2016-2017 学年 第二学期

《通信原理与系统》课程设计报告

所在学院：电子工程学院

学生姓名：居昊

学生学号：2014020911033

任课老师：易伟

2017 年 6 月 7 日

（一）课程设计题目

- 标准题目：选择两种波形，s1(t),s2(t)，进行匹配滤波或者最优相关接收，画出各个部分的波形。
- 扩展题目：加入高斯白噪声，进行信噪比、门限对 Pe 的分析。

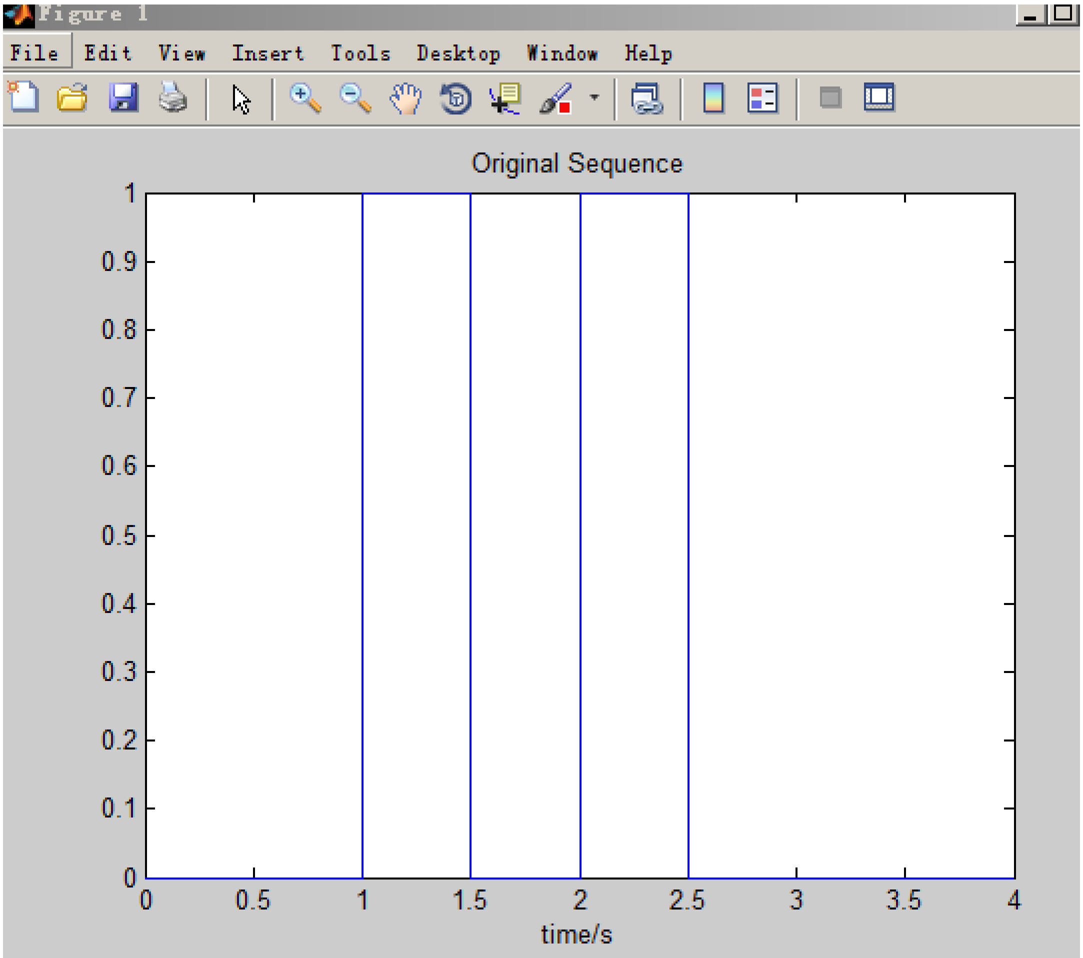
（二）课程设计仿真结果与讨论

1、仿真实验

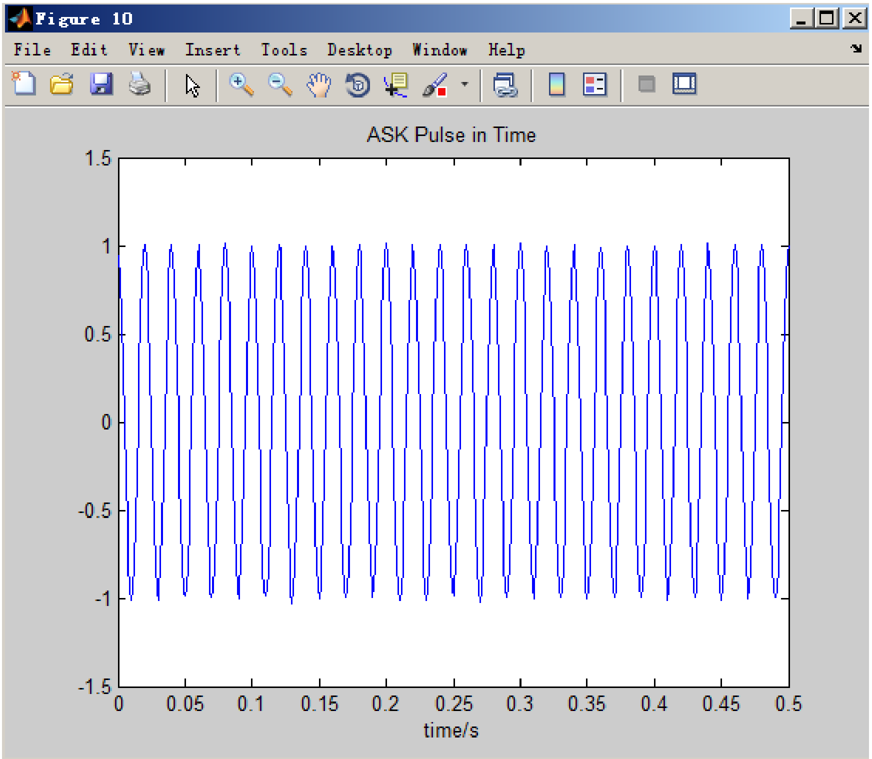
I. ASK

取采样频率 1000Hz，载波频率 50Hz.N 即随机序列码元数可调。为便于观察，先取 N=8

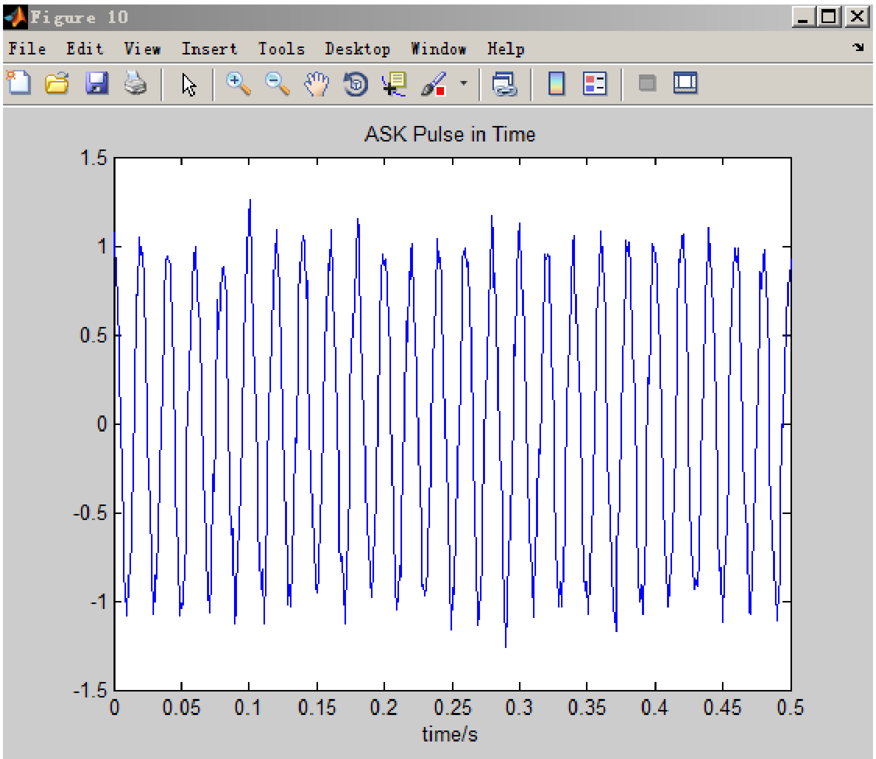
N=8 时生成随机序列如图



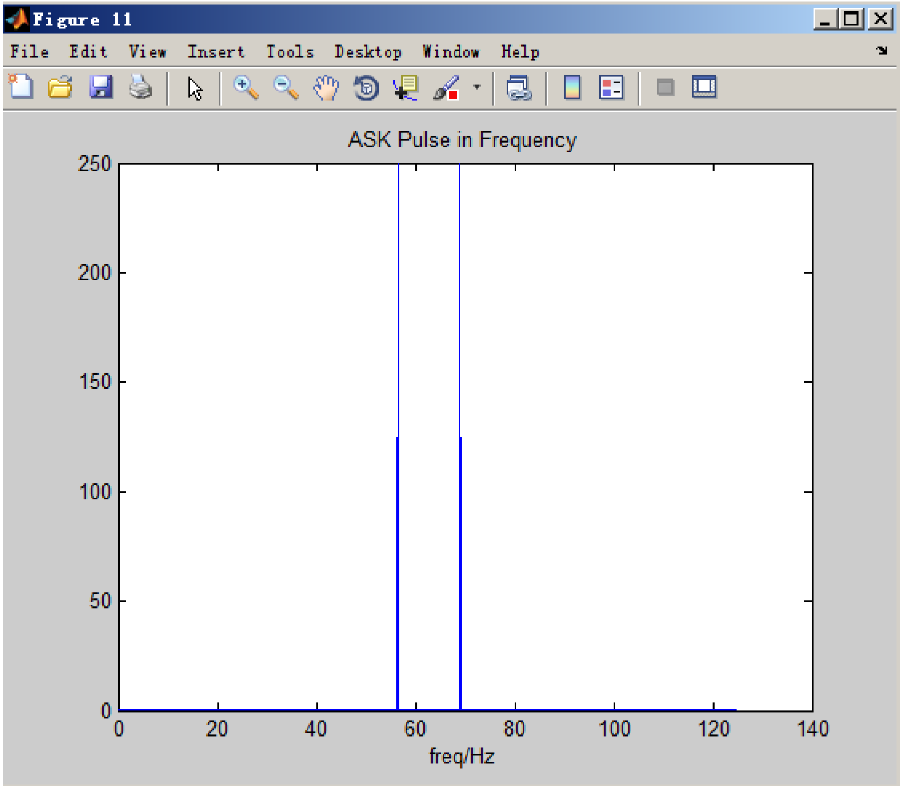
ASK 载波信号（SNR=40dB，信噪比很大，几乎看不出噪声的存在）



减小至 SNR=20dB，可看出较明显的噪声



ASK 信号对应频谱：（以下若无特殊说明则 SNR=40dB）



匹配滤波器的时域冲激响应函数与频谱：

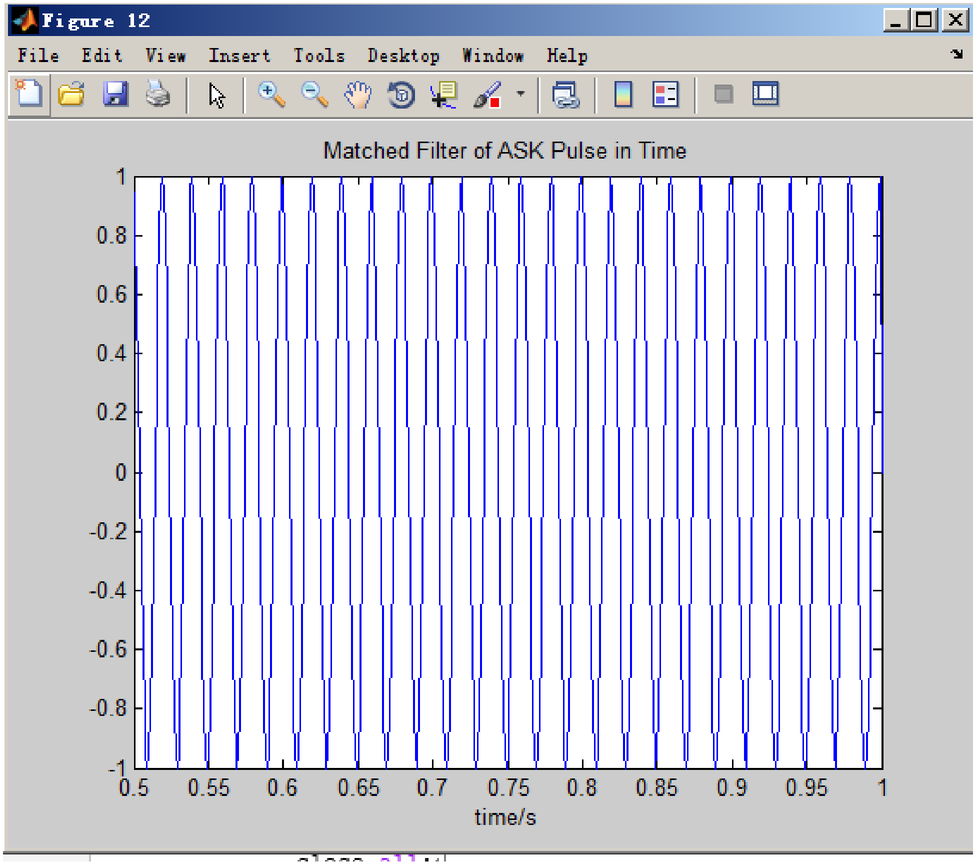


Figure 12
File Edit View Insert Tools Desktop Window Help
Matched Filter of ASK Pulse in Time
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0.5 0.55 0.6 0.65 0.7 0.75 0.8 0.85 0.9 0.95 1
time/s

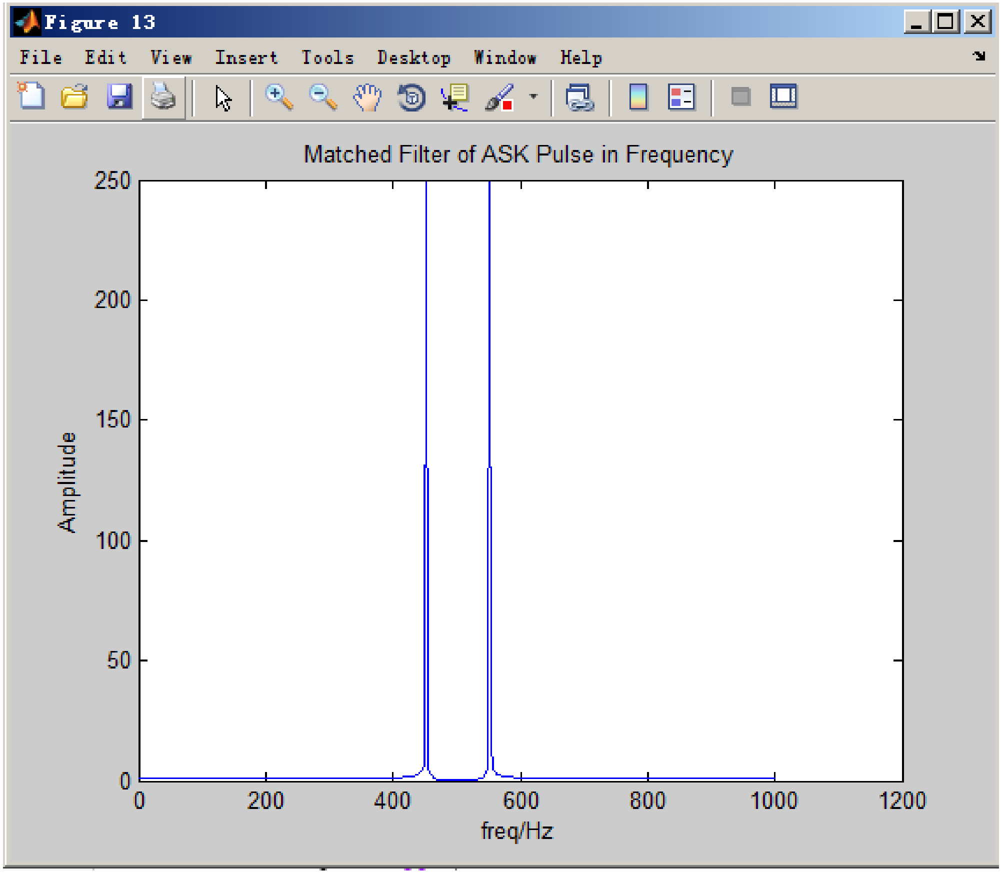
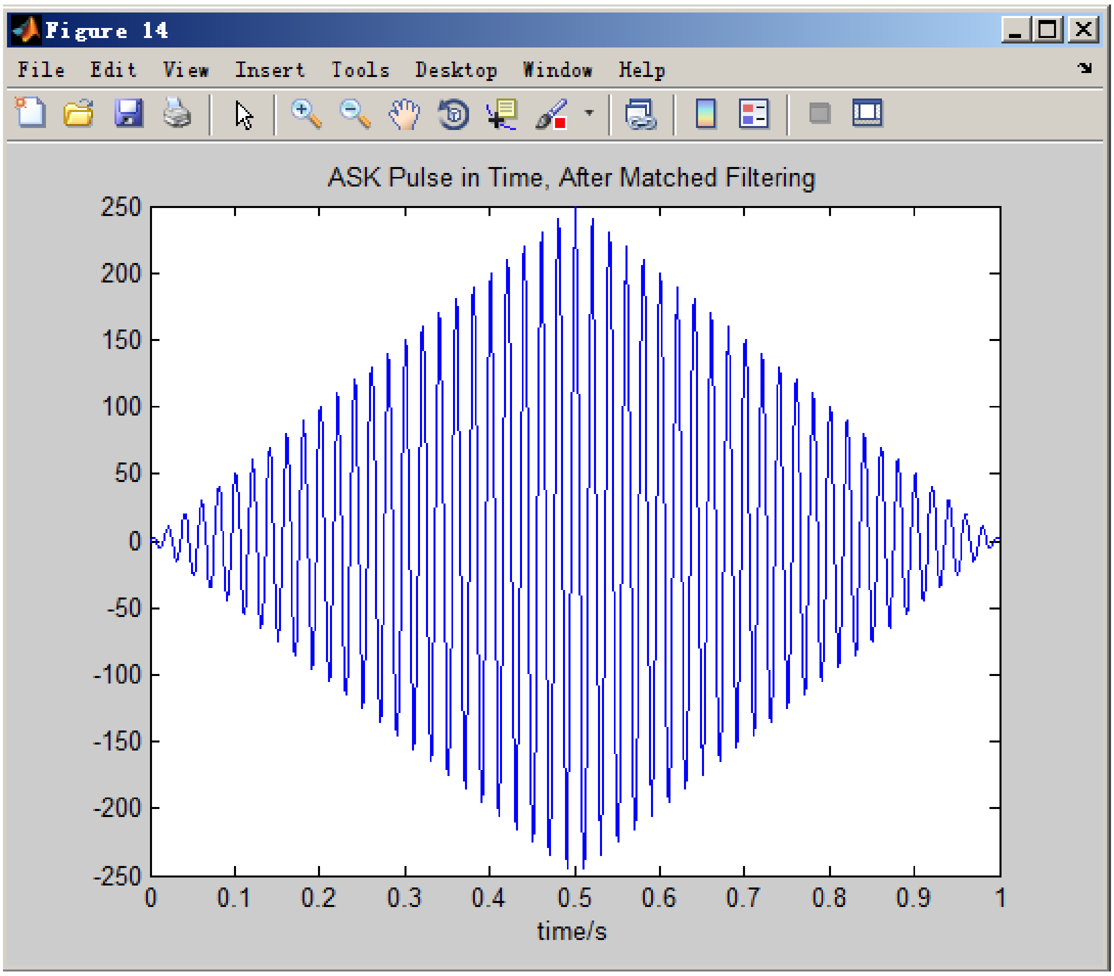
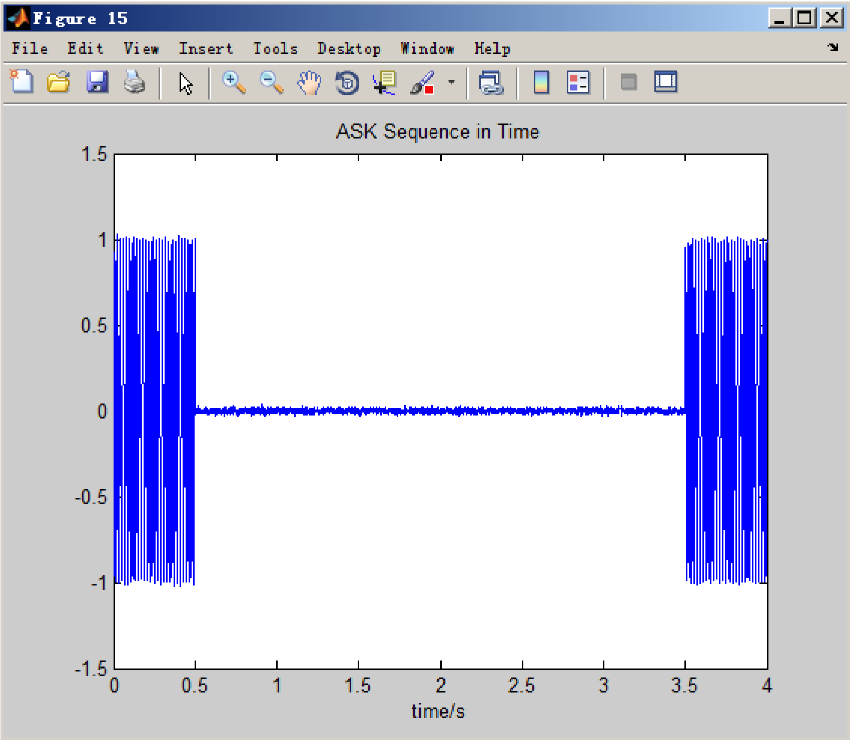


Figure 13
File Edit View Insert Tools Desktop Window Help
Matched Filter of ASK Pulse in Frequency
Amplitude
250
200
150
100
50
0
0 200 400 600 800 1000 1200
freq/Hz

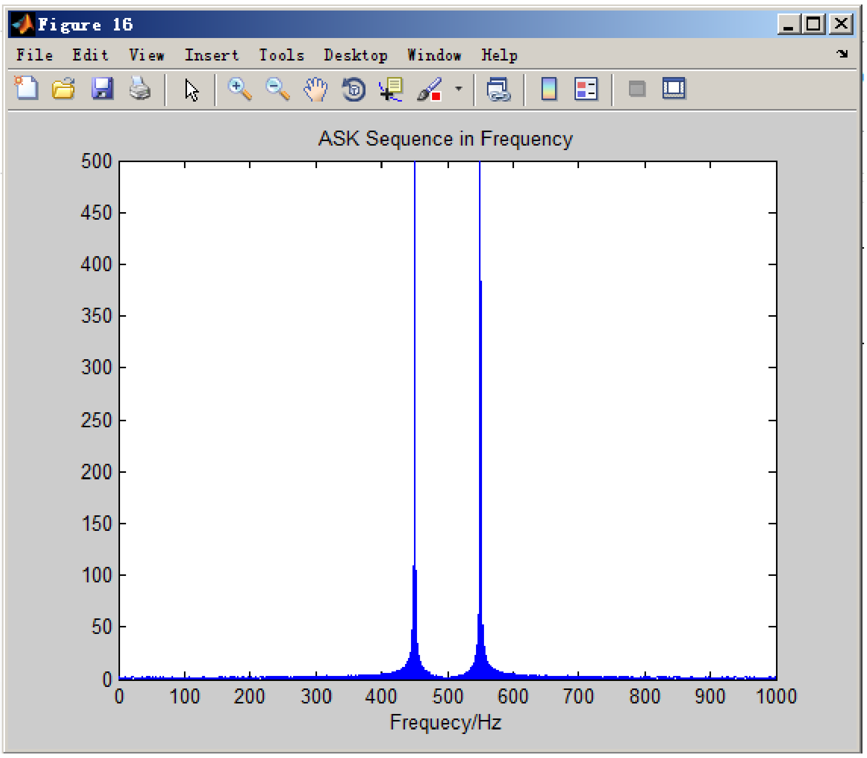
当单个脉冲通过匹配滤波器后，得到下图波形。可观察到信号幅度在 0.5s 即信号结束处达到最大值（也是输出 SNR 最大处）



对此前生成的随机序列进行调制，于是得到 ASK 序列，如图

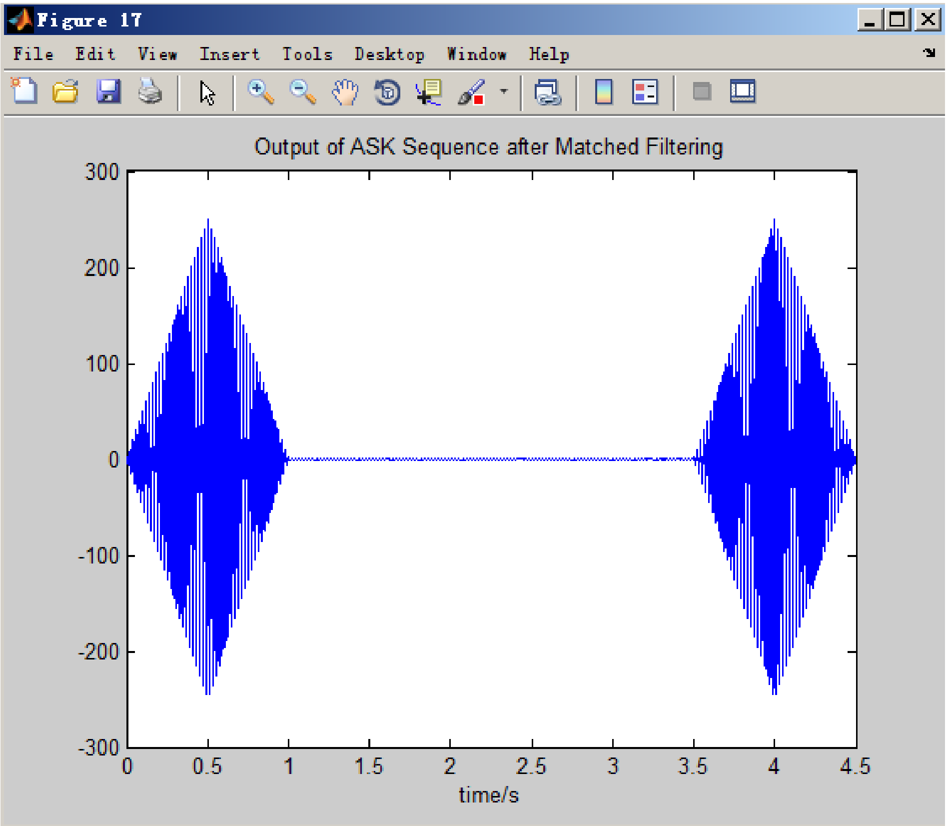


其频谱：

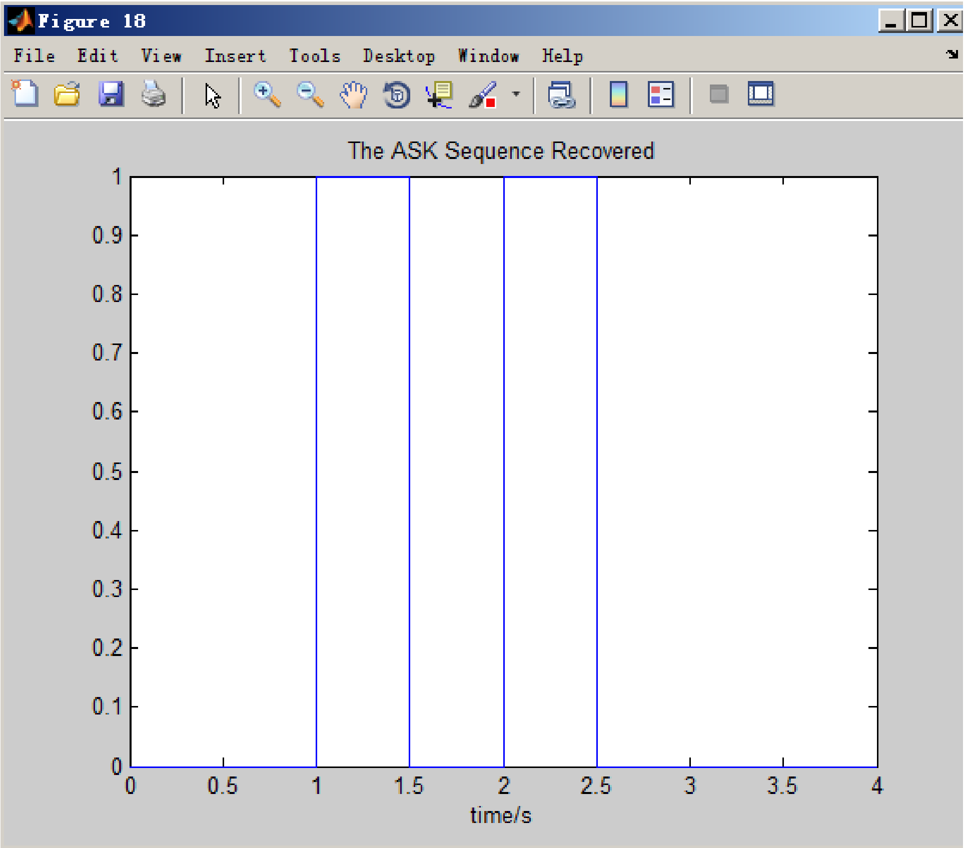


能观察到不明显的旁瓣（离散谱信号）

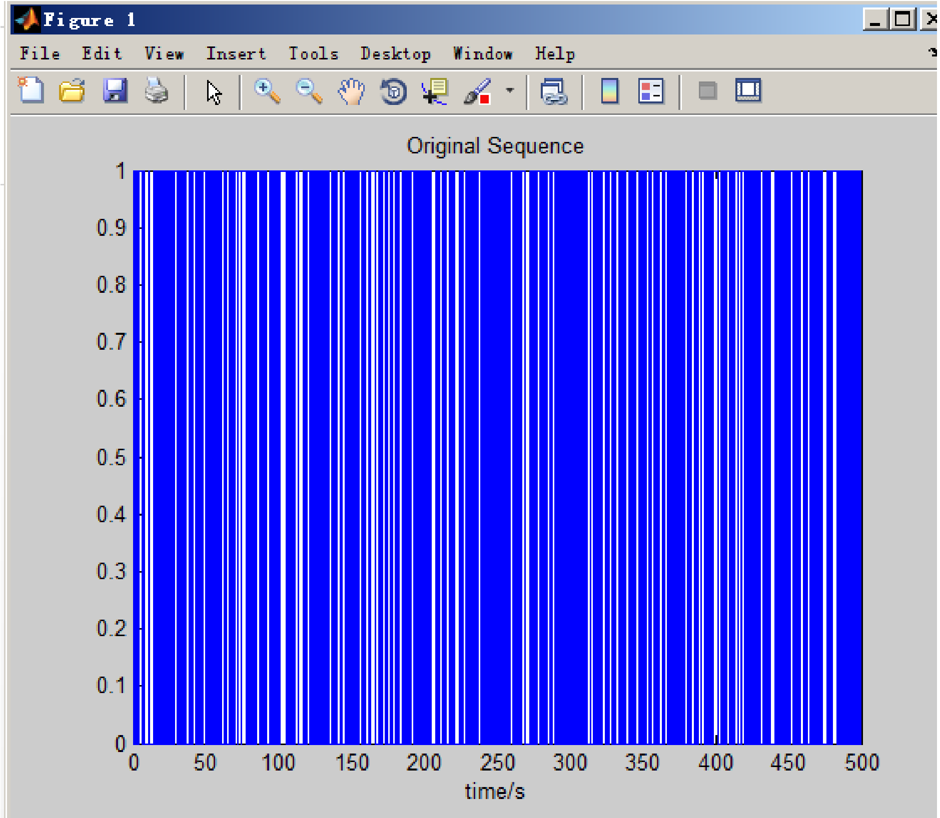
通过匹配滤波器，有：

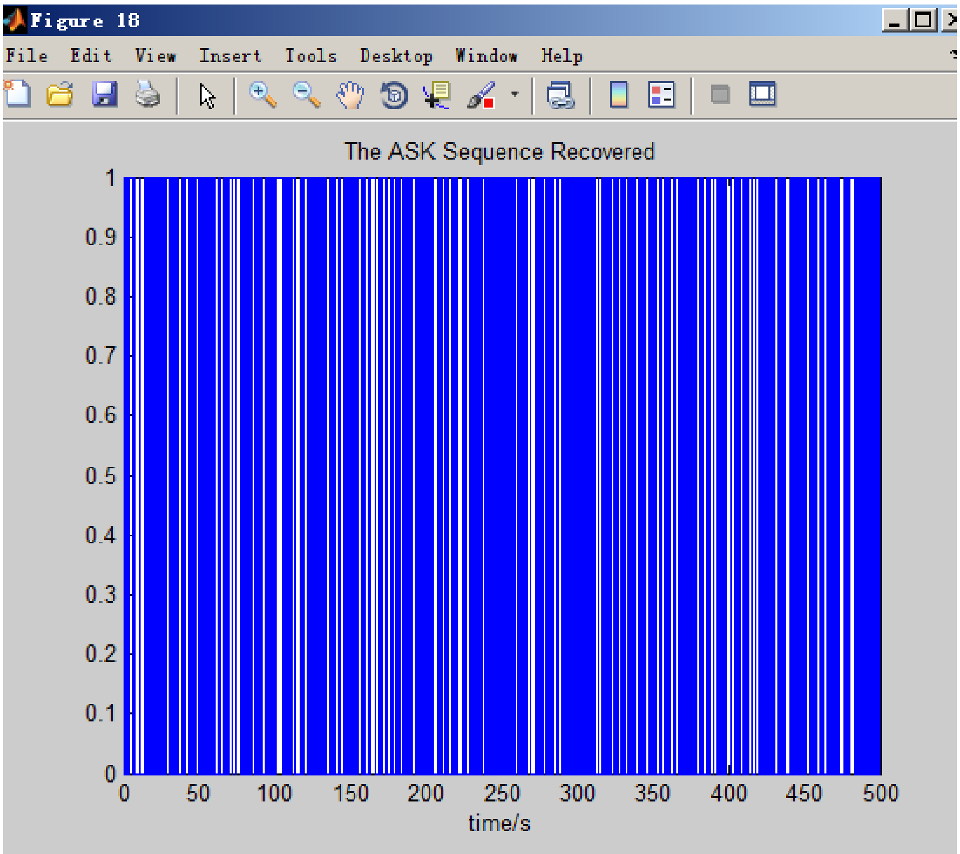


在 t=0.5is(i=1,2,…N)处取定时脉冲，经过保持电路，得到重建的信号



该信号与原信号一致。在放大到 N=1000 的序列里，依旧如此：





II. Linear Frequency Modulation

在信息对抗原理课上学到线性调频脉冲压缩（LFM）调制，常用于雷达，通过脉冲压缩调制，将长脉冲所含的大能量压缩进长度相对短的脉冲中(在雷达中意味着可以获得较大能量又能保持短脉冲的距离分辨率），获得大时宽带宽积信号，具有发射功率峰值低的优点（对于雷达而言意味着截获概率降低）。用于通信系统，可获得相对更高的信息传送速率，降低对发送机的功率要求。

线性调频脉冲信号类似 ASK 调制，包络也是宽度τ 的矩形脉冲，但信号的载频随时间线性变化，其瞬时角频率如(1.1)

$$\omega_i = \omega_0 + \mu t \tag{1.1}$$

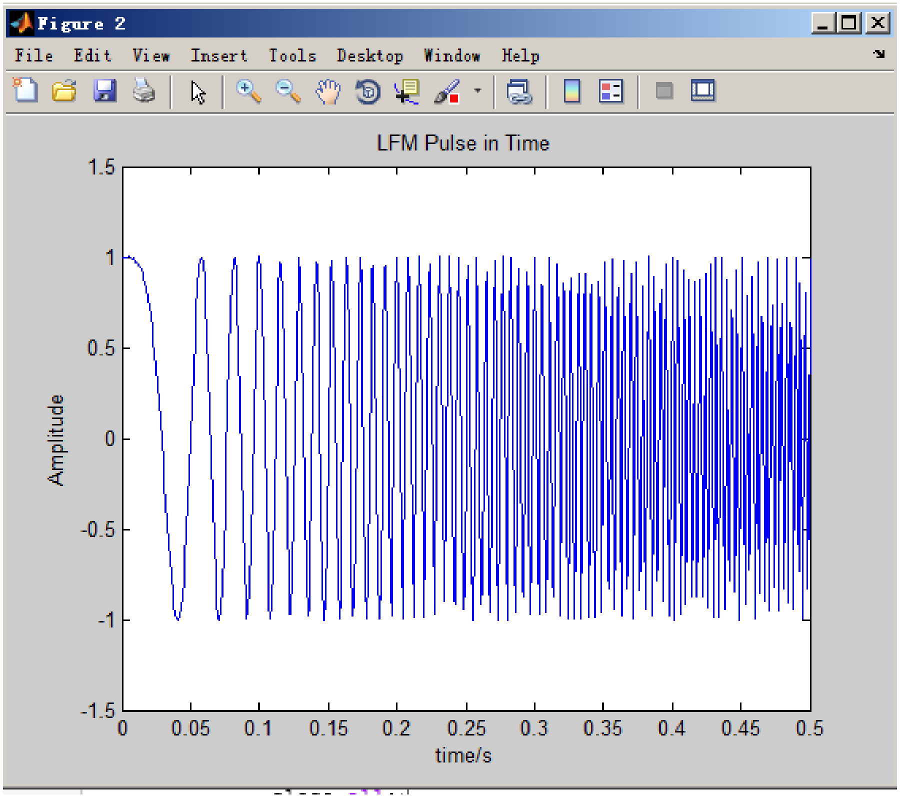
其带宽 B 的表达式如(1.2)

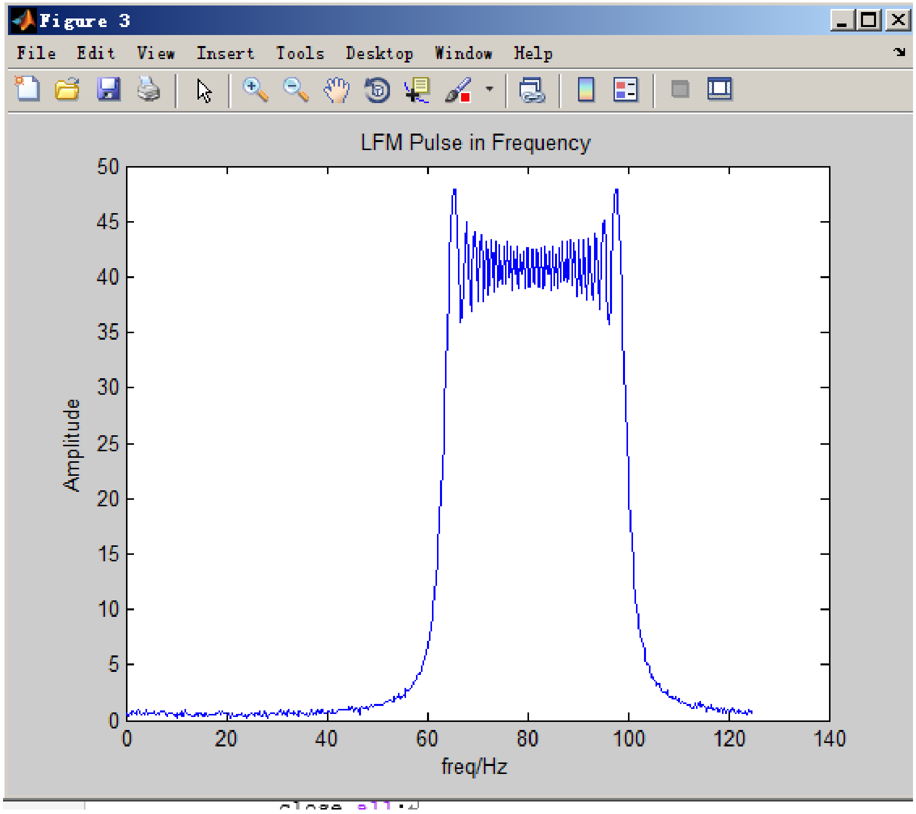
$$B = \frac{\mu\tau}{2\pi} \tag{1.2}$$

即，LFM 脉冲的时域表达式为：

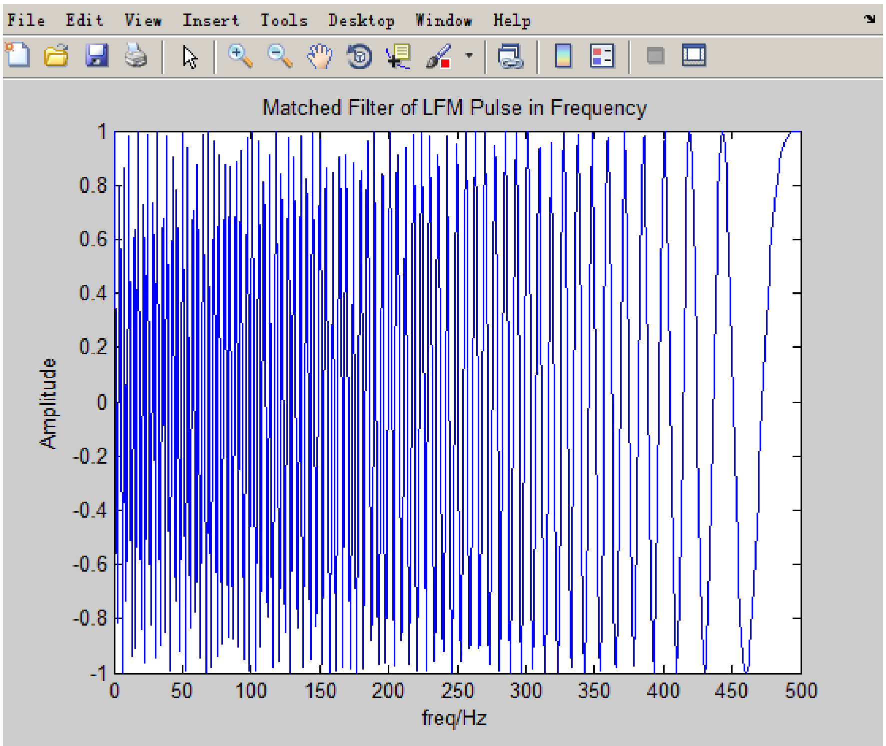
$$s(t) = A rect(\frac{t}{\tau})\cos(\omega_0 t + \frac{1}{2}\mu t^2) \tag{1.3}$$

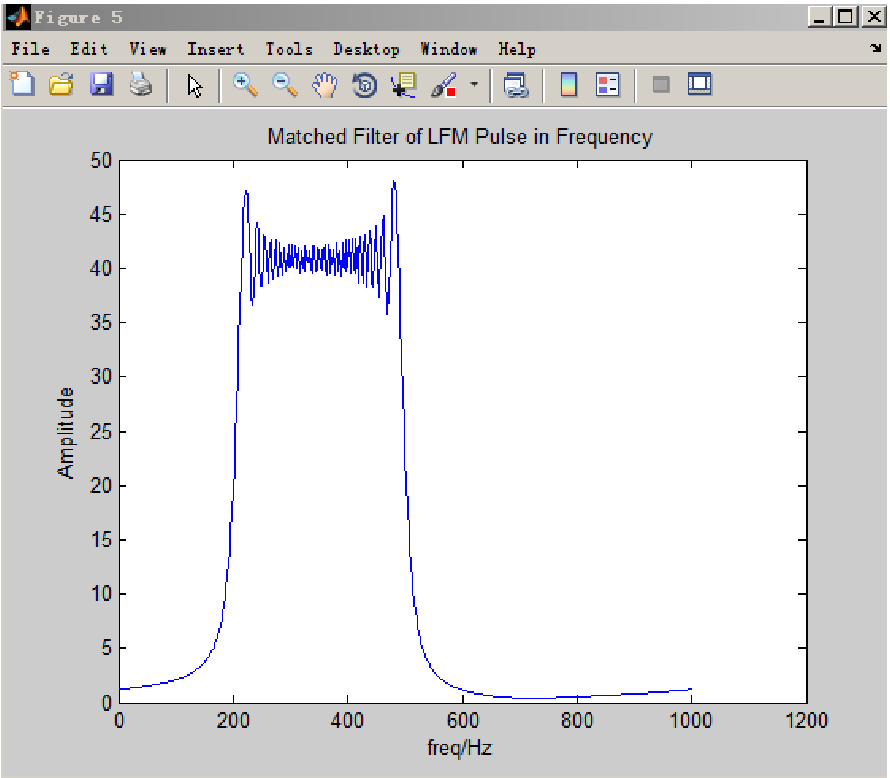
单个 LFM 脉冲的时域、频域图如下（SNR40dB）：



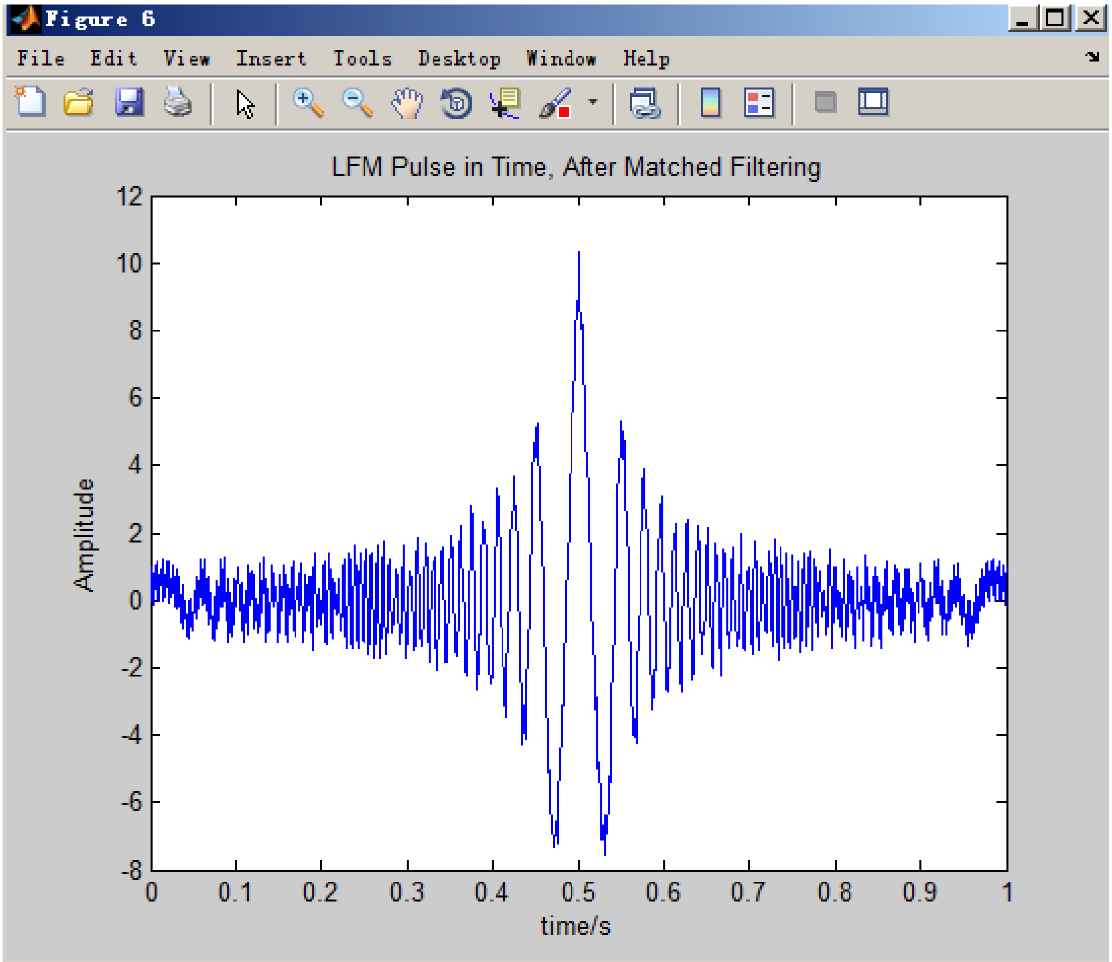


其对应的匹配滤波器时域、频域：

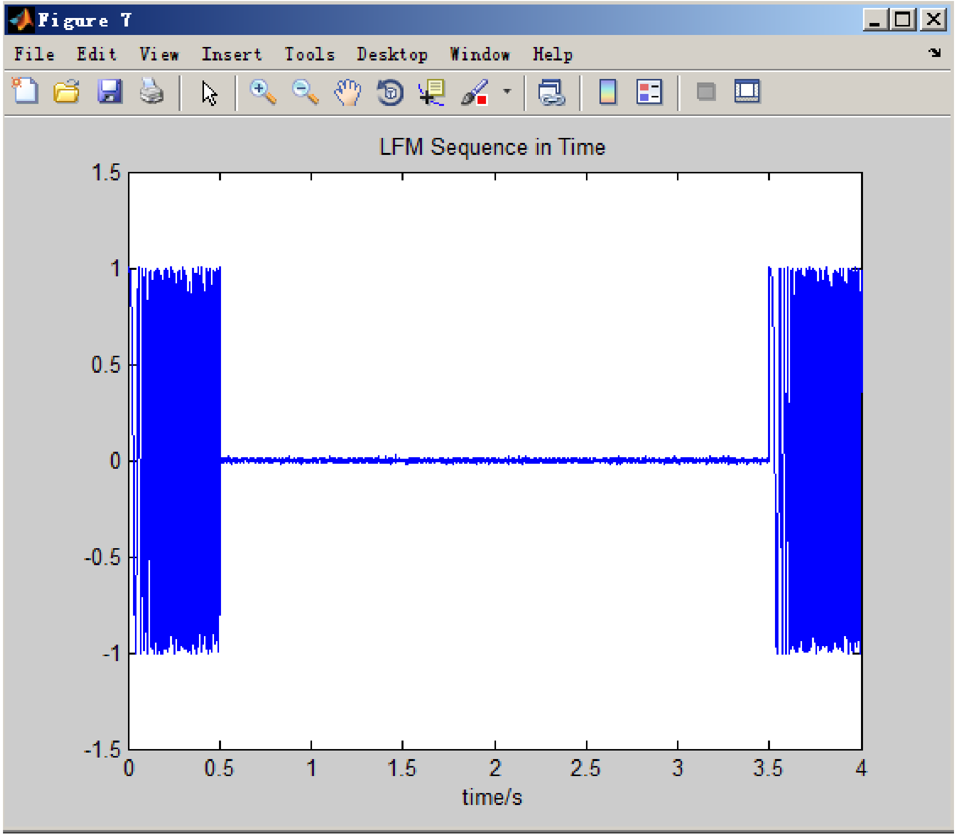




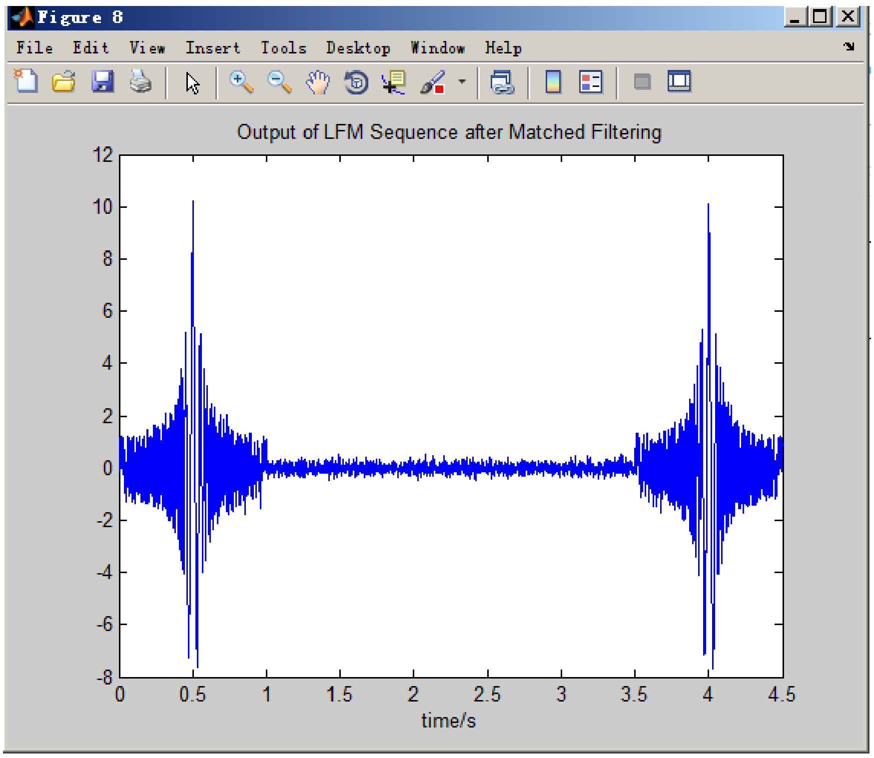
单独的一个脉冲通过匹配滤波器，得到波形：



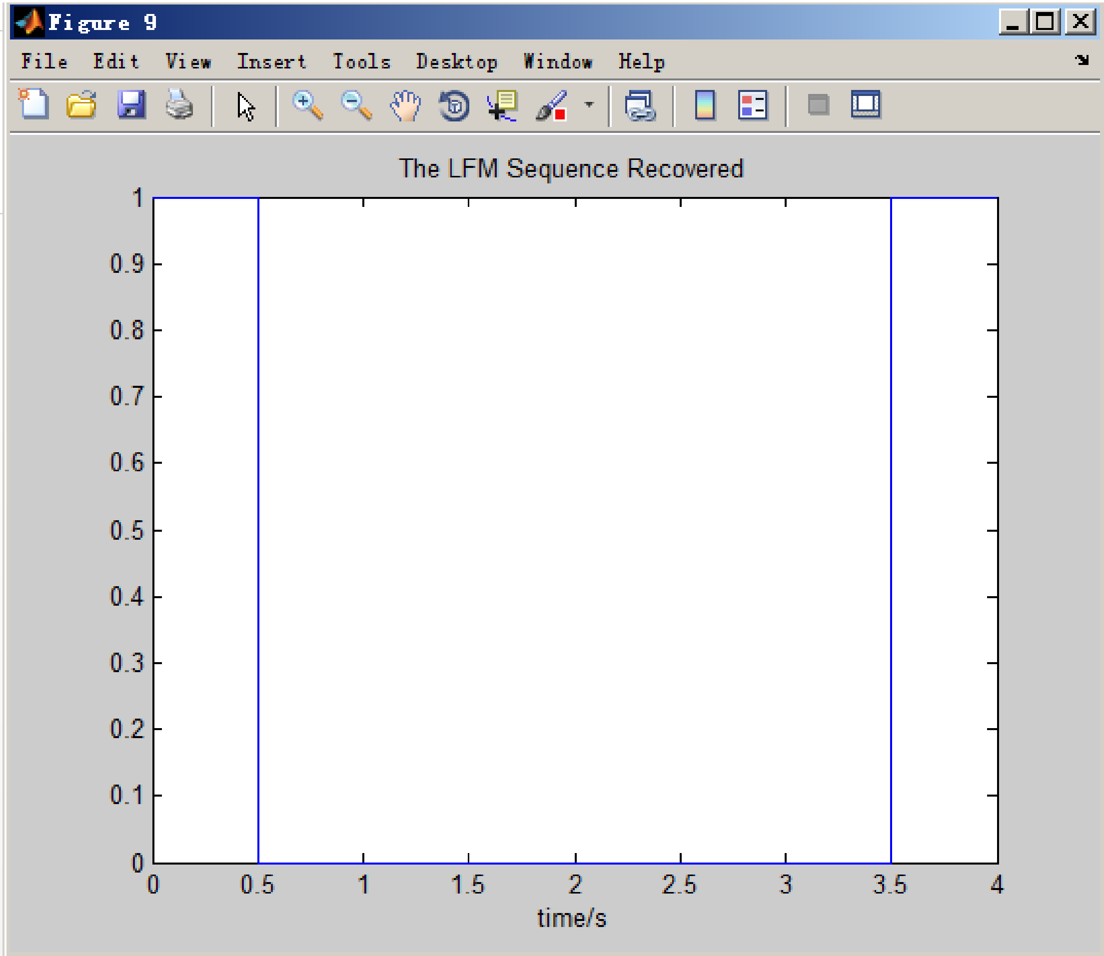
对 ASK 中的 8 位随机序列进行 LFM 调制，得:



将该信号通过匹配滤波器，得到匹配滤波后的波形如图，可观察到每个信号结束处明显的峰值，即 LFM 信号通过匹配滤波器，放大原信号的效果远好于普通的 ASK



对其进行定时脉冲抽样和保持，重建得到接收到的信号：



与原输入信号相同。

2、仿真结果讨论

I. Pe:

在 SNR=0 时，由于没有噪声干扰，n_0 始终=0，所以不论 ASK 或 LFM, 由(1.4)计算出来的 Pe 都等于 0，解调出的码型一定等于原码型。

$$P_e = \frac{1}{2} erfc(\sqrt{\frac{E(1-\rho)}{2n_0}}) \tag{1.4}$$

在有噪声干扰时，运行程序（取 N=1000，SNR=40dB），有：

ThresholdLFM =

7.3159

PeLFM =

     0

ThresholdASK =

   5.0619

PeASK =

    3.8721e-06

可见，同为对原信号进行幅度上的调制（1 为脉冲,0 为无信号）LFM 的绝对阈值较高（但归一化阈值取的是相同的）。由于初始设置的 SNR 较小（40dB），二者皆有很小的 Pe,而 LFM 更佳。调整初始设置的 SNR 至 20dB,得二者误码率

PeLFM =

2.0924e-23

PeASK =

7.8270e-04

依旧可见，LFM 的误码率优于 ASK-但随着输入的 SNR 减小，二者输出的误码率均有增大。说明噪声可以增大误码率，故在实际通信中应尽量提高解调器输入端的 SNR 以减少差错。

II.关于匹配滤波：

观察 ASK,FSK 的单脉冲波形，可见，匹配滤波器 $h(t)=K\times s(t_0-t)$ 可以实现使输入信号在 t_0 处达到最大值，且此最大值数倍于输入信号本身的幅度，换言之，匹配滤波可以在很大程度上改善接收机的信噪比。但它会使输入信号波形产生严重失真。

附：源程序代码

```
close all;

PulseDuration=0.5;% 0.5s per code element
fs=1000;%Sampling Frequency: 1000 points/s
fc=50;%Carrier Frequency. Due to Nyquist Theroem,fc<500Hz
SNR=40;%SNR in dB
numCodeElement=PulseDuration*fs;%Number of samples within a pulse
N=8;%Length of Random Sequence
K=1;%Coefficient of the matched filter

TimeDuration=PulseDuration*N;%Total Time Duration
nofPulseDuration=1:PulseDuration*fs;%n sequence of a pulse
nofTimeDuration=1:TimeDuration*fs;%n sequence of the whole plot

% Random Sequence Generation: a random sequence with the length of N
Gate=0.5;%threshold of the sequence. In this Sequence P(0)=Gate,
P(1)=1-Gate)
```

```
OriginalSignal=unifrnd(0,1,1,N);
Sequence=ones(1,N);
for n=1:N
    if OriginalSignal(n)>=Gate
        Sequence(n)=1;
    else
        Sequence(n)=0;
    end
end
SignalSource=rectpulse(Sequence,numCodeElement);%%Rectangular pulse
shaping, with numCodeElement(PulseDuration*fs) samples per pulse
plot(nofTimeDuration/fs,SignalSource);
title('Original Sequence');
xlabel('time/s');


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% Linear Frequency Modulation
a=1;%Amplitude of LFM Pulse
B=0.3;%Bandwidth of the LFM pulse
miu=B/(2*length(nofPulseDuration));
lfm=a*exp(j*(2*pi*fc*nofPulseDuration+2*pi*miu*nofPulseDuration.*nofP
ulseDuration));% LFM of a pulse
lfmwithNoise=awgn(lfm,SNR);%with White Gaussian Noise added
figure,plot(nofPulseDuration/fs,real(lfmwithNoise));%a Linear
Frequency Modulation(LFM) pulse in time
title('LFM Pulse in Time');
xlabel('time/s');ylabel('Amplitude');
LFM=fftshift(fft(lfmwithNoise));%LFM pulse in Frequency
figure,plot(nofPulseDuration/TimeDuration,abs(LFM));
title('LFM Pulse in Frequency');
xlabel('freq/Hz');ylabel('Amplitude');

% Matched filter of LFM
nTimeDelay=1;
nt0=nTimeDelay+length(lfm);
nofMatchedFilter=nt0-nofPulseDuration;
hLFM=K*lfm(nofMatchedFilter);%reversed part of the matched filter
hLFM=[zeros(1,nTimeDelay) hLFM];%matched filter with a beginning of a
delay of TimeDelay/fs
figure,plot(linspace(1,PulseDuration,length(hLFM)),real(hLFM));%match
ed filter in time
title('Matched Filter of LFM Pulse in Time');
xlabel('time/s');ylabel('Amplitude');
```

```
HLFM=fftshift(fft(hLFM));
figure,plot((1:length(HLFM))/PulseDuration,abs(HLFM));%matched filter
in frequency
% n transforming into w (or f?): frequency=n*(2pi/N)*(fs/2pi),
N=TimeDuration*fs
title('Matched Filter of LFM Pulse in Frequency');
xlabel('freq/Hz');ylabel('Amplitude');

%Signal output of one pulse, after matched filtering
yLFM=conv(hLFM,lfmwithNoise);
tofyLFM=1:length(yLFM);
figure,plot(tofyLFM/fs,real(yLFM));%Signal output in time
title('LFM Pulse in Time, After Matched Filtering');
xlabel('time/s');ylabel('Amplitude');

%Linear phase modulation of the random sequence given above
SignalmodulatedbyLFM=zeros(1,TimeDuration*fs);
    for n1=0:(N-1)
        for n2=1:(PulseDuration*fs)

SignalmodulatedbyLFM(PulseDuration*fs*n1+n2)=SignalSource(PulseDurati
on*fs*n1+n2)*lfm(n2);
        end
    end
SignalmodulatedbyLFM=awgn(SignalmodulatedbyLFM,SNR);
figure,plot(nofTimeDuration/fs,real(SignalmodulatedbyLFM));
title('LFM Sequence in Time');
xlabel('time/s');

%LFM modulated signal passing through its corresponding matched filter
ySignalmodulatedbyLFM=conv(SignalmodulatedbyLFM,hLFM);
nofySignalmodulatedbyLFM=1:length(ySignalmodulatedbyLFM);
figure,plot(nofySignalmodulatedbyLFM/fs,real(ySignalmodulatedbyLFM));
title('Output of LFM Sequence after Matched Filtering');
xlabel('time/s');

%Recovery
ThresholdLFM=(0.5*a+a/(2*SNR)*log((1-Gate)/Gate))*max(ySignalmodulate
dbyLFM)
tofRecovery=1:N;
LFMRecovery=zeros(1,N);
for Samples=1:N

LFMRecovery(Samples)=ySignalmodulatedbyLFM(PulseDuration*fs*Samples);
```

```
    if LFMRecovery(Samples)>ThresholdLFM
        LFMRecovery(Samples)=1;
    else
        LFMRecovery(Samples)=0;
    end
end
LFMRecovered=rectpulse(LFMRecovery,numCodeElement);
figure,plot(nofTimeDuration/fs,LFMRecovered)
title('The LFM Sequence Recovered')
xlabel('time/s')

sumoflfm=0;
for n=1:length(lfm)
    sumoflfm=sumoflfm+real(lfm(n))*real(lfm(n));
end
N0=sumoflfm/(10.^(SNR/10));
SNRLFM=2*0.5*a*a*PulseDuration*fs/N0;
PeLFM=0.5*erfc(sqrt(SNRLFM/4))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ASK
a=1;%Amplitude of ASK=1;
carrier=a*cos(2*pi*fc*nofPulseDuration/fs);% the carrier signal of ASK
carrierwithNoise=awgn(carrier,SNR);%with White Gaussian Noise added
figure,plot(nofPulseDuration/fs,real(carrierwithNoise));%an ASK pulse
in time
title('ASK Pulse in Time');
xlabel('time/s');
ASKPulse=fftshift(fft(carrierwithNoise));%LFM pulse in Frequency
figure,plot(nofPulseDuration/TimeDuration,abs(ASKPulse));
title('ASK Pulse in Frequency');
xlabel('freq/Hz');

% Matched filter
nTimeDelay=1;
nt0=nTimeDelay+length(carrier);
nofMatchedFilterASK=nt0-nofPulseDuration;
hASK=K*carrier(nofMatchedFilterASK);%reversed part of the matched filter
hASK=[zeros(1,nTimeDelay) hASK];%matched filter with a beginning of a
delay of TimeDelay/fs
figure,plot(linspace(1,PulseDuration,length(hASK)),real(hASK));%match
ed filter in time
title('Matched Filter of ASK Pulse in Time');
xlabel('time/s');
```

```
HASK=fftshift(fft(hASK));
figure,plot((1:length(HASK))/PulseDuration,abs(HASK));%matched filter
in frequency
% n transforming into w (or f?): frequency=n*(2pi/N)*(fs/2pi),
N=TimeDuration*fs
title('Matched Filter of ASK Pulse in Frequency');
xlabel('freq/Hz');ylabel('Amplitude');

%Signal output of one pulse, after matched filtering
yASK=conv(hASK,carrierwithNoise);
tofyASK=1:length(yASK);
figure,plot(tofyASK/fs,real(yASK));%Signal output in time
title('ASK Pulse in Time, After Matched Filtering');
xlabel('time/s');

%\
SignalModulatedbyASK=zeros(1,TimeDuration*fs);
    for n1=0:(N-1)
        for n2=1:(PulseDuration*fs)

SignalModulatedbyASK(PulseDuration*fs*n1+n2)=SignalSource(PulseDurati
on*fs*n1+n2)*carrier(n2);
        end
    end
SignalModulatedbyASK=awgn(SignalModulatedbyASK,SNR);
figure,plot(nofTimeDuration/fs,real(SignalModulatedbyASK));
title('ASK Sequence in Time');
xlabel('time/s');

SIGNALASK=fftshift(fft(SignalModulatedbyASK));
figure,plot(nofTimeDuration/TimeDuration,abs(SIGNALASK));
title('ASK Sequence in Frequency');
xlabel('Frequecy/Hz');

%ASK modulated signal passing through its corresponding matched filter
ySignalmodulatedbyASK=conv(SignalModulatedbyASK,hASK);
nofySignalmodulatedbyASK=1:length(ySignalmodulatedbyASK);
figure,plot(nofySignalmodulatedbyASK/fs,real(ySignalmodulatedbyASK));
title('Output of ASK Sequence after Matched Filtering');
xlabel('time/s');

%Recovery of ASK Signal
ThresholdASK=(0.5*a+a/(2*SNR)*log((1-Gate)/Gate))*max(ySignalmodulate
dbyASK)
```

```
tofRecovery=1:N;
ASKRecovery=zeros(1,N);
for Samples=1:N

ASKRecovery(Samples)=ySignalmodulatedbyASK(PulseDuration*fs*Samples);
    if ASKRecovery(Samples)>ThresholdASK
        ASKRecovery(Samples)=1;
    else
        ASKRecovery(Samples)=0;
    end
end
ASKRecovered=rectpulse(ASKRecovery,numCodeElement);
figure,plot(nofTimeDuration/fs,ASKRecovered)
title('The ASK Sequence Recovered')
xlabel('time/s')
PeASK=0.5*erfc(sqrt(SNR/4))
close all;

PulseDuration=0.5;% 0.5s per code element
fs=1000;%Sampling Frequency: 1000 points/s
fc=50;%Carrier Frequency. Due to Nyquist Theroem,fc<500Hz
SNR=40;%SNR in dB
numCodeElement=PulseDuration*fs;%Number of samples within a pulse
N=8;%Length of Random Sequence
K=1;%Coefficient of the matched filter

TimeDuration=PulseDuration*N;%Total Time Duration
nofPulseDuration=1:PulseDuration*fs;%n sequence of a pulse
nofTimeDuration=1:TimeDuration*fs;%n sequence of the whole plot

% Random Sequence Generation: a random sequence with the length of N
Gate=0.5;%threshold of the sequence. In this Sequence P(0)=Gate,
P(1)=1-Gate)
OriginalSignal=unifrnd(0,1,1,N);
Sequence=ones(1,N);
for n=1:N
    if OriginalSignal(n)>=Gate
        Sequence(n)=1;
    else
        Sequence(n)=0;
    end
end
SignalSource=rectpulse(Sequence,numCodeElement);%%Rectangular pulse
```

```
shaping, with numCodeElement(PulseDuration*fs) samples per pulse
plot(nofTimeDuration/fs,SignalSource);
title('Original Sequence');
xlabel('time/s');

%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% Linear Frequency Modulation
a=1;%Amplitude of LFM Pulse
B=0.3;%Bandwidth of the LFM pulse
miu=B/(2*length(nofPulseDuration));
lfm=a*exp(j*(2*pi*fc*nofPulseDuration+2*pi*miu*nofPulseDuration.*nofP
ulseDuration));% LFM of a pulse
lfmwithNoise=awgn(lfm,SNR);%with White Gaussian Noise added
figure,plot(nofPulseDuration/fs,real(lfmwithNoise));%a Linear
Frequency Modulation(LFM) pulse in time
title('LFM Pulse in Time');
xlabel('time/s');ylabel('Amplitude');
LFM=fftshift(fft(lfmwithNoise));%LFM pulse in Frequency
figure,plot(nofPulseDuration/TimeDuration,abs(LFM));
title('LFM Pulse in Frequency');
xlabel('freq/Hz');ylabel('Amplitude');

% Matched filter of LFM
nTimeDelay=1;
nt0=nTimeDelay+length(lfm);
nofMatchedFilter=1:length(lfm);
nofMatchedFilter=nt0-nofPulseDuration;
hLFM=K*lfm(nofMatchedFilter);%reversed part of the matched filter
hLFM=[zeros(1,nTimeDelay) hLFM];%matched filter with a beginning of a
delay of TimeDelay/fs
figure,plot(linspace(1,PulseDuration,length(hLFM)),real(fliplr((hLFM)
)));
title('Matched Filter of LFM Pulse in Time');
xlabel('time/s');ylabel('Amplitude');
HLFM=fftshift(fft(hLFM));
figure,plot((1:length(HLFM))/PulseDuration,abs(HLFM));%matched filter
in frequency
% n transforming into w (or f?): frequency=n*(2pi/N)*(fs/2pi),
N=TimeDuration*fs
title('Matched Filter of LFM Pulse in Frequency');
xlabel('freq/Hz');ylabel('Amplitude');

%Signal output of one pulse, after matched filtering
```

```
yLFM=conv(hLFM,lfmwithNoise);
tofyLFM=1:length(yLFM);
figure,plot(tofyLFM/fs,real(yLFM));%Signal output in time
title('LFM Pulse in Time, After Matched Filtering');
xlabel('time/s');ylabel('Amplitude');

%Linear phase modulation of the random sequence given above
SignalmodulatedbyLFM=zeros(1,TimeDuration*fs);
    for n1=0:(N-1)
        for n2=1:(PulseDuration*fs)

SignalmodulatedbyLFM(PulseDuration*fs*n1+n2)=SignalSource(PulseDurati
on*fs*n1+n2)*lfm(n2);
        end
    end
SignalmodulatedbyLFM=awgn(SignalmodulatedbyLFM,SNR);
figure,plot(nofTimeDuration/fs,real(SignalmodulatedbyLFM));
title('LFM Sequence in Time');
xlabel('time/s');

%LFM modulated signal passing through its corresponding matched filter
ySignalmodulatedbyLFM=conv(SignalmodulatedbyLFM,hLFM);
nofySignalmodulatedbyLFM=1:length(ySignalmodulatedbyLFM);
figure,plot(nofySignalmodulatedbyLFM/fs,real(ySignalmodulatedbyLFM));
title('Output of LFM Sequence after Matched Filtering');
xlabel('time/s');

%Recovery
ThresholdLFM=abs((0.5*a+a/(2*SNR)*log((1-Gate)/Gate))*max(ySignalmodu
latedbyLFM))
tofRecovery=1:N;
LFMRecovery=zeros(1,N);
for Samples=1:N

LFMRecovery(Samples)=ySignalmodulatedbyLFM(PulseDuration*fs*Samples);
    if LFMRecovery(Samples)>ThresholdLFM
        LFMRecovery(Samples)=1;
    else
        LFMRecovery(Samples)=0;
    end
end
LFMRecovered=rectpulse(LFMRecovery,numCodeElement);
figure,plot(nofTimeDuration/fs,LFMRecovered)
title('The LFM Sequence Recovered')
```

```matlab
xlabel('time/s')

sumoflfm=0;
for n=1:length(lfm)
    sumoflfm=sumoflfm+real(lfm(n))*real(lfm(n));
end
N0=sumoflfm/(10.^(SNR/10));
SNRLFM=2*0.5*a*a*PulseDuration*fs/N0;
PeLFM=0.5*erfc(sqrt(SNRLFM/4))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ASK

a=1;%Amplitude of ASK=1;
carrier=a*cos(2*pi*fc*nofPulseDuration/fs);% the carrier signal of ASK
carrierwithNoise=awgn(carrier,SNR);%with White Gaussian Noise added
figure,plot(nofPulseDuration/fs,real(carrierwithNoise));%an ASK pulse in time
title('ASK Pulse in Time');
xlabel('time/s');
ASKPulse=fftshift(fft(carrierwithNoise));%LFM pulse in Frequency
figure,plot(nofPulseDuration/TimeDuration,abs(ASKPulse));
title('ASK Pulse in Frequency');
xlabel('freq/Hz');

%%%Matched filter
nTimeDelay=1;
nt0=nTimeDelay+length(carrier);
nofMatchedFilterASK=nt0-nofPulseDuration;
hASK=K*carrier(nofMatchedFilterASK);%reversed part of the matched filter
hASK=[zeros(1,nTimeDelay) hASK];%matched filter with a beginning of a delay of TimeDelay/fs
figure,plot(linspace(1,PulseDuration,length(hASK)),real(hASK));%matched filter in time
title('Matched Filter of ASK Pulse in Time');
xlabel('time/s');
HASK=fftshift(fft(hASK));
figure,plot((1:length(HASK))/PulseDuration,abs(HASK));%matched filter in frequency
%n transforming into w (or f?): frequency=n*(2pi/N)*(fs/2pi), N=TimeDuration*fs
title('Matched Filter of ASK Pulse in Frequency');
xlabel('freq/Hz');ylabel('Amplitude');

%Signal output of one pulse, after matched filtering
```

```
yASK=conv(hASK,carrierwithNoise);
tofyASK=1:length(yASK);
figure,plot(tofyASK/fs,real(yASK));%Signal output in time
title('ASK Pulse in Time, After Matched Filtering');
xlabel('time/s');

SignalModulatedbyASK=zeros(1,TimeDuration*fs);
    for n1=0:(N-1)
        for n2=1:(PulseDuration*fs)

SignalModulatedbyASK(PulseDuration*fs*n1+n2)=SignalSource(PulseDurati
on*fs*n1+n2)*carrier(n2);
        end
    end
SignalModulatedbyASK=awgn(SignalModulatedbyASK,SNR);
figure,plot(nofTimeDuration/fs,real(SignalModulatedbyASK));
title('ASK Sequence in Time');
xlabel('time/s');

SIGNALASK=fftshift(fft(SignalModulatedbyASK));
figure,plot(nofTimeDuration/TimeDuration,abs(SIGNALASK));
title('ASK Sequence in Frequency');
xlabel('Frequecy/Hz');

%%%ASK modulated signal passing through its corresponding matched filter
ySignalmodulatedbyASK=conv(SignalModulatedbyASK,hASK);
nofySignalmodulatedbyASK=1:length(ySignalmodulatedbyASK);
figure,plot(nofySignalmodulatedbyASK/fs,real(ySignalmodulatedbyASK));
title('Output of ASK Sequence after Matched Filtering');
xlabel('time/s');

%%%Recovery of ASK Signal
ThresholdASK=(0.5*a+a/(2*SNR)*log((1-Gate)/Gate))*max(ySignalmodulate
dbyASK)
tofRecovery=1:N;
ASKRecovery=zeros(1,N);
for Samples=1:N

ASKRecovery(Samples)=ySignalmodulatedbyASK(PulseDuration*fs*Samples);
    if ASKRecovery(Samples)>ThresholdASK
        ASKRecovery(Samples)=1;
    else
        ASKRecovery(Samples)=0;
    end
```

```
end
ASKRecovered=rectpulse(ASKRecovery,numCodeElement);
figure,plot(nofTimeDuration/fs,ASKRecovered)
title('The ASK Sequence Recovered')
xlabel('time/s')
%
% sumofask=0;
% for n=1:length(lfm)
%     sumofask=sumofask+real(ask(n))*real(ask(n));
% end
% N0=sumofask/(10.^(SNR/10));
PeASK=0.5*erfc(sqrt(SNR/4))
```