浙江工业大学

MATLAB & Communication Simulations

Lab Report5

Name 凌智城

Teacher 张昱

Class 通信工程 1803 班

Student ID 201806061211

Department 信息工程学院

Date: May 14th,2021

I. Task_1

A. Code

lab5_1.m

```
%% Consider the following 8-PAM waveforms, where T=1.

clear;
clc;

m = [0,1,2,3,4,5,6,7];     % Equally spaced.
Am = -3.5+m;               % The amplitude of the m-th waveform.

%% Task(1)
% Give the symbol interval and bit interval.
T = 1                      % Symbol interval.
Tb = T/3                   % Bit interval.

plot(Am,0,'*');grid on;title('PAM\_8 Constellation point');
for i=1:length(Am)
    text(Am(i),0,num2str(Am(i)))
end

%% Task(2)
% Give the average energy per symbol and average energy per bit.
Eav = sum(Am.^2)/8         % Average energy per symbol.
E = Eav/3                  % Average energy per bit.

%% Task(3)
% Simulate the symbol error rate and bit error rate of 8-PAM when
the SNR
% per bit is 0dB.

SNRindB = 0;
% simulated error rate
[smld_err_p,smld_err_pb]=smldpe_PAM_8(SNRindB)
```

smldpe_PAM_8.m

```
function [p,pb]=smldpe_PAM_8(snr_in_dB)
% [p]=smldpe_PAM_8(snr_in_dB)
%         SMLDPE_PAM_8 simulates the probability of error for the
```

```
given
%         snr_in_dB, signal to noise ratio in dB.

d=1;
SNR=exp(snr_in_dB*log(10)/10);     % signal to noise ratio per
bit
sgma=sqrt((5.25*d^2)/(6*SNR));     % sigma, standard deviation of
nise
N=10000;                           % number of symbols being
simulated

% generation of the quarternary data source follows
for i=1:N
  temp=rand;                % a uniform random variable over (0,1)
  if (temp<0.125)
    dsource(i)=0;           % with probability 1/8, source output
is "000"
  elseif (temp<0.25)
    dsource(i)=1;           % with probability 1/8, source output
is "001"
  elseif (temp<0.375)
    dsource(i)=2;           % with probability 1/8, source output
is "010"
  elseif (temp<0.5)
    dsource(i)=3;           % with probability 1/8, source output
is "011"
  elseif (temp<0.625)
    dsource(i)=4;           % with probability 1/8, source output
is "100"
  elseif (temp<0.75)
    dsource(i)=5;           % with probability 1/8, source output
is "101"
  elseif (temp<0.875)
    dsource(i)=6;           % with probability 1/8, source output
is "110"
  else
    dsource(i)=7;           % with probability 1/8, source output
is "111"
  end
end

% detection, and probability of error calculation
numoferr=0;
numoferr_p=0;
```

```
for i=1:N
  % The matched filter outputs
  if (dsource(i)==0)
    r=-3.5*d+gngauss(sgma);     % if the source output is "000"
  elseif (dsource(i)==1)
    r=-2.5*d+gngauss(sgma);     % if the source output is "001"
  elseif (dsource(i)==2)
    r=-1.5*d+gngauss(sgma);     % if the source output is "010"
  elseif (dsource(i)==3)
    r=-0.5*d+gngauss(sgma);     % if the source output is "011"
  elseif (dsource(i)==4)
    r=0.5*d+gngauss(sgma);      % if the source output is "100"
  elseif (dsource(i)==5)
    r=1.5*d+gngauss(sgma);      % if the source output is "101"
  elseif (dsource(i)==6)
    r=2.5*d+gngauss(sgma);      % if the source output is "110"
  else
    r=3.5*d+gngauss(sgma);      % if the source output is "111"
  end
  % detector follows
  if (r<-3*d)
    decis=0;                    % decision is "000"
  elseif (r<-2*d)
    decis=1;                    % decision is "001"
  elseif (r<-1*d)
    decis=2;                    % decision is "010"
  elseif (r<0)
    decis=3;                    % decision is "011"
  elseif (r<1*d)
    decis=4;                    % decision is "100"
  elseif (r<2*d)
    decis=5;                    % decision is "101"
  elseif (r<3*d)
    decis=6;                    % decision is "110"
  else
    decis=7;                    % decision is "111"
  end
  if (decis~=dsource(i))        % if it is an error,
    numoferr=numoferr+1;        % increase the error counter
  end
  % Convert the judged decimal number decis into a binary number
  % with a bit width of 3 and store it in the bit_data sequence.
  bit_data((i-1)*3+1:i*3)=dec2bin(decis,3);
```

```
end

% Convert the randomly generated decimal number dsource
% into a binary number with a bit width of 3
% and store it in the bit_map sequence.
for i=1:N
    bit_map((i-1)*3+1:i*3) = dec2bin(dsource(i),3);
end

% Compare bit_map and bit_data, calculate how many bits of data
are wrong.
for i=1:3*N
  if (bit_map(i)~=bit_data(i))
      numoferr_p=numoferr_p+1;
  end
end

p=numoferr/N;                    % probability of error estimate
pb=numoferr_p/N/3;               % probability of bit error estimate
```

B. Figure

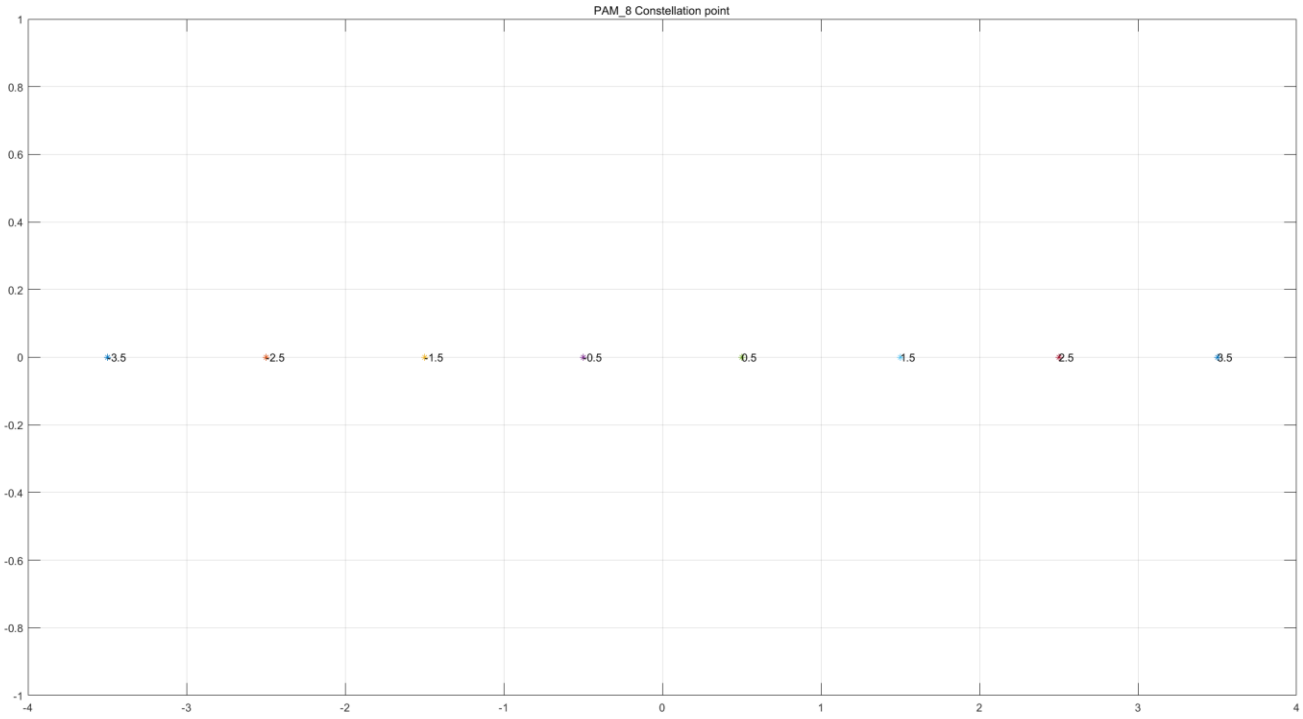

Fig.1 PAM_8 Constellation point.

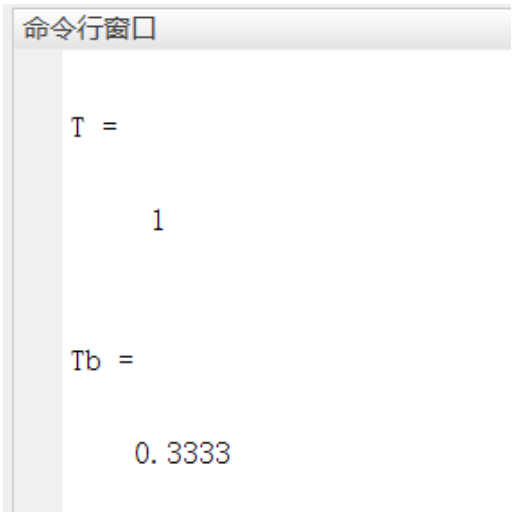

Fig.2 The symbol interval and bit interval.

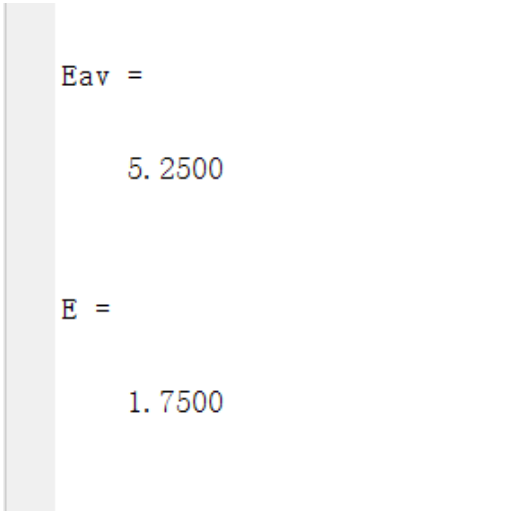

Fig.3 The average energy per symbol and average energy per bit.

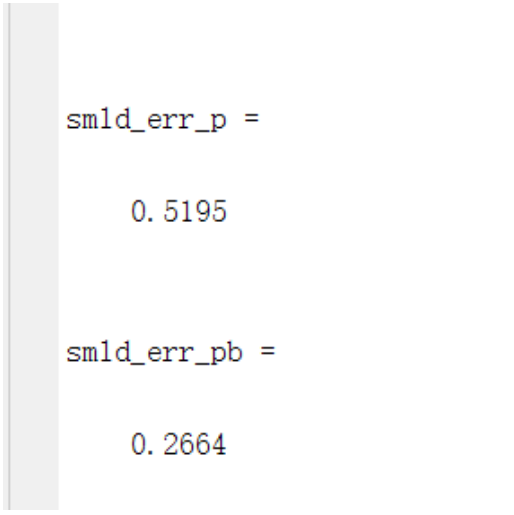

Fig.4 The symbol error rate and bit error rate.

C. Discussion

Each of the 8 PAM signal waveforms can be used to transmit 3 bits of information, so the following information can be assigned to 8 corresponding signal waveforms, 000→0, 001→1, 010→2, 011→3, 100→4,101→5,110→6,111→7.{000,001,010,011,100,101,110,111} called a symbol, the time duration T is called the symbol interval, if the bit rate R=1/Tb, then the bit interval Tb=T/3, see in fig.1 .

The partial signal waveforms are the result of the amplitude weighting of the signal basis function, so these signal waveforms can be geometrically represented as some points on the solid line. Seeing in fig.2, the geometric representation of these 8 PAM signals is the signal constellation diagram.

Average energy per symbol:

$$\mathrm{E}_{\mathrm{av}} = \frac{1}{8}\sum_{\mathrm{k}=1}^{8}\int_{0}^{\mathrm{T}} s_k^2(t)dt = 2[(3.5)^2 + (2.5)^2 + (1.5)^2 + (0.5)^2]/8 = 5.25$$

Average energy per bit:

$$\mathrm{E} = \frac{\mathrm{E}_{av}}{3} = 1.75$$

Symbol error rate :

$$smld\_err\_p = \frac{numoferr}{N}$$

Bit error rate:

$$smld\_err\_pb = \frac{numoferr\_pb}{N}$$

Through simulation and the decision of the signal judge, compare whether the information of each symbol and bit is wrong, and calculate the symbol error rate and bit error rate.

II. Task_2

A. Code

lab5_2.m

```
%% Simulate the band-limited channel.

clear;
clc;

Ts = 1;                 % Symbol interval.
Tb = Ts/3;              % Bit interval.
fs = 1000;              % Sampling rate.
dt = 1/fs;              % Sampling interval.
N = 10;                 % Number of symbols.
t = -N/2 : dt : N/2;    % Sequence transmission time.
m = [0,1,2,3,4,5,6,7]; % Equally spaced.
Am = -3.5+m;            % The amplitude of the m-th waveform.

%% Task(1) Randomly generate a sequence of ten 8-PAM signals.
for i=1:length(t)
    temp=randi(7);
    Sm(i)=Am(temp);
end
[f,Sf] = T2F(t,Sm);     % TF

figure(1)
plot(t,Sm);
xlabel('t');ylabel('Amplitude ');
title('8-PAM waveform');

figure(2)
plot(f,abs(Sf));
xlabel('f/Ts' );ylabel('');
title('8-PAM spectrum');

%% Task(2) Consider a band-limited noiseless channel.
% Generate a band-limited noiseless channel H(f)=1, |f|¡Ü3Hz.
for k = 1 : length(f)
    if abs(f(k))>3
        Hf1(k)=0;
    else
```

```
        Hf1(k)=Ts;
    end
end
for k = 1 : length(f)
    if abs(f(k))>30
        Hf2(k)=0;
    else
        Hf2(k)=Ts;
    end
end

% Plot the band-limited noiseless channel.
figure(3)
plot(f,Hf1,f,Hf2);
grid on;
xlabel ('f/Ts'); ylabel('Noiseless channel spectrum');
axis([ - 40 40 0  1.2]);
legend('cut-off f=3','cut-off f=30');


% Passing through this channel.
Xf1 = Sf.*Hf1;
[t,Xt1] = F2T(f,Xf1);     % FT
Xf2 = Sf.*Hf2;
[t,Xt2] = F2T(f,Xf2);     % FT

figure(4)
subplot(121)
plot(f,abs(Xf1));
xlabel('f/Ts');ylabel('');
axis([ -5 5 0 6]);
title('8-PAM waveform passing noiseless channel spectrum');
legend('cut-off f=3');
subplot(122)
plot(f,abs(Xf2));
xlabel('f/Ts');ylabel('');
axis([ -50 50 0 6]);
title('8-PAM waveform passing noiseless channel spectrum');
legend('cut-off f=30');

figure(5)
subplot(121)
plot(t,abs(Xt1));
axis([ 0 N 0 1]);
```

```
xlabel('t');ylabel('Amplitude');
title('8-PAM waveform passomg noiseless channel waveform');
legend('cut-off f=3');
subplot(122)
plot(t,abs(Xt2));
axis([ 0 N 0 1]);
xlabel('t');ylabel('Amplitude');
title('8-PAM waveform passomg noiseless channel waveform');
legend('cut-off f=30');
```

B. Figure

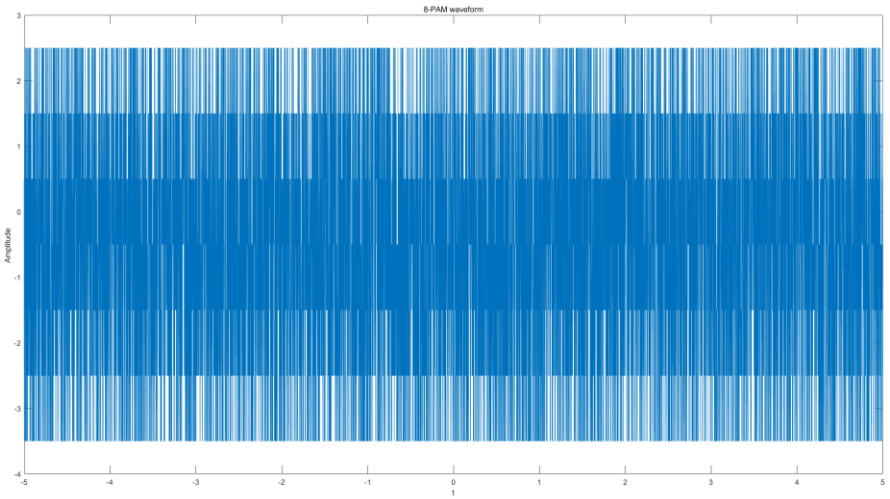

fig.5 8-PAM waveform.

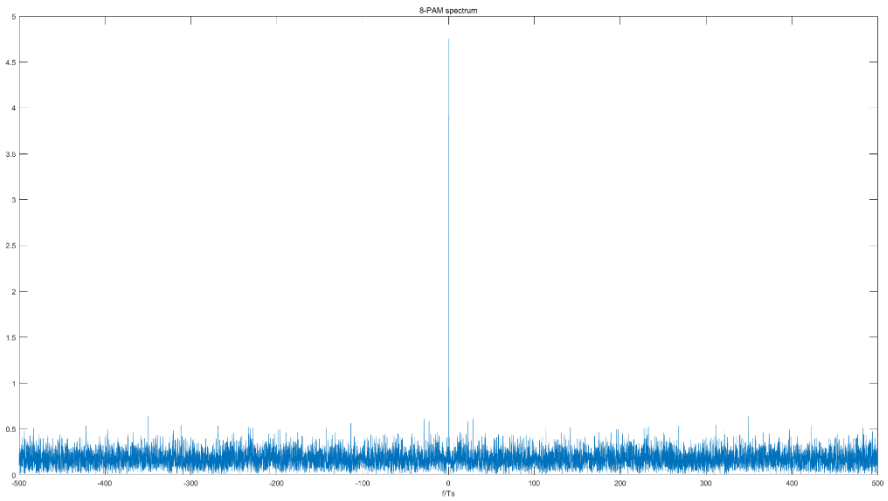

fig.6 8-PAM spectrum.

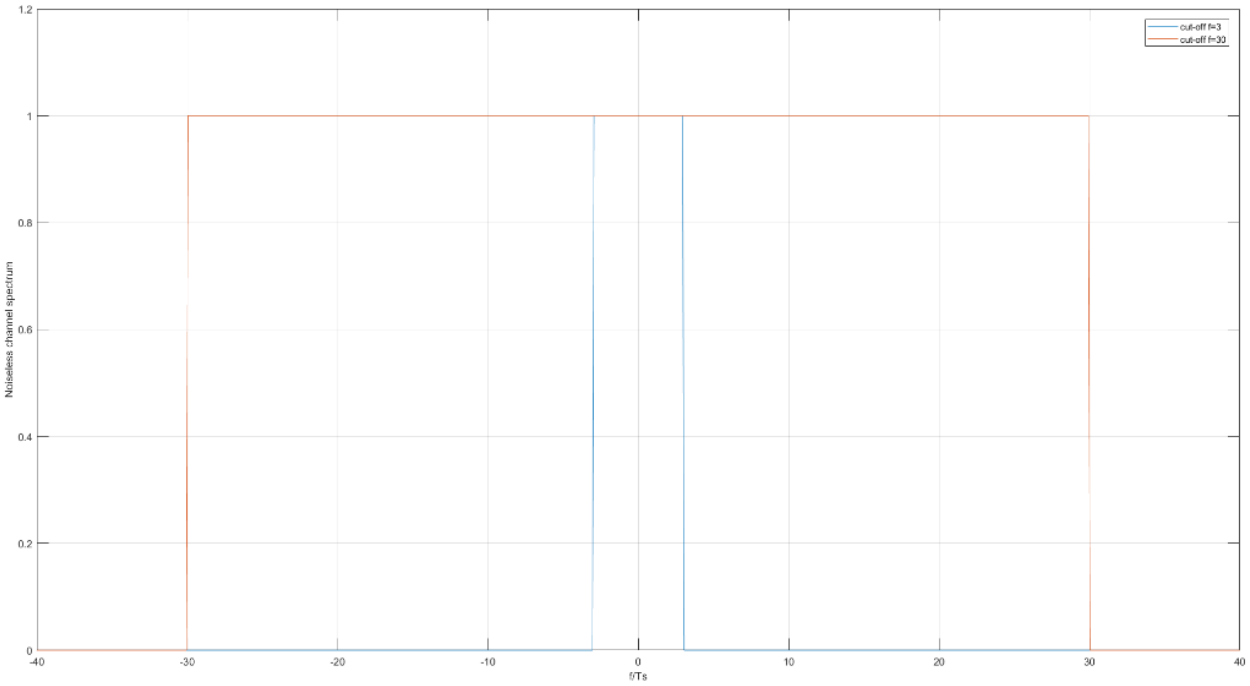

fig.7 Noiseless channel spectrum.

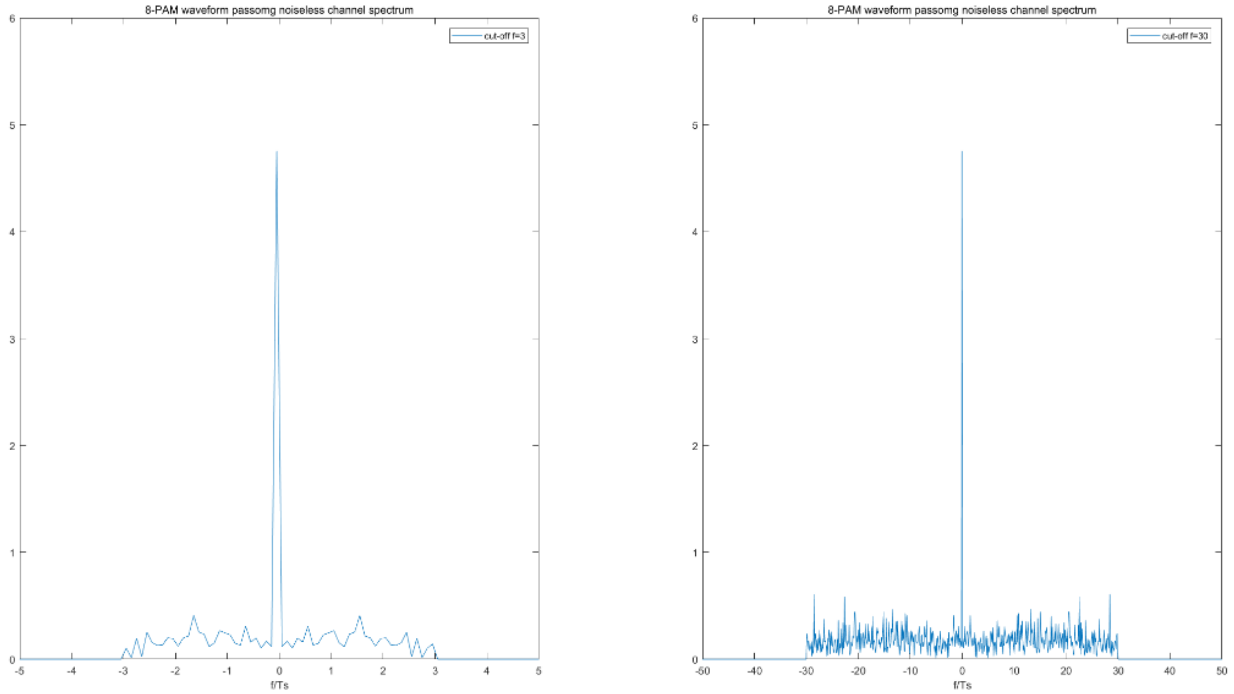

fig.8 8-PAM waveform passing noiseless channel spectrum.

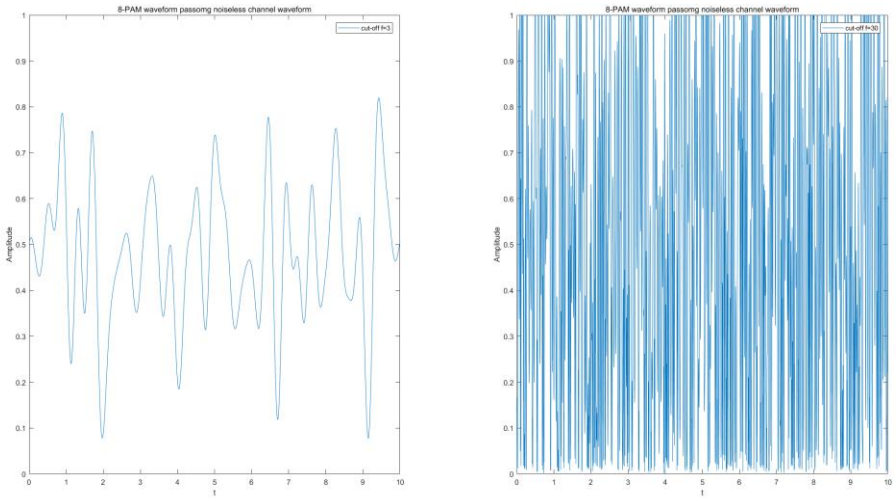

fig.9 8-PAM waveform passing noiseless channel waveform.

C. Discussion

The response of the noise-free band-limited channel is lower than the rectangular window of H(f)=1 at the cut-off frequency. After the 8-PAM signal passes through the band-limited channel, the high-frequency information is filtered out, and the restored information waveform becomes smooth; if Continue to increase the cut-off frequency, the restored signal waveform will contain more original information, and the same bit error rate will decrease accordingly.

The result of time domain convolution is the same as the result of inverse Fourier transform after frequency multiplication.

III. Task_3

A. Code

lab5_3.m

```
%% Validation of band-limited signal design with zero ISI .

clear;
clc;

%% Generate a single symbol
Tb1 = 1;                  % Symbol period.
Tb2 = 0.5;
fs = 1000;                % Sampling rate.
dt = 1/fs;                % Sampling interval.
N_sample1 = Tb1*fs;         % Number of sampling points per symbol.
N_sample2 = Tb2*fs;
N = 8;                   % Number of symbols.
t1 = 0 : dt : (N * N_sample1 - 1) * dt; % Sequence transmission
time.
t2 = 0 : dt : (N * N_sample2 - 1) * dt;
gt1 = [ones(1, N_sample1)];
gt2 = [ones(1, N_sample2)];
base = [-1 -1 1 -1 1 -1 1 -1]; % 0 1 basic sequence.
st1 = [];st2 = [];

% Generate sequence.
for i = 1 : N
   if base(i)==1
      st1 = [st1 gt1];
   else
      st1 = [st1 -1*gt1];
   end
end
for i = 1 : N
   if base(i)==1
      st2 = [st2 gt2];
   else
      st2 = [st2 -1*gt2];
   end
end
```

```
figure(1);
plot(t1, st1);grid on;
title('Raw binary sequence(no ISI Tb=1)');
figure(2);
plot(t2, st2);grid on;
title('Raw binary sequence(ISI Tb=0.5)');

% Ideal rectangular low-pass filter.
h = sinc(t1);
figure(3);
plot(t1,h);grid on;
title('Ideal rectangular low-pass filter')

figure(4);
st_NoISI = conv(h,st1)*dt;
st_NoISI = st_NoISI(1:length(t1));
plot(t1,st_NoISI);grid on;hold on;
n=0.5:1:7.5;
stem(n,base);title('No ISI');

figure(5);
st_ISI = conv(h,st2)*dt;
st_ISI = st_ISI(1:length(t2));
plot(t2,st_ISI);grid on;hold on;
n=0.25:0.5:3.75;
stem(n,base);title('ISI');
```

B. Figure

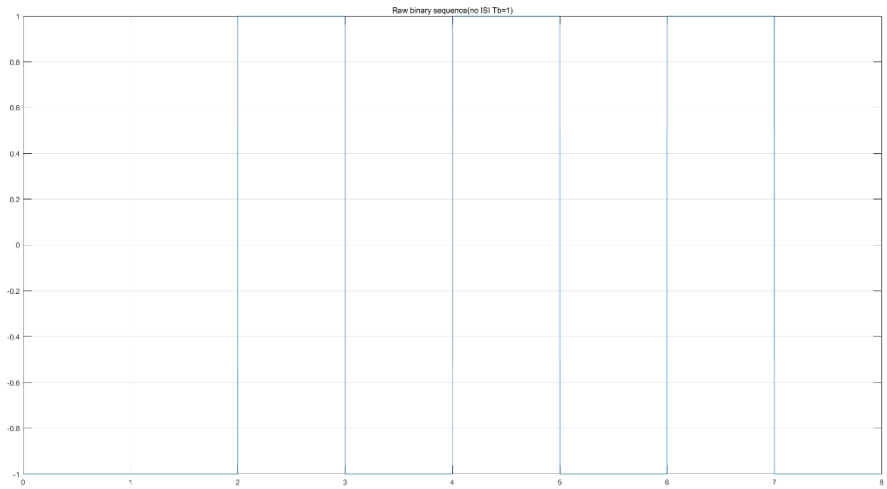

Fig.10 Raw binary sequence(no ISI Tb=1).

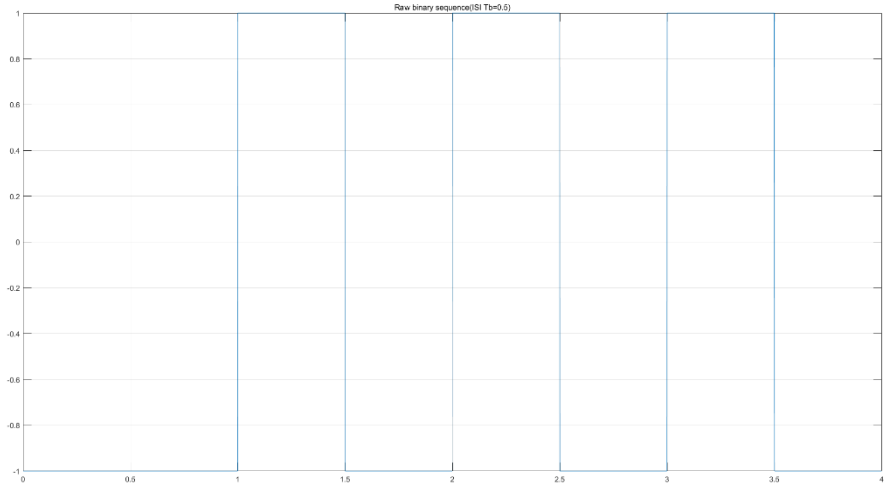

Fig.11 Raw binary sequence(ISI Tb=0.5).

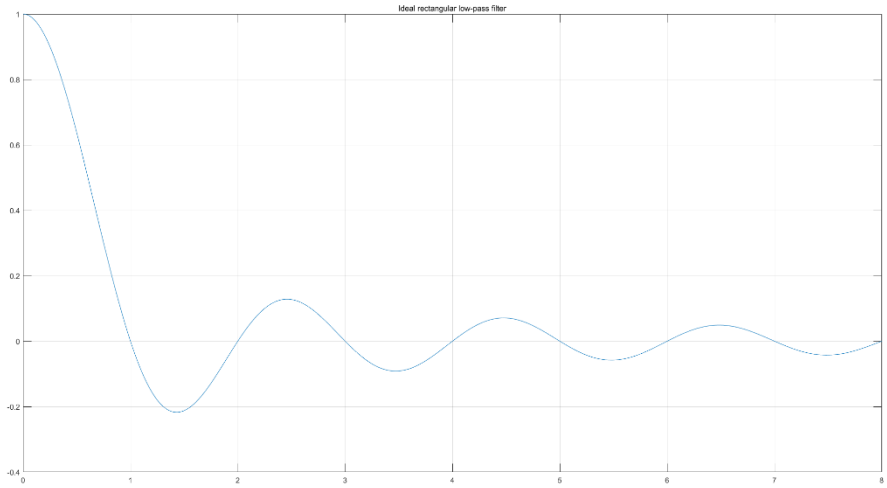

Fig.12 Ideal rectangular low-pass filter.

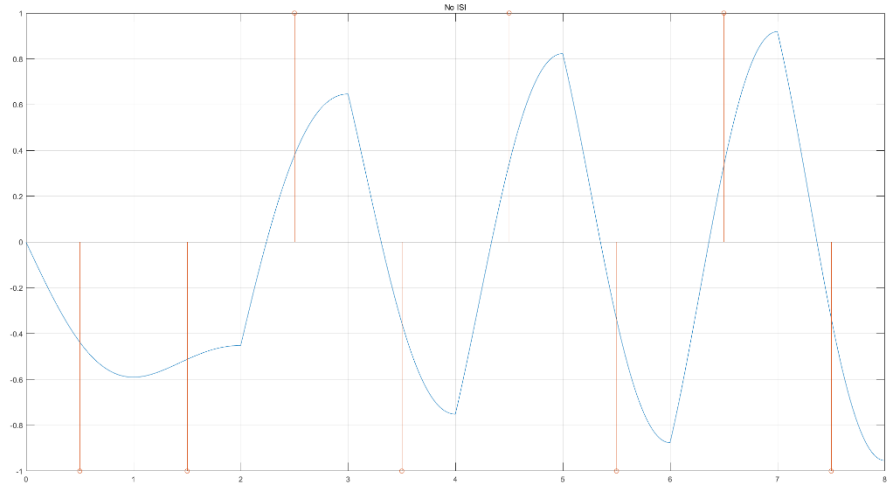

Fig.13 No ISI.

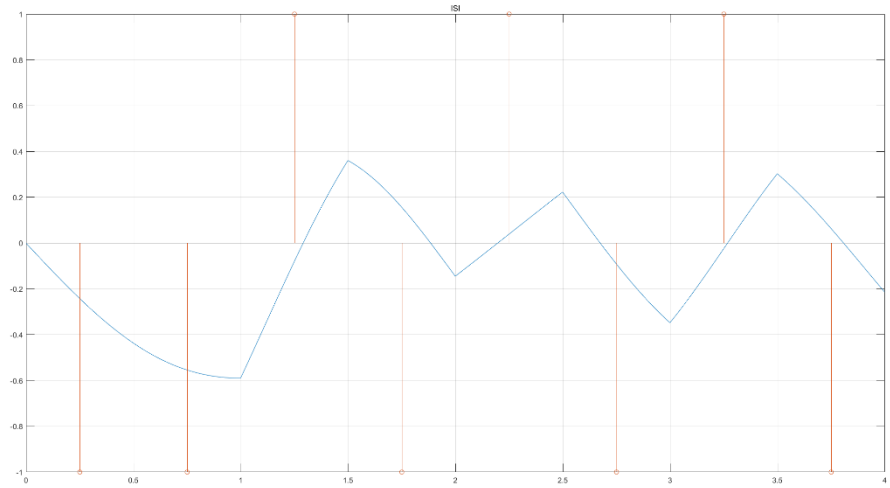

Fig.14 ISI.

C. Discussion

Only when there is a maximum value at the sampling time of this symbol, and it is 0 at the sampling time of other symbols, the inter-symbol interference (ISI) can be eliminated. Time domain condition:

$$x(nT) = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases}$$

Fourier transform X(f):

$$\sum_{m=-\infty}^{\infty} \mathrm{X}(f+\frac{m}{T}) = T$$

The physical meaning is to cut X(f) at intervals of 1/T on the frequency axis, and then translate it into the $\left(-\frac{1}{T_b}, \frac{1}{T_b}\right)$ interval along the frequency axis in segments, and superimpose them, which is equivalent to an ideal rectangular low-pass filter to eliminate ISI.

In fig.13, we can see that after the result of the ideal filter, if it is greater than 0, it is judged as 1, and if it is less than 0, it is judged as -1, and ISI will not appear. In Fig.14, because Tb=0.5, the Nyquist first is not satisfied. The law, ISI appeared.