Lecture Series – 3

Matrices and Arrays in MATLAB

by

Shameer Koya

MATLAB Matrices

- MATLAB treats all variables as matrices. For our purposes a matrix can be thought of as an array, in fact, that is how it is stored.
- Vectors are special forms of matrices and contain only one row OR one column.
- Scalars are matrices with only one row AND one column

Arrays

A rectangular arrangement of numbers is called an **array.**

$$\begin{bmatrix} 6.2 & 0.1 \\ -4.1 & 7.2 \\ 5.0 & 10.6 \\ -7 & 0.0 \end{bmatrix}$$

This is a 4-by-2 array. It has 4 rows, and 2 columns.

The (3,1) entry is the entry in the 3rd row, and 1st column

Later, we will have rectangular arrangement of more general objects. Those will also be called arrays.

Entering a Matrix in MATLAB

- Enter an explicit list of elements
- Load matrices from external data files
- Using built-in functions
- Using own functions in M-files

- A matrix can be created in MATLAB as follows (note the commas AND semicolons):

» matrix = [1 2 3 ; 4 5 6 ; 7 8 9]

Entering a Matrix in MATLAB ....

- >> A = [2 -3 5; -1 4 5]
- A =
- 2 -3 5
- -1 4 5
- >> x = [1 4 7]
- x =
- 1 4 7
- >> x = [1; 4; 7]
- x =
- 1
- 4
- 7

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

$$\mathbf{x} = [1\ 4\ 7]$$

$$\mathbf{x} = \begin{bmatrix} 1 \\ 4 \\ 7 \end{bmatrix}$$

Generating Matrices Using Built-in Functions

Matrix with ZEROS:

zeros (r, c)

Matrix with ONES:

ones (r, c)

IDENTITY Matrix:

eye (r, c)

Random Matrix:

rand(r, c)

r → Rows

c → Columns

Examples

>> A = ones(3,2);

>> B = zeros(3,4);

>> C = rand(2,5);

>> C = magic(4); - Durer's matrix

Built-in Functions to Handle Matrices / Arrays

- Sum() – sum of elements in a column
- '–Transpose of a matrix
- Diag() – diagonal elements of matrix
- Size() – size (dimensions) of matrix
  - size(A)
  - ans =

    2 4 size of A: 2 rows, 4 columns
- Det() – determinant of a matrix.
- Inv() – inverse of a matrix

The Colon operator

- A colon notation is an important shortcut, used when producing row vectors

```
>> 2:5
ans =
2 3 4 5
>> -2:3
ans =
-2 -1 0 1 2 3
```

format - first:step:last

```
>> 0.2:0.5:2.4
ans =
0.2000    0.7000    1.2000    1.7000    2.2000
>> -3:3:10
ans =
-3 0 3 6 9
>> 1.5:-0.5:-0.5    (negative step is also possible)
ans =
1.5000    1.0000    0.5000    0    -0.5000
```

Subscripts and Extracting a Sub-Matrix

- A(i, j)- element i^{th} row, j^{th} column
- A(k)

- A portion of a matrix can be extracted and stored in a smaller matrix by specifying the names of both matrices and the rows and columns to extract. The syntax is:

  A1 = A (r1 : r2 , c1 : c2)

  where r1 and r2 specify the beginning and ending rows and c1 and c2 specify the beginning and ending columns to be extracted to make the new matrix.

Extracting a Sub-Matrix ....

- >> A(3,:)   % extract the 3rd row of A
- ans =

  7 8 9

- >> A(:,2)   % extract the 2nd column of A
- ans =

  2

  5

  8

- >> A([1,3],1:2)   % extract a part of A
- ans =

  1 2

  7 8

```
» A(4,1)
???  Index exceeds matrix dimensions.
```

Adding And Deleting Elements

- Indexing can be used to add and delete elements from a matrix.
- >> A(5,2) = 5 % assign 5 to the position (5,2); the uninitialized elements become zeros
- A =

    1 2 3
    4 5 8
    7 8 9
    0 0 0
    0 5 0

- >> A(4,:) = [2, 1, 2]; % assign vector [2, 1, 2] to the 4th row of A
- >> A(5,[1,3]) = [4, 4]; % assign: A(5,1) = 4 and A(5,3) = 4

Adding And Deleting Elements …

- A(1,2) =5 – will replace the element in position (1,2) with 5
- A(4,:) = [ ] - will delete 4th row
- A(:, 3) = [ ] – will delete 3rd column
- A(1,2) = [ ] – error
  - Can't delete single element in a row or column.

- A(2:2:6) = [ ]
- ans = 1 7 5 3 6 9
  - …………….. How?

Operators

+	addition
-	subtraction
*	multiplication
^	power
'	transpose
\	left division
/	right division

Operations on Matrices


```
>> B = [1 -1 3; 4 0 7]
B =
     1    -1     3
     4     0     7
>> B2 = [1 2; 5 1; 5 6];
>> B  = B + B2'               % add two matrices; why B2' is needed instead of B2?
B =
     2     4     8
     6     1    13
>> B-2                        % subtract 2 from all elements of B
ans =
     0     2     6
     4    -1    11
>> ans = B./4                 % divide all elements of the matrix B by 4
ans =
    0.5000    1.0000    2.0000
    1.5000    0.2500    3.2500
```

```
>> 4/B                        % this is not possible
??? Error using ==> /
Matrix dimensions must agree.
```

Operations on Matrices


```
>> b = [1 3 -2];
>> B = [1 -1 3; 4 0 7]
B =
     1    -1     3
     4     0     7
>> b * B                      % not possible: b is 1-by-3 and B is 2-by-3
??? Error using ==> *
Inner matrix dimensions must agree.
```

```
>> b * B'                     % this is possible: a row vector multiplied by a matrix
ans =
    -8   -10
```

```
>> C = [1 -1 4; 7 0 -1];
>> B .* C                     % multiply element-by-element
ans =
     2    -4    32
    42     0   -13
>> ans.^3 - 2                 % do for all elements: raise to the power 3 and subtract 2
ans =
         6     -66    32766
     74086      -2    -2199
```

Operations on Matrices - Summary

Command	Result
`C = A + B`	sum of two matrices
`C = A - B`	subtraction of two matrices
`C = A * B`	multiplication of two matrices
`C = A .* B`	'element-by-element' multiplication (A and B are of equal size)
`C = A ^ k`	power of a matrix ($k \in Z$; can also be used for A^{-1})
`C = A .^ k`	'element-by-element' power of a matrix
`C = A'`	the transposed of a matrix; A^T
`C = A ./ B`	'element-by-element' division (A and B are of equal size)
`X = A \ B`	finds the solution in the least squares sense to the system of equations $AX = B$
`X = B / A`	finds the solution of $XA = B$, analogous to the previous command

`A = eye(n)`	A is an $n \times n$ identity matrix
`A = zeros(n,m)`	A is an $n \times m$ matrix with zeros (default $m = n$)
`A = ones(n,m)`	A is an $n \times m$ matrix with ones (default $m = n$)
`A = diag(v)`	gives a diagonal matrix with the elements $v_1, v_2, \ldots, v_n$ on the diagonal

Matrix Addition and Subtraction

- Matrix addition and subtraction with MATLAB are achieved in the same manner as with scalars **provided** that the matrices have the same size. Typical expressions are shown below.

- >> C = A + B

- >> D = A - B

Matrix Multiplication

Matrix multiplication with MATLAB is achieved in the same manner as with scalars **provided** that the number of columns of the first matrix is equal to the number of rows of the second matrix. A typical expression is

```
>> E = A*B
```

Array Multiplication

- There is another form of multiplication of matrices in which it is desired to multiply corresponding elements in a fashion similar to that of addition and subtraction. This operation arises frequently with MATLAB, and we will hereafter refer to the process as the **array product** to distinguish it from the standard matrix multiplication form.

MATLAB Array Multiplication

- To form an array product in MATLAB, a period must be placed after the first variable. The operation is commutative. The following two operations produce the same result.

- `>> F=A.*B`

- `>> F=B.*A`

MATLAB Array Multiplication Continuation

If there are more than two matrices for which array multiplication is desired, the periods should follow all but the last one in the expression; e. g., A.*B.*C in the case of three matrices. Alternately, nesting can be used; e.g. (A.*B).*C for the case of three matrices.

MATLAB Array Multiplication Continuation

- The array multiplication concept arises in any operation in which the command could be "confused" for a standard matrix operation. For example, suppose it is desired to form a matrix B from a matrix A by raising each element of A to the 3rd power, The MATLAB command is

- `>> B = A.^3`

Multi-dimensional Array

- Arrays can have more than two dimensions
- A= [1 2; 3 4]
- You can add the 3rd dimension by
- A(:,:,2) = [5 6; 7 8]

Matrixes and vectors command summary

- *x = [1,2,3]* , vector-row,
- *y=[1;2;3]*, vector-column,
- *x=0:0.1:0.8* , vector *x=[0,0.1,0.2,0.3....0.8],*
- *A = [1,3,5;5,6,7;8,9,10]*, matrix,
- *A(1,2)*, element of matrix, 1. row, 2. column,
- *A(:,2)*, second column of matrix,
- *A(1,:)*, first row of matrix ,
- *C=[A;[10,20,30]]* matrix with additional row,
- *A(:,2)=[]*, deleting of second column,
- *B=A(2:3,1:2)*, part of matrix,
- *x'*, transpose.

Matrixes and vectors command summary …

- *size(A)*, matrix size,
- *det(A)*, determinant,
- *inv(A)*, inverse matrix,
- *eye(3)*, unit matrix,
- *zeros(3,4)*, matrix of zeros,
- *rand(3,5)*, matrix of random values,
- *sum(A)*, sum of elements,
- *A*x*, matrix-vector product (if dimensions are corresponding),
- *A.*B*, element multiplication of two matrixes.

Thanks

Questions ??