哈尔滨工业大学（深圳）

通信系统仿真

实验报告

题　　目	无线信道建模仿真方法
专　　业	通信工程
学　　号	190210303
姓　　名	陈兴基
日　　期	2022.5.11

通信系统仿真实验任务书

姓　　名：陈兴基　　　　院　（系）：电子与信息工程学院

专　　业：通信工程　　　　班　　号：1902103

任务起至日期：　2022 年 5 月 11 日至 2022 年 6 月 1 日

实验题目：无线信道建模仿真方法

实验目的：通过对无线信道建模仿真的学习和仿真实验，掌握时变系统抽头延迟线模型；掌握无线信道冲激响应，多普勒滤波器等模型的建模方法，掌握多径信道建模和仿真方法。

实验内容：

实验 3-1： $\tilde{x}(t)=\exp[j2\pi(128)t]+\exp[j2\pi(512)t]$

（1）用上式双音频输入重新仿真 Matlab 例 3；

（2）用上式双音频输入重新仿真 Matlab 例 4；

例 4 中的系统是频率选择性的。通过仿真说明时变系统对两个输入音频的影响不同，从而表明系统的频率选择性。

实验 3-2：扩展例题 5 给出的仿真到 6 路模型，在如下的功率曲线下运行 BER 仿真（假设为平坦衰落）

（1）6 路的功率平均

（2）6 个抽头具有指数下降的功率曲线，其中最后一个抽头比第一个抽头低 10dB。

实验 3-3：对下面三种信道情况：

情况一：有积分—清除接收机的 AWGN 信道；

情况二：瑞利衰落信道加 AWGN 信道，无时延；

情况三：例题 7 中由于多普勒原因引起的衰落信道；

对比三种信道情况下 QPSK 系统，进行 BER 对 Eb/N0 的性能比较。

实验 3-1

1 基本原理

仿真的系统是一个简单系统、其中系统的带通输入是一个形式为 $x(t)=\cos[2\pi(f_0+f_1)t]$ 的“音频”，它对应于一个低通等效信号：

$$\tilde{x}(t)=\exp(j2\pi f_1 t)$$

假设该系统的低通等效冲激响应具有如下形式：

$$\tilde{h}(\tau,t)=\tilde{a}(t)\delta(t-\tau_0)$$

它是一个全通信道，具有延迟 τ_0 和复时变衰减 $\tilde{a}(t)$，这个衰减建模为一个零均值高斯随机过程，其功率谱密度为：

$$S_{\tilde{a}\tilde{a}}(f)=\frac{1}{(j2\pi f)^2+B^2}$$

式中 B 是多普勒带宽。可以证明该时变系统输出的复包络为：

$$\tilde{y}(t)=\tilde{a}(t)\tilde{x}(t-\tau_0)$$

该模型的仿真包括产生输入音频采样值、以及与一个滤波后的复高斯过程相乘，其中的滤波器的传递函数被选定为可产生如下式的功率谱密度：

$$S_{\tilde{a}\tilde{a}}(f)=\frac{1}{(j2\pi f)^2+B^2}$$

所以要求的滤波器具有如下的传递函数，是一个一阶低通滤波器：

$$H(s)=\frac{1}{s+B}$$

其他参数：$f_1=512Hz$，$B=64Hz$，$\tau_0=0$，$f_s=8192$ 采样/秒，仿真长度$=8192$ 个采样。

修改系统模型，使复低通等效冲激响应具有如下形式：

$$\tilde{h}(\tau,t)=\tilde{a}_1(t)\delta(t-\tau_1)+\tilde{a}_2(t)\delta(t-\tau_2)$$

通过考虑如下时不变形式，可证明该模型是频率选择性的：

$$\tilde{h}(\tau,t)=\tilde{a}_1\delta(t-\tau_1)+\tilde{a}_2\delta(t-\tau_2)$$

设$\tau_1=0$，可导出它的传递函数由下式给出：

$$H(f)=\tilde{a}_1+\tilde{a}_2\exp(-j2\pi f\tau_2)$$

上面给出的传递函数在不同频率处有不同的值，因此是频率选择性的。

时变系统的复低通等效输出为：

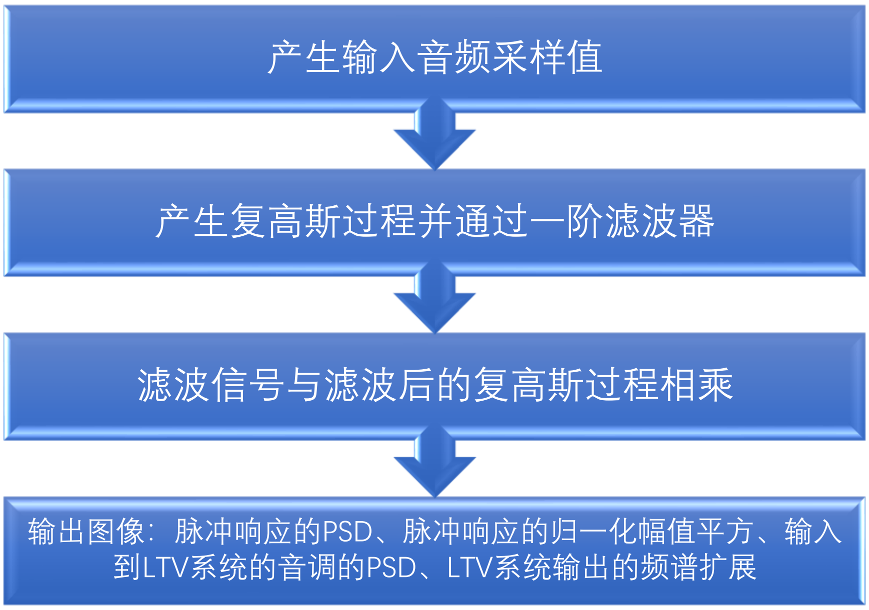
$$\tilde{y}(t)=\tilde{a}_1(t)\tilde{x}(t-\tau_1)+\tilde{a}_2(t-\tau_2)\tilde{x}(\tau-\tau_2)$$

如果系统的输入是一个 BPSK 信号，则系统的时变特性使 BPSK 信号作为时间的函数，发生衰减和相位旋转。由输出的两个延迟分量造成的信道频率选择性，表现为符号间干扰的形式。

BPSK 符号率$=512b/s$， 采样率$=16$个采样/秒。

$\tilde{a}_1(t)$和$\tilde{a}_2(t)$的多普勒带宽$=16Hz$， $\tau_1=0$， $\tau_2=8$个采样（比特持续时间的一半）。

2 仿真实验设计



3 仿真实验实现

（1）用双音频输入重新仿真 Matlab 例 3

- 函数：log_psd

```
function [logpsd,freq,ptotal,pmax] = log_psd(x,n,ts)
% Software given here is to accompany the textbook: W.H. Tranter,
% K.S. Shanmugan, T.S. Rappaport, and K.S. Kosbar, Principles of
% Communication Systems Simulation with Wireless Applications,
% Prentice Hall PTR, 2004.
%
% This function takes the n time domain samples (real or complex)
% and finds the psd by taking (fft/n)^2. The two sided spectrum is
% produced by shifting the psd; The array freq provides the
appropriate
% frequency values for plotting purposes.
% By taking 10*log10(psd/max(psd)) the psd is normalized; values
beow 60db
% are set equal to -60db
%
% n must be an even number, preferably a power of  2
%
y = zeros(1,n);          % initialize y vector
%
h = waitbar(0,'For Loop in PSD Calculation');
for k=1:n
    freq(k) =(k-1-(n/2))/(n*ts);
   y(k) = x(k)*((-1.0)^k);
   waitbar(k/n)
end;
%
v = fft(y)/n;
psd = abs(v).^2;
pmax = max(psd);
ptotal = sum(psd);
logpsd = 10*log10(psd/pmax);
%
% Truncate negative values at -60 dB
%
for k =1:n
```

```
    if(logpsd(k)<-60.0)
      logpsd(k) = -60.0;
    end
end
close(h)
% End of function file.
```

- 主程序

```
% File: c13_tiv1.m
% Software given here is to accompany the textbook: W.H. Tranter,
% K.S. Shanmugan, T.S. Rappaport, and K.S. Kosbar, Principles of
% Communication Systems Simulation with Wireless Applications,
% Prentice Hall PTR, 2004.
%
f1 = 128;                   % default signal frequency
f2 = 512;
bdoppler = 64;              % default doppler sampling
fs = 8192;                  % default sampling frequency
tduration = 1;              % default duration
%
ts = 1.0/fs;                % sampling period
n = tduration*fs;           % number of samples
t = ts*(0:n-1);             % time vector
x = exp(i*2*pi*f1*t)+exp(i*2*pi*f2*t);      % complex signal
zz1 = zeros(1,n);
zz2 = zeros(1,n);
%
% Generate Uncorrelated seq of Complex Gaussian Samples
%
z1 = randn(1,n)+ i*randn(1,n);
z2 = randn(1,n)+ i*randn(1,n);
%
% Filter the uncorrelated samples to generate correlated samples
%
coefft = exp(-bdoppler*2*pi*ts);
h = waitbar(0,'Filtering Loop in Progress');
for k=2:n
   zz1(k) = (ts*z1(k))+coefft*zz1(k-1);
   zz2(k) = (ts*z2(k))+coefft*zz2(k-1);
```

```
   waitbar(k/n)
end
close(h)
y1 = x.*zz1;            % filtered output of LTV system
y2 = x.*zz2;
%
% Plot the results in time domain and frequency domain
%
[psdzz,freq] = log_psd(zz1,n,ts);

figure;
plot(freq,psdzz); grid;
ylabel('Impulse Response in dB')
xlabel('Frequency')
title('PSD of the Impulse Response');

zzz = abs(zz1.^2)/(mean(abs(zz1.^2)));
figure;
plot(10*log10(zzz)); grid;
ylabel('Sq. Mag. of h(t) in dB')
xlabel('Time Sample Index')
title('Normalized Magnitude Square of the Impulse Response in dB');

[psdx1,freq] = log_psd(x,n,ts);
figure;
plot(freq,psdx1); grid;
ylabel('PSD of Tone Input in dB')
xlabel('Frequency')
title('PSD of Tone Input to the LTV System');

[psdy1,freq] = log_psd(y1,n,ts);
figure;
plot(freq,psdy1); grid;
ylabel('PSD of Output in dB')
xlabel('Frequency')
title('Spread Output of the LTV System');
% End of script file.
```

（2）用双音频输入重新仿真 Matlab 例 4

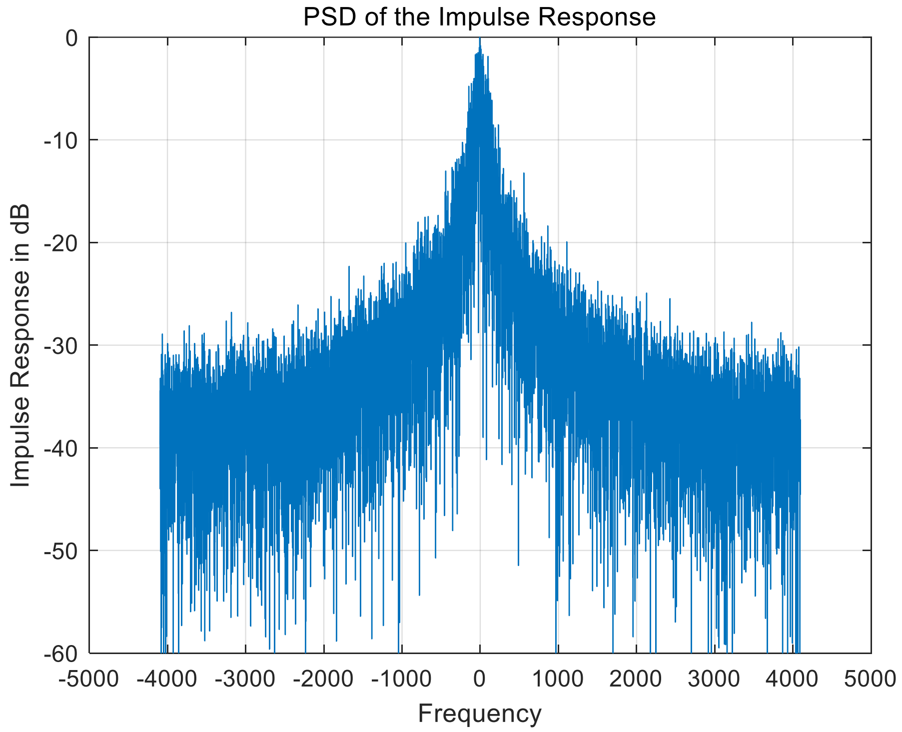
- 主程序

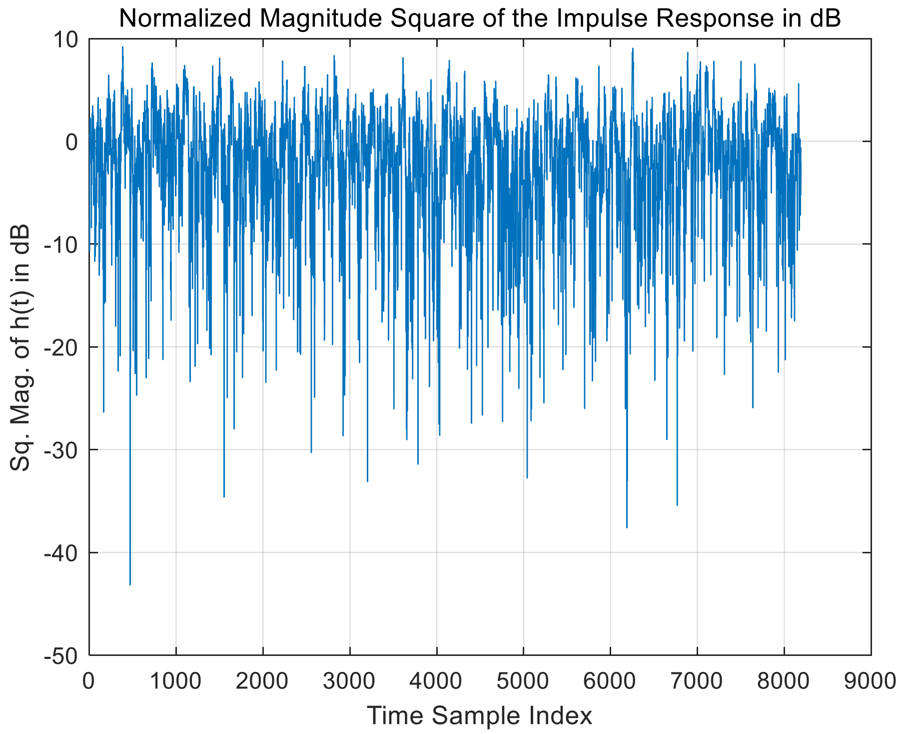
```
% File: c13_tiv2.m
% Software given here is to accompany the textbook: W.H. Tranter,
% K.S. Shanmugan, T.S. Rappaport, and K.S. Kosbar, Principles of
% Communication Systems Simulation with Wireless Applications,
% Prentice Hall PTR, 2004.
%
% Set default parameters
symrate = 512;
nsamples = 16;
nsymbols = 128;
bdoppler = 16;
ndelay = 8;
n = nsymbols*nsamples;
ts = 1.0/(symrate*nsamples);
%
% Generate two uncorrelated seq of Complex Gaussian Samples
%
z1 = randn(1,n)+i*randn(1,n);
z2 = randn(1,n)+i*randn(1,n);
%
% Filter the two uncorrelated samples to generate correlated
sequences
%
coefft = exp(-bdoppler*2*pi*ts);
zz1 = zeros(1,n);
zz2 = zeros(1,n);
for k = 2:n
   zz1(k) = z1(k)+coefft*zz1(k-1);
   zz2(k) = z2(k)+coefft*zz2(k-1);
end
%
% Generate a BPSK (random binry wavefrom and compute the output)
%
M = 2;                                % binary case
%x1 = mpsk_pulses(M,nsymbols,nsamples);
t=0:ts:0.25*(1-ts);                   %持续时间1s
x1=exp(1i*2*pi*128*t)+exp(1i*2*pi*512*t);
y1 = x1.*zz1;                         % first output component
```

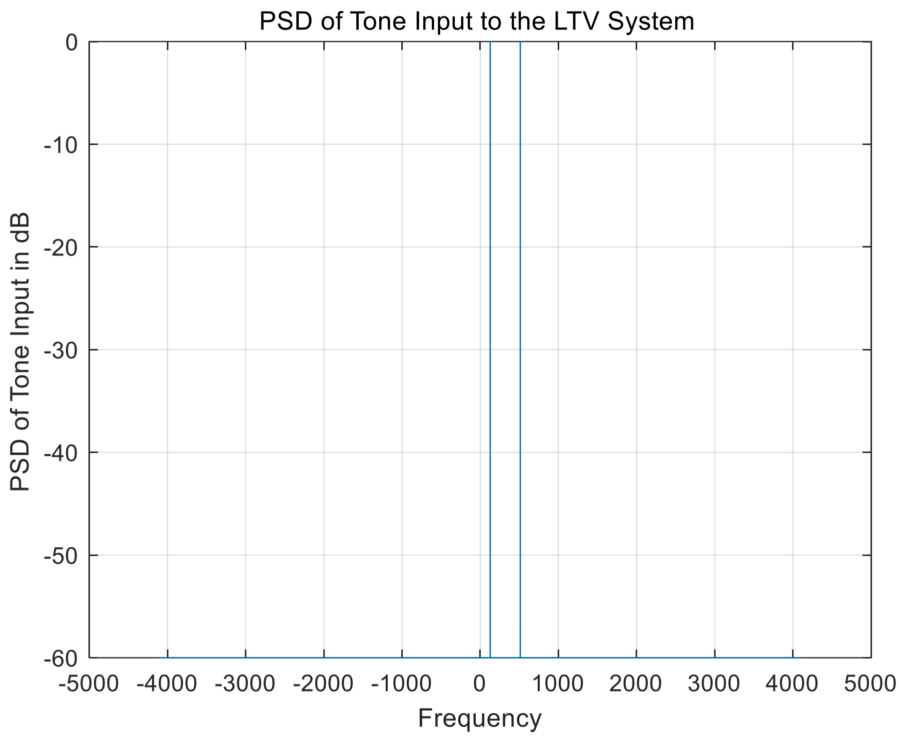
```
y2 = x1.*zz2;                        % second output component
y(1:ndelay) = y1(1:ndelay);
y(ndelay+1:n) = y1(ndelay+1:n)+y2(1:n-ndelay);
%
% Plot the results
%
[psdx1,freq] = log_psd(x1,n,ts);
[psdy,freq] = log_psd(y,n,ts);
[psdzz1,freq] = log_psd(zz1,n,ts);
[psdzz2,freq] = log_psd(zz2,n,ts);

figure;
plot(freq,psdx1);
grid;
title('PSD of the First Component Impulse Response');

figure;
plot(freq,psdy); grid;
title('Spread Output of the LTV System');

nn = 0:499;
figure;
subplot(2,1,1)
plot(nn,imag(x1(1:500)),'r',nn,real(y1(1:500)),'b');
grid;
title('Input and the First Component of the Output');
xlabel('Sample Index')
ylabel('Signal Level')

subplot(2,1,2)
plot(nn,imag(x1(1:500)),'r',nn,real(y(1:500)),'b');
grid;
title('Input and the Total Output')
xlabel('Sample Index')
ylabel('Signal Level')
% End of script file.
```

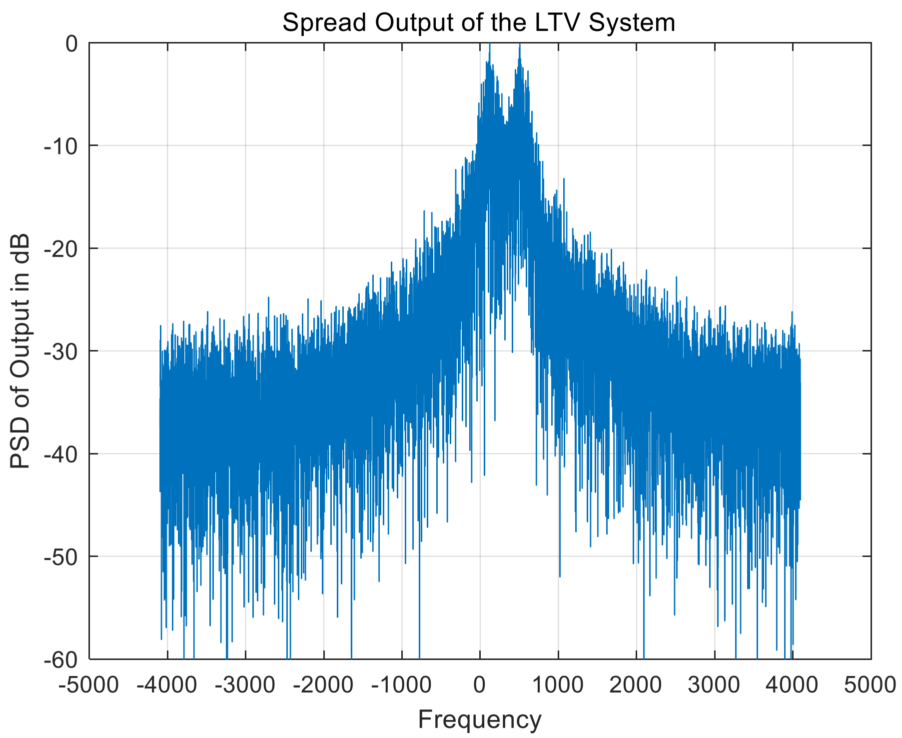
4、仿真实验结论

（1）用双音频输入重新仿真 Matlab 例 3

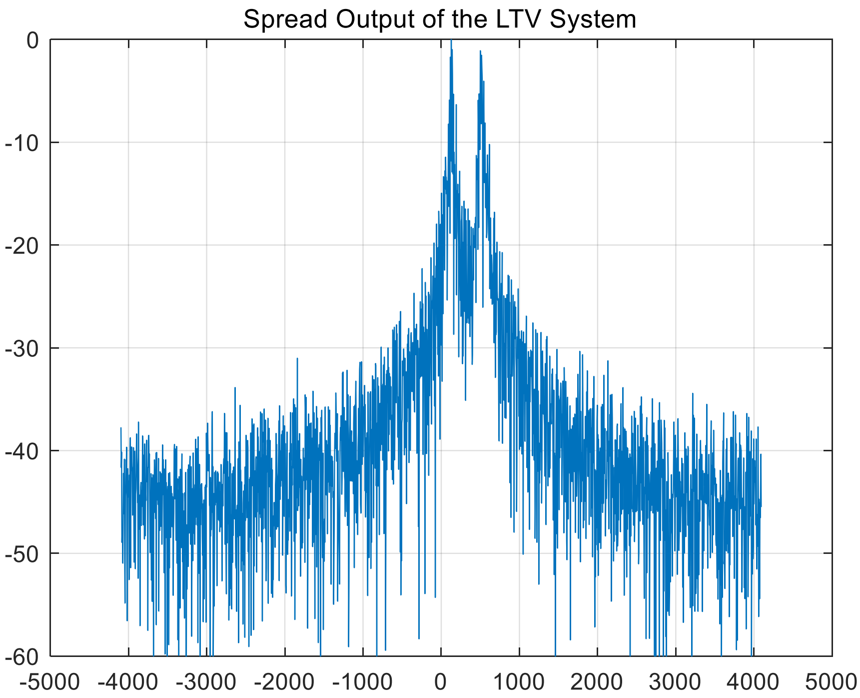


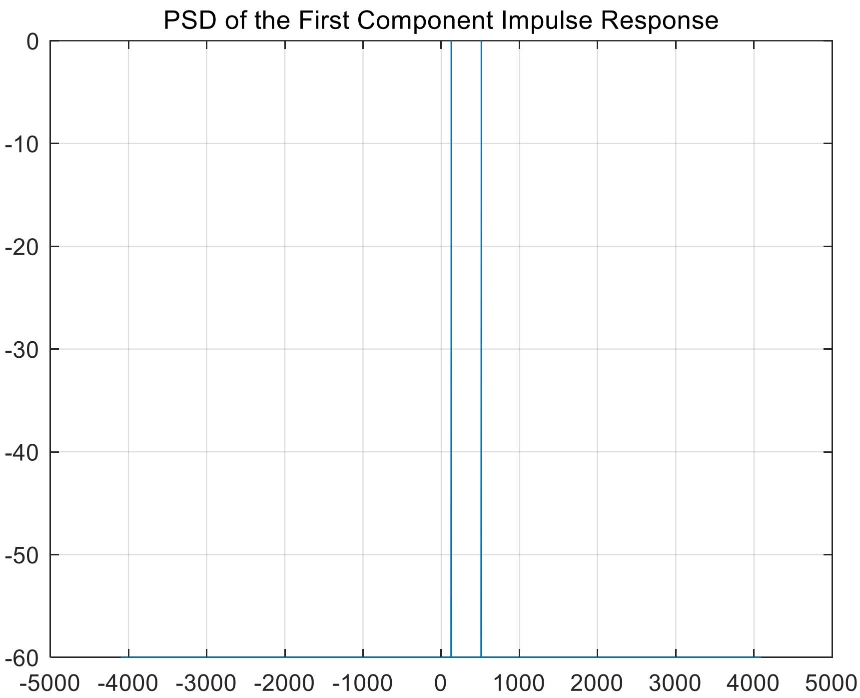


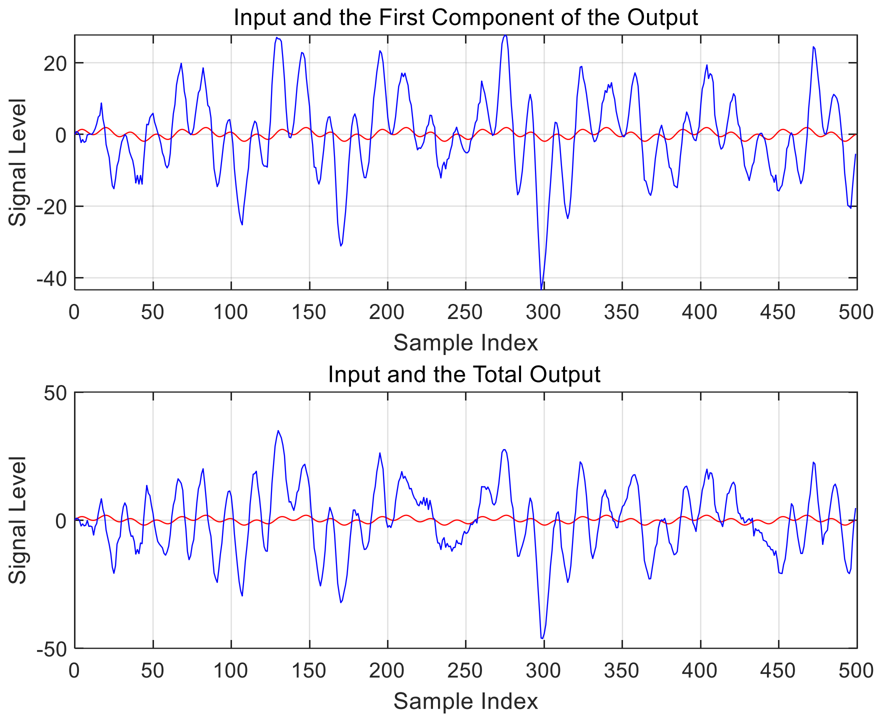

PSD of Tone Input to the LTV System
PSD of Tone Input in dB
0
-10
-20
-30
-40
-50
-60
-5000 -4000 -3000 -2000 -1000 0 1000 2000 3000 4000 5000
Frequency


Spread Output of the LTV System
PSD of Output in dB
0
-10
-20
-30
-40
-50
-60
-5000 -4000 -3000 -2000 -1000 0 1000 2000 3000 4000 5000
Frequency

（2）用双音频输入重新仿真 Matlab 例 4







Input and the First Component of the Output
Signal Level
20
0
-20
-40
0
50
100
150
200
250
300
350
400
450
500
Sample Index
Input and the Total Output
Signal Level
50
0
-50
0
50
100
150
200
250
300
350
400
450
500
Sample Index

实验 3-2

1 基本原理

为了简化仿真模型，作如下假设:

1. 信道中有三条路径，包括一条是没有衰落的 LOS 路径和两条具有瑞利分量的路径。与各路径相应的接收功率级以及路径间的延迟差是仿真的参数。

2. 信道的瑞利衰落仅影响发送信号的幅度，不影响瞬时相位。

3. 在符号间隔内各多径分量的衰减幅度是常数，并与相邻间隔无关(不需要多普勒频谱成形)。

4. 没有使用发射机滤波，接收机模型也是理想的积分-清除接收机。

接收信号可以写为

$$\tilde{y}(t)=\underbrace{a_0\tilde{x}(t)}_{\text{LOS}}+\underbrace{a_1R_1\tilde{x}(t)}_{\text{Rayleigh}}+\underbrace{a_2R_2\tilde{x}(t-\tau)}_{\text{Delayed Rayleigh}}$$

其中 R_1 和 R_2 是表示两条路径瑞利衰落的两个独立瑞利随机变量，τ 是两个瑞利分量的相对延迟。

对上式作傅里叶变换为：

$$\tilde{Y}(f)=a_0\tilde{X}(f)+a_1R_1\tilde{X}(f)+a_2R_2\tilde{X}(f)\exp(-j2\pi f\tau)$$

由此可以推导出信道传递函数为

$$\tilde{H}(f)=a_0+a_1R_1+a_2R_2\exp(-j2\pi f\tau)$$

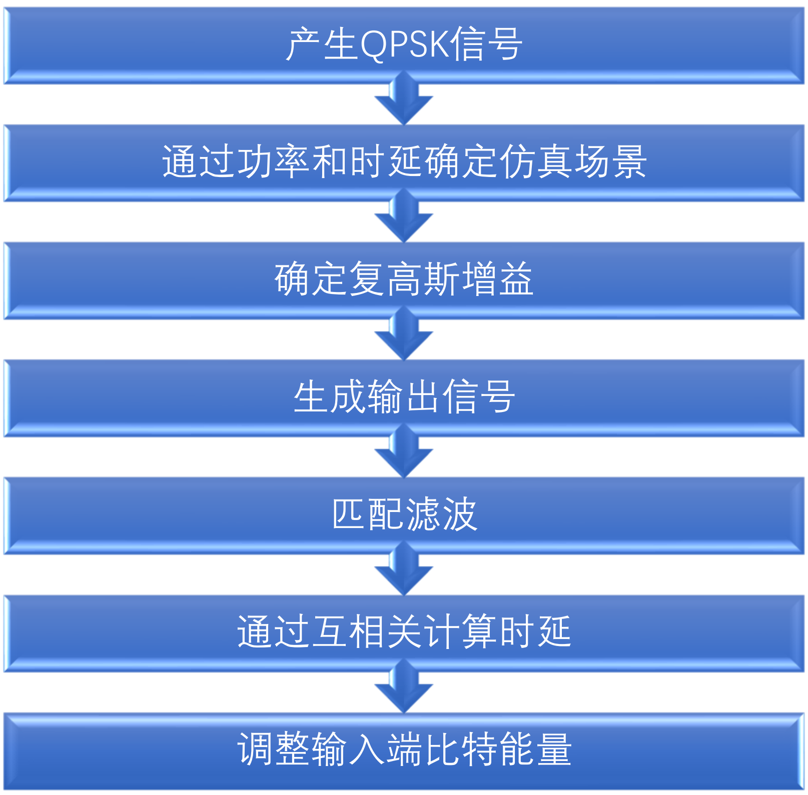
上式清楚的说明如果在信号所占频段内 $f\tau$ 的乘积不能忽略不计，则信道是频率选择性的，而这又将导致时延扩展和 ISI。

a_0、a_1 和 a_2 的值决定了三个多径分量的相对功率级 P_0、P_1 和 P_2。

Scenario	P_0	P_1	P_2	τ (samples)	Comments
1	1.0	0	0	0	Validation
2	1.0	0.2	0	0	Ricean flat fading
3	1.0	0	0.2	0	Ricean flat fading
4	1.0	0	0.2	8	Ricean frequency selective fading
5	0	1.0	0.2	0	Rayleigh flat fading
6	0	1.0	0.2	8	Rayleigh frequency selective fading

上表中给出的六组参数值，对于每种情形，都使用半解析的估计方法来估算BER。在表中，延迟是用采样周期来表示的。由于仿真的采样频率是每个符号 16 样点，因此，$\tau = 8$ 对应半个采样周期的延迟。

2 仿真实验设计



3 仿真实验实现

(1) 6 路的功率平均

```
clear all
clc

NN = 256;                                        % number of symbols
tb = 0.5;                                         % bit time
fs = 16;                                          % samples/symbol
ebn0db = 0:1:14;                                  % Eb/N0 vector
%
% Establish QPSK signals
%
x = random_binary(NN,fs)+1i*random_binary(NN,fs);   % QPSK signal
%
% Input powers and delays
%
%功率平均
p0 = 1/6; p1 = 1/6; p2 = 1/6;
p3 = 1/6; p4 = 1/6; p5 = 1/6;
%delay = input('Enter tau > ');%平坦衰落，delay全为0
delay0 = 0; delay1 = 0; delay2 = 0;
delay3 = 0; delay4 = 0; delay5 = 0;
%
% Set up the Complex Gaussian (Rayleigh) gains
%
gain1 = sqrt(p1)*abs(randn(1,NN) + 1i*randn(1,NN));
gain2 = sqrt(p2)*abs(randn(1,NN) + 1i*randn(1,NN));
gain3 = sqrt(p3)*abs(randn(1,NN) + 1i*randn(1,NN));
gain4 = sqrt(p4)*abs(randn(1,NN) + 1i*randn(1,NN));
gain5 = sqrt(p5)*abs(randn(1,NN) + 1i*randn(1,NN));
for k = 1:NN
   for kk=1:fs
      index=(k-1)*fs+kk;
      ggain1(1,index)=gain1(1,k);
      ggain2(1,index)=gain2(1,k);
      ggain3(1,index)=gain3(1,k);
      ggain4(1,index)=gain4(1,k);
      ggain5(1,index)=gain5(1,k);
   end
```

```
end
y1 = x;
for k=1:delay2
   y2(1,k) = y1(1,k)*sqrt(p0);
end
for k=(delay2+1):(NN*fs)
   y2(1,k)= y1(1,k)*sqrt(p0) + ...
             y1(1,k-delay1)*ggain1(1,k)+...
             y1(1,k-delay2)*ggain2(1,k)+...
             y1(1,k-delay3)*ggain3(1,k)+...
             y1(1,k-delay4)*ggain4(1,k)+...
             y1(1,k-delay5)*ggain5(1,k);
end
%
% Matched filter
%
b = ones(1,fs); b = b/fs; a = 1;
y = filter(b,a,y2);
%
% End of simulation
%
% Use the semianalytic BER estimator. The following sets
% up the semi analytic estimator. Find the maximum magnitude
% of the cross correlation and the corresponding lag.
%
[cor lags] = vxcorr(x,y);
[cmax nmax] = max(abs(cor));
timelag = lags(nmax);
theta = angle(cor(nmax));
y = y*exp(-1i*theta);                                  % derotate
%
% Noise BW calibration
%
hh = impz(b,a); ts = 1/16; nbw = (fs/2)*sum(hh.^2);
%
% Delay the input, and do BER estimation on the last 128 bits.
% Use middle sample. Make sure the index does not exceed number
% of input points. Eb should be computed at the receiver input.
%
index = (10*fs+8:fs:(NN-10)*fs+8);
xx = x(index);
```

```
yy = y(index-timelag+1);
[n1 n2] = size(y2); ny2=n1*n2;
eb = tb*sum(sum(abs(y2).^2))/ny2;
eb = eb/2;
[peideal,pesystem] = qpsk_berest(xx,yy,ebn0db,eb,tb,nbw);
figure(1)
semilogy(ebn0db,peideal,'b*-',ebn0db,pesystem,'r+-')
xlabel('E_b/N_0 (dB)'); ylabel('Probability of Error'); grid
axis([0 14 10^(-10) 1])
legend('Theoretical BER','BER of uniform power over the 6 rays')
```

(2) 6 个抽头具有指数下降的功率曲线，其中最后一个抽头比第一个抽头低 10dB。

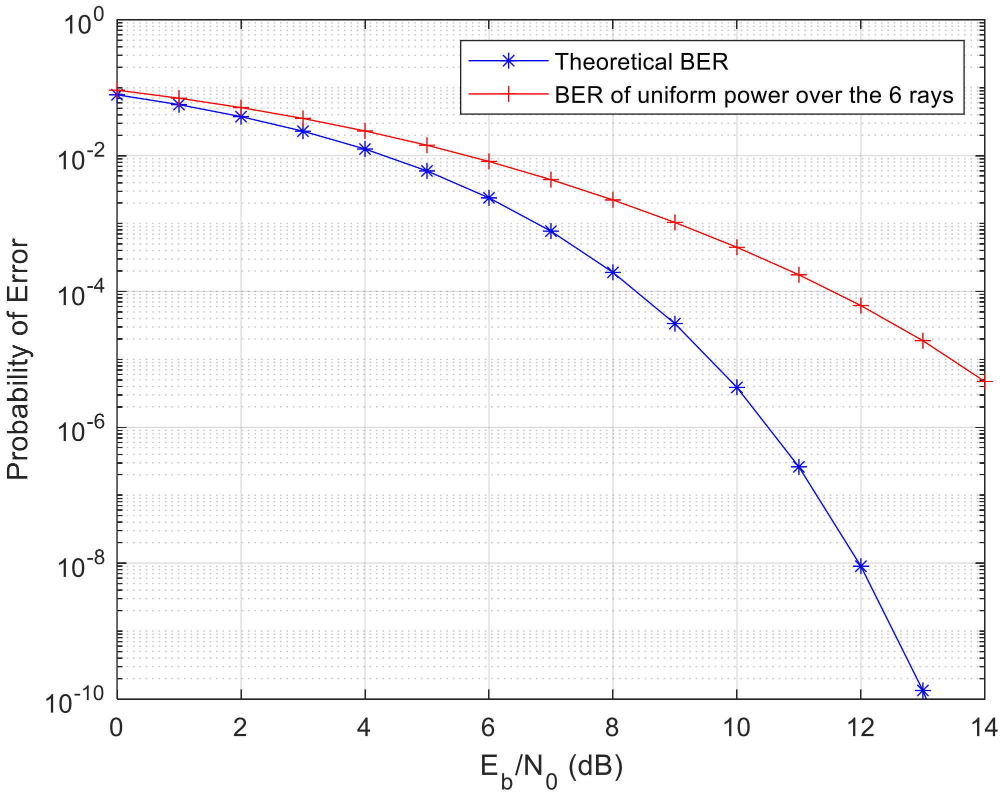
```
clear all
clc

NN = 256;                                % number of symbols
tb = 0.5;                                 % bit time
fs = 16;                                  % samples/symbol
ebn0db = 0:1:14;                          % Eb/N0 vector
%
% Establish QPSK signals
%
x = random_binary(NN,fs)+1i*random_binary(NN,fs);    % QPSK signal
%
% Input powers and delays
%
%功率为指数分布
p0 = 1;
p1 = 0.1*1.5849^4;
p2 = 0.1*1.5849^3;
p3 = 0.1*1.5849^2;
p4 = 0.1*1.5849;
p5 = 0.1;
%delay = input('Enter tau > ');%平坦衰落，delay全为0
delay0 = 0; delay1 = 0; delay2 = 0;
delay3 = 0; delay4 = 0; delay5 = 0;
%
```

```
% Set up the Complex Gaussian (Rayleigh) gains
%
gain1 = sqrt(p1)*abs(randn(1,NN) + 1i*randn(1,NN));
gain2 = sqrt(p2)*abs(randn(1,NN) + 1i*randn(1,NN));
gain3 = sqrt(p3)*abs(randn(1,NN) + 1i*randn(1,NN));
gain4 = sqrt(p4)*abs(randn(1,NN) + 1i*randn(1,NN));
gain5 = sqrt(p5)*abs(randn(1,NN) + 1i*randn(1,NN));
for k = 1:NN
   for kk=1:fs
      index=(k-1)*fs+kk;
      ggain1(1,index)=gain1(1,k);
      ggain2(1,index)=gain2(1,k);
      ggain3(1,index)=gain3(1,k);
      ggain4(1,index)=gain4(1,k);
      ggain5(1,index)=gain5(1,k);
   end
end
y1 = x;
for k=1:delay2
   y2(1,k) = y1(1,k)*sqrt(p0);
end
for k=(delay2+1):(NN*fs)
   y2(1,k)= y1(1,k)*sqrt(p0) + ...
               y1(1,k-delay1)*ggain1(1,k)+...
               y1(1,k-delay2)*ggain2(1,k)+...
               y1(1,k-delay3)*ggain3(1,k)+...
               y1(1,k-delay4)*ggain4(1,k)+...
               y1(1,k-delay5)*ggain5(1,k);
end
%
% Matched filter
%
b = ones(1,fs); b = b/fs; a = 1;
y = filter(b,a,y2);
%
% End of simulation
%
% Use the semianalytic BER estimator. The following sets
% up the semi analytic estimator. Find the maximum magnitude
% of the cross correlation and the corresponding lag.
%
```

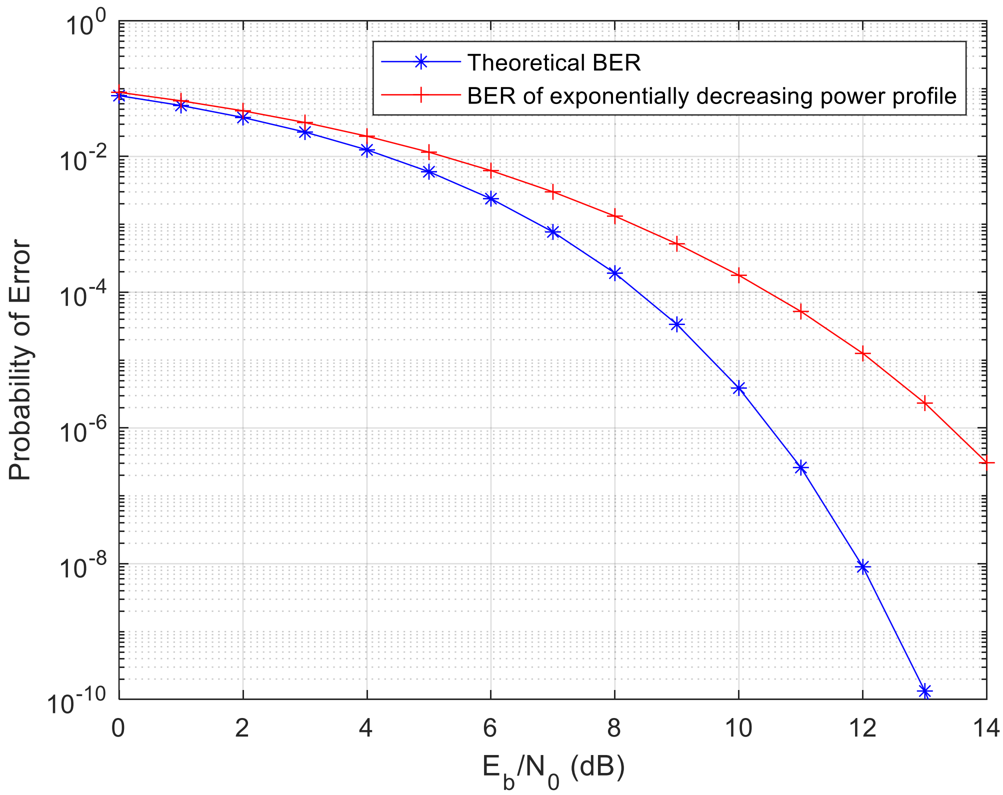
```
[cor lags] = vxcorr(x,y);
[cmax nmax] = max(abs(cor));
timelag = lags(nmax);
theta = angle(cor(nmax));
y = y*exp(-i*theta);                                    % derotate
%
% Noise BW calibration
%
hh = impz(b,a); ts = 1/16; nbw = (fs/2)*sum(hh.^2);
%
% Delay the input, and do BER estimation on the last 128 bits.
% Use middle sample. Make sure the index does not exceed number
% of input points. Eb should be computed at the receiver input.
%
index = (10*fs+8:fs:(NN-10)*fs+8);
xx = x(index);
yy = y(index-timelag+1);
[n1 n2] = size(y2); ny2=n1*n2;
eb = tb*sum(sum(abs(y2).^2))/ny2;
eb = eb/2;
[peideal,pesystem] = qpsk_berest(xx,yy,ebn0db,eb,tb,nbw);
figure(2)
semilogy(ebn0db,peideal,'b*-',ebn0db,pesystem,'r+-')
xlabel('E_b/N_0 (dB)'); ylabel('Probability of Error'); grid
axis([0 14 10^(-10) 1])
legend('Theoretical BER','BER of exponentially decreasing power
profile')
```

4、仿真实验结论

(1) 6 路的功率平均



(2) 6 个抽头具有指数下降的功率曲线，其中最后一个抽头比第一个抽头低 10dB。



实验 3-3

1 基本原理

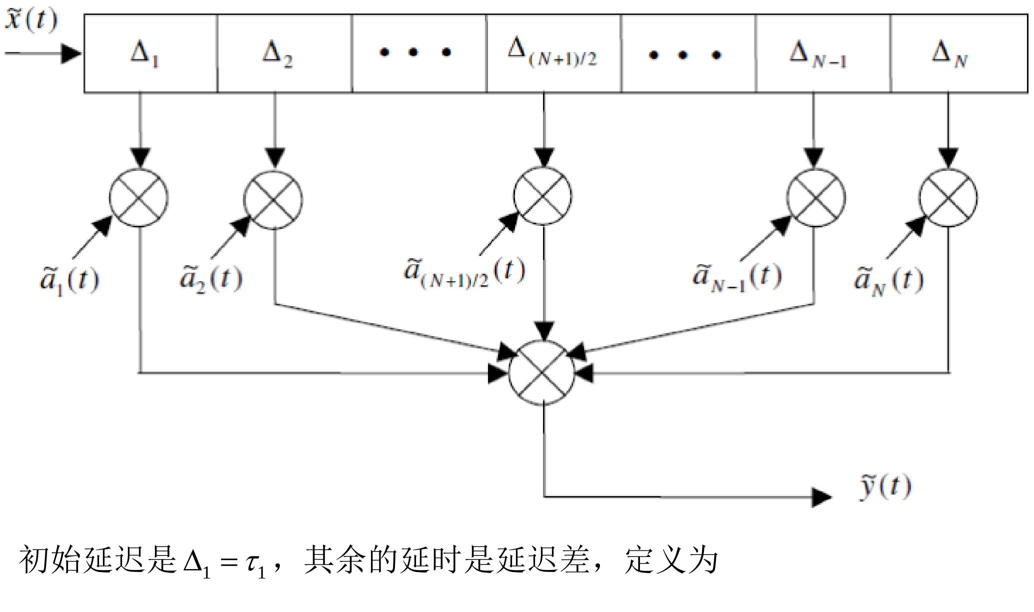
离散多径信道的输入—输出关系由下式给出

$$\tilde{y}(t)=\sum_{k=1}^{N(t)}\tilde{a}_k(t)\tilde{x}(t-\tau_k(t))$$

其中 $\tilde{a}_k(t)$ 是复路径衰减。在上式中，可以假设多径分量的数量和延迟结构的变化比 $\tilde{a}_k(t)$ 的变化要缓慢。 因此，在仿真期间延迟 $\tau_k(t)$ 可以视为常数，于是上式就可以写为

$$\tilde{y}(t)=\sum_{k=1}^{N(t)}\tilde{a}_k(t)\tilde{x}(t-\tau_k)$$

产生一个随机数 N 作为延时的个数；根据延迟值的分布产生 N 个随机数；基于延迟值产生 N 个衰减值。这 3N 个随机数的集合代表了信道的一个瞬时值，其实现方式如框图所示。



初始延迟是 $\Delta_1=\tau_1$，其余的延时是延迟差，定义为

$$\Delta_n=\tau_n-\tau_{n-1}\qquad 2\le n\le N$$

实现框图简单，但当延迟之间只有很小的时间偏移时会出问题。由于所有信号

都得经过采样，在仿真时抽头间隔（即延迟差）必须表示为采样周期的整数倍。因此，采样周期必须非常小，比最小的延迟差还要小，这可能会导致过高的采样率和无法承受的计算负担。

所以仿照仿真散射多径信道所用的方法，建立均匀抽头间隔的 TDL 模型来避免出现这个问题。

一个均匀间隔 TDL 模型的抽头增益

$$\tilde{g}_n(t)=\int_{-\infty}^{\infty}\tilde{c}(\tau,t)\left\{\frac{\sin(2\pi B(\tau-nT))}{2\pi B(\tau-nT)}\right\}d\tau$$

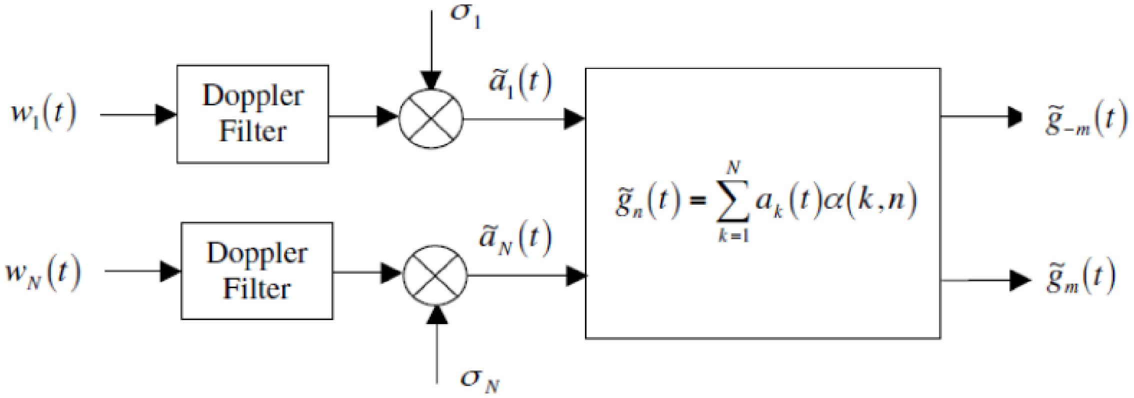
在上式中代入离散多径信道的冲激响应

$$\tilde{c}(\tau,t)=\sum_{k=1}^{N}\tilde{a}_k(t)\delta(\tau-\tau_k)$$

得到抽头增益为

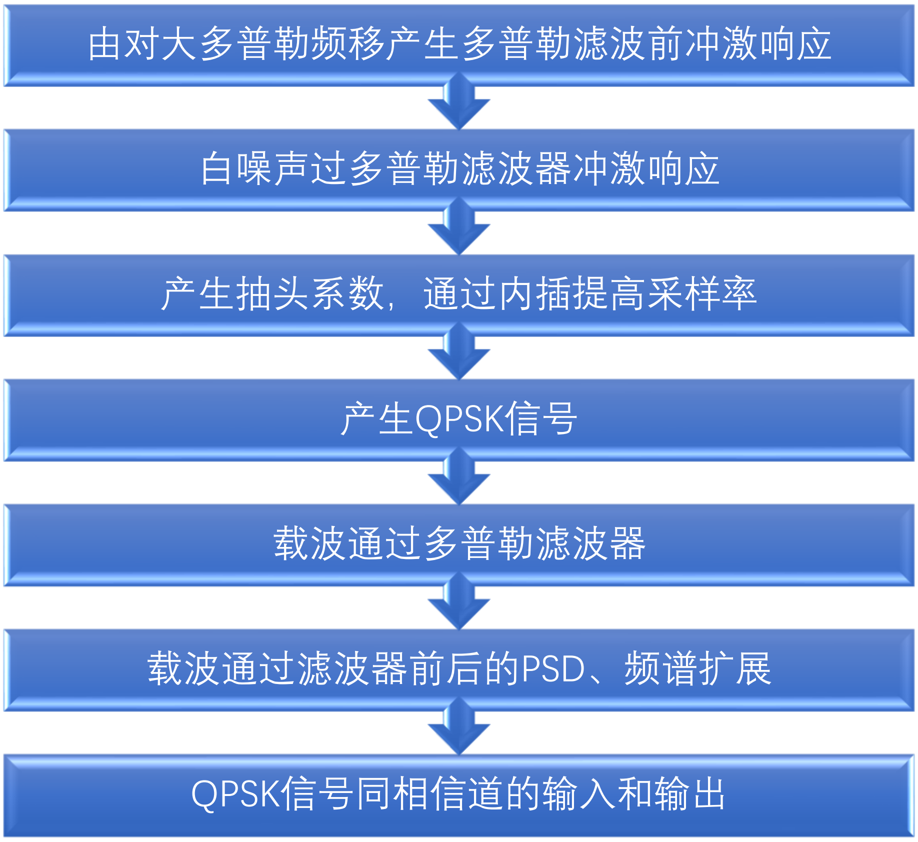
$$\tilde{g}_n(t)=\sum_{k=1}^{N}\tilde{a}_k(t)\mathrm{sinc}\left(\frac{\tau_k}{T}-n\right)=\sum_{k=1}^{N}\tilde{a}_k(t)\alpha(k,n)$$

$$\alpha(k,n)=\mathrm{sinc}\left(\frac{\tau_k}{T}-n\right)$$



抽头增益的产生如图所示(图中为 N 条路径中的 2 条)。 输入一组 N 个独立的、零均值复高斯白噪声过程，滤波来产生合适的多普勒频谱。然后按比例缩放，产生所需的功率曲线。最后按上式作变换后，产生抽头增益过程。

2 仿真实验设计


由对大多普勒频移产生多普勒滤波前冲激响应
白噪声过多普勒滤波器冲激响应
产生抽头系数，通过内插提高采样率
产生QPSK信号
载波通过多普勒滤波器
载波通过滤波器前后的PSD、频谱扩展
QPSK信号同相信道的输入和输出

3 仿真实验实现

```
clear all
clc

%% 带积分清除接收机的普通AWGN信道

Eb = 22:1:28;
No = -50;                              % Eb (dBm) and No
(dBm/Hz)
ChannelAttenuation = 70;               % Channel attenuation in
dB
EbNodB = (Eb-ChannelAttenuation)-No;   % Eb/No in dB
EbNo = 10.^(EbNodB./10);               % Eb/No in linear units
BER_T = 0.5*erfc(sqrt(EbNo));          % BER (theoretical)
N = round(20./BER_T);                  % Symbols to transmit
BER_MC = zeros(size(Eb));              % Initialize BER vector

for k=1:length(Eb)                     % Main Loop
    BER_MC(k) =
MyQPSKrun_usualQPSK(N(k),Eb(k),No,ChannelAttenuation,0,0,0,0);
    disp(['Simulation ',num2str(k*100/length(Eb)),'% Complete']);
end

semilogy(EbNodB,BER_T,'b-',EbNodB,BER_MC,'ro-')%理论误码率
grid on;
xlabel('Eb/No (dB)');
ylabel('Bit Error Rate');
legend('Theoretical BER','MC BER Estimate');


%% 瑞利信道+AWGN

NN = 2^15;                         % number of symbols
tb = 0.5;                          % bit time
fs = 16;                           % samples/symbol
EbNodB1 = 0:1:10;                  % Eb/N0 vector
% Establish QPSK signals
x = random_binary(NN,fs)+1i*random_binary(NN,fs);    % QPSK signal
% Input powers and delays
```

```
%瑞利信道
p0 = 0;
p1 = 0.2;
%delay = input('Enter tau > ');%平坦衰落，delay全为0
delay0 = 0;
delay1 = 0;
delay2 = 0;
% Set up the Complex Gaussian (Rayleigh) gains
gain1 = sqrt(p1)*abs(randn(1,NN) + 1i*randn(1,NN));

for k = 1:NN
   for kk=1:fs
       index=(k-1)*fs+kk;
       ggain1(1,index)=gain1(1,k);
   end
end

y1 = x;

for k=1:delay2
    y2(1,k) = y1(1,k)*sqrt(p0);
end

for k=(delay2+1):(NN*fs)
    y2(1,k)= y1(1,k)*sqrt(p0)+y1(1,k-delay1)*ggain1(1,k);
end

% Matched filter
b = ones(1,fs); b = b/fs; a = 1;
y = filter(b,a,y2);

% End of simulation
% Use the semianalytic BER estimator. The following sets
% up the semi analytic estimator. Find the maximum magnitude
% of the cross correlation and the corresponding lag.
[cor lags] = vxcorr(x,y);
[cmax nmax] = max(abs(cor));
timelag = lags(nmax);
theta = angle(cor(nmax));
y = y*exp(-1i*theta);
```

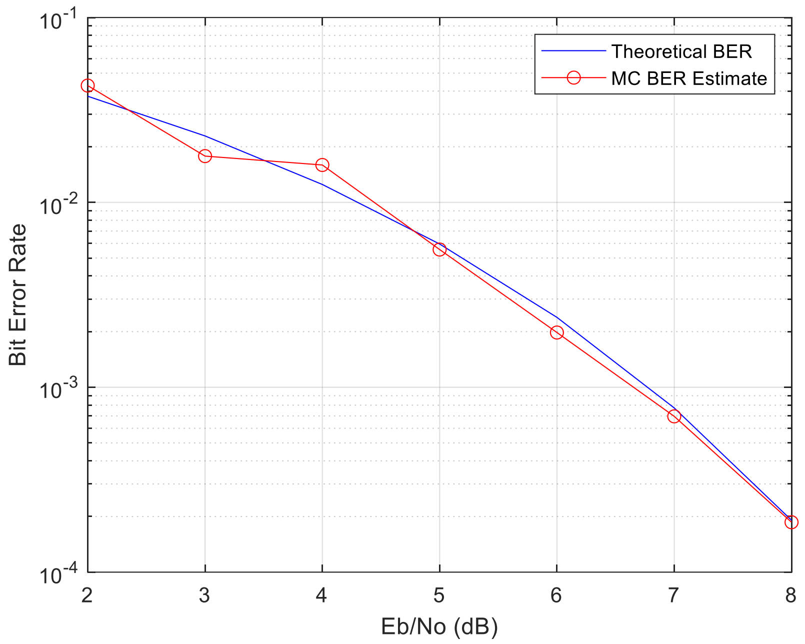
```
% Noise BW calibration
hh = impz(b,a);
ts = 1/16;
nbw = (fs/2)*sum(hh.^2);

% Delay the input, and do BER estimation on the last 128 bits.
% Use middle sample. Make sure the index does not exceed number
% of input points. Eb should be computed at the receiver input.
index = (10*fs+8:fs:(NN-10)*fs+8);
xx = x(index);
yy = y(index-timelag+1);
[n1 n2] = size(y2); ny2=n1*n2;
eb = tb*sum(sum(abs(y2).^2))/ny2;
eb = eb/2;
[peideal_rayleigh,pesystem_rayleigh] =
qpsk_berest(xx,yy,EbNodB1,eb,tb,nbw);

figure(2)
semilogy(EbNodB1,peideal_rayleigh,EbNodB1,pesystem_rayleigh,'ro-')
grid on;
xlabel('E_b/N_0 (dB)');
ylabel('Probability of Error');
legend('AWGN Reference','System Under Rayleigh')


%% 多普勒+AWGN

% Generate tapweights.
fd = 100;
impw = jakes_filter(fd);

% Generate tap input processes and Run through doppler filter.
x1 = randn(1,256)+1i*randn(1,256);
y1 = filter(impw,1,x1);
x2 = randn(1,256)+1i*randn(1,256);
y2 = filter(impw,1,x2);

% Discard the first 128 points since the FIR filter transient.
% Scale them for power and Interpolate weight values.
% Interpolation factor=100 for the QPSK sampling rate of 160000/sec.
z1(1:128) = y1(129:256);
```

```
z2(1:128) = y2(129:256);
z2 = sqrt(0.5)*z2;
m = 100;
tw1 = linear_interp(z1,m);
tw2 = linear_interp(z2,m);

% Generate QPSK signal and filter it.
nbits = 512;
nsamples = 16;
ntotal = 8192;
qpsk_sig =
random_binary(nbits,nsamples)+1i*random_binary(nbits,nsamples);
[b,a] = butter(5,1/16);                    % transmitter filter
parameters

y2 = filter(b,a,qpsk_sig);                 % filtered signalv

% Genrate output of tap1 (size the vectors first).
%input1 = qpsk_sig(1:8184); output1 = tw1(1:8184).*input1;
% Delay the input by eight samples (this is the delay specified
% in term of number of samples at the sampling rate of
% 16,000 samples/sec and genrate the output of tap 2.
%input2 = qpsk_sig(9:8192); output2 = tw2(9:8192).*input2;
% Add the two outptus and genrate overall output.
qpsk_output = tw1(1:8192).*y2;

% Matched filter
b = ones(1,nsamples);
b = b/nsamples; a = 1;                      % matched filter
parameters
y = filter(b,a,qpsk_output);                % matched filter output

%基带信号解调，半解析法求误码率
[cor lags] = vxcorr(qpsk_sig,y);
cmax = max(abs(cor));
nmax = find(abs(cor)==cmax);
timelag = lags(nmax);
theta = angle(cor(nmax));
y = y*exp(-1i*theta);                      % derotate

% Noise BW calibration
```

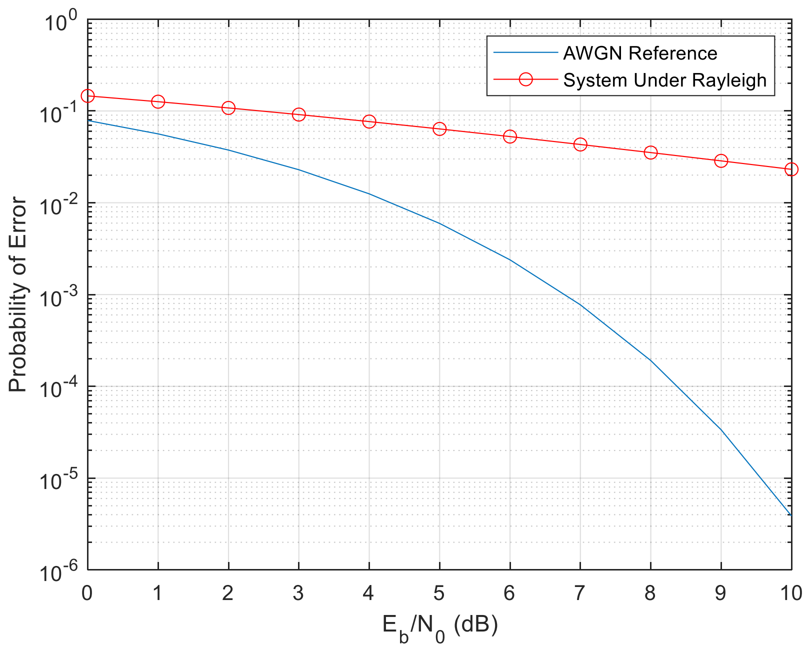
```
hh = impz(b,a);                          % receiver impulse response
nbw = (nsamples/2)*sum(hh.^2);           % noise bandwidth

% Delay the input, and do BER estimation on the last 128 bits.
% Use middle sample. Make sure the index does not exceed number
% of input points. Eb should be computed at the receiver input.
index = (10*nsamples+8:nsamples:(nbits-10)*nsamples+8);
xx = qpsk_sig(index);
yy = y(index-timelag+1);
[n1 n2] = size(y2);
ny2=n1*n2;
eb = tb*sum(sum(abs(y2).^2))/ny2;
eb = eb/2;
[peideal,pesystem] = qpsk_berest(xx,yy,EbNodB,eb,tb,nbw);

figure(3)
semilogy(EbNodB,peideal,EbNodB,pesystem,'ro-');
grid on;
xlabel('E_b/N_0 (dB)');
ylabel('Bit Error Rate');
legend('AWGN Reference','System Under Doppler')
```

4、仿真实验结论

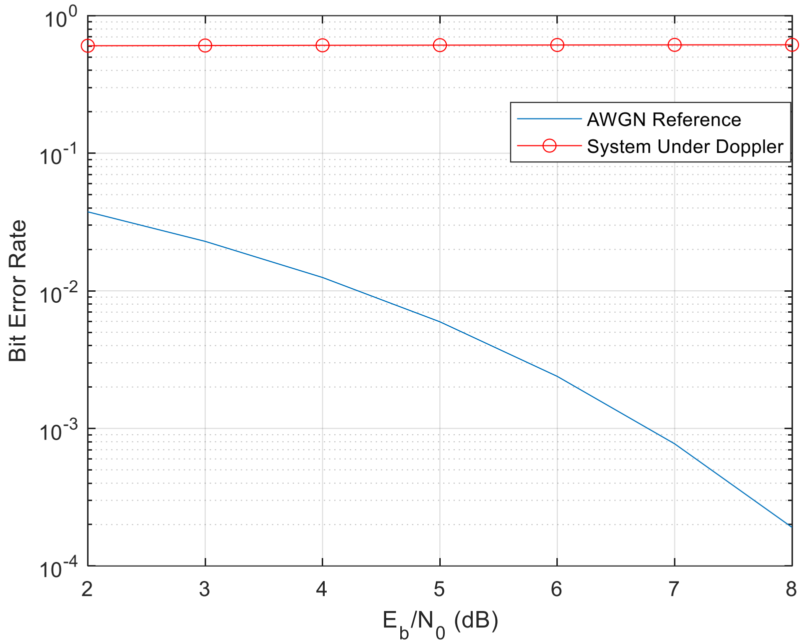
(1) 有积分—清除接收机的 AWGN 信道



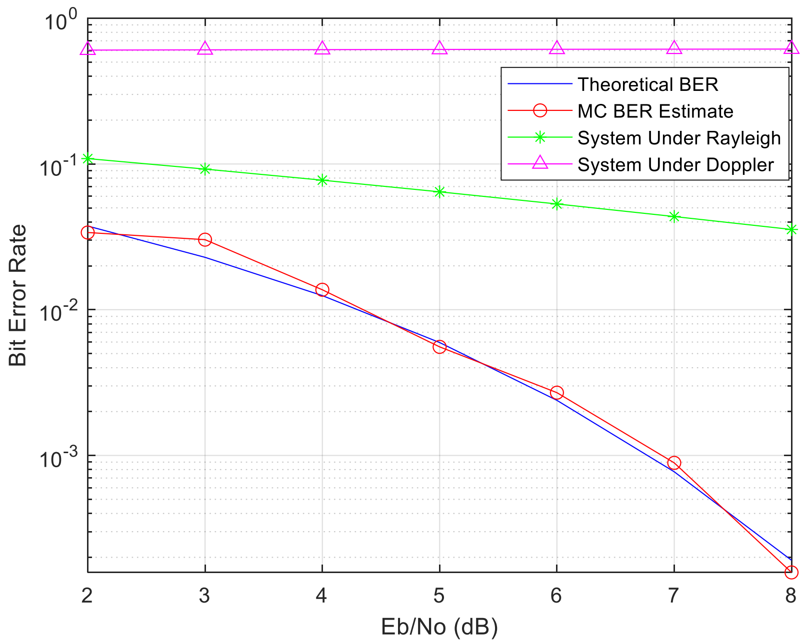
(2) 瑞利衰落信道加 AWGN 信道，无时延



(3) 由于多普勒原因引起的衰落信道



(4) 三者对比



从图中可以看出，AWGN 信道下 BER 与理论 BER 比较接近，而在瑞利衰落信道和多普勒原因引起的衰落信道下 BER 大幅提高。且在多普勒原因引起的衰落信道下 BER 很大，基本不随信噪比的变化而变化。