Digital Signal Processing Lab

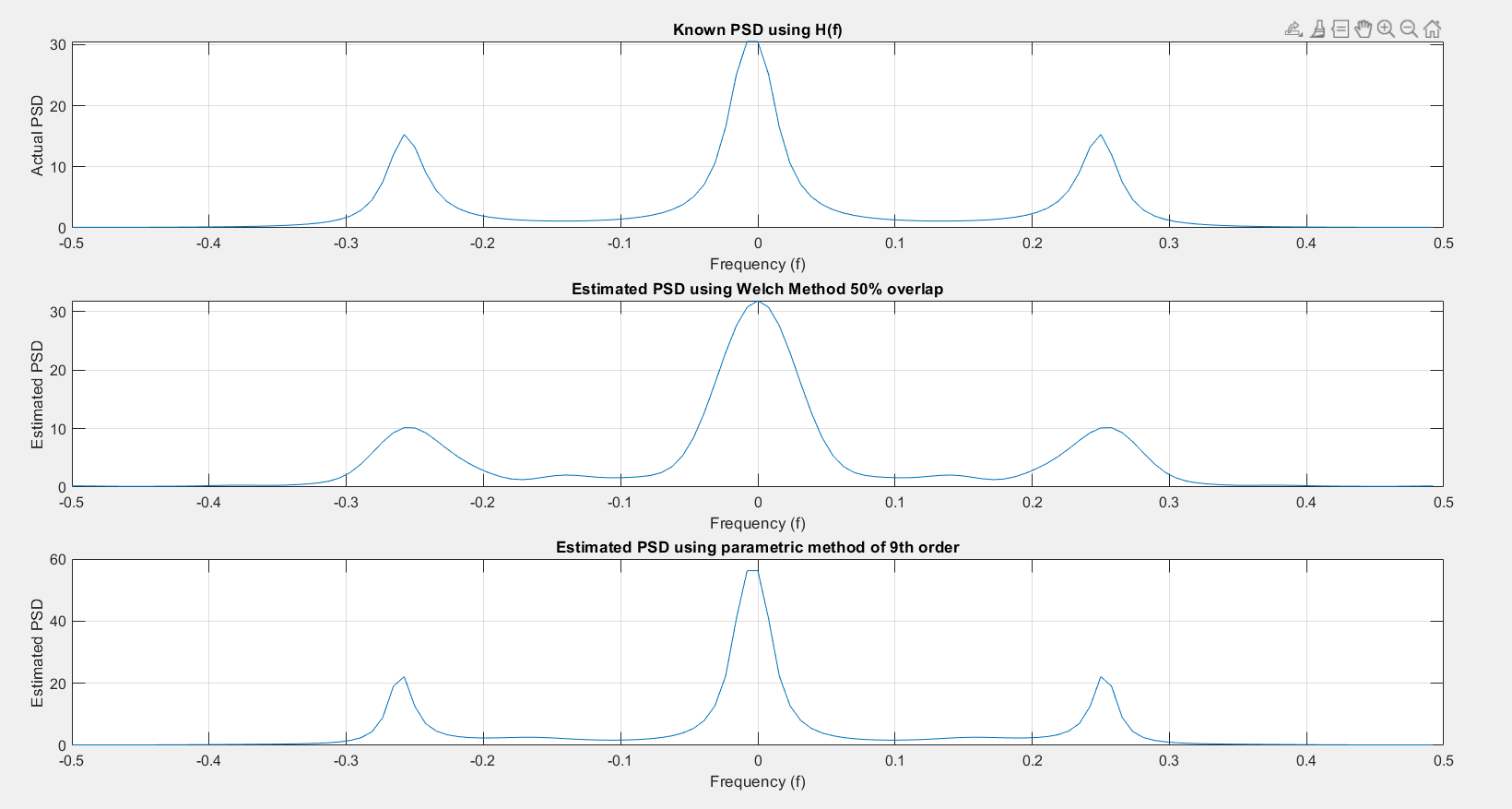
Experiment 4 & 5

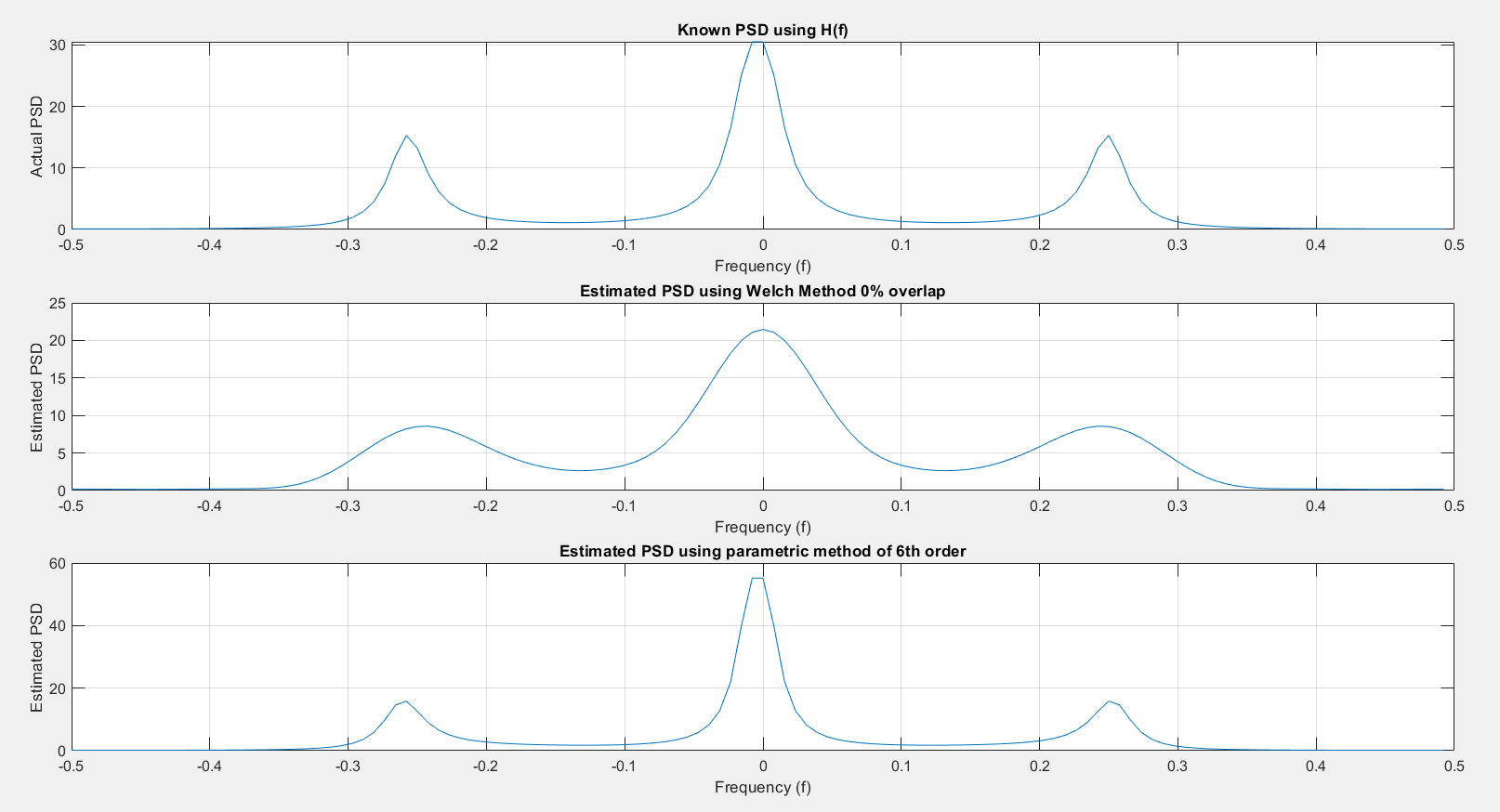
By Hardik Tibrewal (18EC10020)

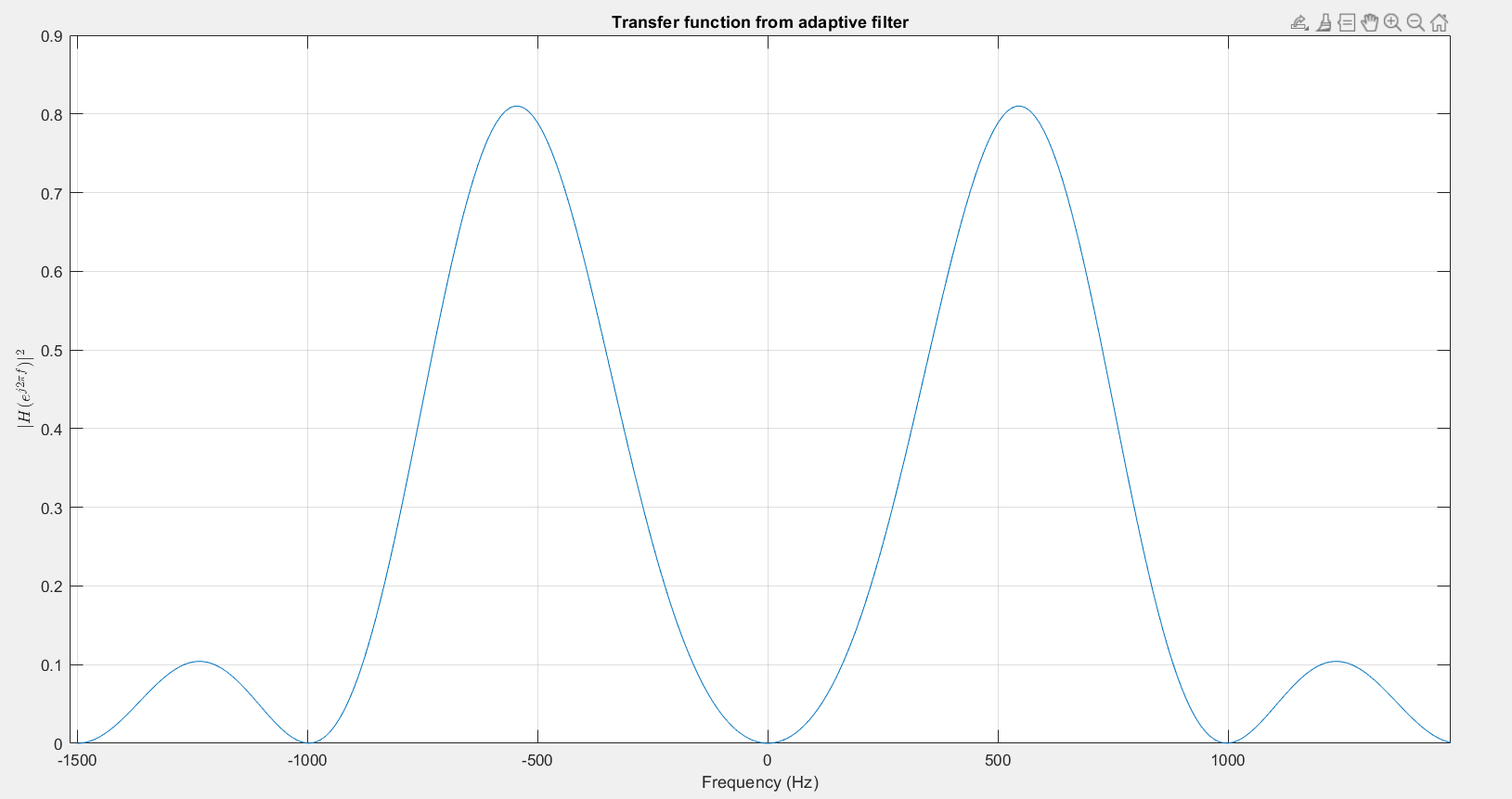
**Aim:**

Power Spectrum Density estimation using Welch, and Yule-Walker methods.

Designing an Adaptive Line Enhancer.

**Plots:** (Denominator of transfer function is 1z0 - 0.9z-1 + 0.81z-2 - 0.729z-3)

****Exp 4:

Exp 5:

**Code:**

Exp 4:

clc

clear all

close all

mean = 0;

std\_dev = 1;

N = 128;

rng('default');

noise = std\_dev.\*randn(N,1) + mean;

denom = 0.9;

A = [1, -denom, denom^2, -denom^3];

X = filter(1, A, noise);

L = 8;

ov\_lap = 0; %fractional overlap

M = round(N/(L\*(1-ov\_lap)+ov\_lap));

D = round(ov\_lap\*M);

window = hamming(M);

U = (1/M)\*sum(window.^2);

X\_divs = zeros(M, L);

l=1; r=M;

for ii = 1:L

X\_divs(:,ii) = X(l:r);

l = r-D+1;

r = l+M-1;

end

f = -0.5:1/N:(0.5-(1/N));

P\_xx = zeros(N, 1);

for ii = 1:L

X\_divs(:,ii) = X\_divs(:,ii).\*window;

P\_xx(:) = P\_xx(:) + (abs(fft(X\_divs(:,ii),N)).^2)/(L\*M\*U);

end

[H, W] = freqz(1,A,N/2);

H = (abs(H).^2).\*std\_dev^2;

H1 = [flip(H); H];

figure();

subplot(311);

plot(f, H1);

grid on;

xlabel('Frequency (f)');ylabel("Actual PSD");title("Known PSD using H(f)");

subplot(312);

plot(f, fftshift(P\_xx));

grid on;

xlabel('Frequency (f)');ylabel("Estimated PSD");title("Estimated PSD using Welch Method "+num2str(100\*ov\_lap)+"% overlap");

%%% Parametric Method %%%

p = 6;

R = xcorr(X)/N;

r = R(N:N+p);

mat = zeros(p,p);

mat2 = -1\*(r(2:p+1));

for ii = 1:p

for jj = 1:p

mat(ii,jj) = r(abs(ii-jj)+1);

end

end

coeff\_a = [1; mat\mat2];

new\_var = sum(coeff\_a.\*r);

[h\_new, w\_new] = freqz(1, coeff\_a(:,1), N/2);

h\_new = (abs(h\_new).^2)\*(new\_var);

l\_new = [flip(h\_new); h\_new];

subplot(313);

plot(f,l\_new);

grid on;

xlabel('Frequency (f)');ylabel("Estimated PSD");title("Estimated PSD using parametric method of "+p+"th order");

Exp 5:

clc

clear all

close all

f = 1000;

fs = 3\*f;

t = 0:1/fs:0.1-1/fs;

N = length(t);

f\_range = -fs/2:fs/N:fs/2-fs/N;

m = 6; mu = 1e-4; epsilon = 1e-3;

x = 2\*sin(2\*pi\*f\*t)';

h = randn(m,1)\*0.01;

x\_n = buffer(x, m, m-1);

x\_n = flip(x\_n, 1);

y = x\_n'\*h;

diff = (x-y);

error = zeros(m, N);

for ii = 1:N

error(:, ii) = x\_n(:, ii)\*diff(ii);

end

update = sum(error,2);

h\_new = h + mu\*update;

change = sum((h\_new-h).^2)/sum(h.^2);

while change >= epsilon

h = h\_new;

y = x\_n'\*h;

diff = (x-y);

for ii = 1:size(x\_n,2)

error(:, ii) = x\_n(:, ii)\*diff(ii);

end

update = sum(error,2);

h\_new = h + mu\*update;

change = sum((h\_new-h).^2)/sum(h.^2);

end

figure();

plot(f\_range, (abs(fft(h\_new, N)).^2));

grid on;

xlabel('Frequency (Hz)'); title('Transfer function from adaptive filter')

ylabel('$|H(e^{j2\pi f})|^2$', 'Interpreter', 'latex');