浙江工业大学

MATLAB & Communication Simulations

Lab Report7

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: Jun. 13th,2021

I. Task_1

Consider 16QAM modulation. The signals are:

$$u_m(t) = A_{mc} g_T(t) \cos 2\pi f_c t + A_{ms} g_T(t) \sin 2\pi f_c t, \quad m = 1, 2, \ldots, M$$

where:

$$A_{mc} = (2m-3)d, m = 0, 1, 2, 3$$
$$A_{ms} = (2m-3)d, m = 0, 1, 2, 3$$

$$g_T(t) = \begin{cases} \sqrt{\dfrac{2}{T}}, & 0 \le t \le T \\ 0, & otherwise \end{cases}$$

where d=1, T=1, fc=5Hz.

Simulate the **bit error rate** of 16QAM system when SNR (average bit SNR) equals 5dB.

A. Code

lab7_1.m

```
%% Consider the following 16QAM modulation, where T=1.

clear;
clc;

%% The signal.
% The corresponding 16QAM signal.
% T = 1;                  % Symbol interval.
% d = 1;
% Tb = T/4;               % Bit interval.
% m = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16];   % Equally
spaced.
% M = 16;                 % QAM:M=16
% A_mc = (2*m-3)*d;       % The amplitude of the m-th waveform.
% A_ms = (2*m-3)*d;
% fc = 5;                 % Carrier frequency.
```

```
% gT=sqrt(2/T);         % Transmission pulse.
% t = 1:1:16;
% for i =1:1:16
%    u_m(t) =
A_mc(t)*gT.*cos(2*pi*fc*t)+A_ms(t)*gT.*sin(2*pi*fc*t);
% end

%% Task(1) Simulate the bit error rate of 16QAM system
% when SNR (average bit SNR) equals 5dB.
SNRindB=5;
[smld_err_p,smld_err_pb] = smldpe_QAM_16(SNRindB)   % simulated
error rate
```

smldpe_QAM_16.m

```
function [p,pb]=smldpe_QAM_16(snr_in_dB)
% [p,pb]=smldpe_QAM_16(snr_in_dB)
%      smldpe_QAM_16  finds the probability of error for the given
%          value of snr_in_dB, SNR in dB.
N=10000;
d=2;                          % min distance between symbols
Eav=10*d^2;                   % energy per symbol
snr=10^(snr_in_dB/10);        % SNR per bit (given)
sgma=sqrt(Eav/(8*snr));       % noise variance
M=16;
% Generation of the data source follows.
for i=1:N
  temp=rand;                  % a uniform R.V. between 0 and 1
  dsource(i)=1+floor(M*temp); % a number between 1 and 16, uniform
end
% Mapping to the signal constellation follows.
mapping=[-3*d  -3*d;
          -d  -3*d;
           d  -3*d;
         3*d  -3*d;
        -3*d    -d;
          -d    -d;
           d    -d;
         3*d    -d;
        -3*d     d;
          -d     d;
           d     d;
         3*d     d;
        -3*d  3*d;
```

```
          -d  3*d;
           d  3*d;
         3*d  3*d];
% Mapping to the signal constellation follows as bit.
bit_mapping=[0 0  0 0;
             0 1  0 0;
             1 0  0 0;
             1 1  0 0;
             0 0  0 1;
             0 1  0 1;
             1 0  0 1;
             1 1  0 1;
             0 0  1 0;
             0 1  1 0;
             1 0  1 0;
             1 1  1 0;
             0 0  1 1;
             0 1  1 1;
             1 0  1 1;
             1 1  1 1];
for i=1:N
  qam_sig(i,:)=mapping(dsource(i),:);
  % bit_qam_sig(i,1:4)=bit_mapping(dsource(i),:,:,:);
end
% received signal
for i=1:N
  [n(1) n(2)]=gngauss(sgma);
  r(i,:)=qam_sig(i,:)+n;
end
% detection and error probability calculation
numoferr=0;
numoferr_pb=0;
for i=1:N
  % Metric computation follows.
  for j=1:M
    metrics(j)=(r(i,1)-mapping(j,1))^2+(r(i,2)-mapping(j,2))^2;
  end
  [min_metric decis] = min(metrics);
  for k=1:4
    if(bit_mapping(decis,k)~=bit_mapping(dsource(i),k))
        numoferr_pb=numoferr_pb+1;
    end
  end
  if (decis~=dsource(i))
```

```
    numoferr=numoferr+1;
  end
end
p=numoferr/(N);                         % probability of error estimate
pb=numoferr_pb/(N*4);                   % probability of bit error
estimate
```

Sampling.m

```
function [T,Samp_Sig]=Sampling(t,Fs,sig)
% Fucntion Name:Sampling
%Input: Tb,Fs:sig OutPut:Samp_Sig
%When you call the Function ,u input the tiem for a bit,the
%Sampling rate and the source signal,then output the Samplint
Signal.
Ts=1/Fs;
Sig=sig;
len=length(Sig);
T=0:Ts:len*t-Ts;
Samp_Sig=T;
for i=0:1:len-1
    for j=1:1:t/Ts
        Samp_Sig(i*t/Ts+j)=Sig(i+1);
    end
end
```

Qfunct.m

```
function [y]=Qfunct(x)
% [y]=Qfunct(x)
%       QFUNCT  evaluates the Q-function.
%           y = 1/sqrt(2*pi) * integral from x to inf of exp(-
t^2/2) dt.
%           y = (1/2) * erfc(x/sqrt(2)).
y=(1/2)*erfc(x/sqrt(2));
```

gngauss.m

```
function [gsrv1,gsrv2]=gngauss(m,sgma)
% [gsrv1,gsrv2]=gngauss(m,sgma)
% [gsrv1,gsrv2]=gngauss(sgma)
% [gsrv1,gsrv2]=gngauss
%       GNGAUSS  generates two independent Gaussian random
variables with mean
```

```
%          m and standard deviation sgma. If one of the input
arguments is missing
%          it takes the mean as 0, and the standard deviation as
the given parameter.
%          If neither mean nor the variance is given, it generates
two standard
%          Gaussian random variables.
if nargin == 0,
  m=0; sgma=1;
elseif nargin == 1,
  sgma=m; m=0;
end;
u=rand;                          % a uniform random variable in
(0,1)
z=sgma*(sqrt(2*log(1/(1-u))));      % a Rayleigh distributed
random variable
u=rand;                          % another uniform random variable
in (0,1)
gsrv1=m+z*cos(2*pi*u);
gsrv2=m+z*sin(2*pi*u);
```

B. Figure

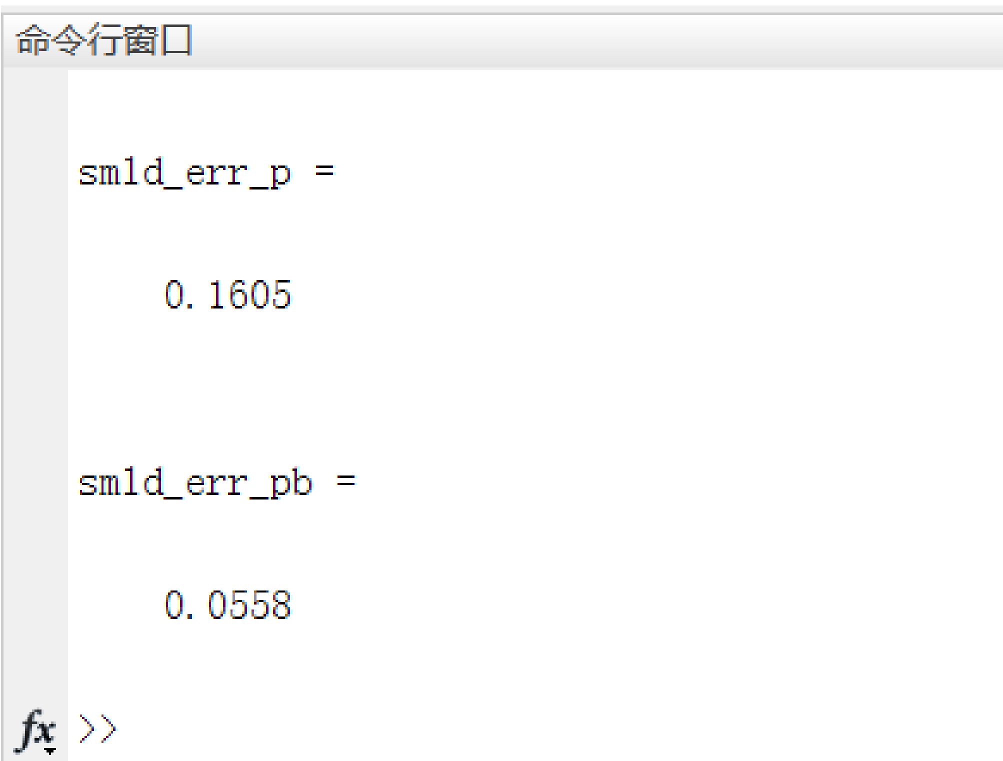


Fig.1 Bit error rate of 16QAM system.

C. Discussion

Monte Carlo simulation of 16QAM signal, according to the definition, the shortest distance between two adjacent points on the constellation diagram is 2. During the simulation process, after adding noise, refer to the mapping to find the shortest mapping point after scrambling, and then calculate symbol error rate. For the bit error rate, each mapping point must be converted to 0 and 1. Each point of 16QAM needs to have four 0s or 1s to represent, that is, bit_mapping in the code. After obtaining the mapping point from the scrambled signal, it is converted into a four-digit 01 sequence, and the number of error bits is counted in comparison with the original mapping table to calculate the bit error rate.

II. Task_2

(1) The waveform can be defined by the following form.

$$u_m(t) = A_{mc} g_T(t)\cos 2\pi f_c t + A_{ms} g_T(t)\sin 2\pi f_c t, \quad m = 1, 2, ..., M$$

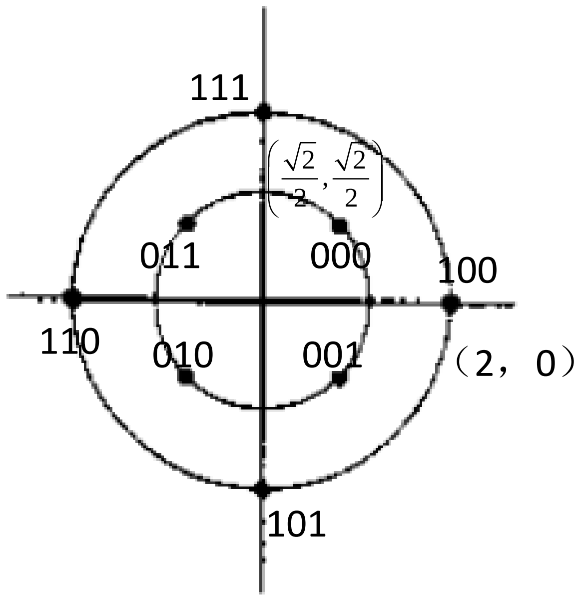
$$g_T(t) = \begin{cases} \sqrt{\frac{2}{T}}, & 0 \le t \le T \\ 0, & otherwise \end{cases}$$

Give the values of A_{mc} and A_{ms} for each constellation point.（What are the corresponding basis signals?）

(2) Plot all possible 8QAM waveforms in time domain, sampling frequency fs=1000Hz and fc=5Hz.

(3) How to calculate the average energy per symbol and per bit of 8QAM, respectively?

(4) Simulate the symbol error rate of 8QAM when SNR-per-bit=5dB.



$M = 8$

A. Code

lab7_2.m

```
%% Consider the following 8QAM modulation.

clear;
clc;

% The corresponding 8QAM signal.
T = 1;                     % Symbol interval.
d = 1;
Tb = T/3;                  % Bit interval.
m = [1,2,3,4,5,6,7,8];     % Equally spaced.
M = 8;                     % QAM:M=8
gT=sqrt(2/T);              % Transmission pulse.

%% Task(1) Give the values of Amc and Ams for each constellation
point.
% The amplitude of the m-th waveform.
A_mc = [sqrt(1/2),sqrt(1/2),-sqrt(1/2),-sqrt(1/2),2,0,-2,0]*d
A_ms = [sqrt(1/2),-sqrt(1/2),-sqrt(1/2),sqrt(1/2),0,-2,0,2]*d

%% Task(2) Plot all possible 8QAM waveforms.
fc = 5;                    % Carrier frequency.
fs = 1000;                 % Sampling frequency.
figure('NumberTitle', 'off', 'Name', '8QAM Time domain
waveforms');
for i=1:1:M
    m_c=A_mc(i)*gT;
    m_s=A_ms(i)*gT;
    % Sampling.
    [~,x_c] = Sampling(T,fs,m_c);
    [t,x_s] = Sampling(T,fs,m_s);
    % Carrier wave.
    carrier_c = cos(2*pi*fc*t);
    carrier_s = sin(2*pi*fc*t);
    % QAM modulation.
    u_m=x_c.*carrier_c+x_s.*carrier_s;
    % Plotting commands follow.
    subplot(2,4,i)
    plot(t,u_m);
    title(sprintf('第%d个时域波形',i))
    grid on;
```

```
    xlabel('t/s')
    ylabel('u\_m(t)')
end
sgtitle('8QAM Time domain waveforms')

%% Task(3) Calculate the average energy per symbol and per bit.
Es = sum(A_mc.^2+A_ms.^2)/8
Eb = Es/3

%% Task(4) Simulate the symbol error rate of 8QAM when SNR-per-
bit=5dB.
% when SNR (average bit SNR) equals 5dB.
SNRindB=5;
[smld_err_p,smld_err_pb] = smldpe_QAM_8(SNRindB,Es)    % simulated
error rate
```

smldpe_QAM_8.m

```
function [p,pb]=smldpe_QAM_8(snr_in_dB,Es)
% [p,pb]=smldpe_QAM_8(snr_in_dB,Es)
%       smldpe_QAM_8  finds the probability of error for the given
%           value of snr_in_dB, SNR in dB.
N=10000;
d=sqrt(2);                    % min distance between symbols
Eav=Es*d^2;                   % energy per symbol
snr=10^(snr_in_dB/10);        % SNR per bit (given)
sgma=sqrt(Eav/(6*snr));       % noise variance
M=8;
% Generation of the data source follows.
for i=1:N
  temp=rand;                  % a uniform R.V. between 0 and 1
  dsource(i)=1+floor(M*temp); % a number between 1 and 8, uniform
end
% Mapping to the signal constellation follows.
mapping=[ sqrt(1/2)*d    sqrt(1/2)*d;
         sqrt(1/2)*d   -sqrt(1/2)*d;
        -sqrt(1/2)*d   -sqrt(1/2)*d;
        -sqrt(1/2)*d    sqrt(1/2)*d;
               2*d           0*d;
               0*d          -2*d;
              -2*d           0*d;
               0*d           2*d;];
% Mapping to the signal constellation follows as bit.
bit_mapping=[0 0 0;
```

```
            0 0 1;
            0 1 0;
            0 1 1;
            1 0 0;
            1 0 1;
            1 1 0;
            1 1 1;];
for i=1:N
  qam_sig(i,:)=mapping(dsource(i),:);
  % bit_qam_sig(i,1:3)=bit_mapping(dsource(i),:,:);
end
% received signal
for i=1:N
  [n(1) n(2)]=gngauss(sgma);
  r(i,:)=qam_sig(i,:)+n;
end
% detection and error probability calculation
numoferr=0;
numoferr_pb=0;
for i=1:N
  % Metric computation follows.
  for j=1:M
    metrics(j)=(r(i,1)-mapping(j,1))^2+(r(i,2)-mapping(j,2))^2;
  end
  [min_metric decis] = min(metrics);
  for k=1:3
    if(bit_mapping(decis,k)~=bit_mapping(dsource(i),k))
        numoferr_pb=numoferr_pb+1;
    end
  end
  if (decis~=dsource(i))
    numoferr=numoferr+1;
  end
end
p=numoferr/(N);                    % probability of error estimate
pb=numoferr_pb/(N*3);              % probability of bit error estimate
```

B. Figure

命令行窗口

```
A_mc =

    0.7071    0.7071   -0.7071   -0.7071    2.0000         0   -2.0000         0


A_ms =

    0.7071   -0.7071   -0.7071    0.7071         0   -2.0000         0    2.0000
```

Fig.2 Corresponding basis signals.

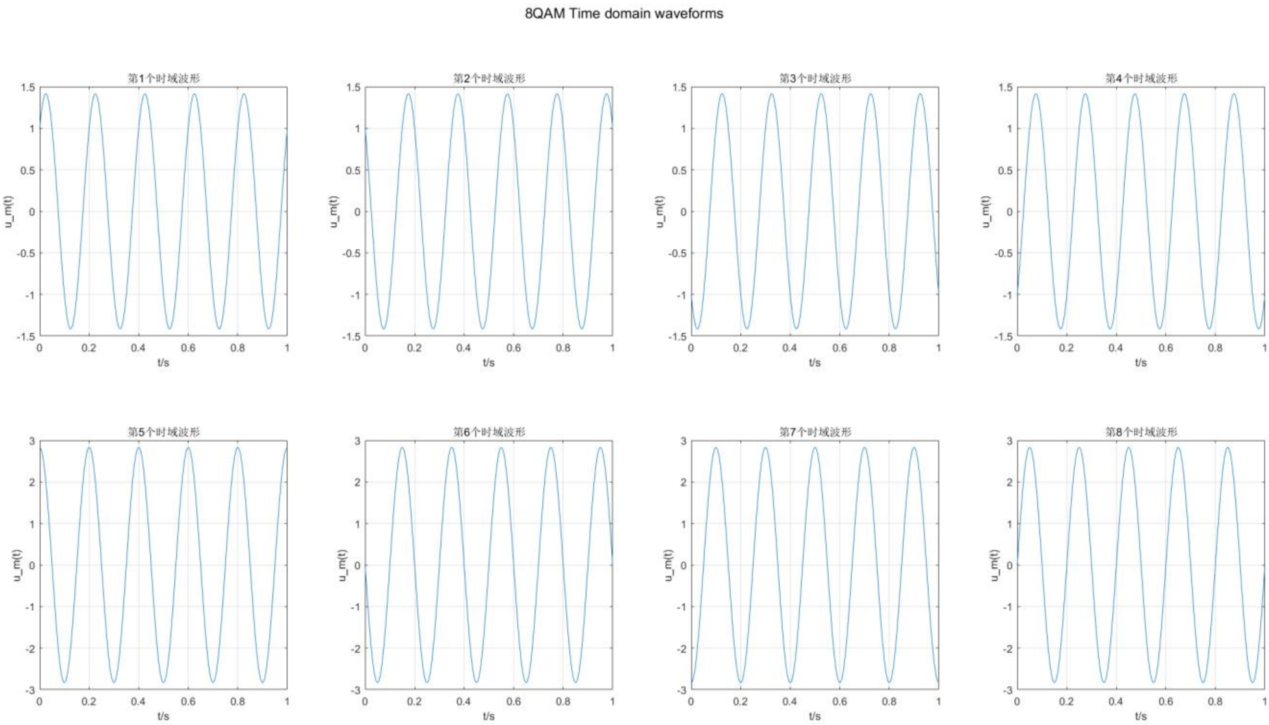


Fig.3 8QAM waveforms.

命令行窗口

```
Es =

    2.5000


Eb =

    0.8333
```

Fig.4 Energy.

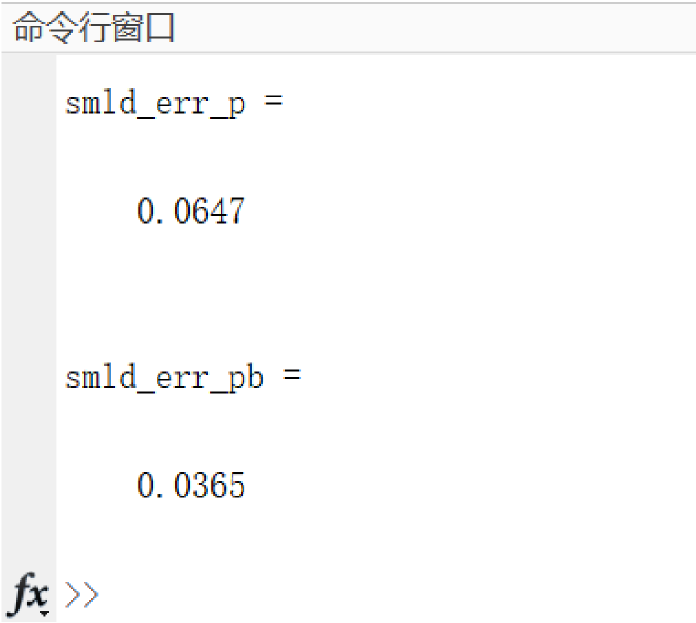


Fig.5 Message bits in Time domain(4ASK).

C. Discussion

This 8-QAM mapping is quite special. There are a total of eight constellation points in the inner ring and the outer ring, and there is no fixed rule for the two coordinates of each point. The 8QAM mapping is similar to the 16QAM in the previous example, except that the mapping relationship is different.

$$E_s = \frac{1}{8}\sum_{k=1}^{8}\int_0^T u_m(t)^2 dt = \frac{1}{8}\sum_{k=1}^{8}(A_{mc}{}^2 + A_{ms}{}^2) = 2.5$$

III. Task_3

$$u_m(t) = \sqrt{\frac{2E_s}{T}}\cos(2\pi f_c t + 2\pi m\Delta f t), \quad m = 0,1,\ldots,M-1, \quad 0 \le t \le T$$

where $f_c = 10Hz$, $E_s = 1$, $T = 1$, $M = 1$, $\Delta f = 2Hz$

Plot all possible 3FSK waveforms in both time and frequency domain. The sampling frequency fs=1000Hz

A. Code

lab7_3.m

```
%% Consider the following 3FSK modulation.

clear;
clc;

%% Task(1) Plot all possible 3FSK waveforms in both time and
frequency domain.
T = 1;                      % Symbol interval.
Es = 1;                     % Energy per symbol.
fc = 5;                     % Carrier frequency.
fs = 1000;                  % Sampling frequency.
delta_f = 2;
M = 3;
amplitude = square(2*Es/T);
figure('NumberTitle', 'off', 'Name', '3FSK waveforms');
for m=1:1:M
    % Sampling.
    [t,x] = Sampling(T,fs,amplitude);
    % Carrier wave.
    carrier = cos(2*pi*fc*t+2*pi*m*delta_f*t);
    % QAM modulation.
    u_m=x.*carrier;
    % T2F.
    [sf,U_m]=T2F(t,u_m);
    % Plotting commands follow.
    subplot(2,3,m)
    plot(t,u_m);
    grid on;
```

```
    title(sprintf('第%d个时域波形',m))
    xlabel('t/s')
    ylabel('u\_m(t)')
    subplot(2,3,m+3)
    plot(sf,abs(U_m));
    grid on;
    title(sprintf('第%d个频域波形',m))
    axis([-30,30,0,0.6])
    xlabel('f/Hz')
    ylabel('U\_m')
end
sgtitle('3FSK waveforms')
```

T2F.m

```
function [f,sf]=T2F(t,st)
%input is time and the signal vectors
%output is freguency and signal spectrum

dt=t(2)-t(1);
T=t(end)-t(1)+dt;
df=1/T; %smapling rate
N=length(st);

f=-N/2*df:df:(N/2-1)*df; %频域抽样点
sf=fft(st);
sf=T/N*fftshift(sf).*exp(-j*2*pi*f*t(1));   %补偿时间移位
```

B. Figure

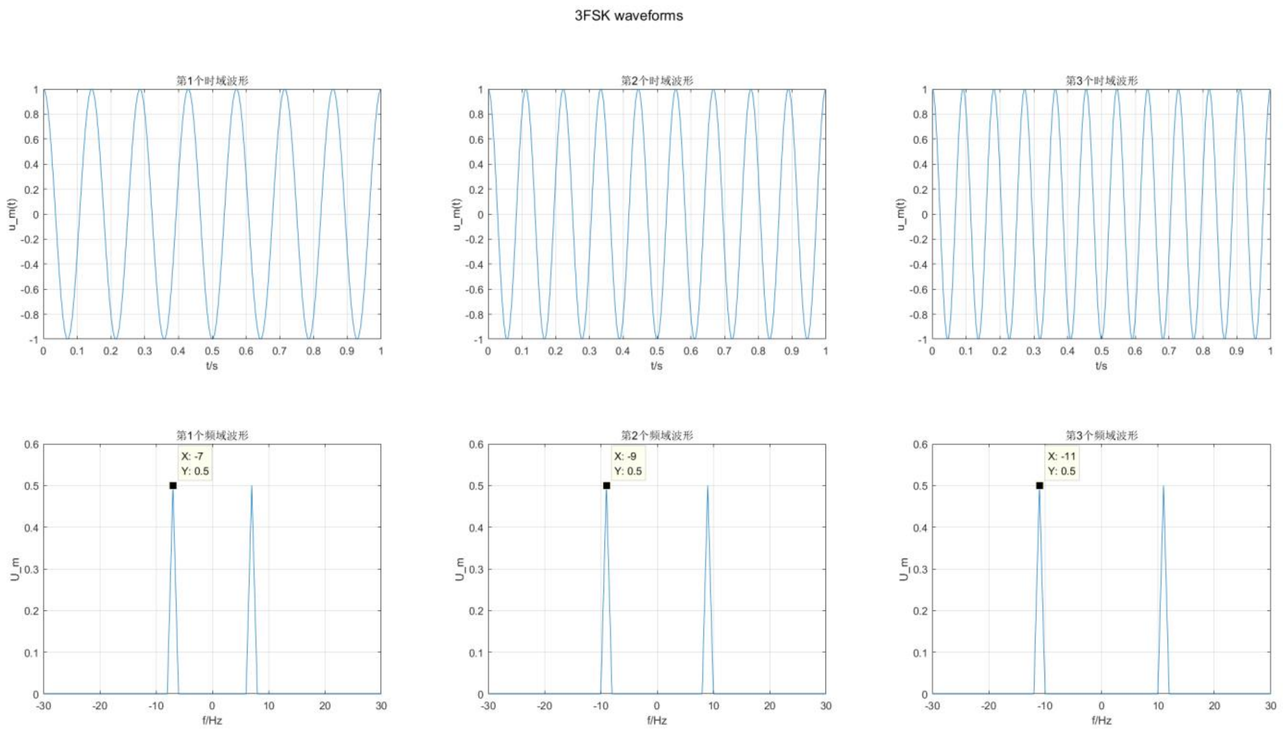


Fig.5 Message bits in Time domain(4ASK).

C. Discussion

Frequency shift keying FSK uses different frequencies to distinguish each signal. 3FSK uses three different signals, such as 5+2*1=7Hz, 5+2*2=9Hz, 5+2*3=11Hz in the frequencies spectrogram.