**SANJIT ANAND U19EC008**

**14-03-2022 PRACTICAL ASSIGNMENT - 6**

**WIRELESS AND MOBILE COMMUNICATION**

**AIM:**

To Simulate Equalization Techniques using AWGN channel considering input as any random data as well as an Image with the help of MATLAB software. Also compare output of both with and without equalization.

**APPARATUS:**

MATLAB Software

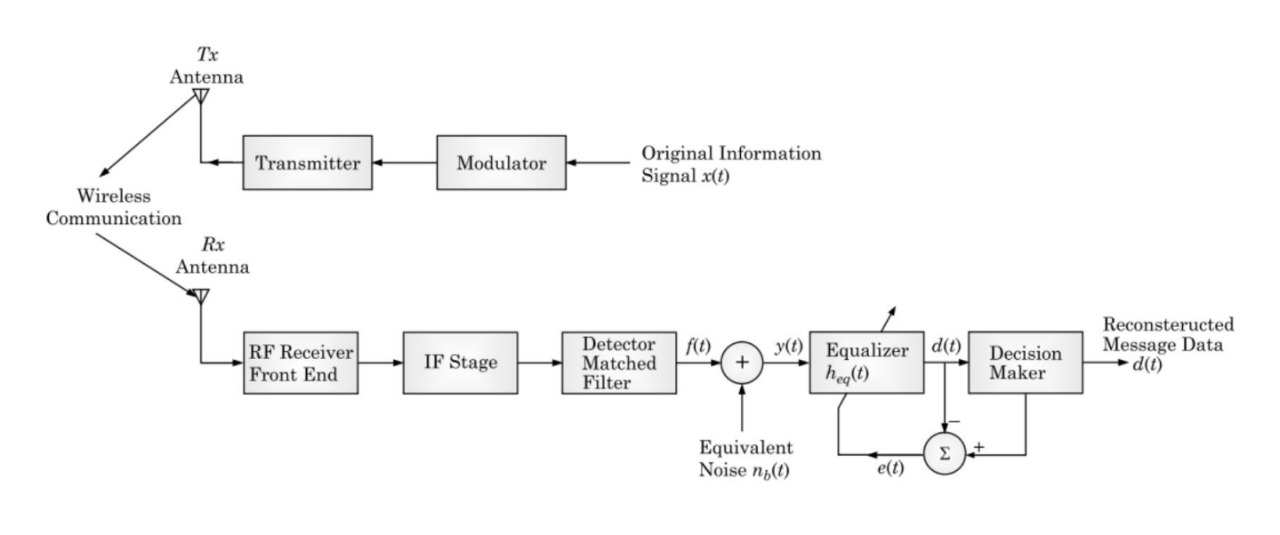
**THEORY:**

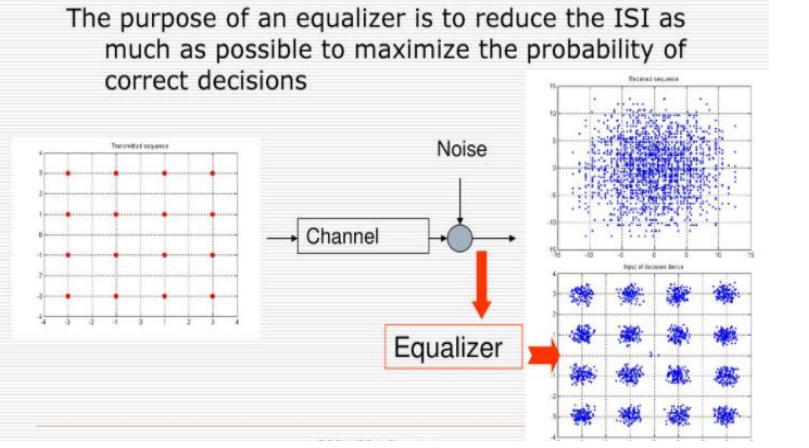
**Equalization Techniques**

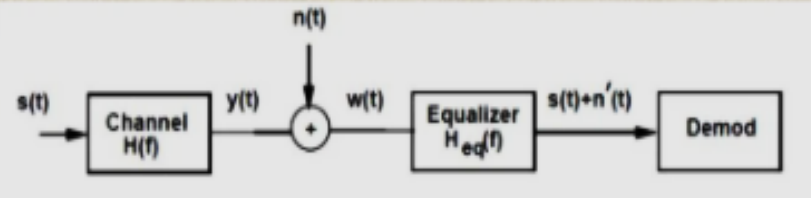
With the growth in wireless communication, there was a requirement to improve the received signal quality because there were a number of channels working at different frequencies which cause signal fading and interference. Thus techniques were evolved to improve received signal of quality. These techniques are ‘Equalization and Diversity’. Equalization is used to compensate intersymbol interference (ISI) while Diversity is used to compensate for fading.

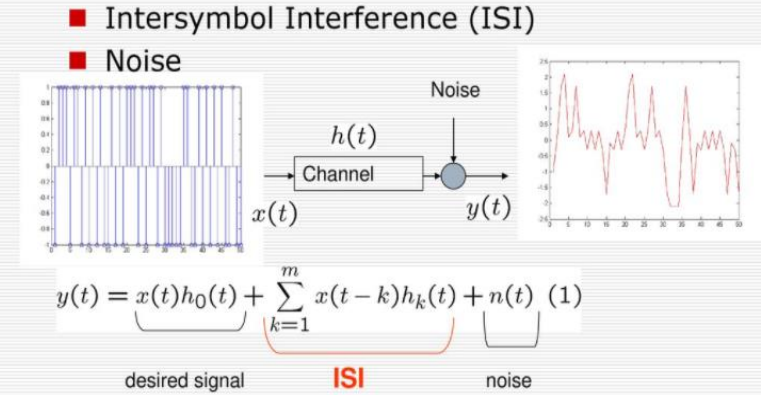
An equalizer is usually implemented at base band or at IF stage in a receiver. Adaptive equalizer works on two operating modes, that are training and tracking. A fixed length training sequence is sent by the transmitter so that the receiver’s equalizer may receive the signal for minimum bit error rule (BER). This sequence is a pseudo random binary signal followed by the user data.

**Block Diagram of Communication Signal with Equalization**









**PROCEDURE:**

1. Input the random data (Let say 1000 in number) and change into required format.

2. Modulate the data with psk

3. Assume SNR and add AWGN

4. Assume Tau, PdB and Rayleigh channel

5. Realize it with filter function

6. Assume LMS as adaptive algorithm object

7. Construct linear equalizer object (lineareq)

8. Equalize signal using equalizer object (equalize)

9. Demodulate the data

10. Convert it into required format

11. Find BER

12. Compare output for both with and without equalizer (Reapet steps 9-11 for without equalizer)

13. Repeat the above procedure for input as an image

**MATLAB CODE & OUTPUTS:**

**Part 1: Using input as Image**

% Lab-06 U19EC008

% Equalization, Input as Image

clc;

clear all;

close all;

id = imread('cameraman.tif');

figure('name','Transmitted Image U19EC008');

imshow(id);

ida = id(:);

ib = de2bi(ida);

ib = ib(:);

M = 4;

x = mod(length(ib),log2(M));

Mod=2;

data\_vector = reshape(ib, [numel(ib)/Mod Mod]);

msg = bi2de(data\_vector);

hMod = comm.QPSKModulator('PhaseOffset',0);

% Modulate using QPSK.

modmsg = step(hMod,msg);

SNR= 40;

modmsg = awgn(modmsg , SNR);

scatterplot(modmsg);

legend('Modulated signal U19EC008');

% Length of training sequence

trainlen = 200;

Tauj = [0.986 0.845 0.237 0.123];

Beta = [-0.1 0 -0.03 0.31];

chan = rayleighchan(1,0,Tauj, Beta);

chanCoeff = chan.AvgPathGaindB + 1i\*chan.PathDelays;

% Introduce channel distortion.

filtmsg = filter(chanCoeff,1, modmsg);

% Equalize the received signal.

% Create an equalizer object.

eq1 = lineareq(8, lms(0.01));

release(hMod);

% Set signal constellation.

eq1.SigConst = step(hMod,(0:M-1)')';

% Equalize.

[symbolest,yd] = equalize(eq1,filtmsg,modmsg(1:trainlen));

scatterplot(filtmsg)

legend('Filtered signal U19EC008')

scatterplot(symbolest)

legend('Equilized signal U19EC008')

% Compute error rates with and without equalization.

hDemod = comm.QPSKDemodulator('PhaseOffset',0);

% Demodulate unequalized signal.

demodmsg\_noeq = step(hDemod,filtmsg);

% Demodulate detected signal from equalizer.

demodmsg = step(hDemod,yd);

bidemodmsg = de2bi(demodmsg);

%% OUTPUT IMAGE WITHOUT EQUALIZATION

de = de2bi(demodmsg\_noeq);

ber\_without\_eq = sum(mean(abs(uint8(de) - data\_vector)))

%Double to uint8 converion

r = uint8(de);

r = r(:);

%Reshaping demodulated output

temp = r(1:length(r),:);

t = reshape(temp,[],8);

%Binary to Decimal conversion

k = bi2de(t);

%Reshaping into matrix of the size of image

p = reshape(k,256,256);

%Displaying output

figure('name','Received without Equalisation U19EC008');

imshow(p);

%% OUTPUT IMAGE WITH EQUALIZATION

de = de2bi(demodmsg);

ber\_with\_eq = sum(mean(abs(uint8(de) - data\_vector)))

%Double to uint8 converion

r = uint8(de);

r = r(:);

%Reshaping demodulated output

temp = r(1:length(r),:);

t = reshape(temp,[],8);

%Binary to Decimal conversion

k = bi2de(t);

%Reshaping into matrix of the size of image

p = reshape(k,256,256);

%Displaying output

figure('name','Received with Equalisation U19EC008');

imshow(p);

disp('Bit Error Rates with Equalisation')

disp(ber\_with\_eq)

disp('Bit Error Rates without Equalisation')

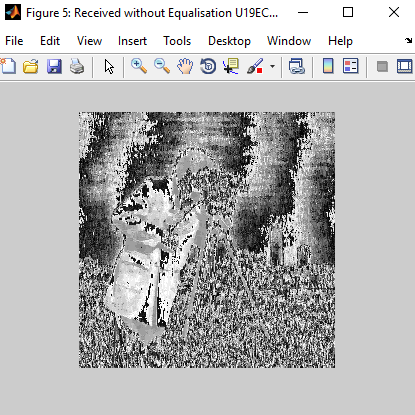
disp(ber\_without\_eq)

**Output- Part 1: Using input as Image**

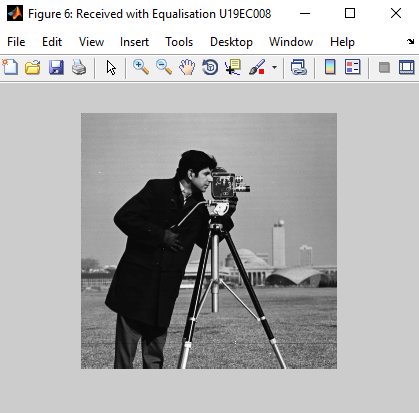
**Transmitted Image**



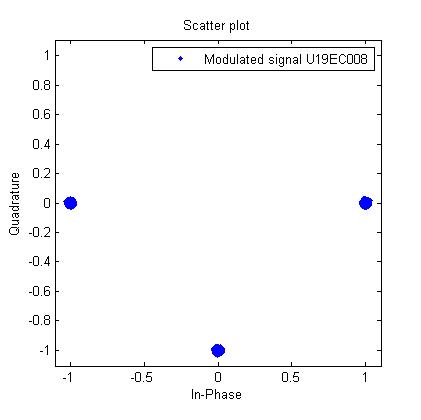
**Received Image without Equalisation**

****

**Received Image with Equalisation**

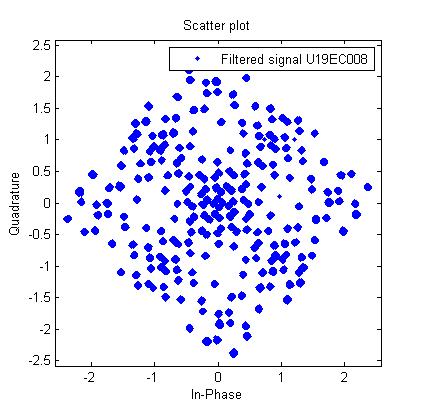
****

**Modulated Signal**

****

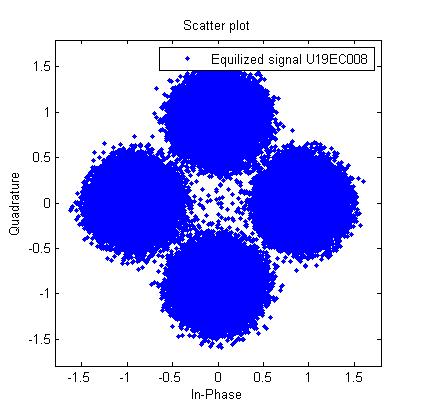
U19EC008

**Filtered Signal**

****

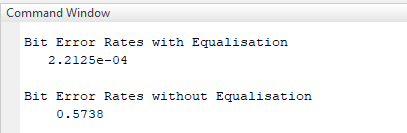
U19EC008

**Equalized Signal**

****

U19EC008

**Bit Error Rates (with & without Equalization)**

****

**Part 2: Using Random data as input**

% Lab-06 U19EC008

clc;

clear all;

close all;

M = 4;

% Generating message signal

msg = randi([0 1], 5000, 1);

[n, m] = size(msg);

Mod = 2;

data\_vector = reshape(msg, [numel(msg)/Mod Mod]);

msg = bi2de(data\_vector);

hMod = comm.QPSKModulator('PhaseOffset',0);

% Modulated signal using QPSK.

k = step(hMod, msg);

SNR = 40;

modmsg = awgn(k,SNR);

scatterplot(modmsg);

legend('Modulated signal');

% Length of training sequence

trainlen = 200;

Tauj = [0.986 0.845 0.237 0.123];

Beta = [-0.1 0 -0.03 0.31];

chan = rayleighchan(1, 0, Tauj, Beta);

chanCoeff = chan.AvgPathGaindB + 1i\*chan.PathDelays;

% Introduce channel distortion

filtmsg = filter(chanCoeff, 1, modmsg);

%%% Equalize the received signal

% Create an equalizer object

eq1 = lineareq(8, lms(0.01));

% Set signal constellation

eq1.SigConst = step(hMod,(0:M-1)')';

% Equalize.

[symbolest,yd] = equalize(eq1,filtmsg,modmsg(1:trainlen));

scatterplot(filtmsg)

legend('Filtered signal')

scatterplot(symbolest)

legend('Equilized signal')

% Compute error rates with and without equalization.

hDemod = comm.QPSKDemodulator('PhaseOffset',0);

% Demodulate unequalized signal

demodmsg\_noeq = step(hDemod,filtmsg);

% Demodulate detected signal from equalizer

demodmsg = step(hDemod,yd);

% Calculating Bit Error Rates

de = de2bi(demodmsg\_noeq);

ber\_without\_eq = sum(mean(abs((de) - data\_vector)))

de = de2bi(demodmsg);

ber\_with\_eq = sum(mean(abs((de) - data\_vector)))

disp('Bit Error Rates with Equalisation')

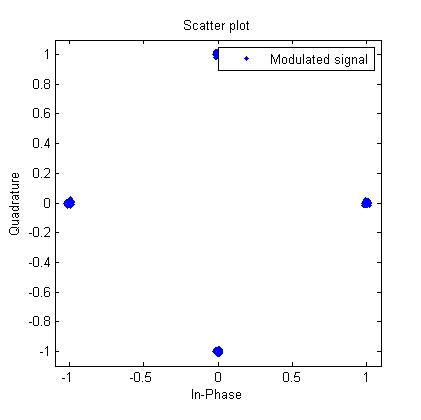
disp(ber\_with\_eq)

disp('Bit Error Rates without Equalisation')

disp(ber\_without\_eq)

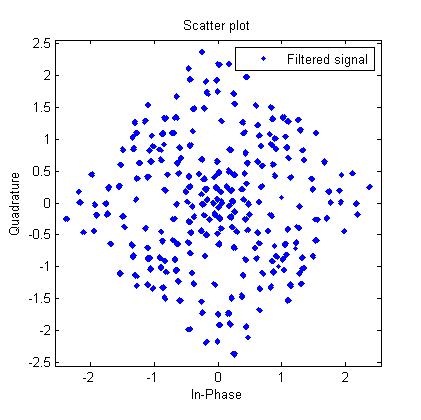
**Output - Part 1: Using Random data as input**

**Modulated Signal**

****

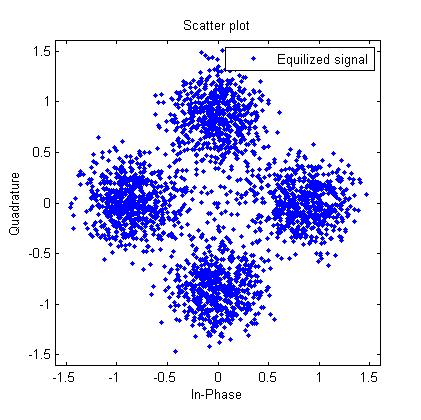
U19EC008

**Filtered Signal**

****

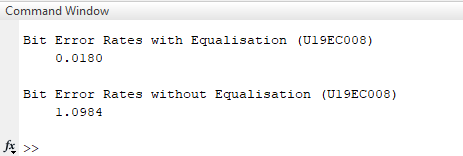
U19EC008

**Equalized Signal**

****

U19EC008

**Bit Error Rates (with & without Equalization)**

****

**CONCLUSION:**

In this experiment, we have implemented and simulate Equalization Techniques using AWGN channel considering input as any random data as well as an Image with the help of MATLAB software. We have noticed that the equalization helps in reducing noise and Inter Symbol Interference. We have also observed that bit Error Rates observed with equalization is reduced than that of without equalization.