Cairo University

Faculty of Engineering

Electronics and Electrical Communication Department

Matched Filters, Correlators, ISI, and raised cosine filters

Submitted by:

Name: Ahmed Mokhtar Mahfouz Emam
Sec: 1
BN: 31
Email: Ahmed.mokhtar5483@gmail.com

Contents

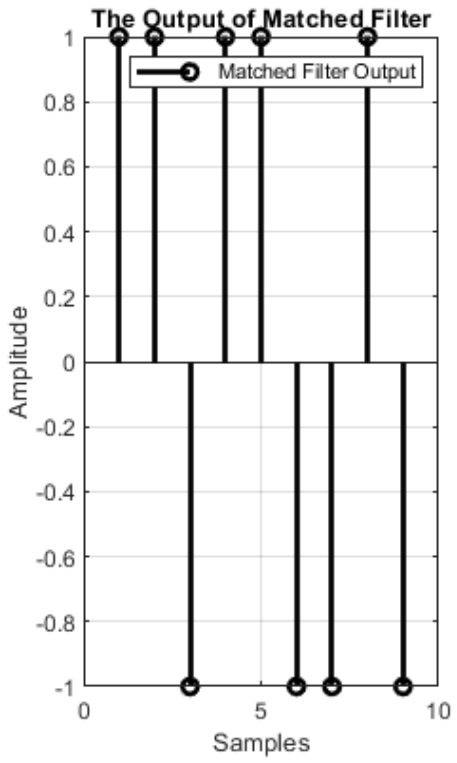
Generate the 10-bit data

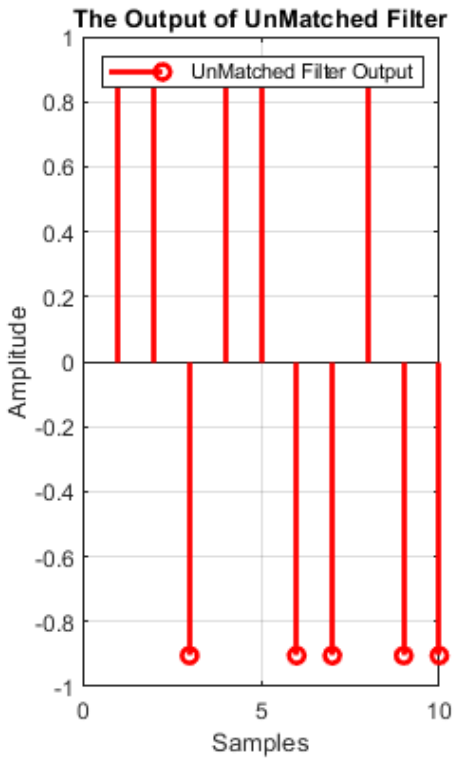
```
bits = randi([0 1], [1 10]);
```

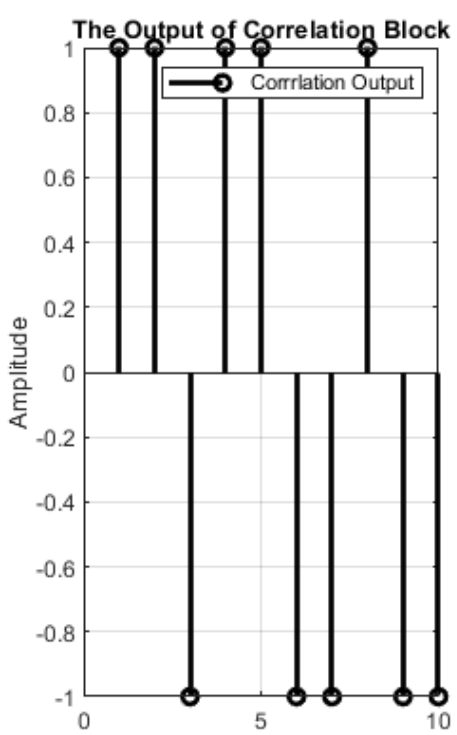
(1) Matched filters and correlators in noise free environment:

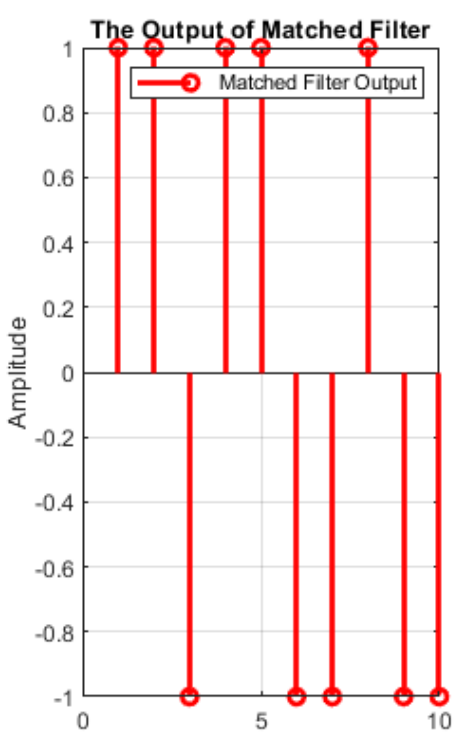
```
d = 2*bits -1;                      % Remaping the data form -1 to 1
data = upsample(d,5);               % Make to 1 or -1 then 4 zeroes
pulse_shaping=[5 4 3 2 1]/sqrt(55); % Design the filter
%%-------------------------------------------------------------------------
y = conv(data,pulse_shaping);         % Making the data analog to transmit
match_filter_1= fliplr(pulse_shaping);% Matched filter Reciver
filter_2 = [1 1 1 1 1]/sqrt(5);       % Rect filter reciver
out_Rx1 = conv(y,match_filter_1);     % output of matched filter reciver (y)
out_Rx2 = conv(y,filter_2);
out_Rx11=downsample(out_Rx1(5:end),5);
out_Rx11=out_Rx11(1:end-1);
out_Rx22=downsample(out_Rx2(5:end),5);
Signal1=out_Rx11(1:end-1);
Signal2=out_Rx22(1:end-1);
h=1;
y=y(1:end-4);
for i=1:5:length(y)
    out_corr(h)=y(i)*pulse_shaping(1)+y(i+1)*pulse_shaping(2)+y(i+2)*pulse_shaping(3)+y(i+3)*pulse_shaping(4)+y(i+4)*pulse_shaping(5);
    h=h+1;
end
% output of Rect filter reciver (y)
figure
subplot(1,2,1)
stem(Signal1,'k','LineWidth',2)
title('The Output of Matched Filter')
xlabel('Samples')
ylabel('Amplitude')
legend('Matched Filter Output')
grid on
subplot(1,2,2)
stem(Signal2,'r','LineWidth',2)
title('The Output of UnMatched Filter')
xlabel('Samples')
ylabel('Amplitude')
legend('UnMatched Filter Output')
grid on
figure                              % showing the outout figure
subplot(1,2,1)
stem(out_corr,'k','LineWidth',2)
title('The Output of Correlation Block')
xlabel('Samples')
ylabel('Amplitude')
legend('Corrrlation Output')
grid on
subplot(1,2,2)
stem(out_Rx11,'r','LineWidth',2)
title('The Output of Matched Filter')
xlabel('Samples')
ylabel('Amplitude')
legend('Matched Filter Output')
grid on
% %% Generate the 10000-bit data
bits = randi([0 1], [1 10000]);
% %% (2) Noise analysis:
N0 = 1./(10.^((-2:1:5)/20));
d = 2*bits -1;                      % Remaping the data form -1 to 1
data = upsample(d,5);               % Make to 1 or -1 then 4 zeroes
y = conv(data,pulse_shaping);       % Making the data analog to transmit(output of transmiter)
noise = zeros(1,length(y));         % initialize a vector for channel noise
noise_int = randn(1,length(y));     % creating intializing channel noise
y_Rx = zeros(length(N0),length(y)); % initialize a matrix for Input reciver(received signal)
out_Rx1 = zeros(length(N0),length(y)+4);
for i = 1:length(N0)
    noise =sqrt(N0(i)/2).*noise_int;   % Scaling the noise to the variance No/2
    y_Rx(i,:) = y+noise;                % adding noise to the input for the reciver
   out_Rx1(i,:) = conv(y_Rx(i,:),match_filter_1); % output of matched filter receiver with noise
   out_Rx44(i,:)=downsample(out_Rx1(i,5:end),5);
   Signal=out_Rx44(i,1:end-1);
   BER1(i)=sum((d>0)~=(Signal>0))/length(d);
end
 out_Rx2 = zeros(length(N0),length(y)+4);
 for i = 1:length(N0)
     noise = sqrt(N0(i)/2).*noise_int;   % Scaling the noise to the variance No/2
     y_Rx(i,:) = y+noise;                % adding noise to the input for the reciver
    out_Rx2(i,:) = conv(y_Rx(i,:),filter_2); % output of matched filter receiver with noise
     out_Rx33(i,:)=downsample(out_Rx2(i,5:end),5);
    Signal=out_Rx33(i,1:end-1);
    BER2(i)=sum((d>0)~=(Signal>0))/length(d);
 end
 BER = zeros(1,length(N0));
for i = 1: length(N0)
    BER(i) = 0.5*erfc(sqrt(1/ N0(i)));
```

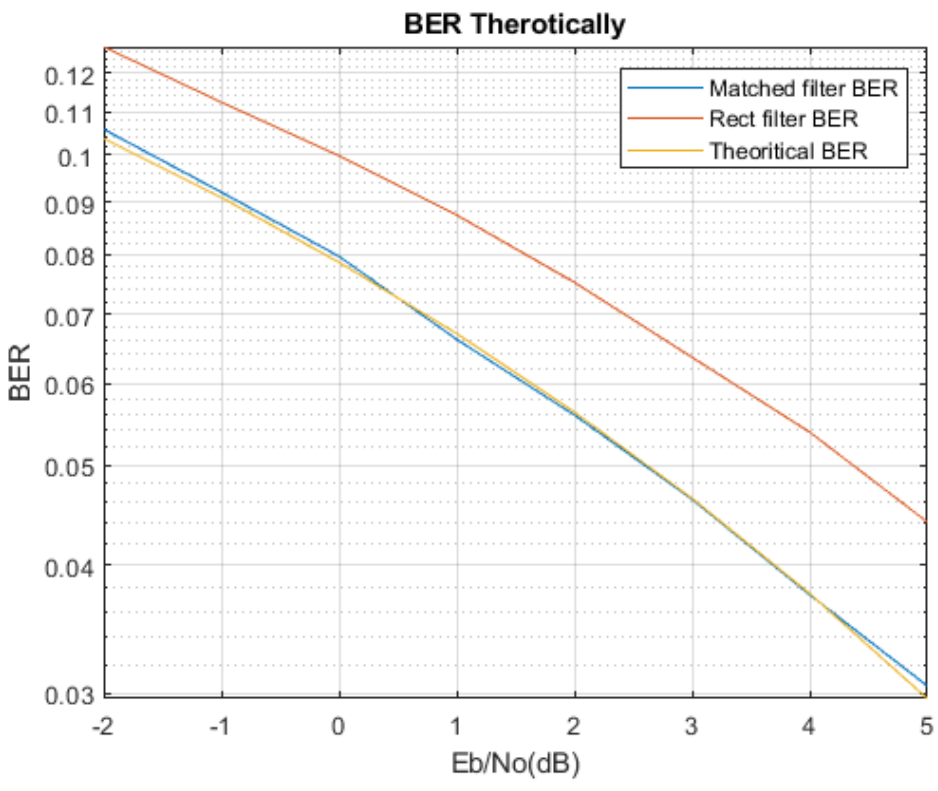
```
end
figure
semilogy([-2:1:5],BER1)
hold on
semilogy([-2:1:5],BER2)
hold on
semilogy([-2:1:5],BER)
title('BER Therotically')
xlabel('Eb/No(dB)')
ylabel('BER')
legend('Matched filter BER','Rect filter BER','Theoritical BER')
grid on
```









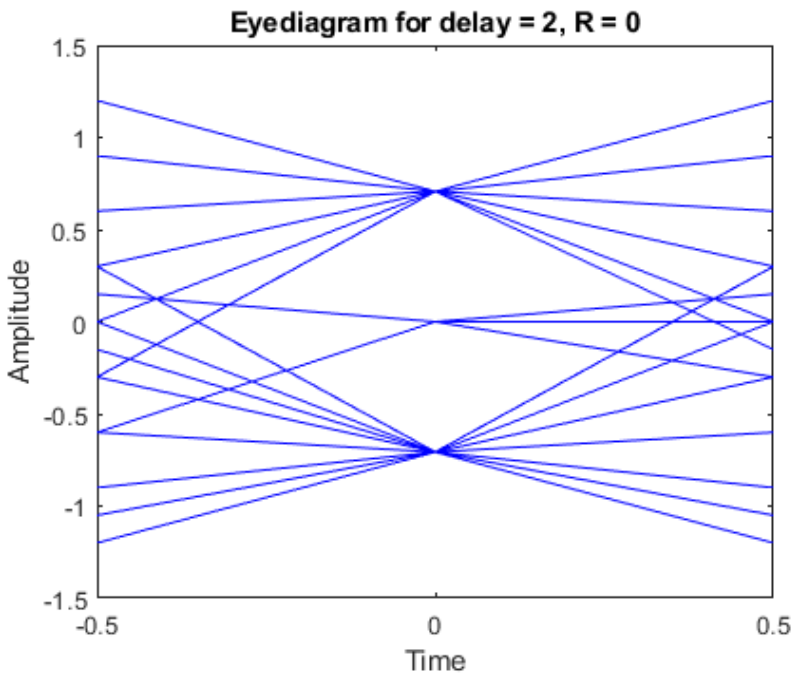


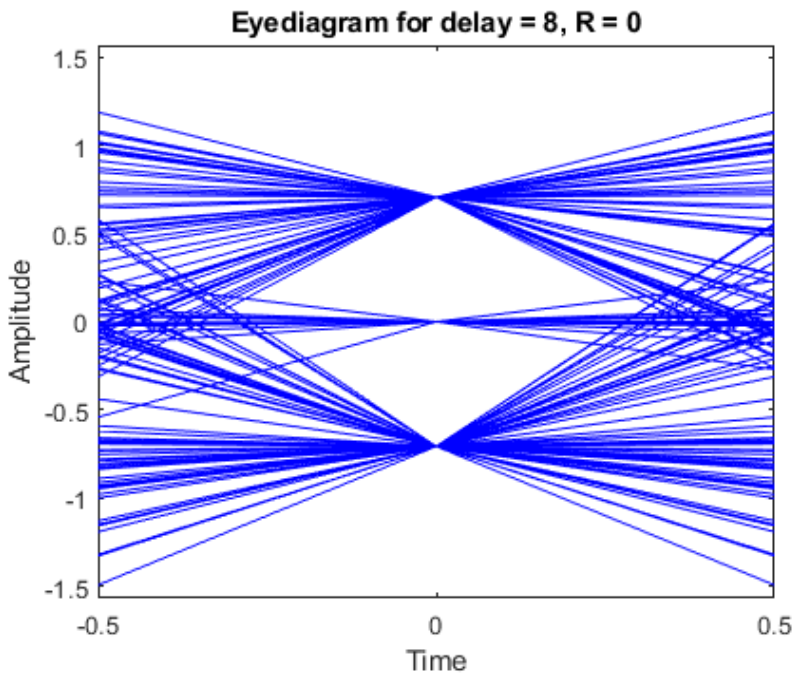
Generate the 10000-bit data

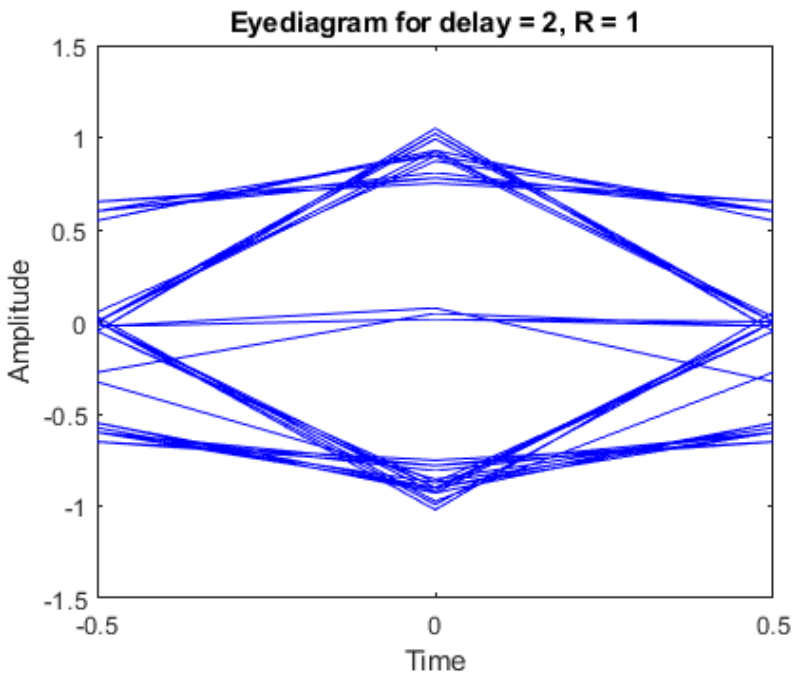
```
bits = randi([0 1], [1 100]);
```

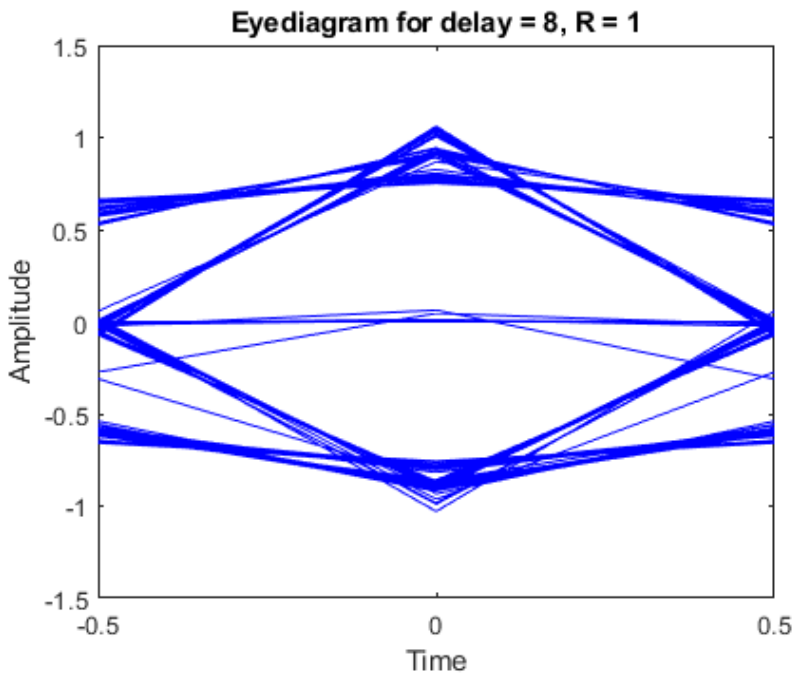
(3) ISI and raised cosine:

```
d = 2*bits -1;
R = [0 0 1 1];
Delay = [2 8 2 8];
Fd =0.5;
Fs =1;
out_Rx3 = zeros(4,length(d));
for i =1:4
    [NUM, DEN] = (rcosine(Fd, Fs,'sqrt', R(i), Delay(i)));
    Y = rcosflt(d, Fd, Fs, 'sqrt', R(i), Delay(i),NUM);
%     out_Rx3(i,:)=filter(NUM, DEN ,d);
    eyediagram(Y,2,1)
    title(['Eyediagram for delay = ',num2str(Delay(i)),', R = ', num2str(R(i))])
end
```









Published with MATLAB® R2020a