Fall 2021

EECS 360: Signal and System Analysis

Lab#3 – Array Operations, Basic Functions and Plotting in MATLAB

MATLAB has two different types of arithmetic operations: matrix arithmetic operations and array arithmetic operations.

1. MATRIX ARITHMETIC OPERATIONS

MATLAB allows arithmetic operations: $+, -, *, \wedge$ to be carried out on matrices. However, remember few things:

Operations	Explanation
A+B	is valid if A and B are of same size.
A*B	is valid if A's number of columns equals B's number of rows
A^2	is valid if A is a square Matrix
Constant*A	multiplies each element of A by a constant

Table 1

PRACTICE 1: I have $A=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix}; B=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix}; C=\begin{bmatrix}1 & 2\\ 3 & 4\\ 5 & 6\end{bmatrix}; D=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$ four matrices.

Perform the following operations and find out which operation works and which one doesn't. Also try to find out the reason for not performing that operation.

$(a) A+B$

$(b) A+D$

$(c) A \wedge 2$

$(d) D \wedge 2$

$(e) A*B$

$(f) A*C$

2. ARRAY ARITHMETIC OPERATIONS

Array arithmetic operations or array operations for short, are done element-by-element. However, since the matrix and array operations are the same for addition $(+)$ and subtraction $(-)$, the character pairs $(.+)$ and $(.-)$ are not used.

Operation	Explanation
.*	Element – by – element multiplication
./	Element – by – element division
.^	Element – by – element exponentiation

Table 2

PRACTICE 2: I have $A=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}, B=\begin{bmatrix}10 & 20\\ 30 & 40\end{bmatrix}; \& C=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{bmatrix}$ three matrices. Perform the following operations and find out which operation works and which one doesn't. Also try to find out the reason for not performing that operation and why similar kind of operations are giving different results.

$(a) A * B$

$(b) A.* D$

$(c) A \wedge 2$

$(d) A \wedge 2$

$(e) A * C$

$(f) A.* C$

3. MATRIX FUNCTIONS

MATLAB provides many matrix functions for various matrix/vector manipulations. Table 3 shows some of these functions.

Functions	Explanation
det	Finds the determinant of a matrix
diag	Finds the diagonal elements of a matrix
eig	Finds the eigenvalues of a matrix
inv	Performs matrix inverse operation
rank	Finds the rank (rank: linearly independent rows and columns) of a matrix

4. MATLAB PLOTTING

The basic MATLAB graphing procedure, for example in 2D, is to take

a) a vector of x- coordinates, $x = x_1, x_2, ..., x_n$ and a vector of y-coordinates, $y = y_1, y_2, ..., y_n$, locate the points (x_i, y_i), with $i = 1, 2, ..., n$ and then join them by straight lines.

b) You need to prepare x and y in an identical array form; namely, x and y are both row arrays or column arrays of the same length.

Example of a simple Graph Matlab Code

```
TIME=0:0.01:5; %I am defining my X - Axis%
FREQUENCY=0.5; % Signal Frequency%
AMPLITUDE=5; %Signal Amplitude%
FUNCTION=AMPLITUDE*sin(2*pi*FREQUENCY*TIME); % Sin Function that I am going to plot. It's actually my Y - axis. %
plot(TIME,FUNCTION) % Finally Plotting. plot is the basic function command in MATLAB to plot something%
```

After running the code, you will get something like Figure 01.

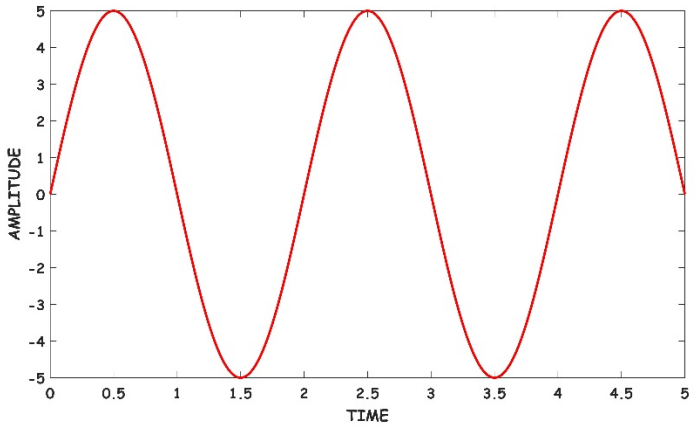


Figure 01

5. ADDING TITLES, AXIS LABELS & ANNOTATIONS

MATLAB enables you to add axis labels and title. You can use following few functions to add those features in your graph:

Functions	Explanation
xlabel	labels the X-axis of the current axes
ylabel	labels the Y-axis of the current axes
title	adds the specified title to the axes
legend	creates a legend for the current plot
axis	specifies the limits for the current axes
grid on	displays the major grid lines for the current axes
grid off	removes all grid lines from the current axes
hold on	retains plots in the current axes so that new plots added to the axes do not delete existing plots.
hold off	sets the hold state to off so that new plots added to the axes clear existing plots and reset all axes properties.

Now I am writing a sample MATLAB code using these functions. (See Figure 02 for output)

```
Time=0:0.01:5;
Frequency=0.5;
Amplitude1=5;
Amplitude2=3;
Function1=Amplitude1*sin(2*pi*Frequency*Time);%1st function that I am going to plot%
plot(Time,Function1) %Plotting my 1st Function%
grid on; % To show major grid lines in the plot%
hold on; % To hold my current plot i.e. Function 1%
Function2=Amplitude2*sin(2*pi*Frequency*Time); %2nd function that I am going to plot%
plot(Time,Function2) %Plotting my 2nd Function%
xlabel("Time in Seconds"); %Labeling the X - Axis%
ylabel("Signal Amplitude"); %Labeling the Y - Axis%
title("Sample Matlab Code for EECS360 Lab#3"); % Creating a Title for the Graph%
legend('5sin(2*pi*ft)','3cos(2*pi*ft)'); % To show what functions I have plotted and what are
those%
axis([0 5 -5 5]); % Setting up the X and Y axes for my whole plot%
```

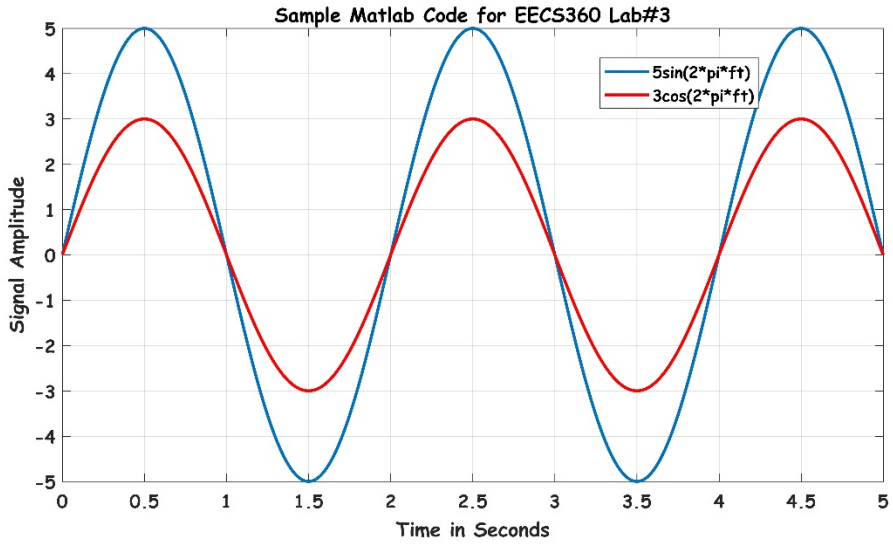


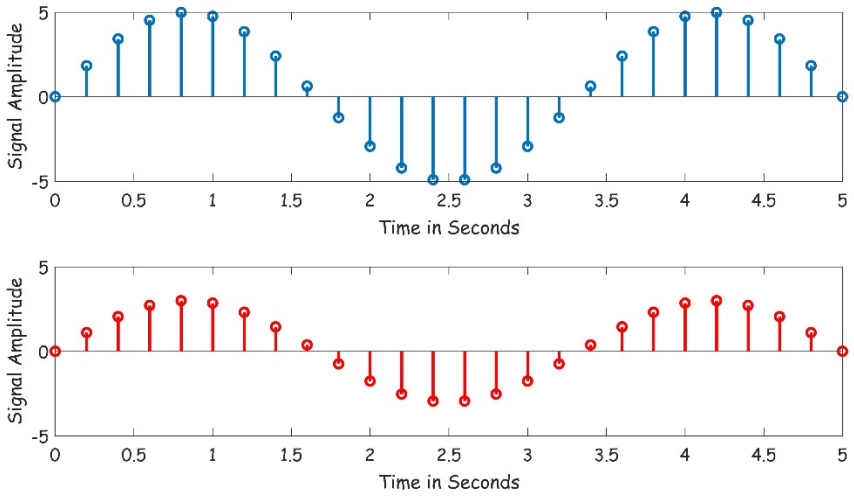
Figure 02

5. FUNCTIONS: stem, figure, subplot

Functions	Explanation
stem	plots the function points discretely
figure	creates multiple windows for figure. We use it if we want to see each plot in a different graph output window.
subplot	creates multiples plots in a single graph window.

For example, I used stem and subplot in the following code. (See Figure 03 for output)

```
Time=0:0.2:5;
Frequency=.3;
Amplitude1=5;
Function1=Amplitude1*sin(2*pi*Frequency*Time);%1st function that I am going to plot%
subplot(2,1,1)
stem(Time,Function1) %Plotting my 1st Function%
xlabel("Time in Seconds"); %Labeling the X - Axis%
ylabel("Signal Amplitude"); %Labeling the Y - Axis%
Function2=Amplitude2*sin(2*pi*Frequency*Time); %2nd function that I am going to plot%
subplot(2,1,2)
stem(Time,Function2) %Plotting my 2nd Function%
xlabel("Time in Seconds"); %Labeling the X - Axis%
ylabel("Signal Amplitude"); %Labeling the Y - Axis%
```



PRACTICE 3: I have two signals of frequency $f_1 = 3(Hz),\ f_2 = 1(Hz)$ and each signal has amplitude of $A = 5.$ Now, you have to write a code to plot:

a) Signal 1
b) Signal 2
c) Signal 1 + Signal 2
d) Signal 1 – Signal 2

You have to show me all four signals separately in a single graph window. (Tips: you have to use subplot command). Additionally, you have to label the axes in each figure accurately.

5. BASIC SIGNAL GENERATION AND PLOTTING IN MATLAB

UNIT STEP PULSE

We can represent a Unit Step Signal as follows:

$$u(t-T) = \begin{cases} 0; & t < T \\ 1; & t > T \end{cases}$$

Here, T could be 0 or either positive or negative

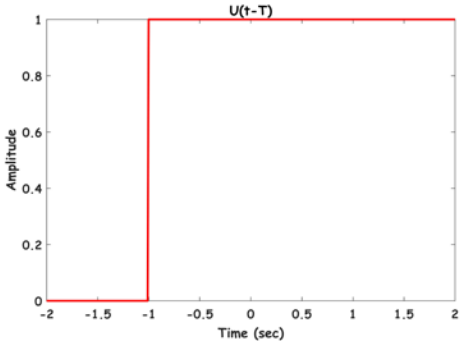
In MATLAB we simply can generate Unit Step Pulse by writing a single line:

unitstep = t >= T

Remember : Here, T is 0 or either a positive or negative numeric value.

SAMPLE MATLAB CODE

```
t=-2:0.01:2; %generating time scale, My X - axis in the plot)%
unitstep1=t>=-1; % U(t-T); where T=-1, i.e. I am plotting U(t+1)%
plot(t,unitstep1) %plotting my Unit step Signal%
```



RECTANGULAR PULSE

We can represent a Rectangular Pulse as follows:

$$rect\left(\frac{t-T}{\tau}\right)=\begin{cases}0; & t<\left(T-\frac{\tau}{2}\right)\\ 1; & \left(T-\frac{\tau}{2}\right)<t<\left(T+\frac{\tau}{2}\right)\\ 0; & t>\left(T+\frac{\tau}{2}\right)\end{cases}$$

$$Here, T\ could\ be\ 0\ or\ either\ positive\ or\ negative;$$

$$\tau = Positive$$

In MATLAB we simply can generate rectangular pulse by writing a single line:

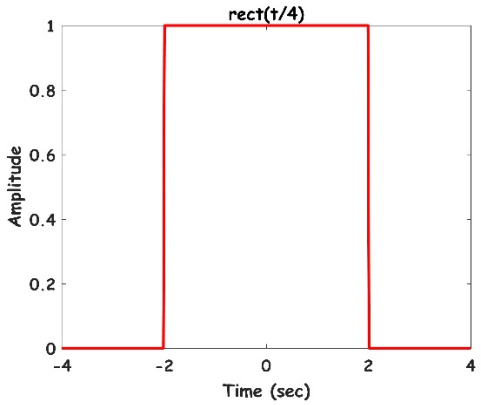
$$rectangularPulse\left(\left(T-\frac{\tau}{2}\right),\left(T+\frac{\tau}{2}\right),t\right)$$

$$Remember: Here, \left(T\pm\frac{\tau}{2}\right)\ is\ just\ a\ positive\ or\ negative\ \text{numeric value.}$$

For example, I want to plot something like: $rect\left(\frac{t}{4}\right)$. I can rewrite the signal as follows: $rect\left(\frac{t-0}{4}\right)$. So, my $T=0, \tau=4.$ In MATLAB I only need to define where my rectangular pulse starts and where it finishes. That means, I need to point out $\left(T-\frac{\tau}{2}\right)=\left(0-\frac{4}{2}\right)=-2$ and $\left(T+\frac{\tau}{2}\right)=\left(0+\frac{4}{2}\right)=2$ in my MATLAB code.

SAMPLE MATLAB CODE

```
t=-4:0.01:4; %generating time scale, My X - axis in the plot%
A=rectangularPulse(-2,2,t); % rect(t/4); where T=0, & tau=4. i.e. I am plotting rect(t/4)%
plot(t,A);
```



IMPULSE SIGNAL

We can represent an impulse signal as follows:

$$\delta(t-T)=\begin{cases} 0; & t\neq T \\ \int_{-\infty}^{\infty}\delta(t-T)=1; & t=T \end{cases}$$

Here, T could be 0 or *either positive or negative;*

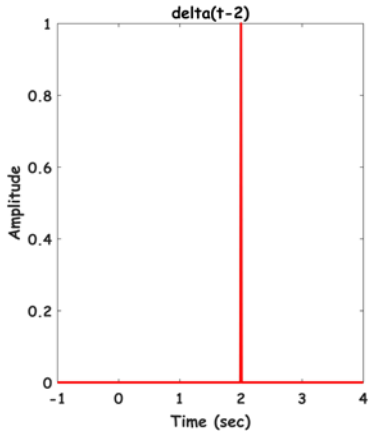
In MATLAB we simply can generate rectangular pulse by writing a single line:

impulse $= t == T$

Re*member* : *Here, T is* 0 or either *a positive or negative* numeric value.

SAMPLE MATLAB CODE

```
t=-1:0.01:4; %generating time scale, My X - axis in the plot%
impulse = t==2; % My impulse Signal is delta(t-2)%
plot(t,impulse) %plotting my impulse signal%
```



LAB ASSIGNMENT

1. Write a MATLAB code

   a) To generate $x(t) = [2 \times rect(\frac{t}{2}) - 4 \times rect(\frac{t+1}{4})] \times u(t+2)$

   b) Plot $2 \times rect(\frac{t}{2})$, $4 \times rect(\frac{t+1}{4})$ and $x(t)$ at the same graph window using subplot command.

   c) Calculate the energy of $x(t)$ without using any integration, i.e. calculate the energy from the graph.

2. Write a MATLAB code

   a) To generate $x(t) = [rect(\frac{t-2}{3}) - 4 \times rect(\frac{t-2}{4})]$

   b) Plot $rect(\frac{t-2}{3})$, $4 \times rect(\frac{t-2}{4})$ and $x(t)$ at the same graph window using subplot command.

   c) Calculate the power of $x(t)$ with the fundamental period of $x(t)$ is $T_0 = 6$ without using any integration, i.e. calculate the energy from the graph.