```

8.3 - Disegnare un grafico

La tecnica è quella del disegno per punti del grafico di una curva *y=F(x)*. Occorre avere *x* e *F(x)* per numerosi punti, di solito equamente distanziati nell'intervallo scelto.

Si definisce la successione di valori del dominio di *F*, come si è detto nella sezione "Tabulare una funzione". Le coppie di punti $(x_1,F(x_1))$, $(x_2,F(x_2))$, $(x_3,F(x_3))$, … , $(x_n,F(x_n))$ sono i punti del grafico di *F* da disegnare nel piano cartesiano *XY*. Ovviamente, maggiore è il numero di punti *x* migliore sarà la definizione del grafico di *F*. Il comando di base per disegnare grafici è *plot(x,y)*, dove *x* è la variabile indipendente e *y* la variabile dipendente.

Per esempio disegniamo il grafico della funzione seno.

```
octave.exe:##> x=-pi:pi/4:pi;

octave.exe:##>y=sinx(x);
octave.exe:##>plot(x,y)
```

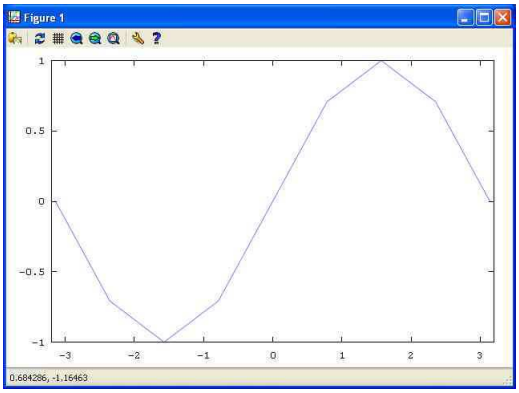


fig. 1 – Grafico della funzione seno in [-π;π] con un passo di π/4.

Il grafico è molto approssimato. Per migliorarne l'aspetto aumentiamo il numero dei punti x scegliendo un passo più piccolo, ad esempio di π/8. Usiamo i tasti freccia su ↑ e freccia giù ↓ per apportare le variazioni necessarie. Con un passo π/16 si nota un notevole miglioramento.

mike_ven_s.maria.c.v@libero.it

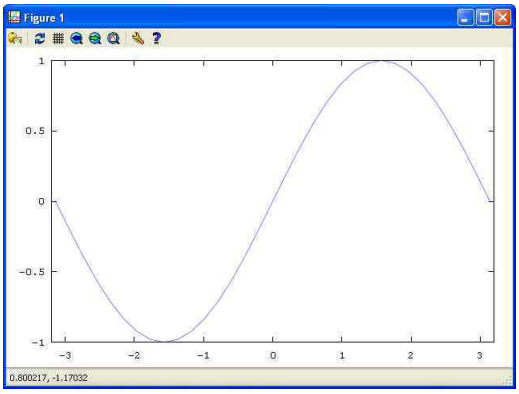


fig. 2 – Grafico della funzione seno in [-π;π] disegnata con un passo di π/16.

L'aspetto del grafico può essere variato. Si provi

```
Octave.exe:## > plot(x,y,'ro')
```

Si vedrà un grafico in cui i punti sono dei circoletti rossi. Nel comando 'ro', la r definisce il colore rosso del grafico e la o ne definisce l'aspetto.

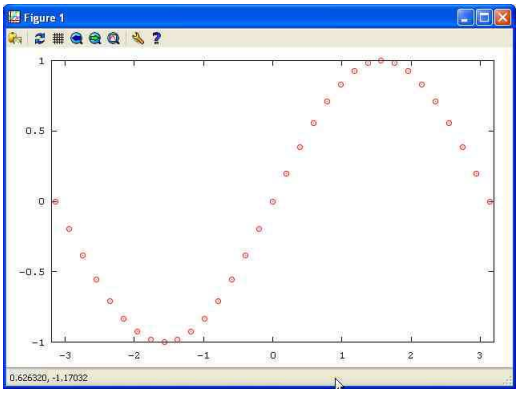


Fig. 3 – Grafico della funzione seno in [-π;π] in colore rosso e stile circoletto.

Le seguenti tabelle riportano i comandi del colore e dello stile dei grafici.

Numero	Colore	Valore colore
1	red	'r'
2	green	'g'
3	blue	'b'
4	magenta	'm'
5	cyan	'c'
6	brown	'b'
	w bianco !!!	'w'
	'k' (nero) ok	'k'

Tab. 1 - Tabella dei colori del comando plot

mike_ven_s.maria.c.v@libero.it

Valore stile	Descrizione stile
'*'	stella
'+'	segno più
'o'	o minuscolo
'x'	simbolo per
'^'	triangolo pieno
's'	quadretto pieno
'd'	diamante
'v'	triangolo pieno con punta in basso
'<'	triangolo quasi pieno con punta in basso
'>'	triangolo quasi pieno con punta in alto
'p'	quadrato con punto al centro
'h'	diamante con punto al centro

Tab. 2 - Tabella degli stili del comando plot

E' possibile disegnare più curve nella stessa finestra con

plot(x,y1,x,y2,x,y3).

Disegnamo le funzioni seno e coseno nella stessa finestra in [0,2π].

```
octave.exe:##>  x=0:pi/50:2*pi;
octave.exe:##>  y1=sin(x);
octave.exe:##>  y2=cos(x);
octave.exe:##>  plot(x,y1,x,y2)
```

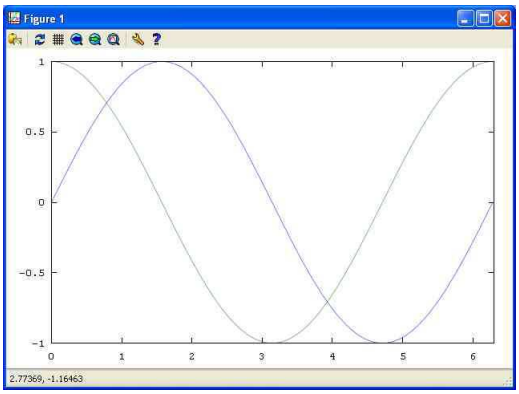


Figura 4 – Grafico delle funzioni seno e coseno nell'intervallo [0,2π].

Per aggiungere una griglia al grafico scrivere:

```
octave.exe:##>  grid on
```

Per dare un nome agli assi:

```
octave.exe:##>  xlabel('angoli in radianti')

octave.exe:##>  ylabel('y-ordinate)')
```

Per dare un titolo al grafico:

```
octave.exe:##>  title('Grafico di y=sin(x)')
```

Si può anche aggiungere una legenda per abbinare al colore di un grafico il tipo di curva. Vediamo come. Visualizziamo prima il grafico di due curve con

mike_ven_s.maria.c.v@libero.it

```
octave.exe:##>  plot(x,sin(x),'r',x,cos(x),'g')
```

e poi con il seguente comando

```
octave.exe:##>  legend('seno','coseno')
```

è plottato il grafico delle stesse due funzioni con una breve linea rossa dopo la parola *seno* e un breve tratto verde dopo la parola *coseno*.

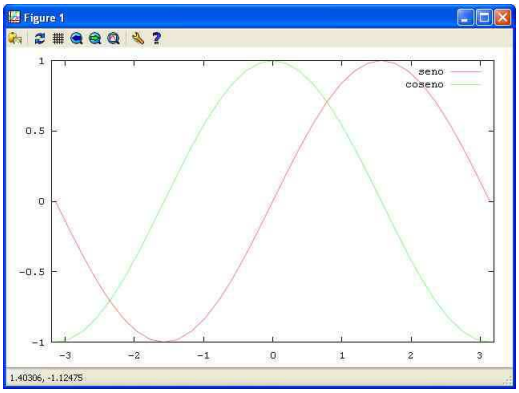


Figura 5 – Grafico delle funzioni seno e coseno con legenda.

Con il comando *hold on* si ordina a Octave di lasciare il precedente grafico nell'area di visualizzazione in modo da farlo apparire insieme al successivo. Per esempio, si plotti prima il grafico della funzione seno

```
octave.exe:##>  plot(x,sin(x),'ro')
```

e si dia il commando

```
octave.exe:##>  hold on
```

poi si plotti la funzione *abs(lnx)*

```
octave.exe:##>  plot(x,abs(log(x)),'b+')
```

la finestra grafica visualizzerà il grafico delle due curve. Si vede che il primo grafico non è stato cancellato.

Infine si digiti

```
octave.exe:##>  legend('seno','abs(lnx)')
```

8.4 - Scala manuale

Si possono impostare le dimensioni delle finestre entro le quali disegnare le funzioni, con il comando *axis*. Il comando *axis* ha un argomento che è un vettore definito da (x_{min}, x_{max},y_{min},y_{max}). Vediamo un esempio.

```
octave.exe:##>  x=-10:0.1:10;
```

octave.exe:##> plot(x,x.*cos(x))

per vedere lo stesso grafico rimpicciolito rispetto alle dimensioni della finestra definita da $-20 \le x \le 20$ e $-20 \le y \le 20$ digitare

```
octave.exe:##>  axis([-20 20 -20 20])
```


mike_ven_s.maria.c.v@libero.it

Per vedere ingrandita una parte del grafico scrivere

```
octave.exe:##>  axis([-2 2 -2 2])
```

oppure

```
octave.exe:##>  axis([-1 2 -2 2])
```

9. - Alcune funzioni

abs(x)
Questa funzione agisce su un solo argomento *x*. Ritorna il valore assoluto di *x*.

```
octave.exe#1> x = [-0.34 2.67; -3.01 1.23]
octave.exe:#1> abs(x)
ans =
       0.34000     2.67000
       3.01000     1.23000

octave.exe:#2> y=abs(x)
y =
       0.34000     2.67000
       3.01000     1.23000
```

acos(x)
Ritorna l'arco in radianti il cui coseno è *x.*

acosd(x)
Ritorna in gradi l'arcocoseno di *x*.

```
octave.exe:##> acosd(0.5)
ans   =   60.000
```

acotd(x)
Ritorna in gradi l'arcotangente di *x*.

```
octave.exe:##> acotd(1)
ans   =   45
```

arg(z)
Calcola l'argomento di z=x+iy definito da θ=arctan(y/x) in radianti.

```
octave.exe:#1> z=3+5i
z  =  3 + 5i

octave.exe:#2> arg(z)
ans  = 1.0304
```

asin(x)
Ritorna l'arco in radianti il cui seno è *x*.

asind(x)
Ritorna in gradi l'arcoseno di *x*.

```
octave.exe:##> asind(0.5)
ans   =   30.000
```


mike_ven_s.maria.c.v@libero.it

atan(x)
Ritorna l'arco in radianti la cui tangente è *x*.

```
octave.exe:##> atan(inf)
ans = 1.5708

octave.exe:##> atan(-inf)
ans = -1.5708

octave.exe:##> atan(sqrt(3)/3)
ans = 0.52360
```

(0.52360 è il valore approssimato di $\pi/6$ radianti, cioè 30°)

atand(x)
Ritorna in gradi l'arcotangente di *x*.

```
octave.exe:##> atand(1)
ans   =   45
```

bar(x)
Ritorna il grafico a barre degli elementi del vettore *x*. Gli elementi di *x* vengono numerati automaticamente. Il primo elemento di *x* ha ascissa 1 e ordinata x_1, il secondo elemento di *x* ha ascissa 2 e ordinata x_2, e così via. Vedere la sezione *Statistica*.

bar(x,y)
Ritorna il grafico a barre degli elementi dei due vettori *x* e *y*. Gli elementi del vettore *x* devono risultare in ordine crescente. Vedere la sezione *Statistica*.

ceil(x)
Questa funzione agisce su un solo argomento *x*. Arrotonda verso più infinito (verso destra).

```
octave.exe:#1> x = [-0.34 2.67; -3.01 1.23]
octave.exe:#2 > ceil(x)
ans =
       -0     3
       -3     2

octave.exe:## > y=ceil(x)
y =
       -0     3
       -3     2
```

conj(z)
Calcola il coniugato di *z*.

```
octave.exe#2> conj(3-5i)
ans  =  3 + 5i
```

conv(x,y)
Moltiplica i due polinomi *x* e *y*. Vedere la sezione *Calcolo numerico*.

cos(x)
Ritorna il coseno di *x* con *x* espresso in radianti.

mike_ven_s.maria.c.v@libero.it

```
octave.exe:##> x=[pi  –pi; 2*pi  pi/4 ]
ans
       3.14159         -3.14159
       6.28319          0.78540

octave.exe:##> cos(x)
ans =
          -1.00000         -1.00000
           1.00000          0.70711
```

cosd(x)
Ritorna il coseno di *x* con *x* espresso in gradi.

```
octave.exe:##> cosd(60)
ans   =   0.5000
```

cotd(x)
Ritorna la cotangente di *x* con *x* espresso in gradi.

```
octave.exe:##> tand(45)
ans   =   1.0000
```

deconv(x,y)
Divide il polinomio *x* per il polinomio *y*. Vedere la sezione *Calcolo numerico*.

det(x)
Calcola il determinante della matrice *x*.

disp(x)
Visualizza sul monitor il contenuto di *x*.

eig(x)
Calcola gli autovalori del vettore *x*.

exp(x)
Calcola e^x.

factor(x)
Scompone in fattori primi l'intero positivo *x*.

```
octave.exe:#1 > factor(12)
ans =
       2   2   3

octave.exe:#2 > factor(17)
ans  =  17

octave.exe:#3 > factor(-1921)
ans  =   -1921

octave.exe:#4 > factor(1921)
ans  =
       17   113

octave.exe:#4 > v=factor(840)
ans  =
       2     2     2     3     5     7
```


mike_ven_s.maria.c.v@libero.it

factorial(x)
Ritorna il fattoriale di un intero positivo o nullo se *x* è una matrice di un solo elemento altrimenti ritorna il fattoriale di tutti gli elementi della matrice.

```
octave.exe:##> factorial(3)
ans = 6

octave.exe:## > x=[ 1  2; 3 4]
ans =
       1      2
       3      4

octave.exe:##> factorial(x)
ans =
      1      2
      6     24
```

fix(x)
Questa funzione agisce su un solo argomento *x*. Arrotonda verso zero.

```
octave.exe:#1> x = [-0.34  2.67; -3.01  1.23]
octave.exe:#2 > fix(x)
ans =
       -0     2
       -3     1

octave.exe:## > y=fix(x)
y =
       -0     2
       -3     1
```

floor(x)
Questa funzione agisce su un solo argomento *x*. Arrotonda verso meno infinito (verso sinistra).

```
octave.exe:#1> x = [-0.34 2.67; -3.01 1.23]
octave.exe:#2 > floor(x)
ans =
       -1     2
       -4     1

octave.exe:## > y=floor(x)
y =
       -1     2
       -4     1
```

fsolve("espressione funzione",a)
Trova la radice dell'equazione *f(x)=0* con approssimazione iniziale *a*. Se si vuole usare una variabile che raccolga il valore della radice, scrivere: *[x]=solve("espressione funzione",a),* dove la variabile in cui verrà messa la radice è *x*. Vedere la sezione *Calcolo numerico*.

hist(x)
Ritorna il grafico a barre orizzontali degli elementi del vettore *x*. Gli elementi di *x* vengono numerati automaticamente.

hist(x,y)
Ritorna il grafico a barre orizzontali degli elementi dei due vettori *x* e *y*. Gli elementi del vettore *x* devono risultare in

mike_ven_s.maria.c.v@libero.it

ordine crescente.

imag(z)
Ritorna la parte immaginaria di *z*.

```
octave.exe:#1> z=3-5i
z  =  3 - 5i

octave.exe:#2> imag(z)
ans   =  -5
```

isprime(n)
Ritorna 1 (*true*) se *n* è primo altrimenti 0 (*false*).

```
octave.exe#> isprime(17)
ans  =  1

octave.exe#> isprime(12)
ans  =  0
```

iscomplex
Ritorna 1 (*true*) se *x* è complesso altrimenti 0 (*false*).

```
octave.exe#> x=3-5i
x  =  3 - 5i

octave.exe#> y= 7
y  =  7

octave.exe#> iscomplex(x)
ans  =  1

octave.exe#> iscomplex(y)
ans  =  0
```

isreal
Ritorna 1 (*true*) se *x* è reale altrimenti 0 (*false*).

```
octave.exe#> x=3-5i
x  =  3 - 5i

octave.exe#> y= 7
y  =  7

octave.exe#> isreal(x)
ans  =  0

octave.exe#> isreal(y)
ans  =  1
```

issquare(x)
Se *x* è una matrice quadrata ritorna la dimensione di *x*, altrimenti ritorna 0.

```
octave.exe#> x=[1 2;3 4];
octave.exe#> issquare(x)
ans  =  2
```


mike_ven_s.maria.c.v@libero.it

```
octave.exe#> y=[1 2 3 4];
octave.exe#> issquare(y)
ans  =  0

octave.exe#> t=[3];
octave.exe#> issquare(t)
ans  =  1
```

isvector(x)
Se *x* è un vettore ritorna 1 (*true*), altrimenti ritorna 0 (*false*).

```
octave.exe#> x=[1 2 3 4];
octave.exe#> isvector(x)
ans  =  1

octave.exe#> y=[1 2;3 4];
octave.exe#> isvector(y)
ans  =  0

octave.exe#> z=[4];
octave.exe#> isvector(z)
ans  =  1
```

length(x)
Se *x* è un vettore determina il numero dei suoi elementi.

log(x)
Calcola il logaritmo naturale di *x*.

log2(x)
Calcola il logaritmo in base 2 di *x*.

log10(x)
Calcola il logaritmo in base 10 di *x*.

max(x)
Se *x* è un vettore ritorna il valore massimo degli elementi di *x*.
Se *x* è una matrice determina il massimo di ogni colonna e lo pone in un vettore riga.

```
Octave.exe> x = [1   5   -1   3];
octave.exe> max(x)
ans =
      5
octave.exe> y = [ -4   5   11;55 -2  6; 22 13  -9]
y  =
        -4        5      11
        55       -2       6
        22       13       9
octave.exe> max(y)
ans  =
      55     13    11
```

mean(x)
Calcola la media degli elementi di *x*. Vedere la sezione *Statistica*.

median(x,dim,opt)
Calcola la mediana degli elementi di *x*. Vedere la sezione *Statistica*.

mike_ven_s.maria.c.v@libero.it

min(x)
Se *x* è un vettore ritorna il valore minimo degli elementi di *x*. Se *x* è una matrice determina il minimo di ogni colonna e lo pone in un vettore riga.

```
Octave.exe> x = [1 5 -1 3];
octave.exe> min(x)
ans =
      -1

octave.exe> y = [ -4 5 11;55 -2 6; 22 13  -9]
y  =
        -4        5      11
        55       -2       6
        22       13       9

octave.exe> min(y)
ans  =
      -4     -2    6
```

mod(a,b)
Restituisce il resto della divisione intera tra *a* e *b*.

```
octave.exe1> mod(6,2)
ans = 0

octave.exe2> mod(7,3)
ans = 1
```

polyderiv(x)
Calcola la derivata simbolica del polinomio *x*. Vedere la sezione *Calcolo numerico*

polyinteg(x)
Determina l'integrale indefinito del polinomio *x*. Vedere la sezione *Calcolo numerico*

polyval(x,a)
Calcola il valore del polinomio *x* in *x=a*. Vedere la sezione *Calcolo numerico*.

pow2(x)
Calcola la potenza di 2.

```
octave.exe2> pow2(3)
ans = 8
```

pow2(a,x)
Calcola $a \cdot 2^x$.

```
octave.exe1> pow2(7,3)
ans = 56
```

primes(n)
Ritorna tutti i primi fino a *n*.

```
octave.exe#> primes(17)
ans  =
       2   3   5   7   11   13   17
```


mike_ven_s.maria.c.v@libero.it

printf()

```
printf(“\a”)   - nota dell'altoparlante
printf(“\n)    - stampa con ritorno a capo
printf(“ciao \n”) - stampa ciao con un ritorno a capo
printf(“ciao”)   - stampa ciao senza ritorno a capo
```

pwd

Octave usa una sua cartella per salvare e leggere i dati attuali. Per conoscere quale cartella sta usando Octave digitare il comando *pwd*:

```
octave.exe.#1> pwd
ans = F:\Programmi\Octave
```

Per cambiare la cartella di lavoro di Octave usare il commando cd:

```
octave.exe.2> cd “f:/Programmi”
```

Per controllare il cambiamento:

```
octave.exe.3> pwd
ans = F:\Programmi
octave.exe.4>
```

rand

Ritorna un numero casuale nell’intervallo (0 e 1).

```
octave.exe:##> rand
ans = 0.38107

octave.exe:##> rand(1)
ans = 0.30881

octave.exe:##> rand(1:2)
ans =
       0.43330   0.76628

octave.exe:##> rand(2:2)
ans =
       0.77854    0.21468
       0.97896    0.43695

octave.exe:##> u=rand(2:2)
u =
       0.778544  0.21468
       0.97896   0.43695
```

rank(x,tol)

Calcola il rango di *x*.

real(z)

Ritorna la parte reale di *z*.

```
octave.exe:#1> z=-3+5i
z  = -3 + 5i
```


mike_ven_s.maria.c.v@libero.it

```
octave.exe:#2> real(z)
ans  =  -3
```

rem(x,y)

Ritorna il resto di *x/y* calcolato con la formula: *x-y.*floor(x./y)*. Se gli argomenti non hanno la stessa dimensione o se sono complessi, viene segnalato un errore.

```
octave.exe> x = [10  11  12;13  14  15;16  17  18]
x =
    10    11    12
    13    14    15
    16    17    18

octave.exe> x = [1  2  3;4  5  6;7  8  9];
y  =
      1  -2   3
      4   5  -6
     -7   8   9

octave.exe> fmod(x,y)
ans  =
        0   1   0
        1   4   3
        2   1   0
```

rem(a,b)

Questa funzione agisce su due argomenti *a* e *b* che non sono matrici. Ritorna il resto della divisione di *a* per *b*.

```
octave.exe:##> rem(7,2)
ans = 1

octave.exe:##> rem(12,5)
ans = 2

octave.exe:##> rem(21,9)
ans = 3
```

roots(x)

Calcola le radici del polinomio *x*. Vedere la sezione *Calcolo numerico*.

round(x)

Questa funzione agisce su un solo argomento *x*. Ritorna l'intero più vicino a *x*.

```
octave.exe#> x = [-0.7  0.7; -2.4  2.4];
ans =
       -0.7   0.7
       -2.4   2.4

octave.exe#4> y=round(x)
y =
       -1     1
       -2     2
```

sign(x)

Questa funzione agisce su un solo argomento *x* e calcola il *signum* di *x*. Ritorna +1 se x>0, -1 se x<0, 0 se x=0. La funzione *sign* è così definita:

mike_ven_s.maria.c.v@libero.it

$$signum(x) = \begin{cases} 1 & se\ x > 0 \\ 0 & se\ x = 0 \\ -1 & se\ x < 0 \end{cases}$$

```
octave.exe#> x = [-0.7  0.7; -2.4  2.4];
ans =
       -0.7   0.7
       -2.4   2.4

octave.exe#> sign(x)
ans =
       -1     1
       -1     1
```

sin(x)

Ritorna il seno di *x* con *x* espresso in radianti.

Inseriamo nel sistema la matrice quadrata

$$x = \begin{pmatrix} \frac{pi}{2} & \frac{-pi}{2} \\ \frac{-pi}{4} & \frac{pi}{4} \end{pmatrix}$$

e usiamola come argomento della funzione seno.

```
octave.exe:##> x=[pi/2  –pi/2; -pi/4  pi/4 ]
ans =
      1.57080         -1.57080
     -0.78540          0.78540
```

Introducendo il comando

```
octave.exe:##> sin(x)
ans =
       1.00000         -1.00000
      -0.70711          0.70711
```

Octave risponde calcolando il seno di ogni elemento della matrice X.

sind(x)

Ritorna il seno di *x* con *x* espresso in gradi.

```
octave.exe:##> sind(30)
ans   =   0.5000
```

size

Per conoscere le dimensioni di una matrice usare **size** nella seguente forma:

```
[r,c]=size(nome_matrice).
```

Octave risponderà ponendo il numero di righe e di colonne nelle variabili *r* e *c* che abbiamo usato. La prima conterrà il numero di righe, la seconda il numero di colonne della matrice passata come argomento a **size**.

```
octave.exe1> a = [1  2; 3  4];
octave.exe2> [r,s] = size(a)
```


mike_ven_s.maria.c.v@libero.it

```
r = 2
s = 2
```

Le variabili *r* e *s* possono avere un nome diverso:

```
octave.exe3> b = [5;7;9]
b =
       5
       7
       9
octave.exe4> [m,n] = size(b)
m = 3
n = 1
```

sqrt(x)

Questa funzione agisce su un solo argomento *x*. Calcola la radice quadrata di *x*. Se *x* è negativo il risultato è complesso. Se *x* è un vettore o una matrice, calcola la radice quadrata di ogni elemento e mette il risultato in un vettore con le stesse dimensioni di *x*.

```
octave.exe> sqrt(49)
ans =  7
octave.exe> sqrt(-49)
ans =  0 + 7i
octave.exe> x = [-2,3-5i]
ans =
    -2 + 0i    3 – 5i
octave.exe> sqrt(x)
ans =
     0.0000 + 1.4142i    2.1013 – 1.1897i

octave.exe> x = [4   9;36   81]
x=
      4    9
     36   81

octave.exe> sqrt(x)
ans =
      2    3
      6    9
```

Inseriamo nel sistema la matrice

$$x = \begin{pmatrix} 1 & -2 \\ 3 & 4 \end{pmatrix}$$

```
octave.exe:## > x=[ 1  -2; 3 4]
ans =
       1     -2
       3      4

octave.exe:##> sqrt(x)
ans =
       1.00000 + 0.00000i           0.00000 + 1.41421i
       1.73025 + 0.00000i           2.00000 + 0.00000i

octave.exe:##> z=sqrt(x)
z =
       1.00000 + 0.00000i           0.00000 + 1.41421i
       1.73025 + 0.00000i           2.00000 + 0.00000i
```


mike_ven_s.maria.c.v@libero.it

stairs(x)
Ritorna il grafico a gradini degli elementi del vettore *x*. Gli elementi di *x* vengono numerati automaticamente.

stairs(x,y)
Ritorna il grafico a gradini degli elementi dei due vettori *x* e *y*. Gli elementi del vettore *x* devono risultare in ordine crescente.

std(x)
Calcola la deviazione standard degli elementi di *x*. (Vedere la sezione *Statistica*)

sum(x,dim)
Se *x* è una matrice e *dim* è 1, somma gli elementi di ogni colonna. Se *x* è una matrice e *dim* è 2, somma gli elementi di ogni riga. Se *x* è una matrice e *dim* è omesso allora *sum(x)=sum(x,1)*. Se *x* è un vettore somma gli elementi del vettore e *sum(x)=sum(x,2)*.

```
octave.exe:#1> x=[1 2 3; 4 5 6;7 8 9];

octave.exe:#2> sum(x)
ans =
       12   15   18

octave.exe:#3> sum(x,1)
ans =
       12   15   18

octave.exe:#4> sum(x,2)
ans =
        6
       15
       24

octave.exe:#5> sum(x,3)
ans  =
       1      2      3
       4      5      6
       7      8      9

octave.exe:#6> y=[1 2 3 4 5];

octave.exe:#7> sum(y)
ans  =  15

octave.exe:#8> sum(y,1)
ans  =
       1   2   3   4   5

octave.exe:#9> sum(y,2)
ans  =  15
```

tan(x)
Ritorna la tangente di *x* con *x* espresso in radianti.

tand(x)
Ritorna la tangente di *x* con *x* espresso in gradi.

```
octave.exe:##> tand(45)
ans   =   1.0000
```


mike_ven_s.maria.c.v@libero.it

trace(x)
Calcola la traccia di *x*, cioè la somma degli elementi della diagonale principale di *x*.

```
octave.exe:##>  x=[1  2  3; 4  5  6; 7  8  9]
x  =
        1   2    3
        4   5    6
        7   8    9

octave.exe:##>  trace(x)
ans  =  15

octave.exe:##>  y=[ 4  5  6];

octave.exe:##>  trace(y)
ans  =  4

octave.exe:##>  t=[10];

octave.exe:##>  trace(t)
ans  =  10
```

var(x)
Calcola la varianza dei valori di *x*. (Vedere la sezione *Statistica*.)

^2
L'operatore ^2 agisce sull'argomento *x* e ritorna il quadrato di *x*.

```
octave.exe:##> (4)^2
ans = 16

octave.exe:##> (-3)^2
ans = 9
```

Se *x* è una matrice quadrata di ordine maggiore di uno, l'operazione *x^2* è l'operazione di prodotto della matrice *x* per se stessa.

```
octave.exe:## > x=[ 1  -2; 3 4]
ans =
       1      -2
       3       4

octave.exe:## > y=x.^2
y =
        1       4
        9      16
```

.^n
Loperatore *.^n* ritorna una matrice i cui elementi sono la potenza ennesima, elemento per elemento, degli elementi della matrice *x*. Non si confonda questo operatore con *^n*.

Questa funzione agisce sull'argomento *x*. Ritorna la potenza ennesima di *x*, elemento per elemento.

```
octave.exe:## > x=[ 1  -2; 3 4]
ans =
       1   -2
       3    4
```


mike_ven_s.maria.c.v@libero.it

```
octave.exe:## > y=x.^3
y =
      1    -8
      27   64
```

^n

L'operatore *^n* esegue la potenza ennesima di *x* se *x* è una matrice con un solo elemento.

```
octave.exe:##> (-3)^3
ans = -27
```

Se *x* è una matrice quadrata esegue la potenza ennesima della matrice, riga per colonne.
Ad esempio se *x* è una matrice quadrata e *n=3*, l'operazione *x^3* è l'operazione di prodotto di matrici *x·x·x*, riga per colonne.

```
octave.exe:##> x=[ 1  -2; 3 4]
octave.exe:##> x^3
ans =
         -35   -30
          45    10
```

10. - Calcolo numerico

10.1 - Rappresentazione di un polinomio

Un polinomio viene rappresentato in un vettore riga con i suoi coefficienti scritti secondo l'ordine decrescente delle potenze della variabile.

Si consideri il polinomio nella variabile x:

$$x^3-5x^2+3x-7$$

Inseriamo i suoi coefficienti in ordine decrescente in un vettore *x*

```
Octave.exe:##> x=[1  -5   3  -7]
x =
    1   -5    3   -7
```

10.2 - Calcolo del valore di un polinomio

Per calcolare il valore del polinomio in 10.1 scriviamo

```
Octave.exe:##> polyval(x,1)
Ans =  -8
octave.exe:##> y= polival(x,0)
y = -7
```

Invece di inserire il nome della matrice si può inserire tutta la matrice dei coefficienti del polinomio

```
Octave.exe:##> polyval([1  -5  3  -7],0)
Ans = -7
```


mike_ven_s.maria.c.v@libero.it

10.3 - Radici di un polinomio

Per determinare le radici del polinomio x^3-5x^2+3x-7 si deve usare il comando *roots*. Si inserisca nel sistema la matrice *x* dei coefficienti e poi si dia a *roots* l'argomento x.

```
Octave.exe:##> x=[1  -5  3  -7]
x =
     1   -5   3   -7

Octave.exe:##> roots(x)
ans =
        4.67857 + 0.00000i
        0.16071 + 1.21258i
        0.16071 + 1.21258i
```

10.4 - Moltiplicazione di polinomi

Il comando per moltiplicare due polinomi è *conv(x,y)*, dove *x* e *y* sono le matrici dei coefficienti dei due polinomi. Si considerino i due polinomi

$$x^2-3x+2$$

e

$$x^2-x-1.$$

Eseguiamo con Octave il loro prodotto.

```
Octave.exe:##> x=[1  -3  2]
x =
     1   -3   2

Octave.exe:##> y=[1  -1  -1]
y =
     1   -1   -1

octave.exe:##>  z=conv(x,y)
z=
     1   -4   4   1   -2
```

Il vettore *z* contiene i coefficienti del polinomio $x^4-4x^3+4x+x-2$, che è il prodotto dei polinomi *x* e *y*.

10.5 - Divisione di polinomi

La divisione di due polinomi *x* e *y* si ottiene con il comando *deconv(x,y)*.

Dividiamo il polinomio

$$x^4-4x^3+4x+x-2$$

per il polinomio

$$x^2-x-1$$

```
Octave.exe:##1> x=[1  -4   4   1   2]
Octave.exe:##2> y=[1  -1   1]
Octave.exe:##3>  deconv(x,y)
ans =
      1   -3   2
```

Il quoziente della divisione dei due polinomi è il polinomio x^2-3x+2.

mike_ven_s.maria.c.v@libero.it

10.6 - Derivata di un polinomio

La derivata di un polinomio *x* si determina con il comando *polyderiv(x)*. Calcoliamo la derivata del polinomio

$$x^3-5x+1.$$

```
octave.exe:##> x=[1   0   -5   1]
x =
       1   0   -5   1

octave.exe:##> y=polyderiv(x)
y =
       3   0   -5
```

cioè la derivata del polinomio dato è $3x^2-5$.

Se si vuole calcolare il valore della derivata prima del nostro polinomio x^3-5x+1 in zero, bisogna usare il comando *polyval(y,0)*, dove *y* è la sua derivata prima.

```
octave.exe:##> z=polyval(y,0)
z =  -2
```

10.7 - Integrale indefinito di un polinomio

L'integrale di un polinomio *x* si determina con il comando *polyinteg(x)*. Consideriamo il polinomio $3x^2+2x-1$ e calcoliamo il suo integrale indefinito.

```
octave.exe:##> x=[3   2   1]
x =
       3   2   1
octave.exe:##> y=polyinteg(x)
y =
       1   1   -1   0
```

cioè l'integrale indefinito è x^3+x^2-x.

10.8 - Quadratura

Definiamo prima la funzione per la quadratura.

```
octave.exe:##> function y=f(x)
>  y=sin(x)
>  endfunction
octave.exe:##> [v,ier,nfunc,err]=quad("f",0,pi/2)
>  y  =  0,70711
>  y  =  0.85033
>  y  =  1
>  v  =  1
ier = 0
nfun = 21
err = 1.1102e-14
```


mike_ven_s.maria.c.v@libero.it

10.9 - Autovalori

Sia A una matrice quadrata di ordine *n* su un campo *K*. Uno scalare *λ* (*K* si chiama autovalore di *A* se esiste un vettore colonna non nullo in K_n tale che $Av= \lambda v$.
Ogni vettore che soddisfa questa relazione si chiama autovettore di *A* appartenente all'autovalore *λ*. Ogni multiplo scalare *kv* è un tale autovettore, poiché $A(kv)=k(v)=k(\lambda v)=\lambda(kv)$.
L'insieme E_λ di tutti gli autovettori che appartengono a *λ* è un sottospazio di K_n e si chiama autospazio di *λ*.
Determiniamo gli autovalori della matrice a = [1 2;3 2].

```
octave.exe:1> a = [1 2;3 2]
a =
       1      2
       3      2
octave.exe:2> eig(a)
ans =
      -1
       4
```

10.10 - Radici di equazioni non lineari

Per risolvere l'equazione

1) f(x)=0

usare la funzione *fsolve*:

```
[x,fval,info] = fsolve("espressione funzione",x0),
```

oppure

```
[x]=fsolve("espressione funzione",x0) ,
```

oppure

```
fsolve("espressione funzione",x0)  .
```

Il valore x_0 è l'approssimazione iniziale della radice dell'equazione 1). L'espressione che definisce la funzione deve essere racchiusa tra virgolette.

x	Valore della radice
fval	
info	1 se il metodo converge
x_0	Approssimazione iniziale della radice x

Risolvere l'equazione $(1/2)^x=\ln(x)$.

```
octave.exe:1>fsolve("(0.5)^x-log(x)",1.1)
ans = 1.4441
octave.exe:2>
```


mike_ven_s.maria.c.v@libero.it

_____

Risolvere l'equazione $(1/2)^x=\cos(x)$.

```
octave.exe:1>fsolve("(0.5)^x-cos(x)",1.1)
ans = 1.0769
octave.exe:2>
```

_____

Risolvere l'equazione $(1/2)^x=\cos(x)$.

```
octave.exe:1> [x,INFO, MSG]=fsolve("(0.5)^x-cos(x)",1.1)
x =  1.0769
INFO = 1.5543e-015
MSG =  1
octave.exe:2>
```

oppure

```
octave.exe:1> [x]=fsolve("(0.5)^x-cos(x)",1.1)
x =  1.0769
octave.exe:2>
```

10.11 - Risoluzione di sistemi lineari

Un sistema lineare può essere scritto in forma matriciale:

$$1) \qquad AX=B,$$

dove *A* è la matrice dei coefficienti, *X* è il vettore colonna delle indeterminate e *B* è il vettore colonna dei termini noti. Moltiplicando a sinistra ambo i membri della 1) per A^{-1}, si ha

$$2) \qquad A^{-1}AX=A^{-1}B$$

cioè

$$3) \qquad X=A^{-1}B.$$

Si voglia risolvere il sistema lineare:

```
2x-3y+4z=1
x-y+z=-2
3x+2y+2z=3.
```

Inseriamo la matrice *A* dei coefficienti:

```
Octave:1> A=[2,-3,4;1,-1,1;3,2,2];
```

Assicuriamoci che il sistema ammetta una soluzione unica:

```
octave:2> det(A)
ans=
    9
```

Inseriamo la matrice dei termini noti:

```
octave:3> B=[1;-2;3];
```

Possiamo calcolare la soluzione nel seguente modo:

mike_ven_s.maria.c.v@libero.it

```
octave:4>X=inv(A)*B
X =
       -3.2222
        2.5556
        3.7778
```

oppure

```
octave:5> X=A\B
 X =
       -3.2222
        2.5556
        3.7778
```

La soluzione esatta del sistema lineare è:

```
x=-29/9; y=23/9;  z= 34/9.
```

11. - Statistica

11.1 - Grafici a barre

Una sequenza osservata di dati può essere visualizzata in un grafico a barre con il comando

bar(x)

che ritorna il grafico a barre degli elementi del vettore $x=[x_1, x_2, x_3,\ldots,x_n]$. Gli elementi di *x* vengono numerati automaticamente. Il primo elemento di *x* ha ascissa 1 e ordinata x_1, il secondo elemento di *x* ha ascissa 2 e ordinata x_2, e così via.

Si supponga di avere osservato la sequenza

1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1

e la si rappresenti in Octave con un vettore riga *x*:

```
octave:1>x:=[1   2   3   4   5   6   5   4   3   2   1];
```

Per avere il grafico a barre di *x* scrivere

```
octave:2>bar(x)
```

Si otterrà il seguente grafico

mike_ven_s.maria.c.v@libero.it

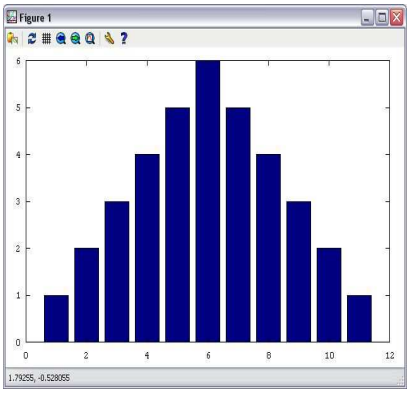


fig. 1 - Grafico a barre di un vettore.

nel quale le ascisse rappresentano il numero d'ordine dei dati nella sequenza osservata.

Se si hanno coppie di dati *x* e *y* il comando *bar(x,y)* visualizza un grafico a barre in cui le gli elementi di *x* sono in ascissa e gli elementi di *y* in ordinata. Gli elementi del vettore x devono essere scritti in ordine crescente.

Ad esempio

```
octave:1>x=[1 2 3 4 5 6];
octave:2>y=[123 234 456 345 180 99];
octave:3>bar(x,y)
```

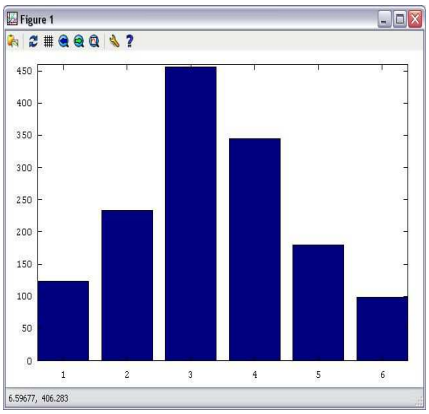


fig. 2 - Grafico a barre.

11.2 - Media

mean(x)
Se *x* è un vettore calcola la media aritmetica dei termini di *x*. Se *x* è una matrice calcola la media aritmetica dei termini di ogni colonna di *x* e mette il risultato in un vettore riga.

```
octave.exe:##> x=[1  2  3]
x  =
       1   2   3

octave.exe:##> mean(x)
ans  =  2

octave.exe:##>  y=[1  2  3; 4  5  6; 7  8  9]
y  =
```


mike_ven_s.maria.c.v@libero.it

```
      1   2    3
      4   5    6
      7   8    9

octave.exe:##> mean(x)
ans  =
       4   5   6
```

mean(x,dim,opt)
Se *x* è un vettore calcola la media di tipo *opt* dei termini di *x*. Se *x* è una matrice calcola la media di tipo *opt* dei termini secondo la dimensione verticale se *dim=1* e pone il risultato in un vettore riga, secondo la dimensione orizzontale se *dim=2*. Se *dim=1* il risultato è messo in un vettore riga, se *dim=2* il risultato è messo in un vettore colonna. I parametri *dim* e *opt* sono opzionali.

dim	**Calcolo**
1	per colonne
2	per righe

Tabella 1

opt	**Tipo di media**
“a”	aritmetica
“g”	geometrica
“h”	armonica

Tabella 2

11.3 - Mediana

median(x,dim)
Se x è un vettore calcola il valore della mediana degli elementi di x. Se x è una matrice calcola il valore della mediana di ogni colonna e pone il risultato in un vettore riga. Il parametro *dim* è opzionale, ma quando è presente calcola per colonne se vale 1, per righe se vale 2 (vedi tabelle 1 e 2 in 11.2).

```
octave.exe:##> x=[1  2  3  4  5  6];

octave.exe:##> median(x)
ans  =  3.5000

octave.exe:##> y=[1  2  3  4  5];
octave.exe:##> median(y)
ans  =  3

octave.exe:##>  z=[1  2  3; 4  5  6; 7  8  9]
z  =
      1   2    3
      4   5    6
      7   8    9

octave.exe:##> mean(z)
ans  =
      4   5   6

octave.exe:##> mean(z,1)
ans  =
      4   5   6

octave.exe:##> mean(z,2)
ans  =
       2
       5
       8
```


mike_ven_s.maria.c.v@libero.it

11.4 - Deviazione standard

std(x)
std(x,opt)
std(x,opt,dim)

Se *x* è un vettore calcola la deviazione standard degli elementi di *x.* Se *x* è una matrice calcola la deviazione standard di ogni colonna e pone il risultato in un vettore riga. L'argomento *opt* determina il tipo di deviazione da calcolare, si veda la seguente tabella 3. L'argomento *dim* determina la dimensione lungo la quale deve essere calcolata la deviazione standard (tabella 1 in 11.2).

opt	Tipo di deviazione
0	σ_{n-1}
1	σ_n

Tabella 3

Risulta *σ(x)=std(x)=std(x,0)=std(x,0,1)*.

```
octave.exe:##> x=[1  2  3  4  5  6];

octave.exe:##> std(x)
ans  =  1.8708

octave.exe:##>  y=[1  1.5  2;2.5  2.7  3;3  3.4   3.7]
ans  =
       1.0000  1.5000  2.0000
       2.5000  2.7000  3.0000
       3.0000  3.4000  3.7000

octave.exe:##>        std(y)
ans  =
      1.04083  0.96090  0.85440

octave.exe:##>        std(y,0)
ans  =
      1.04083  0.96090  0.85440

octave.exe:##>        std(y,0,1)
ans  =
      1.04083  0.96090  0.85440

octave.exe:##>        std(y,0,2)
ans  =
       0.50000
       0.25166
       0.35119

octave.exe:##>        std(y,1,2)
ans  =
       0.40825
       0.20548
       0.28674

octave.exe:##>        std(y,1,1)
ans  =
      1.84984   0.78457  0.69761
```


mike_ven_s.maria.c.v@libero.it

11.5 - Varianza

var(x)
var(x,opt)
var(x,opt,dim)
Se *x* è un vettore calcola la varianza dei valori di *x*. Se *x* è una matrice calcola la varianza degli elementi delle colonne e pone il risultato in un vettore riga. L'argomento *opt* determina il tipo di varianza da calcolare, si veda la tabella 4. L'argomento *dim* determina la dimensione lungo la quale deve essere calcolata la varianza (tabella 1 in 11.2).

opt	Tipo di deviazione
0	σ^2_{n-1}
1	σ^2_n

Tabella 4

```
octave.exe:##> x=[1  2  3  4  5  6];
octave.exe:##> var(x)
ans = 3.5000
octave.exe:##> var(x,1)
ans = 2.9167
```

12. - Programmazione

12.1 - If … else selezione

In Octave la forma dell'istruzione *if* è la seguente:

If espressione
    Istruzioni
Elseif espressione
    Istruzioni
Else
    Istruzioni
End

```
octave.exe:##> x= 10; y=2
>  if x>y
>  z=100
>  else
>  z=0
>  end
z  =  100
```

12.2 - Cicli

12.2.1 - For
Il ciclo for viene usato per ripetere un numero predefinito di volte una o più istruzioni. La sua sintassi è

for varibile = vettore
    istruzione
end

mike_ven_s.maria.c.v@libero.it

```
octave.exe:##> for n=1:4  n  end
n=1
n=2
n=3
n=4
```

Per non stampare sul video ogni valore di n usate il punto e virgola come nel seguente esempio.

```
octave.exe:##> for n=1:15  n;  end
```

Chiedete poi al sistema il valore di n. Vi verrà restituito l'ultimo valore di n.

```
octave.exe:##> n
n = 15
```

Per numerare per due da zero a 10 scrivere

```
octave.exe:##> for n=1:2:10  n  end
n =  1
n =  3
n =  5
n =  7
n =  9
```

Provate il seguente esempio in cui la variabile *p* è un vettore con un unico elemento.

```
octave.exe:##> for n=1:4
p=factorial(n);
end

octave.exe:##> disp(p)
24
```

Se si indicizza la variabile *p*, come vettore di 4 elementi, con *disp(p)* si ottiene il valore di tutti i fattoriali degli interi da 1 a 4.

```
octave.exe:##> for n=1:4
p(n)=factorial(n);
end

octave.exe:##> disp(p)
1   2   6   24
```

12.2.2 - While

La sintassi di while è

while espressione
istruzioni
end

Si legge: mentre è vera l'espressione esegui l'istruzione.

```
octave.exe:##> a=4;
octave.exe:##> while a>1
>  a=a-1
>  end
a  =  3
a  =  2
a  =  1
```


mike_ven_s.maria.c.v@libero.it

12.3 - Scrivere una funzione

Una funzione può avere una delle seguenti strutture:

function nomefunzione
Corpo della funzione
endfunction

function nomefunzione (lista degli argomenti)
Corpo della funzione
endfunction

function variabile di ritorno = nomefunzione (lista degli argomenti)
Corpo della funzione
endfunction

function [lista variabili di ritorno] = nomefunzione (lista degli argomenti)
Corpo della funzione
endfunction

function [output-list]=nomefunzione(input-list)
istruzioni
 # commenti
endfunction

Scriviamo una semplice funzione che accetti in entrata le misure dei lati di un rettangolo e che dia in uscita l'area e il perimetro del rettangolo. Chiamiamola *calcola*.

```
Octave.exe1> function [area,perimetro]=calcola(a,b)
> area=a*b
> perimetro=2*a+2*b
> endfunction
```

Invochiamo la funzione appena definita, con gli argomenti *a* e *b*, misure dei lati del rettangolo:

```
octave.exe.2> calcola(3,4)
area = 12
perimetro = 12
ans = 12
```

Octave ha eseguito correttamente la funzione ponendo il valore dell'area e del perimetro del rettangolo nelle variabili *area* e *perimetro*.

Scriviamo adesso una nuova funzione per eseguire lo stesso tipo di calcolo. Chiamiamola *rettangolo*. Operiamo in modo che sia il sistema a chiedere in ingresso le misure dei lati.

```
octave.exe.1> function rettangolo
> a=input("inserisci il primo lato:   ")
> b=input("inserisci il secondo lato:   ")
> area=a*b
> perimetro=2*a+2*b
> endfunction
octave.exe2>
```

Nelle stringhe "*inserisci il primo lato:* " e "*inserisci il secondo lato:* " di avviso all'utente della funzione *rettangolo*, gli spazi dopo i due punti hanno lo scopo di migliorare la leggibilità. Proviamo la funzione appena creata. Digitiamone il nome *rettangolo* nella riga di comando e forniamo i dati richiesti.

mike_ven_s.maria.c.v@libero.it

```
octave.exe2> rettangolo
inserisci il primo lato:  7
a = 7
inserisci il secondo lato:  8
b = 8
area = 56
perimetro = 30
octave.exe.3>
```

Queste funzioni, una volta usciti da Octave, andranno perse. Per usarle in una diversa sessione di lavoro dovremo digitarle di nuovo. In generale è conveniente scrivere una funzione in un file. Apriamo un editor di testo, per esempio *SciTe*, e digitiamo la funzione dal nome *rettangolo* esattamente come segue:

```
function rettangolo
  a=input("inserisci il primo lato _ > ")
    b=input("inserisci il secondo lato _ > ")
      perimetro=2*a+2*b
    area=a*b
endfunction
```

poi salviamo il file con il nome *rettangolo.m* nella cartella di octave. Fate attenzione a dare al file salvato lo stesso nome della funzione aggiungendo l'estensione *.m*.

Scriviamo ancora come esempio una funzione che calcoli la media di più valori dal nome *media1* usando un editor di testo:

```
function media1(x)
     Media=sum(x)/length(x)
endfunction
```

salviamo il file con il nome media1.m nella cartella di octave. Creiamo un vettore *y* con qualche elemento e invochiamo la funzione *media1* con argomento *y*.

Esecuzione

```
octave.exe1> y = [1 2 3 4 5 6 7 8 9];
octave.exe2>media1(y)
Media = 5
octave.exe3>
```

Octave ha una funzione predefinita per calcolare la media: *mean(x)*.

13. - Esempi di funzioni

13.1 - Area di un rettangolo

```
function rettangolo1
     a=input("inserisci il primo lato _ > ")
     b=input("inserisci il secondo lato _ > ")
          area=a*b
         perimetro=2*a+2*b
endfunction
```


mike_ven_s.maria.c.v@libero.it

13.2 - Una funzione può chiamare un' altra funzione.

Facciamo un esempio.

```
function [area,perimetro]=calcola(a,b)
     area=a*b
     perimetro=2*a+2*b
endfunction

function rettangolo2
     a=input("inserisci il primo lato _ > ")
     b=input("inserisci il secondo lato _ > ")
calcola(a,b)
endfunction
```

13.3 - Fattoriale

```
function fat=fattoriale(n)
if (n>0)
     fat=n*fattoriale(n-1);
  else
     fat=1;
  end
endfunction
```

oppure

```
function risposta=factorial(n)
    if (n<0)
      risposta=-1;
      return
    endif
risposta=1;
  if (n==0)
      risposta=1;
      return
    endif
if (n>0)
   for i=1:n
     risposta=risposta*i;
    endfor
end
```

13.4 - **Syracuse** (Congettura di Collatz)

```
function syrac=syr(n)
 syrac=[n];
 while (n>1)
     if (mod(n,2)==0)
            n=n/2;
          else
            n=3*n+1;
     endif
  syrac=[syrac,n];
 endwhile
endfunction
```


mike_ven_s.maria.c.v@libero.it

Esecuzione

```
octave.exe1> syr(3)
ans =
       3   10   5   16   8   4   2   1
octave.exe2> x=syr(5)
x=
       5   16   8   4   2   1
```

13.5 - Grafico della sequenza di Collatz

```
function plotsyr(n)
   x=syr(n);
  plot(x)
endfunction
```

Ad esempio con *plotsyr(121)* si ottiene il grafico della sequenza generata dal numero intero 121 nell'algoritmo di Collatz. Le ordinate sono i numeri della successione e le ascisse il loro numero d'ordine.

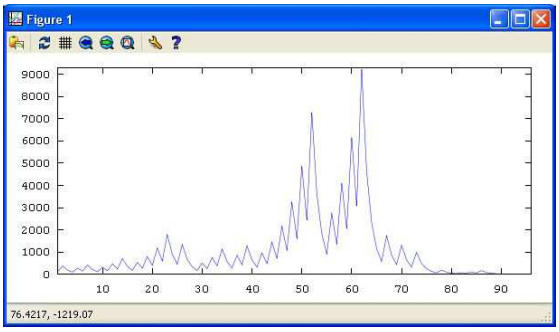


fig. 1 - Grafico della sequenza di Collatz generata dal numero intero 121

13.6 - Grafico della sequenza di Collatz con il comando *plot(x,y)*

```
function syrac=syrplot(n)
 x=[1];
 syrac=[n];
 t=1;
 while (n>1)
     if (mod(n,2)==0)
            n=n/2;
          else
            n=3*n+1;

          endif
          t=t+1;
          x=[x,t];
  syrac=[syrac,n];
endwhile
plot(x,syrac)
endfunction
```

Si noti che il primo elemento del vettore *x* è stato posto a 1 e il primo elemento del vettore *syrac* che conterrà gli elementi della successione è stato posto uguale a *n*. In questo modo, alla fine dell'iterazione, i due vettori conterranno lo stesso numero di elementi e il comando *plot(x,syrac)* potrà funzionare correttamente.

mike_ven_s.maria.c.v@libero.it

13.7 - Grafico della sequenza di Collatz con il comando *line(x,y)*

```
function syrac=syrline(n)
 x=[1];
 syrac=[n];
 t=1;
   while (n>1)
       if (mod(n,2)==0)
            n=n/2;
          else
            n=3*n+1;
       endif
            t=t+1;
            x=[x,t];
      syrac=[syrac,n];
   endwhile
line(x,syrac)
endfunction
```

Si veda la nota alla funzione *syrplot*.

13.8 - Potenza ad esponente intero

```
function pot=potint(n,i)
pot=1;
  if i>0
        for j=1:1:i
            pot=pot*n;
        endfor
   endif
  if i<0
        pot="errore - l'esponente i della funzione potint(x,i) deve essere un intero
positivo";
  endif
endfunction
```

Quando si userà questa funzione occorrerà passarle un esponente intero.

13.9 - Media

```
function media(x)
   media=mean(x)
endfunction
```

13.10 - Varianza

```
function varianza(x)
   varianza=var(x)
endfunction
```

13.11 - Somma degli elementi di un vettore

Per sommare gli elementi di un vettore riga o di un vettore colonna occorre prima conoscere il numero degli elementi del vettore *x*. Per questo si può invocare la funzione size nel seguente modo:

mike_ven_s.maria.c.v@libero.it

```
[r,c]=size(x).
```

Nelle variabili *r* e *c* saranno disponibili il numero di righe e di colonne del vettore. Poi con un ciclo *for* si potranno sommare tutti gli elementi del vettore. Se si tratta di un vettore colonna l'indice del ciclo andrà da 1 a *c*:

```
sum=0;
for i=1:c
  sum=sum+x(i);
endfor
```

Se in invece si tratta di un vettore colonna l'indice andrà da 1 a *r*:

```
sum=0;
for i=1:r
    sum=sum+x(i);
endfor
```

Ecco la codifica di due funzioni, la prima adatta a sommare gli elementi di un vettore riga (*h* sta ad indicare che è una somma orizzontale, *v* sta ad indicare che è una somma verticale)

```
function sum=sommavetth(x)
# somma gli elementi di un vettore riga
sum=0;
[r,c]=size(x);
for i=1:c
    sum=sum+x(i);
endfor
endfunction
```

```
function sum=sommavettv(x)
#somma gli elementi di un vettore colonna
sum=0;
[r,c]=size(x);
      for i=1:r
        sum=sum+x(i);
      endfor
endfunction
```

E' possibile riunire le due funzioni in un' unica funzione. Occorrerà verificare appena dopo l'invocazione di [r,c]=size(x) quale delle due variabili è uguale a 1. Se risulta *r=1* allora si tratta di un vettore riga e l'indice varierà fino a *c*, se invece risulta *c=1* allora si tratta di un vettore colonna e l'indice varierà fino a *r*. Ecco la codifica della funzione:

```
function sum=sommavett(x)
# somma gli elementi di un vettore riga o colonna
sum=0;
[r,c]=size(x);
         if r==1
             for i=1:c
                 sum=sum+x(i);
             endfor
         else
             for i=1:r
                  sum=sum+x(i);
             endfor
    endif
endfunction
```


mike_ven_s.maria.c.v@libero.it

13.12 - Generazione di numeri casuali in un prefissato intervallo

La funzione *rand* ritorna un numero casuale nell'intervallo (0;1). Usandola opportunamente possiamo ottenere numeri dell'intervallo [1, 10] moltiplicando semplicemente *rand* per 10 e arrotondando il risultato con *fix* (o *floor*): *fix(10*rand)*. La formula per generare un intero positivo tra 1 e 10 è: *1+fix(10*rand)*.

```
function casualint
for i=1:20 #ritorna 20 numeri casuali in [1;10]
  fix(10*rand)+1
endfor
endfunction
```

oppure

```
function casualint
for i=1:20 #ritorna 20 numeri casuali in [1;10]
  floor(10*rand)+1
endfor
endfunction
```

Con la seguente funzione *generacasuali(n)* si generano *n* casuali nell'intervallo [1,10].

```
function generacasuali(n)
   for i=1:n     #ritorna n casuali in [1;10]
     casualint;
   endfor
endfunction
```

Questa funzione chiama l'altra funzione *casualint* un numero *n* prefissato di volte che è passato come argomento in *generacasuali(n)*.

13.13 - Aggiungere elementi in un vettore

Siano x=[ ], y=3 e z=-5. La linea di codice

```
x=[x y]
```

costruisce il vettore x=[3] e se successivamente si pone

```
x=[x z]
```

si ottiene il vettore

```
x=[3 -5].
```

Scriviamo una funzione per costruire un vettore che contenga come elementi, in ordine crescente, i primi sette interi positivi e che mostri durante l'elaborazione la trasformazione del vettore riga:

```
function vector
    x=[]
        for i=1:7
            x=[x  i]
        endfor
endfunction
```

Se non si vuole vedere la trasformazione intermedia del vettore *x*, bisogna aggiungere il punto e virgola alla seconda e quarta riga, dopo il vettore *x*:

```
function vector
    x=[];
        for i=1:7
            x=[x  i];
        endfor
    x
endfunction
```


mike_ven_s.maria.c.v@libero.it

13.14 - Simulazione del lancio di un dado

```
function dado1
 y=[];                    # inizializza il vettore y
  for i=1:6               # inizia il ciclo per lanciare 6 volte il dado
    casuale=fix(6*rand)+1  # genera un intero positivo in [1;6]
    y=[y casuale];         # aggiunge a destra di y il numero generato
  endfor                   # fine del ciclo dopo i 6 lanci
  y                        # visualizza gli elementi del vettore y
endfunction


function dado2(n)          # n rappresenta il numero di lanci
 y=[];                     # inizializza il vettore y
  for i=1:n                # inizia il ciclo degli n lanci
    casuale=fix(6*rand)+1  # genera un intero positivo in [1;6]
    y=[y casuale];         # aggiunge a destra di y il numero generato
  endfor                   # termina il ciclo degli n lanci
  y                        # visualizza gli elementi del vettore y
endfunction



# LANCIO DI UN DADO
# Data: 26 luglio 2008
# Autore: Michele Ventrone
# Località: Santa Maria Capua Vetere (CE)
# Nazione: Italia
# Simulazione del lancio di un dado e legge dei grandi numeri
# Esempio: per lanciare un dado 60 volte scrivere: dado(60)
# Al termine dell'esecuzione si ottiene la frequenza relativa degli eventi
1,2,3,4,5,6
# Questo codice può essere usato da chiunque
function dado(n)                    # n rappresenta il numero di lanci
 a1=0;                              # si azzera il contatore degli 1
 a2=0;                              # si azzera il contatore dei 2
 a3=0;                              # si azzera il contatore dei 3
 a4=0;                              # si azzera il contatore dei 4
 a5=0;                              # si azzera il contatore dei 5
 a6=0;                              # si azzera il contatore dei 6
  for i=1:n                         # inizia il ciclo per lanciare il dado n volte
    casuale=fix(6*rand)+1;          # genera un intero positivo in [1;6]
    if casuale==1 a1=a1+1; endif    # si incrementa di 1 il contatore degli 1
    if casuale==2 a2=a2+1; endif    # si incrementa di 1 il contatore dei 2
    if casuale==3 a3=a3+1; endif    # si incrementa di 1 il contatore dei 3
    if casuale==4 a4=a4+1; endif    # si incrementa di 1 il contatore dei 4
    if casuale==5 a5=a5+1; endif    # si incrementa di 1 il contatore dei 5
    if casuale==6 a6=a6+1; endif    # si incrementa di 1 il contatore dei 6
  endfor                            # termine del ciclo dopo gli n lanci
  a1                                # visualizza la frequenza assoluta degli 1
  a2                                # visualizza la frequenza assoluta dei 2
  a3                                # visualizza la frequenza assoluta dei 3
  a4                                # visualizza la frequenza assoluta dei 4
  a5                                # visualizza la frequenza assoluta dei 5
  a6                                # visualizza la frequenza assoluta dei 6
  a1+a2+a3+a4+a5+a6                 # somma delle frequenze
endfunction
```


mike_ven_s.maria.c.v@libero.it