EECS 360: Signal and System Analysis

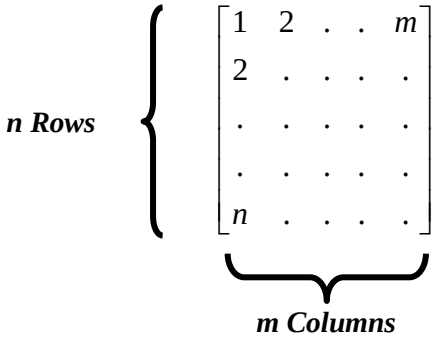
Fall 2021

Lab#2 – Matrix and Array in MATLAB

Matrices are the basic elements of the MATLAB environment. A matrix is a two-dimensional array consisting of m rows and n columns. Special cases are column vectors (n = 1) and row vectors (m = 1). **In this lab, we will illustrate how to create array and matrices and how to apply different operations on matrices.**

1. ENTERING A VECTOR

We define a matrix of dimension (n × m) as follows, where n stands for number of rows and m stands for number of columns.



A vector is a special case of a matrix, where either number of rows or number of columns goes to 1.

Row Vector: An array of dimension (1 × n) is called a row vector.
Column Vector: An array of dimension (m × 1) is called a column vector.

The elements of vectors in MATLAB are enclosed by square brackets and are separated by spaces or by commas. For example, in Matlab, you can define a Row Vector as:

A = [1 2 3 4] *or* A = [1,2,3,4].

Column vectors are created in a similar way, however, semicolon (;) must separate the components of a column vector. For example, Column Vector:

B= [1; 2; 3; 4]

******* A row vector is converted to a column vector using the transpose operator and vice versa. The transpose operation is denoted by an apostrophe or a single quote (').

$$A = B'$$

or

$$B = A'$$

PRACTICE 1: Create one row vector (1×5) and one column vector (5×1) and perform transpose operation to convert them from one type to another (i.e. row to column and vice versa).

2. ACCESSING THE ELEMENTS OF VECTOR

To access n^{th} element of a vector A, we simply can write, A(n). For example, to access the 3^{rd} element of vector A, we can write, A(3). Furthermore, to access blocks of elements, we use MATLAB's colon notation (:). For example, to access the first three elements of A, we write, A (1:3).

PRACTICE 2: Create one row vector (1×7) and one column vector (6×1) and access last three elements from each vector and store those in a two new vectors. Now add these two vectors.

3. ENTERING A MATRIX

A matrix is an array of numbers. To type a matrix into MATLAB you must:

- begin with a square bracket, [
- separate elements in a row with spaces or commas (,)
- use a semicolon (;) to separate rows
- end the matrix with another square bracket,]

For example,

$$A = [1 \quad 2 \quad 3; 4 \quad 5 \quad 6; 7 \quad 8 \quad 9]$$

or

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

4. ACCESSING THE ELEMENTS OF A MATRIX

To access (m, n) element of a matrix A, we simply can write, A(m,n) . Here, m = row and n= column. For example, A(1,3) stands for an element of 1st row and 3rd column.

Now to substitute any specific element of the matrix, we simply specify the element with our desired value. For example, if I want to substitute A(1,3) with 0, simply I can write: A(1,3)=0.

PRACTICE 3: Create one matrix of 3 rows and 4 columns with your desired numbers, access the 3^{rd} element of 2^{nd} row and 2^{nd} element of 4^{th} column. Now, substitute those values with 5.

5. COLON & LINEAR SPACING OPERATOR

Often, we need to create big array or matrix with many elements and it's time consuming to input each element one by one. On the other hand, we can use colon (:) operator to create a big array with many elements as follows:

$$A = Initial\ Value : increment : End\ Value$$

$$A = 1 : 0.5 : 5$$

$$Here, 1 = Initial\ Value; 0.5 = increment; 5 = End\ Value$$

If you do not mention the increment, MATLAB takes 1 automatically as increment value

$$A = Initial\ Value : End\ Value$$

$$A = 2 : 5$$

$$Here, 2 = Initial\ Value; 5 = End\ Value; 1 = Default\ \text{Increment};$$

Additionally, there is another command to generate linearly spaced vectors: 'linspace'. It works same as the colon operator (:) but gives direct control over the number of points. For example:

$$A = linspace(Initial\ Value, End\ Value) \Rightarrow A = linspace(1,5)$$

or

$$A = linspace(Initial\ Value, End\ Value, Number\ \text{of Points}) \Rightarrow A = linspace(1,5,10)$$

If you do not mention how many points you want, MATLAB creates default 100 linearly spaced point in between your initial and end value.

PRACTICE 4: Create one array of data interval $[0, 2\pi]$ with 100 elements. Now, substitute elements from 21 to 70 with numerical value 5.

6. COLON OPERATOR IN MATRIX

The colon operator can also be used to pick out a certain row or column. For example, the statement $A(m:n, k:l)$ specifies rows m to n and columns k to l. Subscript expressions refer to portions of a matrix. For example, suppose I have a matrix A or dimension (4×4) i.e. 4 rows and 4 columns.

a) If I want to access only 2nd row, I can write: A(2,:)
b) If I want to access only 3rd column, I can write A(:,3)
c) If I want to access columns from 1 to 3, I can write A(:,1:3)
d) If I want to access common elements among rows 2 & 3 and columns 3 & 4, write A(1: 2,3: 4)

PRACTICE 5: Create the following matrix and access the portion $\begin{matrix}10 & 11\\ 14 & 15\end{matrix}$ from the matrix.

$$\begin{bmatrix}1 & 2 & 3 & 4\\ 5 & 6 & 7 & 8\\ 9 & 10 & 11 & 12\\ 13 & 14 & 15 & 16\end{bmatrix}$$

7. CREATING SUB MATRIX & DELETING ROW/COLUMN

Suppose I have a matrix A or dimension (4×4) i.e. 4 rows and 4 columns. To extract a submatrix B consisting of rows 2 and 3 and columns 1 and 2 of the matrix A, do the following

$$B = A([2\quad 3],[1\quad 2])$$

a) To access rows 2 and 3 and all columns, B=A ([2 3],:)
b) To interchange row 2 and 3 in our new matrix, B=A ([3 2],:)
c) To delete a row or column of a matrix, use the empty vector operator, [ ]. If I want to delete 3rd row, I can write A(3,:) = [].

PRACTICE 6: Suppose I have a matrix $\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{bmatrix}$. Now create a new matrix as follows from the given matrix.

$$\begin{bmatrix}9 & 8\\ 3 & 2\end{bmatrix}$$

PRACTICE 7: Suppose I have a matrix $\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{bmatrix}$. Now create a new matrix as follows using only the given matrix.

$$\begin{bmatrix} 1 & 2 & 3 & 10 & 20 & 30 \\ 4 & 5 & 6 & 40 & 50 & 60 \\ 7 & 8 & 9 & 70 & 80 & 90 \\ -1 & -2 & -3 & 1 & 0 & 0 \\ -4 & -5 & -6 & 0 & 1 & 0 \\ -7 & -8 & -9 & 0 & 0 & 1 \end{bmatrix}$$

8. MATRIX GENERATION USING FUNCTION

MATLAB provides functions that generates elementary matrices. The matrix of zeros, the matrix of ones, and the identity matrix are returned by the function zeros, ones, and eye, respectively. Here is a list of elementary matrix function:

Function	Explanation
eye(m,n)	Returns an m-by-n matrix with 1 on the main diagonal
eye(n)	Returns an n-by-n square identity matrix
zeros(m,n)	Returns an m-by-n matrix of zeros
ones(m,n)	Returns an m-by-n matrix of ones
diag(A)	Extracts the diagonal of matrix A
rand(m,n)	Returns an m-by-n matrix of random numbers
size(A)	Returns the dimensions of a matrix or vector
exp(A)	Returns the exponential e^A for each element in array A.
expm(A)	Computes the matrix exponential of A.
diag(A)	Returns a square diagonal matrix with the elements of vector A on the main diagonal.
diag(A,k)	Places the elements of vector A on the k^{th} diagonal. k=0 represents the main diagonal, k>0 is above the main diagonal, and k<0 is below the main diagonal.
inv(A)	Computes the inverse of square matrix A.
eig(A)	Returns a column vector containing the eigenvalues of square matrix A.
[P,Q] = eig(A)	returns diagonal matrix Q of eigenvalues and matrix P whose columns are the corresponding right eigenvectors, so that A*P = P*Q.
prod(A)	Returns the product of the array elements of A.
numel(A)	returns the number of elements in array A
reshape(A,sz)	reshapes A using the size vector, sz, to define size(B). For example, reshape(A,[2,3]) reshapes A into a 2-by-3 matrix. sz must contain at least 2 elements, and prod(sz) must be the same as numel(A).

Table: 2.1

PRACTICE 8: Run eye(4,3); eye(3); ones(3,1); zeros(2;3); diag(eye(4,3)); size(eye(4,3)) commands and try to understand what is going on in each command.

PRACTICE 9: Create a random (8×2) matrix using 'rand' command. Then reshape this matrix into a (4×4) matrix. After that extract the diagonal elements from that new (4×4) matrix.