Communication-2

Project Fall 2019

Submitted by:
Ahmed Abdou Mohamed Allam (2180001)
Abir Mohamed Roushdy (1152077)

Submitted to: Prof:Ahmed Khattab
12/25/2019

Abstract

We have learned in digital Communications course lots of modulation schemes , and compared between them according to many aspects.

In this Report as part of confirming these comparisons we are comparing between BPSK and BFSK according to BER and The used Bandwidth.

Table of Contents

1. Introduction

As most of the communication channels are bandpass in nature, digital signals are modulated by a carrier wave of appropriate frequency before transmission. In order to distinguish the transmitted bits in the presence of noise with minimum probability of error, a receiver has to be carefully designed. For the detection of binary signals in spectrally white noise, the matched filter is optimum in the sense that the output signal-to-noise ratio (SNR) is maximized at the end of the bit period and thus the probability of error in the threshold detection is minimized.

In this project, we would like to design the optimal receiver for the detection of binary Frequency Shift Keying (FSK) and Phase Shift Keying (PSK) signals and investigate the performance in each case.

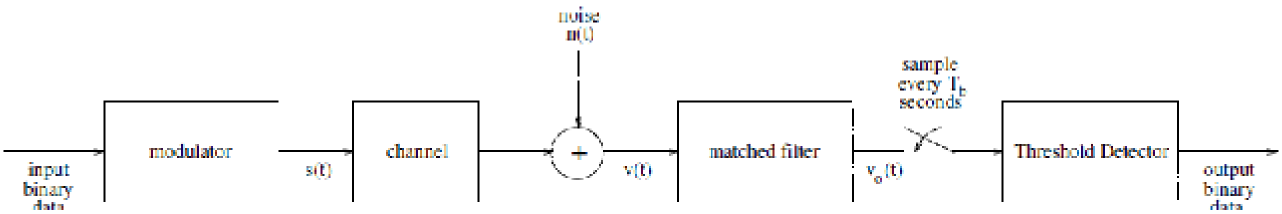


Figure 1: Block Diagram of a Binary Data Transmission System.

2) Showing Steps of Stimulating BPSK and BFSK

1) We have Generated a random Sequence of Bits “InputBits” with a number that is taken as an input from user to show our procedures for now, we will assume that the user wants 100 bits , and then it’s converted to “Polar NRZ Line Code” shown in “Figure 2”

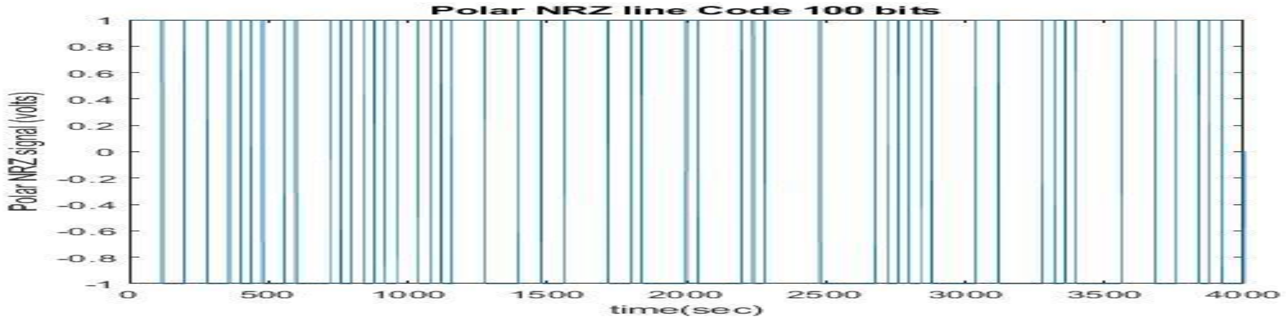


Figure 2

The Effective Bandwidth by using First Null method To compute it can be get from “Figure3”

B.W=0.02466 HZ “Approximatly”

Effective B.W= Rb/B.W where Rb=1/40 bits/sec therefore Effective B.W=1.013 bits/sec/Hz

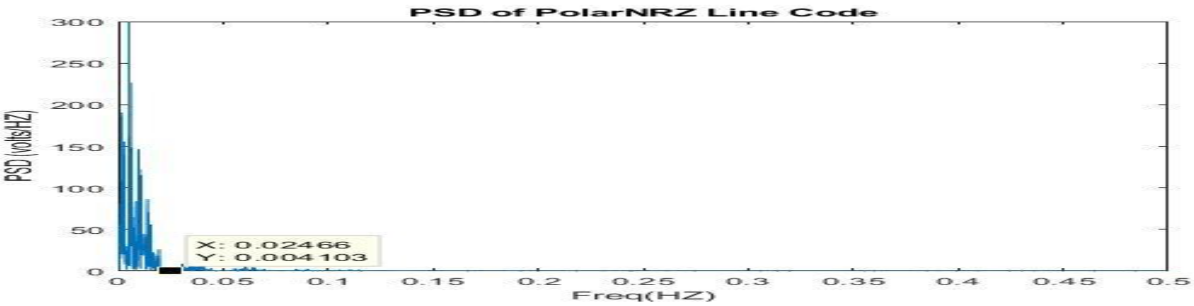


Figure 3

2) We have Generated a zero Mean White Gaussian Noise With Sp(f)=1 Volt/Hz as Shown in “Figure 4”

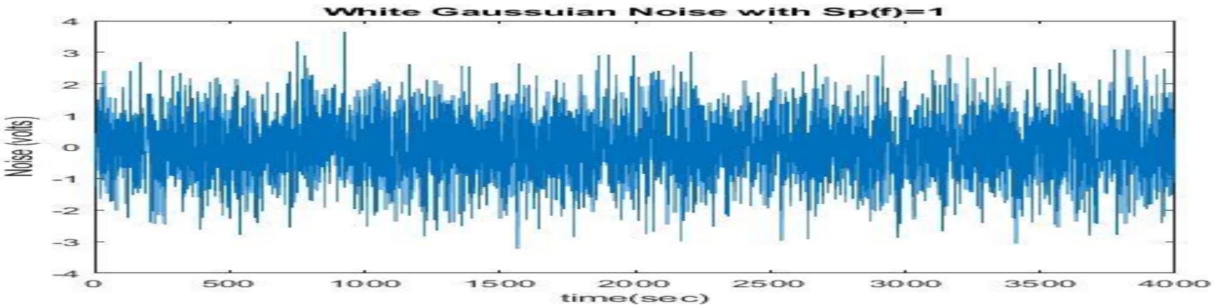


Figure 4

3) Since the noise sequence w(nTs) has a bandwidth much larger than the effective bandwidth of the signal pulses, the noise outside the effective bandwidth is irrelevant to the transmission of the data. We therefore define an effective input signal-to-noise ratio SNRe such that:

SNRe=(A^2*Tb)/No, Where No=2 and Tb=40 sec

4) With a fixed value of No = 2, for each chosen value of SNRe in the range of -4 db to 4 dB, generate a sequence of FSK modulated pulses and another sequence of PSK modulated pulses using the value of A calculated. This procedure yields the transmitted signal s(t).

The sampled noise sequence is added to each of the s(t) representing the FSK and PSK sequences. This mixture yields the noisy signal v(t) resulted from transmitting through the channel.

Shown In “Figure 5” Modulated Signal and in “Figure 6” Transmitted signal in of case of BPSK Modulation Scheme at SNRe=2 of 100 bits (Zoomed in Just to be clear in Graph)

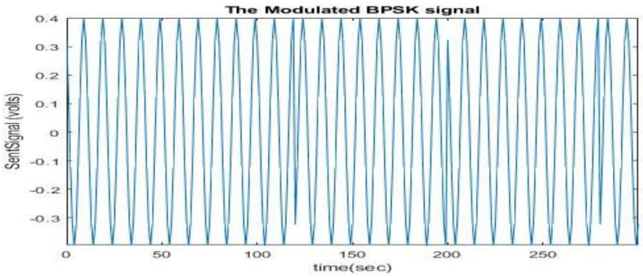


Figure 5

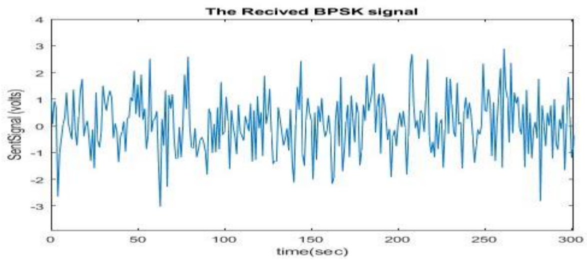


Figure 6

Shown In “Figure 7” Modulated Signal and in “Figure 8” Transmitted signal in of case of BFSK Modulation Scheme at SNRe=2 of 100 bits (Zoomed in Just to be clear in Graph)

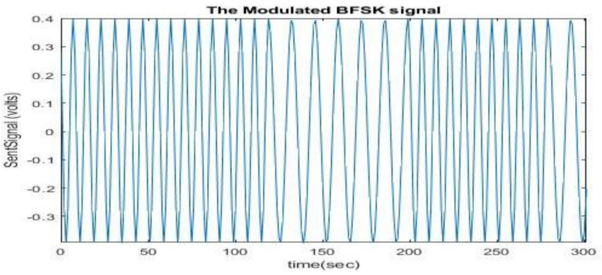


Figure 7

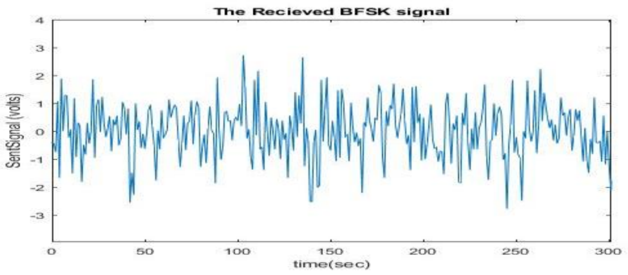


Figure 8

5)Each of the noisy signal sequences is then passed through the matched filter designed respectively for

the specific transmission scheme. Denote the output of the matched-filter at the end of each bit period

by vo(kTb).

Shown In “Figure 9” Output of Matched Filter Signal in of case of BPSK Modulation Scheme and in “Figure 10” “Figure 11” Output of Matched Filters Signals in of case of BFSK Modulation Scheme at SNRe=2 of 100 bits (Zoomed in to be clear)

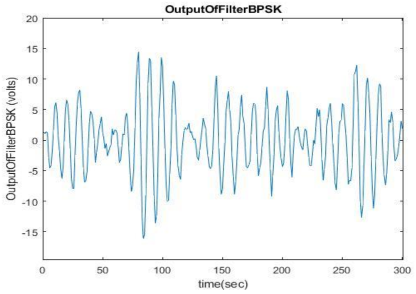


Figure 9

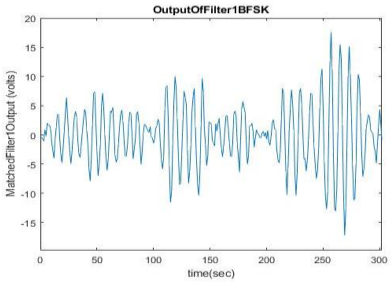


Figure 10

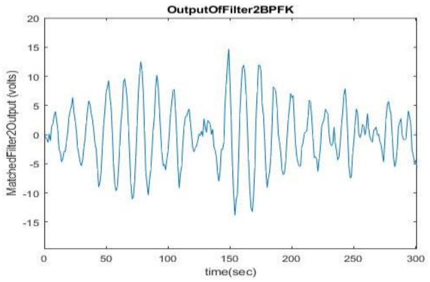


Figure11

Shown In “Figure 12” Output of Matched Filter Signal after sampling in of case of BPSK Modulation Scheme and in “Figure 13” “Figure 14” Output of Matched Filters Signals after sampling in of case of BFSK Modulation Scheme at SNRe=2 of 100 bits

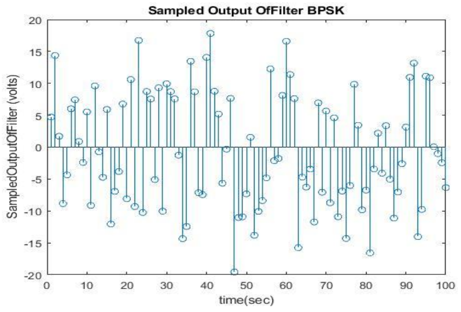


Figure 12

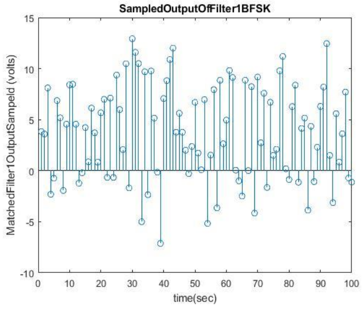


Figure 13

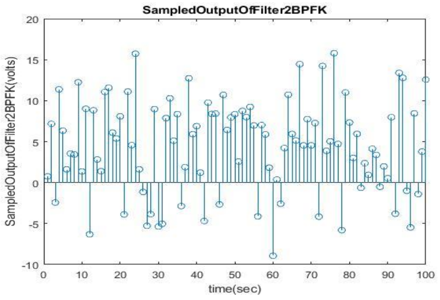
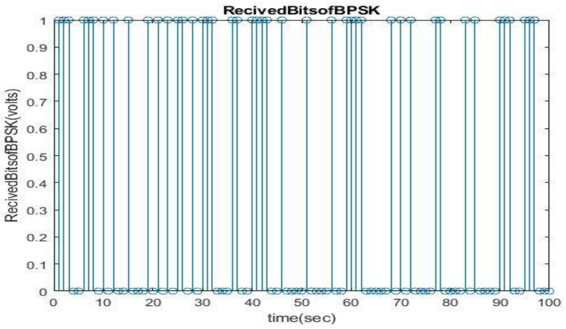


Figure14

6)Compare vo(kTb) with the respective threshold in each of the cases and decide if the transmitted bit bk is 1 or 0. We will get the Received bits as in “Figure 15” and in “Figure 16 “



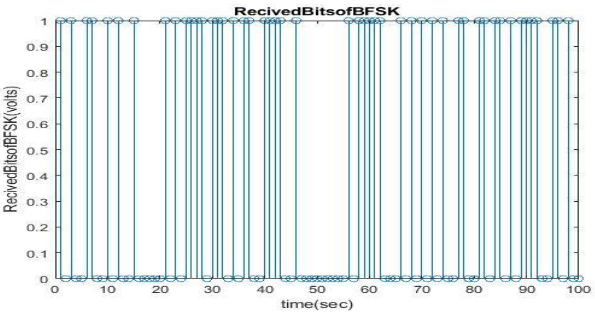


Figure 15

Figure 16

7) For a suitably selected number of bits transmitted in the sequence, evaluate the bit-error rate (experimental probability of error) of the each of the transmission schemes at the specific SNRe. Bit-error rate is the number of erroneous bit at the output in comparison with the transmitted binary sequence.

So by Applying the Experiment again twenty Times and getting the average BER. With SNR=2

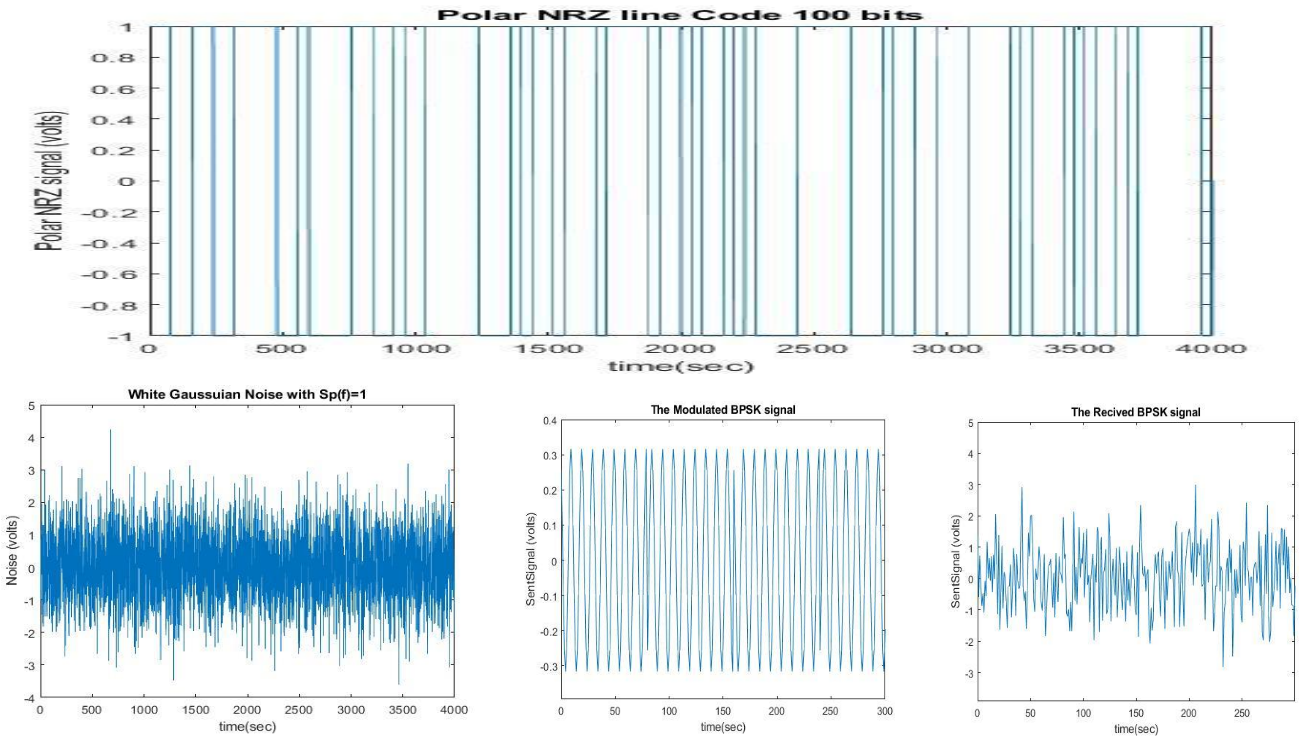
We got the following Results for BPSK and BFSK

BER|BPSK=   0.0410

BER|BFSK=   0.1130

8) Applying all of this using SNRe=0

You see the Following results in “Figure 17”



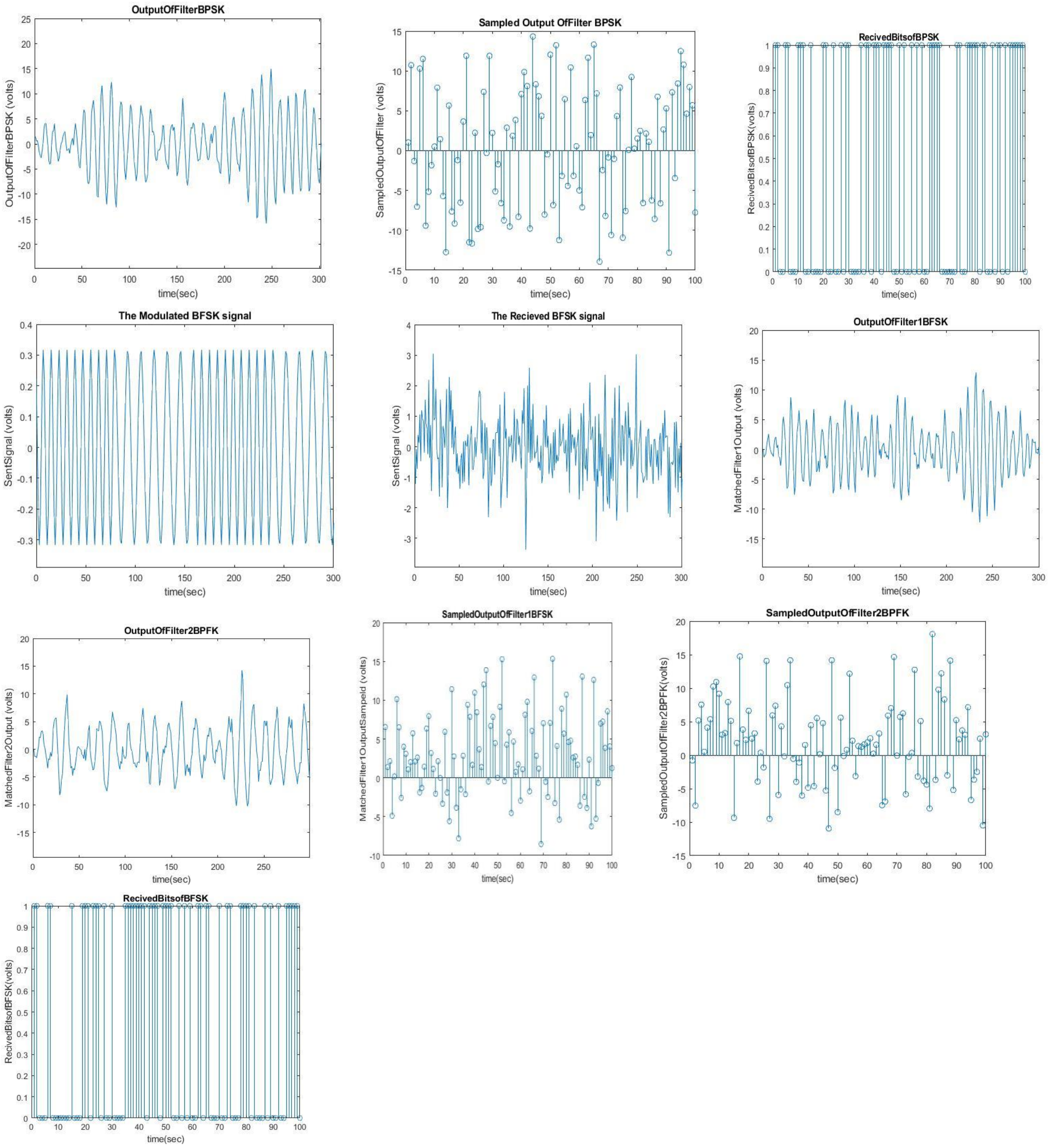


Figure 17

by Applying the Experiment again twenty Times and getting the average BER. With SNR=0

We got the following Results for BPSK and BFSK

BER|BPSK= 0.0805

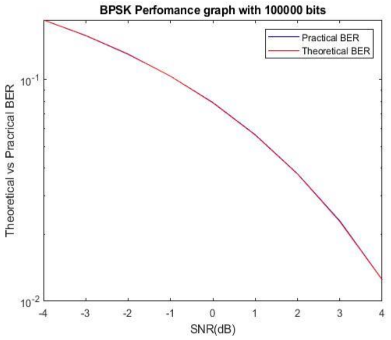
BER|BFSK= 0.1510

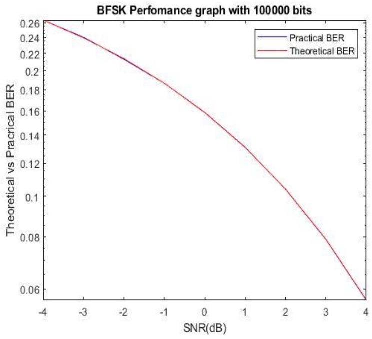
9) By plotting the BER graphs for 1000000 bits we found the following:

“Figure 18” BER for BPSK practical and theoretical

“Figure19” BER for BFSK practical and theoretical

“Figure20” both previous two figures on the same graph





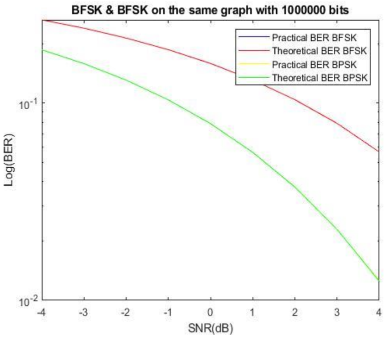


Figure 18

Figure 19

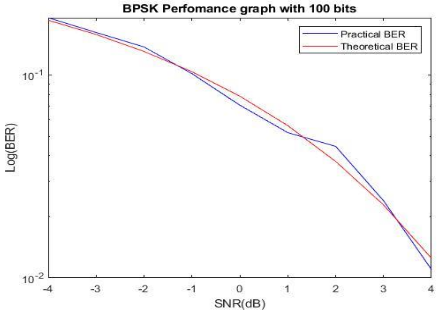
Figure20

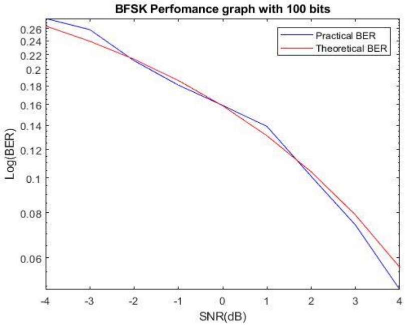
By plotting the BER graphs for 100 bits we found the following:

“Figure 21” BER for BPSK practical and theoretical

“Figure22” BER for BFSK practical and theoretical

“Figure23” both previous two figures on the same graph





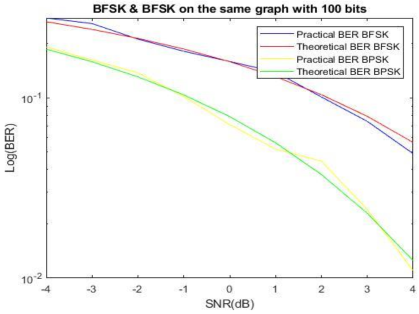


Figure 21

Figure 22

Figure23

3)Comments:

- In our Experiment we have approved that the theoretical BER for both BFSK and BPSK is correct
- The BER of BFSK is lower than that of BPSK .
- The theoretical results match the practical as we increase the number of sent bits.

4)Appendix:

Main.m

```
clc;
clear;
%Author:Ahmed A.Allam 27/11/2019
%We Will divide the Project into functions for Modularity
%This functions are of basic blocks mentioned in the Project Document
Tb=40;
No=2;

SNRe=-4:1:4;
BER_Pr_BPSK=zeros(1,length(SNRe));
BER_Pr_BFSK=zeros(1,length(SNRe));
BER_Th_BPSK=zeros(1,length(SNRe));
BER_Th_BFSK=zeros(1,length(SNRe));
No_of_sent_bits=100000;
for i=1:length(SNRe)
   for k=1:20
    RealizationError=BPSK(SNRe(i),No_of_sent_bits);
    BER_Pr_BPSK(i)=BER_Pr_BPSK(i)+RealizationError;
   end
   BER_Pr_BPSK(i)=BER_Pr_BPSK(i)/20;

   for k=1:20
    RealizationError=BFSK(SNRe(i),No_of_sent_bits);
    BER_Pr_BFSK(i)=BER_Pr_BFSK(i)+RealizationError;
   end
   BER_Pr_BFSK(i)=BER_Pr_BFSK(i)/20;
   A=sqrt((10^(SNRe(i)/10))*2*(No/Tb));
   %We have made all of Our Calculations based on Effective SNR
   %So here in "Thoritical BER" we will use Effective SNR
   BER_Th_BPSK(i)=0.5*erfc(sqrt(((A^2*Tb)/2)/No));
   BER_Th_BFSK(i)=0.5*erfc(sqrt(((A^2*Tb)/2)/(No*2)));
   %I have Knew that when it didn't go well

end

figure(1);
semilogy(SNRe,BER_Pr_BPSK,'b');
hold on;
semilogy(SNRe,BER_Th_BPSK,'r');
title(['BPSK Perfomance graph with ',num2str(No_of_sent_bits),' bits']);
xlabel('SNR(dB)');
ylabel('Log(BER)');
legend('Practical BER','Theoretical BER');
hold off;
figure(2);
semilogy(SNRe,BER_Pr_BFSK,'b');
hold on;
semilogy(SNRe,BER_Th_BFSK,'r');
title(['BFSK Perfomance graph with ',num2str(No_of_sent_bits),' bits']);
xlabel('SNR(dB)');
ylabel('Log(BER)');
legend('Practical BER','Theoretical BER');
figure(3)
semilogy(SNRe,BER_Pr_BFSK,'b');
hold on;
semilogy(SNRe,BER_Th_BFSK,'r');
hold on;
semilogy(SNRe,BER_Pr_BPSK,'p');
```

```
hold on;
semilogy(SNRe,BER_Th_BPSK,'g');
title(['BFSK & BFSK on the same graph with ',num2str(No_of_sent_bits),' bits']);
xlabel('SNR(dB)');
ylabel('Log(BER)');
legend('Practical BER BFSK','Theoretical BER BFSK','Practical BER BPSK','Theoretical BER
BPSK');
```

BPSK.m

```
function [BER] = BPSK(SNRe,No_of_sent_bits)

Tb=40;
No=2;
% No_of_sent_bits=100;%
% SNRe=2;
%first of all we will generate the random bits that will be used in the
%whole project [EquiProbable as given]
InputBits=round(rand(1,No_of_sent_bits));
%[t,x] = PolarNRZ(InputBits,Tb,No_of_sent_bits);
%going on with BPSK
A=sqrt((10^(SNRe/10))*2*(No/Tb));
[ModulatedBPSK,t,Wc]=ModulatorBPSK(InputBits,A);

%now we have to deal with the channel the channel is ideal but it adds AWGN
%to the signal

Noise=wgn(1,length(ModulatedBPSK),10*log10(No/2));
RecivedSignal=Noise+ModulatedBPSK;
% plot(t,RecivedSignal);%
% hold off%
% figure(7);
% plot(t,Noise);%
%title("White Gaussuian Noise with Sp(f)=1");
% xlabel('time(sec)');
%ylabel('Noise (volts)');
% figure(7);
% plot(t,ModulatedBPSK);%
% title("The Modulated BPSK signal");
% xlabel('time(sec)');
% ylabel('SentSignal (volts)');
%
% figure(7);
% plot(t,RecivedSignal);%
% title("The Recived BPSK signal");
% xlabel('time(sec)');
% ylabel('SentSignal (volts)');

%we will now Go on with the BPSK Demodulator and the matched filter
RecivedSampels=MatchedFilterBPSKAndSampler(RecivedSignal,Tb,Wc,No_of_sent_bits);
RecivedBits=DecisionMakingBPSK(RecivedSampels,InputBits);
BER=ErrorCaluculation(InputBits,RecivedBits);
end
```

ModulatorBPSK.m

```
function [ModulatedBPSK,t,Wc] = ModulatorBPSK(InputBits,A)
Tb=40; %bit duration
N=40;
Ts=Tb/N;
 Wc = (4*2*pi)/Tb;
```

```
% Ws=(N*2*pi)/Tb;
% %now we need to generate Polar NRZ signal
% % Another function will Be used to generate The PolarNRZ Signal

% ModulatedBPSK=A*x.*cos(Wc*t);
% figure (1);
% plot(t,ModulatedBPSK);
% hold on
%We will use another Approach that the commented;as it takes a Shorter time
InterPoint=0;
for i=1:length(InputBits)
    t_Sample=Ts:Ts:Tb;
    InterPoint=InterPoint+1;
    if (InputBits(i)==1)
        ModulatedBPSK(1,InterPoint:InterPoint+(Tb/Ts)-1)=A*cos(Wc*t_Sample);
    else
        ModulatedBPSK(1,InterPoint:InterPoint+(Tb/Ts)-1)=-A*cos(Wc*t_Sample);
    end
InterPoint=InterPoint+(Tb/Ts)-1;
end
t=0:1:length(ModulatedBPSK)-1;

end
```

MatchedFilterBPSKAndSampler.m

```
function [SampledOutputOffilter] =
MatchedFilterBPSKAndSampler(ReceivedSignal,Tb,Wc,NO_Of_bits)

%We have found it Hard to use correlator so we have decided to use matched
%filter
%We know that we only need to deal with amplitude So one Matched filter is
%ok

t=0:1:Tb-1; %Tb times
t_Filter=Tb-t;
MatchedFilter= cos(Wc * t_Filter);
OutputOFFilter=conv(ReceivedSignal,MatchedFilter);
SampledOutputOffilter=zeros(1,NO_Of_bits);
for i=1:NO_Of_bits
    SampledOutputOffilter(i)=OutputOFFilter(i*(Tb));
end

end
```

DecisionMakingBPSK.m

```
function [RecivedBits] = DecisionMakingBPSK(FilterSamples,GeneratedBits)

%Here we see According to zero Theresold if the Filterd sample is postive
%or negative and accordingly we "Predict" if 1 or 0 was sent
RecivedBits=zeros(1,length(GeneratedBits));
for i=1:length(GeneratedBits)
        if (FilterSamples(i)>0)
            RecivedBits(i)=1;
             %the critical value zero is choosen to be zero
        end
end

end
```

BFSK.m

```
function [BER] = BFSK(SNRe,No_of_sent_bits)



Tb=40;
No=2;
% No_of_sent_bits=100;%
% SNRe=2;%
 %first of all we will generate the random bits that will be used in the
%whole project [EquiProbable as given]
InputBits=round(rand(1,No_of_sent_bits));
%going on with BFSK
A=sqrt((10^(SNRe/10))*2*(No/Tb));
%As we see the Previous part of the code is the same of BPSK
[ModulatedBFSK,t,Wc1,Wc2]=ModulatorBFSK(InputBits,A);
%now we have to deal with the channel the channel is ideal but it adds AWGN
%to the signal

Noise=wgn(1,length(ModulatedBFSK),10*log10(No/2));
RecivedSignal=Noise+ModulatedBFSK;
% figure(7);
% plot(t,ModulatedBFSK);%
% title("The Modulated BFSK signal");
% xlabel('time(sec)');
% ylabel('SentSignal (volts)');
%
% figure(8);
% plot(t,RecivedSignal);%
% title("The Recieved BFSK signal");
% xlabel('time(sec)');
% ylabel('SentSignal (volts)');



%we will now Go on with the BFSK Demodulator and the matched filter
[SampledOutputFilter1,SampledOutputFilter2]=MatchedFilterBFSKAndSampler(RecivedSignal,Tb,
Wc1,Wc2,No_of_sent_bits);
RecivedBits=DecisionMakingBFSK(SampledOutputFilter1,SampledOutputFilter2,InputBits);
BER=ErrorCaluculation(InputBits,RecivedBits);




end
```

ModulatorBFSK.m

```
function [ModulatedBFSK,t,W1,W2] = ModulatorBFSK(InputBits,A)

Tb=40; %bit duration
N=40;
n0=4;
n1=1;
Ts=Tb/N;
W1=(2*pi*(n0+n1))/Tb;
W2=(2*pi*(n0-n1))/Tb;

%We will do it same as the Document but in one shoot
InterPoint=0;
for i=1:length(InputBits)
```

```
%It's used for evrey sample to generate Cosine
    t_Sample=Ts:Ts:Tb;
    InterPoint=InterPoint+1;
    if (InputBits(i)==1)
        ModulatedBFSK(1,InterPoint:InterPoint+(Tb/Ts)-1)=A*cos(W1*t_Sample);
    else
        ModulatedBFSK(1,InterPoint:InterPoint+(Tb/Ts)-1)=A*cos(W2*t_Sample);
    end
    InterPoint=InterPoint+(Tb/Ts)-1;
end
t=0:1:length(ModulatedBFSK)-1;
% figure (1);
% plot(t,ModulatedBFSK);
% hold on

end
```

MatchedFilterBFSKAndSampler.m

```
function
[SampledOutputOfFilter1,SampledOutputOfFilter2]=MatchedFilterBFSKAndSampler(ReceivedSigna
l,Tb,Wc1,Wc2,No_of_sent_bits)
%We have found it Hard to use correlator so we have decided to use matched
%filter
%We know that we only need to deal with Two Basis Functions So Tw Matched
%filters is needed

t=0:1:Tb-1; %Tb times
t_Filter=Tb-t;
MatchedFilter1= cos(Wc1 * t_Filter);
MatchedFilter2= cos(Wc2 * t_Filter);
%We are going to deal with evrey matched Filter on its own
%Now we are going to get the Output of Matched Filter
OutputOFFilter1=conv(ReceivedSignal,MatchedFilter1);
OutputOFFilter2=conv(ReceivedSignal,MatchedFilter2);
%Now we are sampling both Output Signals
SampledOutputOfFilter1=zeros(1,No_of_sent_bits);
for i=1:No_of_sent_bits
    SampledOutputOfFilter1(i)=OutputOFFilter1(i*(Tb));
end


SampledOutputOfFilter2=zeros(1,No_of_sent_bits);
for i=1:No_of_sent_bits
    SampledOutputOfFilter2(i)=OutputOFFilter2(i*(Tb));
end




end
```

DecisionMakingBFSK.m

```
function
RecivedBits=DecisionMakingBFSK(SampledOutputFilter1,SampledOutputFilter2,GeneratedBits)

%Here we see According to which is Greater the Sample of the first matched
%filter or the second and accordingly we "Predict" if 1 or 0 was sent
```

```
RecivedBits=zeros(1,length(GeneratedBits));
for i=1:length(GeneratedBits)
        if (SampledOutputFilter1(i)>SampledOutputFilter2(i))
            RecivedBits(i)=1;
             %the critical value if they are equal is choosen to be zero
        end
end

end
```

ErrorCaluculation.m

```
function [BER] = ErrorCaluculation(SentBits,RecivedBits)
%Here we see how many bits are wrong
OutOfOrderBits=0;
for i=1:length(SentBits)
    if(SentBits(i)~=RecivedBits(i))
        OutOfOrderBits=OutOfOrderBits+1;
    end
end
%So bit error is the ratio how many bits are wrong and total bits
BER=OutOfOrderBits/length(SentBits);
End
```

PolarNRZ.m

```
function [t,x] = PolarNRZ(InputBits,Tb,No_of_sent_bits)

T = No_of_sent_bits*Tb;
N=40;
Ws=(N*2*pi)/Tb;
n =40;
t = 0:(2*pi)/Ws:T;

x = zeros(1,length(t)); % output signal

for i = 0:length(InputBits)-1
  if InputBits(i+1) == 1
    x(i*n+1:(i+1)*n) = 1;
  else
    x(i*n+1:(i+1)*n) = -1;
  end
end
% figure(1);
%  plot(t,x);
% title(['Polar NRZ line Code ',num2str(No_of_sent_bits),' bits']);
% xlabel('time(sec)');
% ylabel('Polar NRZ signal (volts)');
%
%  [PSD,F] = periodogram(x,[],[],1);
%   figure;
%   plot(F,PSD);
%   title("PSD of PolarNRZ Line Code");
%   xlabel('Freq(HZ)');
% ylabel('PSD (volts/HZ)');
end
```