EXPERIMENT-2

NAME: Chris Parmar

PRN : 20220802034

SUB : DSP LAB ASSIGNMENT 2

Title UnitImpulseResponseinMATLAB **Aim:**
TogeneratetheunitimpulsesignalandobserveitsresponseusingMATLAB.

Requirements:
MATLAB software (Any version)
Basic understanding of signal processing
Basic knowledge of discrete-time signals and impulse response
Theory:
Theunitimpulsesignal,denotedasδ[n],isoneofthemostfundamentalsignals
in signal processing. It is defined as:
δ[n]={ 1,0,}if n=0

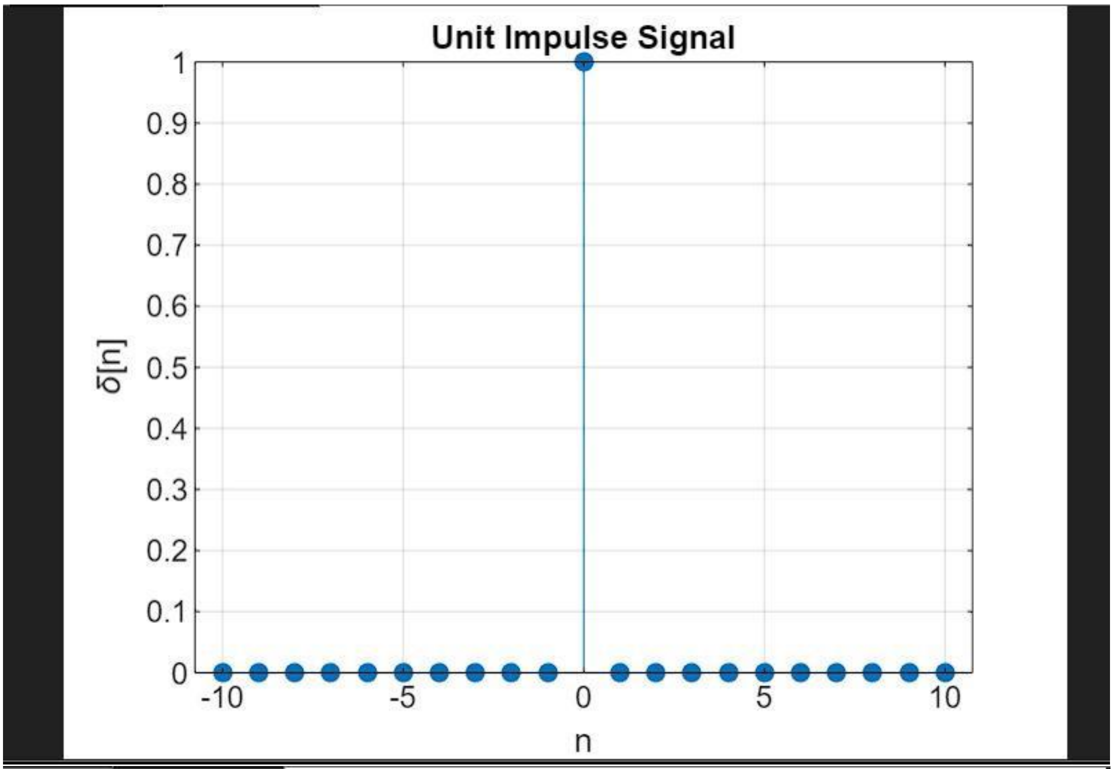
The impulse signal is often used to test the dynamic properties of a system. In terms of system analysis, the impulse response helps in determining the behavior of a linear time-invariant (LTI) system. When an LTI system is excited by a unit impulse input, the output of the system is known as the system's impulse response.
The discrete-time impulse response function is useful for analyzing systems' stability, frequency response, and time-domain behavior.

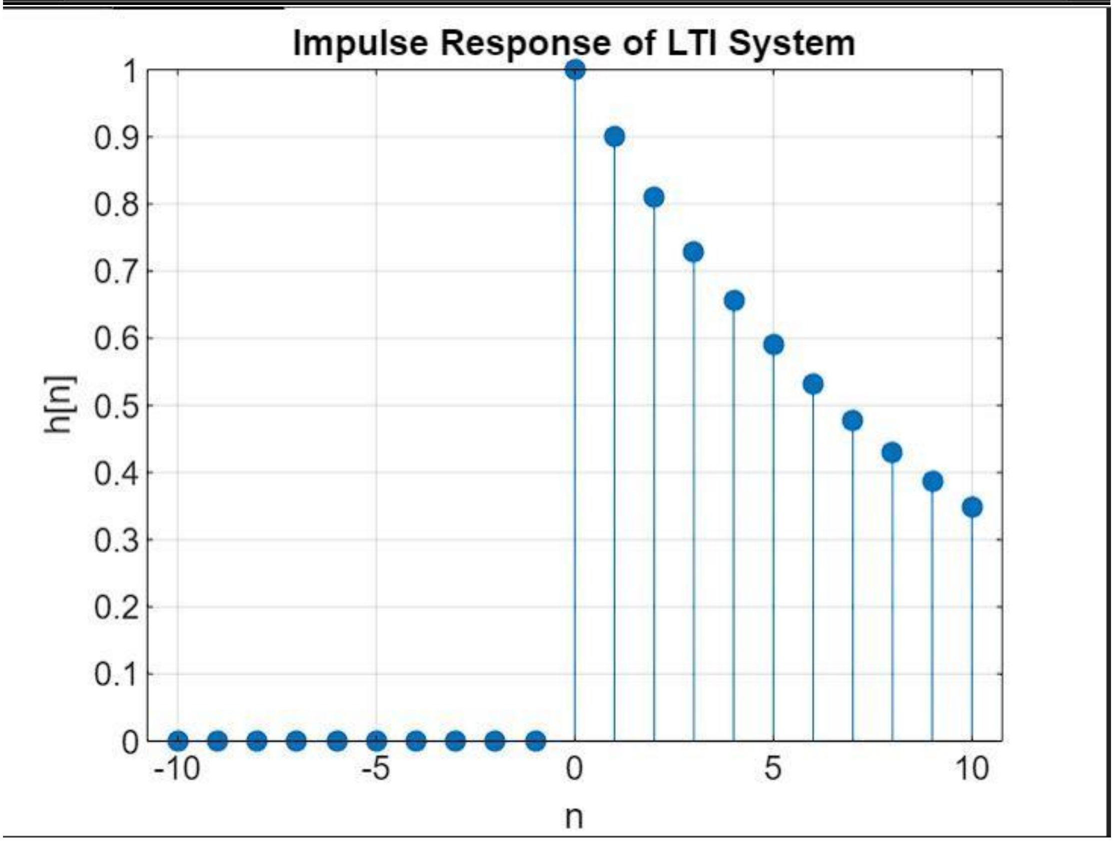
MATLAB Code:

/MATLAB Drive/Unit_implseee.m

```
% MATLAB code to generate and plot the unit impulse signal

% Define the number of points (n range)
n = -10:10; % Discrete time index from -10 to 10

% Generate unit impulse signal
impulse_signal = (n == 0); % Impulse occurs when n = 0

% Plot the impulse signal
stem(n, impulse_signal, 'filled');
title('Unit Impulse Signal');
xlabel('n');
ylabel('\delta[n]');
grid on;

% Impulse response of an LTI system (for example: h[n] = (0.9)^n u[n])
h_n = (0.9).^n .* (n >= 0); % LTI system response

figure;
stem(n, h_n, 'filled');
title('Impulse Response of LTI System');
xlabel('n');
ylabel('h[n]');
grid on;
```

OUTPUT:





Observation:

The unit impulse signal is a discrete-time signal that takes the value 1 at n=0 and 0 at all other points.

The impulse response of the system gives us information about how the system reacts to an impulse input. In this case, the system response decays exponentially after n=0 with a factor of 0.9. **Calculation:**

The impulse signal is created based on the condition n == 0.
The response of the LTI system to this unit impulse is calculated using the formula h[n] = (0.9)^n u[n] for n >= 0.
Conclusion:
The unit impulse function is an essential tool for analyzing the time-domain response of discrete-time systems. The response of the system to this unit impulse can be studied to derive various properties of the system, including stability and frequency characteristics.

Title: Step, Sine, and Cosine Signals in MATLAB
Aim:
To generate step, sine, and cosine signals in MATLAB and observe their properties.

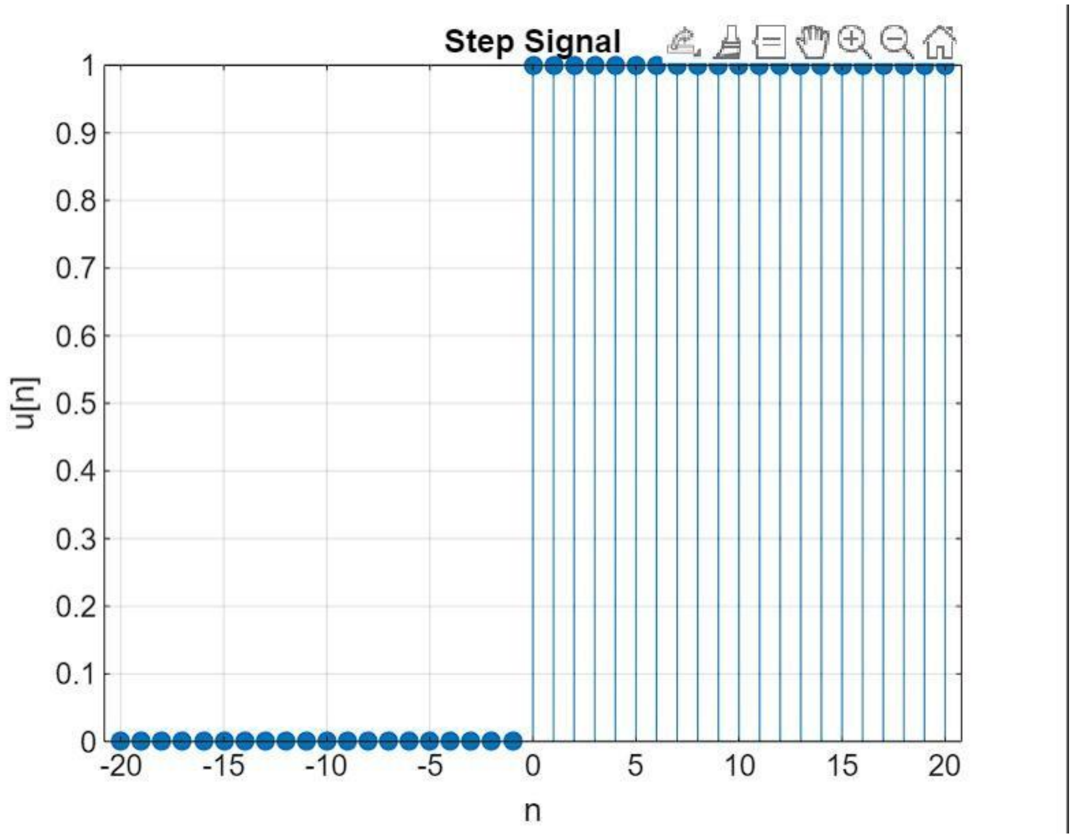
Requirements:
MATLAB software (Any version)
Basic understanding of signal processing
Basic knowledge of discrete-time signals and time-domain representation
Theory:
Step Signal: A step signal, also known as a unit step function, is defined as:
u[n]={ 1,0,}
if n≥0 if
n<0
It represents a sudden transition from 0 to 1 and is often used to analyze the behavior of dynamic systems.
Sine Signal: A sine signal is a periodic signal defined by the equation:
x[n]=A·sin(2πfn+ϕ) where:
- A is the amplitude,
- f is the frequency,
- n is the time index, and ➢ ϕ is the phase offset.

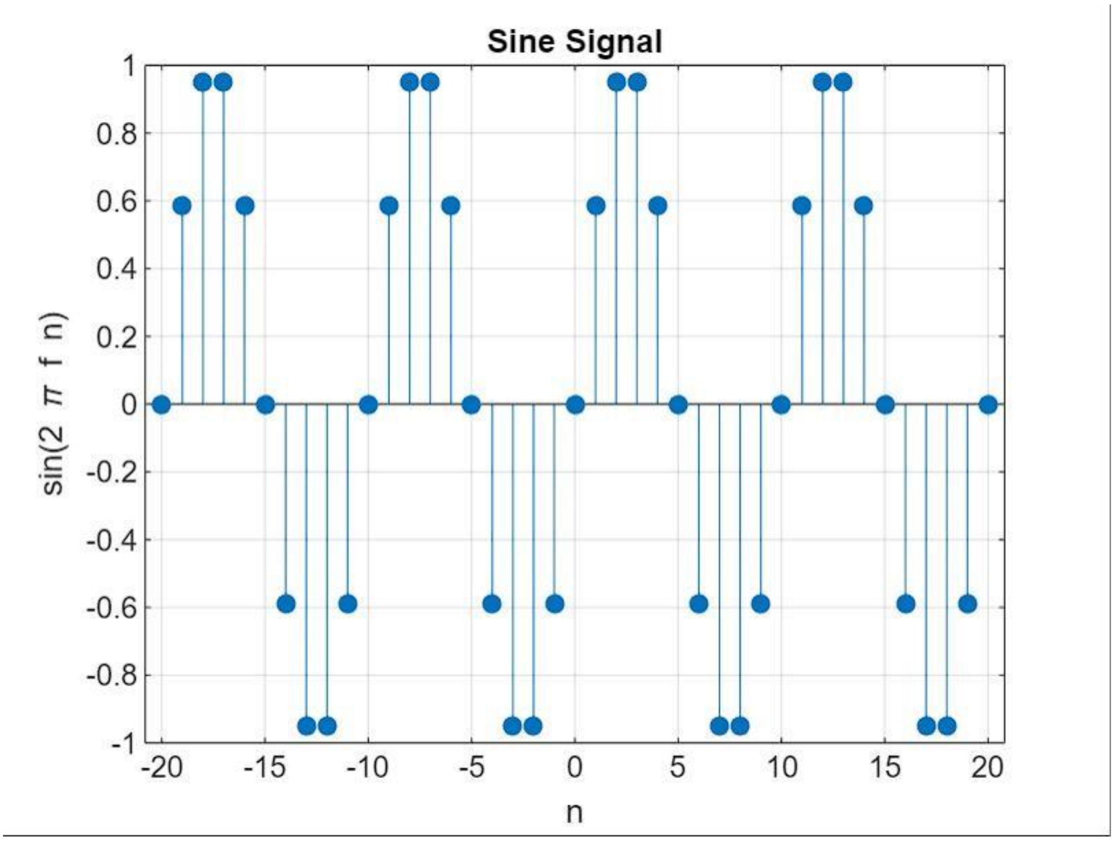
Cosine Signal: A cosine signal is similar to the sine signal but shifted in phase by
x[n]=A·cos(2πfn+ϕ)
These signals are fundamental in understanding the behavior of many systems and are used widely in both time-domain and frequency-domain analyses.
MATLAB Code:

/MATLAB Drive/lab2b.m

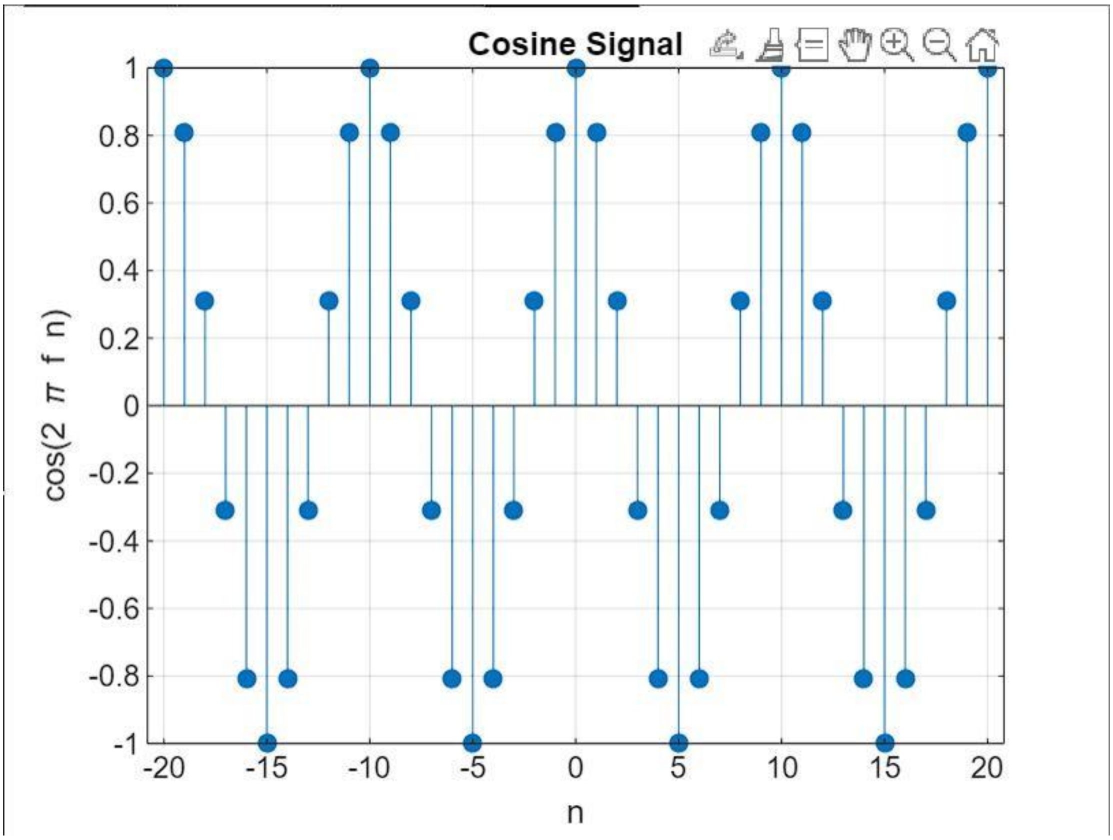
```matlab
% MATLAB code to generate step, sine, and cosine signals
% Define the time vector (n range)
n = -20:20; % Discrete time index from -20 to 20

%% 1. Step Signal
step_signal = (n >= 0); % Step signal is 1 for n >= 0, 0 otherwise
% Plot step signal
figure;
stem(n, step_signal, 'filled');
title('Step Signal');
xlabel('n');
ylabel('u[n]');
grid on;

%% 2. Sine Signal
A = 1;         % Amplitude
f = 0.1;       % Frequency in Hz
phi = 0;       % Phase shift

% Generate sine signal
sine_signal = A * sin(2 * pi * f * n + phi);

% Plot sine signal
figure;
stem(n, sine_signal, 'filled');
title('Sine Signal');
xlabel('n');
ylabel('sin(2 \pi f n)');
grid on;

%% 3. Cosine Signal
% Generate cosine signal
cosine_signal = A * cos(2 * pi * f * n + phi);

% Plot cosine signal
figure;
stem(n, cosine_signal, 'filled');
title('Cosine Signal');
xlabel('n');
```

OUTPUT:





Sine Signal
sin(2 π f n)
n
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-20
-15
-10
-5
0
5
10
15
20



Cosine Signal
cos(2 π f n)
n
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-20
-15
-10
-5
0
5
10
15
20

Observation:
<u>Step Signal:</u>

- The step signal transitions from 0 to 1 at n = 0.
- This signal is often used to test system responses.

<u>Sine Signal:</u>

- The sine wave oscillates between +1 and -1 with a periodicity determined by the frequency f.
- It represents a continuous wave that oscillates as a function of time. <u>Cosine Signal:</u>

- The cosine wave is similar to the sine wave but starts from its maximum amplitude (A) at n = 0.
- Like the sine wave, it oscillates between +1 and -1.

Calculation:
The step signal is generated using the condition (n >= 0).
The sine signal is calculated using the formula
sin(2πfn+ϕ), where f is the frequency of the signal. The
cosine signal is calculated similarly using cos(2πfn+ϕ).
Conclusion:
The step signal is a basic building block in signal processing and is used to test system responses.
Sine and cosine signals are periodic and can be used to represent oscillating signals in both time and frequency domains.
These signals form the foundation for understanding more complex signal behaviors in both time-domain and frequency-domain analysis.

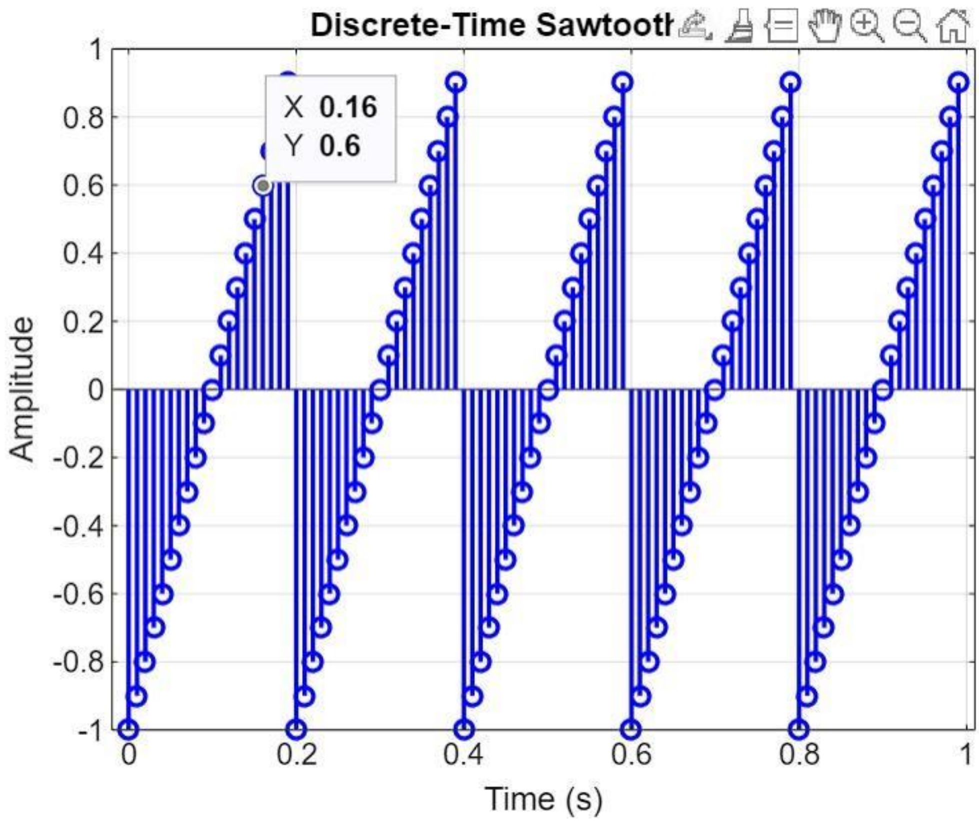
Title: Generation of Sawtooth Wave in MATLAB **Aim:**
To generate and visualize a sawtooth wave signal using MATLAB.

Requirements:
MATLAB software (Any version)
Basic understanding of signal processing and waveform generation
Theory:
A sawtooth wave is a periodic waveform that ramps upward and drops sharply.
It is named due to its resemblance to the teeth of a saw.
The sawtooth wave is widely used in signal processing, audio synthesis, and communication systems. It contains all harmonics of the fundamental frequency, and its spectrum is rich in odd and even harmonics, making it useful for sound synthesis.
MATLAB Code:

/MATLAB Drive/sawtooth_wave.m

```matlab

Fs = 1000;
T = 1;
f = 5;
t = 0:1/Fs:T-1/Fs;
y = sawtooth(2*pi*f*t);
figure;
stem(t, y, 'b', 'LineWidth', 1.5);
xlabel('Time (s)');
ylabel('Amplitude');
title('Discrete-Time Sawtooth Wave');
grid on;

```

OUTPUT:



Observation:

1. The sawtooth wave is generated with a periodic linear rise followed by a sharp drop, corresponding to the wave's frequency and time period.
2. The wave has a rich harmonic content due to the sharp discontinuity at the end of each period.
3. The time period and frequency of the wave are inversely related:

Calculation:

4. Frequency (f): The number of cycles per second is set as 5 Hz.
5. Time period (T): The time for one complete cycle is calculated as
6. Sampling frequency (Fs): The number of samples per second is set to 1000 Hz to capture the waveform accurately.
7.

Conclusion:

The sawtooth wave was successfully generated using MATLAB. The periodic waveform with a linear increase and a sudden drop was plotted. The characteristics of the wave such as its frequency and time period can be easily modified by adjusting the parameters. This waveform is fundamental in many signal processing applications, especially in audio and communication systems.