Al-Mustaqbal University / College of Engineering & Technology
Department (Chemical and Petrochemical engineering)
Class (second stage)
Subject (Computer Programming)
Lecturer (Abrar Falah Naji)
2nd term – Lecture No.2 & Lecture Name (*One-dimensional array (vector)*)

Vectors

An **array** is a collection of numbers, called **elements**, referenced by one or more indices running over different index sets. In MATLAB, the index sets are always sequential integers starting with 1. The **dimension** of the array is the number of indices needed to specify an element. The **size** of an array is a list of the sizes of the index sets.

A **matrix** is a two-dimensional array with special rules for addition, multiplication, and other operations. The two dimensions are called the **rows** and the **columns**.

A **vector** is a matrix in which one dimension has only the index 1. A **row vector** has only one row and a **column vector** has only one column.

1. Entering Vectors and Matrices

There are several ways to enter vectors and matrices in Matlab. These include:

1- Entering an explicit list of elements.
2- Loading matrices from external data files.
3- Generating matrices using functions.

To enter a vector or matrix explicitly, there are a few basic rules to follow:

• Separate the elements of a row with spaces or commas.
• Use a semicolon ; or returns, to indicate the end of each row.
• Surround the entire list of elements with square brackets, [ ].

For example, to input a 3×1 vector:

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

The entries must be enclosed in square brackets.

```
>> v =[ 1 3  sqrt(5)]
v =
   1.0000 3.0000 2.2361
```

Spaces is very important:

```
>> v2 =[3+ 4 5]
v2 =
    7     5
>> v3 =[3 +4 5]
v3 =
    3     4     5
```

Al-Mustaqbal University / College of Engineering & Technology
Department (Chemical and Petrochemical engineering)
Class (second stage)
Subject (Computer Programming**)**
Lecturer (Abrar Falah Naji)
2nd term – Lecture No.2 & Lecture Name (*One-dimensional array (vector)*)

We can do certain arithmetic operations with vectors of the same length, such as v and v3 in the previous section.

```
>> v + v3 ans =
    4.0000 7.0000 7.2361
>> v4 =3*v
v4 =
    3.0000 9.0000 6.7082
>> v5 =2*v -3*v3
v5 =
   -7.0000 -6.0000 -10.5279
```

We can build row vectors from existing ones:

```
>> w =[1 2 3], z = [8 9]
w =
    1    2    3
z =
    8    9
>> v6=[w z]
v6 =
    1    2    3    8    9
```

A vector can be defined by previously defined vectors.

```
>> x = [2 4 -1]
x =
    2    4   -1
>> x1 = [x  5 8 ]
x1 =
    2    4   -1    5    8
```

An special array is the **empty matrix**, which is entered as [] .and can be used to delete a part of any matrix or vector.

```
>> w=[4 5 6 7]
w =
    4    5    6    7
```

Al-Mustaqbal University / College of Engineering & Technology
Department (Chemical and Petrochemical engineering)
Class (second stage)
Subject (Computer Programming)
Lecturer (Abrar Falah Naji)
2nd term – Lecture No.2 & Lecture Name (*One-dimensional array (vector)*

```
>>w(4)=[];w
w =
    4     5     6
```

The elements of a matrix can be defined with algebraic expressions placed at the appropriate location of the element. Thus

```
>> a = [ sin(pi/2) sqrt(2) 3+4 6/3 exp(2) ]
a=
    1.0000    1.4142    7.0000    2.0000    7.3891
```

Column Vectors

The column vectors have similar constructs to row vectors. When defining them, entries are separated by ; or "newlines".

Note how the column vector is defined, using brackets and semicolons to separate the different rows. To define a column vector x:

```
x=[1; -2; 4]
x =
1
-2
4
```

or write

```
>>x=[1
2
3]
x =
1
-2
4
>> c =[ 1; 3; sqrt(5)]
c =
1.0000
3.0000
2.2361
```

Al-Mustaqbal University / College of Engineering & Technology
Department (Chemical and Petrochemical engineering)
Class (second stage)
Subject (Computer Programming)
Lecturer (Abrar Falah Naji)
2nd term – Lecture No.2 & Lecture Name (*One-dimensional array (vector)*

Transposing

We can convert a row vector into a column vector (and vice versa) by a process called *transposing*, denoted by **'**

```
> A=[ 1 2 3]
A =
    1    2    3
>>B= A'
B =
    1
    2
    3
```

Vectors Addition and subtraction

Addition and subtraction of a number to or from a vector can be made. In this case, the number is added to or subtracted from all the elements of the vector. For example

```
>> x=[-1; 0; 2];
>> y=x-1
y =
   -2
   -1
    1
```

If we look to make a simple addition and subtraction of vectors. The notation is the same as found in most linear algebra. We will define two vectors then add or subtract them:

```
>> v = [1; 2; 3]
v =
    1
    2
    3
>> b = [2; 4; 6]
b =
    2
    4
    6
```

```
>> v+b
ans =
     3
     6
     9
>> v-b
ans =
    -1
    -2
    -3
>> sin(v)
ans =
    0.8415
    0.9093
    0.1411
>> log(v)
ans =
         0
    0.6931
    1.0986
>> pi*v
ans =
    3.1416
    6.2832
    9.4248
```

Vectors multiplication

Multiplication of vectors and matrices must follow special rules. In the example above, the vectors are both column vectors with three entries. You cannot add a row vector to a column vector. In the case of multiplication vectors, the number of columns of the vector on the left must be equal to the number of rows of the vector on the right.

```
>> b = [2; 4; 6];
>> v = [1; 2; 3];
>> v*b
??? Error using ==> *
Inner matrix dimensions must agree.
>> v*b'
```

Al-Mustaqbal University / College of Engineering & Technology
Department (Chemical and Petrochemical engineering)
Class (second stage)
Subject (Computer Programming)
Lecturer (Abrar Falah Naji)


```
ans =
     2     4     6
     4     8    12
     6    12    18
>> v'*b
ans =
    28
```

2. element-wise operation

There are many times where we want to do an operation to every entry in a vector or matrix. Matlab will allow you to do this with "element-wise" operations. For example, suppose you want to multiply each entry in vector v with its corresponding entry in vector b. In other words, suppose you want to find v(1)*b(1), v(2)*b(2), and v(3)*b(3). It would be nice to use the "*" symbol since you are doing some sort of multiplication, but since it already has a definition, we have to come up with something else. The programmers who came up with Matlab decided to use the symbols ".*" to do this.

```
>> v.*b
ans =
     2
     8
    18
```

Also for division we must use "./"

```
>> v./b
ans =
    0.5000
    0.5000
    0.5000
```

Note that

v .* b multiplies each element of v by the respective element of b.

v ./ b divides each element of v by the respective element of b.

v .\ b divides each element of b by the respective element of v.

v .^ b raise each element of v by the respective b element.

Al-Mustaqbal University / College of Engineering & Technology
Department (Chemical and Petrochemical engineering)
Class (second stage)
Subject (Computer Programming)
Lecturer (Abrar Falah Naji)
2nd term – Lecture No.2 & Lecture Name (*One-dimensional array (vector)*)

The Colon Operator

The colon operator ' : ' is understood by 'matlab' to perform special and useful operations. If two integer numbers are separated by a colon, 'matlab' will generate all of the integers between these two integers.

In particular, you will be able to use it to extract or manipulate elements of matrices. The following command creates a row vector whose components increase arithmetically:

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

And, if three numbers, integer or non-integer, are separated by two colons, the middle number is interpreted to be a "step" and the first and third are interpreted to be "limits". Thus

```
>> b = 0.0 : 0.2 : 1.0
b =
      0    0.2000    0.4000    0.6000    0.8000    1.0000
```

Suppose you want to create a vector with elements between 0 and 20 evenly spaced in increments of 2. Then you have to type:

```
>> t = 0:2:20
t =
     0     2     4     6     8    10    12    14    16    18    20
>> m=0.32:0.1:0.6
m =
    0.3200    0.4200    0.5200
>>w= -1.4:-0.3:-2
w =
   -1.4000  -1.7000   -2.0000
```

The format is **first:step:last**. The result is always a row vector.

A negative step is also allowed. The command has similar results; it creates a vector with linearly spaced entries.

There is another way to create row arrays is to use linspace functions:

```
>> A=linspace(1,2,5)
A =
    1.0000    1.2500    1.5000    1.7500    2.0000
>> A=linspace(0,20,11)
A =
     0  24     6     8    10  12    14    16    18    20
```

Al-Mustaqbal University / College of Engineering & Technology
Department (Chemical and Petrochemical engineering)
Class (second stage)
Subject (Computer Programming)
Lecturer (Abrar Falah Naji)
2nd term – Lecture No.2 & Lecture Name (*One-dimensional array (vector)*

Referencing elements

It is frequently necessary to call one or more of the elements of a vector. Each dimension is given a single index. Some examples using the definitions above:

Evaluate a vector A:

```
>> A=0:10:100
A =
    0    10    20    30    40    50    60    70    80    90   100
```

Now after definition of a vector A, try to type:

```
>> A(10)
ans =
   90
>> B=A(1:5)
B =
    0    10    20    30    40
>> C=A(1:2:10)
C =
    0    20    40    60    80

>> A(6:2:10)
ans =
   50    70    90
```

Al-Mustaqbal University / College of Engineering & Technology
Department (Chemical and Petrochemical engineering)
Class (second stage)
Subject (Computer Programming**)**
Lecturer (Abrar Falah Naji)
2nd term – Lecture No.2 & Lecture Name (*One-dimensional array (vector)*)

Exercise 1:

Calculate Reynold Number at a diameter D=0.2 m, using different velocity's as u=0.1,0.2,0.3 …1 m/s knowing that:

Re= (ρud)/μ , μ=0.001 , ρ =1000

Solution:

Type the code in the command window

```
d=0.2;
u=0.1:.1:1;
m=0.001;
p=1000;
Re=u*p*d/m
```

The results will be

```
Re =
1.0e+005 *
0.2000  0.4000  0.6000  0.8000  1.0000  1.2000  1.4000  1.6000  1.8000  2.0000
```

Exercise 2:

Estimate the average density of a Water/Ethanol mixture at different water compositions knowing that.

Water density =1000 kg/m^3

Ethanol density =780 kg/m^3

Mixture density= Xwater × Water density + Xethanol × Ethanol density

Solution:

```
Pwater=1000; Pethanol=780;
Xwater=0:.1:1
Xethanol=1-Xwater;
Pav=Pwater*Xwater+Pethanol*Xethanol
Xwater =
         0    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000    0.7000    0.8000    0.9000    1.0000
Pav =
   780   802   824   846   868   890   912   934   956   978  1000
```