Digital Signal Processing Lab

Experiment 4 – Power Spectrum Estimation

Utkarsh Patel (18EC30048)

1 Objective

Generation of a random sequence $x(n)$ of known PSD and estimation of the PSD from its finite length via non-parametric Welch method and parametric Yule-Walker AR method

2 Theory

2.1 Generation of signal

A white Gaussian random sequence $r(n)$ is generated with zero mean and variance of σ^2. This signal was passed through a known filter $h(n)$. Let $x(n) = h(n) * r(n)$. The power spectral density of signal $x(n)$ is given as $\left|H(e^{j2\pi f})\right|^2 \sigma^2$ as a function of frequency.

2.2 Welch method

The PSD of a random sequence $x(n)$ is defined as

$$P_{xx}(f) = \lim_{M\to\infty}\left[\frac{1}{2M+1}E\left\{\left|\sum_{n=-M}^{M} x(n)e^{-j2\pi fn}\right|^2\right\}\right]$$

The PSD $P_{xx}(f)$ is estimated in classical periodogram method as

$$\hat{P}_{xx}(f) = \frac{1}{M}\left|\sum_{n=0}^{M-1} x(n)e^{-j2\pi fn}\right|^2$$

It can be shown that the periodogram estimate is asymptotically unbiased however its variance doesn't decay to zero as $M \to \infty$. This drawback can be recovered by segmenting the data $x(n)$ and calculating the average of the periodogram of each segment. The method can be described as follows:

1. The sequence $x(n)$ is divided into L overlapping blocks, with each block having length M with D samples common between two consecutive blocks as
$$x_i(n) = x(n + iD), 0 \le n < M, 0 \le i < L$$
2. PSD of each block is estimated as
$$\hat{P}_{xx}^{(i)}(f) = \frac{1}{MU}\left|\sum_{n=0}^{M-1} x_i(n)w(n)e^{-j2\pi fn}\right|^2, 0 \le i < L$$
where $w(n)$ is the window function of length M and U is a normalization factor for the power in the window function defined as
$$U = \frac{1}{M}\sum_{n=0}^{M-1} w^2(n)$$
3. The Welch spectrum estimate is given as
$$P_{xx}^{W}(f) = \frac{1}{L}\sum_{i=0}^{L-1} \hat{P}_{xx}^{(i)}(f)$$

2.3 Yule-Walker AR method

This method models the source of data generation and estimates PSD by estimating parameters of that model. No windowing occurs in this method. This is particularly useful when only a short data segment is available. The simplest model that is used in practice is a N^{th} order AR model described as by

$$H(z) = \frac{1}{1 + \sum_{k=1}^{N} a_k z^{-k}}$$

The data sequence under observation is assumed to be generated by passing a zero mean white sequence to the above filter.

The steps involved in this method are as follows:

1. The auto-correlation $r(n)$ is computed for the signal $x(n)$.
2. The parameters of N^{th} order AR model can be computed using following relation
$$\begin{bmatrix} r_0 & r_1 & \cdots & r_{N-1} \\ r_1 & r_0 & \cdots & r_{N-2} \\ \vdots & \vdots & \vdots & \vdots \\ r_{N-1} & r_{N-2} & \cdots & r_0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix} = - \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_N \end{bmatrix}$$
3. The variance of white Gaussian noise (which is assumed to generate this random sequence $x(n)$) is estimated as $\sigma^2 = r(0) + \sum_{k=1}^{N} a_k r(k)$
4. The PSD estimated using Yule-Walker method is given as
$$P_{xx}^{YW}(f) = \frac{\sigma^2}{\left|1 + \sum_{k=1}^{N} a_k e^{-j2\pi fk}\right|}$$

3 Results

3.1 Part A: Welch Method

A white Gaussian random sequence with zero mean and variance of σ^2 is generated, and is passed through the filter

$$H(z) = \frac{1}{(1 - 0.9z^{-1})(1 - 0.9jz^{-1})(1 + 0.9jz^{-1})}$$

to get the sequence $x(n)$. The PSD of $x(n)$ is estimated using Welch method.

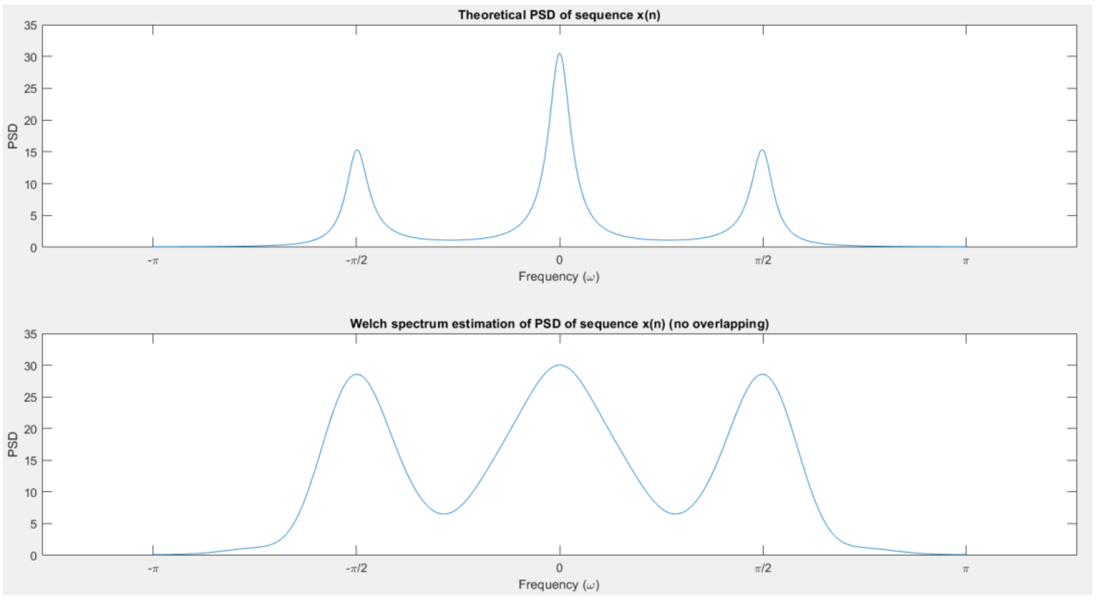

Fig. 1. *Welch Method Estimation with Zero Overlapping*

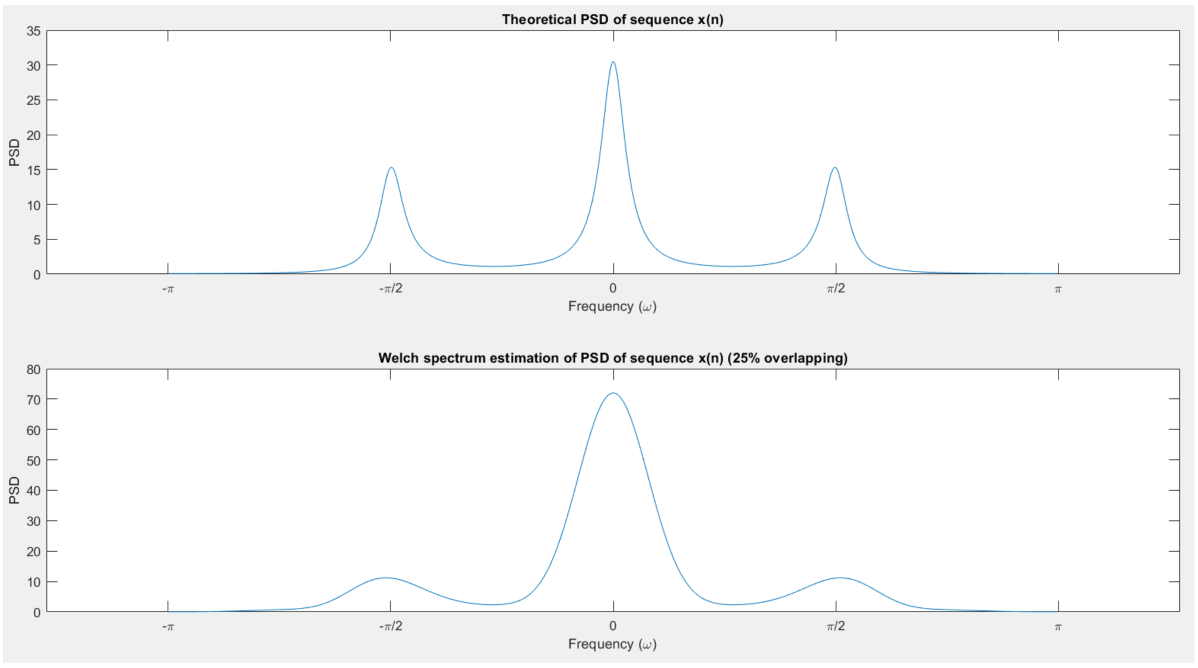

Fig. 2. *Welch Method Estimation with 25% Overlapping*

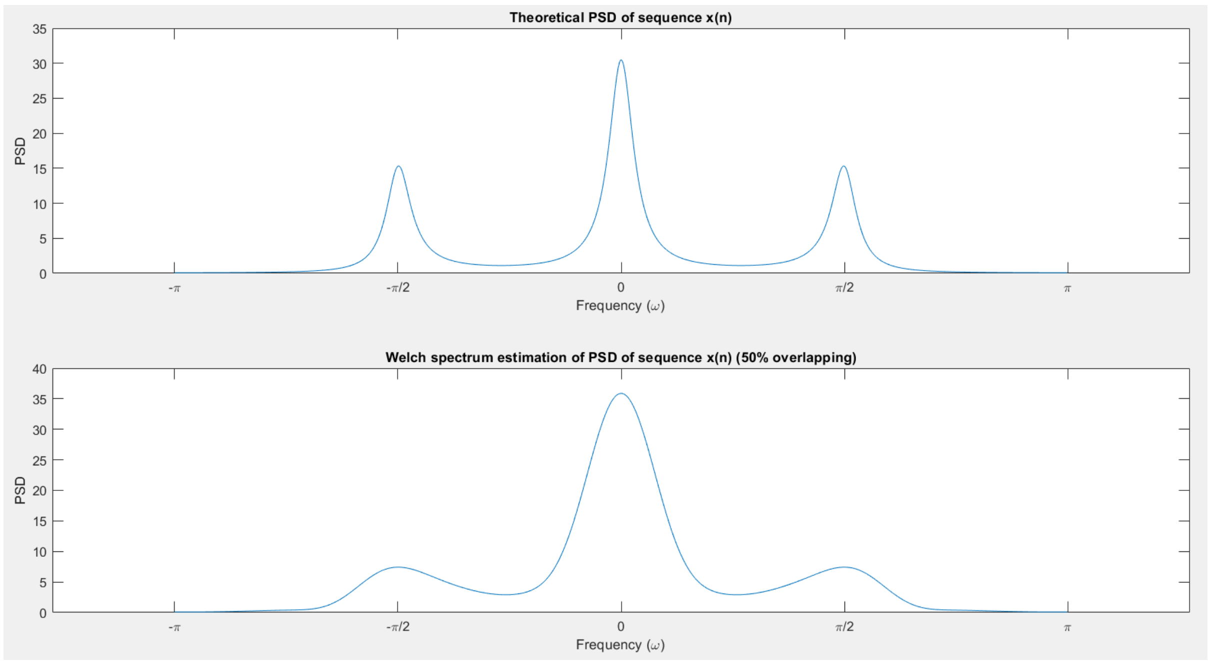

Fig. 3. *Welch Method Estimation with 50% Overlapping*

The same procedure is performed for transfer function given as

$$H(z) = \frac{1}{(1-0.9z^{-1})(1-0.8jz^{-1})}$$

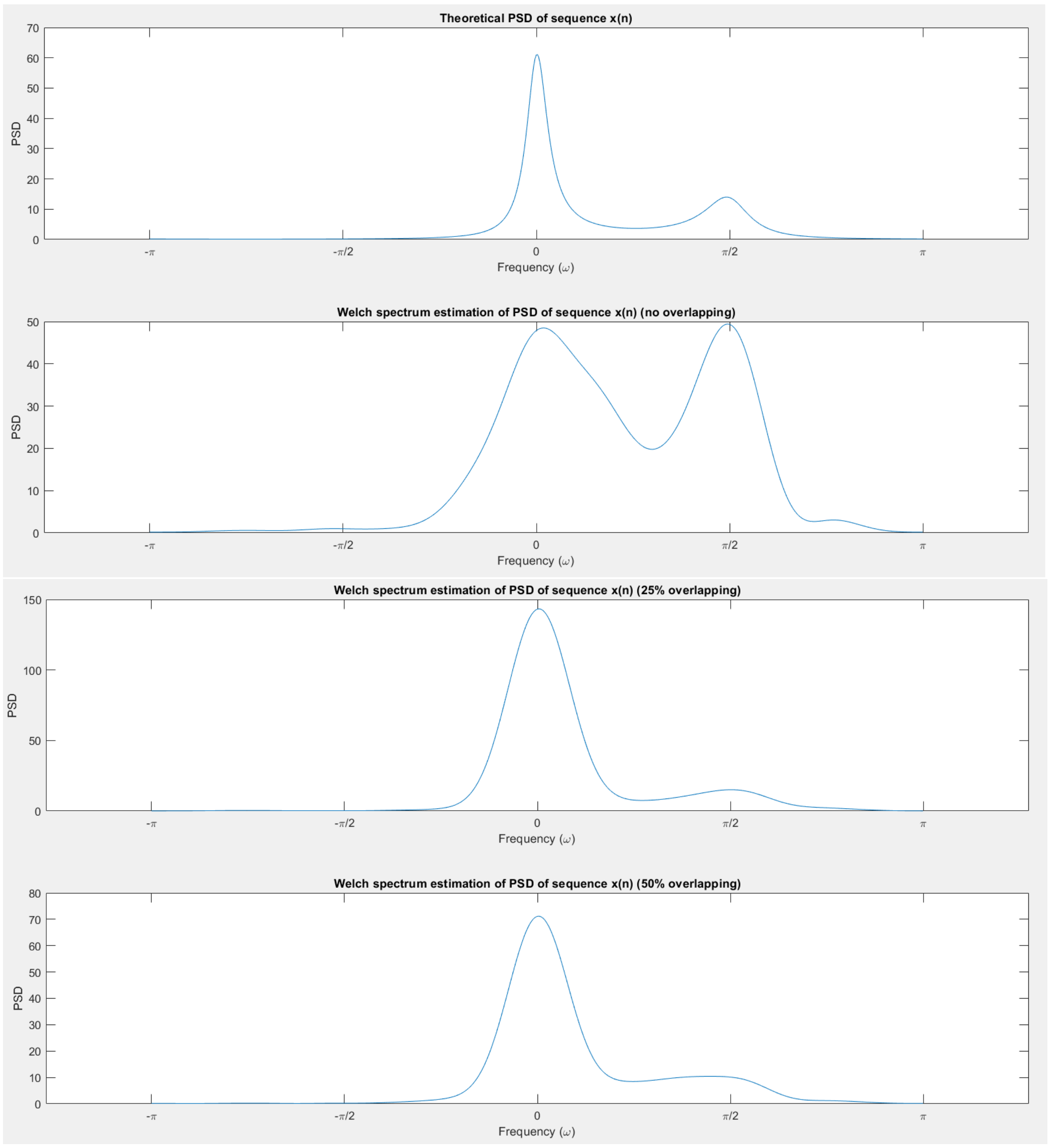

Fig. 4. *Welch Method Estimation for another transfer function*

3.2 Part B: Yule-Walker AR Method

A white Gaussian random sequence with zero mean and variance of σ^2 is generated, and is passed through the filter

$$H(z) = \frac{1}{(1-0.9z^{-1})(1-0.9jz^{-1})(1+0.9jz^{-1})}$$

to get the sequence $x(n)$. The PSD of $x(n)$ is estimated using Yule-Walker method.

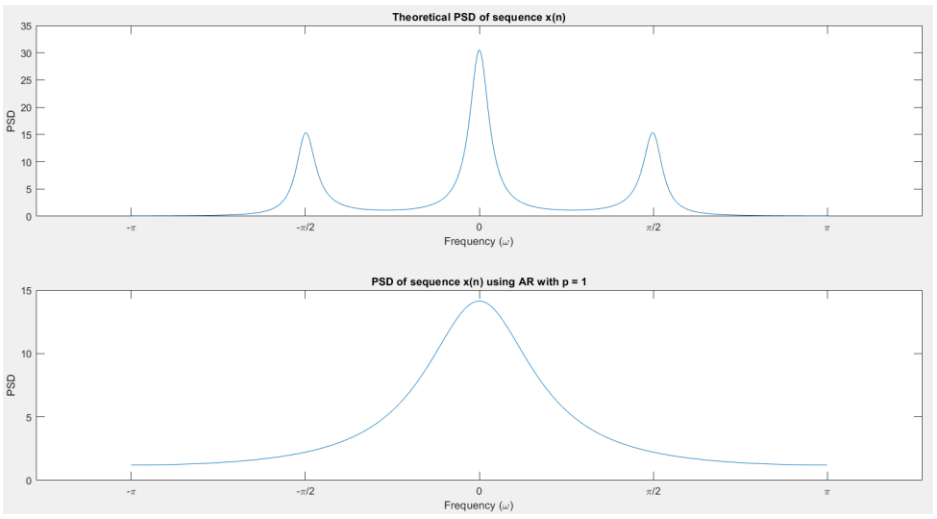

Fig. 5. *Yule-Walker AR PSD estimation with first order model*

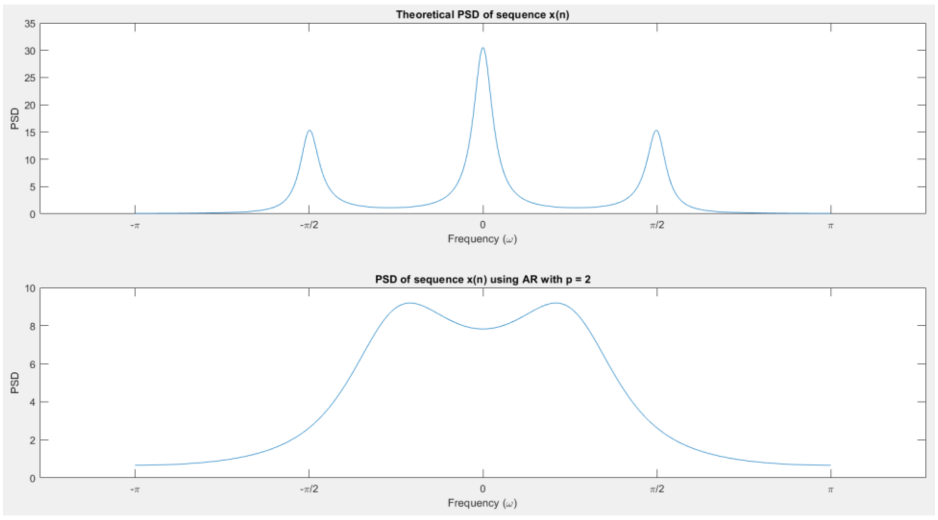

Fig. 6. *Yule-Walker AR PSD estimation with second order model*

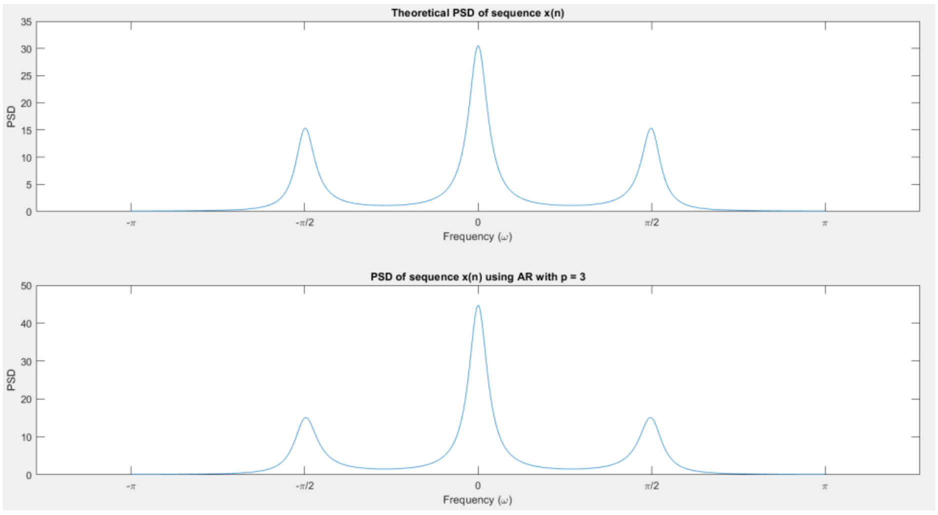

Fig. 7. *Yule-Walker AR PSD estimation with third order model*

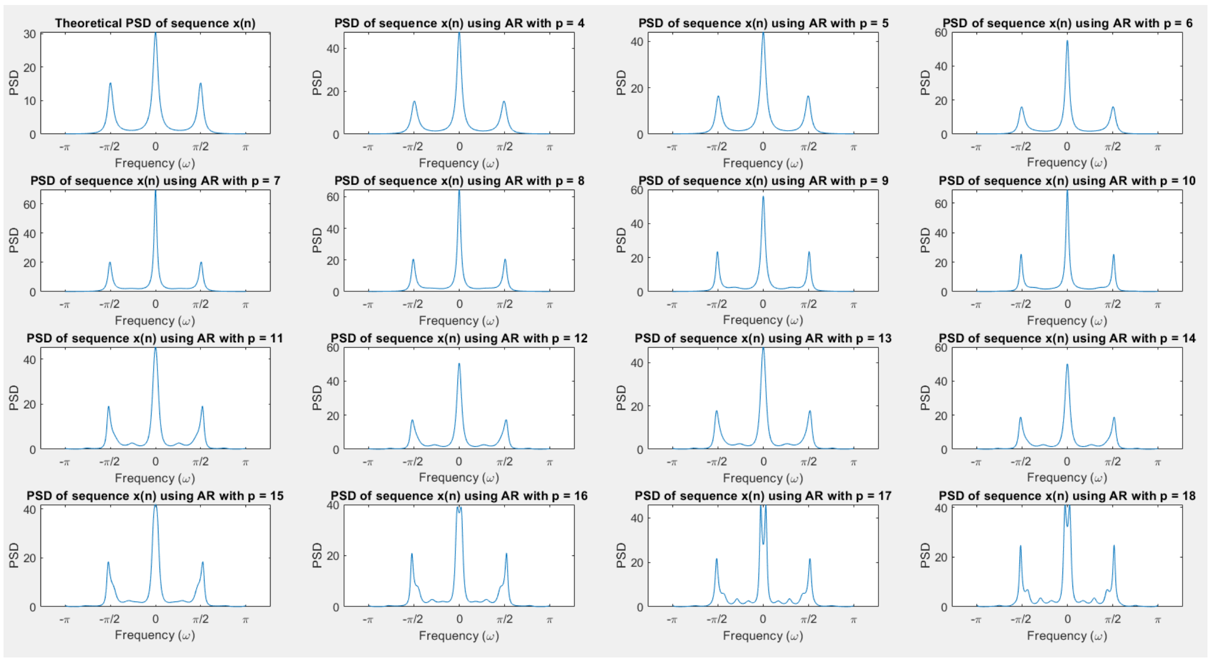

Fig. 8. *Yule-Walker AR PSD estimation with order of model in range [4, 18]*

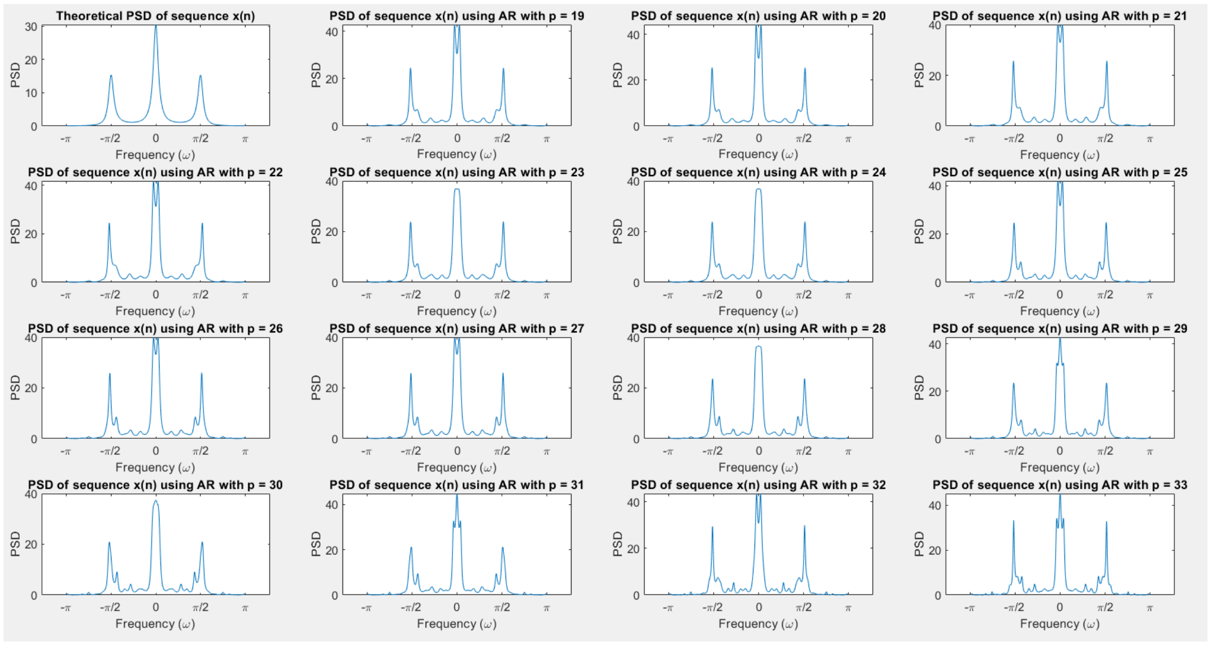

Fig. 9. *Yule-Walker AR PSD estimation with order of model in range [19, 33]*

4 Discussion

- In this experiment, power spectral density (PSD) of a random sequence (with known PSD) is estimated using non-parametric Welch method and parametric Yule-Walker AR modelling.

- Considering a random sequence $x(n)$, its PSD is given as

$$P_{xx}(f) = \lim_{M\to\infty}\left[\frac{1}{2M+1}E\left\{\left|\sum_{n=-M}^{M} x(n)e^{-j2\pi fn}\right|^2\right\}\right]$$

- Due to the limit in the above equation, it becomes practically impossible to compute the PSD with the above relation. However, the PSD can be estimated using the relation

$$\hat{P}_{xx}(f) = \frac{1}{M}\left|\sum_{n=0}^{M-1} x(n)e^{-j2\pi fn}\right|^2$$

  by taking M sufficiently large enough. Through the bias converges to zero as M is scaled up to infinity, the variance of this estimation doesn't converge to zero.

- In Welch method, we use this approximation. The sequence $x(n)$ is split into overlapping parts, and its PSD is estimated following the procedure as given in Section 2.2.

- From Fig. 1, it can be observed that the central lobe peak is same as the theoretical PSD, however the side lobe peaks are much more than their theoretical counterpart.

- The overlapping between the components is increased in subsequent simulation where it can be observed from Fig. 2 and 3, that as the overlapping is increased, the estimated PSD converges to the theoretical PSD.

- Yule-Walker method is a completely different approach to estimate PSD of the random sequence. In this framework, the random sequence is assumed to be generated by a system which is a N^{th} order AR model which has been excited by a white Gaussian noise with zero mean and variance of σ^2. The parameters of the AR model and variance of white Gaussian excitation is estimated with techniques as described in Section 2.3.

- The simulation was caried out by choosing another transfer function. The same results were observed in this case as well (Fig. 4).

- From Fig. 5 and 6, it can be observed that first and second order AR modelling produces the PSD distribution which is entirely different from the theoretical counterpart.

- However, in Fig. 7, when the order of model is increased to 3, appreciable plots are generated which resembles the theoretical plot. This is expected because the transfer function used in this case

$$H(z) = \frac{1}{(1-0.9z^{-1})(1-0.9jz^{-1})(1+0.9jz^{-1})}$$

  has only 3 poles.

- In subsequent simulation, effect of order of AR modelling is observed on the estimated PSD of the random sequence. From Fig. 8 and 9, it can be observed that for order of AR model between 4 and 8, the plots match with the theoretical plot. However, when order is increased beyond 8, the PSD plot starts getting distorted.

Appendix

MATLAB Codes

1 Welch method

```
% Author: Utkarsh Patel (18EC30048)
% Experiment 4 - Part 1

N = 128; % length of white noise
L = 8;   % components used
D = 0;   % overlapping within component
% D = f * (N / L), where f is percentage overlapping
M = N / L;% length of each component without overlapping
w = -1 * pi: 0.01: pi;

mean = 0; % mean of noise
var = 1;  % variance of noise

h = TF(N); % transfer function of system

h_f = var * abs(freqz(h, 1, w)).^2; % expected PSD of x(n)
subplot(2, 1, 1);
plot(w, h_f);
xticks([-pi -pi/2 0 pi/2 pi]);
xticklabels({'-\pi','-\pi/2','0','\pi/2','\pi'});
title('Theoretical PSD of sequence x(n)');
xlabel('Frequency (\omega)');
ylabel('PSD');

rng('default');
r = normrnd(mean,sqrt(var),[1 N]); % white noise
% Generating output sequence x(n)
x = conv(h, r);
x = x(1: N);

win = ham(M); % hamming window
U = 0; % power of the window (to be used in normalization)
for i = 1: M
    U = U + win(i) * win(i);
end
U = U / M;

Pff = 0; % Overall PSD

for i = 0: L - 1
    x_ = zeros(1, M);
    for j = 0: M - 1
        x_(j + 1) = x(j + 1 + i * D) * win(j + 1);
    end
    Pff_i = abs(freqz(x_, 1, w)).^2;
    Pff_i = Pff_i / (M * U); % PSD of individual components
    Pff = Pff + Pff_i / L;
end

subplot(2, 1, 2);
plot(w, Pff);
xticks([-pi -pi/2 0 pi/2 pi]);
xticklabels({'-\pi','-\pi/2','0','\pi/2','\pi'});
title('Welch spectrum estimation of PSD of sequence x(n) (50% overlapping)');
xlabel('Frequency (\omega)');
ylabel('PSD');


function win = ham(M)
    % hamming window of length M
    win = zeros(1, M);
    for i = 0: M - 1
        win(i + 1) = 0.54 - 0.46 * cos(2 * pi * (i) / (M - 1));
    end
end


function h = TF(N)
    % transfer function of system
    h = zeros(1, N);
    for n = 0: N - 1
        h(n + 1) = (9/10)^n/2 + (-9i/10)^n*(1/4 + 1i/4) + (9i/10)^n*(1/4 - 1i/4);
    end
end
```

2 Yule-Walker AR Modelling

```
% Author: Utkarsh Patel (18EC30048)
% Experiment 4 - Part 2

N = 128; % length of white noise
w = -1 * pi: 0.01: pi;

mean = 0; % mean of noise
var = 1;  % variance of noise

h = TF(N); % transfer function of system

h_f = var * abs(freqz(h, 1, w)).^2; % expected PSD of x(n)
subplot(2, 1, 1);
plot(w, h_f);
xticks([-pi -pi/2 0 pi/2 pi]);
xticklabels({'-\pi','-\pi/2','0','\pi/2','\pi'});
title('Theoretical PSD of sequence x(n)');
xlabel('Frequency (\omega)');
ylabel('PSD');

rng('default');
r = normrnd(mean,sqrt(var),[1 N]); % white noise
% Generating output sequence x(n)
x = conv(h, r);
x = x(1: N);

% Vary p to change pole order
AR(x, N, 3)

function AR(x, N, p)
    % Computing the AR model of order p
    w = -1 * pi: 0.01: pi;
    R = zeros(1, N); % auto-correlation of x
    assert(p < N);
    for i = 0: p
        for j = 0: N - i - 1
            R(i + 1) = R(i + 1) + x(j + 1) * x(j + i + 1);
        end
        R(i + 1) = R(i + 1) / N;
    end
    A = zeros(p, p);
    b = zeros(p, 1);
    for i = 1: p
        b(i) = -R(i + 1);
    end
    idx = 0;
    for i = 0: p - 1
        for j = 0: p - 1
            A(i + 1, j + 1) = R(abs(idx - j) + 1);
        end
        idx = idx + 1;
    end
    C = linsolve(A, b);
    var_ = R(1);
    for i = 1: p
        var_ = var_ + C(i) * R(i + 1);
    end
    den = zeros(1, p + 1);
    den(1) = 1;
    for i = 1: p
        den(i + 1) = C(i);
    end
    P = abs(freqz(den, 1, w)).^2;
    P = var_ ./ P;
    subplot(2, 1, 2);
    plot(w, P);
    xticks([-pi -pi/2 0 pi/2 pi]);
    xticklabels({'-\pi','-\pi/2','0','\pi/2','\pi'});
    title('PSD of sequence x(n) using AR with p = ' + string(p));
    xlabel('Frequency (\omega)');
    ylabel('PSD');
end


function h = TF(N)
    % transfer function of system
    h = zeros(1, N);
    for n = 0: N - 1
        h(n + 1) = (9/10)^n/2 + (-9i/10)^n*(1/4 + 1i/4) + (9i/10)^n*(1/4 - 1i/4);
    end
end
```