Matched filter

```
clc;
close all;
clear all;

fs = 1.0e4;
t = 0:1/fs:0.005;
signal = cos(2*pi*1000*t)';
shifted_signal = -1*delayseq(signal,5);
output = conv(signal,shifted_signal);
subplot(3,1,1)
plot(t.*1000,signal)
title('Input')
subplot(3,1,2)
plot(t.*1000,shifted_signal)
title('Shifted')
xlabel('msec')
t1 = 0:1/(fs*2):0.005;
subplot(3,1,3)
plot(t1.*1000,output)
title('Output')
xlabel('msec')
```

%>>>>>>>>> MATLAB code for binary ASK modulation and de-modulation >>>>>>>%

```
clc;
clear all;
close all;
x=[ 1 0 0 1 1 0 1];                          % Binary Information
bp=.000001;                                  % bit period
disp(' Binary information at Trans mitter :');
disp(x);
%XX representation of transmitting binary information as digital signal XXX
bit=[];
for n=1:1:length(x)
   if x(n)==1;
     se=ones(1,100);
   else x(n)==0;
      se=zeros(1,100);
   end
    bit=[bit se];
end
t1=bp/100:bp/100:100*length(x)*(bp/100);
subplot(3,1,1);
plot(t1,bit,'lineWidth',2.5);grid on;
axis([ 0 bp*length(x) -.5 1.5]);
ylabel('amplitude(volt)');
xlabel(' time(sec)');
title('transmitting information as digital signal');
%XXXXXXXXXXXXXXXXXXXXXXX Binary-ASK modulation
XXXXXXXXXXXXXXXXXXXXXXXXXXX%
A1=10;                  % Amplitude of carrier signal for information 1
A2=5;                   % Amplitude of carrier signal for information 0
br=1/bp;                                     % bit rate
f=br*10;                                     % carrier frequency
t2=bp/99:bp/99:bp;
ss=length(t2);
m=[];
for (i=1:1:length(x))
   if (x(i)==1)
      y=A1*cos(2*pi*f*t2);
   else
      y=A2*cos(2*pi*f*t2);
   end
   m=[m y];
end
t3=bp/99:bp/99:bp*length(x);
subplot(3,1,2);
```

```
plot(t3,m);
xlabel('time(sec)');
ylabel('amplitude(volt)');
title('waveform for binary ASK modulation coresponding binary information');
%XXXXXXXXXXXXXXXXXXXX Binary ASK demodulation
XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
mn=[];
for n=ss:ss:length(m)
  t=bp/99:bp/99:bp;
  y=cos(2*pi*f*t);                                        % carrier siignal
  mm=y.*m((n-(ss-1)):n);
  t4=bp/99:bp/99:bp;
  z=trapz(t4,mm)                                          % intregation
  zz=round((2*z/bp))
  if(zz>7.5)                                  % logic level = (A1+A2)/2=7.5
    a=1;
  else
    a=0;
  end
  mn=[mn a];
end
disp(' Binary information at Reciver :');
disp(mn);
%XXXXX Representation of binary information as digital signal which achived
%after ASK demodulation
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
bit=[];
for n=1:length(mn);
    if mn(n)==1;
       se=ones(1,100);
    else mn(n)==0;
        se=zeros(1,100);
    end
     bit=[bit se];
end
t4=bp/100:bp/100:100*length(mn)*(bp/100);
subplot(3,1,3)
plot(t4,bit,'LineWidth',2.5);grid on;
axis([ 0 bp*length(mn) -.5 1.5]);
ylabel('amplitude(volt)');
xlabel(' time(sec)');
title('recived information as digital signal after binary ASK demodulation');
%>>>>>>>>>>>>>>>>>>>>>>>>>>>>> end of program >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>%
```

```
%Purpose: Demonstrate how to match filter correctly
BW=3.84e6;
fs = 50*BW; %sample rate
T= 1/fs; %sample period
fc = 330e6; %center freq
chirpLen=0.075; %chirp length
A=3; %amplitude of voltage signal (normally unknown)
Ar=2; %amplitude of reference voltage signal (normally unknown)
%create the signal withought noise and zero padded on either side (zero
%padding not necessary because xcorr does that, I'm just demonstrating that
%signals don't need to be the same length.)
sig=[zeros(1,ceil(chirpLen*fs)),A*chirp(t,0,t(end),BW),zeros(1,ceil(chirpLen*fs))];
%create the reference chirp
ref_chirp=Ar*chirp(t,0,t(end),BW);
t=[0:T:(length(ref_chirp)-1)*T];
%normalize reference chirp: The reference chirp needs to have energy of 1
%so that it doesn't bias the output of the match filter. A filter shouldn't
%be applying gain to the signal or changing the units. The signal is in
%volts, so we divide by the square root of the energy to normalize it.
%If you know the signal's amplitude (for CW or FMCW):
energy=Ar^2/2*chirpLen;
%If you don't know the signal's amplitude, integrate to find energy (if it is noiseless):
   %energy=trapz(t,ref_chirp.^2)
ref_chirp=ref_chirp/sqrt(energy);
% perform match filtering
[R,lags] = xcorr(sig,ref_chirp); %signals don't need to be the same length
%R is the sum of each data sample as the signals are shifted past
%eachother, so to make the numerical integration correct, you need to
%multiply by dx which is T in this case. Then to get the filtered voltage
%signal in units of energy, you need to square it.
R=(abs(R*T)).^2; %absolute value only necessary if signals are complex
% take only positive side
R = R(lags>=0);
lags=lags(lags>=0);
[matchFiltPeak,index]=max(R);
figure()
plot(lags*T,R)
xlim([index-250 index+250]*T)
display(['Energy in signal was: ',num2str(A.^2/2*chirpLen)])
display(['which is the same as the peak of the match filter: ',num2str(matchFiltPeak)])
```

%>>>>>>>>> MATLAB code for binary PSK modulation and de-modulation >>>>>>>%

```
clc;
clear all;
close all;


x=[ 1 0 0 1 1 0 1];                          % Binary Information
bp=.000001;                                        % bit period
disp(' Binary information at Trans mitter :');
disp(x);

%XX representation of transmitting binary information as digital signal XXX
bit=[];
for n=1:1:length(x)
   if x(n)==1;
      se=ones(1,100);
   else x(n)==0;
       se=zeros(1,100);
   end
    bit=[bit se];

end
t1=bp/100:bp/100:100*length(x)*(bp/100);
subplot(3,1,1);
plot(t1,bit,'lineWidth',2.5);grid on;
axis([ 0 bp*length(x) -.5 1.5]);
ylabel('amplitude(volt)');
xlabel(' time(sec)');
title('transmitting information as digital signal');



%XXXXXXXXXXXXXXXXXXXXXXXX Binary-PSK modulation
XXXXXXXXXXXXXXXXXXXXXXXXXXXX%
A=5;                                  % Amplitude of carrier signal
br=1/bp;                                          % bit rate
f=br*2;                                      % carrier frequency
t2=bp/99:bp/99:bp;
ss=length(t2);
m=[];
for (i=1:1:length(x))
   if (x(i)==1)
      y=A*cos(2*pi*f*t2);
```

```
    else
        y=A*cos(2*pi*f*t2+pi);   %A*cos(2*pi*f*t+pi) means -A*cos(2*pi*f*t)
    end
    m=[m y];
end
t3=bp/99:bp/99:bp*length(x);
subplot(3,1,2);
plot(t3,m);
xlabel('time(sec)');
ylabel('amplitude(volt)');
title('waveform for binary PSK modulation coresponding binary information');




%XXXXXXXXXXXXXXXXXXXX Binary PSK demodulation
XXXXXXXXXXXXXXXXXXXXXXXXXXXX
mn=[];
for n=ss:ss:length(m)
  t=bp/99:bp/99:bp;
  y=cos(2*pi*f*t);                                        % carrier siignal
  mm=y.*m((n-(ss-1)):n);
  t4=bp/99:bp/99:bp;
  z=trapz(t4,mm)                                             % intregation
  zz=round((2*z/bp))
  if(zz>0)                                      % logic level = (A+A)/2=0
                        %becouse A*cos(2*pi*f*t+pi) means -A*cos(2*pi*f*t)
    a=1;
  else
    a=0;
  end
  mn=[mn a];
end
disp(' Binary information at Reciver :');
disp(mn);


%XXXXX Representation of binary information as digital signal which achived
%after PSK demodulation
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
bit=[];
for n=1:length(mn);
    if mn(n)==1;
       se=ones(1,100);
```

```
    else mn(n)==0;
        se=zeros(1,100);
    end
     bit=[bit se];

end
t4=bp/100:bp/100:100*length(mn)*(bp/100);
subplot(3,1,3)
plot(t4,bit,'LineWidth',2.5);grid on;
axis([ 0 bp*length(mn) -.5 1.5]);
ylabel('amplitude(volt)');
xlabel(' time(sec)');
title('recived information as digital signal after binary PSK demodulation');



%>>>>>>>>>>>>>>>>>>>>>>>>>> end of program >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>%
```

%>>>>>>>>> MATLAB code for binary FSK modulation and de-modulation >>>>>>>%

```
clc;
clear all;
close all;


x=[ 1 0 0 1 1 0 1];                          % Binary Information
bp=.000001;                                        % bit period
disp(' Binary information at Trans mitter :');
disp(x);

%XX representation of transmitting binary information as digital signal XXX
bit=[];
for n=1:1:length(x)
   if x(n)==1;
     se=ones(1,100);
   else x(n)==0;
      se=zeros(1,100);
   end
    bit=[bit se];

end
t1=bp/100:bp/100:100*length(x)*(bp/100);
subplot(3,1,1);
plot(t1,bit,'lineWidth',2.5);grid on;
axis([ 0 bp*length(x) -.5 1.5]);
ylabel('amplitude(volt)');
xlabel(' time(sec)');
title('transmitting information as digital signal');



%XXXXXXXXXXXXXXXXXXXXXXXX Binary-FSK modulation
XXXXXXXXXXXXXXXXXXXXXXXXXXXX%
A=5;                                    % Amplitude of carrier signal
br=1/bp;                                                % bit rate
f1=br*8;                    % carrier frequency for information as 1
f2=br*2;                    % carrier frequency for information as 0
t2=bp/99:bp/99:bp;
ss=length(t2);
m=[];
for (i=1:1:length(x))
   if (x(i)==1)
```

```
      y=A*cos(2*pi*f1*t2);
   else
      y=A*cos(2*pi*f2*t2);
   end
   m=[m y];
end
t3=bp/99:bp/99:bp*length(x);
subplot(3,1,2);
plot(t3,m);
xlabel('time(sec)');
ylabel('amplitude(volt)');
title('waveform for binary FSK modulation coresponding binary information');



%XXXXXXXXXXXXXXXXXXXX Binary FSK demodulation
XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
mn=[];
for n=ss:ss:length(m)
  t=bp/99:bp/99:bp;
  y1=cos(2*pi*f1*t);              % carrier siignal for information 1
  y2=cos(2*pi*f2*t);              % carrier siignal for information 0
  mm=y1.*m((n-(ss-1)):n);
  mmm=y2.*m((n-(ss-1)):n);
  t4=bp/99:bp/99:bp;
  z1=trapz(t4,mm)                                % intregation
  z2=trapz(t4,mmm)                                % intregation
  zz1=round(2*z1/bp)
  zz2= round(2*z2/bp)
  if(zz1>A/2)      % logic lavel= (0+A)/2 or (A+0)/2 or 2.5 ( in this case)
    a=1;
  else(zz2>A/2)
    a=0;
  end
  mn=[mn a];
end
disp(' Binary information at Reciver :');
disp(mn);


%XXXXX Representation of binary information as digital signal which achived
%after demodulation
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

```
bit=[];
for n=1:length(mn);
   if mn(n)==1;
      se=ones(1,100);
   else mn(n)==0;
       se=zeros(1,100);
   end
    bit=[bit se];

end
t4=bp/100:bp/100:100*length(mn)*(bp/100);
subplot(3,1,3)
plot(t4,bit,'LineWidth',2.5);grid on;
axis([ 0 bp*length(mn) -.5 1.5]);
ylabel('amplitude(volt)');
xlabel(' time(sec)');
title('recived information as digital signal after binary FSK demodulation');



%>>>>>>>>>>>>>>>>>>>>>>>>>>>> end of program >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>%
```

Ber ask psk fsk

```
clc;
clear all;
close all;
EbNodB = 0:1:20;
EbNo  = 10.^(EbNodB/10);

pe_bpsk = (1/2)*erfc(sqrt(EbNo));
pe_bfsk = (1/2)*erfc(sqrt(EbNo/2));
pe_bask = (1/2)*erfc(sqrt(EbNo/4));

pe_dpsk = (1/2)*exp(-EbNo);
pe_ncfsk = (1/2)*exp(-EbNo/2);
pe_ncask = (1/2)*exp(-EbNo/4);

semilogy(EbNodB,pe_bpsk,'-r o','LineWidth',2);
hold on;
semilogy(EbNodB,pe_bfsk,'-b o','LineWidth',2);
hold on;
semilogy(EbNodB,pe_bask,'-g o','LineWidth',2);
hold on;
semilogy(EbNodB,pe_dpsk,'-k s','LineWidth',2);
hold on;
semilogy(EbNodB,pe_ncfsk,'-c s','LineWidth',2);
legend();
hold on;
semilogy(EbNodB,pe_ncask,'-m s','LineWidth',2);
legend ('Coherent BPSK','Coherent BFSK','Coherent BASK','DPSK', 'Non-Coherent FSK',
'Non-Coherent ASK');
xlabel ('Eb/No(dB)');
ylabel('BER');
axis tight;
axis([0 20 10e-9 1]);
grid on;
```

Snr vs ber

```
clc;
close all;
SNR=10;
snr = 10*log10(SNR);

n = 100;
m = randi([0 1], 1,n);
bit=[];
for i=1:1:length(m)
   if m(i)==1
      se=ones(1,100);
   else
      se=zeros(1,100);
   end
   bit=[bit se];
end
bp=0.001;
t=bp/100:bp/100:100*length(m)*(bp/100);
subplot(3,1,1);
plot(t, bit, 'linewidth',.5);
grid on;
axis([0 bp*length(m) -.5 1.5]);
sigma=sqrt(1/(2*snr));

y=bit+sigma.*randn(1,length(bit));
subplot(3,1,2);
plot(t,y, 'linewidth',.5);
grid on;
axis([0 bp*length(m) -.5 1.5]);
z=y>0.5;
subplot(3,1,3);
plot(t,z,'linewidth',.5);
grid on;
axis([0 bp*length(m) -.5 1.5]);
```