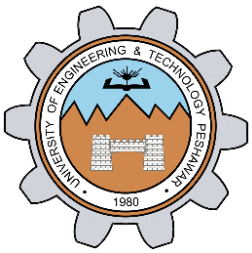
INTRODUCTION TO SIGNALS IN MATLAB

LAB # 04



Spring 2023

CSE301L Signals & Systems Lab

Submitted by: **Ali Asghar**

Registration No. : **21PWCSE2059**

Class Section: **C**

"On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work."

Submitted to:

Engr. Sumayyea Salahuddin

Date:

March 24, 2023

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Lab Objective(s):

Objectives of this Lab are.

- Discrete Signal representation in Matlab
- Matlab Graphics
- Two Dimensional Plots
- Plot and subplot
- Different Plotting Functions Used in Matlab

Task # 01:

Given the signals:

x1[n] = [2 5 8 4 3]

x2[n] = [4 3 2]

a) Write a Matlab program that adds these two signals. Use vector addition and multiplication.

b) Instead of using vector addition and multiplication, use for loop to add and multiply the signals.

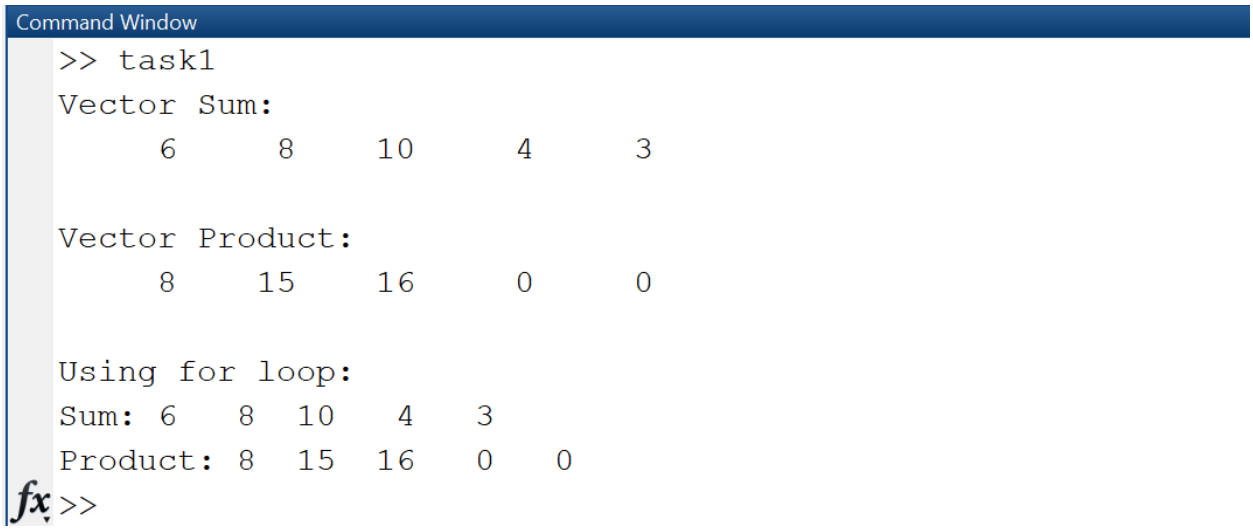
Problem Analysis:

In MATLAB, working with signals is relatively simple. Here, we demonstrate how to work with signals in MATLAB.

Algorithm:

- Write given signals.
- Add them and store the result in addition.
- Multiply them and store the result in product.
- Display addition and product.
- Then using for loop, add and multiply given signals and store them in S & P respectively.
- Display S and P.

Output / Graphs / Plots / Results:



Discussion and Conclusion:

As a result, MATLAB makes investigating signals relatively simple.

Task # 02:

Amplitude scaling by a factor β causes each sample to get multiplied by β. Write a user-defined function that has two input arguments: (i) a signal to be scaled and (ii) scaling factor β. The function should return the scaled output to the calling program. In the calling program, get the discrete time signal as well as the scaling factor from user and then call the above-mentioned function.

Problem Analysis:

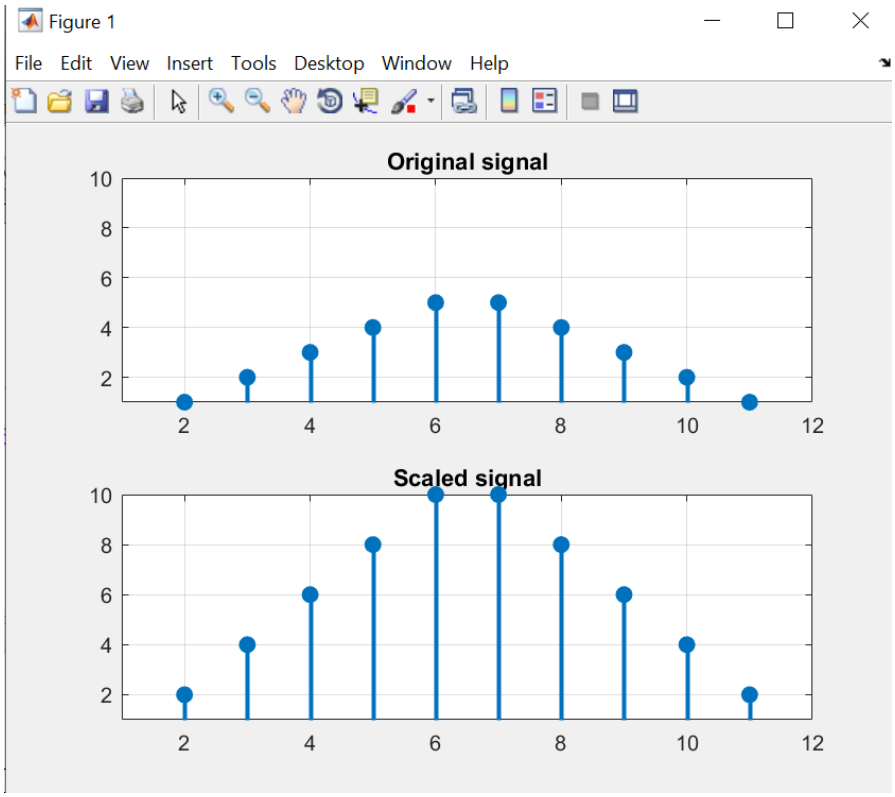
We demonstrate how MATLAB's amplification procedures on signals work.

Algorithm:

- Make the function ampl_scaling and make signal amplification logic in it.
- Then take the signal as input from the user.
- Pass the signal to ampl_scaling function and store the returned signal in A.
- Display the resultant signal A.

Output / Graphs / Plots / Results:

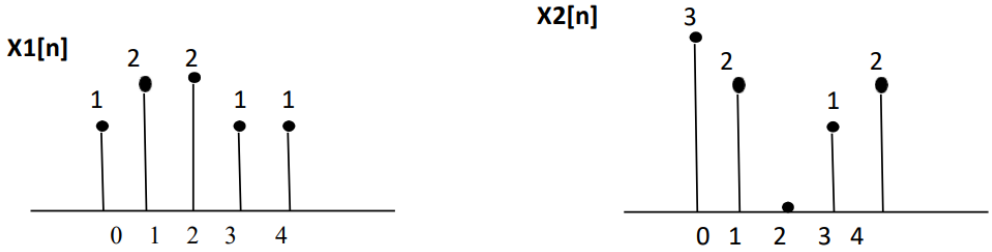
```
Command Window
  Enter value of beta factor
  2
  Enter signal
  [0 1 2 3 4 5 5 4 3 2 1 0]
  The scaled output is :0   2   4   6   8   10   10   8   6   4   2   0
fx >>
```



Discussion and Conclusion:

Matlab allows for the amplification of signals.

Task # 03:



Write a MATLAB program to compare the signals x1[n] and x2[n]. Determine the index where a sample of x1[n] has smaller amplitude as compared to the corresponding sample of x2[n]. Use for loop.

Problem Analysis:

Signal comparison is demonstrated using MATLAB.

Algorithm:

- Write given signals.
- Create an empty vector index for storing the index values
- Compare the given two signals and store the indices in index vector using for loop
- Check if the index vector is empty and then display “All samples in x1 have larger amplitude than the corresponding samples in x2”.
- If the index vector is not empty, then display the indices where x1 has smaller amplitude than x2.

Output / Graphs / Plots / Results:

```
>> task3
The index where x1 has smaller amplitude than x2 is 1  5.
>>
```

Discussion and Conclusion:

Two signals were effectively compared by us in MATLAB.

Task # 04:

Plot the two curves y1 = 2x + 3 and y2 = 4x + 3 on the same graph using different plot styles.

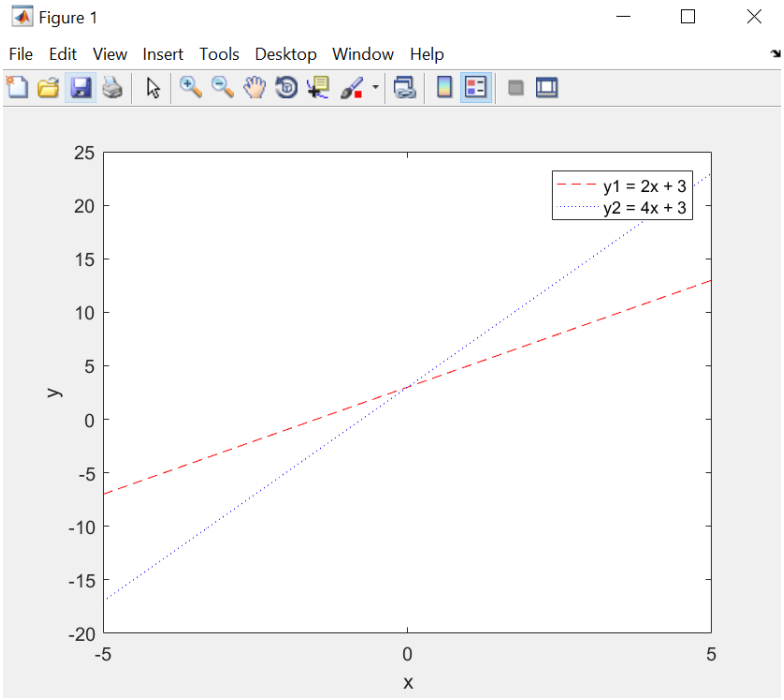
Problem Analysis:

We can use MATLAB to plot signal curves in different plotting styles.

Algorithm:

- Write the two curves.
- Plot them using different types of plotting methods.

Output / Graphs / Plots / Results:



Discussion and Conclusion:

MATLAB can plot signals in various methods.

Task # 05:

Make two separate functions for signal addition and multiplication. The functions should take the signals as input arguments and return the resultant signal. In the main program, get the signals from user, call the functions for signal addition and multiplication, and plot the original signals as well as the resultant signals.

Problem Analysis:

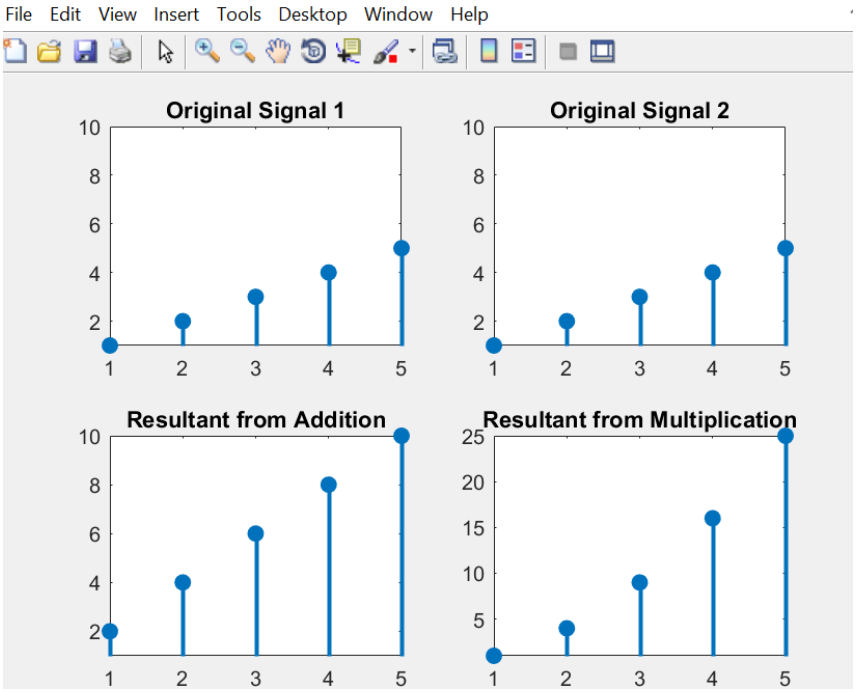
Create function for signal addition and multiplication that take signals as input and return the resultant signal. Use user input to plot the original signals and the resultant signals.

Algorithm:

- Make two functions signal_addition and signal_multiplication to add and multiply two signals.
- Then take two signals S1 and S2 from user such that length of S1 is equal to length of S2(using while loop).
- Then add and multiply these two signals using above user-defined functions.

Output / Graphs / Plots / Results:

```
>> task5
Enter the first signal: [1 2 3 4 5]
Enter the second signal: [1 2 3 4 5]
>>
```



Discussion and Conclusion:

We can use MATLAB to perform certain operations on signals and then analyze it through graphs.

Task # 06:

Given the signals:

$$X1[n] = 2\delta[n] + 5\delta[n-1] + 8\delta[n-2] + 4\delta[n-3] + 3\delta[n-4]$$
$$X2[n] = \delta[n-4] + 4\delta[n-5] + 3\delta[n-6] + 2\delta[n-7]$$

Write a Matlab program that adds these two signals. Plot the original signals as well as the final result.

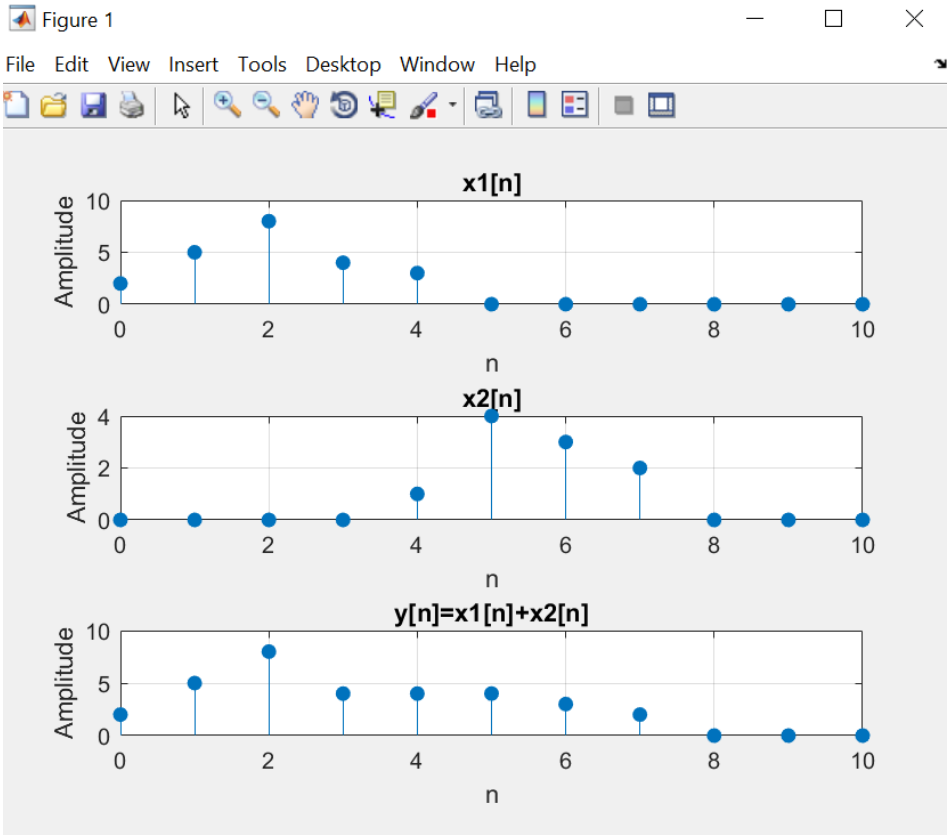
Problem Analysis:

We use MATLAB to add given two signals.

Algorithm:

- Write the two signals.
- Add the signals.
- Plot the signals.

Output / Graphs / Plots / Results:



Discussion and Conclusion:

The two signals were successfully combined using the MATLAB Built-In Add Function.

Task # 07:

Take a discrete-time signal from user. Count the number of samples with amplitude greater than a threshold of 3 and less than a threshold of -3 (use for loop).

Problem Analysis:

We frequently need to count the signal's threshold samples. So we use MATLAB to carry out that.

Algorithm:

- Take the signal as input from the user.
- Initialize count variable to zero to for counting number of samples.
- Loop through each sample in the signal and increment count variable by 1 whenever the if condition is true.
- Display the value of count.

Output / Graphs / Plots / Results:

```
Command Window
>> task7
Enter a discrete-time signal: [1 2 3 4 5 -2 -8 -10]
The number of samples with amplitude greater than 3 and less than -3 is:4
```

Discussion and Conclusion:

The threshold is put up against a function in MATLAB.

Task # 08:

Write your own function to downsample a signal i.e. retains odd numbered samples of the original signal and discard the even-numbered (downsampling by 2). The function must take a signal as input and return the downsampeled version of that signal. See Figure 4.7 for example. Call this function from a Matlab file. Verify your result by using the command "downsample". Plot the original signal, downsampled signal determined by your program, and downsampled signal obtained by the command downsample

Problem Analysis:

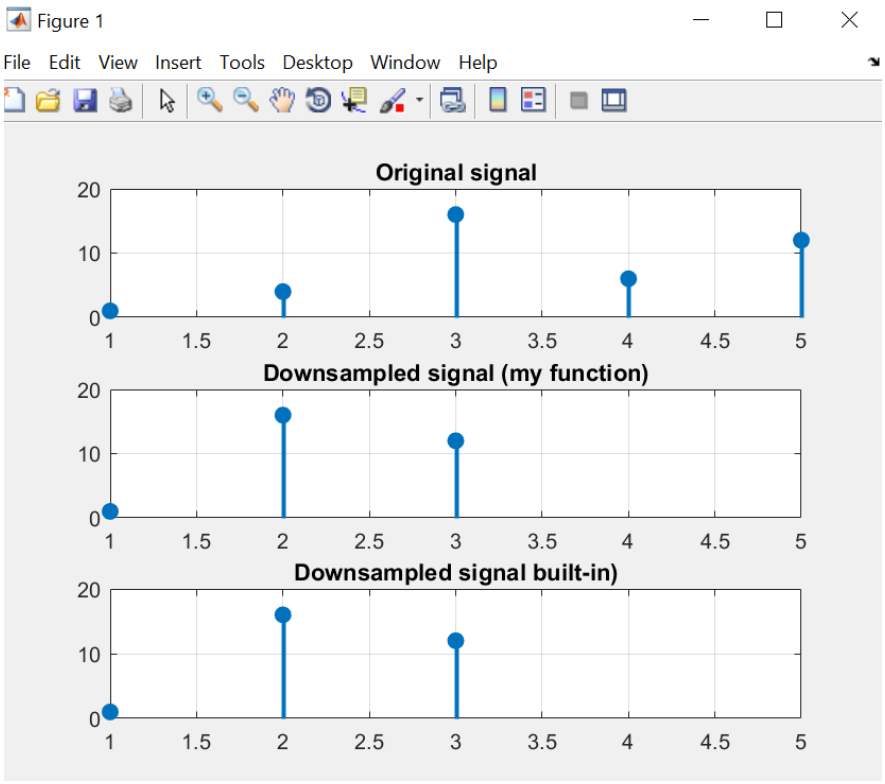
Signal downsampling is a crucial procedure in Signal Processing. We examine how to carry out it in MATLAB.

Algorithm:

- Make a function myDownsample and make logic for downsampling a signal.
- Then take a signal as input from the user.
- Pass the signals to the user-defined downsample function myDownsample.
- Call the built-in downsample function for comparison.
- Plot resultant signals.

Output / Graphs / Plots / Results:

```
>> task8
Enter a discrete-time signal: [1 4 16 6 12]
>>
```

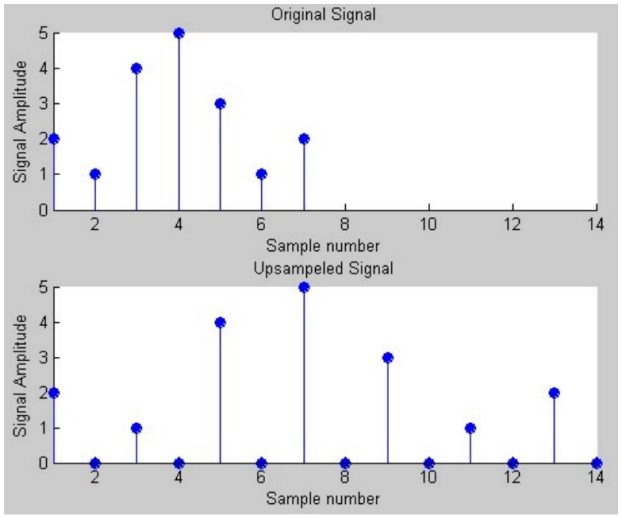


Discussion and Conclusion:

MATLAB can be used to downsample a signal.

Task # 09:

Write your own function to upsample a signal i.e. copy the 1st sample of original signal in the new signal and then place an extra sample of 0, copy the 2nd sample of original signal and then place a 0, and so on. See Figure 4.8 for example. Call this function from a Matlab file. Verify your result by using the command "upsample". Plot the original signal, upsampled signal determined by your program, and upsampled signal obtained by the command upsample.



Problem Analysis:

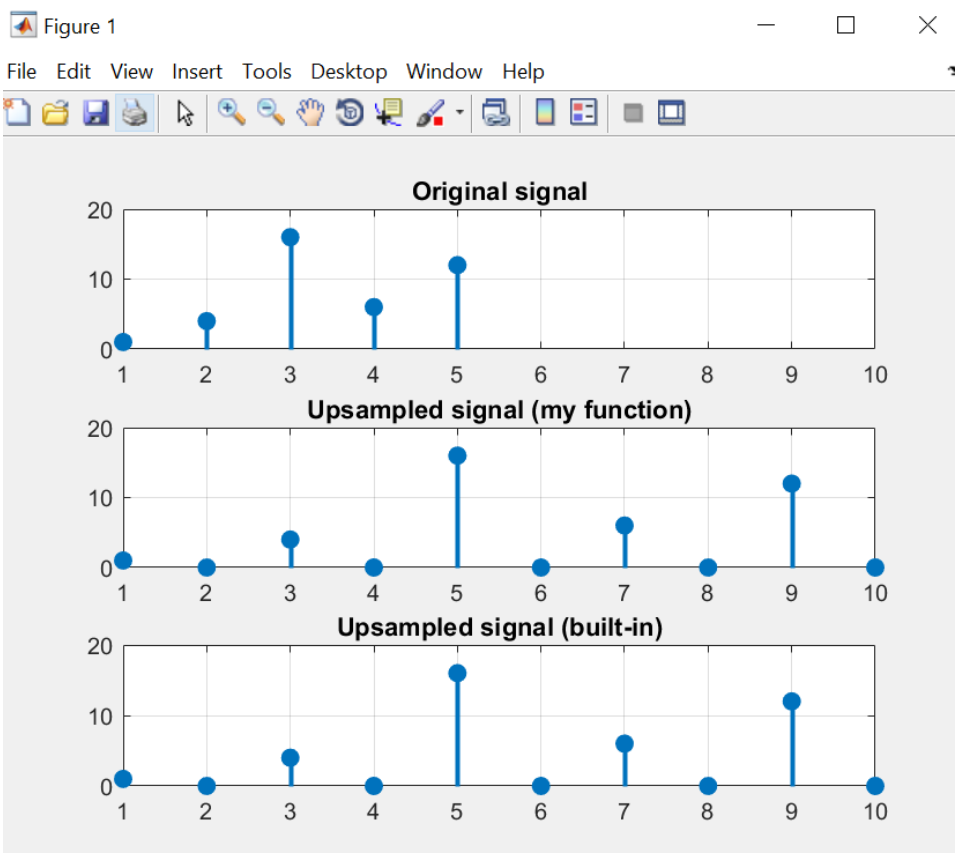
We also perform upsampling, which is similar to downsampling.

Algorithm:

- Make a function myUpsample and make logic for upsampling a signal.
- Then take a signal as input from the user.
- Pass the signals to the user-defined upsample function myUpsample.
- Call the built-in upsample function for comparison.
- Plot resultant signals.

Output / Graphs / Plots / Results:

```
>> task9
Enter a discrete-time signal: [1 4 16 6 12]
>>
```



Discussion and Conclusion:

MATLAB can be used to Upsample a signal.

Task # 10:

Plotting 3-D graphics with Matlab This is a complementary task for practicing 3d graphs in Matlab.

Surf command is used in Matlab for plotting 3D graphs, while the meshgrid command is used for setting up 2D plane.

clear all

```
close all
% set up 2-D plane by creating a -2:.2:2 sequence and copying it to all rows of x size (-2:.2:2)
times and vice versa
[x,y] = meshgrid([-2:.2:2]);

% plot 3D on plane

Z = x.*exp(-x.^2-y.^2);
figure

% surface plot, with gradient(Z) determining color distribution

surf(x,y,Z,gradient(Z))

% display color scale, can adjust location similarly to legend
colorbar
```

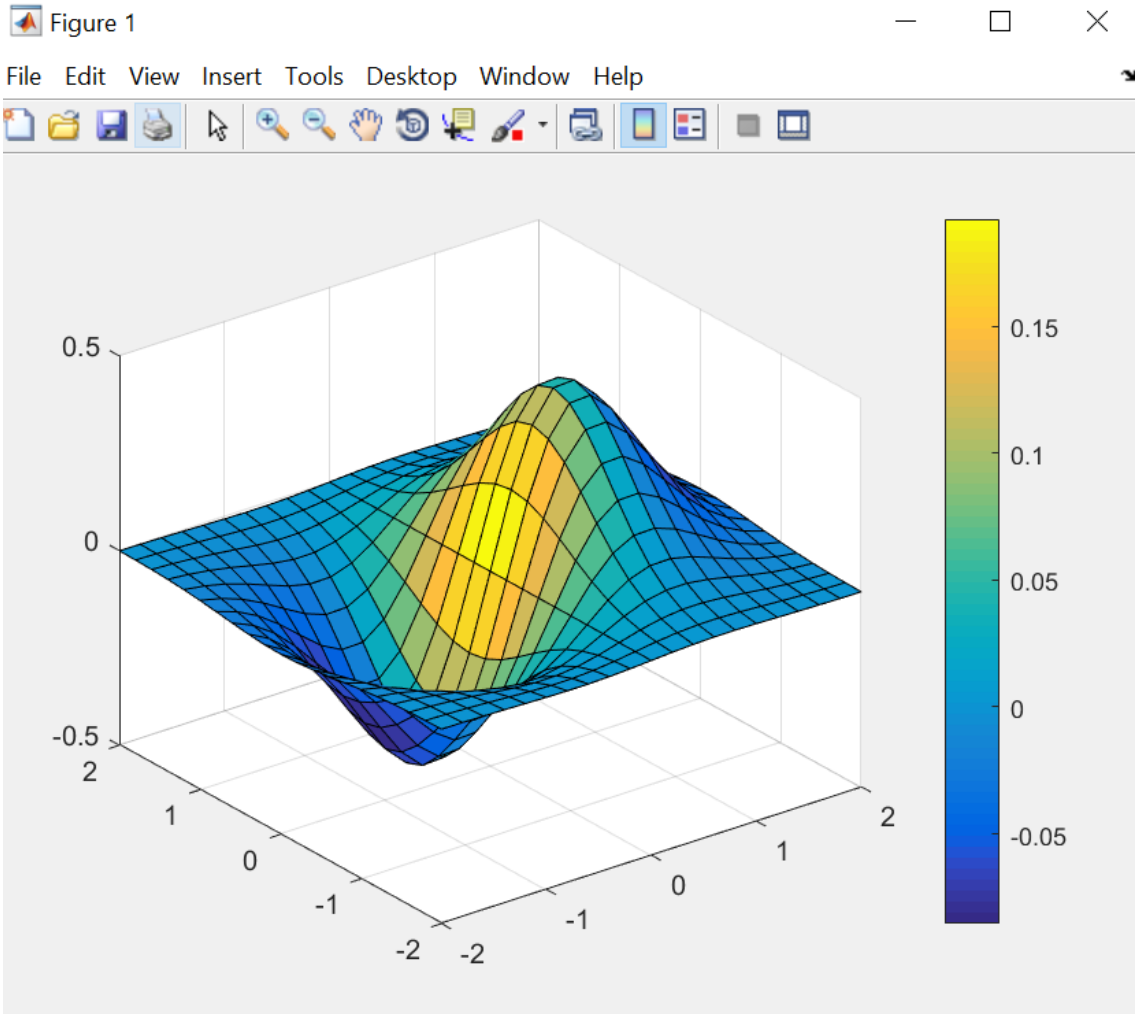
Problem Analysis:

We will see how to plot 3D graphs in MATLAB.

Algorithm:

- Write the code
- Execute code
- Note Observations

Output / Graphs / Plots / Results:



Discussion and Conclusion:

MATLAB can be used for 3D plotting.