VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRICAL ENGINEERING

LAB MANUAL

BEEE302P – DIGITAL SIGNAL PROCESSING LAB

FALL 2025 - 26

FACULTY

Dr. IYSWARYA ANNAPOORANI K

SCHOOL OF ELECTRICAL ENGINEERING

Vision of the School

To offer an education in electrical engineering that provides strong fundamental knowledge, skills for employability, cross-disciplinary research and creates leaders who provide technological solutions to societal and industry problems.

Mission of the School

M1: Provide personalized experiential learning in industry sponsored laboratories to prepare students in electrical engineering with strong critical thinking and employability skills.

M2: Foster design thinking, creativity and cross-disciplinary research with highly qualified faculty to create innovators and entrepreneurs in the broad area of electrical engineering.

M3: Collaborate with national and international partners to provide innovative solutions to societal and industry challenges.

PROGRAMME SPECIFIC OUTCOMES (PSOs)

On completion of the B. Tech. (Electrical and Electronics Engineering) programme, graduates will be able to

PSO1 (BL3): Design Electrical and Electronic systems using extensive knowledge of science and engineering.

PSO2 (BL4): Analyze power electronic circuits and power systems considering technical, economic and environmental constraints.

PSO3 (BL3): Apply modern intelligent computational tools to the solution of electrical engineering problems and engage in lifelong learning to adapt to technological advancements.

Program Educational Objectives (PEOs)

PEO 1: Graduates will excel in solving industry problems, succeed as engineering practitioners, innovators, and entrepreneurs, or pursue higher education in electrical engineering and related fields.

PEO 2: Graduates will function with social responsibility, team spirit and environmental awareness and develop products that are reliable, cost effective and safe.

PEO 3: Graduates will demonstrate strong soft skills, uphold ethical standards and professional codes of practice, and continually adapt to technological advancements through lifelong learning.

School of Electrical Engineering

Evaluation Rubrics for Hardware Lab

Lab CAM		Lab FAT		Total
Total Marks	**Weightage**	**Total Marks**	**Weightage**	**CAM + FAT**
100 (Min)	**60%**	**50**	**40%**	**100**

Rubric	Excellent	Good	Satisfactory
Pre-Lab (Circuit Diagram, Background Theory) (3)	**(3)**	**(2)**	**(1)**
	Neat, accurate, and complete; follows standard conventions; all components labeled correctly. Thorough explanation with relevant concepts and clear connections to the experiment; no errors.	**Diagram mostly accurate and complete; minor errors in conventions or labeling. Adequate explanation with most relevant concepts; a few minor errors or unclear connections.**	**Incomplete or inaccurate; significant errors in conventions or missing labels. Incomplete or unclear explanation with limited concepts; contains major errors.**
In-Lab Performance (Connection & Execution) (4)	**(4)**	**(3)**	**(1-2)**
	Correctly sets up and conducts the experiment with no errors; follows all steps systematically. Obtains results with high precision; fully consistent with theoretical predictions or expectations.	**Completes the experiment with minor errors or guidance; follows most steps correctly. Obtains results with moderate precision; some minor discrepancies in theoretical match.**	**Struggles to set up or conduct the experiment; requires significant guidance or corrections. Results are inaccurate or inconsistent with theoretical predictions; lacks proper validation.**

Post Lab (Calculation, Viva-Voce) **(3)**	**(3)**	**(2)**	**(1)**
	All calculations are accurate, complete, and presented clearly with proper units and methods shown. Answers all questions confidently, accurately, and demonstrates deep understanding of the experiment.	**Calculations are mostly accurate; minor errors in presentation, units, or methods. Answers most questions accurately with reasonable understanding; minor gaps in knowledge.**	**Calculations are incomplete, mostly inaccurate, or lack clarity and proper units. Struggles to answer questions or demonstrates limited understanding of the experiment.**

School of Electrical Engineering

Evaluation Rubrics for Software/Programming Lab

Lab CAM		Lab FAT		Total
Total Marks	Weightage	Total Marks	Weightage	CAM + FAT
100 (Min)	60%	50	40%	100

Rubrics	Excellent	Good	Satisfactory
Pre-Lab (Circuit Diagram/Algorithm & Background Theory) (3)	(3)	(2)	(1)
	Neat, accurate, and complete; follows standard conventions; all components labeled correctly. Thorough explanation with relevant concepts and clear connections to the experiment; no errors.	Circuit/Algorithm mostly accurate and complete; minor errors in conventions or labeling. Adequate explanation with most relevant concepts; a few minor errors or unclear connections.	Incomplete or inaccurate; significant errors in conventions or missing labels. Incomplete or unclear explanation with limited concepts; contains major errors.
In-Lab Performance (Circuit/coding /interfacing & Execution) (4)	(4)	(3)	(1-2)
	Circuit/code is optimized and no errors. Thoroughly tests, validates, and documents results accurately and independently.	Circuit/code is functional with minor errors. Tests and validates with minimal assistance; documentation is adequate.	Circuit/code is functional but partially complete and more errors. Testing/validation is incomplete or requires significant help.
Post Lab	(3)	(2)	(1)

(Result Analysis, Viva-Voce) (3)	**Provides a detailed, accurate interpretation of results with insights into improvements and implication. Answers all questions confidently, accurately, and demonstrates deep understanding of the experiment.**	**Provides a clear and correct analysis with minor gaps or limited insights. Answers most questions accurately with reasonable understanding; minor gaps in knowledge.**	**Analysis is basic, with partial interpretation of results. Struggles to answer questions or demonstrates limited understanding of the experiment**

BEEE302P	Digital Signal Processing Lab	L	T	P	C
		0	0	2	1
Pre-requisite	BEEE204L	Syllabus version			
		1.0			

Course Objectives

1. Computation of FFT to communication systems.
2. Design IIR and FIR filters and interfacing of digital signal processor for real world application.

Course Outcomes

On completion of this course, the students will be able to:

1. Design and perform frequency analysis of continuous time and discrete time signals.
2. Design and implement, digital filters with real time constraints.
3. Design a typical digital signal processing system for specific applications in real world.

Indicative Experiments

1	Analysis of continuous time and discrete time signals
2	Convolution of discrete time signals
3	Correlation of discrete time signals
4	Computation of DFT
5	Spectral analysis of signals
6	Design of analog Butterworth filters
7	Design of analog Chebyshev filters
8	Design of an IIR elliptical band pass filter
9	Design of FIR filters using window functions
10	Waveform generation using CC studio of TMS320C6748
11	Computation of convolution using CC studio of TMS320C6748
12	ECG signal smoothening using CC studio of TMS320C6748 for real time applications
Total Laboratory Hours	30 hours

Text Book

John G. Proakis, D. G. Manolakis, Digital Signal Processing Principles, Algorithms and Applications, 2016, 4th edition, Pearson Education

Reference Book

Lawrence R Rabiner and Bernard Gold, Theory and Application of Digital Signal Processing, 2016, Pearson Education

Mode of assessment: Continuous assessment, FAT

Recommended by Board of Studies	19-02-2022		
Approved by Academic Council	No. 65	Date	17-03-2022

COURSE

ARTICULATION

MATRIX CO – PO – PSO

MAPPING

CO No	**Statement**
CO1	Perform frequency analysis of continuous time and discrete time signals.
CO2	Design of digital filters with real time constraints
CO3	Design a typical digital signal processing system for specific applications in real world

CO	**PO**												**PSO**		
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**	**3**
CO1	3	2	1	1	2	-	-	2	2	2	-	1	2	2	2
CO2	3	2	1	1	2	-	-	2	2	2	-	1	2	3	2
CO3	3	2	1	1	2	-	-	2	2	2	-	1	2	3	2

LIST OF EXPERIMENTS

1. **Generation of discrete time sequences**
   - Unit step sequence
   - Unit Impulse sequence
   - Sinusoidal sequence
   - Ramp sequence
   - Exponential sequence

2. **Mathematical operations on signals:**
   - Addition
   - Multiplication
   - Shifting
   - Sampling

3. **Time-domain Analysis of Signals (Radar Signals) & LTI Systems using MATLAB**
   - Linear Convolution and circular Convolution
   - Comparison of linear and circular convolution
   - Auto-Correlation and Cross-Correlation

4. **Frequency-domain Analysis of Signals and LTI Systems**
   - DFT & IDFT – Magnitude and Phase response

5. **ECG signal analysis using SP Tool box**

   IIR Filter – Butterworth (LPF, HPF, BPF & BRF) in MATLAB

   - Chebychev (LPF, BPF)

6. **Speech signal analysis using SP Tool box and FDA Tool box in MATLAB**

   FIR Filter– Windowing (Hamming, Hanning, blackman, rectangular and Kaiser)

GENERATION OF DISCRETE TIME SEQUENCES

Ex. No: 1 **DATE:**

Aim:

To generate the following discrete time sequences using MATLAB

1. Unit step sequence
2. Unit Ramp sequence
3. Impulse sequence
4. Sinusoidal sequence
5. Exponential sequence

Equipments required: MATLAB software

Program:

1. % Unit step response

```
a=input('Enter the desired length of the sequence
=');
b=input('Enter the sampling=');
x=0:a-1;
y=cos(2*pi*b*x);
stem(x,y);
xlabel('time index');
ylabel('Amplitude');
title('generation of unit step sequence');
```

2. % Unit Ramp response

```
a=input('Enter the desired length of the sequence=');
b=input('Enter the sampling=');
x=0:a-1;
y=x;
stem(x,y);
xlabel('time index');
ylabel('Amplitude');
title('generation of unit ramp sequence');
```

3. % %Impulse sequence

```
a=input('Enter the desired length of the sequence
=');
b=input('Enter the sampling=');
x=0:a-1;
y=[cos(2*pi*b) zeros(1,a-1)];
stem(x,y);
xlabel('time index');
ylabel('Amplitude');
disp y;
title('generation of unit impulse sequence')
```

4. %Sinusoidal sequence

```
N=50;
N=0:1:N-1;
a=input('Enter the desired length of the sequence=');
b=input('Enter the sampling=');
n=0:a-1;

x1=cos(pi*n);
subplot(3,2,1),stem(n,x1);
xlabel('n'),ylabel('x1(n)');
title('Sinusoidal sequence');

x2=cos(pi/2*n);
subplot(3,2,2),stem(n,x2);
xlabel('n'),ylabel('x2(n)');
title('Sinusoidal sequence');

x3=cos(pi/4*n);
subplot(3,2,3),stem(n,x3);
xlabel('n'),ylabel('x3(n)');
title('Sinusoidal sequence');

x4=cos(pi/8*n);
subplot(3,2,4),stem(n,x4);
xlabel('n'),ylabel('x4(n)');
title('Sinusoidal sequence');
```

```
x5=cos(pi/16*n);
subplot(3,2,5),stem(n,x5);
xlabel('n'),ylabel('x5(n)');
title('Sinusoidal sequence');


x6=cos(pi/32*n);
subplot(3,2,6),stem(n,x6);
xlabel('n'),ylabel('x6(n)');
title('Sinusoidal sequence');
```

5. %Exponential sequence

```
a=input('Enter the desired length of the sequence=');
b=input('Enter the sampling=');
n=0:a-1;
x2=exp(-n);
subplot(2,2,3),stem(n,x2);
xlabel('n'),ylabel('x2(n)');
title('Exponential sequence');
```

OUTPUT:

RESULT:

MATHEMATICAL OPERATIONS ON SIGNALS

Ex. No: 2 **DATE:**

Aim:

To perform the following mathematical operations on signals using MATLAB

1. Addition
2. Multiplication
3. Sampling
4. Shifting

Equipments required: MATLAB software

Program:

Addition

```
function [y,n] = sigadd(x1,n1,x2,n2)
% implements y(n) = x1(n)+x2(n)
% -----------------------------
% [y,n] = sigadd(x1,n1,x2,n2)
% y = sum sequence over n, which includes n1 and n2
% x1 = first sequence over n1
% x2 = second sequence over n2 (n2 can be different
from n1)
%
n = min(min(n1),min(n2)):max(max(n1),max(n2)); %
duration of y(n)
y1 = zeros(1,length(n)); y2 = y1; % initialization
y1(find((n>=min(n1))&(n<=max(n1))==1))=x1; % x1 with
duration of y
y2(find((n>=min(n2))&(n<=max(n2))==1))=x2; % x2 with
duration of y
```

```
y = y1+y2;
```

```
n1 = -2:20; x1= [1:12,11:-1:1];
n2 = -2:20; x2= [1:12,11:-1:1];
[y,n] = sigadd(x1,n1,x2,n2)
```

Multiplication

```
function [y,n] = sigmult(x1,n1,x2,n2)
% implements y(n) = x1(n)*x2(n)
% -----------------------------
% [y,n] = sigmult(x1,n1,x2,n2)
% y = product sequence over n, which includes n1 and
n2
% x1 = first sequence over n1
% x2 = second sequence over n2 (n2 can be different
from n1)
n = min(min(n1),min(n2)):max(max(n1),max(n2)); %
duration of y(n)
y1 = zeros(1,length(n)); y2 = y1; %
y1(find((n>=min(n1))&(n<=max(n1))==1))=x1; % x1 with
duration of y
y2(find((n>=min(n2))&(n<=max(n2))==1))=x2; % x2 with
duration of y
y = y1 .* y2; % sequence multiplication
n1 = -2:20; x1= [1:12,11:-1:1];
n2 = -2:20; x2= [1:12,11:-1:1];
[y,n] = sigmult(x1,n1,x2,n2)
```

Sampling

```
T=0.1;t=0:0.05/200:T;
x=cos(200*pi*t);
subplot(2,2,1);
plot(t,x);
title('original input signal');
xlabel('time');
ylabel('ampliude');
s1=400;
tn1=0:(1/s1):T;
xn1=cos(200*pi*tn1);
subplot(2,2,2);
stem(tn1,xn1);
title('Sampled signal when fs>2fm');
xlabel('Time index');
ylabel('amplitude');
s2=200;
tn2=0:(1/s2):T;
xn2=cos(200*pi*tn2);
subplot(2,2,3);
stem(tn2,xn2);
title('Sampled signal when fs=2fm');
xlabel('Time index');
ylabel('amplitude');
s3=50;
tn3=0:(1/s3):T;
xn3=cos(200*pi*tn3);
subplot(2,2,4);
stem(tn3,xn3);
```

```
title('Sampled signal when fs<2fm');
xlabel('Time index');
ylabel('amplitude');
```

Shifting

```
function [y,n] = sigshift(x,m,k)
% implements y(n) = x(n-k)
% -------------------------
% [y,n] = sigshift(x,m,k)
%
n = m+k; y = x;
n = -2:20; x= [1:12,11:-1:1];

[x11,n11] = sigshift(x,n,5);

[x12,n12] = sigshift(x,n,-4);

[x1,n1] = sigadd(2*x11,n11,-3*x12,n12);

stem(n1,x1);

 xlabel('n');

ylabel('x(n)');
```

OUTPUT:

RESULT:

TIME-DOMAIN ANALYSIS OF SIGNALS (RADAR SIGNALS) & LTI SYSTEMS USING MATLAB

Ex. No: 3 **DATE:**

Aim:

To generate the following time domain signals using MATLAB

1. Linear convolution
2. Circular convolution
3. Comparison of Linear convolution and Circular convolution
4. Cross Correlation
5. Auto Correlation

Equipments required: MATLAB software

Program:

Linear convolution

```
a=input('Enter the first sequence =');
b=input('Enter the second sequence=');
c=conv(a,b);
M=length(c)-1;
N=0:1:M;
disp('o/p sequence=')
disp(c);
subplot(3,1,1);
stem(a)
subplot(3,1,2);
stem(b)
subplot(3,1,3);
stem(N,c);
xlabel('time index n');
ylabel('Amplitude');
```

Circular Convolution

```
a=input('Enter the first sequence x(n) =');
b=input('Enter the second sequence h(n)=');
n1=length(a);
n2=length(b);
N=max(n1,n2);
x=[a zeros(1,N-n1)];
for  i=1:N
k=i;
for j=1:n2
H(i,j)=x(k)*b(j);
k=k-1;
if(k==0)
    k=N;
end
end
end
y=zeros(1,N);
m=H';
for j=1:N
    for i=1:n2
        y(j)=m(i,j)+y(j)
    end
end
 subplot(3,1,1);
stem(a)
subplot(3,1,2);
stem(b)
subplot(3,1,3);
```

```
stem(y);
xlabel('time index n');
ylabel('Amplitude');
```

Comparison of linear and circular convolution

```
function [yc]=circonv(x,h,N);
Nx=length(x);
Nh=length(h);
x=[x,zeros(1,N-Nx)]
h=[h,zeros(1,N-Nh)]
m=[0:1:N-1];
M=mod(-m,N);
h=h(M+1);
for n=1:1:N
m=n-1;
p=0:1:N-1;
q=mod(p-m,N);
hm=h(q+1);
H(n,:)=hm;
end
yc=x*H';
```

```
clear all;
x=[1,1,1,2,1,1];
h=[1,1,2,1];
Nx=length(x);
```

```
Nh=length(h);
N=max(Nx,Nh);
yc=circonv(x,h,N);
y=conv(x,h);
n=0:1:Nx-1;
subplot(2,2,1)
stem(n,x);
xlabel('n'), ylabel('x(n)')
title ('Input Sequence')
n=0:1:Nh-1;
subplot(2,2,2)
stem(n,h);
xlabel('n'), ylabel('h(n)')
title ('Impulse Sequence')
n=0:1:N-1;
subplot(2,2,3)
stem(n,yc);
xlabel('n'), ylabel('yc(n)')
title ('Output Sequence (circular convolution)')
n=0:1:Nx+Nh-2;
subplot(2,2,4)
stem(n,y);
xlabel('n'), ylabel('y(n)')
title ('Output Sequence (Linear convolution)')
```

Cross correlation

```
x=input('Enter the first sequence=');
y=xcorr(x,x);
figure
```

```
subplot(2,1,1)
stem(x)
subplot(2,1,2)
stem(fliplr(y))
```

Auto correlation

```
x=input('Enter the first sequence=');
h=input('Enter the second sequence=');
y=xcorr(x,h);
figure
subplot(2,1,1)
stem(x)
subplot(2,1,2)
stem(fliplr(y))
```

OUTPUT:

RESULT:

FREQUENCY-DOMAIN ANALYSIS OF SIGNALS AND LTI SYSTEMS

Ex. No: 4 DATE:

Aim:

To generate the following frequency domain signals using MATLAB

1. Discrete Fourier Transform
2. Inverse Discrete Fourier Transform

Equipments required: MATLAB software

Program:

Discrete time Fourier Transform

I method

```
x=input('Enter the sequence=');
h=input('Enter the length of FFT=');
y=fft(x,h)
subplot(3,1,1);
stem(x)
subplot(3,1,2);
stem(h)
subplot(3,1,3);
stem(y);
```

II method

```
N=input('Enter the length of the sequence');
M=input('Enter the length of DFT=');
u=input('Enter the sequence');
U=fft(u,M);
t=0:1:N-1;
```

```
subplot(3,1,1);
stem(t,u);
title('Original time domain sequence');
xlabel('Time index');
ylabel('Ampliude');
subplot(3,1,2);
k=0:1:M-1;
stem(k,abs(U))
title('Magnitude of the dft samples');
xlabel('Frequency index K');
ylabel('magnitude');
subplot(3,1,3);
stem(k,angle(U))
title('Phase of the dft3 samples');
xlabel('Frequency index k');
ylabel('Phase');
disp('Magnitude of DFT');
disp(abs(U));
disp('Phase of DFT');
disp(angle(U));
```

III method:

```
DFT
clc
clear all
x=input('Sequence for N pt dft=');
N=length(x)
```

```
X=zeros(N,1)
for k=0:N-1
    for n=0:N-1
        X(k+1)=X(k+1)+x(n+1)*exp(-j*pi*2*n*k/N)
    end
end
t=0:N-1;
subplot(3,1,1);
stem(t,x);
xlabel('Time(s)');
ylabel('Amplitude');
title('Time domain-input sequence');

subplot(3,1,2);
stem(t,X);
xlabel('Frequency');
ylabel('|X(k)|');
title('Frequency domain-Magnitude response');

subplot(3,1,3);
stem(t,angle(X));
xlabel('Frequency');
ylabel('Phase');
title('Frequency domain-Phase response');
X
angle(X)
```

IDFT

```
N=input('Enter the length of the sequence');

M=input('Enter the length of DFT=');

u=input('Enter the sequence');

U=ifft(u,M);

t=0:1:N-1;

subplot(3,1,1);

stem(t,u);

title('Original frequency domain sequence');

xlabel('Time index');

ylabel('Ampliude');

subplot(3,1,2);

k=0:1:M-1;

stem(k,abs(U))

title('Magnitude of the idft samples');

xlabel('Frequency index K');

ylabel('magnitude');

subplot(3,1,3);

stem(k,angle(U))

title('Phase of the idft samples');

xlabel('Frequency index k');
```

```
ylabel('Phase');

disp('Magnitude of IDFT');

disp(abs(U));

disp('Phase of IDFT');

disp(angle(U));
```

OUTPUT:

RESULT:

ECG SIGNAL ANALYSIS

Ex. No: 5 **DATE:**

Aim:

To analyse the ECG signal from IIR filter using SP tool box.

1. Butterworth filter (LPF, HPF, BPF & BRF)
2. Chebychev (LPF, BPF, BRF)

Equipments required: MATLAB software

Program:

Butterworth Low pass filter

```
clear all;
alphap=0.4
alphas=30;
fp=400;
fs=800;
F=2000;
omp=2*fp/F; oms=2*fs/F;
% To find the cutoff frequency and order of the
filter
[n,wn]=buttord(omp,oms,alphap,alphas)
% System function of the filter
[b,a]=butter(n,wn)
w=0:0.1:pi;
[h,om]=freqz(b,a,w,'whole');
m=abs(h);
an=angle(h);
subplot(2,1,1), plot(om/pi,20*log(m));grid;
ylabel('Gain in dB');
```

```
xlabel('Normalized frequency');
subplot(2,1,2), plot(om/pi,an);grid;
ylabel('Phase in Radians');
xlabel('Normalized frequency');
```

Butterworth Band pass filter

```
clear all;
alphap=2;
alphas=20;
wp=[0.2*pi,0.4*pi];
ws=[0.1*pi,0.5*pi];
% To find the cutoff frequency and order of the
filter
[n,wn]=buttord(wp/pi,ws/pi,alphap,alphas)
% System function of the filter
[b,a]=butter(n,wn)
w=0:0.01:pi;
[h,ph]=freqz(b,a,w);
m=20*log10(abs(h));
an=angle(h);
subplot(2,1,1), plot(ph/pi,m);grid;
ylabel('Gain in dB');
xlabel('Normalized frequency');
subplot(2,1,2), plot(ph/pi,an);grid;
ylabel('Phase in Radians');
xlabel('Normalized frequency');
```

Butterworth high pass filter

```
clear all;
```

```
alphap=0.4
alphas=30;
fp=400;
fs=800;
F=2000;
omp=2*fp/F; oms=2*fs/F;
% To find the cutoff frequency and order of the
filter
[n,wn]=buttord(omp,oms,alphap,alphas)
% System function of the filter
[b,a]=butter(n,wn,'HIGH')
w=0:0.1:pi;
[h,om]=freqz(b,a,w);
m=20*log(abs(h));
an=angle(h);
subplot(2,1,1), plot(om/pi,m);grid;
ylabel('Gain in dB');
xlabel('Normalized frequency');
subplot(2,1,2), plot(om/pi,an);grid;
ylabel('Phase in Radians');
xlabel('Normalized frequency');
```

Butterworth Band reject filter

```
clear all;
alphap=2;
alphas=20;
ws=[0.2*pi,0.4*pi];
wp=[0.1*pi,0.5*pi];
```

```
% To find the cutoff frequency and order of the
filter
[n,wn]=buttord(wp/pi,ws/pi,alphap,alphas)
% System function of the filter
[b,a]=butter(n,wn)
w=0:0.01:pi;
[h,ph]=freqz(b,a,w);
m=20*log10(abs(h));
an=angle(h);
subplot(2,1,1), plot(ph/pi,m);grid;
ylabel('Gain in dB');
xlabel('Normalized frequency');
subplot(2,1,2), plot(ph/pi,an);grid;
ylabel('Phase in Radians');
xlabel('Normalized frequency');
```

Chebyshev Low pass filter

```
clear all;
alphap=1;
alphas=15;
ws=0.2*pi;
wp=0.3*pi;
% To find the cutoff frequency and order of the
filter
[n,wn]=cheb1ord(wp/pi,ws/pi,alphap,alphas)
% System function of the filter
[b,a]=cheby1(n,alphap,wn)
w=0:0.01:pi;
[h,ph]=freqz(b,a,w);
```

```
m=20*log10(abs(h));
an=angle(h);
subplot(2,1,1), plot(ph/pi,m);grid;
ylabel('Gain in dB');
xlabel('Normalized frequency');
subplot(2,1,2), plot(ph/pi,an);grid;
ylabel('Phase in Radians');
xlabel('Normalized frequency');
```

Chebyshev Band pass filter

```
clear all;
alphap=1;
alphas=20;
ws=[0.2*pi,0.4*pi];
wp=[0.1*pi,0.5*pi];
% To find the cutoff frequency and order of the
filter
[n,wn]=buttord(wp/pi,ws/pi,alphap,alphas)
% System function of the filter
[b,a]=cheby1(n,alphap,wn)
w=0:0.01:pi;
[h,ph]=freqz(b,a,w);
m=20*log10(abs(h));
an=angle(h);
subplot(2,1,1), plot(ph/pi,m);grid;
ylabel('Gain in dB');
xlabel('Normalized frequency');
subplot(2,1,2), plot(ph/pi,an);grid;
ylabel('Phase in Radians');
```

```
xlabel('Normalized frequency');
```

Chebyshev Band reject filter

```
clear all;
alphap=2;
alphas=20;
ws=[0.2*pi,0.4*pi];
wp=[0.1*pi,0.5*pi];
% To find the cutoff frequency and order of the
filter
[n,wn]=cheb2ord(wp/pi,ws/pi,alphap,alphas)
% System function of the filter
[b,a]=cheby2(n,alphas,wn,'stop')
w=0:0.01:pi;
[h,ph]=freqz(b,a,w);
m=20*log10(abs(h));
an=angle(h);
subplot(2,1,1), plot(ph/pi,m);grid;
ylabel('Gain in dB');
xlabel('Normalized frequency');
subplot(2,1,2), plot(ph/pi,an);grid;
ylabel('Phase in Radians');
xlabel('Normalized frequency');
```

OUTPUT:

RESULT:

SPEECH SIGNAL ANALYSIS USING SP TOOL BOX AND FDA TOOL BOX IN MATLAB

Ex. No: 6 **DATE:**

Aim:

To analyse the speech signal from FIR filter using SP tool box in MATLAB.

1. FIR Low pass – Rectangular and Hamming
2. FIR High pass –Rectangular and Blackman
3. FIR Band pass Rectangular and hamming
4. FIR Band reject- Rectangular and Hamming
5. FIR Kaiser – Low pass filter

Equipments required: MATLAB software

Program:

FIR Low pass – Rectangular and Hamming

```
clear all
wc=0.5*pi;
N=25;
alpha=(N-1)/2
eps=0.001;
n=0:1:N-1;
hd=sin(wc*(n-alpha+eps))./(pi*(n-alpha+eps));
wr=boxcar(N);
hn=hd.*wr';
w=0:0.01:pi;
h=freqz(hn,1,w);
plot(w/pi,abs(h));
hold on
wh=hamming(N);
hn=hd.*wh';
w=0:0.01:pi;
```

```
h=freqz(hn,1,w);
plot(w/pi,abs(h),'-.'); grid;
xlabel('Normalized Frequency\omega\pi');
ylabel('Magnitud'); hold off
```

FIR High pass –Rectangular and Blackman

```
clear all
wc=0.5*pi;
N=25;
alpha=(N-1)/2
eps=0.001;
n=0:1:N-1;
hd=sin(pi*(n-alpha+eps))-sin(wc*(n-
alpha+eps))./(pi*(n-alpha+eps));
wr=boxcar(N);
hn=hd.*wr';
w=0:0.01:pi;
h=freqz(hn,1,w);
plot(w/pi,abs(h));
hold on
wb=blackman(N);
hn=hd.*wb';
w=0:0.01:pi;
h=freqz(hn,1,w);
plot(w/pi,abs(h),'-.'); grid;
xlabel('Normalized Frequency\omega\pi');
ylabel('Magnitud'); hold off
```

FIR Band pass Rectangular and hamming

```
clear all
wc1=0.25*pi;wc2=0.75*pi;
N=25;
alpha=(N-1)/2
eps=0.001;
n=0:1:N-1;
hd=sin(wc2*(n-alpha+eps))-sin(wc1*(n-
alpha+eps))./(pi*(n-alpha+eps));
wr=boxcar(N);
hn=hd.*wr';
w=0:0.01:pi;
h=freqz(hn,1,w);
plot(w/pi,abs(h));
hold on
wh=hamming(N);
hn=hd.*wh';
w=0:0.01:pi;
h=freqz(hn,1,w);
plot(w/pi,abs(h),'-.'); grid;
xlabel('Normalized Frequency\omega\pi');
ylabel('Magnitud'); hold off
```

FIR Band reject- Rectangular and Hamming

```
clear all
wc1=0.25*pi;wc2=0.75*pi;
N=25;
alpha=(N-1)/2
```

```
eps=0.001;
n=0:1:N-1;
hd=sin(wc1*(n-alpha+eps))-sin(wc2*(n-
alpha+eps))+sin(pi*(n-alpha+eps))./(pi*(n-
alpha+eps));
wr=boxcar(N);
hn=hd.*wr';
w=0:0.01:pi;
h=freqz(hn,1,w);
plot(w/pi,abs(h));
hold on
wh=hamming(N);
hn=hd.*wh';
w=0:0.01:pi;
h=freqz(hn,1,w);
plot(w/pi,abs(h),'-.'); grid;
xlabel('Normalized Frequency\omega\pi');
ylabel('Magnitud'); hold off
```

FIR Kaiser – Low pass filter

```
clear all;
wc=0.5*pi;
N=25;
b=fir1(N,wc/pi, kaiser(N+1, 0.5));
w=0:0.01:pi;
h=freqz(b,1,w);
plot(w/pi,20*log10(abs(h)));
hold on
b=fir1(N,wc/pi, kaiser(N+1, 3.5));
```

```
w=0:0.01:pi;
h=freqz(b,1,w);
plot(w/pi,20*log10(abs(h)));
hold on
b=fir1(N,wc/pi, kaiser(N+1, 8.5));
w=0:0.01:pi;
h=freqz(b,1,w);
plot(w/pi,20*log10(abs(h)));
xlabel('Normalized Frequency\omega\pi');
ylabel('Magnitude in dB'); hold off
```

OUTPUT:

RESULT: