

# Namal University, Mianwali

## Department of Electrical Engineering

EE 345 (L) – Digital Signal Processing (Lab)

Lab – 1 **Report**

**MATLAB Review: Basic Operations and Signals**

|  |  |
| --- | --- |
| Student Name | Student ID |
| **Muhammad Junaid** | **NIM-BSEE-2021-10** |

## Instructor: Zulaikha Kiran Lab Engineer: Faizan Ahmad

# Introduction

The purpose of this lab is to enable the student’s basic review of MATLAB operations and functions, and matrix operations using MATLAB.

# Course Learning Outcomes

CLO1: Develop algorithms to perform signal processing techniques on digital signals using MATLAB and DSP Kit DSK6713

CLO3: Deliver a report/lab notes/presentation/viva, effectively communicating the design and analysis of the given problem

# Equipment

 Software

o MATLAB

# Instructions

1. This is an individual lab. You will perform the tasks individually and submit a report.
2. Some of these tasks are for practice purposes only while others (marked as ‘Exercise’) have to be answered in the report.
3. When asked to display an image/ graph in the exercise, either save it as jpeg or take a screenshot, in order to insert it in the report.
4. The report should be submitted on the given template, including:
   1. Code (copy and pasted, NOT a screenshot)
   2. Output values (from command window, can be a screenshot)
   3. Output figure/graph (as instructed in 3)
   4. Explanation where required
5. The report should be properly formatted, with easy to read code and easy to see figures.

**Objectives:**

* How to get familiar with the interface of MATLAB
* Basic arithmetic operators and commands
* MATLAB variables
* Matrix manipulation
* Signal Plotting

**Tools used:**

* MATLAB

**Basic Arithmetic Operations:**

|  |  |  |
| --- | --- | --- |
| **Operations** | **Symbol** | **Example** |
| Addition | + | 3 + 22 |
| Subtraction | - | 54.4 – 16.5 |
| Multiplication | \* | 3.14 \* 6 |

Division / 10/2

**Variables Naming Rule:**

Variable names can contain up to 63 characters.

Punctuation characters are not allowed, because many of them have special meanings in MATLAB.

**MATLAB Special Variables** ans Default variable name for results

pi Value of 

inf Infinity

NaN Not a number e.g. 0/0

realmin The smallest usable positive real number realmax The largest usable positive real number

**Types of Variables**

## Type Examples

Integer 1362,-5656

Real 12.33,-56.3

Complex X=12.2 – 3.2j (j = sqrt(-1))

Complex numbers in MATLAB are represented in rectangular form.

To separate real & imaginary part

H = real(X)

K= imag(X)

Conversion between polar & rectangular

C1= 1-2i

Magnitude: mag\_c1 = abs(C1) Angle: angle\_c1 = angle(C1) Note that angle is in radians

**MATLAB Matrices**

MATLAB treats all variables as matrices. For our purposes a matrix can be thought of as an array, in fact, that is how it is stored.

Vectors are special forms of matrices and contain only one row OR one column.

Scalars are matrices with only one row AND one column.

A matrix can be created in MATLAB as follows (note the commas AND semicolons):

» matrix = [1 , 2 , 3 ; 4 , 5 ,6 ; 7 , 8 , 9]

## Row Vector

A matrix with only one row is called a row vector. A row vector can be created in MATLAB as follows (note the commas):

» rowvec = [12 , 14 , 63]

rowvec =

12 14 63

Row vector can also defined in a following way:

rowvec = 2 : 2 : 10;

rowvec =

2 4 6 8 10

## Column Vector

A matrix with only one column is called a column vector. A column vector can be created in MATLAB as follows (note the semicolons):

» colvec = [13 ; 45 ; -2]

colvec =

13

45 -2

## Extracting a column vector

A column vector can be extracted from a matrix. As an example we create a matrix below:

» matrix=[1,2,3;4,5,6;7,8,9] Here we extract the column 2 of matrix and

make a column vector: here colon denote all rows and 2 denote second column

matrix = » col\_two =matrix(: , 2)

1 2 3 col\_two = 2

4 5 6 5

7 8 9 8

## Extracting a row vector

A row vector can be extracted from a matrix. As an example we create a matrix below:

» matrix=[1,2,3;4,5,6;7,8,9] Here we extract row 2 of matrix and make a row

vector.

matrix = here colon denote all column and 2 denote second row. 1 2 3 » rowvec =matrix(2, : ) 4 5 6 rowvec =

7 8 9 4 5 6

matrix(2:3,1:2)

4 5

7 8

## Concatenation

New matrices may be formed out of old ones Suppose we have: a = [1 2; 3 4]

a = [1 2 ;3 4]

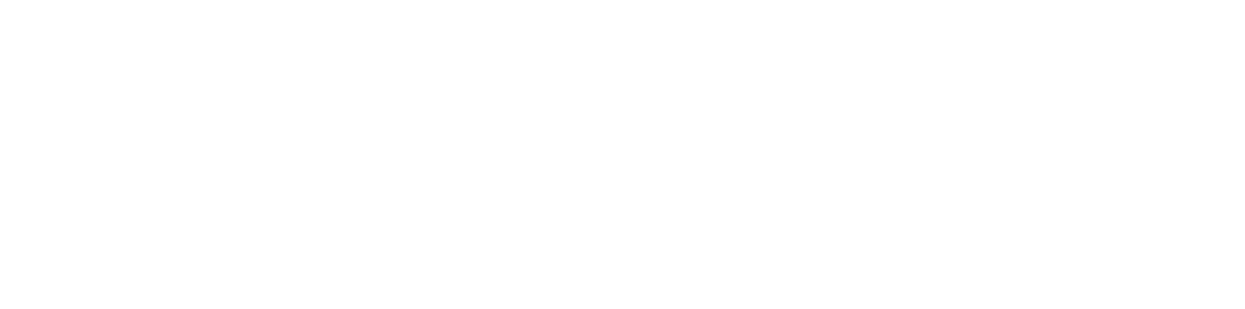
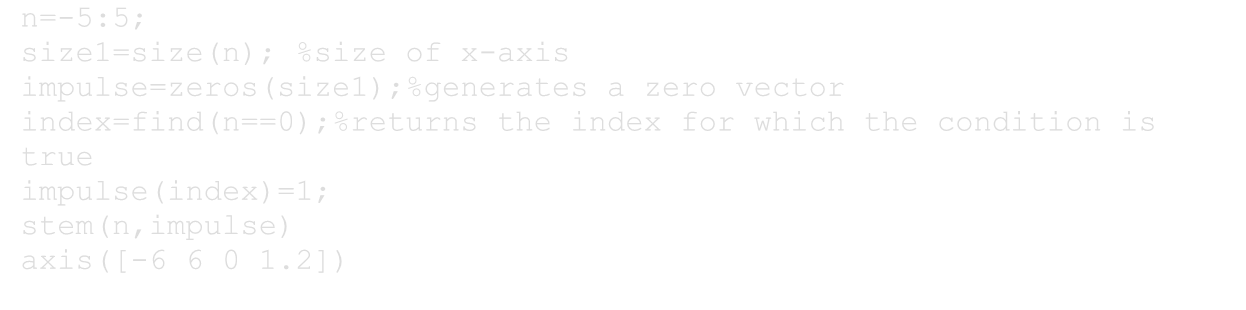
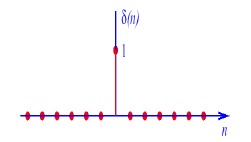
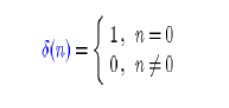
Input output

[a , a, a] ans = 1 2 1 2 1 2

3 4 3 4 3 4

**The Unit Delta (Impulse) function:**

The unit sample, denoted by S(n), is defined by and plays the same role in discrete-time signal processing that the unit Impulse plays in continuous-time signal processing.



n=

-

5:5

;

size1=size(n);

%size of x

-

axis

impulse

zeros(size1);

=

%generates a zero vector

index=find(n==0);

%returns the index for which the condition is

true

impulse(index)=1;

stem(n,impulse)

axis([

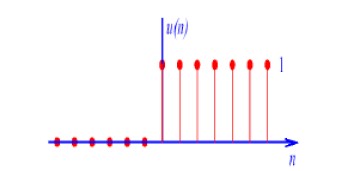
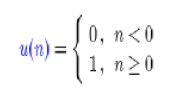
-

6

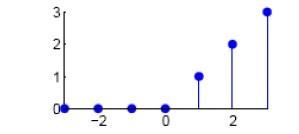
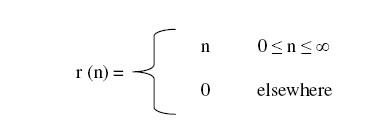
6 0 1.2])

**The Unit step function:**

The unit step, denoted by u(n), is defined by



**The Unit Ramp Function:**



Lab Exercise

**Question 1.1**

Perform the given commands

**CODE for all below commands**

clear all

clc

a=[1 4 3;4 2 6 ;7 8 9]

K=det(a)

K=inv(a)

K=a'

K=min(a)

K=min(min(a))

K=max(a)

K=max(max(a))

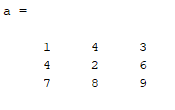
K=a.^2

K=sum (a)

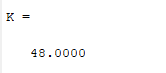
K=sum(sum(a))

K=size (a)

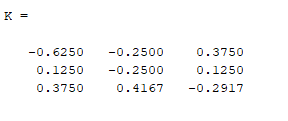
* **Let a=[1 4 3;4 2 6 ;7 8 9]**



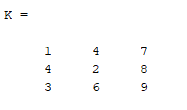
* **Determinant of matrix a. det(a) = ?**



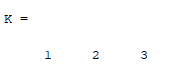
* **Inverse of matrix a. inv(a) = ?**



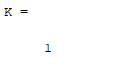
* **Transpose of matrix a. a’ = ?**



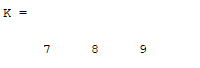
* **A row vector containing the minimum element from each column of a. min(a) = ?**



* **The smallest element in matrix a. min(min(a)) = ?**



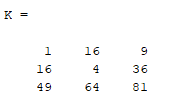
* **A row vector containing the maximum element from each column of a. max(a) = ?**



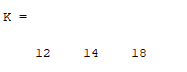
* **The max element from matrix a. max(max(a)) = ?**



* **Bitwise calculate the square of each element of matrix a. a.^2 = ?**



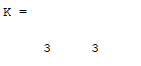
* **Treats the columns of ‘a’ as vectors, returning a row vector of the sums of each column. sum (a) = ?**



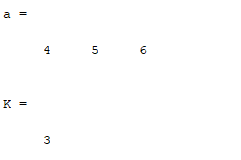
* **Sum of all the elements in the matrix. sum(sum(a)) = ?**



* **Give the number or rows and the number of columns of matrix a. size (a) = ?**



* **Let a =[ 4 5 6] , find the number of elements in row vector. length(a) = ?**



**Explain**

1. a(1:2,1:2)

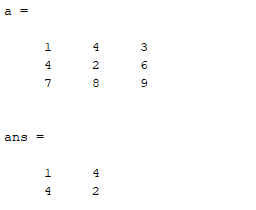
* **CODE**

clear all

clc

a=[1 4 3;4 2 6 ;7 8 9]

a(1:2,1:2)



b) a(1:2)

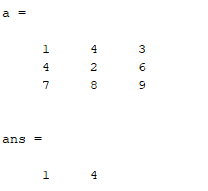
* **CODE**

clear all

clc

a=[1 4 3;4 2 6 ;7 8 9]

a(1:2)



c) a(3)

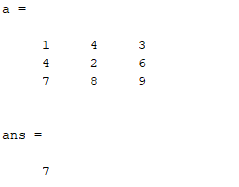
* **CODE**

clear all

clc

a=[1 4 3;4 2 6 ;7 8 9]

a(3)



**Explanation**

In MATLAB, 'a' undergoes crucial operations: det(a) for determinant, inv(a) for the inverse, and a' for transposition. Functions min(a) and max(a) yield row vectors for column-wise minima and maxima. Nested operations min(min(a)) and max(max(a)) identify the smallest and largest elements. Element-wise squaring (a.^2) and sum(a) for column-wise sums are performed. sum(sum(a)) gives the total sum. size(a) provides matrix dimensions, and length(a) counts row vector elements. Submatrix a(1:2,1:2) extracts the top-left 2x2 portion, a(2,1) accesses a specific element, and a(3) selects an element in the third row and first column, succinctly illustrating 'a' analysis in MATLAB.

**Question 1.2**

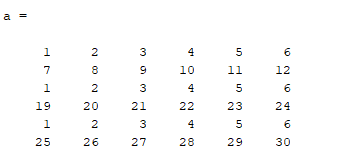
1. Generate a 6x6 matrix A

**CODE**

clear all

clc

a=[1,2,3,4,5,6 ; 7,8,9,10,11,12 ; 1,2,3,4,5,6 ; 19,20,21,22,23,24 ; 1,2,3,4,5,6 ; 25,26,27,28,29,30]



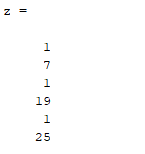
1. Generate a 6x1 matrix z

**CODE**

clear all

clc

z=[1 ; 7 ; 1 ; 19 ; 1 ; 25]



1. Solve linear system of equations Ax=z

**CODE**

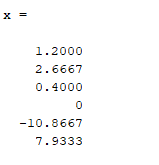
clear all

clc

a=[6,2,3,4,5,6 ; 7,8,9,10,11,12 ; 1,2,3,4,5,6 ; 19,20,6,22,23,24 ; 1,2,3,3,5,6 ; 25,23,27,28,29,30]

z=[7 ; 9 ; 1 ; 19 ; 1 ; 25]

x= (inv(a))\*z



1. Compute determinant of matrix A

**CODE**

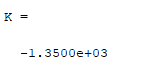
clear all

clc

a=[6,2,3,4,5,6 ; 7,8,9,10,11,12 ; 1,2,3,4,5,6 ; 19,20,6,22,23,24 ; 1,2,3,3,5,6 ; 25,23,27,28,29,30]

z=[7 ; 9 ; 1 ; 19 ; 1 ; 25]

K=det(a)



1. Extract a 4x4 matrix from the 6x6 matrix and 4x1 matrixes from 6x1

**CODE**

clear all

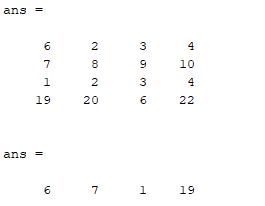
clc

a=[6,2,3,4,5,6 ; 7,8,9,10,11,12 ; 1,2,3,4,5,6 ; 19,20,6,22,23,24 ; 1,2,3,3,5,6 ; 25,23,27,28,29,30]

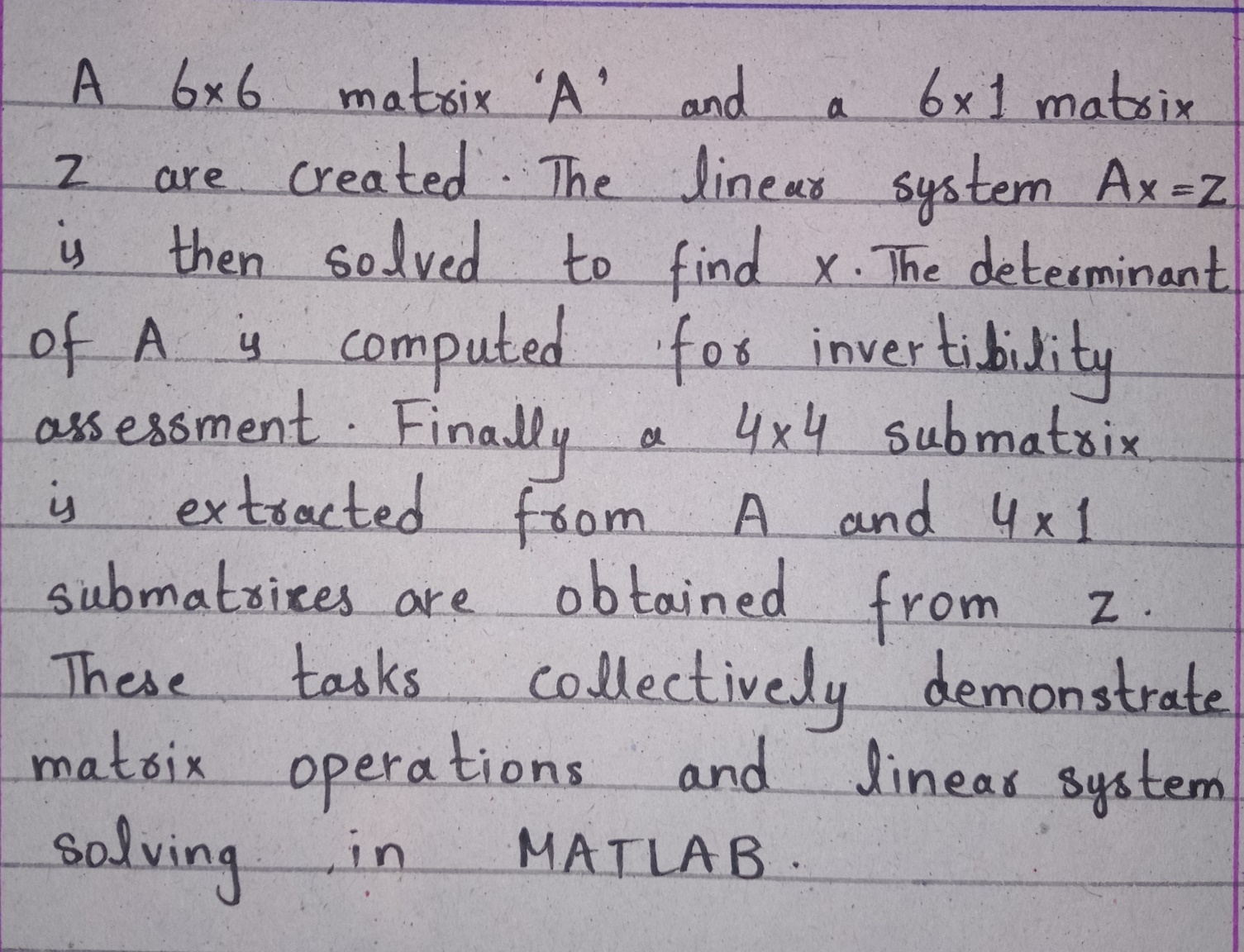
z=[7 ; 9 ; 1 ; 19 ; 1 ; 25]

a(1:4 , 1:4)

a(1:4)



**Explanation**



**Question 1.3**

**1)**

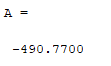


**CODE**

clear all

clc

A= (35.7\*(64-(7^4)))/(45+(5^3))



**2)**

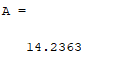


**CODE**

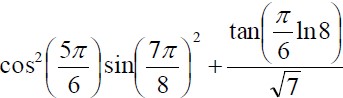
clear all

clc

A=(((3^7)\*log10(76))/((7^3)+564))+((913)^(1/3))



**3)**

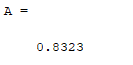


**CODE**

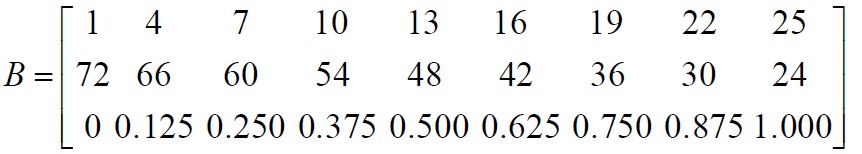
clear all

clc

A=(((cos((5\*pi)/6))^2)\*(sin((7\*pi)/8)^2))+((tan((pi\*log(8))/6)/(sqrt(7))))



Create the matrix shown below by using the vector notation for creating vectors with constant spacing and/or the linspace command when entering the rows.



**CODE**

clear all

clc

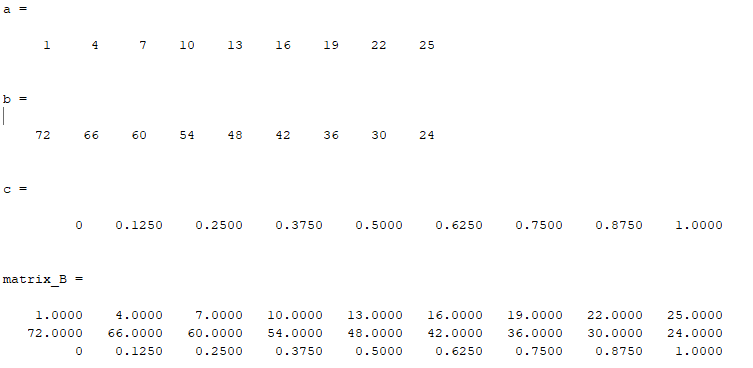
a=linspace(1,25,9)

b=linspace(72,24,9)

c=linspace(0,1,9)

matrix\_B=[a;b;c]

**Output**



**Question 1.4**

1. Write a MATLAB code that generates Delta (Impulse) Function.
   * Use the plotting function stem to make the graphs

**CODE**

clear all

clc

n= -5:5;

sizej = size(n);

impulse = zeros(sizej);

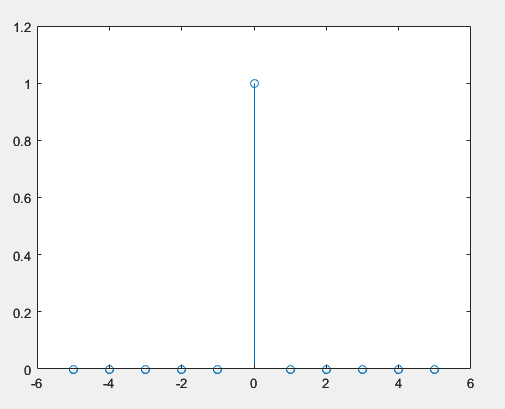
index = find(n==0);

impulse(index) = 1;

stem(n,impulse)

axis([-6 6 0 1.2])

**OUTPUT**



* + Replace the stem command in the above code with the plot command and run the code again. How does this change the plot? And why?

**CODE**

clear all

clc

n = -5:5;

sizej = size(n);

impulse = zeros(sizej);

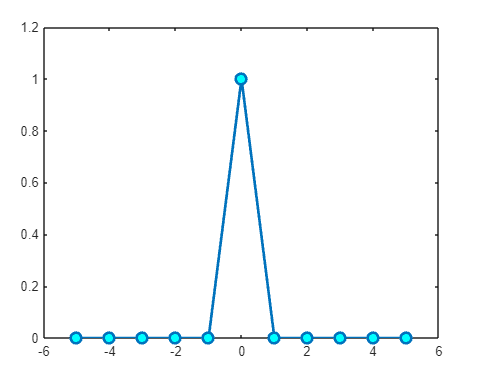
index = find(n == 0);

impulse(index) = 1;

plot(n, impulse, 'o-', 'MarkerFaceColor', 'c', 'MarkerSize', 8, 'LineWidth', 2);

axis([-6 6 0 1.2]);

**OUTPUT**

****

**Explanation**

Replacing `stem` with `plot` connects data points with lines, creating a continuous curve instead of distinct stems. `plot` naturally connects points, resulting in a smoother representation compared to the discrete stems produced by `stem`, which is specifically designed for stem plots.

1. Write a MATLAB code that generate Unit Step function:
   * Use the plotting function stem to make the graphs
   * Replace the stem command in the above code with the plot command and run the code again. How does this change the plot?

**CODE**

clear all

clc

n= -5:5;

sizej = size(n);

STEP = zeros(sizej);

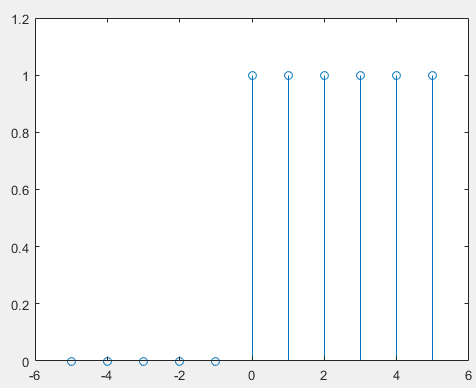
index = find(n>=0);

STEP(index) = 1;

stem(n,STEP)

axis([-6 6 0 1.2])

**OUTPUT**



* + Replace the stem command in the above code with the plot command and run the code again. How does this change the plot?

**CODE**

clear all

clc

n = -5:5;

sizej = size(n);

STEP = zeros(sizej);

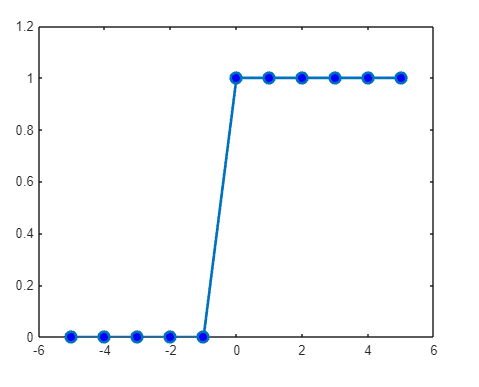
index = find(n >= 0);

STEP(index) = 1;

plot(n, STEP, 'o-', 'MarkerFaceColor', 'b', 'MarkerSize', 8, 'LineWidth', 2);

axis([-6 6 0 1.2]);

**OUTPUT**

****

**Explanation**

Replacing `stem` with `plot` connects data points with lines, resulting in a continuous curve for the step function instead of distinct stems.

1. Write a MATLAB code that generate Unit Ramp function
   * Use the plotting function plot to make the graphs

**CODE**

n=-6:6;

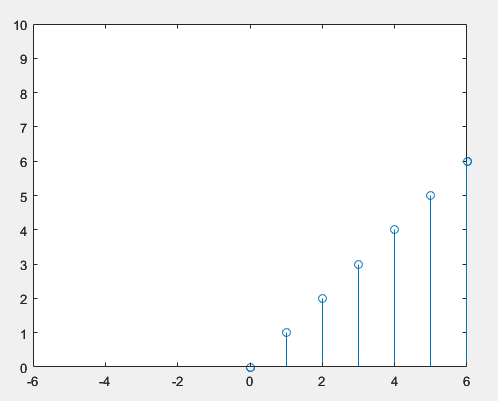
size1=size(n);

index=find(n<=0)

stem(n,n)

axis([-6 6 0 10])

**OUTPUT**



* + Replace the plot command in the above code with the stem command and run the code again. How does this change the plot?

**CODE**

n = -6:6;

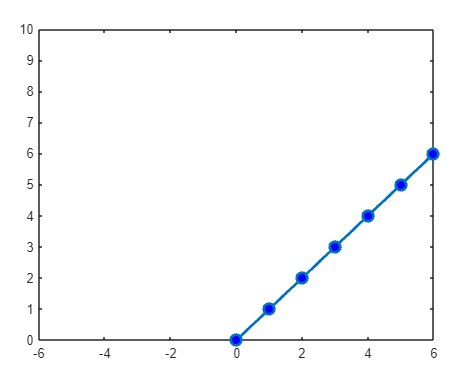
size1 = size(n);

index = find(n <= 0);

plot(n, n, 'o-', 'MarkerFaceColor', 'b', 'MarkerSize', 8, 'LineWidth', 2);

axis([-6 6 0 10]);

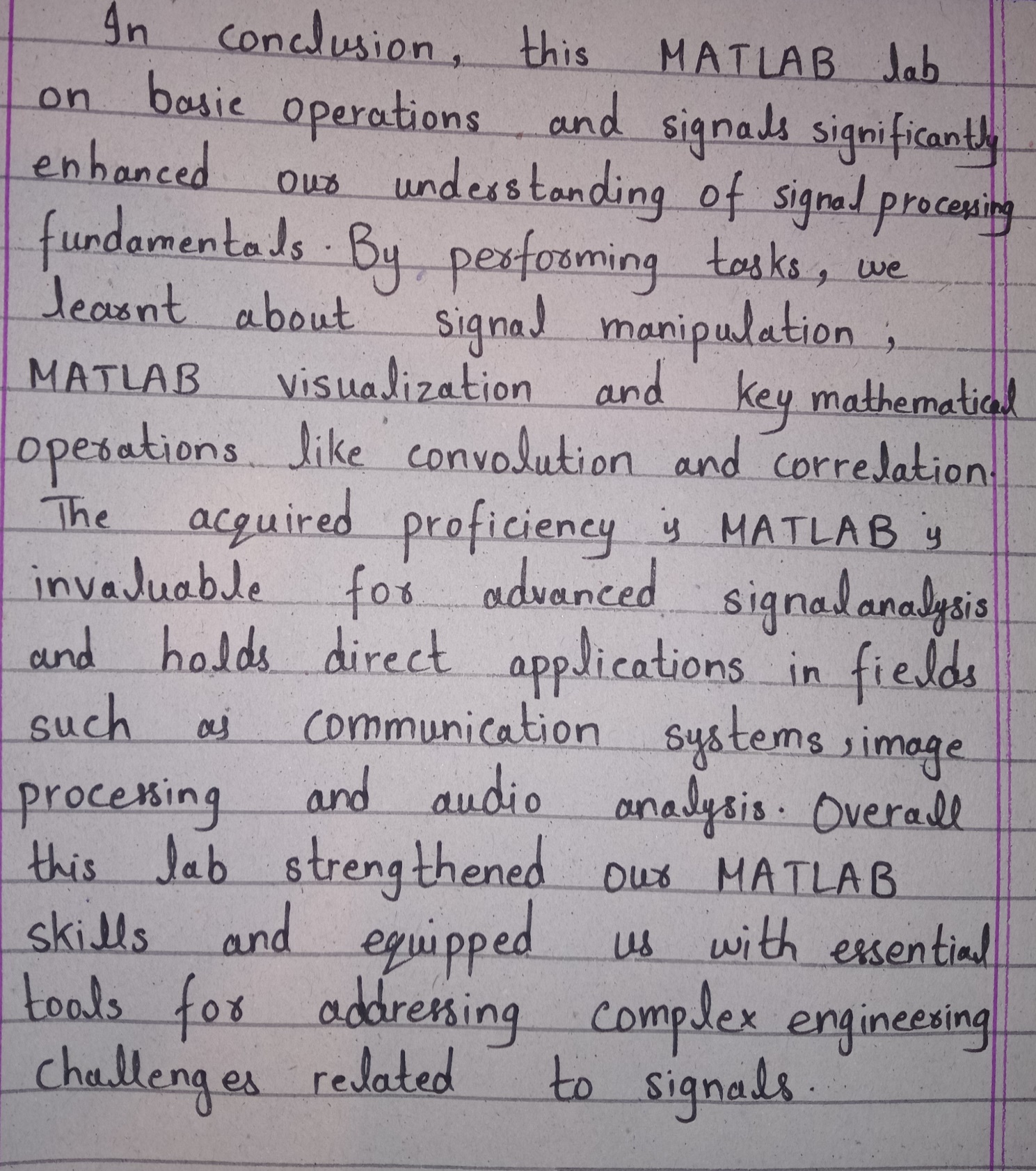
**OUTPUT**

****

**Explanation**

Replacing `plot` with `stem` in the code results in a plot where individual stems rise from the x-axis to each data point, creating a distinct representation compared to the continuous curve produced by the `plot` function.

**CONCLUSION**

****

## Evaluation Rubric

* **Method of Evaluation**: In-lab marking by instructors, Report submitted by students
* **Measured Learning Outcomes**:

CLO1: Develop algorithms to perform signal processing techniques on digital signals using MATLAB and DSP Kit DSK6713 CLO3: Deliver a report/lab notes/presentation/viva, effectively communicating the design and analysis of the given problem

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Excellent 10 | Good  9-7 | Satisfactory 6-4 | Unsatisfactory 3-1 | Poor 0 | Marks Obtained |
| Tasks (CLO1-P3) | All tasks completed correctly. Correct code with proper comments. | Most tasks completed correctly. | Some tasks completed correctly. | Most tasks incomplete or incorrect. | All tasks incomplete or incorrect. |  |
| Output (CLO1-P3) | Output correctly shown with all  Figures/Plots displayed as  required and properly labelled | Most Output/Figures/Plots displayed with proper labels | Some Output/Figures/Plots displayed with proper labels  OR Most Output/Figures/Plots displayed but without proper  labels | Most of the required  Output/Figures/Plots not displayed | Output/Figures/Plots not displayed |  |
| Answers (CLO1-P3) | Meaningful answers to all questions. Answers show the understanding of the student. | Meaningful answers to most questions. | Some correct/ meaningful answers with some irrelevant ones | Answers not understandable/ not relevant to questions | Does not write any Answer |  |
| Report (CLO3-A3) | Report submitted with proper  grammar and  punctuation with proper  conclusions  drawn and good formatting | Report submitted with proper conclusions drawn with good formatting but  some grammar mistakes OR proper grammar but not very good formatting | Some correct/ meaningful conclusions. Some parts of the document not properly  formatted or some grammar  mistakes | Conclusions not based on results. Bad formatting with no proper grammar/punctuation | Report not submitted |  |
|  |  |  | Total | | |  |