الجامعة المستنصرية

كلية الهندسة

قسم هندسة الموارد المائية

Lecture 4: Matrix Operations

المحاضرة الرابعة: عمليات المصفوفة

المنهاج الدراسي لمادة البرمجه والتطبيقات (ماتلاب)

الكورس الدراسي الاول / المرحلة الثانية

من اعداد د. صباح حسن لفته

b. Multiplication of Matrices:

MATLAB provides two ways to do multiplication between two matrices, each one has a unique advantage. The first mothed is called ***matrix multiplication*** which donated by (*). In order to use this procedure, the number of columns in first matrix must match the number of rows in second matrix, otherwise the multiplication can not be performed between the two matrices. The production of this process is a matrix with columns of first matrix and rows of second matrix. In matrix multiplication method, the elements of the rows in the first matrix are multiplied with corresponding columns in the second matrix. Each element in the (i, j)th location, in the resulting matrix C, is the summation of the products of elements in ith row of first matrix with the corresponding element in the jth column of the second matrix. The following example explain the method of matrix multiplication.

Let say we have two matrices A & B,

$$\overset{\substack{\textbf{Matrix A}\\ \textbf{(3x2)}}}{\begin{bmatrix}1 & 2\\ 3 & 4\\ 5 & 6\end{bmatrix}} \xrightarrow{\quad (*) \quad} \overset{\substack{\textbf{Matrix B}\\ \textbf{(2x3)}}}{\begin{bmatrix}2 & 4 & 6\\ 3 & 5 & 7\end{bmatrix}} \Rightarrow \overset{\substack{\textbf{Matrix C}\\ \textbf{(3x3)}}}{\begin{bmatrix}8 & 14 & 20\\ 18 & 32 & 46\\ 28 & 50 & 72\end{bmatrix}}$$

So, the size of the resulted matrix is three columns by three rows.

Example: Write a code script to do matrix multiplication between matrix X (4x3) and matrix Y (3x4),

```
>> %% Matrix multiplication between X & Y

>> X = [4, 7, 2; 1, 5, 11; 6, 4,1; 6, 8, 2];

>> Y = [2, 4, 3, 6; 1, 3, 7, 4; 2, 3, 12, 9];

>> Z = X * Y          ↵

   Z =

        19    43    85    70
        29    52   170   125
        18    39    58    61
        24    54    98    86
```

The second method of matrix multiplication is called ***dot product*** (element by element method) and represented by the (.*) operator. To use this method, its essential both matrices have similar size, that means this procedure is not applicable for matrices with different sizes. The produced matrix owns same number of columns and rows of the original matrices. The following example explain the method of matrix multiplication.

Example: Write a code script to do dot product between matrix A (3x3) and matrix B (3x3),

```
>> %% Dot Product Multiplication between A & B

>> A = [1, 2, 3; 3, 4, 5; 6, 7, 8];
>> B = [8, 7, 6; 5, 4, 3; 3, 2, 1];

>> C = A .* B        ↵

   C =

        8    14    18

       15     16    15

       18    14     8
```

3. Useful Matrix Operations:

MATLAB library contains many useful functions to do different matrix operations as follows:

Average of matrix elements (mean function)

This statement helps to find the average values of matrix elements and donated by ***mean(x)*** in which x is a matrix.

Example: Create a matrix with size of (3x3) and determine the average values in two cases: one for each column elements and second for all matrix element.

```
>> S=[1 2 3; 5 6 7; 3 1 2];
>> Avg1=mean(S);
>> Avg2=mean(Avg1)        ↵
```

MATLAB will execute the above code and return the following result:

```
   Avg1 =

            3     3     4

   Avg2 =     3.3333
```

Diagonal Function (diag)

Diag is another useful function use to display the matrix diagonal elements and represented by ***(diag(x))***.

Example: Write a code to determine the diagonal elements of the following matrices.

Matrix A **Matrix B**

$$\begin{bmatrix} 3 & 14 & 9 \\ 6 & 8 & 6 \\ 4 & 7 & 10 \end{bmatrix} \qquad \begin{bmatrix} 9 & 14 & 2 \\ 4 & 5 & 1 \\ 1 & 8 & 13 \end{bmatrix}$$

```
>> A = [3 14 9; 6 8 6; 4 7 10];
>> B = [9 14 2; 4 5 1; 1 8 13];
>> DA= diag(A);
>> DB= diag(B);      ↵
```

MATLAB will execute the above code and return the following result:

```
DA =
        3
        8
        10
DB =
        9
        5
        13
```

Minimum and Maximum Functions (min. & max.)

MATLAB offers another two helpful functions (***min.*** and **max.**), the first function is used to find the minimum value of the matrix elements, while the second one uses to find the maximum value among matrix elements.

Example: Write a code to determine the maximum and minimum values of the following matrices.

Matrix A **Matrix B**

$$\begin{bmatrix} 3 & 14 & 9 \\ 6 & 8 & 6 \\ 4 & 7 & 10 \end{bmatrix} \qquad \begin{bmatrix} 9 & 14 & 2 \\ 4 & 5 & 1 \\ 1 & 8 & 13 \end{bmatrix}$$

```
>> %% Script code to find minimum and maximum values
>> A = [3 14 9; 6 8 6; 4 7 10];
>> B = [9 14 2; 4 5 1; 1 8 13];
   >> MinA= min(A);
   >> MinB=min(B);
     >> MaxA=max(A);
     >> MaxB=max(B);
       >> MnA=min(MinA);
       >> MnB=min(MinB);
       >> MxA=max(MaxA);
       >> MxB=max(MaxB);              ↵
```

MATLAB will execute the above code and return the following result:

```
MnA =     3
MnB =     1
MxA =     14
MxB =     14
```

Summation Function (sum)

Sum function is used in MATLAB to summate elements of any matrix by column and represented by ***sum(x)***, in which all elements arranged in columns will be summated. Also, summation of the matrix elements can be calculated using a special syntax (e.g. ***sum(sum(x))***).

Example: Write a code to determine the summation of the two matrices in two cases: first find the total value for each column, then determine the total value of each matrix.

Matrix A **Matrix B**

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

```
>> %% Script code to find summation values of matrices (A & B)
>> A = [1 9 7; 6 8 6; 5 4 2];
>> B = [4 8 4; 4 2 1; 3 5 9];
  >> SuA=sum(A);
  >> SuB=sum(B);
  >> STA=sum(SuA);
  >> STB=sum(SuB);        ↵
```

The results will be as followed,

```
   SuA =     12    21    15
   STA =     48
   SuB =     11    15    14
   STB =     40
```

Sort Function (sort)

To arrange matrix elements, we need to use sort function which is donated by (***sort(x)***) as we did with the vectors. The difference here that elements of each column in the matrix will be arranged from smaller to larger value.

Example: Write a code to sort the elements of the two matrices in two cases.

$$\begin{array}{cc} \textbf{Matrix A} & \textbf{Matrix B} \\ \begin{bmatrix} 1 & 9 & 7 \\ 6 & 8 & 6 \\ 5 & 4 & 2 \end{bmatrix} & \begin{bmatrix} 4 & 8 & 4 \\ 4 & 2 & 1 \\ 3 & 5 & 9 \end{bmatrix} \end{array}$$

```
>> %% Script code to sort the elements of matrices (A & B)
>> A = [1 9 7; 6 8 6; 5 4 2];
>> B = [4 8 4; 4 2 1; 3 5 9];
>> SortA=sort(A);
>> SortB=sort(B);        ↵
```

```
   SortA =
         1    4    2
         5    8    6
         6    9    7
     SortB =
         3    2    1
         4    5    4
         4    8    9
```

Helpful example

Example: Generate 8 normal vectors using semi-colon method, so that each vector contains 10 elements, then do the following:

1- Construct tow matrices from the constructed vectors, the size of each one is (10x4).

2- Create matrix with size (4x4) by multiplying the two matrices generated in (1).

3- Determine the inverse of the (4x4) matrix and transpose it.

4- Display the diagonal elements of the third matrix and sum them.

5- Find the minimum, maximum, mean, and the determinant of the three matrices.

6- Display the elements of third column of first matrix, second column of second matrix, and last rows of the third matrix.

```
>> %% Script code to do matrix operations
>> x1=1:10;
>> x2=-10:-1;
>> x3=1:0.5:5.5;
>> x4= 2:2:20;
>> x5=-20:2:-2;
>> x6=1:3:28;
>> x7=-5:1:4;
>> x8=0:0.2:1.8;
>> X=[x1´ x2´ x3´ x4´];
>> Y=[x5´ x6´ x7´ x8´];
>> Z=X´*Y;
>> Zinv=inv(Z);
>>Ztrns=Z´;
>> Zdiag=diag(Z);
>> ZsumD=sum(Zdiag);
>> Xmin=min(X);
>> Xmax=max(X);
>> Xmean=mean(mean(X));
>> Xdet=det(X);
>> Ymin=min(Y);
>> Ymax=max(Y);
>> Ymean=mean(mean(Y));
>> Ydet=det(Y);
>> Zmin=min(Z);
>> Zmax=max(Z);
>> Zmean=mean(mean(Z));
>> Zdet=det(Z);
>> Xdis=X(:,3);
>> Ydis=Y(:,2);
>> Zdis=Z(4,:);
```