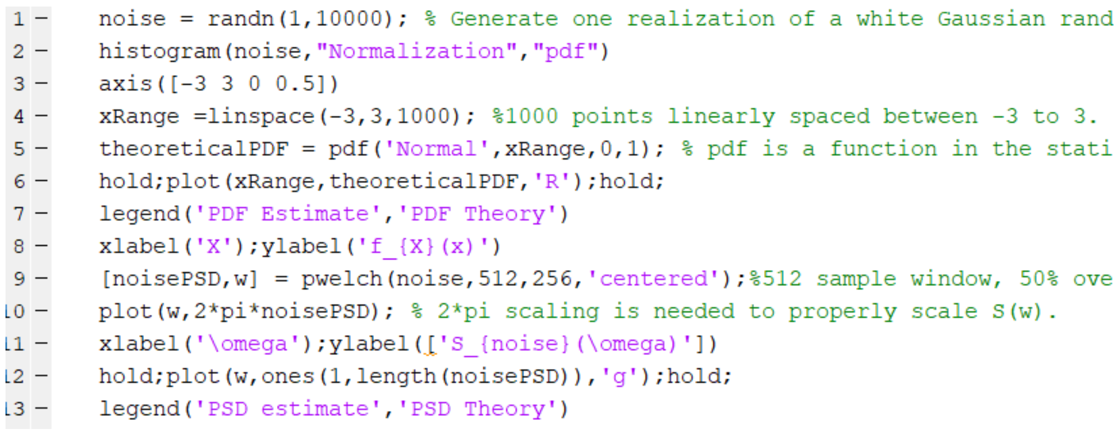
Western Engineering

ECE 9023 – Assignment 1

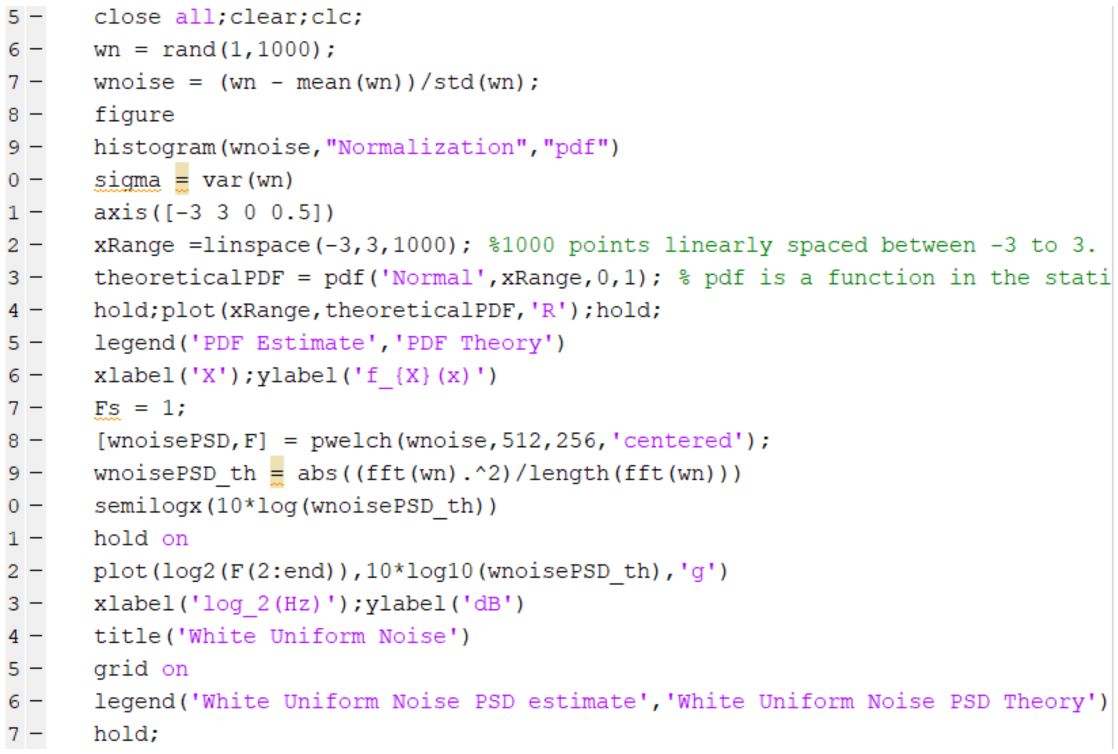
Student Name: Peizhi Yu
Western ID: 251187268
Due Date: Feb 19, 2021

1. (10 marks) Generate the following random signals in MATLAB. Plot the estimated probability density function (PDF) and power spectral density (PSD) for each. Comment on how close the estimated PDF and PSD are to their theoretical values. (see "wgn.mlx" for guidance).

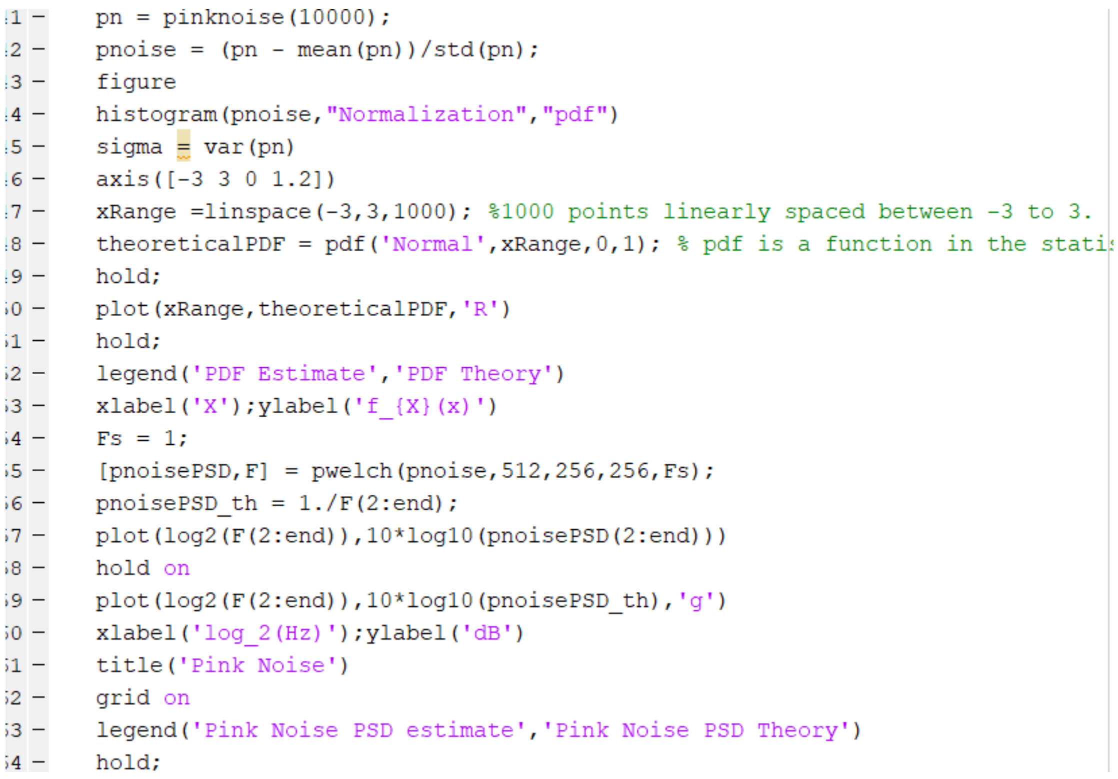
The following screen shot is the MATLAB code with comment



```
noise = randn(1,10000); % Generate one realization of a white Gaussian rand
histogram(noise,"Normalization","pdf")
axis([-3 3 0 0.5])
xRange =linspace(-3,3,1000); %1000 points linearly spaced between -3 to 3.
theoreticalPDF = pdf('Normal',xRange,0,1); % pdf is a function in the stati
hold;plot(xRange,theoreticalPDF,'R');hold;
legend('PDF Estimate','PDF Theory')
xlabel('X');ylabel('f_{X}(x)')
[noisePSD,w] = pwelch(noise,512,256,'centered');%512 sample window, 50% ove
plot(w,2*pi*noisePSD); % 2*pi scaling is needed to properly scale S(w).
xlabel('\omega');ylabel(['S_{noise}(\omega)'])
hold;plot(w,ones(1,length(noisePSD)),'g');hold;
legend('PSD estimate','PSD Theory')
```



```
close all;clear;clc;
wn = rand(1,1000);
wnoise = (wn - mean(wn))/std(wn);
figure
histogram(wnoise,"Normalization","pdf")
sigma = var(wn)
axis([-3 3 0 0.5])
xRange =linspace(-3,3,1000); %1000 points linearly spaced between -3 to 3.
theoreticalPDF = pdf('Normal',xRange,0,1); % pdf is a function in the stati
hold;plot(xRange,theoreticalPDF,'R');hold;
legend('PDF Estimate','PDF Theory')
xlabel('X');ylabel('f_{X}(x)')
Fs = 1;
[wnoisePSD,F] = pwelch(wnoise,512,256,'centered');
wnoisePSD_th = abs((fft(wn).^2)/length(fft(wn)))
semilogx(10*log(wnoisePSD_th))
hold on
plot(log2(F(2:end)),10*log10(wnoisePSD_th),'g')
xlabel('log_2(Hz)');ylabel('dB')
title('White Uniform Noise')
grid on
legend('White Uniform Noise PSD estimate','White Uniform Noise PSD Theory')
hold;
```



```
pn = pinknoise(10000);
pnoise = (pn - mean(pn))/std(pn);
figure
histogram(pnoise,"Normalization","pdf")
sigma = var(pn)
axis([-3 3 0 1.2])
xRange =linspace(-3,3,1000); %1000 points linearly spaced between -3 to 3.
theoreticalPDF = pdf('Normal',xRange,0,1); % pdf is a function in the stati
hold;
plot(xRange,theoreticalPDF,'R')
hold;
legend('PDF Estimate','PDF Theory')
xlabel('X');ylabel('f_{X}(x)')
Fs = 1;
[pnoisePSD,F] = pwelch(pnoise,512,256,256,Fs);
pnoisePSD_th = 1./F(2:end);
plot(log2(F(2:end)),10*log10(pnoisePSD(2:end)))
hold on
plot(log2(F(2:end)),10*log10(pnoisePSD_th),'g')
xlabel('log_2(Hz)');ylabel('dB')
title('Pink Noise')
grid on
legend('Pink Noise PSD estimate','Pink Noise PSD Theory')
hold;
```

a. White Gaussian Noise, mean = 0, variance = 1

The following figure is the PDF. of generated white gaussian noise.

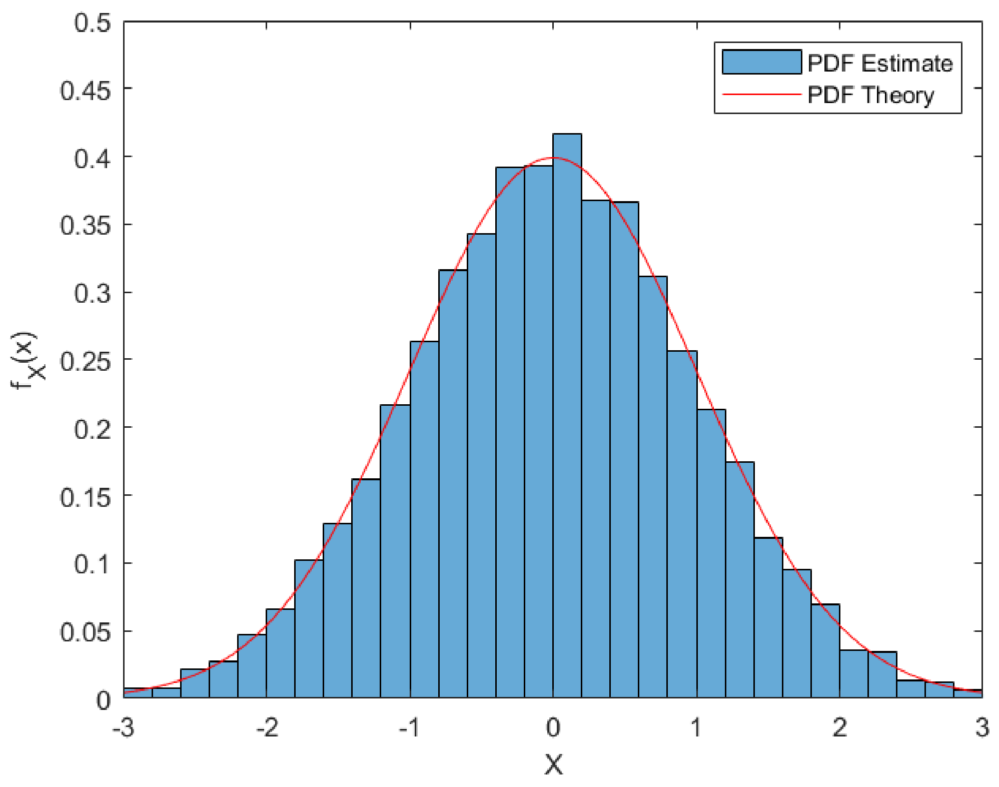


Figure 1. Probability Density Function of White Gaussian Noise

The following figure is the PSD. of generated white gaussian noise.

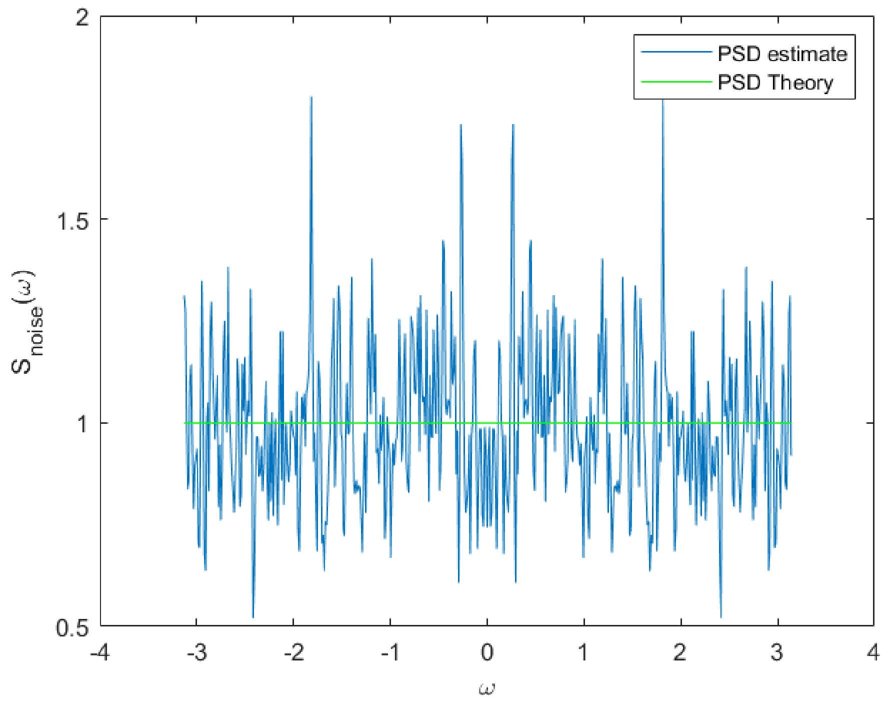


Figure 2. Power Spectral Density of White Gaussian Noise

b. White Uniform Noise, mean = 0, variance = 1

The following figure is the PDF. of generated white Uniform noise.

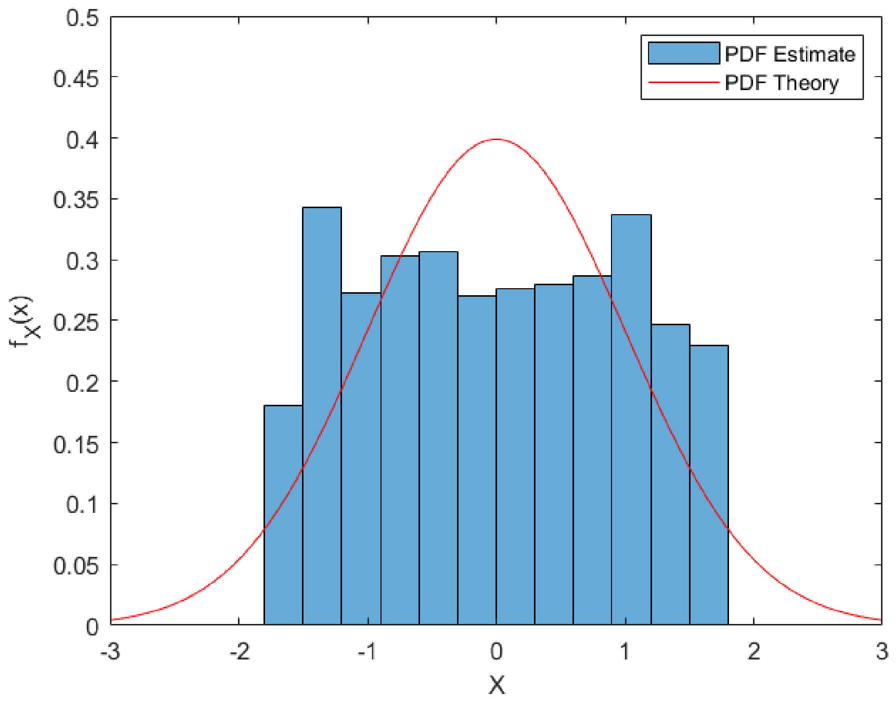


Figure 3. Probability Density Function of White Uniform Noise

The following figure is the PSD. of generated white uniform noise.

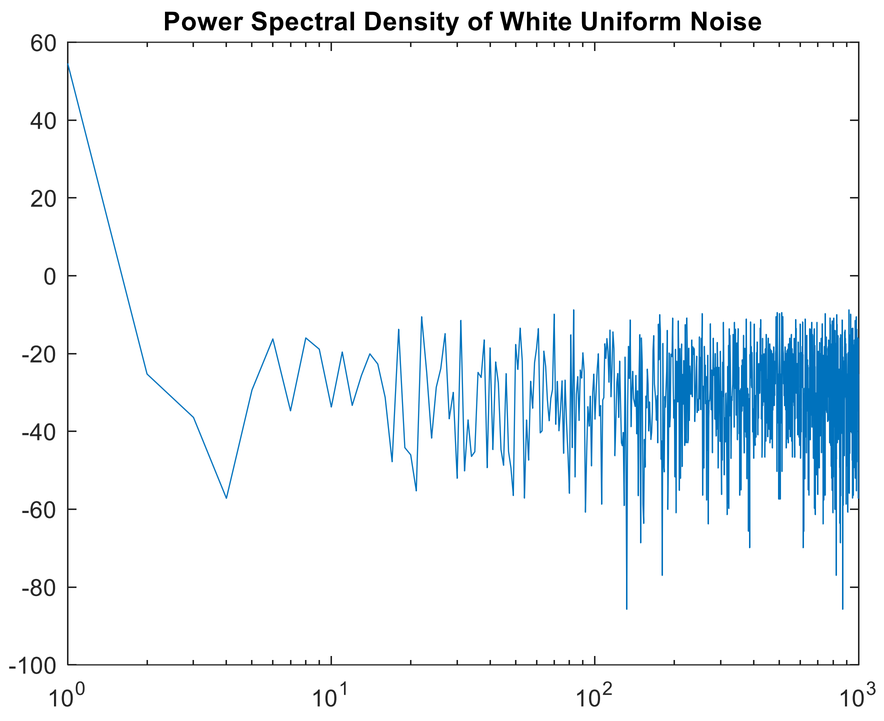


Figure 4. Power Spectral Density of White uniform Noise

c. Pink Gaussian Noise, mean = 0, variance = 1
The following figure is the PDF of pink gaussian noise.

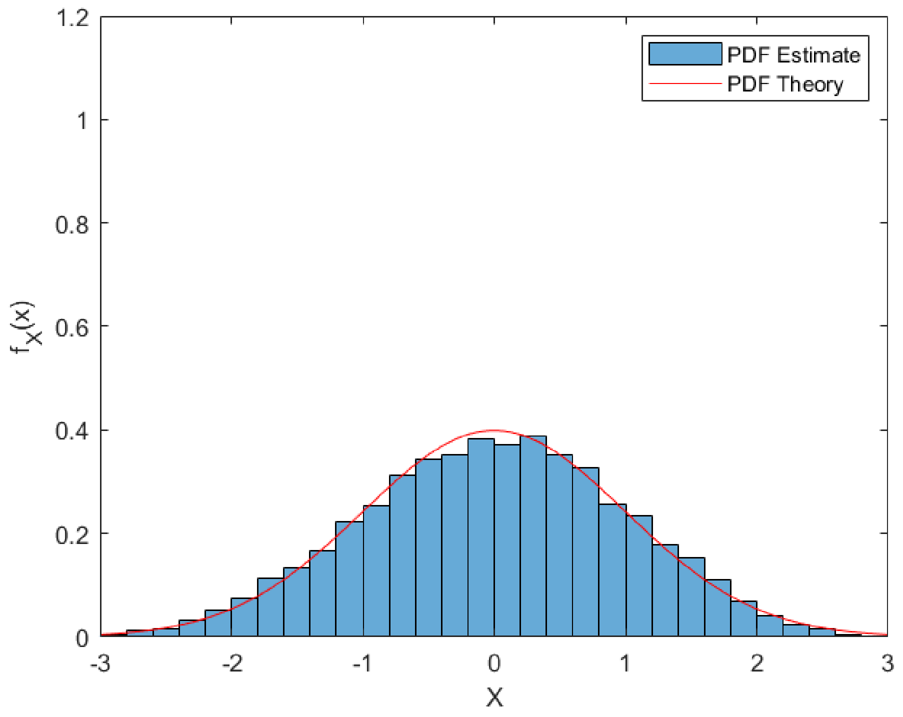


Figure 5. Probability Density Function of Pink Gaussian Noise

The following figure is the plot of generated pink gaussian noise.

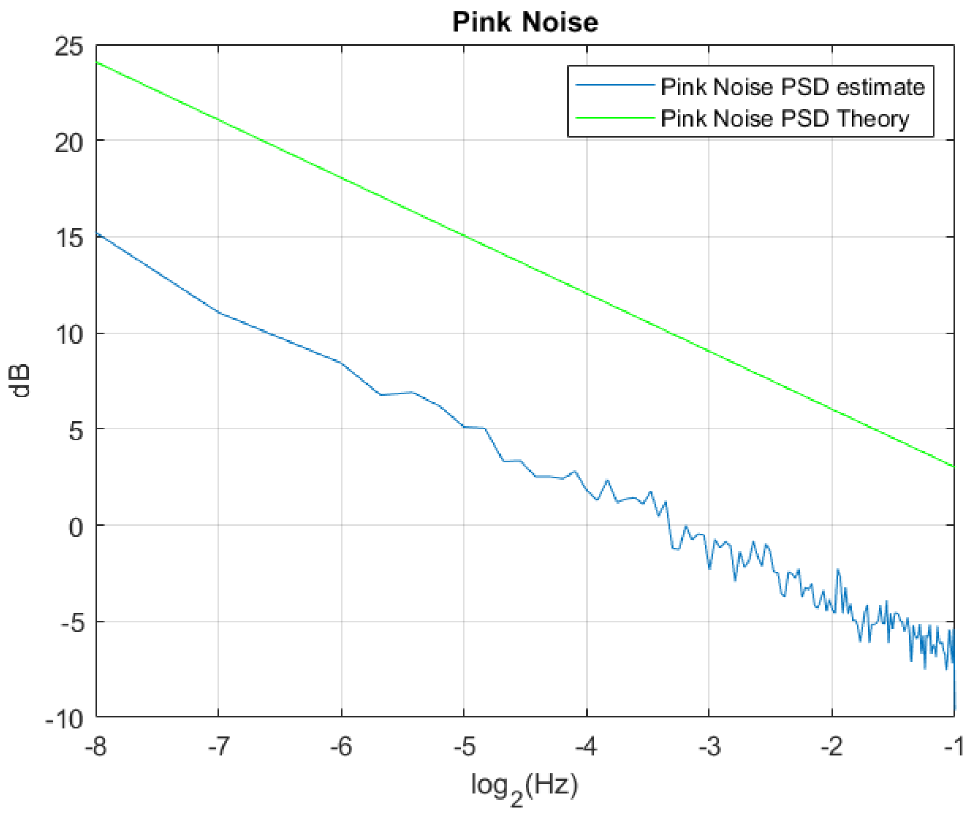
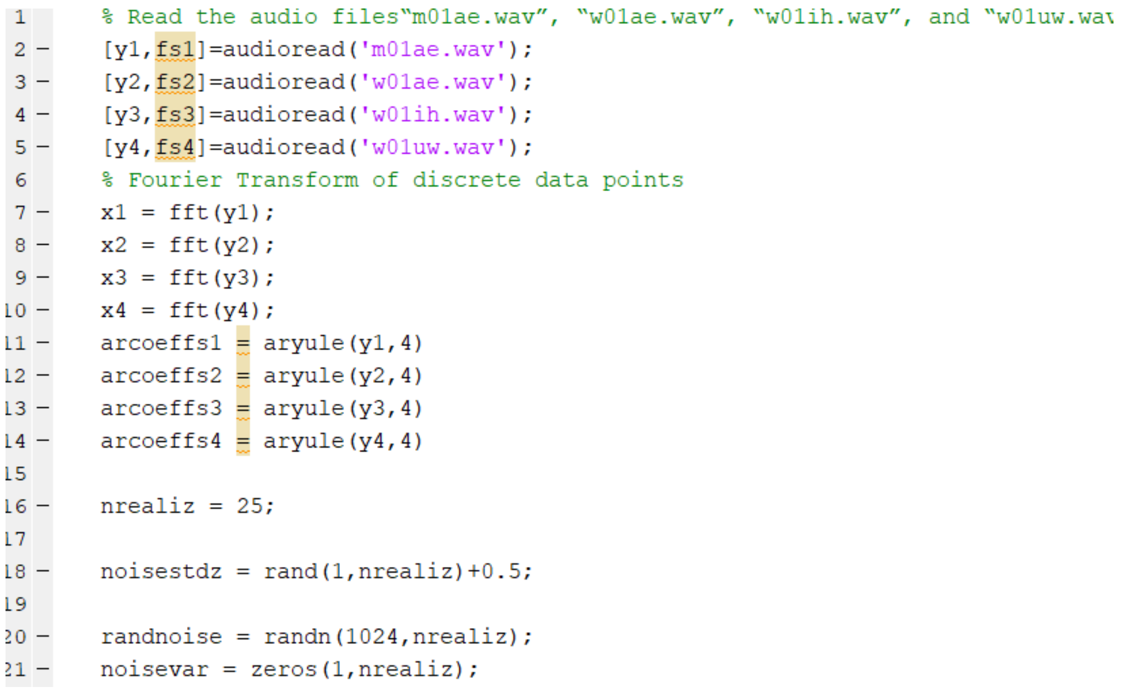


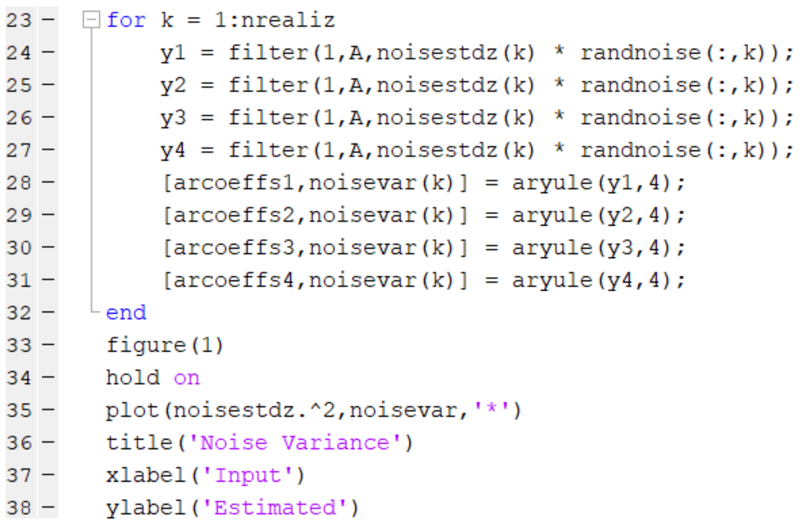
Figure 6. Power spectral density of Pink gaussian noise

2. (10 marks) In this exercise, we will apply AR modeling to speech samples. Download “m01ae.wav”, “w01ae.wav”, “w01ih.wav”, and “w01uw.wav” from “Resources -> MATLAB” directory. Complete the following:
a. For each speech sample, plot the estimated variance of the white noise input against the model order, with the model order ranging from 1 to 25. See the documentation for “aryule” command for accessing the estimated variance. Comment on the results. What would be a good model order for modelling these waveforms?
b. For each speech sample and the chosen model order, compute and plot the periodogram and AR spectral estimates. See “LinearPredictionExample.mlx” for guidance. Comment on the results. In particular, what is the AR spectral estimate trying to model? Are the AR spectral estimates the same across the four speech samples?

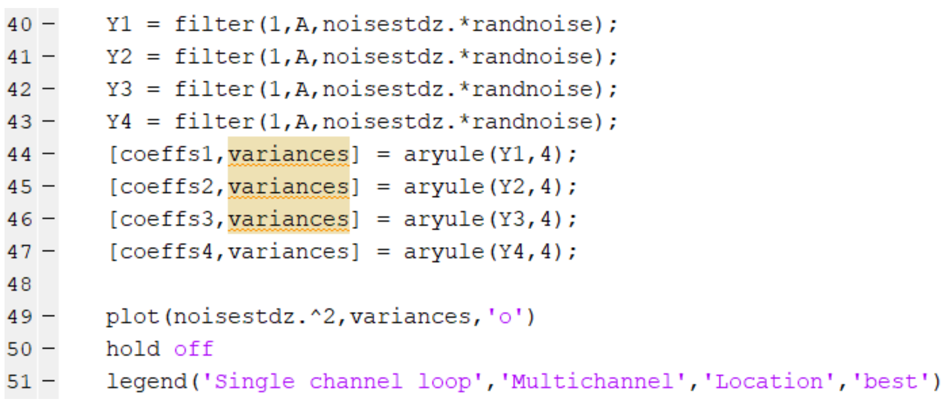
The following screen shot is the MATLAB code with comment



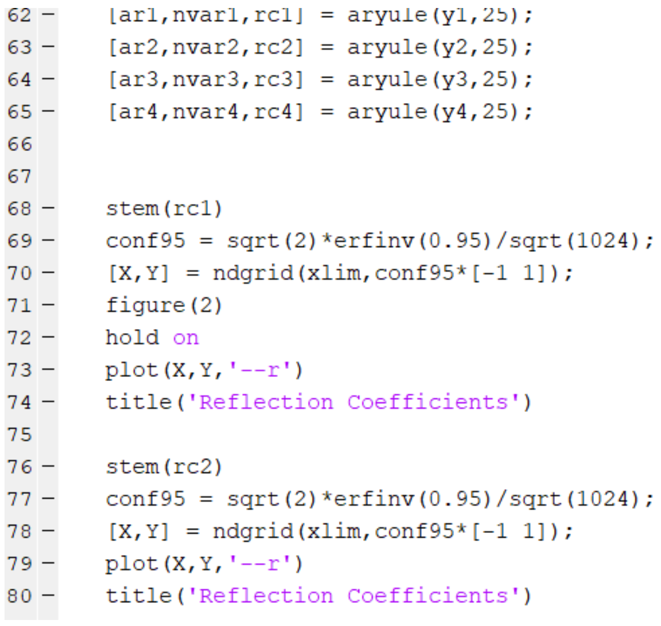
```
% Read the audio files“m01ae.wav”, “w01ae.wav”, “w01ih.wav”, and “w01uw.wa
[y1,fs1]=audioread('m01ae.wav');
[y2,fs2]=audioread('w01ae.wav');
[y3,fs3]=audioread('w01ih.wav');
[y4,fs4]=audioread('w01uw.wav');
% Fourier Transform of discrete data points
x1 = fft(y1);
x2 = fft(y2);
x3 = fft(y3);
x4 = fft(y4);
arcoeffs1 = aryule(y1,4)
arcoeffs2 = aryule(y2,4)
arcoeffs3 = aryule(y3,4)
arcoeffs4 = aryule(y4,4)

nrealiz = 25;

noisestdz = rand(1,nrealiz)+0.5;

randnoise = randn(1024,nrealiz);
noisevar = zeros(1,nrealiz);
```



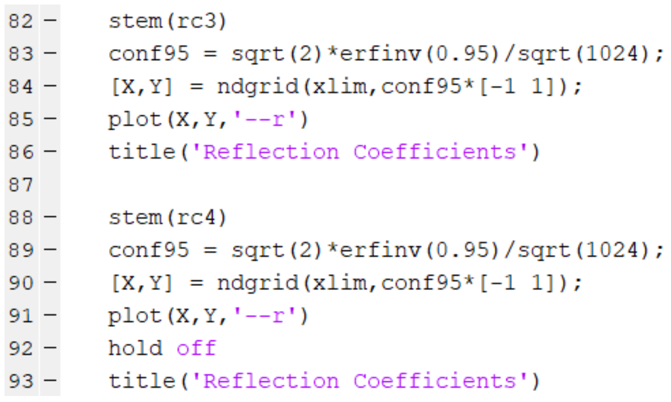
```
for k = 1:nrealiz
    y1 = filter(1,A,noisestdz(k) * randnoise(:,k));
    y2 = filter(1,A,noisestdz(k) * randnoise(:,k));
    y3 = filter(1,A,noisestdz(k) * randnoise(:,k));
    y4 = filter(1,A,noisestdz(k) * randnoise(:,k));
    [arcoeffs1,noisevar(k)] = aryule(y1,4);
    [arcoeffs2,noisevar(k)] = aryule(y2,4);
    [arcoeffs3,noisevar(k)] = aryule(y3,4);
    [arcoeffs4,noisevar(k)] = aryule(y4,4);
end
figure(1)
hold on
plot(noisestdz.^2,noisevar,'*')
title('Noise Variance')
xlabel('Input')
ylabel('Estimated')
```



```matlab
Y1 = filter(1,A,noisestdz.*randnoise);
Y2 = filter(1,A,noisestdz.*randnoise);
Y3 = filter(1,A,noisestdz.*randnoise);
Y4 = filter(1,A,noisestdz.*randnoise);
[coeffs1,variances] = aryule(Y1,4);
[coeffs2,variances] = aryule(Y2,4);
[coeffs3,variances] = aryule(Y3,4);
[coeffs4,variances] = aryule(Y4,4);

plot(noisestdz.^2,variances,'o')
hold off
legend('Single channel loop','Multichannel','Location','best')
```



```matlab
[ar1,nvar1,rc1] = aryule(y1,25);
[ar2,nvar2,rc2] = aryule(y2,25);
[ar3,nvar3,rc3] = aryule(y3,25);
[ar4,nvar4,rc4] = aryule(y4,25);


stem(rc1)
conf95 = sqrt(2)*erfinv(0.95)/sqrt(1024);
[X,Y] = ndgrid(xlim,conf95*[-1 1]);
figure(2)
hold on
plot(X,Y,'--r')
title('Reflection Coefficients')

stem(rc2)
conf95 = sqrt(2)*erfinv(0.95)/sqrt(1024);
[X,Y] = ndgrid(xlim,conf95*[-1 1]);
plot(X,Y,'--r')
title('Reflection Coefficients')
```



```matlab
stem(rc3)
conf95 = sqrt(2)*erfinv(0.95)/sqrt(1024);
[X,Y] = ndgrid(xlim,conf95*[-1 1]);
plot(X,Y,'--r')
title('Reflection Coefficients')

stem(rc4)
conf95 = sqrt(2)*erfinv(0.95)/sqrt(1024);
[X,Y] = ndgrid(xlim,conf95*[-1 1]);
plot(X,Y,'--r')
hold off
title('Reflection Coefficients')
```

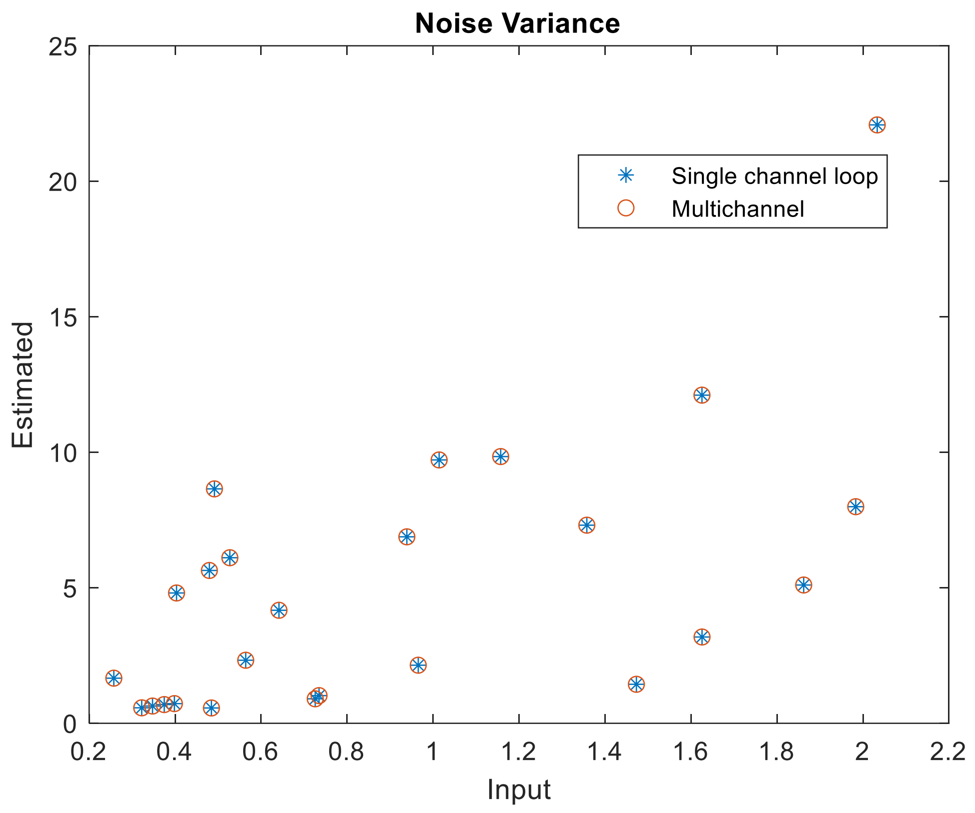


Figure 7. Noise Variance

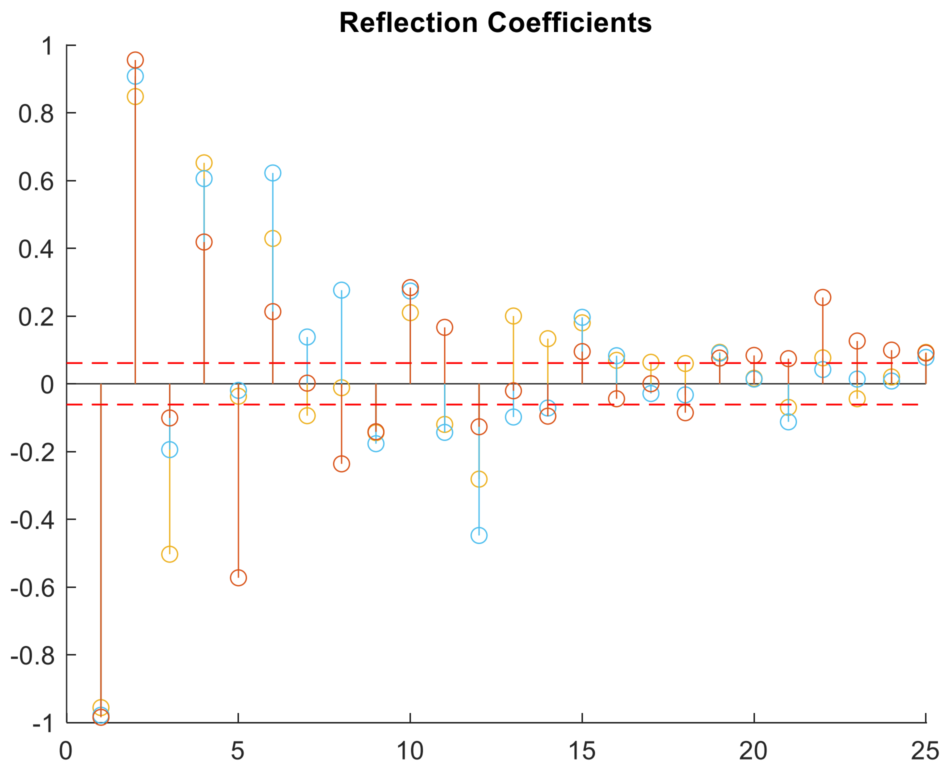


Figure 8. Reflection Coefficients

3. (5 marks) The input to a Wiener filter of length two is described by the difference equation, u(n) = x(n) + v2(n), where x(n) = 0.3x(n-1) + 0.64 x(n-2) + v1(n), and v1(n) and v2(n) are zero-mean white noise processes of variances 0.4 and 0.2 respectively. The desired input is given by the difference equation, d(n) = 0.1x(n) + 0.52 x(n-1). We derived the equations for the error performance surface and the Wiener filter for this example in class. Do the following in MATLAB (see "WienerFilterExample1.m" for guidance):

a. Plot the error performance surface as function of the weights.

c. *Plot the contours of the error performance surface and indicate the Wiener solution on this plot.*

d. *Plot the gradient vectors and comment on their orientation.*

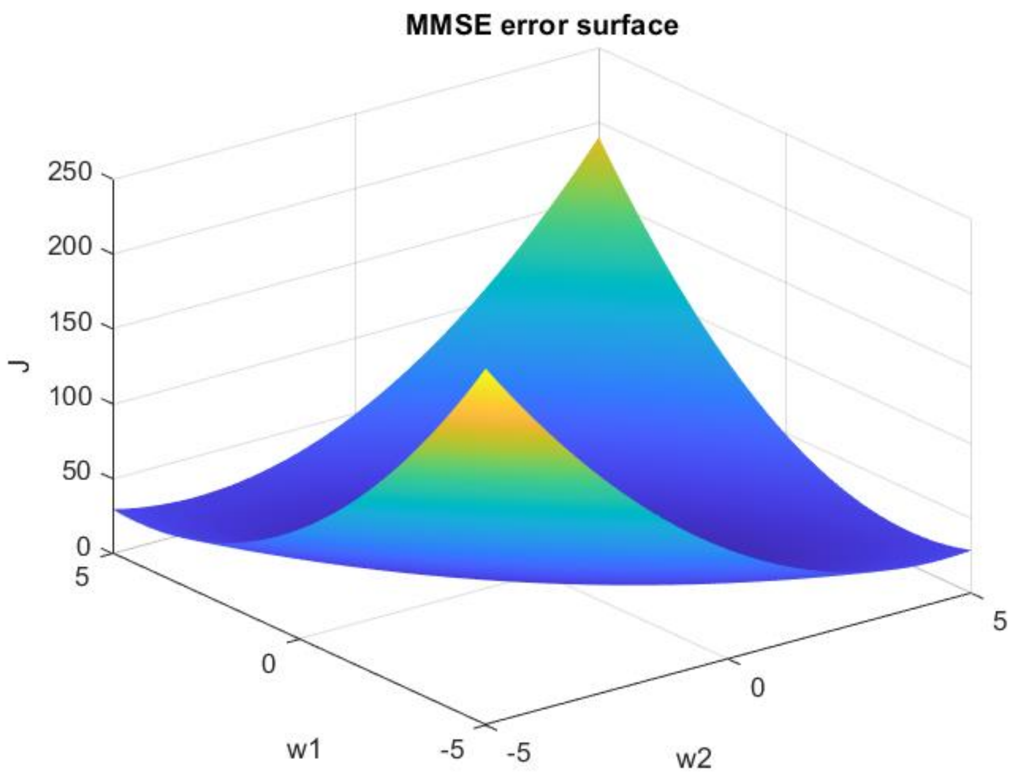


Figure 9. Error performance surface

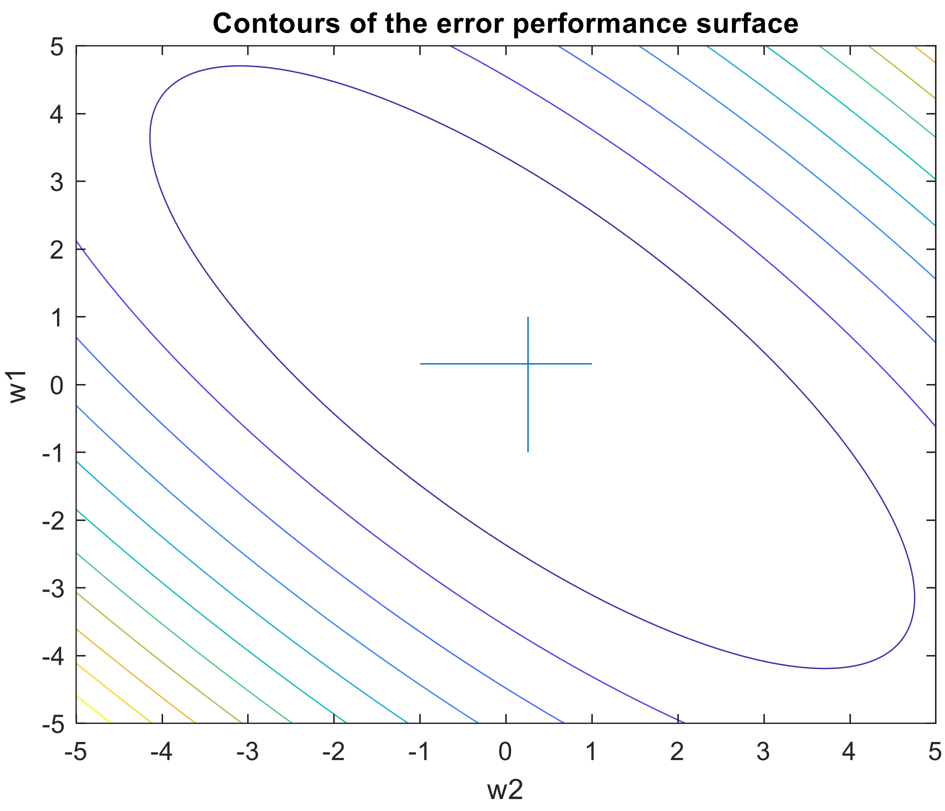


Figure 10. Contours of the error performance surface

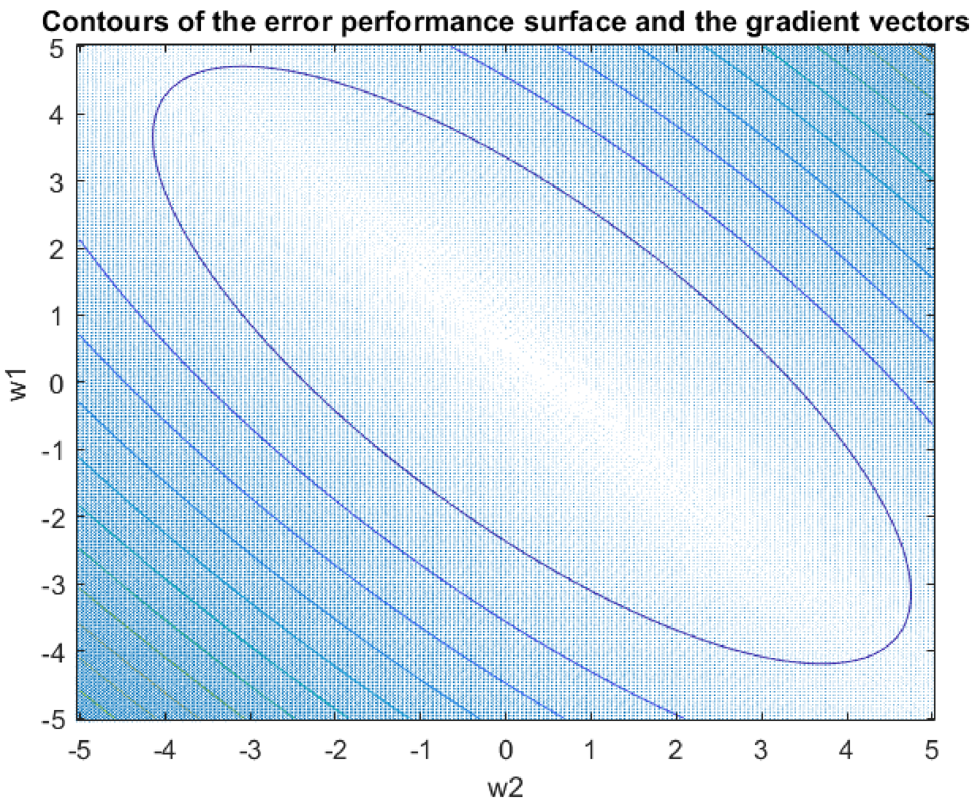


Figure 11. Contours of the error performance surface and gradient vectors

4. (15 marks) Consider a one-step adaptive predictor for a generic second order real AR process defined by the difference equation u(n) + a1 u(n-1) +a2 u(n-2) = v(n), where v(n) is a zero-mean white noise process with variance σ_v^2.

a. *Given u(n) + a1 u(n-1) +a2 u(n-2) = v(n), and variance*

Given

$$u(n) + a_1 u(n-1) + a_2 u(n-2) = v_n \quad , \quad \sigma_v^2$$

$$\underline{a} = [1 \quad a_1 \quad a_2]^T$$

$u(n)$ is real process , $r_u(-1) = r_u(1)$

$$\begin{bmatrix} r_u(0) & r_u(1) \\ r_u(1) & r_u(0) \end{bmatrix} \begin{bmatrix} -a_1^* \\ -a_2^* \end{bmatrix} = \begin{bmatrix} r_u(1) \\ r_u(2) \end{bmatrix}$$

$L = 0$, $\sum_{k=0}^{m} a_k^* r_u(-k) = \sigma_v^2$, then

$$r_u(0) + a_1^* r_u(-1) + a_2^* r_u(-2) = \sigma_v^2$$

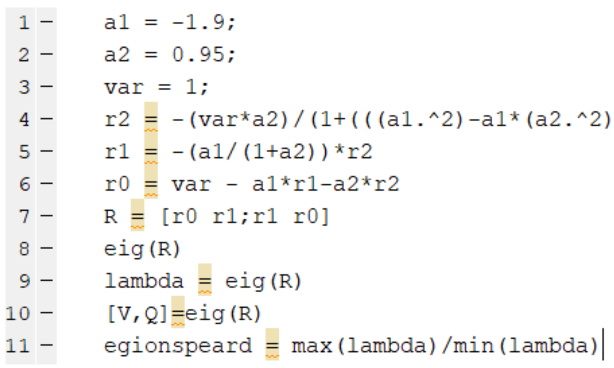
$$\rightarrow r_u(0) + a_1 r_u(1) + a_2 r_u(2) = \sigma_v^2 \quad ①$$

$$\begin{cases} -a_1^* r_u(0) - a_2^* r_u(1) = r_u(1) & ② \\ -a_1^* r_u(1) - a_2^* r_u(0) = r_u(2) & ③ \end{cases}$$

Simply ①. ②. ③ :

$$\begin{cases} r(0) = \sigma_v^2 - a_1 r(1) - a_2 r(2) \\ r(1) = \dfrac{-a_1}{1+a_2} r(2) \\ r(2) = -\sigma_v^2 a_2 \Big/ \left(1 + \dfrac{a_1^2 - a_1 a_2^2}{1+a_2} - a_2^2\right) \end{cases}$$

The following screen shot is the MATLAB code with comment



```
a1 = -1.9;
a2 = 0.95;
var = 1;
r2 = -(var*a2)/(1+(((a1.^2)-a1*(a2.^2))/(1+a2))-(a2.^2))
r1 = -(a1/(1+a2))*r2
r0 = var - a1*r1-a2*r2
R = [r0 r1;r1 r0]
eig(R)
lambda = eig(R)
[V,Q]=eig(R)
egionspeard = max(lambda)/min(lambda)
```

b. *If a1= -1.9, a2=0.95,* σ_v^2 *= 1, compute the eigenvalues, eigenvectors, and the eigenvalue spread*

The eigenvalues are

```
ans =

    0.3700
    1.0245
```

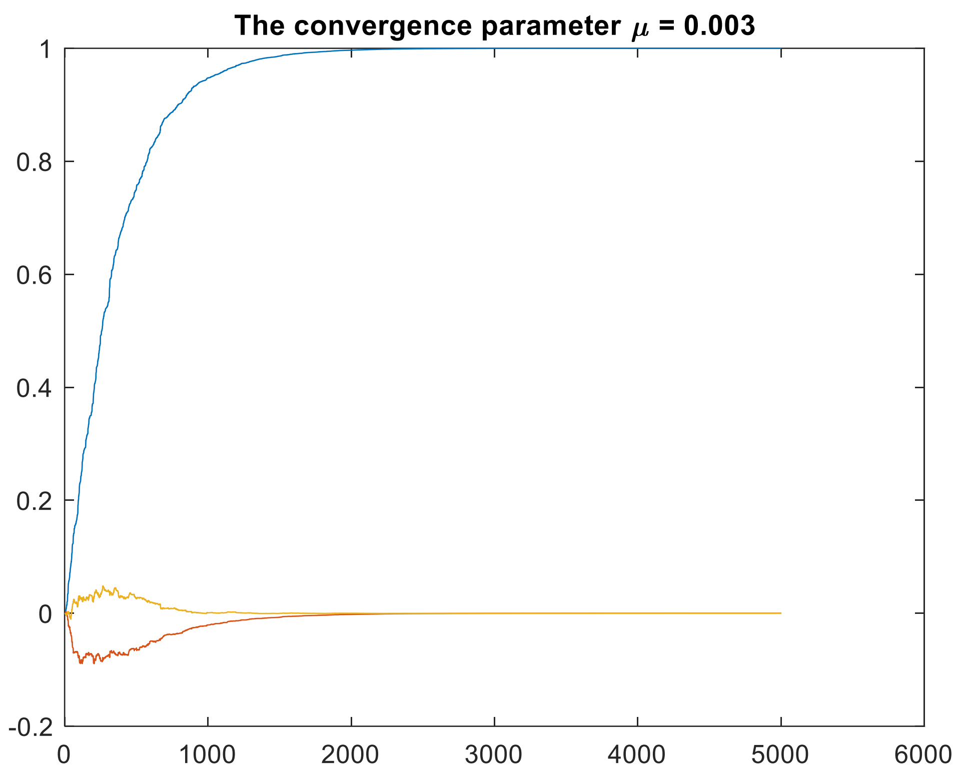
The eigenvectors are

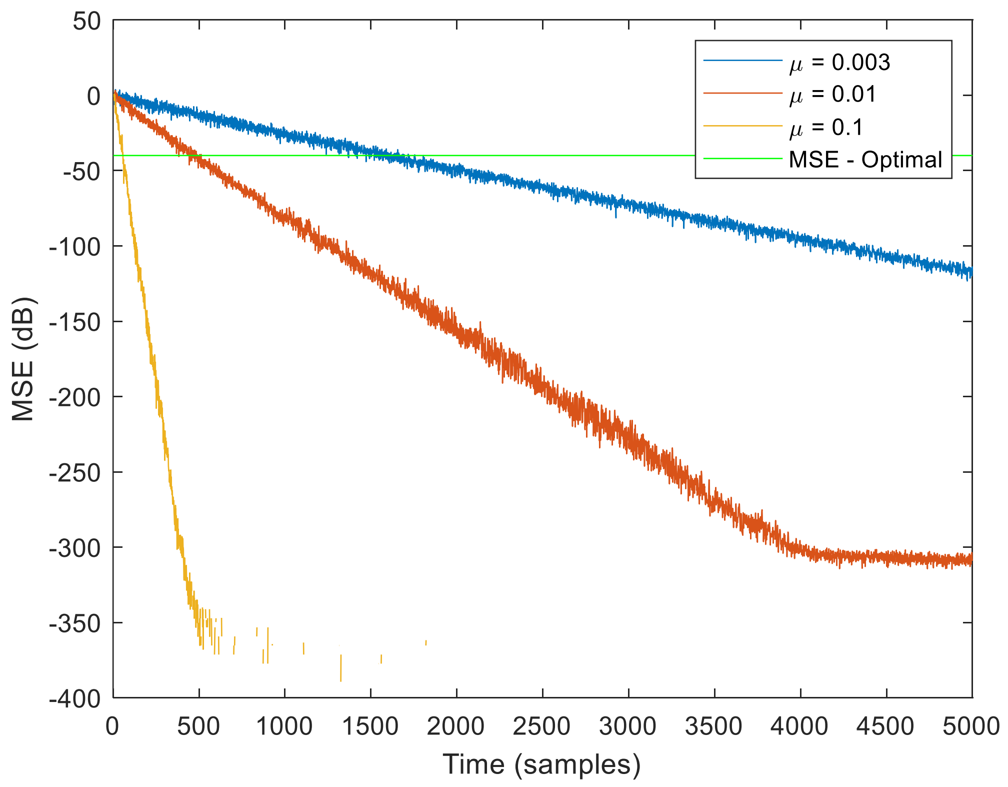
```
Q =

    0.3700         0
         0    1.0245
```

The eigenvalue spread is

```
egionspeard =

    2.7694
```

c. Define $e(n) = u(n) - \hat{u}(n)$, $\varepsilon_1(n) = 1.9 - w_1(n)$, and $\varepsilon_2(n) = -0.95 - w_2(n)$. Using the LMS algorithm with the convergence parameter, μ = 0.003, plot the power spectral densities for e(n), $\varepsilon_1(n)$, and $\varepsilon_2(n)$. Include these plots in your assignment, clearly labeling the axes. What observations can you make from these plots?


The convergence parameter μ = 0.003
1
0.8
0.6
0.4
0.2
0
-0.2
0
1000
2000
3000
4000
5000
6000


μ = 0.003
μ = 0.01
μ = 0.1
MSE - Optimal
MSE (dB)
Time (samples)
50
0
-50
-100
-150
-200
-250
-300
-350
-400
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000