Digital Signal Processing Lab

Experiment 4b

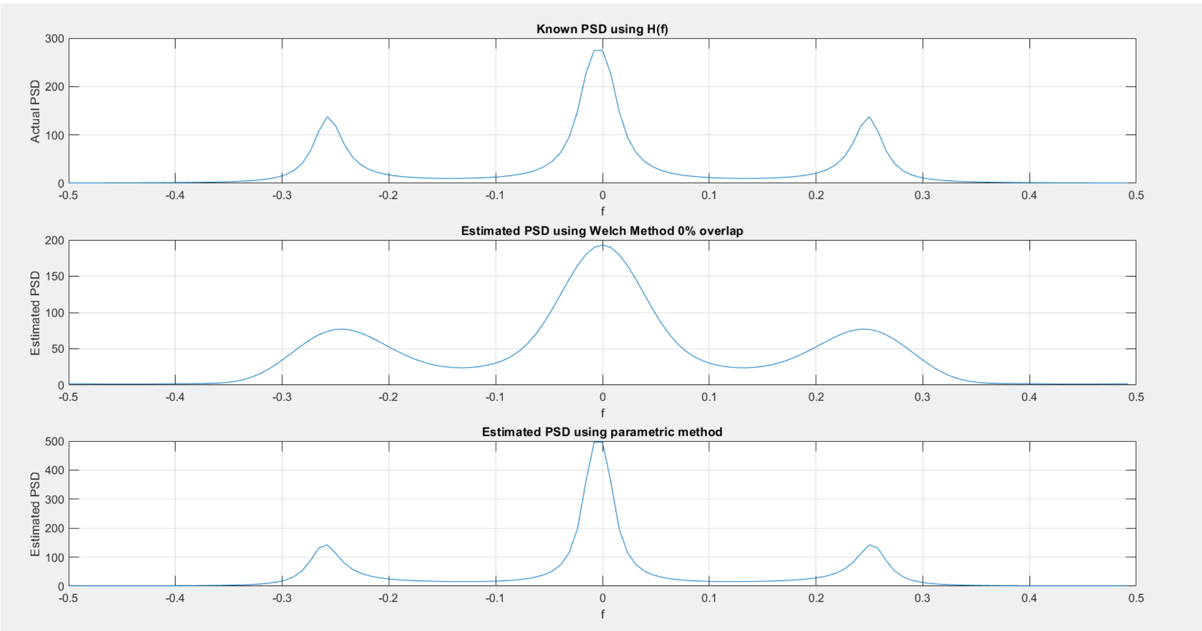
By Hardik Tibrewal (18EC10020)

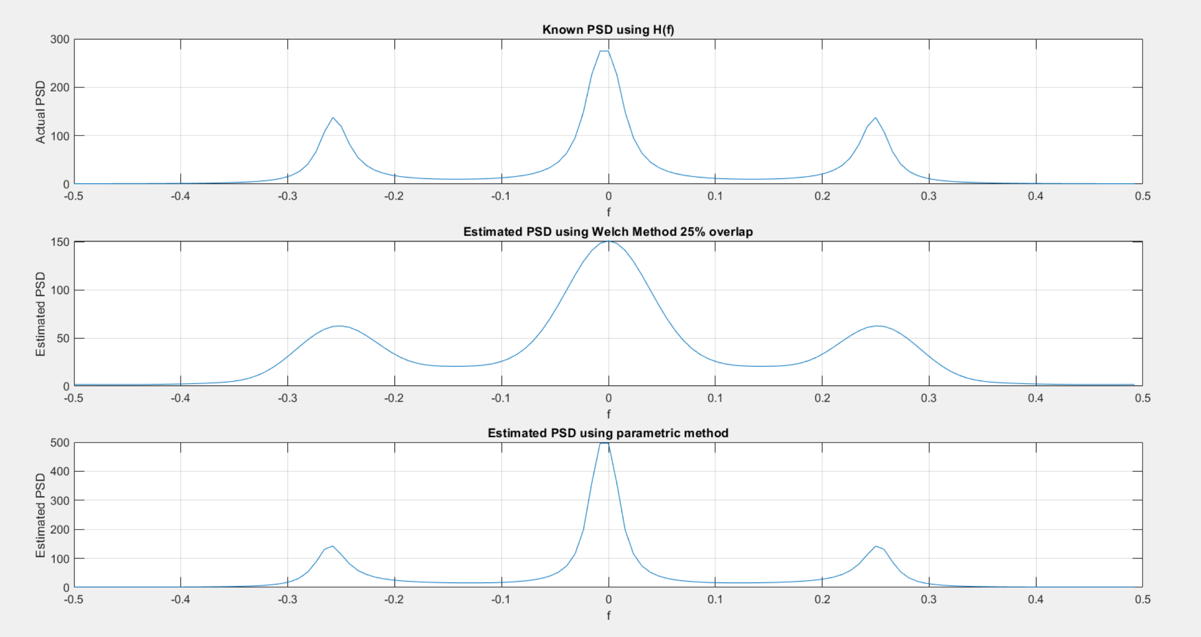
Aim:

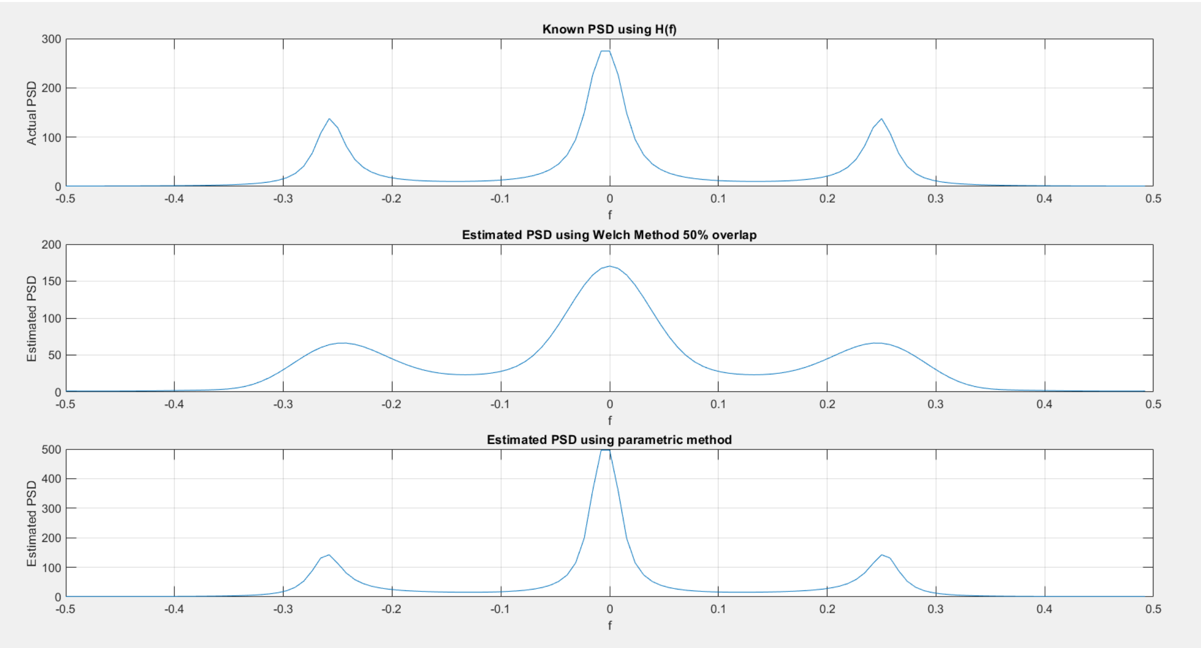
Power Spectrum Estimation using Parametric Method (Yule-Walker AR Model)

Plots:

(Denominator of transfer function is $1z^{0} - 0.9z^{-1} + 0.81z^{-2} - 0.729z^{-3}$)




Known PSD using H(f)
Actual PSD
f
Estimated PSD using Welch Method 25% overlap
Estimated PSD
f
Estimated PSD using parametric method
Estimated PSD
f


Known PSD using H(f)
Actual PSD
f
Estimated PSD using Welch Method 50% overlap
Estimated PSD
f
Estimated PSD using parametric method
Estimated PSD
f

Code:

```
clc
clear all
close all

mean = 0;
std_dev = 3;
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;
for ov_lap = [0, 0.25, 0.5]
    M = 16;
    D = ov_lap;
    L = floor((N-1)/(M*(1-D)));
    num_blocks = ceil((N-1)/(M*(1-D)));
    X_divs = zeros(num_blocks, M);
    end_case = L ~= num_blocks;

    for ii = 1:L
        X_divs(ii,:) = X((1+(ii-1)*floor(M*(1-D))):(M+(ii-1)*floor(M*(1-D))));
    end
    if end_case
        idx = 1+L*floor(M*(1-D));
        X_divs(L+1,:) = [X(idx:end)' zeros(1, M-(N-idx+1))];
    end

    n = 0:1:(M-1);
    hamming = 0.54 - 0.46*cos(2*pi*n/(M-1));

    U = (1/M)*sum(hamming.*hamming);

    P_n = zeros(num_blocks, M);
    for ii = 1:num_blocks
        P_n(ii,:) = X_divs(ii,:).*hamming;
    end

    f = -0.5:1/N:(0.5-(1/N));
    cosine = 0; sine = 0;
    P_f = zeros(num_blocks, N);

    for ii = 1:num_blocks
        for F = 1:length(f)
            cosine = 0; sine = 0;
            for jj = 1:M
                cosine = cosine + cos(2*pi*f(F)*jj)*P_n(ii,jj);
                sine = sine + sin(2*pi*f(F)*jj)*P_n(ii,jj);
            end
            idx = floor((N - length(f))/2)+F;
            P_f(ii, idx) = (cosine^2 + sine^2)/(M*U);
        end
    end

    Pw_f = zeros(1, N);
    for ii = 1:num_blocks
```

```
        Pw_f = Pw_f + P_f(ii,:);
    end

    Pw_f = Pw_f/num_blocks;

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

    figure();
    subplot(311);
    l1 = (abs(H).^2).*std_dev^2;
    l2 = flip(l1);
    l = [l2' l1'];
    plot(f, l);
    grid on;
    xlabel('f');ylabel("Actual PSD");title("Known PSD using H(f)");

    subplot(312);
    plot(f, Pw_f);
    grid on;
    xlabel('f');ylabel("Estimated PSD");title("Estimated PSD using Welch Method "+num2str(D*100)+"%
overlap");

    p = 6;
    r = zeros(p+1);
    for ii = 0:p
        for jj = 1:(N-ii)
            r(ii+1) = r(ii+1) + X(jj)*X(jj+ii);
        end
        r(ii+1) = r(ii+1)/N;
    end

    mat = zeros(p,p);
    mat2 = zeros(1,p);

    for ii = 1:p
        mat2(1,ii) = -r(ii+1);
    end

    for ii = 1:p
        for jj = 1:p
            mat(ii,jj) = r(abs(ii-jj)+1);
        end
    end

    mat_inv = inv(mat);
    coeff_a = mat2*mat_inv;
    coeff_a =  coeff_a';

    new_std_dev = 0;
    for ii = 1:p
        new_std_dev = new_std_dev + coeff_a(ii,1)*r(ii+1);
    end

    new_std_dev = new_std_dev + r(1);
    A_new = ones(p+1);
    for ii = 1:p
        A_new(ii+1) = coeff_a(ii);
    end

    [h_new, w_new] = freqz(1, A_new(:,1), N/2);
```

```
    subplot(313);
    l2_new =  (abs(h_new).^2)*(new_std_dev);
    l1_new = flip(l2_new);
    l_new = [l1_new' l2_new'];
    plot(f,l_new);
    grid on;
    xlabel('f');ylabel("Estimated PSD");title("Estimated PSD using parametric method");
end
```