SAYISAL HABERLEŞME MATLAB UYGULAMALARI

1-KONVOLÜSYON UYGULAMALARI

Matlab ile hazır olarak kullanılan conv,conv2,convn hazır fonksiyonları bulunmakla birlikte konvolüsyon sonucunun '0' sıfır indisli değerinin de bulunması için aşağıdaki fonksiyon yazılmış ve bu fonksiyon kullanılarak bir örnek uygulama gerçekleştirilmiştir.

```
function [y,ny] = conv_m(x,nx,h,nh)
nyb=nx(1)+nh(1);
nye = nx(length(x)) + nh(length(h));
ny=[nyb:nye] ;
y = conv(x,h);
end
```

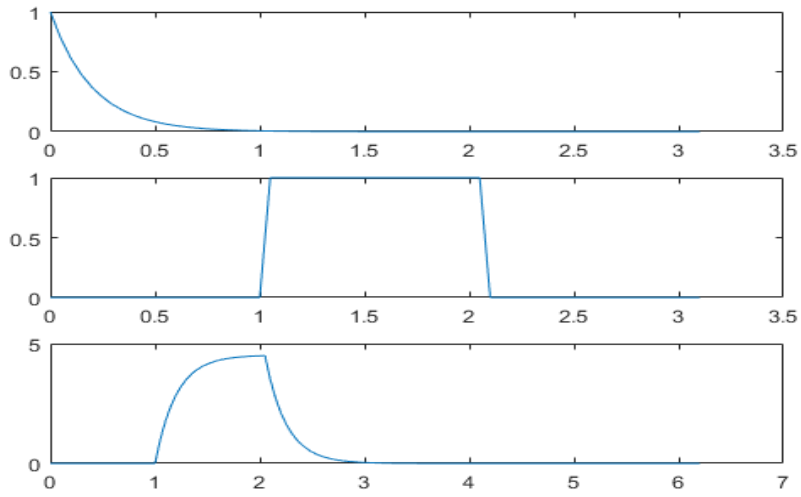
Örnek-1

```
clear all
close all
clc

x=[3. 11, 7, 0, -1. 4, 2];
h=[2, 3, 0, -5, 2, 1];
nx =[-3:3];
nh=[-1:4];
[y,ny]=conv_m(x,nx,h,nh)
```

```
SONUÇ:
  y =6    31    47     6   -51    -5    41    18   -22    -3     8     2


ny =-4    -3    -2    -1     0     1     2     3     4     5     6     7
```

Örnek-2

```
clear all
close all
clc

t=0:0.05:3.1;
x=exp(-5*t);
subplot(3,1,1)
plot(t,x);
h=[zeros(1,length(t)/3) ones(1,length(t)/3) zeros(1,length(t)/3)] ;
subplot(3,1,2)
plot(t,h)
subplot(3,1,3)
y=(conv(h,x));
t2=0:0.05:6.2;
plot(t2,y)
```



Toplanabilir Gürültü

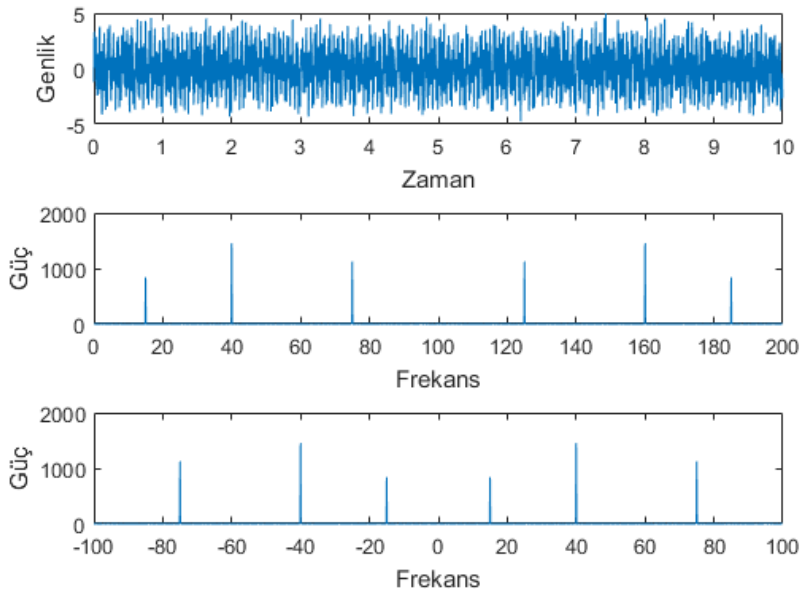
Burada snr (Sinyal/Gürültü Oranı) değerini -10 ile 10 arasında değiştirerek sinyalde oluşan bozulmayı gözlemleyiniz.

```
clear all
close all
clc
t = 0:0.001:5;
x = (1.3)*sin(2*pi*1*t) + (1.7)*sin(2*pi*4*t);
plot(t,x)
snr=0;
figure
y=awgn(x,snr);
plot(t,y)
xlabel('Zaman')
ylabel('Genlik')
```

2-FOURİER DÖNÜŞÜMÜ UYGULAMALARI

```
clear all
close all
clc
fs = 200;
t = 0:1/fs:10-1/fs;
x = (1.3)*sin(2*pi*15*t) + (1.7)*sin(2*pi*40*t) + 1.5*sin(2*pi*75*t);
snr=0;
x= awgn(x,snr);
subplot(3,1,1)
plot(t,x)
xlabel('Zaman')
ylabel('Genlik')
```

```
y = fft(x);
n = length(x);
f = (0:n-1)*(fs/n);
power = abs(y).^2/n;
subplot(3,1,2)
plot(f,power)

xlabel('Frekans')
ylabel('Güç')

y0 = fftshift(y);
f0 = (-n/2:n/2-1)*(fs/n);
power0 = abs(y0).^2/n;
subplot(3,1,3)
plot(f0,power0)
xlabel('Frekans')
ylabel('Güç')
```




```
clear all
close all
clc
t=-5:0.05:5;
x=[zeros(1,length(t)/3) ones(1,length(t)/3) zeros(1,length(t)/3)];
plot(t,x)
y=fft(x);
figure
plot(abs(fftshift(y)))
```

3-ÖRNEKLEME VE KUANTALAMA:

```
t = [0:.1:2*pi]; % Times at which to sample the sine function
sig = sin(t); % Original signal, a sine wave
partition = [-1:.2:1]; % Length 11, to represent 12 intervals
codebook = [-1.2:.2:1]; % Length 12, one entry for each interval
[index,quants] = quantiz(sig,partition,codebook); % Quantize.
plot(t,sig,'x',t,quants,'.')
legend('Original signal','Quantized signal');
axis([-.2 7 -1.2 1.2])
```

***EKLEME YAPILACAK

****

4-KODLAMA TEKNİKLERİ

```
% MANCHESTER
function MANCHESTER(h)
h=[1 0 0 1 1 0 1 0 1 0];
clf;
n=1;
h=~h;
l=length(h);
h(l+1)=1;
while n<=length(h)-1;
   t=n-1:0.001:n;
if h(n) == 0
   if h(n+1)==0
      y=-(t<n)+2*(t<n-0.5)+1*(t==n);
   else
      y=-(t<n)+2*(t<n-0.5)-1*(t==n);
   end
   d=plot(t,y);grid on;
   title('Line code MANCHESTER');
   set(d,'LineWidth',2.5);
   hold on;
   axis([0 length(h)-1 -1.5 1.5]);
   disp('one');
else
   if h(n+1)==0
      y=(t<n)-2*(t<n-0.5)+1*(t==n);
   else
      y=(t<n)-2*(t<n-0.5)-1*(t==n);
   end
   %y=(t>n-1)+(t==n-1);
   d=plot(t,y);grid on;
```

```
    title('Line code MANCHESTER');
    set(d,'LineWidth',2.5);
    hold on;
    axis([0 length(h)-1 -1.5 1.5]);
    disp('zero');
end
n=n+1;
%pause;
end
```

%AMINRZ

```
function AMINRZ(h)
h=[1 0 0 1 1 0 1 0 1 0];
clf;
n=1;
l=length(h);
h(l+1)=1;
ami=-1;
while n<=length(h)-1;
    t=n-1:0.001:n;
if h(n) == 0
    if h(n+1)==0
        y=(t>n);
    else
        if ami==1
            y=-(t==n);
        else
            y=(t==n);
        end
    end
    d=plot(t,y);grid on;
    title('Line code AMI NRZ');
    set(d,'LineWidth',2.5);
    hold on;
    axis([0 length(h)-1 -1.5 1.5]);
    disp('zero');
else
    ami=ami*-1;
    if h(n+1)==0
        if ami==1
            y=(t<n);
        else
            y=-(t<n);
        end
    else
        if ami==1
            y=(t<n)-(t==n);
        else
            y=-(t<n)+(t==n);
        end
```

```
    end
    %y=(t>n-1)+(t==n-1);
    d=plot(t,y);grid on;
    title('Line code AMI NRZ');
    set(d,'LineWidth',2.5);
    hold on;
    axis([0 length(h)-1 -1.5 1.5]);
    disp('one');
end
n=n+1;
pause;
end
```

5-KODLAMA TEKNİKLERİ

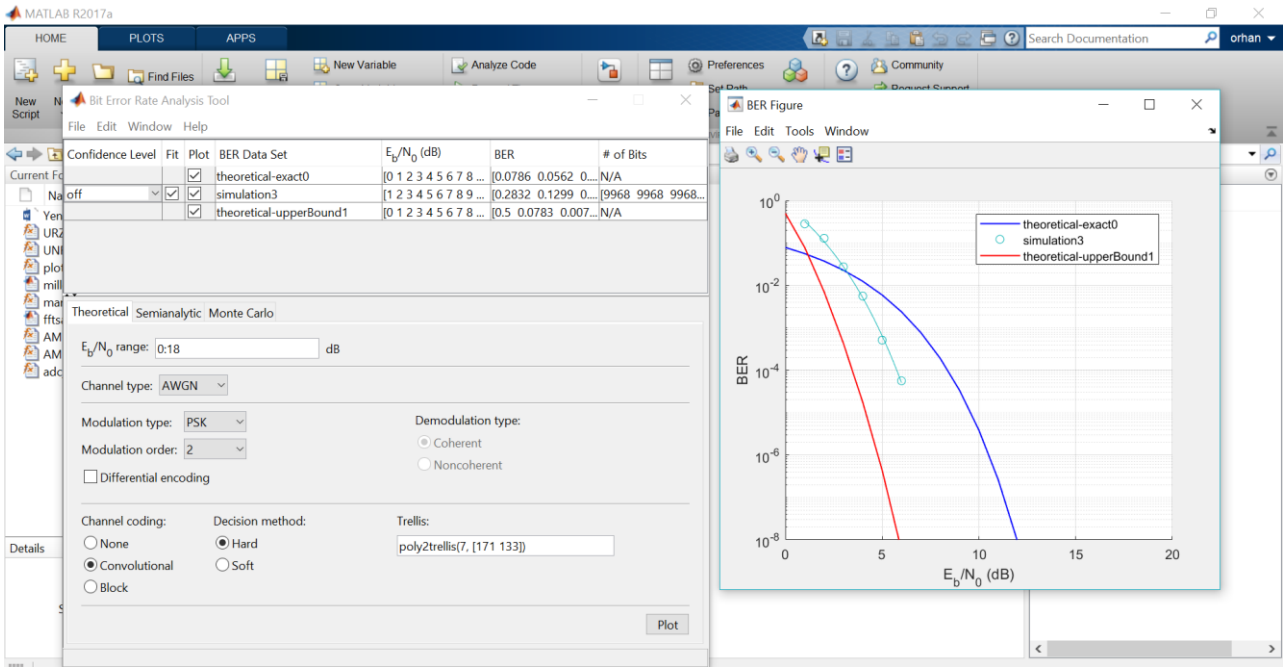
```
% AMIRZ
function AMIRZ(h)
h=[1 0 0 1 1 0 1 0 1 0];
clf;
n=1;
l=length(h);
h(l+1)=1;
ami=-1;
while n<=length(h)-1;
    t=n-1:0.001:n;
if h(n) == 0
    if h(n+1)==0
        y=(t>n);
    else
        if ami==1
            y=-(t==n);
        else
            y=(t==n);
        end
    end
    d=plot(t,y);grid on;
     title('Line code AMI RZ');
    set(d,'LineWidth',2.5);
    hold on;
    axis([0 length(h)-1 -1.5 1.5]);
    disp('zero');
else
    ami=ami*-1;
    if h(n+1)==0
        if ami==1
            y=(t<n-0.5);
        else
            y=-(t<n-0.5);
        end
    else
        if ami==1
```

```
            y=(t<n-0.5)-(t==n);
        else
            y=-(t<n-0.5)+(t==n);
        end

    end
    %y=(t>n-1)+(t==n-1);
    d=plot(t,y);grid on;
    title('Line code AMI RZ');
    set(d,'LineWidth',2.5);
    hold on;
    axis([0 length(h)-1 -1.5 1.5]);
    disp('one');
end
n=n+1;
%pause;
End
```

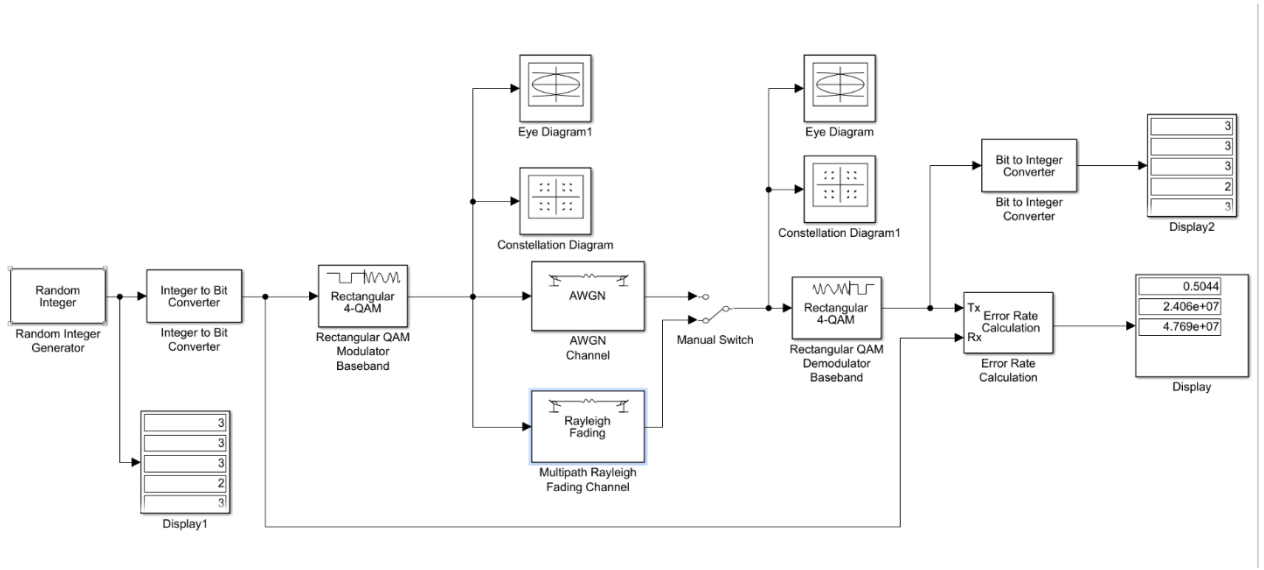
```
% MİLLER
h=[1 1 0 0 1 0 1 1 0 0 0 1 1 1 1 0 1];
est_initial=-1;
clc;
close all
con=est_initial;%Set 1  -1
long=length(h);%Number of bits of the signal
n=1;%Initial state for "while" loop
ac=[];%Null matrix to code signal.
bits=[];%Null matrix to original signal.
h(long+1)=0;%Valor de extensin de la seal
while n<=long%Code to finished the length of the signal.
    if h(n)==1 %If the bit is 1
        bit=[ones(1,100)];
        s=[con*ones(1,50) -con*ones(1,50)];
        con=con*-1;%Switch state of the signal
    else %If the bit is 0
        bit=[zeros(1,100)];
        s=[con*ones(1,100)];
        if h(n+1)==0%If the next bit is 0
            con=con*-1;%Switch state of the signal
        end
    end
    ac=[ac s];%Accumulate miller code.
    bits=[bits bit];%Accumulate signal
    n=n+1;%Increment of the cycle
    s=[];%Reset temporal matrix s.
end



subplot(2,1,1);plot(bits,'LineWidth',2);
```

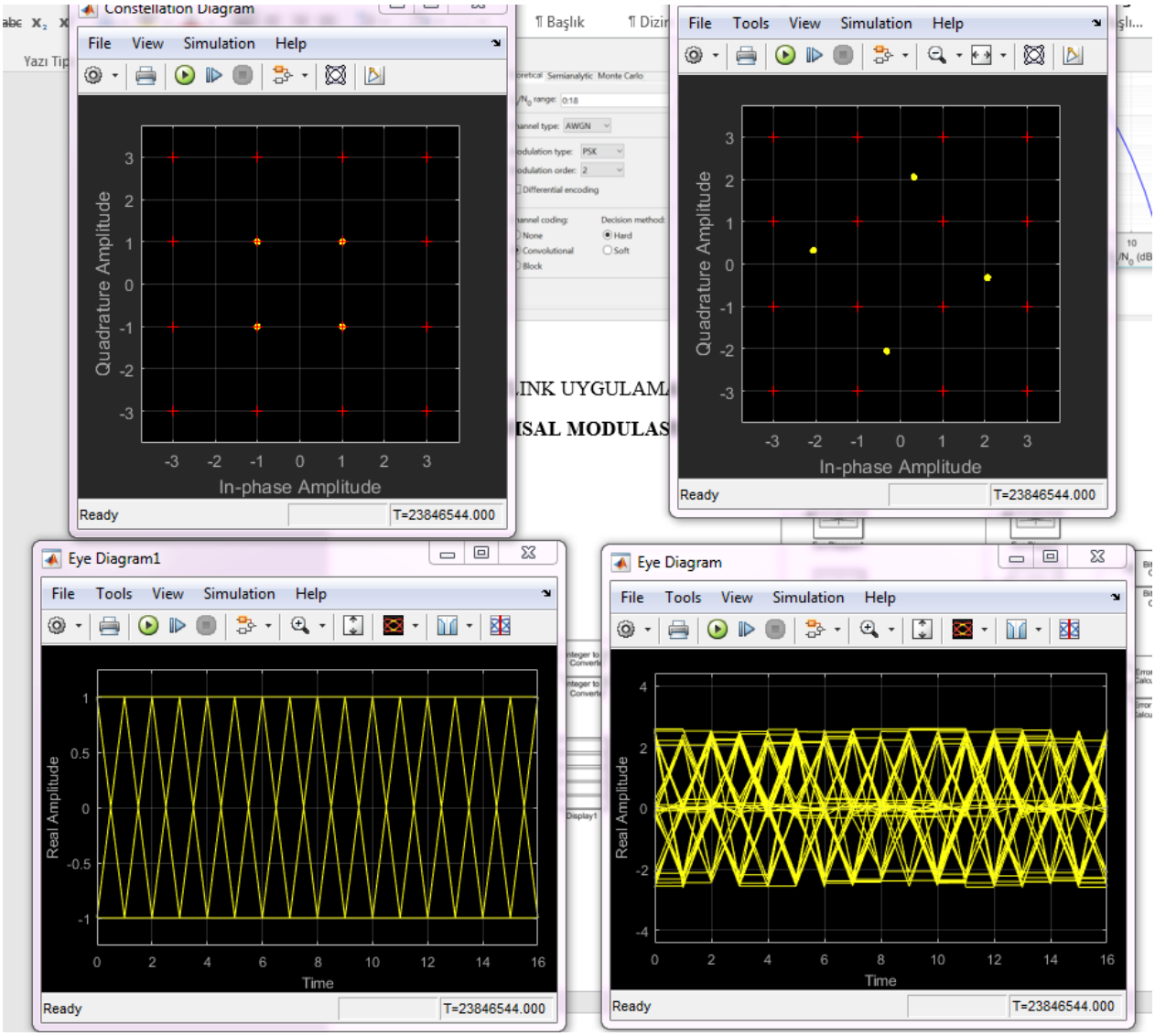
```
title('INPUT SIGNAL');
set(gca,'xtick',0:100:100*long)%
axis([0 100*(length(h)-1) -2 2])%
grid on %
subplot(2,1,2);plot(ac,'LineWidth',2)%
title('MILLER CODE')
set(gca,'xtick',0:100:100*long)%
axis([0 100*(length(h)-1) -2 2])%
grid on %
```

6-BİT HATA ORANI (BER) ANALİZİ BERTOOL TOOLBOX KULLANIMI



7-SAYISAL MODULASYON UYGULAMALARI (PSK, QAM, 16QAM)

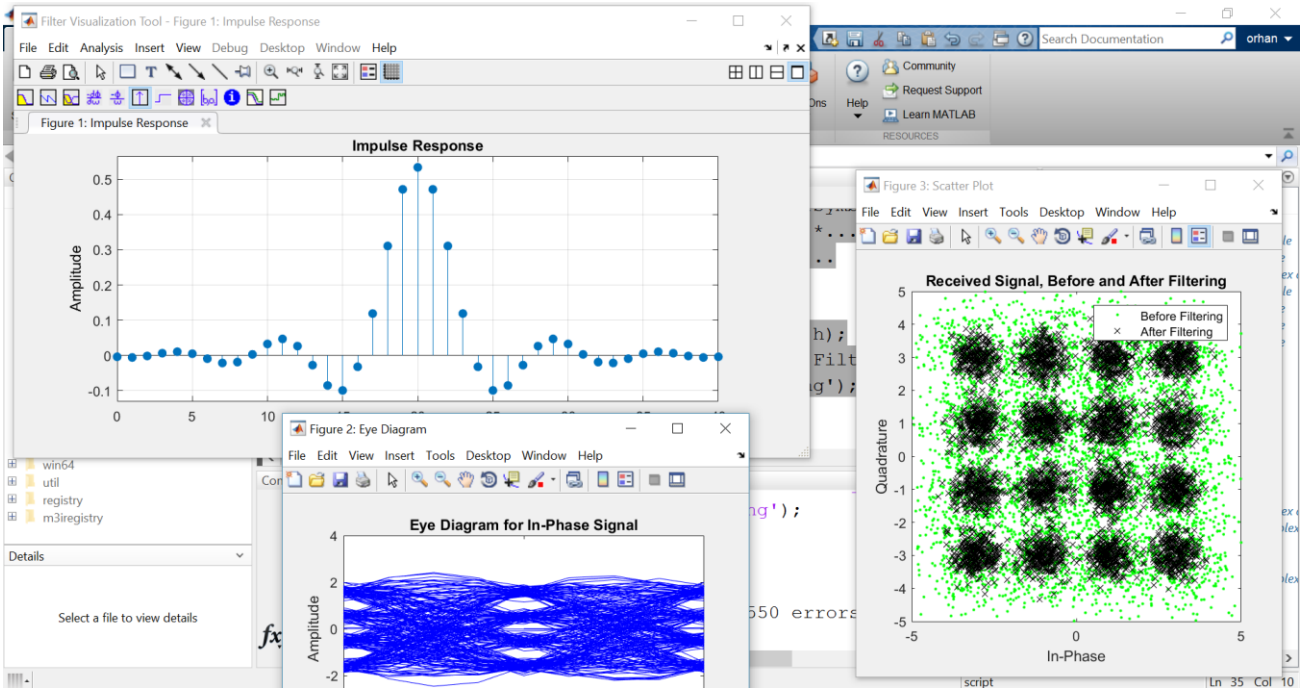




8- DARBE ŞEKİLLENDİRME VE BER ÜZERİNDEKİ ETKİSİ

```
M = 16;                          % Size of signal constellation
k = log2(M);                     % Number of bits per symbol
numBits = 3e5;                   % Number of bits to process
numSamplesPerSymbol = 4;         % Oversampling factor
span = 10;          % Filter span in symbols
rolloff = 0.25;     % Roloff factor of filter
rrcFilter = rcosdesign(rolloff, span, numSamplesPerSymbol);
fvtool(rrcFilter,'Analysis','Impulse')
rng default                              % Use default random number
generator
dataIn = randi([0 1], numBits, 1);  % Generate vector of binary data
dataInMatrix = reshape(dataIn, length(dataIn)/k, k); % Reshape data
into binary 4-tuples
```

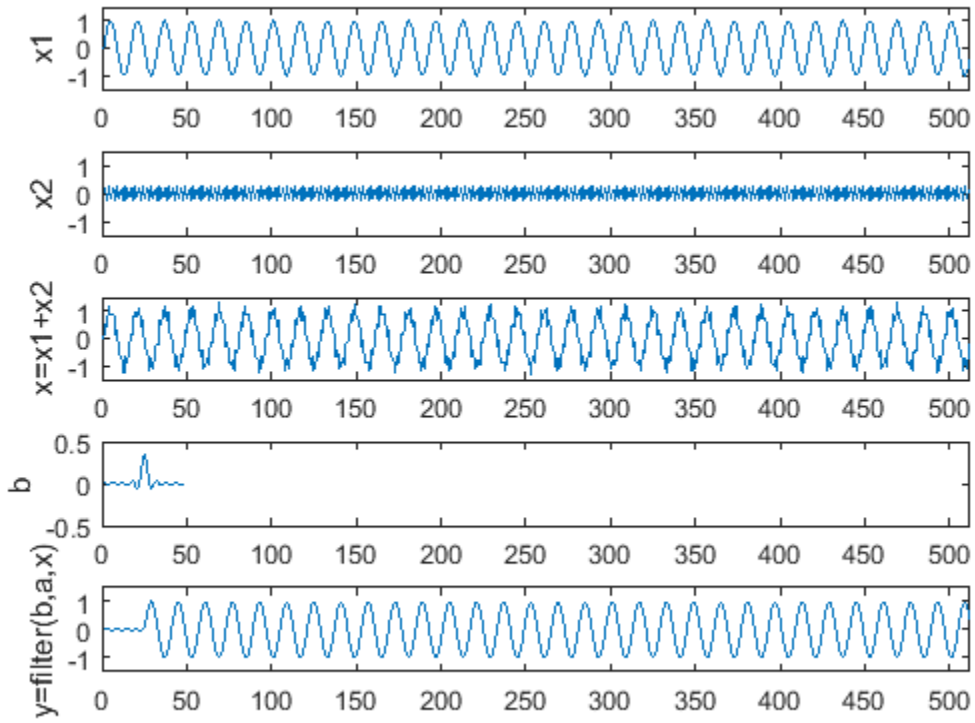
```
dataSymbolsIn = bi2de(dataInMatrix);                    % Convert to
integers
dataMod = qammod(dataSymbolsIn, M);
txSignal = upfirdn(dataMod, rrcFilter, numSamplesPerSymbol, 1);
EbNo = 10;
snr = EbNo + 10*log10(k) - 10*log10(numSamplesPerSymbol);
rxSignal = awgn(txSignal, snr, 'measured');
rxFiltSignal = upfirdn(rxSignal,rrcFilter,1,numSamplesPerSymbol);    %
Downsample and filter
rxFiltSignal = rxFiltSignal(span+1:end-span);                        %
Account for delay
dataSymbolsOut = qamdemod(rxFiltSignal, M);
dataOutMatrix = de2bi(dataSymbolsOut,k);
dataOut = dataOutMatrix(:);                  % Return data in column
vector
[numErrors, ber] = biterr(dataIn, dataOut);
fprintf('\nThe bit error rate = %5.2e, based on %d errors\n', ...
    ber, numErrors)
eyediagram(txSignal(1:2000),numSamplesPerSymbol*2);
h = scatterplot(sqrt(numSamplesPerSymbol)*...
    rxSignal(1:numSamplesPerSymbol*5e3),...
    numSamplesPerSymbol,0,'g.');
hold on;
scatterplot(rxFiltSignal(1:5e3),1,0,'kx',h);
title('Received Signal, Before and After Filtering');
legend('Before Filtering','After Filtering');
axis([-5 5 -5 5]); % Set axis ranges
hold off;
```



9. Filtreleme Uygulamaları

%% gaussian filtre kullanarak gürültü filtreleme

```
Fs=8e3;
Ts=1/Fs;
Ns=512;
t=[0:Ts:Ts*(Ns-1)];
f1=500;
f2=3200;
x1=sin(2*pi*f1*t);
x2=0.3*sin(2*pi*f2*t);
x=x1+x2;
a = 1;
b = fir1(48,0.35,'low');
y=filter(b,a,x);
subplot(5,1,1); plot(x1);ylabel('x1');axis([0,length(x),-1.5,1.5]);
subplot(5,1,2); plot(x2);ylabel('x2');axis([0,length(x),-1.5,1.5]);
subplot(5,1,3); plot(x);ylabel('x=x1+x2');axis([0,length(x),-1.5,1.5]);
subplot(5,1,4); plot(b);ylabel('b');axis([0,length(x),-0.5,0.5]);
subplot(5,1,5); plot(y);ylabel('y=filter(b,a,x)');axis([0,length(x),-1.5,1.5]);
```



10. uygulama ödev olarak verilecektir.