Pimpri Chinchwad Education Trust's
PIMPRI CHINCHWAD COLLEGE OF ENGINEERING NIGDI, PUNE-44
DEPARTMENT OF E & TC

SUB: Digital Signal Processing
Year & Branch: T.E. (E& TC)
2020-21

MINI-PROJECT REPORT

ON

"SIGNAL PROCESSING AND ANALYSIS (USING MATLAB)"

By

B1 Batch

Mrs. Sonali Sawant — **Course Faculty**
Dr. M.T Kolte — **HOD (E&TC)**

INDEX

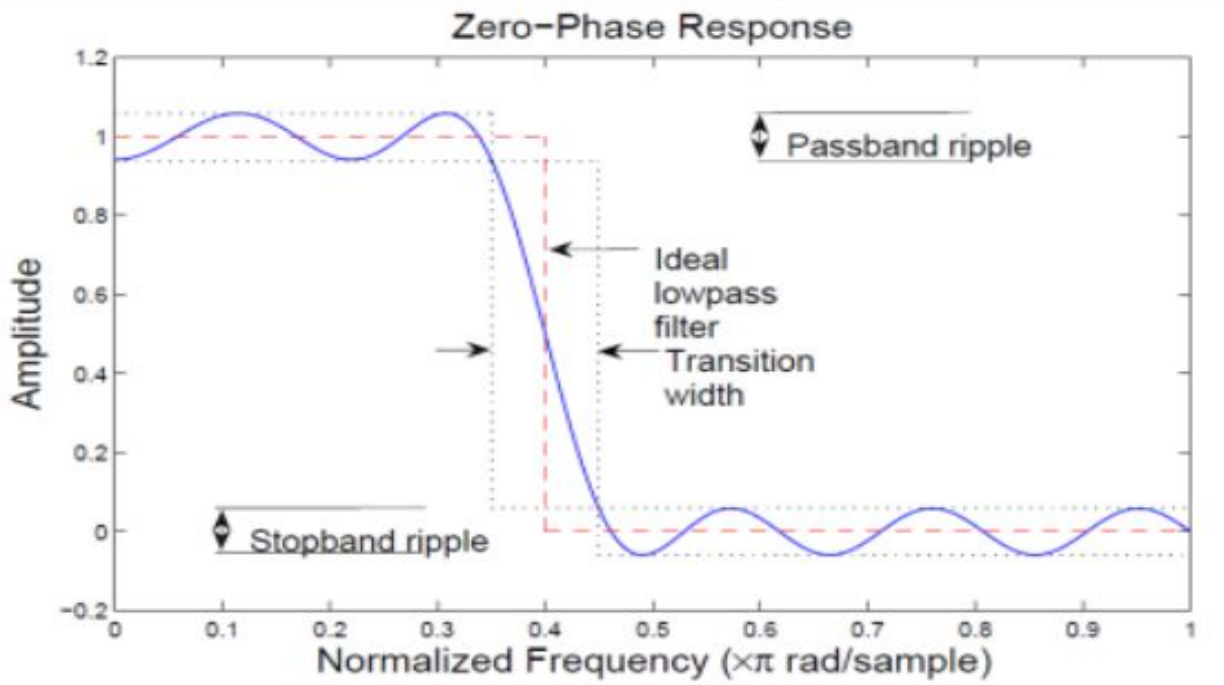
ABSTRACT

- Digital filters introduce delay in your signal. Depending on the filter characteristics, the delay can be constant over all frequencies, or it can vary with frequency.
- The type of delay determines the actions you have to take to compensate for it.
- The `grpdelay` function allows you to look at the filter delay as a function of frequency. Looking at the output of this function allows you to identify if the delay of the filter is constant or if it varies with frequency (in other words, if it is frequency-dependent).
- Filter delay that is constant over all frequencies can be easily compensated for by shifting the signal in time.
- FIR filters usually have constant delay. On the other hand, delay that varies with frequency causes phase distortion and can alter a signal waveform significantly.

TOOLS USED

- MATLAB R2020B

DIGITAL LOW PASS FILTER



- **Practical FIR designs typically consist of filters that have a transition width and maximum passband and stopband ripples that do not exceed allowable values.**

- **In addition to those design specifications, one must select the filter order, or, equivalently, the length of the truncated impulse response.**

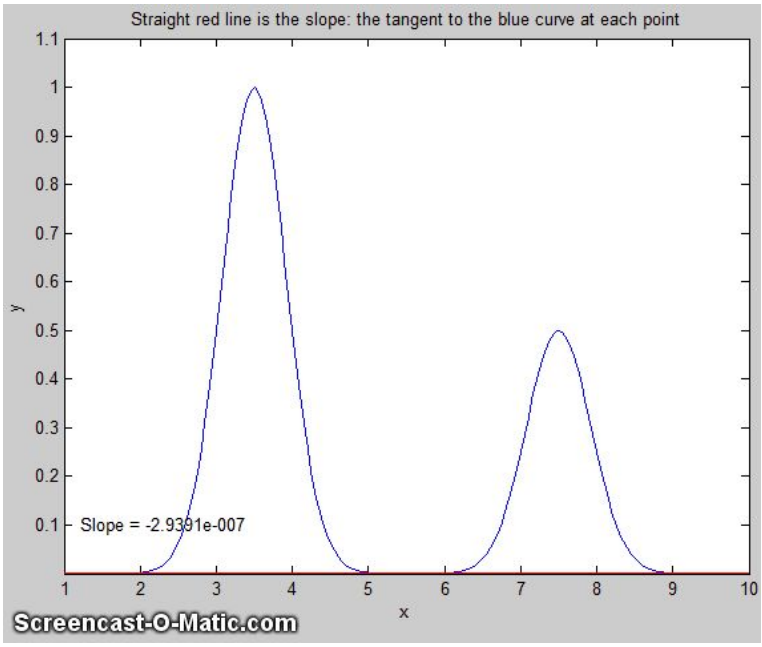
Take Derivatives of a Signal

Differentiate a signal without increasing the noise power. MATLAB®'s function `diff` amplifies the noise, and the resulting inaccuracy worsens for higher derivatives. To fix this problem, use a differentiator filter instead.

Analyze the displacement of a building floor during an earthquake. Find the speed and acceleration as functions of time.

Load the file `earthquake`. The file contains the following variables:

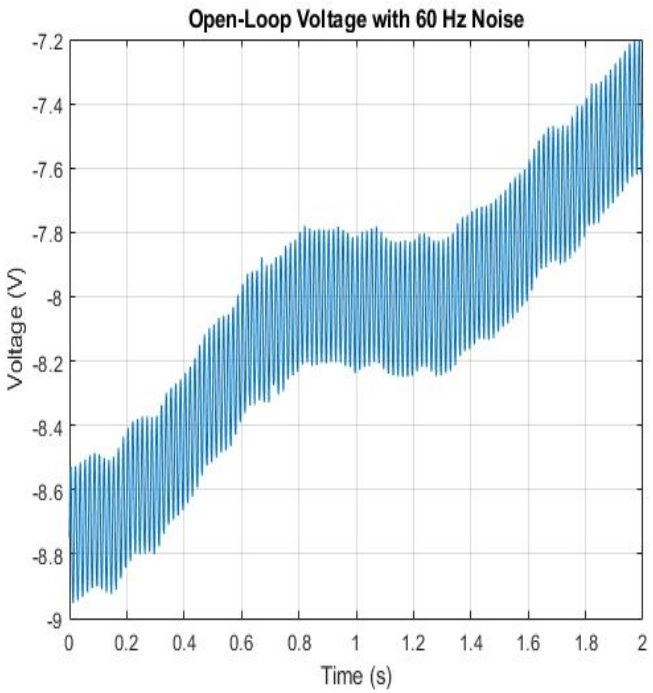
`drift`: Floor displacement, measured in centimeters
`t`: Time, measured in seconds
`Fs`: Sample rate, equal to 1 kHz



Remove the 60 Hz Hum from a Signal

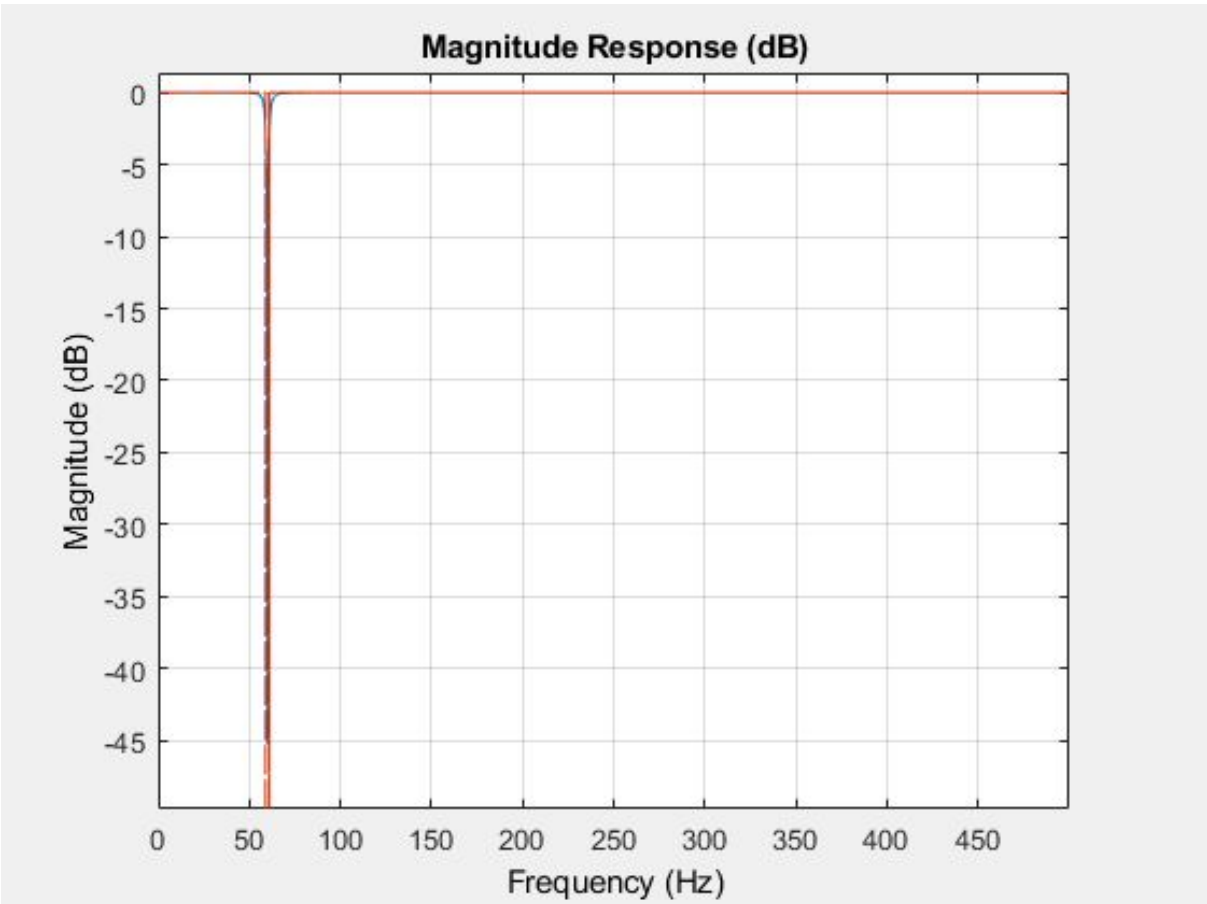
- Alternating current in India and several other countries oscillates at a frequency of 60 Hz. Those oscillations often corrupt measurements and have to be subtracted.
- Study the open-loop voltage across the input of an analog instrument in the presence of 60 Hz power-line noise. The voltage is sampled at 1 kHz.

```
load openloop60hertz, openLoop
= openLoopVoltage;

Fs = 1000;
t = (0:length(openLoop)-1)/Fs;

plot(t,openLoop)
ylabel('Voltage (V)')
xlabel('Time (s)')
title('Open-Loop Voltage with
60 Hz Noise')
grid
```



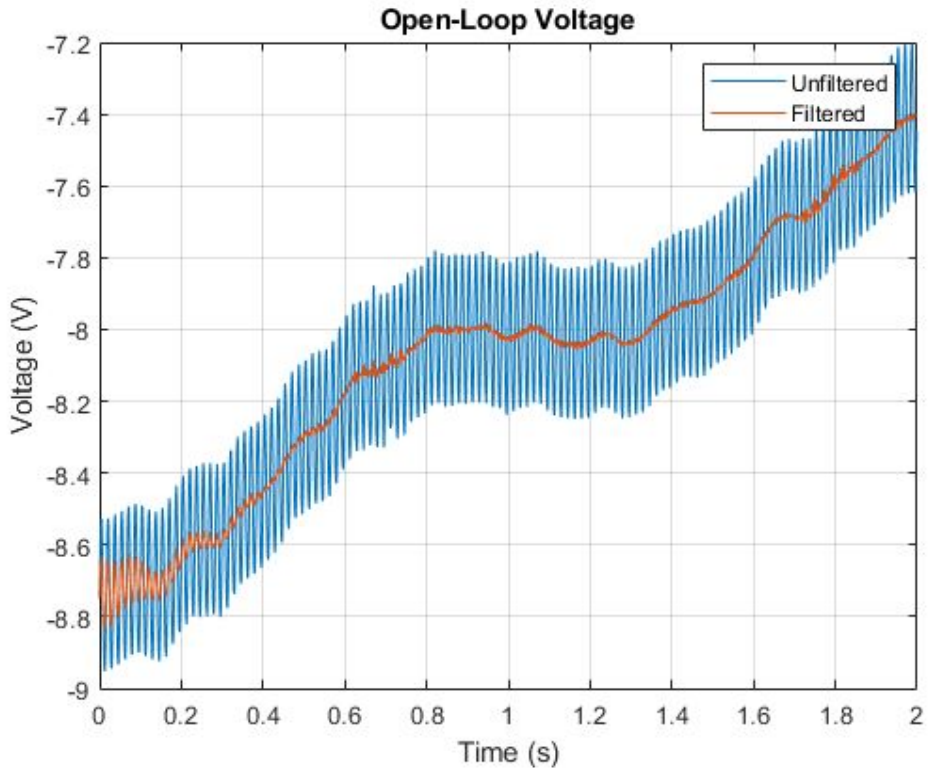
- Eliminate the 60 Hz noise with a Butterworth notch filter.
- Use `designfilt` to design it.
- The width of the notch is defined by the 59 to 61 Hz frequency interval.
- The filter removes at least half the power of the frequency components lying in that range.

```
d = designfilt('bandstopiir','FilterOrder',2, ...

'HalfPowerFrequency1',59,'HalfPowerFrequency2',6
1,

'DesignMethod','butter','SampleRate',Fs);
```



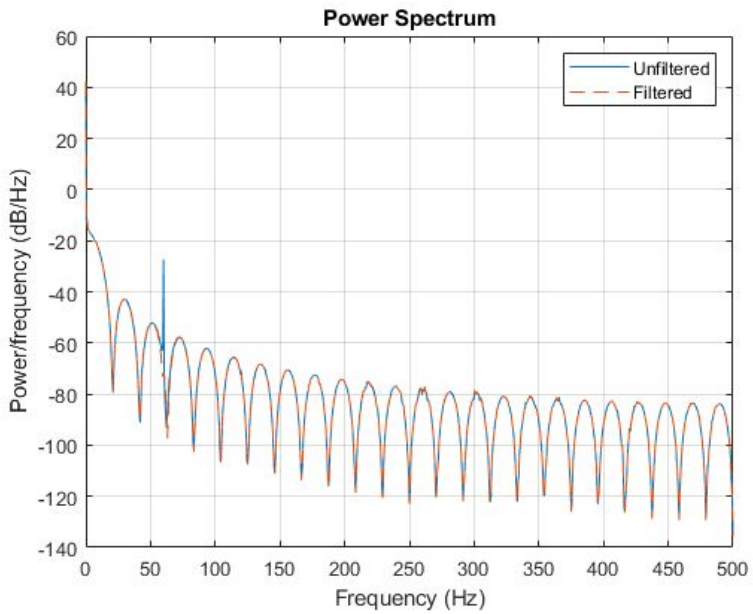
❖ Filter the signal with `filtfilt` to compensate for filter delay. Note how the oscillations decrease significantly.

```
buttLoop = filtfilt(d,openLoop);


plot(t,openLoop,t,buttLoop)
ylabel('Voltage (V)')
xlabel('Time (s)')
title('Open-Loop Voltage')
legend('Unfiltered','Filtered')
grid
```



❖ Use the periodogram to see that the "spike" at 60 Hz has been eliminated.

```
[popen,fopen] = periodogram(openLoop,[],[],Fs);
[pbutt,fbutt] = periodogram(buttLoop,[],[],Fs);

plot(fopen,20*log10(abs(popen)),fbutt,20*log1
0(abs(pbutt)),'--')

ylabel('Power/frequency (dB/Hz)')
xlabel('Frequency (Hz)')
title('Power Spectrum')
legend('Unfiltered','Filtered')
grid
```



Non-GUI Coding

```
%Designing a digital filter using designfilt function
lpfilt =
designfilt('lowpassfir','PassbandFrequency',0.05,...
    'stopbandfrequency',0.06,'PassbandRipple',0.01,...
    'StopbandAttenuation',120,'DesignMethod','kaiserwin');
fvtool(lpfilt);

file1 = 'Ensoniq-ZR-76-01-Dope-77.wav'; %loading the audio
file for testing the filter
[y,Fs1] = audioread(file1); clear file1; %reading the audio
file
%sound(y,Fs1);
totalTime = length(y)./Fs1;
t = (0: totalTime/(length(y)):
totalTime-(totalTime/length(y)));

%plotting input audio signal
subplot(2,1,1)
plot(t, y);
title('Input signal');
xlabel('time');
ylabel('amplitude');

%passing the sampled audio signals through filter
dataIn = y;
dataOut = filter(lpfilt,dataIn); %filter the sampled data
audiowrite('filteredAudio1.wav',dataOut, Fs1); %write the
filtered data to audio file

%plotting filtered audio signal
subplot(2,1,2)
nfft = fft(y);
plot(t, dataOut);
title('filtered signal');
xlabel('time');
ylabel('amp');

%Just practicing how to sample a signal
%stem(t1,x1);
```

2

```
%%
%Derivative of a function in single variable 'x'
syms x;
f=input('Enter a function in x:');
y=diff(f);
disp('The derivative of entered function is:')
y
%%
%Derivative at particular value of x:
i=input('Enter the value of x:');
z=vpa(subs(y,x,i));
disp('The derivative at entered value of x is:')
z
%%
%Second derivative of function in x
y1=diff(f,2)
disp('The second derivative of entered function:')
y1
%%
%Derivative of a constant
const=input('Enter a constant value:');
c=sym(const);
a=diff(c);
disp('The derivative of constant:')
a
```

```
%%
%Derivative of function in two variables s & t(Partial
derivative)
syms s t
b=input('Enter a function in variables s,t:');
c=diff(b,t);
disp('The partial derivative of entered function w.r.t t is:')
c
d=diff(b,s);
disp('The partial derivative of entered function w.r.t s is:')
d
e=diff(b,t,2);
disp('The second derivative of function w.r.t t:')
e
g=diff(b, s, 2);
disp('The second derivative of function w.r.t s:')
g
%%
%SPECIAL CASE:Derivative of function with respect to unspecified
variable:
syms a b t
f = sin(a*t + b);
h=diff(f);
disp('The derivative is taken with respect to alphabet nearest to
''x'':')
h
```

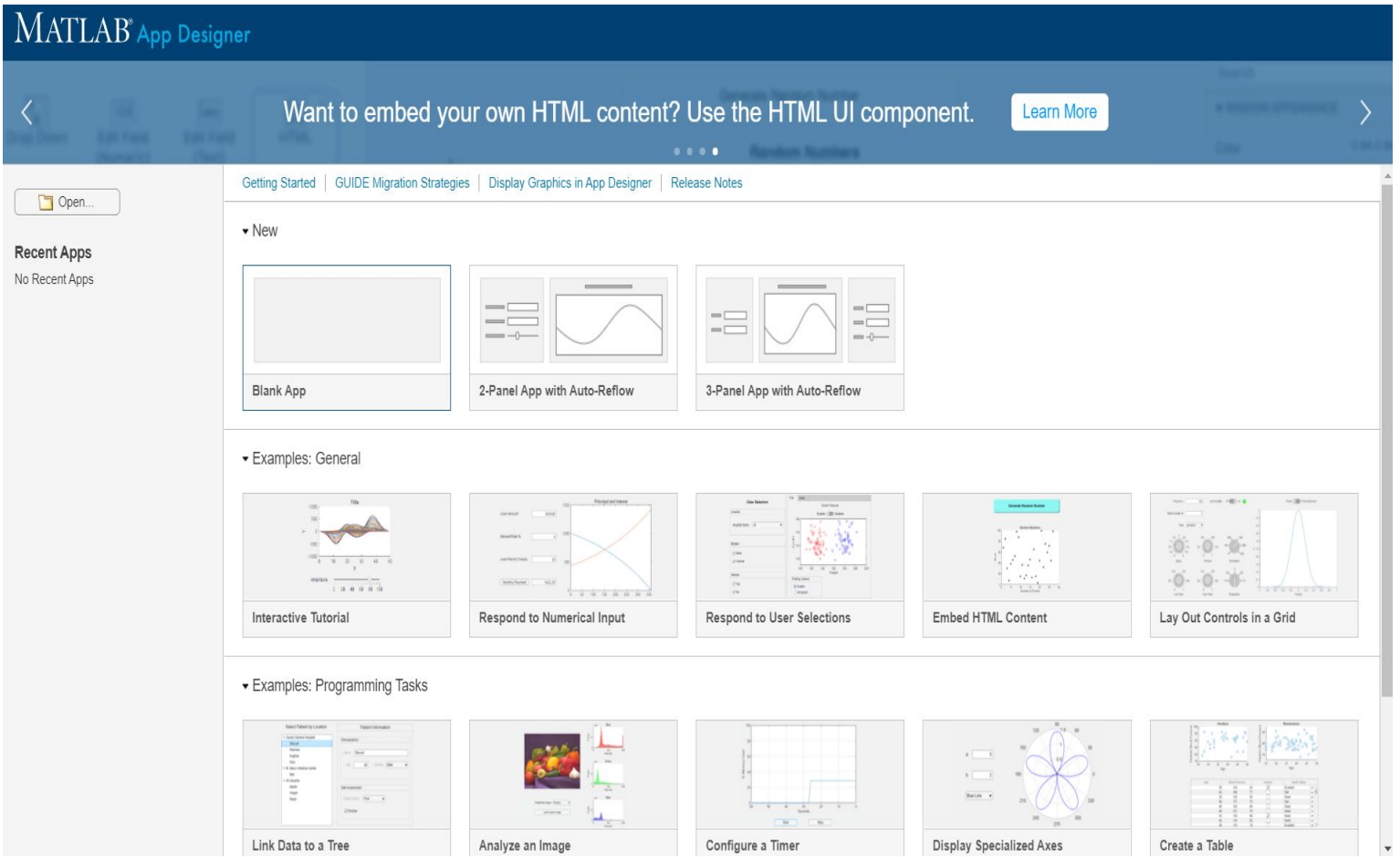
```
%%
%Derivative of exponential function:
syms theta r
p = exp(r*theta);
k=diff(p);
disp('The derivative of exponential function is:')
k
```

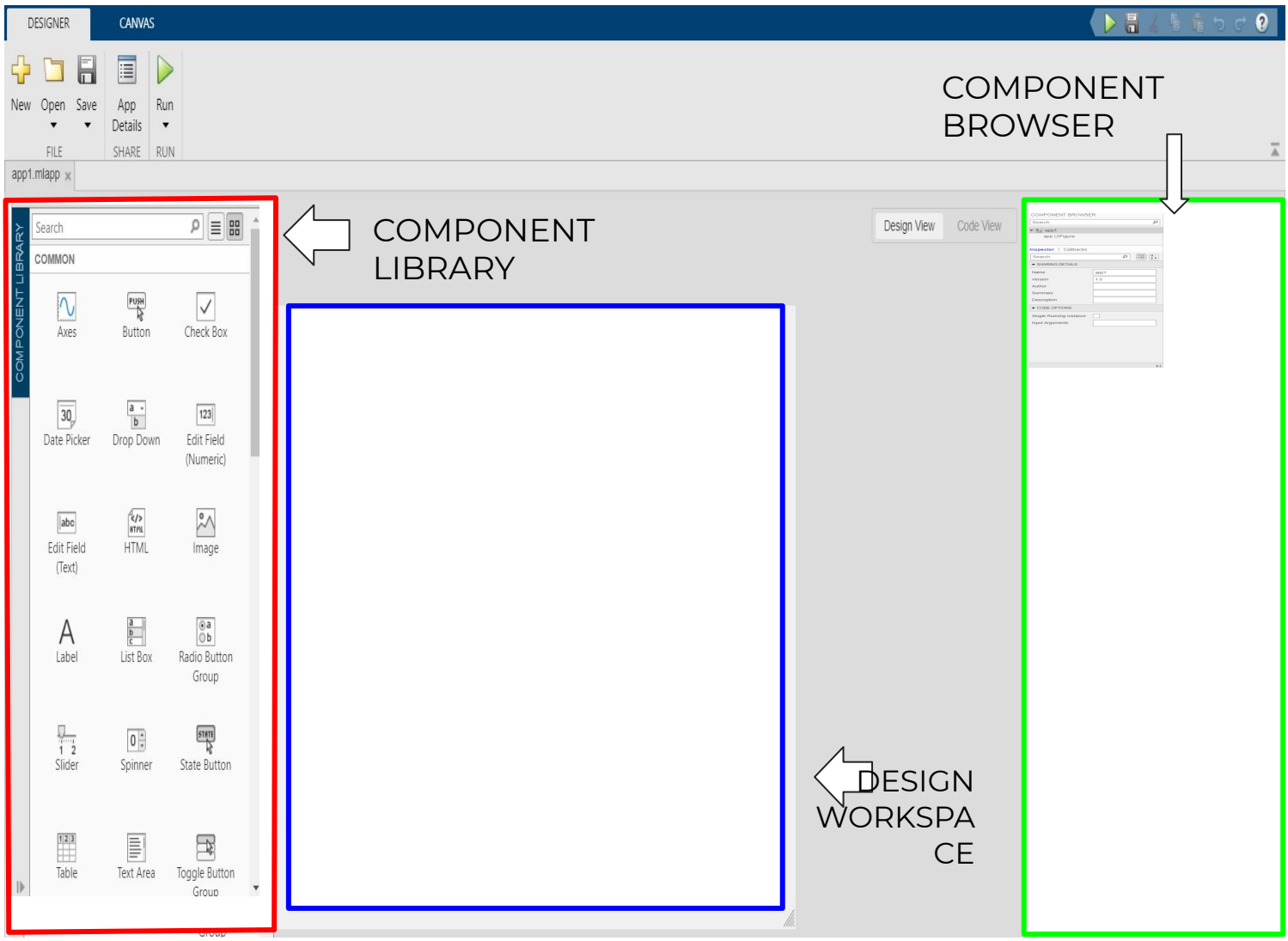
Code for Removing 60Hz Hum

```
load openloop60hertz.mat, openLoop = openLoopVoltage;


Fs = 1000;
t = (0:length(openLoop)-1)/Fs;


subplot(2,2,1);
plot(t,openLoop)
ylabel('Voltage (V)')
xlabel('Time (s)')
title('Open-Loop Voltage with 60 Hz Noise')
grid


d = designfilt('bandstopiir','FilterOrder',2, ...
               'HalfPowerFrequency1',59,'HalfPowerFrequency2',61, ...
               'DesignMethod','butter','SampleRate',Fs);


buttLoop = filtfilt(d,openLoop);


subplot(2,2,2);
plot(t,openLoop,t,buttLoop)
ylabel('Voltage (V)')
xlabel('Time (s)')
title('Open-Loop Voltage')
legend('Unfiltered','Filtered')
grid


[popen,fopen] = periodogram(openLoop,[],[],Fs);
[pbutt,fbutt] = periodogram(buttLoop,[],[],Fs);


subplot(2,2,3);
plot(fopen,20*log10(abs(popen)),fbutt,20*log10(abs(pbutt)),'--')
ylabel('Power/frequency (dB/Hz)')
xlabel('Frequency (Hz)')
title('Power Spectrum')
legend('Unfiltered','Filtered')
grid
```

GETTING STARTED WITH MATLAB APP DESIGNER TOOLBOX

- App Designer lets you create professional apps without having to be a professional software developer.
- Drag and drop visual components to lay out the design of your graphical user interface (GUI) and use the integrated editor to quickly program its behavior.




DESIGNER
CANVAS
New
Open
Save
App Details
Run
FILE
SHARE
RUN
app1.mlapp
COMPONENT BROWSER
COMPONENT LIBRARY
Search
COMMON
Axes
Button
Check Box
Date Picker
Drop Down
Edit Field (Numeric)
Edit Field (Text)
HTML
Image
Label
List Box
Radio Button Group
Slider
Spinner
State Button
Table
Text Area
Toggle Button Group
Design View
Code View
DESIGN WORKSPACE

GUI INTERFACING AND DESIGNING OF THE MATLAB TOPICS :

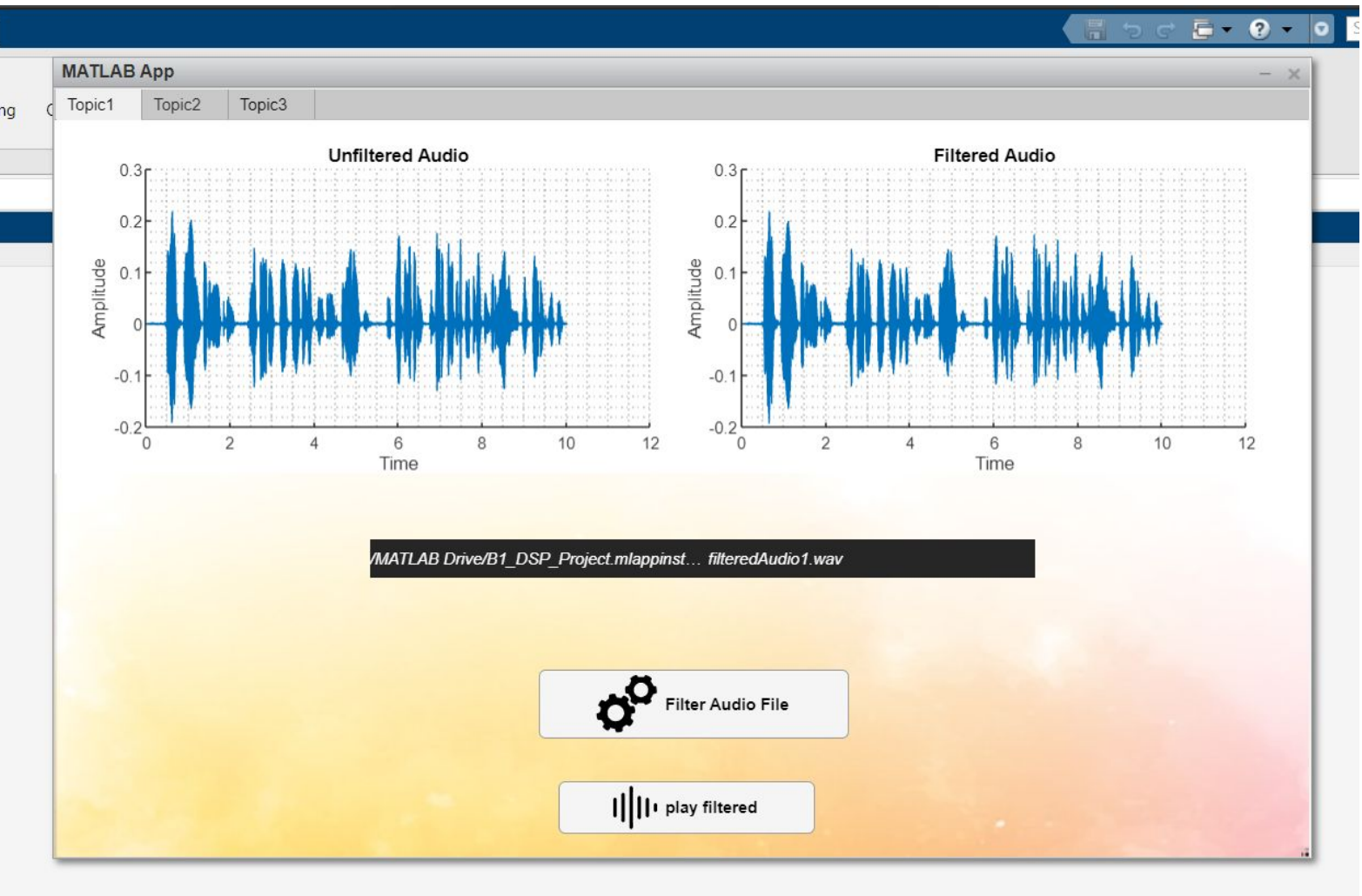
1) Digital Low Pass Filter
2) Derivative Calculator
3) 60 hz noise remover

1) DIGITAL LOW PASS FILTER :

1) In this, we have to input the unfiltered Audio by clicking on the “Filter audio file” Button

2) The Digital Low Pass Filter then filters the audio and gives the final audio output.

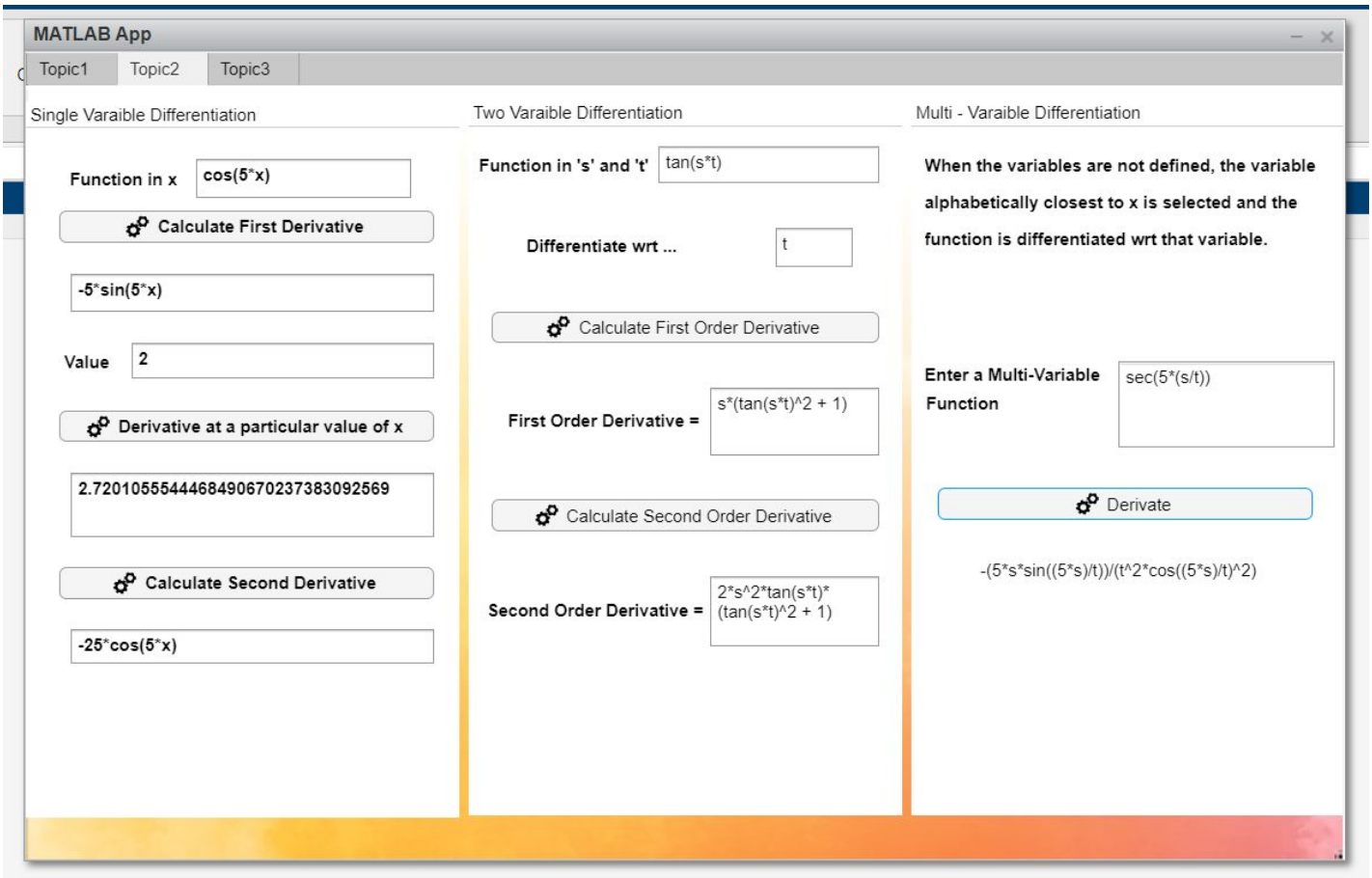
3) The final audio can be played by clicking on the “play filtered” button.



2) DERIVATIVE CALCULATOR :

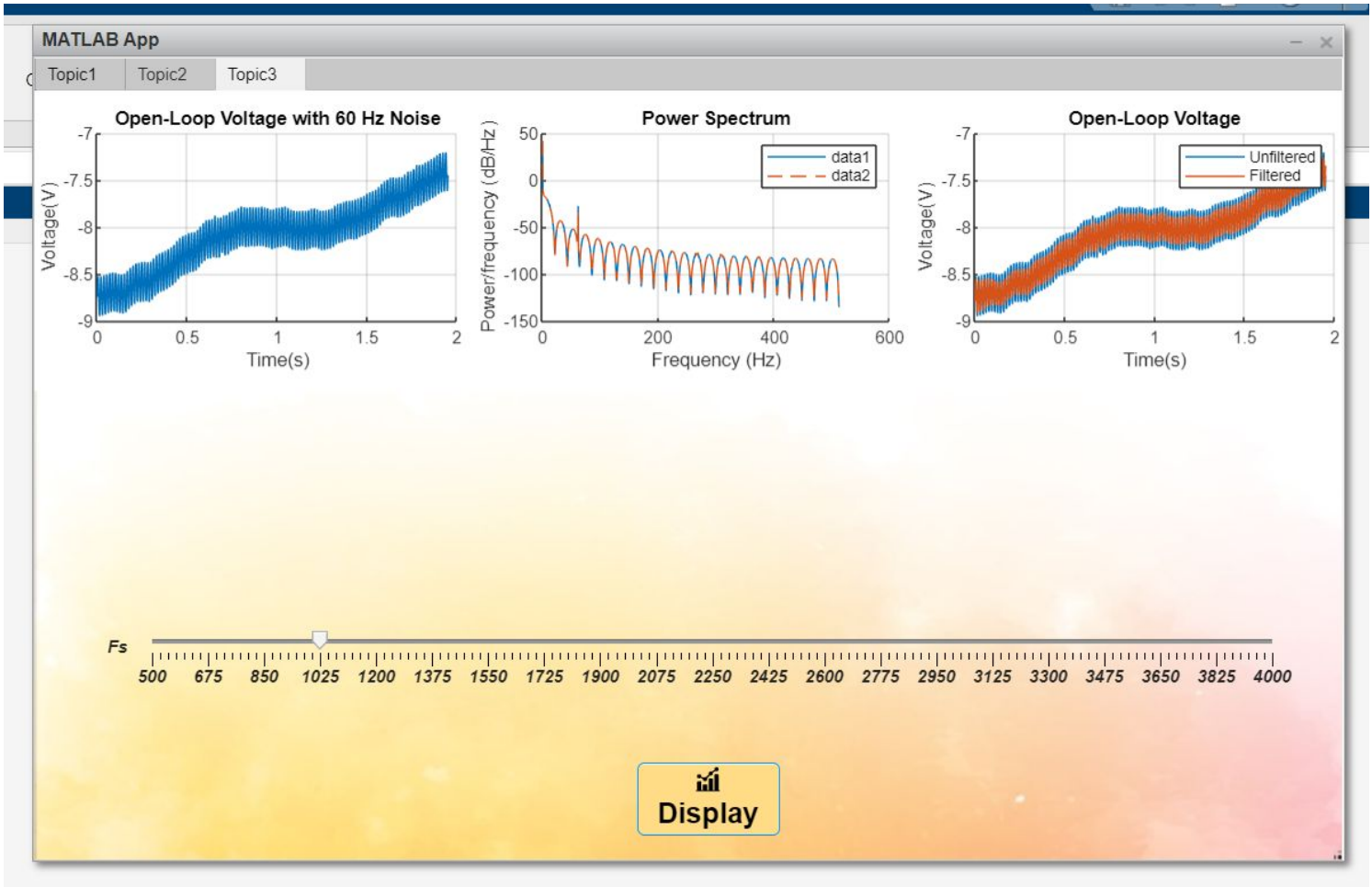
1) The derivative calculator can calculate the first as well as the second order derivatives of the entered function.

2) It can calculate single
Var differentiation,
Two - variable
Differentiation and multi - variable
differentiation.



3) 60 Hz NOISE REMOVER:

Input Sampling Frequency = 1025 Hz



ADVANTAGES OF MATLAB:

- Matlab coder is also present which helps in converting the code in Matlab to any other language like C++, JAVA, etc. which increases the readability and helps the programmers to read the code easily.

- Matlab has its pre-defined libraries and tools which enable the users to build GUI (Graphical User Interface) for their respective programs. This is also a great help for the users who don't have any prior experience and knowledge in Matlab.

- Matlab environment has its predefined functions and libraries which helps the programmers to use it easily. There are various complex mathematical problems which we face in our daily life; this can be executed in Matlab with a single function or code.

- Similarly, Matlab provides a range of toolkits that are used in many fields like aerospace engineering, communications and signal processing, etc.

APPLICATIONS OF MATLAB:

There are THREE important applications of MATLAB which can be considered to be its key features.

Numeric Computation:

Numeric Computation deals with numeric values and relies on vector and matrix calculations. It also makes extensive use of computer graphics, including interactive graphical expositions of numerical algorithms.

Data Analysis and Visualization:

It allows for analyzing and visualizing data in a section. Data from different files can be accessed simultaneously using this software.

Application Development:

Applications and components can be deployed to a variety of platforms. Using Matlab functions, graphical user interfaces can be created very easily. It is possible to generate an application with a built-in interface.

PROJECT CODE LINK AND REFERENCES

TOPIC 1: DIGITAL LOW PASS FILTER

https://in.mathworks.com/help/signal/ug/practical-introduction-to-digital-filter-design.html

DERIVATIVE CALCULATOR

https://in.mathworks.com/help/signal/ug/take-derivatives-of-a-signal.html

60HZ NOISE REMOVER

https://in.mathworks.com/help/signal/ug/remove-the-60-hz-hum-from-a-signal.html

TASK ALLOCATION (PERCENTAGE OF EFFORTS BY EACH MEMBER)

1) MATLAB CODING:

Topic 1: Digital Filter Design

TEETB201 M SHARANYA RAO - 80 %
TEETB207 SAMIKSHA MEHTRE - 80 %
TEETB217 SHUBHAM PANIGRAHI - 100 %

Topic 2: Derivative Calculator

TEETB211 NAINA SAXENA - 100 %
TEETB215 MANSI PACHANKAR - 100 %
TEETB218 PRANAV PATHARKAR - 100 %

Topic 3: 60 Hz Noise remover

TEETB202 AJINKYA MAHAJAN - 90 %
TEETB212 NANDISH NELGI - 80 %

2) DESIGN, CODING AND GUI

TEETB206 SHRUSHTI MANDAOGADE - 100 %
TEETB212 SHREYAS MURKUTE - 100 %
TEETB213 HARSHAL NIKHADE - 100 %

3) DOCS/PPT TEAM

TEETB208 MANDAR MHATRE - 100 %
TEETB204 PARAS MAHAJAN - 100 %
TEETB211 SHIVAM MORE - 80 %
TEETB203 ATHARVA MAHAJAN - 50 %
TEETB271 NEHA SAWALE - 100 %

TASK ALLOCATION (PERCENTAGE OF EFFORTS BY EACH MEMBER)

1) TESTING TEAM:

TEETB216 PALRECHA NANCY MUKESH - 100%
TEETB272 PRASHANT RATHOD - 50 %
TEETB205 KUNAL MANCHARE - 100 %