3. Matrici e algebra lineare in MATLAB

Riferimenti bibliografici

- Getting Started with MATLAB, Version 7, The MathWorks, www.mathworks.com (Capitolo 2)
- Mathematics, Version 7, The MathWorks, www.mathworks.com (Capitolo 1)

Matrici, vettori e scalari

- La matrice è il tipo di dato fondamentale in MATLAB
- Le matrici sono rappresentate come array bidimensionali di numeri.
- Assumono un significato particolare le matrici 1 x 1, usate per rappresentare gli scalari e le matrici con una sola riga o una sola colonna, usate per rappresentare i vettori.
- Mentre altri linguaggi di programmazione operano generalmente su scalari, il linguaggio di programmazione MATLAB consente di operare direttamente su matrici.

Creare matrici

Si può creare una matrice in MATLAB:

- Inserendo esplicitamente i dati nella matrice
- Caricando una matrice da un file di dati esterno
- Generando una matrice per mezzo di una delle funzioni predefinite disponibili in MATLAB
- Generando una matrice in funzioni predisposte dall'utente e salvate in un M-file

Creare matrici inserendo manualmente i dati

- I dati si inseriscono nella Finestra dei Comandi seguendo queste regole:
  - Gli elementi all'interno di una riga sono separati da virgola (,) o spazio
  - Le righe sono separate dal punto e virgola (;)
  - L'intera lista di elementi deve essere racchiusa tra parentesi quadre ([ ])

Creare matrici inserendo manualmente i dati

- Esempio: definiamo una variabile A alla quale assegnamo la seguente matrice:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

```
A = [1 2 3; 4 5 6; 7 8 9]
```

Generare matrici usando le funzioni predefinite

- MATLAB fornisce le seguenti funzioni per generare matrici:
  - zeros: genera una matrice in cui tutti gli elementi hanno valore zero
  - ones: genera una matrice in cui tutti gli elementi hanno valore uno
  - rand e randn: generano una matrice in cui tutti gli elementi assumono un valore casuale secondo una distribuzione rispettivamente uniforme e normale
  - eye: genera una matrice in cui tutti gli elementi sulla diagonale principale valgono uno e gli altri elementi valgono zero

Generare matrici usando le funzioni predefinite

- Esempio: creiamo una matrice di 3 righe e 4 colonne i cui elementi valgano tutti zero

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

```
B = zeros(3, 4)
```

Generare matrici usando le funzioni predefinite

- Esempio: creiamo una matrice di 3 righe e 4 colonne i cui elementi valgano tutti uno

$$C = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

```
C = ones(3, 4)
```

Generare matrici usando le funzioni predefinite

- Esempio: creiamo due matrici di 7 righe e 5 colonne i cui elementi sono numeri casuali.

  La prima matrice contiene elementi generati secondo una distribuzione uniforme, la seconda elementi generati secondo una distribuzione normale

```
D = rand(7, 5)
```

```
E = randn(7, 5)
```

Generare matrici usando le funzioni predefinite

- Esempio: creiamo una matrice identità 5 x 5:

```
I = eye(5)
ans =
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     1     0
     0     0     0     0     1
```

Estrarre elementi dalle matrici

- Per estrarre l'elemento corrispondente alla riga i e alla colonna j della matrice A scriviamo

  A(i, j)

- Esempio: per la matrice A dell'esempio precedente

```
>> A(1, 3)
ans =
      3
```

- NOTA BENE: l' enumerazione degli indici delle righe e delle colonne in MATLAB **inizia da 1**!

Operatore ':'

- ':' è uno degli operatori MATLAB più importanti
- Ad esempio l'espressione 1:10 permette di generare un vettore contenente la successione dei numeri interi da 1 a 10:

```
>> 1 : 10
ans =
     1 2 3 4 5 6 7 8 9 10
```

- ':' può quindi essere usato per specificare range di valori

Operatore ':'

- Si possono ottenere successioni di numeri con passo non unitario specificandone il passo
- Ad esempio l'espressione 1:0.5:10 permette di generare un vettore contenente una successione di numeri da 1 a 10 con passo 0.5:

```
>> 1 : 0.5 : 10
ans =
      1.0000 1.5000 2.0000 2.5000 ...
```

- La sintassi dell'operatore ':' è pertanto
  start : step : stop

Operatore ':'

- L'operatore ':' può essere usato per estrarre sotto-matrici.
- Ad esempio con A(1:2, 3) si estrae il vettore 2x1 corrispondente alla prima e seconda riga ed alla terza colonna della matrice A.

```
>> A(1:2, 3)
ans =
      3
      6
```

Operatore ':'

- L'operatore ':' da solo si riferisce a tutti gli elementi di una riga o di una colonna.
- Ad esempio scrivendo A(2, :) si estrae il vettore 1x3 corrispondente alla seconda riga della matrice A.

```
>> A(2, :)
ans =
     4     5     6
```

Operatore ':'

- Espressioni costruite con l'operatore ':' possono essere usate all'interno di comandi.
- Ad esempio sum(A(:, 1)) calcola la somma degli elementi nella prima colonna della matrice A.

```
>> sum(A(:, 1))
ans =
     12
```

Eliminazione di righe e colonne

- È possibile eliminare righe e colonne da una matrice utilizzando le parentesi quadre [], attribuite alla riga/colonna che si vuole eliminare.
- Ad esempio, eliminiamo la seconda colonna della matrice A:

```
>> A(:, 2) = []
A =
     1     3
     4     6
     7     9
```

Dimensioni di una matrice

- Il comando size permette di ottenere informazioni sulle dimensioni di una matrice:
  - size(m, 1) restituisce il numero di righe della matrice m
  - size(m, 2) restituisce il numero di colonne di m
  - size(m) restituisce un vettore riga contenente il numero di righe e il numero di colonne della matrice m
- Esempio:

```
>> size(B, 1)
ans =
     3
>> size(B, 2)
ans =
     4
```

```
>> size(B)
ans =
     3     4
```

Somma e sottrazione di matrici

- I comandi di somma e sottrazione operano sulle matrici elemento per elemento.
- Definiamo ad esempio i vettori riga M = [1 2] e N = [3 4], e calcoliamo la somma M + N:

```
>> M = [1 2];
>> N = [3 4];
>> M + N
ans =
      4     6
```

Somma e sottrazione di matrici

- Calcoliamo ora la differenza:

```
>> N – M
ans =
      2     2
```

- Le operazioni di somma e sottrazione richiedono che le matrici che si sommano o si sottraggono abbiano tutte le stesse dimensioni.

Moltiplicazione di matrici

- La moltiplicazione di matrici è definita nel senso di moltiplicazione righe per colonne.
- Per poter effettuare la moltiplicazione il numero di colonne del primo fattore deve essere uguale al numero di righe del secondo fattore
- Il risultato sarà una matrice avente un numero di righe pari a quello del primo fattore e un numero di colonne pari a quello del secondo fattore
- Se R è una matrice m x n e S è una matrice n x p, Q = R * S sarà una matrice m x p.

Moltiplicazione di matrici

- Esempio: definiamo due matrici R e S e ne calcoliamo il prodotto Q = R * S

$$R = \begin{bmatrix} 0 & 1 \\ 3 & 0 \\ 5 & 2 \end{bmatrix} \qquad S = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$Q = RS = \begin{bmatrix} 1 \\ 6 \\ 12 \end{bmatrix}$$

```
>> R = [0 1; 3 0; 5 2];
>> S = [2; 1];
>> Q = R * S
Q =
     1
     6
     12
```

Prodotto elemento per elemento

- L'operatore .* consente di operare il prodotto elemento per elemento.
- Riconsideriamo ad esempio i vettori riga M = [1 2] e N = [3 4], e calcoliamo il prodotto elemento per elemento M .* N.

```
>> M .* N
ans =
     3     8
```

- Allo stesso modo l'operatore ./ consente di eseguire la divisione elemento per elemento.

Potenze di matrici

- Data la matrice quadrata A e il numero intero positivo p, A^p moltiplica A per sé stessa p-1 volte.
- Esempio: calcoliamo il quadrato della matrice A precedentemente definita.

```
>> A^2
ans =
     30    36    42
     66    81    96
     102   126   150
```

- Come per moltiplicazione e divisione, l'operatore .^ esegue l'elevamento a potenza elemento per elemento.

Operazioni con scalari

- Si può sommare, sottrarre, moltiplicare o dividere per uno scalare tutti gli elementi di una matrice.
- Esempio: sommiamo 10 a tutti gli elementi della matrice A precedentemente definita

```
>> A + 10
ans =
      11    12    13
      14    15    16
      17    18    19
```

- E' anche possibile assegnare un valore scalare a tutti gli elementi in un determinato range di indici: es. A(1:2, 1:2) = 0 assegna 0 ai quattro elementi in alto a sinistra nella matrice A.

Prodotto scalare

- L'operatore * applicato ad un vettore riga ed un vettore colonna consente di calcolare il prodotto scalare dei due vettori.
- Calcoliamo ad esempio il prodotto scalare di U = [3 5] e V = [1; 4]:

```
>> U = [3 5];
>> V = [1; 4];
>> U * V
ans =
      23
```

- NOTA BENE: L'ordine dei fattori è importante! Se digitiamo V * U, MATLAB calcola il prodotto tensoriale di U e V.

Concatenazione orizzontale

- Due matrici con ugual numero di righe possono essere concatenate orizzontalmente, cioè affiancate orizzontalmente, per formare una nuova matrice.
- Ad esempio, possiamo concatenare le matrici M(1x2) e N(1x2) precedentemente definite per ottenere una nuova matrice P (1x4).
- In MATLAB si esegue la concatenzazione orizzontale includendo le matrici da concatenare, separate da spazio, tra parentesi quadre:

```
>> P = [M N]
P =
     1     2     3     4
```

Concatenazione verticale

- Due matrici con ugual numero di colonne possono essere concatenate verticalmente, cioè affiancate verticalmente, per formare una nuova matrice.
- Ad esempio, possiamo concatenare le matrici M(1x2) e N(1x2) precedentemente definite per ottenere una nuova matrice P (2x2).
- In MATLAB si esegue la concatenzazione orizzontale includendo le matrici da concatenare, separate da punto e virgola (';'), tra parentesi quadre:

```
>> P = [M; N]
P =
     1     2
     3     4
```

Trasposizione

- L'operatore ' esegue la trasposizione di una matrice.
- Calcoliamo ad esempio la matrice trasposta della matrice R precedentemente definita:

$$R = \begin{bmatrix} 0 & 1 \\ 3 & 0 \\ 5 & 2 \end{bmatrix}$$

$$R^T = \begin{bmatrix} 0 & 3 & 5 \\ 1 & 0 & 2 \end{bmatrix}$$

```
>> R'
ans =
     0     3     5
     1     0     2
```

- Nota: nel caso di vettori, la trasposizione trasforma un vettore riga in un vettore colonna e viceversa.

Rango

- Data una matrice, il comando rank ne calcola il rango
- Calcoliamo ad esempio il rango della matrice T:

$$T = \begin{bmatrix} 10 & 0 & 2 \\ 3 & 3 & 1 \\ 4 & 1 & 0 \end{bmatrix}$$

```
>> T = [10 0 2; 3 3 1; 4 1 0];
>> rank(T)

ans =
     3
```

Determinante

- Data una matrice quadrata, il comando det ne calcola il determinante
- Calcoliamo ad esempio il determinante della matrice T, precedentemente definita

```
>> det(T)

ans =
        -28
```

Inversione

- Data una matrice quadrata non singolare il comando inv ne calcola l'inversa.
- Calcoliamo ad esempio l'inversa della matrice T:

```
>> inv(T)
ans =
         0.0357   -0.0714    0.2143
        -0.1429    0.2857    0.1429
         0.3214    0.3571   -1.0714
```

Diagonalizzazione

- Il comando eig permette di calcolare gli autovalori e gli autovettori di una matrice quadrata non singolare
- Esempio: calcoliamo il vettore lambda degli autovalori della matrice T:

```
>> lambda = eig(T)
lambda =
        10.8200
        -0.8532
         3.0332
```

Diagonalizzazione

- Per ottenere sia gli autovalori che gli autovettori della matrice T scriviamo:

```
>> [V, D] = eig(T)
V =
        -0.8580   -0.1801    0.0666
        -0.3742   -0.1134   -0.9704
        -0.3518    0.9771   -0.2321

D =
        10.8200         0         0
         0        -0.8532         0
         0              0    3.0332
```

- La matrice V contiene gli autovettori di T, mentre D è la matrice diagonale i cui elementi sulla diagonale principale sono gli autovalori di T

Diagonalizzazione

- Verifica: applicando la formula della diagonalizzazione si deve riottenere la matrice T:

$$T = VDV^{-1}$$

```
>> T = V * D * inv(D)
T =
        10.0000  0.0000   2.0000
        3.0000   3.0000   1.0000
        4.0000   1.0000   0.0000
```

- Si possono anche calcolare i coefficienti del polinomio caratteristico di una matrice per mezzo del comando poly.

Elementi sulla diagonale principale

- Il comando diag estrae in un vettore gli elementi sulla diagonale principale di una matrice.
- Esempio: diag(D) estrae gli elementi sulla diagonale principale della matrice D, cioè gli autovalori di T:

```
>> diag(D)
ans =
        10.8200
        -0.8532
         3.0332
```

- Il comando trace calcola la traccia di una matrice, ad esempio trace(D) calcola la somma degli autovalori di T.

Norme

- Il comando norm permette di calcolare la norma di un vettore v.
- Esempio: calcoliamo la norma 1, 2 e infinito del vettore v = [-1 2 0]

```
>> v = [-1 2 0];
>> [norm(v, 1) norm(v) norm(v, inf)]

ans =
        3.0000    2.2361    2.0000
```

- Lo stesso comando norm può essere usato per calcolare norme di matrici.