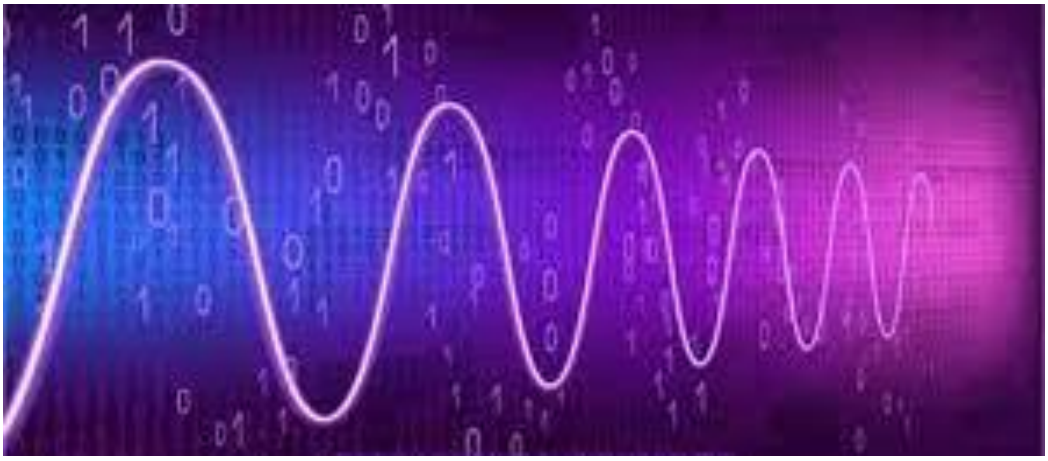
# DIGITAL SIGNAL PROCESSING LAB MANUAL 4

Dr. Ahsan Latif, Ms. Anosh Fatima



WINTER SEMESTER 2022

UNIVERSITY OF AGRICULTURE, FAISALABAD (UAF)

# MATLAB-Matrix and Array Operations

```
a = [1 2 3; 4 5 6; 7 8 10]
```

```
a = 3×3

     1     2     3
     4     5     6
     7     8    10
```

---

## Matrix and Array Operations

MATLAB allows you to process all of the values in a matrix using a single arithmetic operator or function.

```
a + 10
```

```
ans = 3×3

    11    12    13
    14    15    16
    17    18    20
```

```
sin(a)
```

```
ans = 3×3

    0.8415    0.9093    0.1411
   -0.7568   -0.9589   -0.2794
    0.6570    0.9894   -0.5440
```

To transpose a matrix, use a single quote (').

```
a'
```

```
ans = 3×3

     1     4     7
     2     5     8
     3     6    10
```

## TASK

- Open New Script from Toolbar
- Write code in Editor Window
- Click Run from Tool Bar to see output in command window.
- Save all MATLAB Files in Separate folder called DSP Labs

- Write name of file: your group number and lab number. E.g., G2Lab3
- Click add to the path
- Check results

You can perform standard matrix multiplication, which computes the inner products between rows and columns, using the * operator. For example, confirm that a matrix times its inverse returns the identity matrix:

```
p = a*inv(a)
```

```
p = 3×3

    1.0000         0   -0.0000
         0    1.0000         0
         0         0    1.0000
```

Notice that `p` is not a matrix of integer values. MATLAB stores numbers as floating-point values, and arithmetic operations are sensitive to small differences between the actual value and its floating-point representation. You can display more decimal digits using the `format` command:

```
format long
p = a*inv(a)
```

```
p = 3×3

   1.000000000000000                   0  -0.000000000000000
                   0   1.000000000000000                   0
                   0                   0   0.999999999999998
```

Reset the display to the shorter format using

```
format short
```

---

`format` affects only the display of numbers, not the way MATLAB computes or saves them.

To perform element-wise multiplication rather than matrix multiplication, use the `.*` operator:

```
p = a.*a
```

```
p = 3×3

     1     4     9
    16    25    36
    49    64   100
```

The matrix operators for multiplication, division, and power each have a corresponding array operator that operates element-wise. For example, raise each element of `a` to the third power:

```
a.^3
```

```
ans = 3×3

           1           8          27
          64         125         216
         343         512        1000
```

## Concatenation

*Concatenation* is the process of joining arrays to make larger ones. In fact, you made your first array by concatenating its individual elements. The pair of square brackets [] is the concatenation operator.

```
A = [a,a]
```

```
A = 3×6

     1     2     3     1     2     3
     4     5     6     4     5     6
     7     8    10     7     8    10
```

Concatenating arrays next to one another using commas is called *horizontal* concatenation. Each array must have the same number of rows. Similarly, when the arrays have the same number of columns, you can concatenate *vertically* using semicolons.

```
A = [a; a]
```

```
A = 6×3

     1     2     3
     4     5     6
     7     8    10
     1     2     3
     4     5     6
     7     8    10
```

## Complex Numbers

Complex numbers have both real and imaginary parts, where the imaginary unit is the square root of -1.

```
sqrt(-1)
```

```
ans = 0.0000 + 1.0000i
```

To represent the imaginary part of complex numbers, use either `i` or `j`.

```
c = [3+4i, 4+3j; -i, 10j]
```

```
c = 2×2 complex

   3.0000 + 4.0000i   4.0000 + 3.0000i
   0.0000 - 1.0000i   0.0000 +10.0000i
```

**Solve following Exercise 1 and 2 in two separate scripts in MATLAB.**

## Exercise 1

1. Create any matrix, called X of size 4*4 and display it, then add 5 in its all values.
2. Take transpose of original matrix X.
3. Take power of 2, element wise, of original Matrix X.
4. Take power of 3, element wise, of original Matrix X.
5. Create matrix A and B of size 3*3 and display their sum.
6. Create matrix A and B of size 3*3 and display their product.
7. Concatenate original Matrix X with itself and display.
8. Create matrix A and B of size 3*3 and concatenate them horizontally and display.
9. Create matrix A and B of size 3*3 and concatenate them vertically and display.

## Exercise 2

1. Create any matrix, with non-zero elements, called W of size 3*3 and display it, then add 3 in its all values.
2. Take transpose of original matrix W.
3. Take power of 2, element wise, of original Matrix W.
4. Take power of 3, element wise, of original Matrix W.
5. Create matrix C and D of size 2*2 and display their sum.
6. Create matrix X and Y of size 2*2 and display their product.
7. Concatenate original Matrix W with matrix C, D, X and Y one by one and display all new concatenated matrices named as WC, WD, WX, WY respectively.
8. Create matrix A and B of size 5*5 and concatenate them horizontally and display.
9. Create matrix A and B of size 2*2 and concatenate them vertically and display.