EECS 455 Solution to Problem Set 3

1. (a) Is it possible to have reliably communication with a data rate of 2.5Mbps using power $P = 3 \times 10^{-12}$ Watts with a bandwidth of $W = 1$ MHz and a noise power spectral density of $\frac{N_0}{2} = \frac{10^{-18}}{2}$ Watts/Hz?

   Soltuion: According to Shannon's theorem reliable communication is possible if

$$\begin{aligned} R &< W \log_2(1 + \frac{P}{N_0 W}) \\ &= 10^6 \log_2(1 + \frac{3 \times 10^{-12}}{10^{-18} 10^6}) \\ &= 10^6 \log_2(1+3) \\ &= 10^6 \log_2(4) \\ &= 10^6 \times 2 \\ &= 2\text{Mbps} \end{aligned}$$

   Thus reliable communication is possible upto the rate of 2Mbps. So 2.5Mbps is not possible.

   (b) A communication system uses BPSK in a (null-to-null) bandwidth of 1 MHz with power $P = 5 \times 10^{-12}$ Watts in the presence of white Gaussian noise with two-side power spectral density $\frac{N_0}{2} = \frac{10^{-18}}{2}$. An error probability of $Q(\sqrt{20})$ is desired. What data rate is possible?

   Soltuion: The error probability for BPSK is

$$P_e = Q(\sqrt{\frac{2E}{N_0}})$$

   So the signal to noise ratio must satisfy

$$\begin{aligned} \frac{E}{N_0} &= 10 \\ \frac{PT}{N_0} &= 10 \\ \frac{P/R}{N_0} &= 10 \\ R &= \frac{P}{10 N_0} \\ &= \frac{5 \times 10^{-12}}{10 \times 10^{-18}} \\ &= 5 \times 10^5 \end{aligned}$$

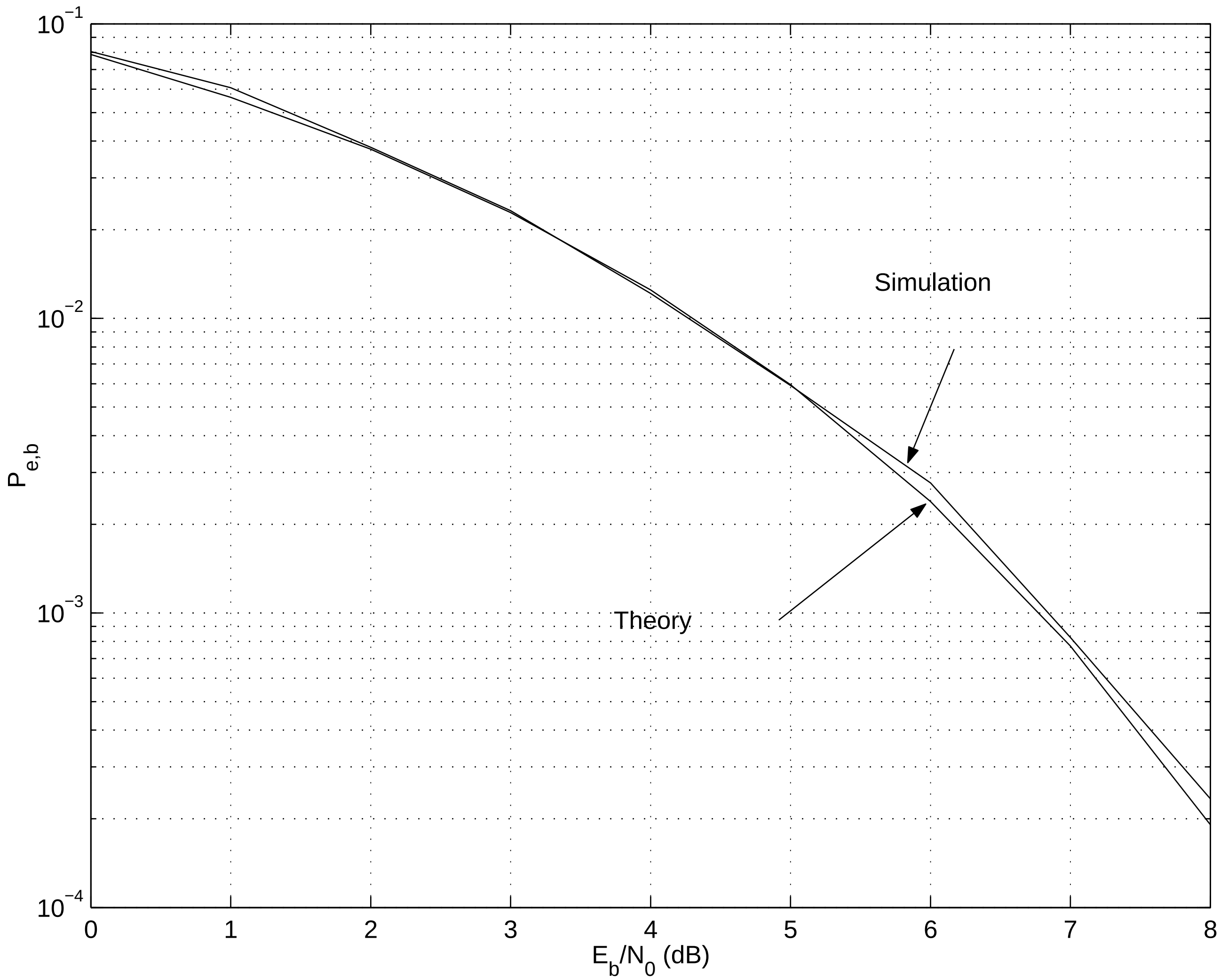
   Thus there is enough power to provide reliable communication at a data rate of 500 kbps. In addition, at that data rate the (null-to-null) bandwidth is 1MHz so there is also enough bandwidth.

(c) For the same parameters as part (b) except the error probability requirement was $Q(\sqrt{2})$ what data rate would be possible?

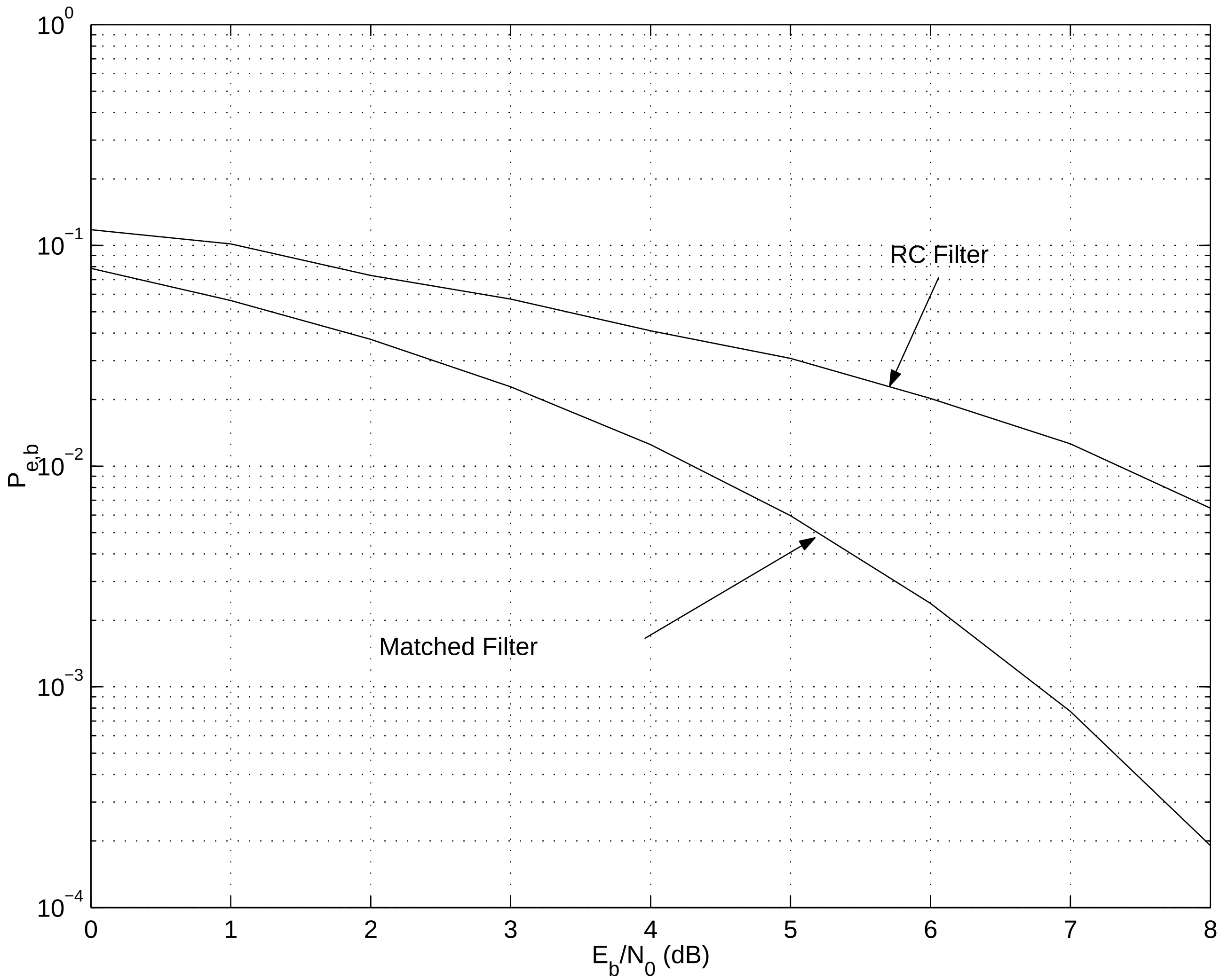
Solution: Since the energy requirement has been reduced by a factor of 10 the data rate could potentially increase by a factor of 10. However, the bandwidth would not allow it. Thus the maximum data rate is still 500kbps.

2. (a) Simulate a communication system that uses rectangular pulses of amplitude ± 1 to transmit data. The channel is an additive white Gaussian noise channel. The receiver uses a matched filter followed by a sampler and a decision device. For $E_b/N_0 = 0, 1, ..., 8$ determine (and plot) the simulated bit error probability. Compare with the theoretical bit error probability.

   Solution The Matlab code is attached.



   (b) Repeat the above experiment except with a RC filter with impulse response $h(t) = \alpha e^{-\alpha t} u(t)$. Use the value $\alpha = 1.256/T$. Compare the result to part (a).




```
%  This program simulates a rectangular pulse shape signal
%  in the presence of noise
clear;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                         %
%                 Simulation Parameters                   %
%                                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


Tb=1;                     % Bit duration of data


Nbcount=input('Number of bit errors counted = ');
Nbp=input('Number of bits in a packet= ');
Nsb=input('Number of samples per bit= ');
```

```
fmax=Nsb/(2*Tb)            % Simulation bandwidth
N=2*Nbp*Nsb;               % Simulation samples
N2=N/2;                    % Half the number of samples
fs=2*fmax;                 % Sampling Frequency
df=2*fmax/N                % Frequency spacing
dt=1./(df.*N)              % Time spacing
t=(1:N)*dt-dt;             % Time samples
Tmax=N*dt                  % Simulation time
f=(1:N)*df-df;             % Frequency samples
f2=f-N/2*df;
rb=1/Tb;                   % Data rate
fc=0                       % Center Frequency


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                              %
%                  Generate Pulse Shape                        %
%                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
x1f(1:N)=ones(1,N)*1e-80;
x1f(1:N/2)=x1f(1:N/2)+Tb*sinc(f(1:N/2)*Tb).*exp(-j*pi*f(1:N/2)*Tb);
x1f(N/2+2:N)=conj(fliplr(x1f(2:N/2)));
x1t=real(ifft(x1f)./dt);

Peb=zeros(1,9);

for ml=1:9
EbN0dB(ml)=ml-1
EbN0=10^(EbN0dB(ml)/10);
P=1;                        % Received Power
Eb=P*Tb;                   % Received Energy
N0=Eb/EbN0;                % Noise power

Nberrors(ml)=0;
Np=0;
while Nberrors(ml)<Nbcount

b=sign(rand(1,Nbp)-0.5);
zf=zeros(1,N);
for k=1:Nbp
   zf=zf+b(k)*exp(-j*2*pi*f*(k-1)*Tb);
end

x2f=x1f.*zf;
```

```
x2t=real(ifft(x2f)./dt);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                         %
%                  Generate Noise                         %
%                                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

sigma=sqrt(N0*fmax);
nt=sigma*randn(1,N);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                         %
%                  Add signal to noise                    %
%                                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
rt=x2t+nt;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                         %
%                  Filter the signal and noise            %
%                                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

rf=fft(rt)*dt;
yf=rf.*x1f;
yt=real(ifft(yf)./dt);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                         %
%                  Sample the filter output               %
%                                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
sampling_time=(1:Nbp)*Nsb+1;
bhat=sign(yt(sampling_time));
Np=Np+1;
Nberrors(ml)=Nberrors(ml)+sum(sign(abs(b-bhat)));
if (mod(Np,100) == 0)
[Np, Nberrors(ml)]
end;
end

Peb(ml)=Nberrors(ml)/(Np*Nbp)
petheory(ml)=q(sqrt(2*EbN0))
```

```
end
hold off
semilogy(EbN0dB,Peb)
hold on
semilogy(EbN0dB,petheory,’r’)
```

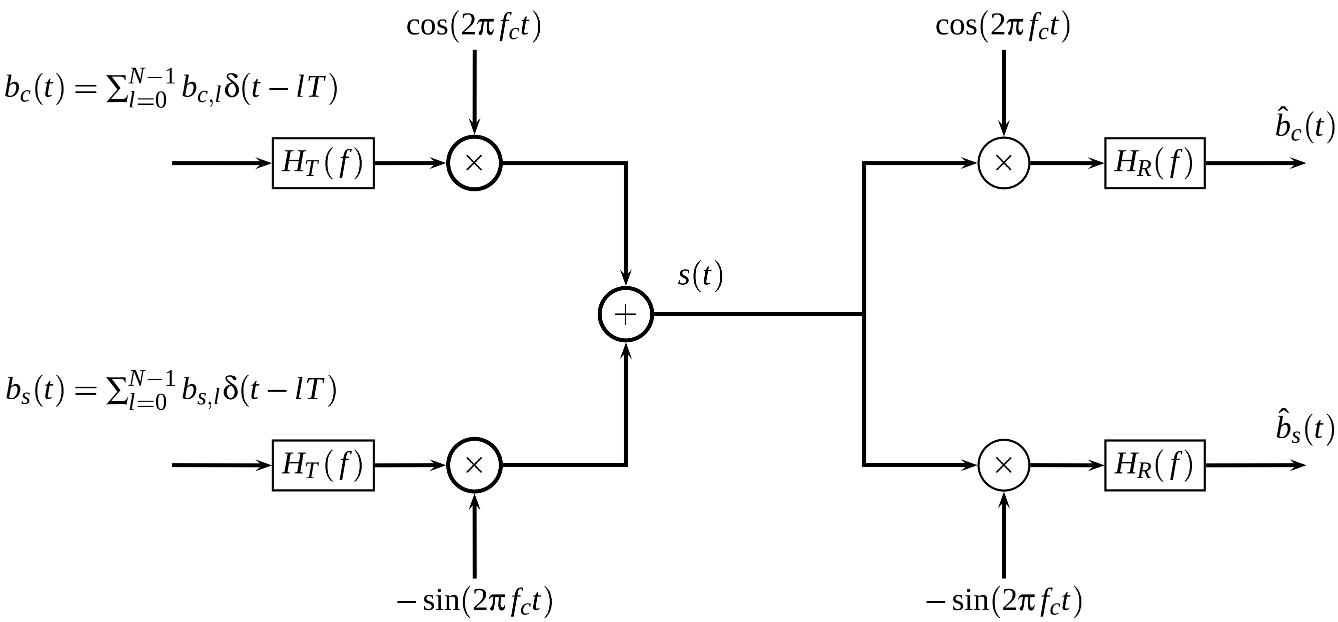
```
%  This program simulates a rectangular pulse shape signal
%  in the presence of noise with an RC receiver filter
clear;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                         %
%                Simulation Parameters                    %
%                                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


Tb=1;                        % Bit duration of data
alpha=input('RC Filter Parameter=');
Nbcount=input('Number of bit errors counted = ');
Nbp=input('Number of bits in a packet= ');
Nsb=input('Number of samples per bit= ');

fmax=Nsb/(2*Tb)              % Simulation bandwidth
N=2*Nbp*Nsb;                 % Simulation samples
N2=N/2;                      % Half the number of samples
fs=2*fmax;                   % Sampling Frequency
df=2*fmax/N                  % Frequency spacing
dt=1./(df.*N)                % Time spacing
t=(1:N)*dt-dt;               % Time samples
Tmax=N*dt                    % Simulation time
f=(1:N)*df-df;               % Frequency samples
f2=f-N/2*df;
rb=1/Tb;                     % Data rate
fc=0                         % Center Frequency


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                         %
%                Generate Pulse Shape                     %
%                                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
x1f(1:N)=ones(1,N)*1e-80;
x1f(1:N/2)=x1f(1:N/2)+Tb*sinc(f(1:N/2)*Tb).*exp(-j*pi*f(1:N/2)*Tb);
x1f(N/2+2:N)=conj(fliplr(x1f(2:N/2)));
x1t=real(ifft(x1f)./dt);

Peb=zeros(1,9);

for ml=1:9
EbN0dB(ml)=ml-1
EbN0=10^(EbN0dB(ml)/10);
```

```
P=1;                       % Received Power
Eb=P*Tb;                  % Received Energy
N0=Eb/EbN0;               % Noise power

Nberrors(ml)=0;
Np=0;
while Nberrors(ml)<Nbcount

b=sign(rand(1,Nbp)-0.5);
zf=zeros(1,N);
for k=1:Nbp
   zf=zf+b(k)*exp(-j*2*pi*f*(k-1)*Tb);
end

x2f=x1f.*zf;
x2t=real(ifft(x2f)./dt);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                          %
%                 Generate Noise                           %
%                                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

sigma=sqrt(N0*fmax);
nt=sigma*randn(1,N);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                          %
%                 Add signal to noise                      %
%                                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
rt=x2t+nt;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                          %
%                 Filter the signal and noise              %
%                                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                x3f(1:N/2)=alpha./(alpha+j*2*pi*f(1:N/2));
                x3f(N/2+2:N)=conj(fliplr(x3f(2:N/2)));

rf=fft(rt)*dt;
yf=rf.*x3f;
yt=real(ifft(yf)./dt);
```

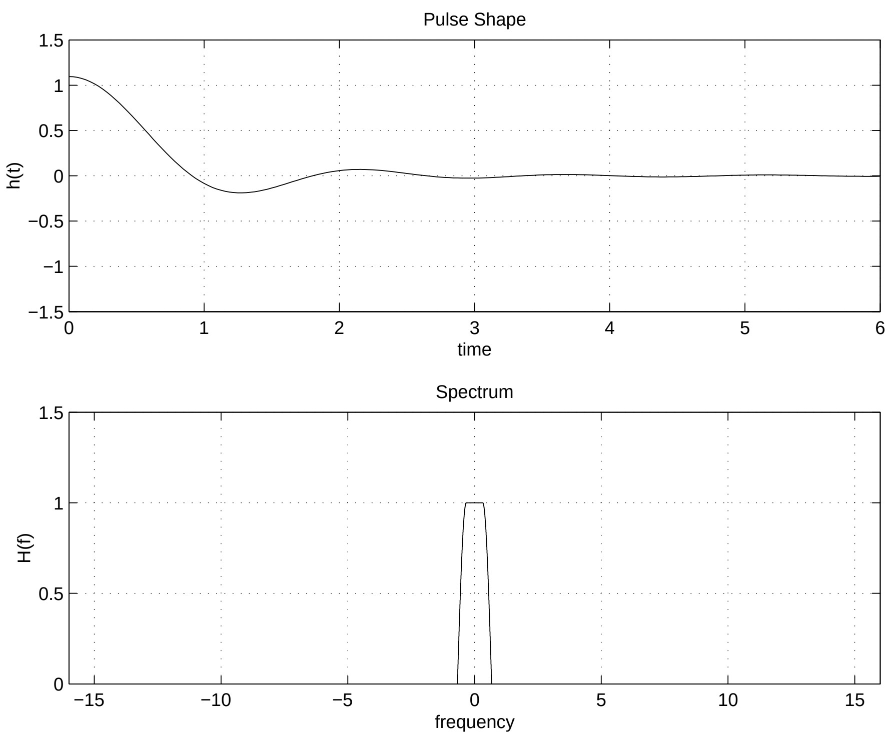
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                       %
%                 Sample the filter output              %
%                                                       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
sampling_time=(1:Nbp)*Nsb+1;
bhat=sign(yt(sampling_time));
Np=Np+1;
Nberrors(ml)=Nberrors(ml)+sum(sign(abs(b(9:Nbp)-bhat(9:Nbp))));
if (mod(Np,10) == 0)
[Np, Nberrors(ml)]
end;
end           %loop for counting errors

Peb(ml)=Nberrors(ml)/(Np*(Nbp-8))

end           %loop for different SNR
hold off
semilogy(EbN0dB,Peb)
hold on
```

3. Simulate the communication system shown below. The system uses square-root raised cosine pulses with $\beta = 0.35$ data amplitude ± 1 to transmit data on the cosine and sine branch. Show the data waveform, the transmitted signal in the time domain $s(t)$ and frequency domain $S(f)$. Plot the output of the mixers at the receiver in the frequency domain (show the double frequency terms). Plot the waveform at the output of the receiver filters for a sequence of 8 bits. Make an eye diagram for one of the outputs (use 512 bits for the eye diagram). Assume $T = 1$ and $f_c = 4$ for the simulation. Use a maximum frequency for the simulation of 16Hz.

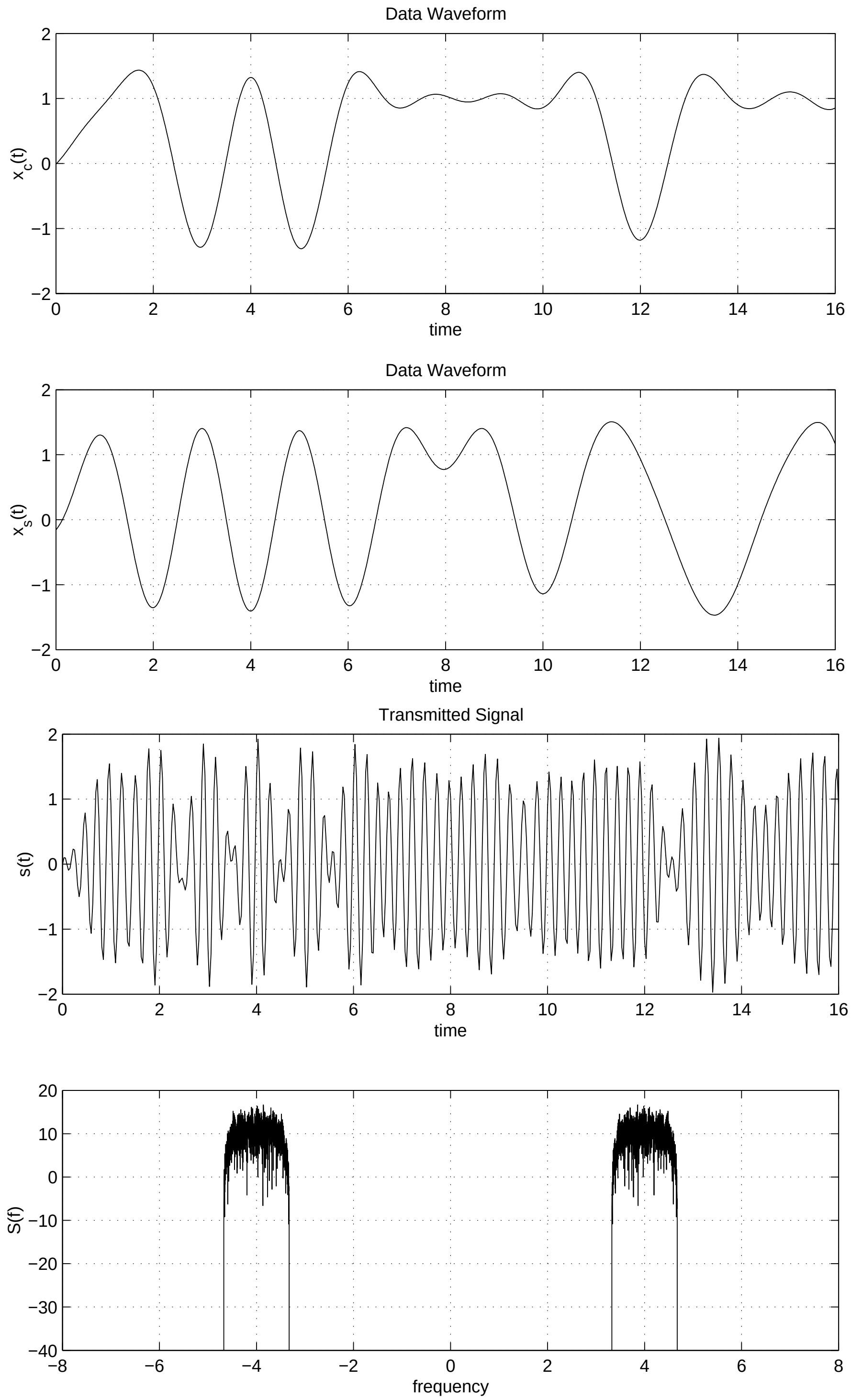
Solution:

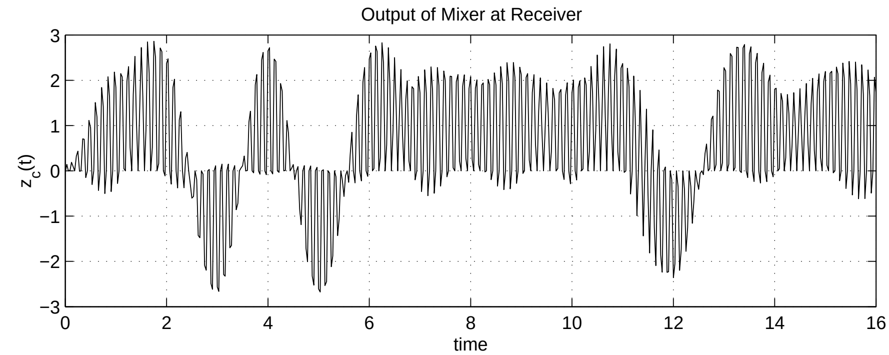


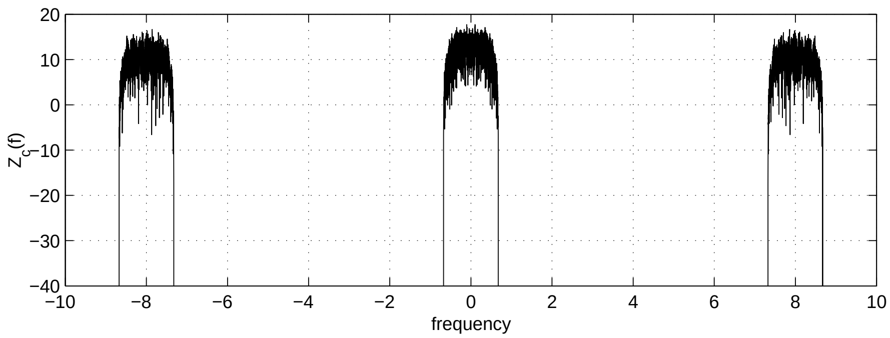
$b_c(t) = \sum_{l=0}^{N-1} b_{c,l}\delta(t - lT)$
$b_s(t) = \sum_{l=0}^{N-1} b_{s,l}\delta(t - lT)$
$H_T(f)$
$H_T(f)$
$\cos(2\pi f_c t)$
$-\sin(2\pi f_c t)$
$s(t)$
$\cos(2\pi f_c t)$
$-\sin(2\pi f_c t)$
$H_R(f)$
$H_R(f)$
$\hat{b}_c(t)$
$\hat{b}_s(t)$

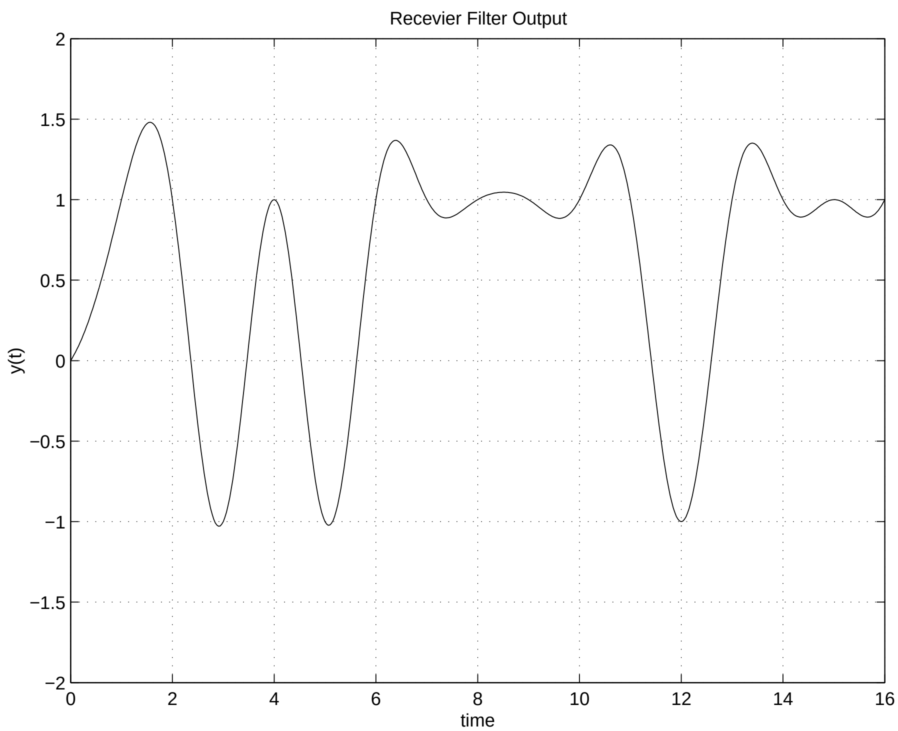


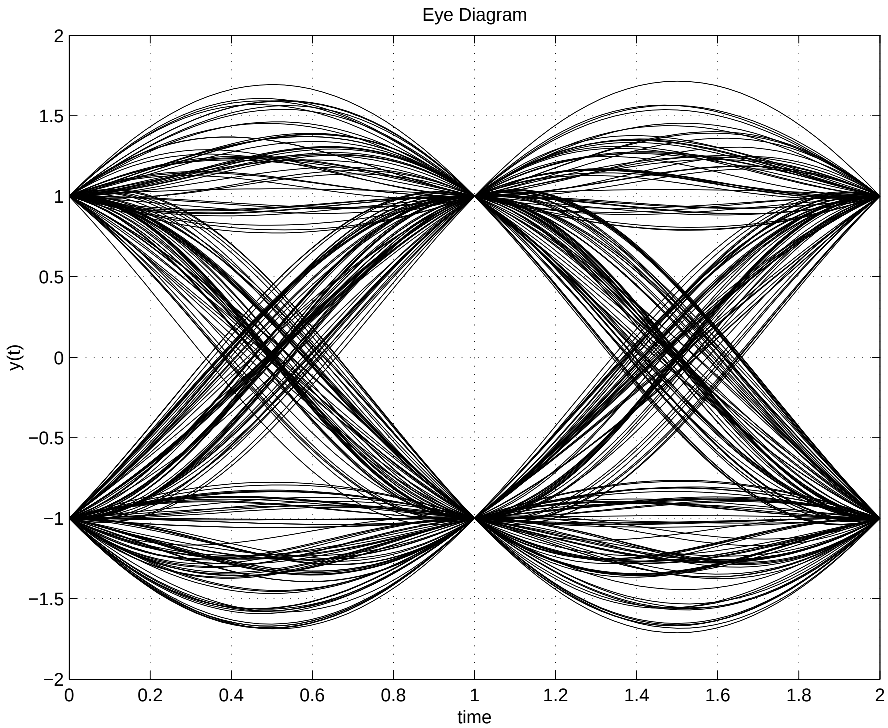
Pulse Shape
h(t)
time
Spectrum
H(f)
frequency


Data Waveform
x_c(t)
time
Data Waveform
x_s(t)
time
Transmitted Signal
s(t)
time
S(f)
frequency


Output of Mixer at Receiver
$z_c(t)$
time
3
2
1
0
−1
−2
−3
0
2
4
6
8
10
12
14
16


$Z_c(f)$
frequency
20
10
0
−10
−20
−30
−40
−10
−8
−6
−4
−2
0
2
4
6
8
10


Recevier Filter Output
y(t)
time
2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
0
2
4
6
8
10
12
14
16




```
%  This program simulates a raised cosine pulse shape signal
%  filtered by an raised cosine filter.

clear;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                        %
%                Simulation Parameters                   %
%                                                        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


Tb=1;                     % Bit duration of data
%EbN0dB=input('E_b/N_0 (dB)= ');
%EbN0=10^(EbN0dB/10);
%P=1;                       % Received Power
%Eb=P*Tb;                  % Received Energy
%N0=Eb/EbN0;               % Noise power
beta=input('Raised Cosine Filter Parameter=');


Nb=input('Number of bits simulated= ');
fmax=input('Maximum frequency of simulation= ');
N=input('Number of samples = ');

N2=N/2;                   % Half the number of samples
```

```
fs=2*fmax;                  % Sampling Frequency
df=2*fmax/N                 % Frequency spacing
dt=1./(df.*N)               % Time spacing
Nsb=Tb/dt
t=(1:N)*dt-dt;              % Time samples
Tmax=N*dt                   % Simulation time
f=(1:N)*df-df;              % Frequency samples
f2=f-N/2*df;
rb=1/Tb;                    % Data rate
fc=4                        % Center Frequency

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                       %
%                 Generate Signals                      %
%                                                       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
N1=round((1-beta)/(2*Tb)/df)
N2=round((1+beta)/(2*Tb)/df)
h1f(1:N)=zeros(1,N);
h1f(1:N1)=sqrt(Tb*ones(1,N1));
h1f(N1:N2)=sqrt(0.5*Tb*(1+cos(pi*Tb/beta*(f(N1:N2)-(1-beta)/(2*Tb)))));
h1f(N/2+2:N)=conj(fliplr(h1f(2:N/2)));
h1t=real(ifft(h1f)./dt);

figure(1)
subplot(2,1,1), plot(t(1:8*Nsb)/Tb,h1t(1:8*Nsb));
grid
axis([0 6*Tb -1.5 1.5])
xlabel('time');
ylabel('h(t)');
title('Pulse Shape')
subplot(2,1,2), plot(f2(1:N),fftshift(abs(h1f(1:N))));
axis([-fmax fmax 0 1.5])
grid
xlabel('frequency');
ylabel('H(f)');
title('Spectrum')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                       %
%           Generate data for cosine branch             %
%                                                       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

bc=sign(rand(1,Nb)-0.5);
```

```
bcf=zeros(1,N);
for k=1:Nb
  bcf=bcf+bc(k)*exp(-j*2*pi*f*k*Tb);
end

xcf=h1f.*bcf;
xct=real(ifft(xcf)./dt);

figure(2)
subplot(2,1,1), plot(t/Tb,xct);
grid
axis([0 16 -2 2])
xlabel('time');
ylabel('x_c(t)');
title('Data Waveform')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                       %
%          Generate data for sine branch               %
%                                                       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

bs=sign(rand(1,Nb)-0.5);
bsf=zeros(1,N);
for k=1:Nb
  bsf=bsf+bs(k)*exp(-j*2*pi*f*k*Tb);
end

xsf=h1f.*bsf;
xst=real(ifft(xsf)./dt);

subplot(2,1,2), plot(t/Tb,xst);
grid
axis([0 16 -2 2])
xlabel('time');
ylabel('x_s(t)');
title('Data Waveform')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                       %
%                Mix the signal to a carrier            %
%                                                       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

st=xct.*cos(2*pi*fc*t)-xst.*sin(2*pi*fc*t);
```

```
sf=fft(st)*dt;

figure(3)
subplot(2,1,1), plot(t/Tb,st)
xlabel('time')
ylabel('s(t)')
grid
axis([0 16 -2 2])
subplot(2,1,2), plot(f2,10*log10(fftshift(abs(sf))))
grid
xlabel('frequency')
ylabel('S(f)')
axis([-8 8 -40 20])
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                       %
%               Mix the signal back to baseband         %
%                                                       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

zct=2*st.*cos(2*pi*fc*t);
zcf=fft(zct)*dt;
zst=-2*st.*sin(2*pi*fc*t);
zsf=fft(zst)*dt;

figure(4)
hold off
subplot(2,1,1), plot(t/Tb,zct)
axis([0 8 -3 3])
xlabel('time')
ylabel('z_c(t)')
subplot(2,1,2), plot(f2,10*log10(fftshift(abs(zsf))))
grid
axis([-10 10 -40 20])
xlabel('frequency')
ylabel('Z_c(f)')
title('Output of Mixer at Receiver')


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                       %
%               Filter the signal                       %
%                                                       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

yf=zcf.*h1f;
```

```
yt=real(ifft(yf)./dt);


figure(5)
subplot(1,1,1), plot(t/Tb,yt)
grid on
axis([0 16 -2 2])
xlabel('time')
ylabel('y(t)')
title('Recevier Filter Output')
hold off


figure(6)
%subplot(1,1,1), plot(t(1:2*Nsb+1)/Tb,yt(1:2*Nsb+1))
%grid
%yscale=2*ceil(max(yt/2));
%axis([0 2 -2 2])
for k=2:floor(Nb/2)-10
plot(t(1:2*Nsb+1)/Tb,yt((k*2)*Nsb+1:(k*2+2)*Nsb+1))
hold on
end
xlabel('time');
ylabel('y(t)');
title('Eye Diagram')
grid on
hold off
```