Laboratorio di Matematica Computazionale
A.A. 2008-2009

Lab. 1 - Introduzione a Matlab

Alcune informazioni su Matlab

Matlab è uno strumento per il calcolo scientifico utilizzabile a più livelli, dalla calcolatrice tascabile, alla simulazione ed analisi di sistemi complessi.

Il nome Matlab è una abbreviazione di Matrix-Laboratory.
La struttura di base è infatti la **matrice**: ogni quantità (variabile) viene trattata come una matrice di dimensioni $m \times n$.
Uno scalare reale è una matrice 1×1.

- In Matlab non è necessario dichiarare esplicitamente all'inizio del lavoro una variabile in termini delle sue dimensioni e del tipo dei suoi coefficienti (interi, reali, complessi)
  $\rightarrow$ notevole semplificazione

- è già predefinito un ampio insieme di matrici elementari (matrice identità, matrice nulla...)
  $\rightarrow$ matrici più complesse possono essere costruite rapidamente partendo da queste matrici fondamentali

- sono predefiniti vari operatori algebrici fra matrici di uso comune, quali ad esempio somma, prodotto, elevamento a potenza, nonché il calcolo del determinante o del rango di una matrice;
- sono predefinite numerose funzioni primitive di uso generale, dette **built-in functions**. Esse permettono di risolvere problemi complessi, ad esempio il calcolo degli autovettori ed autovalori di una matrice, la risoluzione efficiente di sistemi lineari, oppure la ricerca degli zeri di una funzione.

Le raccolte di funzioni dedicate ad uno specifico argomento vengono dette **toolboxes**. La finanza, la statistica, l'analisi dei segnali e delle immagini sono alcuni dei campi a cui sono dedicati dei toolboxes di Matlab

Dove trovare ulteriori informazioni su Matlab?

- sul sito ufficiale di Matlab `www.mathworks.com` sono disponibili numerosi manuali (in inglese) sia introduttivi che dedicati più approfonditamente ad aspetti specifici (programmazione, grafica, toolboxes...)
- sui siti di numerose università sono riportati tutorial ed esempi di problemi studiati con l'uso di Matlab

Matlab è un software a pagamento. Esiste un software gratuito, Octave, che ne riproduce buona parte delle funzioni fondamentali (con minime differenze di sintassi e una grafica un po' più povera). Octave può essere scaricato alla pagina web `www.octave.org`.

Per iniziare...

All'avvio di Matlab appare il prompt `>>`, ovvero la linea da cui digitare le istruzioni nello spazio di lavoro.

Il comando `demo` mostra degli esempi significativi di possibili applicazioni del software. Il comando `doc` introduce ad alcuni aspetti di base di Matlab e mostra quali pacchetti (toolboxes) siano installati nella versione in uso.

L'**`help`** (`doc`) di MATLAB permette di ottenere informazioni dettagliate su qualsiasi comando.
Ad esempio: **`help sqrt`** (oppure `doc sqrt`). Il solo comando **`help`** elenca gli argomenti per i quali è disponibile la guida, suddivisi in grandi aree tematiche (funzioni elementari, trattamento di matrici, grafica...)

Alcuni trucchi utili...

- è possibile richiamare "storicamente" i comandi precedentemente digitati nella sessione di lavoro usando i tasti $\uparrow, \downarrow$
- è possibile spostarsi lungo la linea di comando corrente e modificare la riga scritta utilizzando i tasti $\rightarrow$, $\leftarrow$
- è possibile completare un'istruzione già precedentemente digitata scrivendone le prime lettere e utilizzando poi il tasto $\uparrow$

Alcuni comandi fondamentali da conoscere...

- l'istruzione `diary mywork.dat` apre il file di testo mywork.dat nel quale viene trascritto (a partire da quel momento) il flusso delle istruzioni digitate (è una cronaca del lavoro svolto). L'istruzione `diary off` interrompe la scrittura della cronaca e chiude il file mywork.dat
- l'istruzione `whos` elenca le variabili attualmente attive in memoria e dà alcune informazioni importanti sulle loro caratteristiche (tipo di oggetto, dimensioni in memoria..)
- l'istruzione `save area.mat` permette di salvare nel file binario area.mat il contenuto di tutte le variabili attive in memoria in quel momento.
- l'istruzione `save area.mat z x` salva le sole variabili `z` e `x`
- l'istruzione `load area.mat` ricarica le variabili salvate nel file area.mat e le rende attive in memoria (verificare con `whos`)
- il comando `quit` termina la sessione di lavoro e chiude Matlab.

Scalari in Matlab

In Matlab non è necessario definire e dichiarare le variabili.
Tutte le variabili vengono trattate in **doppia precisione** (8 byte), senza distinzione fra interi, reali e reali a doppia precisione.

Iniziamo ad usare Matlab come una semplice calcolatrice:
ad esempio scriviamo

```
>>z=3*2
```

assegnando così alla variabile `z` il valore 6.
Se scriviamo solamente

```
>>3*2
```

il valore 6 viene assegnato alla variabile `ans` (abbreviazione di answer). Tale variabile contiene sempre l'ultimo valore non esplicitamente assegnato dall'utente ad una variabile. Il `;` alla fine dell'istruzione sopprime la visualizzazione a schermo del risultato (ma non l'esecuzione effettiva dell'operazione!). Ad esempio, assegnamo alla variabile `a` il risultato di una certa operazione, senza visualizzarlo, e poi richiamiamo `a` (senza `;`) per vederne il valore

```
>>a=sqrt(100);
>>a
```

Se `a` e `b` sono due variabili scalari, abbiamo: la somma `a+b`, la sottrazione `a-b`, il prodotto `a*b`, la divisione `a/b`, la potenza `a^b`.

Ricordiamo che in Matlab vale la usuale precedenza fra operazioni, ad esempio la moltiplicazione (e divisione) ha precedenza sulla addizione (e sottrazione) e l'elevamento a potenza ha precedenza su addizione, sottrazione, moltiplicazione e divisione. Ad esempio:

```
>> 3+2*4
   11
>> 3*2^4
   48
```

Per alterare l'ordine delle operazioni ci si serve delle parentesi tonde. Anche quando non si vogliano alterare le precedenze, l'uso delle parentesi tonde è comunque sempre buona norma per chiarezza.

```
>> (3+2)*4
   20
>> (3*2)^4
   1296
```

Esercizi

- Posto $a = 3, b = 2$, calcolare $\dfrac{3}{a+b}$, $\dfrac{a+b}{2}$, $\dfrac{a+b}{2a}$, $\dfrac{1}{8^{\frac{1}{3}}}$, $\sqrt[4]{64}$
- Posto $x = 4, y = 2$, calcolare $\left(\dfrac{3}{x+y}\right)^3$
- Se $x = 10, y = 5, z = 2$, calcolare $\dfrac{3x-2y}{5z^2}$ $(= 1)$

- Per $a = 8$, calcolare $\dfrac{a + \sqrt[3]{a}}{2a + 4}$ $(= 0.5)$
- Se $a = \frac{1}{3}, b = \frac{1}{5}$, calcolare $\dfrac{a^{-3}}{(1 - b + 3a)^2}$ $(= 8.\overline{3})$

Variabili predefinite: sono `pi` (pigreco), `i,j` (unità immaginarie), `eps` (epsilon macchina). Ogni variabile può essere tuttavia sovrascritta, ad esempio possiamo assegnare `pi=5` (attenzione!). Per cancellare il valore di una variabile (o se è predefinita riportarla al suo valore di default) usiamo il comando `clear`. Ad esempio

```
>>pi
   3.1416
>>pi=5;
>> clear pi
>> pi
     3.1416
```

Il comando `clear all` cancella il valore di tutte le variabili (provare ad usare tale comando in combinazione con il comando `whos` che elenca le variabili presenti nello spazio di lavoro).

Vettori in Matlab

Per introdurre un **vettore riga** è sufficiente inserire i valori separati da spazi bianchi o virgole, ad esempio per introdurre $w \in \mathbb{R}^{1\times 3}$:

```
>> w=[1 2 3]
```

oppure

```
>> w=[1, 2, 3]
```

Per introdurre un **vettore colonna** basta inserire fra parentesi quadre i valori della componenti del vettore stesso separati da un punto e virgola, ad esempio per introdurre $v \in \mathbb{R}^{3\times1}$:

```
>> v=[1; 2; 3]
```

Il comando `v=[1:10]` genera un vettore riga di dieci componenti dato dai valori 1,2,...,10.
Il comando `v=[1:.5:10]` genera un vettore riga di venti componenti dato dai valori 1,1.5,2,2.5,...,9.5,10, ovvero con passo 0.5. La

sintassi generale è `v=[valore_iniz:passo:valore_finale]`. Il passo può essere anche negativo, ad ex. `v=[10:-.5:1];`

Il comando `linspace(valore_iniz, valore_fineale, N)` genera N valori equispaziati fra `valore_iniz` e `valore_finale` (estremi compresi). Ad esempio

```
>>  v=linspace(0,1,5)
         0    0.2500    0.5000    0.7500    1.0000
```

Per **accedere alla componente di un vettore**, ad esempio alla terza, e assegnare alla variabile `z` tale valore, scriviamo

`z=v(3)`.
Attenzione: in Matlab l'indicizzazione inizia da 1 e non da zero!

Nota: esiste in Matlab la parola chiave **`end`** per accedere all'ultimo elemento di un vettore. Ad ex., se **`v`** ha dieci elementi, **`v(end)`** equivale a **`v(10)`**.

Matlab produce un messaggio di errore quando si cerchi di accedere ad una componente non definita, ad esempio se `v` ha dieci elementi e vogliamo accedere a `v(11)`, oppure se vogliamo accedere a `v(0)` o a `v(-2)`.

Per controllare la dimensione di una variabile, usiamo il comando `size`, ad esempio `size(v)`. Questo comando è anche utile quando Matlab segnala un conflitto di dimensioni fra quantità che si vogliono manipolare.
Inoltre, dato un vettore `v`, il comando `length(v)` ne restituisce la lunghezza.

Il comando **`zeros(n,1)`** produce un vettore colonna di lunghezza n con elementi tutti nulli. Il comando **`zeros(1,n)`** produce un vettore riga di lunghezza n con elementi tutti nulli. Il comando **`ones(n,1)`** (**`ones(1,n)`**) genera un vettore colonna (riga) con tutte le componenti pari a 1.

Operazioni su vettori

Dato un vettore v di n componenti, si può calcolare in Matlab:

- vettore trasposto: `v'` (verificare le dimensioni di `v'`!)
- modulo del vettore $||v|| = \sqrt{\sum_{i=1}^{n} v_i^2}$: comando **`norm(v)`** (equivalente alla norma 2 del vettore: **`norm(v,2)`**)

Siano ora v, w due vettori riga di $\mathbb{R}^n$, con componenti v_i e w_i, $i = 1, \ldots, n$ rispettivamente. Si ha:

- somma algebrica $v + w = (v_1 + w_1, ..., v_n + w_n)$. In Matlab: `v+w`
- prodotto scalare $(v, w) = (v_1 w_1 + v_2 w_2 + ... + v_n w_n)$. In Matlab: `v*w'` (oppure `dot(v,w)`)
- prodotto vettoriale $(v \wedge w)$. In Matlab: `cross(v,w)`

Attenzione alle dimensioni dei vettori!

Esistono anche delle operazioni su vettori "componente per componente", che in Matlab si eseguono usando la sintassi "punto". Dati v, w vettori riga di $\mathbb{R}^n$, con componenti v_i e w_i, $i = 1, \ldots, n$, si ha

- prodotto componente per componente (attenzione: differente dal prodotto scalare!). Esso genera un vettore dato da $(v_1 w_1, v_2 w_2, ..., v_n w_n)$. In Matlab: `v.*w`. Se i due vettori non hanno la stessa dimensione, si genera un errore

- elevamento a potenza componente per componente: ex. vogliamo calcolare il cubo di ciascuna componente, ovvero calcolare il vettore $(v_1^3, v_2^3, ..., v_n^3)$. In Matlab: `v.^3`

Istruzioni di manipolazione di sottoblocchi di vettori e di concatenazione

Siano `v=[1 2 3 4 5]` e `w=[100 200]`. Per sostituire alle ultime due componenti di `v` le componenti di `w`, scriviamo

```
>> v=[1 2 3 4 5]; w=[100 200];
>> v(end-1:end)=w;
```

Per eliminare da `v` la terza e la quarta componente usiamo il vettore vuoto `[]`:

```
>> v=[1 2 3 4 5];
>> v(3:4)=[];
   v=
      1 2 5
```

Infine, per concatenare due vettori usiamo la sintassi

```
>> z=[v w]
     z=
       1 2 3 4 5 100 200
```

Esercizi

- Generare gli interi da 28 a 80 con passo 1

- generare gli interi da -13 a 75 con passo 2
- generare gli interi da 22 a -10 con passo -4
- generare 100 punti equispaziati tra 2 e 3
- generare 125 punti equispaziati tra -1 e 5
- generare i punti tra -2.7 a 8.3 con passo 1.5 (cosa si osserva?)
- generare 150 punti equispaziati tra -2 e 3
- sia $x = [-3, 5, 8, 0, 1, 5, -2, 4]$:

  a. imporre 6° elemento =100

  b. imporre $1^\circ, 2^\circ, 3^\circ$ elemento $= [5, 6, 7]$

  c. togliere 4° elemento

  d. aggiungere in testa $= [1, 2, 3]$

  e. aggiungere in coda $= [10, 11, 12]$

  f. togliere, con un solo comando, dal 4° al 7° elemento compresi

Matrici in Matlab (primi comandi)

Per assegnare le matrici

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

diamo i comandi, rispettivamente

```
>> A=[1 2 3; 4  5 6];
>> B=zeros(2,3);
```

Possiamo calcolare

```
>> C=A+B;
>> D=A*B';
```

oppure

```
>> A= eye(5);
>> B= rand(5);
>> C= B-A;
>> s=A(1,2)+C(3,3);
```

Istruzioni di manipolazione di sottoblocchi di matrici e di concatenazione

Sia `A=eye(4)` e `B=hilb(2)`. Per sostituire alle ultime due righe e colonne di `A` la matrice `B`, scriviamo

```
>> A=eye(4); B=hilb(2);
>> A(3:4,3:4)=B;
```

Per eliminare da `A` la terza colonna usiamo il vettore vuoto `[]`:

```
>> A=pascal(4);
>> A(:,3)=[];
```

Infine, per concatenare due matrici usiamo la sintassi (attenzione alle dimensioni!)

```
>> A=eye(3,2); B=zeros(3,4);
>> C=[A,B];
```

Esercizi

- Sia

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ -1 & 0 & -1 & 0 \\ 4 & 5 & 6 & 7 \\ 5 & 4 & 3 & 2 \end{bmatrix}$$

  a. assegnare il valore 100 agli elementi della 3° colonna
  b. assegnare il valore -3 agli elementi della 2° riga
  c. assegnare il valore $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ alla sottomatrice definita dalle colonne 2 e 3 e dalle righe 3 e 4
  d. sia $C = [1, 1, 1, 1]$ e $B = [0, 0, 0, 0, 0]^T$: costruire

$$H = \begin{bmatrix} A & \\ & B \\ C & \end{bmatrix}$$

- Assegnati i vettori $u = [1, 0, 2, -3]$ e $v = [3, 0, 2, 1]$

  a. calcolarne il prodotto scalare; cosa fornisce invece il prodotto $v * u$?

b. calcolare i vettori colonna z, w, y definiti, componente per componente, da

$$z_i = u_i * v_i,\ w_i = u_i^{v_i},\ y_i = u_i/v_i$$

- assegnate le matrici

$$A = \begin{bmatrix} 2 & 1 & 0 \\ 3 & 1 & 0 \\ -2 & 1 & 1 \end{bmatrix}, \quad E = \begin{bmatrix} 1 & -1 & 1 \\ 2 & -1 & 0 \\ 3 & 2 & 1 \end{bmatrix}$$

  a. calcolare i prodotti di matrici AE e EA; sono uguali?

  b. si indichi con B la matrice costituita dalle prime due colonne di A e con C la matrice costituita dalle ultime due righe di E. Calcolare i prodotti BC e CB: in cosa si differenziano?

- verificare le seguenti proprietà del determinante di matrici

  a. $\det(A) = \det(A^T)$

  b. se A è $n \times n$, $\det(\alpha A) = \alpha^n \det(A^T) \quad \forall \alpha \in \mathbb{R}$

  c. $\det(AE) = \det(A)\det(E)$

  d. $\det\left(A^{-1}\right) = 1/\det(A)$