**北京科技大学实验报告**

学院：计算机与通信工程学院 专业：通信工程 班级：通信1701

姓名：胡成成 学号：41724260 实验日期： 2020年 5月 2日

**实验名称：无线通信信道建模仿真**

**实验目的：**

1. 熟悉信道衰落对移动通信系统性能的影响；
2. 掌握移动多径信道特性及信道模型；
3. 掌握不同信道衰落条件下对传输信号误码率的影响；

**实验内容：**

1. 编写Matlab程序仿真建立不同信道模型：高斯信道、Rayleigh信道，Rcian信道及多径衰落信道模型；
2. 编写Matlab程序仿真输入QPSK调制信号，并让信号通过上述信道模型，接收端解调后计算上述信道条件下的误码率性能；
3. 分析信道参数、信噪比对误码率性能的影响；
4. 分析仿真中观察的波形数据，撰写实验报告。

**实验原理：**

1. **QPSK**

QPSK信号是由串行二进制信息序列经串－并变换，变成个并行数据流，每一路的数据率是R/m，R是串行输入码的数据率。I/Q信号发生器 将每一个m比特的字节转换成一对（）数字，分成两路速率减半的序列，电平发生器分别产生双极性二电平信号I(t)和Q(t)，然后对和进行调制，相加后所得。

四相相移调制(QPSK)是利用载波的四种不同相位差来表征输入的数字信息，是四进制移相键控。QPSK是在M=4时的调相技术，它规定了 四种载波相位，分别为45°，135°，225°，275°，调制器输入的数据是二进制数字序列，为了能和四进制的载波相位配合起来，则需要把二进制数据 变换为四进制数据，这就是说需要把二进制数字序列中每两个比特分成一组，共有四种组合，即00，01，10，11，其中每一组称为双比特码元。每一个双比 特码元是由两位二进制信息比特组成，它们分别代表四进制四个符号中的一个符号。

1. **瑞利分布模型**

在移动无线信道中，瑞利模型是常见的用于描述平坦衰落信号或独立多径分量接收包络统计时变特性的一种经典模型。瑞利分布的概率密度函数（PDF）为：



其中，是包络检波之前的接收信号包络的时间平均功率。R 的相位 θ 服从 0 到 2π 之间的均匀分布，即:



则接收信号包络不超过某特定值 R的累计概率分布函数（CDF）为:



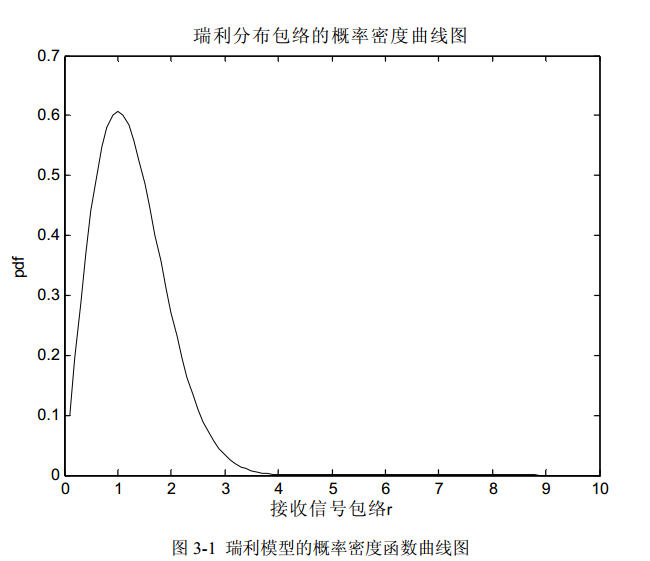


图1 瑞利模型的概率密度函数曲线图

1. **莱斯分布模型**

当接收端存在一个主要的静态（非衰落）信号时，如LOS分量（在郊区和农村等开阔区域中，接收端经常会接收到的）等，此时接收端接收的信号的包络就服从莱斯分布。在这种情况下，从不同角度随机到达的多径分量迭加在静态的主要信号上，即包络检波器的输出端就会在随机的多径分量上迭加一个直流分量。当主要信号分量减弱后，莱斯分布就转变为瑞利分布。莱斯分布的概率密度函数为：



其中C是指主要信号分量的幅度峰值，是0阶第一类修正贝赛尔函数。为了更好的分析莱斯分布，定义主信号的功率与多径分量方差之比为莱斯因子K，则K的表达式可以写为 :



莱斯分布完全由莱斯因子K决定。图3-2所示为莱斯模型的概率密度函数曲线图。

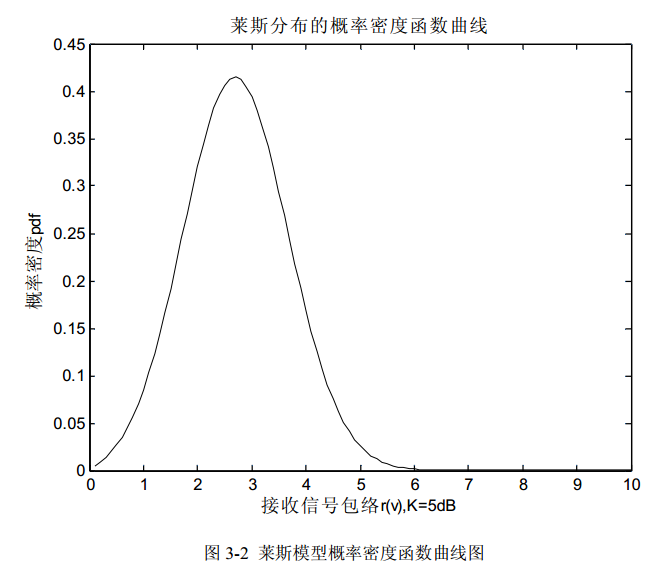


图2 莱斯模型概率密度函数曲线图

1. **高斯分布模型**

常指加权高斯白噪声信道。这种噪声假设为在整个信道带宽下功率谱密度（PDF）为常数，并且振幅符合高斯概率分布。高斯信道对于评价系统性能的上界具有重要意义，对于实验中定量或定性地评价某种调制方案、误码率性能等有重要作用。



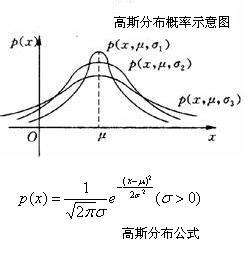


图3高斯分布的公式与概率密度函数分布图

1. **多径效应**

由于接收者所处地理环境的复杂性、使得接收到的信号不仅有直射波的主径信号，还有从不同建筑物反射过来以及绕射过来的多条不同路径信号。而且它们到达时的信号强度，到达时间以及到达时的载波相位都是不一样的。所接收到的信号是上述各路径信号的矢量和，称这类自干扰为多径干扰或多径效应。

电波传播信道中的多径传输现象所引起的干涉延时效应。在实际的无线电波传播信道中(包括所有波段)，常有许多时延不同的传输路径。各条传播路径会随时间变化，参与干涉的各分量场之间的相互关系也就随时间而变化，由此引起合成波场的随机变化，从而形成总的接收场的衰落。因此，多径效应是衰落的重要成因。多径效应对于数字通信、雷达最佳检测等都有着十分严重的影响。

多径效应移动体(如汽车)往来于建筑群与障碍物之间，其接收信号的强度，将由各直射波和反射波叠加合成。多径效应会引起信号衰落。各条路径的电长度会随时间而变化，故到达接收点的各分量场之间的相位关系也是随时间而变化的。这些分量场的随机干涉，形成总的接收场的衰落。各分量之间的相位关系对不同的频率是不同的。因此，它们的干涉效果也因频率而异，这种特性称为频率选择性。在宽带信号传输中，频率选择性可能表现明显，形成交调。与此相应，由于不同路径有不同时延，同一时刻发出的信号因分别沿着不同路径而在接收点前后散开，而窄脉冲信号则前后重叠。

多径时延特性可用时延谱或多径散布谱（即不同时延的信号分量平均功率构成的谱）来描述。与时延谱等价的是频率相关函数。实际上，人们只简单利用时延谱的某个特征量来表征。例如，用最大时延与最小时延的差，表征时延谱的尖锐度和信道容许传输带宽。这个值越小，信道容许传输频带越宽。

**实验步骤：**

1. 预习信道衰落、多径效应及多种信道模型；
2. 画出仿真流程图；
3. 编写Matlab程序并上机调试；
4. 观察分析波形、数据；
5. 撰写实验报告

**实验结果：**

1. QPSK信号的调制与解调

% 调相法

clear all

close all

t=[-1:0.01:7-0.01];

tt=length(t);

x1=ones(1,800);

for i=1:tt

if (t(i)>=-1 & t(i)<=1) | (t(i)>=5& t(i)<=7);

x1(i)=1;

else x1(i)=-1;

end

end

t1=[0:0.01:8-0.01];

t2=0:0.01:7-0.01;

t3=-1:0.01:7.1-0.01;

t4=0:0.01:8.1-0.01;

tt1=length(t1);

x2=ones(1,800);

for i=1:tt1

if (t1(i)>=0 & t1(i)<=2) | (t1(i)>=4& t1(i)<=8);

x2(i)=1;

else x2(i)=-1;

end

end

f=0:0.1:1;

xrc=-0.5+0.5\*cos(pi\*f);

y1=conv(x1,xrc)/5.5;

y2=conv(x2,xrc)/5.5;

n0=randn(size(t2));

f1=1;

i=x1.\*cos(2\*pi\*f1\*t);

q=x2.\*sin(2\*pi\*f1\*t1);

I=i(101:800);

Q=q(1:700);

QPSK=sqrt(1/2).\*I+sqrt(1/2).\*Q;

QPSK\_n=(sqrt(1/2).\*I+sqrt(1/2).\*Q)+n0;

n1=randn(size(t2));

i\_rc=y1.\*cos(4\*pi\*f1\*t3);

q\_rc=y2.\*sin(4\*pi\*f1\*t4);

I\_rc=i\_rc(101:800);

Q\_rc=q\_rc(1:700);

QPSK\_rc=(sqrt(1/2).\*I\_rc+sqrt(1/2).\*Q\_rc);

QPSK\_rc\_n1=QPSK\_rc+n1;

figure(1)

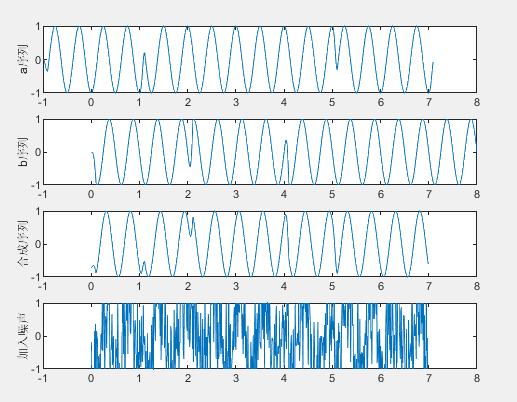
subplot(4,1,1);plot(t3,i\_rc);axis([-1 8 -1 1]);ylabel('a序列');

subplot(4,1,2);plot(t4,q\_rc);axis([-1 8 -1 1]);ylabel('b序列');

subplot(4,1,3);plot(t2,QPSK\_rc);axis([-1 8 -1 1]);ylabel('合成序列');

subplot(4,1,4);plot(t2,QPSK\_rc\_n1);axis([-1 8 -1 1]);ylabel('加入噪声');

运行结果：



1. QPSK加噪声调制解调过程

% 设定 T=1,加入高斯噪声

clear all

close all

% 调制

bit\_in = randi([0 1],1e3,1);

bit\_I = bit\_in(1:2:1e3);

bit\_Q = bit\_in(2:2:1e3);

data\_I = -2\*bit\_I+1;

data\_Q = -2\*bit\_Q+1;

data\_I1=repmat(data\_I',20,1);

data\_Q1=repmat(data\_Q',20,1);

for i=1:1e4

data\_I2(i)=data\_I1(i);

data\_Q2(i)=data\_Q1(i);

end;

f=0:0.1:1;

xrc=0.5+0.5\*cos(pi\*f);

data\_I2\_rc=conv(data\_I2,xrc)/5.5;

data\_Q2\_rc=conv(data\_Q2,xrc)/5.5;

f1=1;

t1=0:0.1:1e3+0.9;

n0=rand(size(t1));

I\_rc=data\_I2\_rc.\*cos(2\*pi\*f1\*t1);

Q\_rc=data\_Q2\_rc.\*sin(2\*pi\*f1\*t1);

QPSK\_rc=(sqrt(1/2).\*I\_rc+sqrt(1/2).\*Q\_rc);

QPSK\_rc\_n0=QPSK\_rc+n0;

% 解调

I\_demo=QPSK\_rc\_n0.\*cos(2\*pi\*f1\*t1);

Q\_demo=QPSK\_rc\_n0.\*sin(2\*pi\*f1\*t1);

% 低通滤波

I\_recover=conv(I\_demo,xrc);

Q\_recover=conv(Q\_demo,xrc);

I=I\_recover(11:10010);

Q=Q\_recover(11:10010);

t2=0:0.05:1e3-0.05;

t3=0:0.1:1e3-0.1;

% 抽样判决

data\_recover=[];

for i=1:20:10000

data\_recover=[data\_recover I(i:1:i+19) Q(i:1:i+19)];

end;

bit\_recover=[];

for i=1:20:20000

if sum(data\_recover(i:i+19))>0

data\_recover\_a(i:i+19)=1;

bit\_recover=[bit\_recover 1];

else

data\_recover\_a(i:i+19)=-1;

bit\_recover=[bit\_recover -1];

end

end

error=0;

dd = -2\*bit\_in+1;

ddd=[dd'];

ddd1=repmat(ddd,20,1);

for i=1:2e4

ddd2(i)=ddd1(i);

end

for i=1:1e3

if bit\_recover(i)~=ddd(i)

error=error+1;

end

end

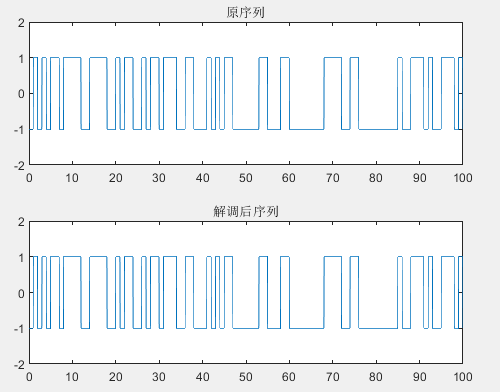
p=error/1000;

figure(1)

subplot(2,1,1);plot(t2,ddd2);axis([0 100 -2 2]);title('原序列');

subplot(2,1,2);plot(t2,data\_recover\_a);axis([0 100 -2 2]);title('解调后序列');

运行结果：



3. QPSK不加噪声调制解调过程

% 设定 T=1, 不加噪声

clear all

close all

% 调制

bit\_in = randint(1e3, 1, [0 1]);

bit\_I = bit\_in(1:2:1e3);

bit\_Q = bit\_in(2:2:1e3);

data\_I = -2\*bit\_I+1;

data\_Q = -2\*bit\_Q+1;

data\_I1=repmat(data\_I',20,1);

data\_Q1=repmat(data\_Q',20,1);

for i=1:1e4

data\_I2(i)=data\_I1(i);

data\_Q2(i)=data\_Q1(i);

end;

t=0:0.1:1e3-0.1;

f=0:0.1:1;

xrc=0.5+0.5\*cos(pi\*f);

data\_I2\_rc=conv(data\_I2,xrc)/5.5;

data\_Q2\_rc=conv(data\_Q2,xrc)/5.5;

f1=1;

t1=0:0.1:1e3+0.9;

I\_rc=data\_I2\_rc.\*cos(2\*pi\*f1\*t1);

Q\_rc=data\_Q2\_rc.\*sin(2\*pi\*f1\*t1);

QPSK\_rc=(sqrt(1/2).\*I\_rc+sqrt(1/2).\*Q\_rc);

% 解调

I\_demo=QPSK\_rc.\*cos(2\*pi\*f1\*t1);

Q\_demo=QPSK\_rc.\*sin(2\*pi\*f1\*t1);

I\_recover=conv(I\_demo,xrc);

Q\_recover=conv(Q\_demo,xrc);

I=I\_recover(11:10010);

Q=Q\_recover(11:10010);

t2=0:0.05:1e3-0.05;

t3=0:0.1:1e3-0.1;

data\_recover=[];

for i=1:20:10000

data\_recover=[data\_recover I(i:1:i+19) Q(i:1:i+19)];

end;

ddd = -2\*bit\_in+1;

ddd1=repmat(ddd',10,1);

for i=1:1e4

ddd2(i)=ddd1(i);

end

figure(1)

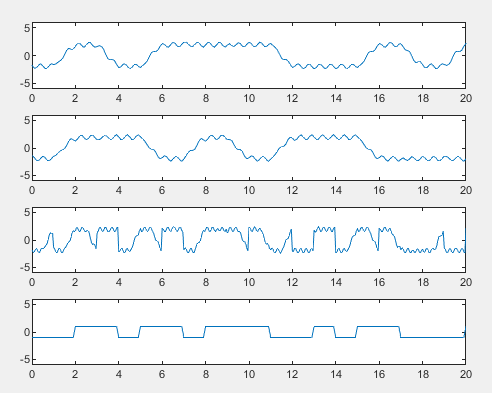
subplot(4,1,1);plot(t3,I);axis([0 20 -6 6]);

subplot(4,1,2);plot(t3,Q);axis([0 20 -6 6]);

subplot(4,1,3);plot(t2,data\_recover);axis([0 20 -6 6]);

subplot(4,1,4);plot(t,ddd2);axis([0 20 -6 6]);

运行结果：



1. QPSK误码率分析

% QPSK误码率分析

SNRindB1=0:2:10;

SNRindB2=0:0.1:10;

for i=1:length(SNRindB1)

[pb,ps]=cm\_sm32(SNRindB1(i));

smld\_bit\_err\_prb(i)=pb;

smld\_symbol\_err\_prb(i)=ps;

end;

for i=1:length(SNRindB2)

SNR=exp(SNRindB2(i)\*log(10)/10);

theo\_err\_prb(i)=Qfunct(sqrt(2\*SNR));

end;

title('QPSK误码率分析');

semilogy(SNRindB1,smld\_bit\_err\_prb,'\*');

axis([0 10 10e-8 1]);

hold on;

% semilogy(SNRindB1,smld\_symbol\_err\_prb,'o');

semilogy(SNRindB2,theo\_err\_prb);

legend('仿真比特误码率','理论比特误码率');

hold off;

function[y]=Qfunct(x)

y=(1/2)\*erfc(x/sqrt(2));

function[pb,ps]=cm\_sm32(SNRindB)

N=10000;

E=1;

SNR=10^(SNRindB/10);

sgma=sqrt(E/SNR)/2;

s00=[1 0];

s01=[0 1];

s11=[-1 0];

s10=[0 -1];

for i=1:N

dsource1(i)=[1 0 1 1 0 0 0 1 0 1 1 0 1 0 1 1];

numofsymbolerror=0;

numofbiterror=0;

for i=1:N

n=sgma\*randn(size(s00));

if((dsource1(i)==0)&(dsource2(i)==0))

r=s00+n;

elseif((dsource1(i)==0)&(dsource2(i)==1))

r=s01+n;

elseif((dsource1(i)==1)&(dsource2(i)==0))

r=s10+n;

else

r=s11+n;

end;

c00=dot(r,s00);

c01=dot(r,s01);

c10=dot(r,s10);

c11=dot(r,s11);

c\_max=max([c00 c01 c10 c11]);

if (c00==c\_max)

decis1=0;decis2=0;

elseif(c01==c\_max)

decis1=0;decis2=1;

elseif(c10==c\_max)

decis1=1;decis2=0;

else

decis1=1;decis2=1;

end;

symbolerror=0;

if(decis1~=dsource1(i))

numofbiterror=numofbiterror+1;

symbolerror=1;

end;

if(decis2~=dsource2(i))

numofbiterror=numofbiterror+1;

symbolerror=1;

end;

if(symbolerror==1)

numofsymbolerror=numofsymbolerror+1;

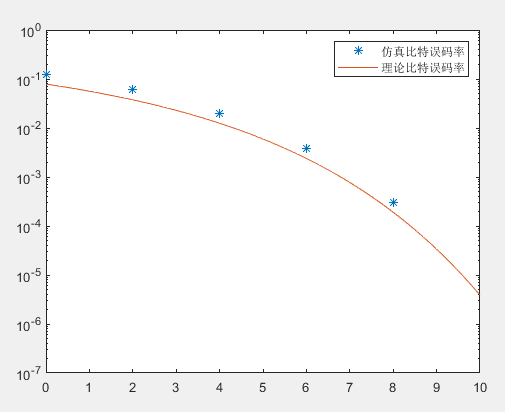
end;

end;

ps=numofsymbolerror/N;

pb=numofbiterror/(2\*N);

运行结果：



1. 信道模型建立

①产生高斯信号：

function H = Gauss\_model(std\_val,N)

% Gauss\_model Channel Model

% std\_val 方差

H = sqrt(std\_val)\*randn(N,1);

②产生瑞利信号：

function H = Ray\_model(L)

% Rayleigh Channel Model

H = (randn(1,L) + j\*randn(1,L))/sqrt(2); % 能量为 1

③产生莱斯信号：

function H = Ric\_model(K\_dB,L)

% Ric\_model Channel Model

K = 10^(K\_dB/10);

H = sqrt(K/(K+1)) + sqrt(1/(K+1))\*Ray\_model(L);

④信道模拟作图：

% 信道模拟

N=1e5;

level=50;

% Gauss model 高斯

std\_val = 0.2; % 方差=0.2;均值=0

Gauss\_ch = Gauss\_model(std\_val, N);

% 直方图的纵轴为频次，概率密度的纵轴为频率，但两者大致的分布曲线是一样的

[temp, x] = hist(abs(Gauss\_ch), level);

plot(x, temp, 'k-s');

% Rayleigh model 瑞利

Rayleigh\_ch = Ray\_model(N); % 获得瑞利信道函数

[temp1,y] = hist(abs(Rayleigh\_ch(1,:)), level); % hist结果是落入每个区间的个数

% Rician model 莱斯

K\_dB = 5;

Rician\_ch = Ric\_model(K\_dB, N);

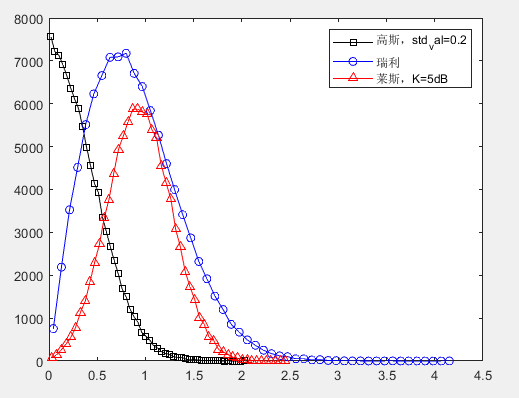
[temp2, z] = hist(abs(Rician\_ch), level);

figure(1)

plot(x, temp, 'k-s',y, temp1, 'b-o',z, temp2, 'r-^');

legend('高斯，std\_val=0.2','瑞利','莱斯，K=5dB');

运行结果：



1. 多径效应

①二径信道

clear all

ht=10;hr=2;

c=3e8;f=1e9;lambda=c/f;

R=-1;

d=1:0.5:10000;

d1=sqrt(d.^2+(ht-hr)^2);

d2=sqrt(d.^2+(ht+hr)^2);

a1=exp(j\*2\*pi.\*d1/lambda)./d1;

a2=R\*exp(j\*2\*pi.\*d2/lambda)./d2;

a=abs(a1+a2);

ld=log10(d);la=log10(a);

figure (4)

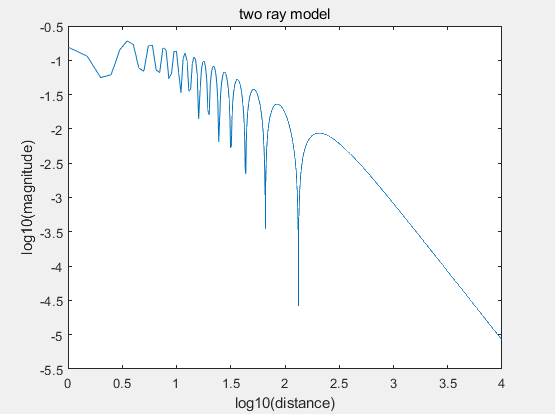
plot(ld,la);

xlabel('log10(distance)')

ylabel('log10(magnitude)')

title( 'two ray model')

运行结果：



②频率选择性衰落

% ②频率选择性衰落

clear all;

s=[ones(1,10),zeros(1,90)]; % transmitted signal

s\_f=fft(s);

x=s\_f([1:50]);

y=s\_f([51:100]);

signal\_f=[y,x]; %input spectrum

dt=5/10; % each time interval is 0.01 micro sec

df=1/(100\*dt);

f\_s=df\*([0:99]-50);% frequecy vector

an=[1,0.3,-0.8,0.5,-0.4,0.2]; %amplitudes

f=f\_s;

w=2\*pi\*f;

tn\_1=[0,1,2,3,4,5]; % arrival times for case 1

for i=1:6;

h1(i,:)=an(i)\*exp(-j\*w\*tn\_1(i));

end

h\_1=sum(h1(:,1:end));%transfer function

y\_1=h\_1.\*signal\_f;%output spectrum

tn\_2=[0,0.1,0.2,0.3,0.4,0.5]; % arrival times for case 2

for i=1:6;

h2(i,:)=an(i)\*exp(-j\*w\*tn\_2(i));

end

h\_2=sum(h2(:,1:end));%transfer function

y\_2=h\_2.\*signal\_f;%output spectrum

figure(1)

subplot(2,3,1);

plot(f\_s,abs(signal\_f));

ylabel('magnitude');title('I/P spectrum')

subplot(2,3,4);

plot(f\_s,angle(signal\_f));

ylabel('Phase');

xlabel('Frequency(MHz)');

subplot(2,3,2);

plot(f,abs(h\_1));

title('channel 1')

subplot(2,3,5);

plot(f,angle(h\_1));

xlabel('Frequency(MHz)');

subplot(2,3,3);

plot(f,abs(h\_2));

title('channel 2')

subplot(2,3,6);

plot(f,angle(h\_2));

xlabel('Frequency(MHz)');

figure(2)

subplot(2,3,1);

plot(f\_s,abs(signal\_f));

ylabel('magnitude');title('I/P spectrum')

subplot(2,3,4);

plot(f\_s,angle(signal\_f));

ylabel('Phase');

xlabel('Frequency(MHz)');

subplot(2,3,2);

plot(f,abs(y\_1));

title('O/P spectrum 1')

subplot(2,3,5);

plot(f,angle(y\_1));

xlabel('Frequency(MHz)');

subplot(2,3,3);

plot(f,abs(y\_2));

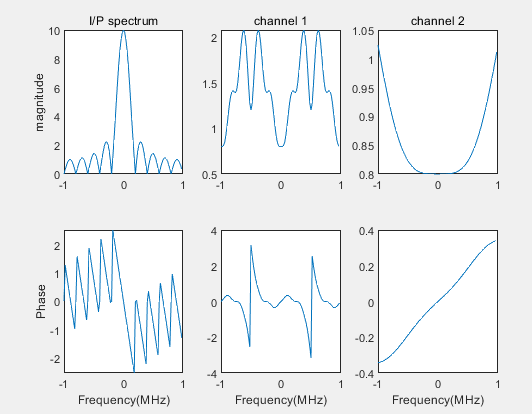
title('O/P spectrum 2')

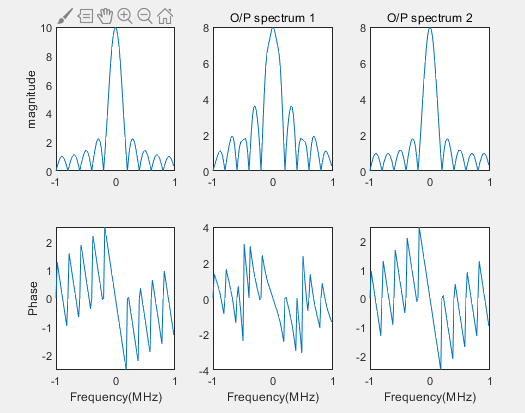
subplot(2,3,6);

plot(f,angle(y\_2));

xlabel('Frequency(MHz)');

运行结果：





**思考题：**

1. 除了本实验采用的信道模型，请搜集了解其他常见的无线通信信道模型；

答：①平坦衰落：当信号传输带宽较窄(<10Mb/s)时,可以忽略频率选择性的影响,认为在信号传输带宽内具有相同的电平衰落深度,这种衰落称为平衰落. 平衰落储备是当考虑热噪声的影响时,为保证设备的门限误码率必须留有的电平余量,即自由空间条件下收信电平与门限电平的差值.

②时不变：平稳的信道或系统，不随时间的变化而变化；时变性：不稳定的信道或系统，随时间的变化而变化。

③广义平稳信道、不相关散射信道、广义平稳不相关散射信道

1. 除了平坦性衰落和频率选择性衰落，还有由多普勒效应引起的快慢衰落，请简述其成因及不同衰落的区别。

答：①慢衰落产生的原因：路径损耗，这是慢衰落的主要原因。由大气折射、大气湍流、大气层结等平均大气条件的变化而引起的，通常与频率的关系不大，而主要与气象条件、电路长度、地形等因素有关。它反映了中等范围内数百波长量级接收电平的均值变化而产生的损耗，一般遵从对数正态分布。

②快衰落产生原因：主要由于多径传播而产生的衰落，移动台附近的散射体（地形，地物和移动体等）引起的多径传播信号在接收点相叠加，造成接收信号快速起伏。它反映微观小范围内数十波长量级接收电平的均值变化而产生的损耗，其变化率比慢衰落快。

③当信号变化的频率比信道变化的频率小的多时为慢衰落，大的多时为快衰落

**北京科技大学实验报告**

学院：计算机与通信工程学院 专业：通信工程 班级：通信1701

姓名：胡成成 学号：41724260 实验日期： 2020年 5月 8日

**实验名称：多址接入技术仿真实验**

**实验目的：**

1、理解多址接入技术原理；

2、掌握CDMA（码分多址）调制与解调原理；

3、熟悉CDMA仿真系统建模；

**实验内容：**

1、学习多址接入技术原理；

2、编写MATLAB程序代码实现CDMA系统仿真，CDMA系统框图如图所示；

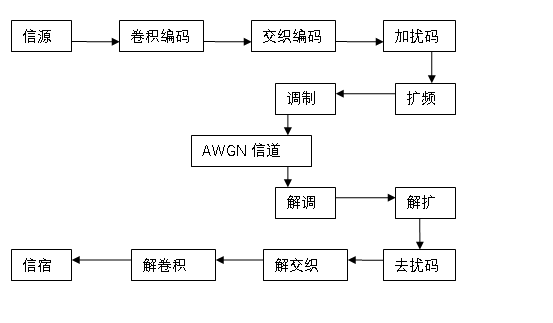


图2-1 CDMA系统框图

3、分析仿真中观察的波形数据，撰写实验报告。

**实验原理：**

1、多址接入技术原理

从移动通信网的构成方面来讲，大部分移动通信系统都有一个或多个移动台。基站要和许多移动台同时通信，因而基站通常是多路的，有多个信道；而每个移动台只供一个用户使用，是单路的。许多用户同时通话，以不用的通道分隔，防止相互干扰；各用户信号通过在射频频段上的复用，从而建立各自的信道，以实现双边通信的连接。可见，基站的多路工作和移动台的单路工作是移动通信的一大特点。在移动通信业务区内，移动台之间或移动台与市话用户之间是通过基站同时建立各自的信道，从而实现多址连接的。

多址接入方式的数学基础是信号的正交分割原理，原理上与固定通信中的信号复用相似，但有所不同。多路复用的目的是区分多个通路，通常在基带和中频上实现，而多址划分是区分不同的用户地址，同时为了实现多址信号之间互不干扰，不同用户无线电信号之间必须满足正交特性。信号的正交性是通过信号正交参量来实现的。当正交参量仅考虑时间、频率和码型时，无线电信号写成: s(c,f,t)=c(t)s(f,t)，式中，c(t)是码型函数；s(f,t)是时间t和频率f的函数。

有多种方式来区分不同的用户地址，如频分多址（FDMA）是以传输信号载波频率的不同来区分；时分多址（TDMA）是以传输信号存在的时间不同来区分；码分多址（CDMA）是以传输信号的码型不同来区分。

每一代通信系统有自己独特的多址接入技术。多址接入技术的目的是让多个用户能同时接入基站，享受基站提供的通信服务，保证各个用户之间的信号不会互相干扰。

第一代移动通信系统（1G）主要采用频分多址接入方式（FDMA），第二代移动通信系统（2G）主要采用时分多址接入方式（TDMA），第三代移动通信系统（3G）主要采用码分多址接入方式（CDMA），第四代通信系统（4G）主要采用正交频分复用多址接入方式（OFDMA），而非正交多址接入方式（NOMA）是下一代移动通信系统（5G）一个热门的技术。

2、CDMA基础及原理

CDMA多址技术的原理是基于扩频技术，即将需传送的具有一定信号带宽的信息数据，用一个带宽远大于信号带宽的高速伪随机码进行调制，使原数据信号的带宽被扩展，再经载波调制并发送出去。接收端由使用完全相同的伪随机码，与接收的带宽信号作相关处理，把宽带信号换成原信息数据的窄带信号即解扩，以实现信息通信。

**（1）扩频通信**

扩频通信技术是一种信息传输方式，其信号所占有的频带宽度远大于所传信息所需的最小带宽；频带的扩展是通过一个独立的码序列来完成，用编码及调制的方法来实现的，与所传信息数据无关；在接收端则用同样的码进行相关同步接受、解扩及恢复所传信息数据。

**（2）扩频通信理论基础**

香农公式：C=Wlog2(1+S/N)

1、在给定的传输速率C不变的条件下，频带宽度W和信噪比S/N是可以互换的。即可通过增加频带宽度的方法，在较低的信噪比情况下，传输信息。

2、扩展频谱换取信噪比要求的降低，正是扩频通信的重要特点，并由此为扩频通信的应用奠定了基础。

**（3）扩频通信系统的分类**

直接序列扩频（DS）、跳频扩频（FH）、跳时扩频（TH）、混合方式（以上三种基本方式的不同组合），在实际的CDMA系统中，直接序列扩频得到了广泛的认可和应用，所以在本次实验中主要研究直接序列扩频技术。

**（4）直接序列扩频**

DS-SS是直接用具有高码率的扩频码序列在发送端去扩展信号的频谱。在收端，用相同的扩频码序列去进行解扩，把展宽的扩频信号还原成原始的信息。

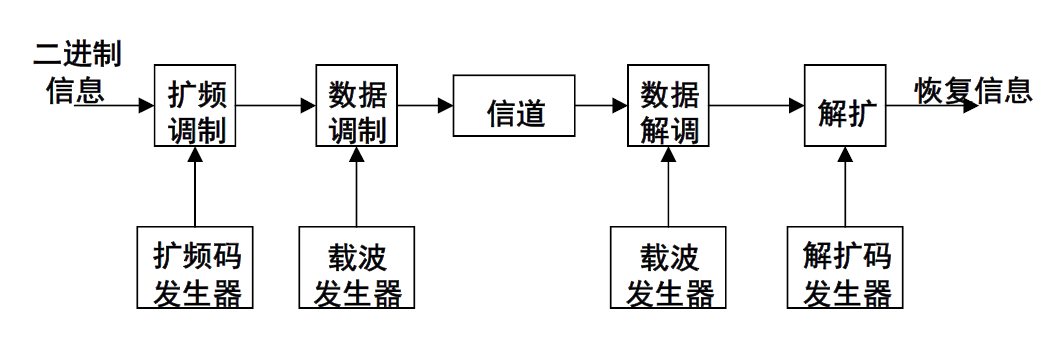


图3-1 直接序列扩频的发射机和接收机框图

**（5）扩频码序列**

在扩频系统中，信号频谱的扩展是通过扩频码实现的，扩频系统的性能与扩频码的性能有很大的关系，对扩频通常提出下列要求：易于产生；具有随机性；扩频码应该具有尽可能长的周期，使干扰者难以从扩频码的一小段中重建整个码序列；扩频码应该具有良好的自相关和互相关特性，以利于接收时的捕获和跟踪，以及多用户检测等。扩频码中应用最多的是M序列，又称最大长度序列，还有GOLD序列、WALSH码序列等，本次试验中我们主要运用了M序列和正交GOLD序列。

**实验步骤**

**CDMA仿真程序编程设计**

最外层循环的时候是信噪比，根据每种信噪比下，首先产生各个用户的发射数据，根据用户数，每一行是每个用户的发射数据，然后进行Gray编码、4-QAM调制，调制完后，对信号进行扩频，扩频部分是通过spread函数完成的，是完成扩频的功能。扩频时，看输入参数是否满足要求，满足要求后，用扩频码与每一个数据符号相乘，对原始符号进行扩展，完成扩频。扩频完成后，再通过脉冲成形滤波器计算每个用户信号功率。当用户数大于1时，所有用户数据相加。相加完成后，如果需要通过瑞利衰落信道，再让信道系数与发射数据相乘。接收端加入高斯白噪声，首先根据信噪比计算高斯白噪声标准差，根据标准差加入白噪声。如果通过瑞利衰落信道后，我们还需要进行一个信道补偿。最后通过脉冲成形滤波器进行降采样，之后进行数据解扩，解扩部分是通过despread函数完成的，解扩完成后，对信号进行4-QAM解调、Gray编码逆映射，分别对用户原始数据和解扩后数据进行比较，最后统计误比特率。以上就是完成扩频的主程序。

**实验结果：**

1. DS-CDMA通信系统仿真

function [mout] = mseq(n, taps, inidata, num)

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% n : m序列的阶数n

% taps : 反馈寄存器的连接位置

% inidata : 寄存器的初始值序列

% num : 输出的m序列的个数

% mout : 输出的m序列，如果num>1,则每一行为一个m序列

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if nargin < 4

num = 1;

end

mout = zeros(num,2^n-1);

fpos = zeros(n,1);

fpos(taps) = 1;

for ii=1:2^n-1

mout(1,ii) = inidata(n); % 寄存器的输出值

temp = mod(inidata\*fpos,2); % 计算反馈数据

inidata(2:n) = inidata(1:n-1); % 寄存器移位一次

inidata(1) = temp; % 更新第1个寄存器的值

end

if num > 1 %如果要输出多个m序列，生成其他m序列

for ii=2:num

mout(ii,:) = shift(mout(ii-1,:),1);

end

end

function [outregi] = shift(inregi,shiftr)

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% inrege : 输入序列

% shiftr : 循环右移的位数

% outregi : 输出序列

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

v = length(inregi);

outregi = inregi;

shiftr = rem(shiftr,v);

if shiftr > 0

outregi(:,1:shiftr) = inregi(:,v-shiftr+1:v); %循环移位

outregi(:,1+shiftr:v) = inregi(:,1:v-shiftr);

elseif shiftr < 0

outregi(:,1:v+shiftr) = inregi(:,1-shiftr:v);

outregi(:,v+shiftr+1:v) = inregi(:,1:-shiftr);

end

function [gout] = goldseq(m1, m2, num)

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% m1 : m序列1

% m2 : m序列2

% num : 生成的Gold序列个数

% gout : 生成的Gold序列输出

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if nargin < 3 %如果没有指定生成的Gold序列个数，默认为1

num = 1;

end

gout = zeros(num,length(m1));

for ii=1:num %根据Gold序列生成方法生成Gold序列

gout(ii,:) = xor(m1,m2);

m2 = shift(m2,1);

end

function [ber] = dscdma(user,seq)

% user: 同时进行扩频通信的用户数

% seq: 扩频码1:M-序列2:Gold序列3:正交Gold序列

% ber： 该用户数下的误码率

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 初始化部分\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

sr = 25600.0; % 符号速率

nSymbol=1000; %每种信噪比下发送的符号数

M = 4; % 4-QAM调制

br = sr \* log2(M); % 比特速率

graycode=[0 1 3 2]; % Gray编码规则

EbNo=0:2:10; % Eb/No 变化范围

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 脉冲成形滤波器参数\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

delay = 10; % 升余弦滤波器时延

Fs = 8; % 滤波器过采样数

rolloff = 0.5; % 升余弦滤波器滚降因子

rrcfilter = rcosine(1,Fs,'fir/sqrt',rolloff,delay);

%设计根升余弦滤波器

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 扩频码产生参数\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% user = user1; % 用户数

stage = 3; % m序列的阶数

ptap1 = [1 3]; % m序列1的寄存器连接方式

ptap2 = [2 3]; % m序列2的寄存器连接方式

regi1 = [1 1 1 ]; % m序列1的寄存器初始值

regi2 = [1 1 1]; % m序列2的寄存器初始值

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 扩频码的生成\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

switch seq

case 1 % M-序列

code = mseq(stage,ptap1,regi1,user);

case 2 % Gold 序列

m1 = mseq(stage,ptap1,regi1);

m2 = mseq(stage,ptap2,regi2);

code = goldseq(m1,m2,user);

case 3 % 正交Gold 序列

m1 = mseq(stage,ptap1,regi1);

m2 = mseq(stage,ptap2,regi2);

code = [goldseq(m1,m2,user),zeros(user,1)];

end

code = code \* 2 - 1;

clen = length(code);

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 衰落信道参数\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ts = 1 / Fs / sr/ clen; % 信道采样时间间隔

t=(0:nSymbol\*Fs\*clen-1+2\*delay\*Fs)\*ts; % 每种信噪比下的符号传输时间

%fd = 160; % 多普勒频移[Hz]

%h=rayleigh(fd,t);

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 仿真开始\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

for indx=1:length(EbNo)

indx

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 发射端\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

data = randsrc(user,nSymbol,[0 :3]) ; % 产生各个用户的发射数据

data1=graycode(data+1); % Gray编码

data1 = qammod(data1,M); % 4-QAM 调制

[out] = spread(data1,code); % 扩频

out1=rcosflt(out.',sr,Fs\*sr,'filter',rrcfilter);

% 通过脉冲成形滤波器

spow = sum(abs((out1)).^2) / nSymbol; % 计算每个用户信号功率

if user > 1 % 用户数大于1时，所有用户数据相加

out1=sum(out1.');

else

out1=out1.';

end

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 通过瑞利衰落信道\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% out1=h.\*out1;

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 接收端\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

sigma = sqrt(0.5 \* spow \* sr / br \* 10^(-EbNo(indx)/10));

% 根据信噪比计算高斯白噪声方差

y=[];

for ii=1:user

y(ii,:)=out1+sigma(ii).\*(randn(1,length(out1))+j\*randn(1,length(out1))); % 加入高斯白噪声（AWGN）

%y(ii,:)=y(ii,:)./h;

% 假设理想信道估计

end

y=rcosflt(y.',sr,Fs\*sr,'Fs/filter',rrcfilter);

% 通过脉冲成形滤波器进行滤波

y=downsample(y,Fs); % 降采样

for ii=1:user

y1(:,ii)=y(2\*delay+1:end-2\*delay,ii);

end

yd = despread(y1.',code); % 数据解扩

demodata = qamdemod(yd,M); % 4-QAM 解调

demodata=graycode(demodata+1); % Gray编码逆映射

[err,ber(indx)]=biterr(data,demodata,log2(M)); % 统计误比特率

end

function [out] = spread(data, code)

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% data : 输入数据序列

% code : 扩频码序列

% out : 扩频后的输出数据序列

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

switch nargin

case { 0 , 1 } %如果输入参数个数不对，提示错误

error('缺少输入参数');

end

[hn,vn] = size(data);

[hc,vc] = size(code);

if hn > hc %如果扩频码数小于输入的待扩频的数据序列，提示错误

error('缺少扩频码序列');

end

out = zeros(hn,vn\*vc);

for ii=1:hn

out(ii,:) = reshape(code(ii,:).'\*data(ii,:),1,vn\*vc);

end

function out = despread(data, code)

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% data : 输入数据序列

% code : 解扩使用的扩频码序列

% out : 解扩后的输出数据序列

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

switch nargin %如果输入参数个数不对，提示错误

case { 0 , 1 }

error('缺少输入参数');

end

[hn,vn] = size(data);

[hc,vc] = size(code);

out = zeros(hc,vn/vc);

for ii=1:hc

xx=reshape(data(ii,:),vc,vn/vc);

out(ii,:)= code(ii,:)\*xx/vc;

end

%主程序1：m-序列DS-CDMA在AWGN信道下的性能仿真

clear all

user=[1 4 7 ];

seq=1;

for index=1:length(user)

ber(index,:)=dscdma(user(index),seq);

end

EbNo=0:2:10;

semilogy(EbNo,ber(1,:),'-kx',EbNo,ber(2,:),'-ko',EbNo,ber(3,:),'-k\*');

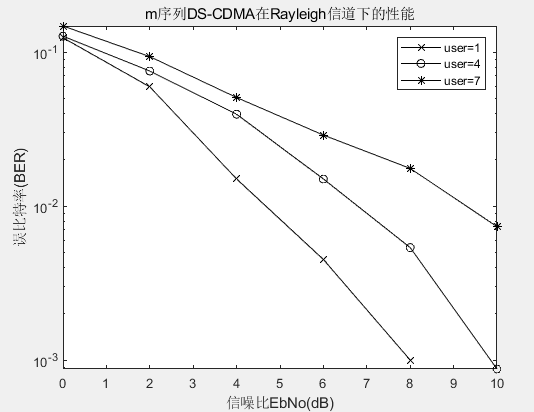
legend('user=1','user=4','user=7','user=9')

title('m序列DS-CDMA在Rayleigh信道下的性能')

xlabel('信噪比EbNo(dB)')

ylabel('误比特率(BER)')

运行结果：



%主程序2：正交Gold序列DS-CDMA在Rayleigh信道下的性能：

user=[1 4 7 9];

seq=3;

for index=1:length(user)

ber(index,:)=dscdma(user(index),seq);

end

EbNo=0:2:10;

semilogy(EbNo,ber(1,:),'-kx',EbNo,ber(2,:),'-ko',EbNo,ber(3,:),'-k\*',EbNo,ber(4,:),'-k.');

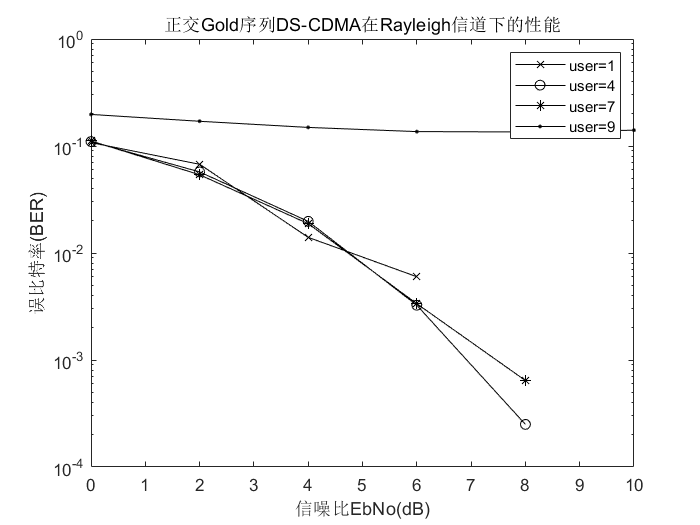
legend('user=1','user=4','user=7','user=9')

title('正交Gold序列DS-CDMA在Rayleigh信道下的性能')

xlabel('信噪比EbNo(dB)')

ylabel('误比特率(BER)')

运行结果：

****

**思考题**

1. 试说明CDMA系统的特点。

答：①抗干扰能力强。这是扩频通信的基本特点，是所有通信方式无法比拟的。

②宽带传输，抗衰落能力强。

③由于采用宽带传输，在信道中传输的有用信号的功率比干扰信号的功率低得多，因此信号好像隐蔽在噪声中;即功率话密度比较低，有利于信号隐蔽。

④利用扩频码的相关性来获取用户的信息，抗截获的能力强。

⑤多个用户同时接收，同时发送。

1. TDMA、CDMA、FDMA三种多址接入技术有什么不同？

答：多址技术：多用户共用无线资源的方式。

①FDMA（频分多址）：将总频段划分为不同的小频道分配给不同的用户。

优点：简单，易实现，技术成熟

缺点：频率利用率低，容量小

②TDMA（时分多址）：将时间段划分为小时隙，分配给不同的用户。（GSM）

优点：容量大，频率利用率高

缺点：技术复杂，严格的同步要求。

③CDMA（码分多址）：不同的用户采用各自独立的编码序列。

优点：容量最大，频率利用率高，质量好。背景噪声受限的系统，软容量。

缺点：起步太晚，用户群体少。

**扩展题**

1. 试对OFDMA通信系统进行建模仿真。

答：OFDMA通信系统仿真如下

%OFDMA Channel Estimation Based on Comb Pilot

%IFFT\_bin\_length: IFFT和FFT的点数

%carrier\_count: 子载波个数

%bits\_per\_symbol: 每符号上的比特数

%symbols\_per\_carrier: 每桢的OFDM符号数

%Ｘ：欲发送的二进制比特流

clear all;

clc;

IFFT\_bin\_length=512;

carrier\_count=270;

bits\_per\_symbol=2;

symbols\_per\_carrier=12;

LI=7 ; %导频之间的间隔

Np=ceil(carrier\_count/LI)+1;%导频数 %加1的原因：使最后一列也是导频

N\_number=carrier\_count\*symbols\_per\_carrier\*bits\_per\_symbol;

carriers=1:carrier\_count+Np;

GI=8; % guard interval length

N\_snr=20; % 每比特信噪比

snr=8; %信噪比间隔

%------------------------------------------------------------

% vector initialization

X=zeros(1,N\_number);

X1=[];

X2=[];

X3=[];

X4=[];

X5=[];

X6=[];

X7=[];

Y1=[];

Y2=[];

Y3=[];

Y4=[];

Y5=[];

Y6=[];

Y7=[];

XX=zeros(1,N\_number);

dif\_bit=zeros(1,N\_number);

dif\_bit1=zeros(1,N\_number);

dif\_bit2=zeros(1,N\_number);

dif\_bit3=zeros(1,N\_number);

X=randi([0,1],1,N\_number);%产生二进制随即序列（非0即1）

%--------------------------------------------------------

%QPSK调制：(1 1)->pi/4;(0 1)->3\*pi/4;(0 0)->-3\*pi/4;(1,0)->-pi/4;

s=(X.\*2-1)/sqrt(2);

sreal=s(1:2:N\_number);

simage=s(2:2:N\_number);

X1=sreal+j.\*simage;

%---------------------------------------------------------

%产生随机导频信号

%--------------------------------------------------------

train\_sym=randi([0,1],1,2\*symbols\_per\_carrier);

t=(train\_sym.\*2-1)/sqrt(2);

treal=t(1:2:2\*symbols\_per\_carrier);

timage=t(2:2:2\*symbols\_per\_carrier);

training\_symbols1=treal+j.\*timage;

training\_symbols2=training\_symbols1.';

training\_symbols=repmat(training\_symbols2,1,Np);

%disp(training\_symbols)

pilot=1:LI+1:carrier\_count+Np;

if length(pilot)~=Np

pilot=[pilot,carrier\_count+Np];

end

%--------------------------------------------------------

%串并转换

X2=reshape(X1,carrier\_count,symbols\_per\_carrier).';

%---------------------------------------------------------

%插入导频

signal=1:carrier\_count+Np;

signal(pilot)=[];

X3(:,pilot)=training\_symbols;

X3(:,signal)=X2;

%X3=cat(1,training\_symbols,X2);

IFFT\_modulation=zeros(symbols\_per\_carrier,IFFT\_bin\_length);

IFFT\_modulation(:,carriers)=X3;

%IFFT\_modulation(:,conjugate\_carriers)=conj(X3);

X4=ifft(IFFT\_modulation,IFFT\_bin\_length,2);

%X5=X4.';

%加保护间隔（循环前缀）

for k=1:symbols\_per\_carrier;

for i=1:IFFT\_bin\_length;

X6(k,i+GI)=X4(k,i);

end

for i=1:GI;

X6(k,i)=X4(k,i+IFFT\_bin\_length-GI);

end

end

%---------------------------------------------------------

%并串转换

X7=reshape(X6.',1,symbols\_per\_carrier\*(IFFT\_bin\_length+GI));

%---------------------------------------------------------

%信道模型:带多普勒频移的瑞利衰落信道

fd=100; %多普勒频移

r=6; %多径数

a=[0.123 0.3 0.4 0.5 0.7 0.8]; %多径的幅度

d=[2 3 4 5 9 13]; %各径的延迟

T=1; %系统采样周期

th=[90 0 72 144 216 288]\*pi./180;%相移

h=zeros(1,carrier\_count);

hh=[];

for k=1:r

%deta=[zeros(1,d(k)-1),1,zeros(1,carrier\_count-d(k))];

h1=a(k)\*exp(j\*((2\*pi\*T\*fd\*d(k)/carrier\_count)));

%h1=a(k)\*exp(j\*((2\*pi\*T\*fd\*d(k)/carrier\_count)));

hh=[hh,h1];

end

h(d+1)=hh;

%noise=randn(1,length(X7))+j.\*randn(1,length(X7));

%--------------------------------------------------------

channel1=zeros(size(X7));

channel1(1+d(1):length(X7))=hh(1)\*X7(1:length(X7)-d(1));

channel2=zeros(size(X7));

channel2(1+d(2):length(X7))=hh(2)\*X7(1:length(X7)-d(2));

channel3=zeros(size(X7));

channel3(1+d(3):length(X7))=hh(3)\*X7(1:length(X7)-d(3));

channel4=zeros(size(X7));

channel4(1+d(4):length(X7))=hh(4)\*X7(1:length(X7)-d(4));

channel5=zeros(size(X7));

channel5(1+d(5):length(X7))=hh(5)\*X7(1:length(X7)-d(5));

channel6=zeros(size(X7));

channel6(1+d(6):length(X7))=hh(6)\*X7(1:length(X7)-d(6));

%---------------------------------------------------------------

Tx\_data=X7+channel1+channel2+channel3+channel4;

%---------------------------------------------------------------

%---------------------------------------------------------------

%----------------------------------------------------------------

%加高斯白噪声

Error\_ber=[];%误比特率

Error\_ber1=[];

Error\_ber2=[];%误比特率

Error\_ber3=[];

%Error\_ser=[];%误符号率

for snr\_db=0:snr:N\_snr

code\_power=0;

code\_power=[norm(Tx\_data)]^2/(length(Tx\_data));%信号的符号功率

%bit\_power=var(Tx\_data);

bit\_power=code\_power/bits\_per\_symbol;%比特功率

noise\_power=10\*log10((bit\_power/(10^(snr\_db/10))));%噪声功率

noise=wgn(1,length(Tx\_data),noise\_power,'complex');%产生GAUSS白噪声信号

Y7=Tx\_data+noise;

%-------------------------------------------------------

%串并变换

Y6=reshape(Y7,IFFT\_bin\_length+GI,symbols\_per\_carrier).';

%去保护间隔

for k=1:symbols\_per\_carrier;

for i=1:IFFT\_bin\_length;

Y5(k,i)=Y6(k,i+GI);

end

end

Y4=fft(Y5,IFFT\_bin\_length,2);

Y3=Y4(:,carriers);

%-------------------------------------------------------------

%LS信道估计

H=[];

Y2=Y3(:,signal);

Rx\_training\_symbols=Y3(:,pilot);

Rx\_training\_symbols0=reshape(Rx\_training\_symbols,symbols\_per\_carrier\*Np,1);

training\_symbol0=reshape(training\_symbols,1,symbols\_per\_carrier\*Np);

training\_symbol1=diag(training\_symbol0);

%disp(training\_symbols)

training\_symbol2=inv(training\_symbol1);

Hls=training\_symbol2\*Rx\_training\_symbols0;

Hls1=reshape(Hls,symbols\_per\_carrier,Np);

HLs=[];

HLs1=[];

if ceil(carrier\_count/LI)==carrier\_count/LI

for k=1:Np-1

HLs2=[];

for t=1:LI

HLs1(:,1)=(Hls1(:,k+1)-Hls1(:,k))\*(t-1)./LI+Hls1(:,k);

HLs2=[HLs2,HLs1];

end

HLs=[HLs,HLs2];

end

else

for k=1:Np-2

HLs2=[];

for t=1:LI

HLs1(:,1)=(Hls1(:,k+1)-Hls1(:,k))\*(t-1)./LI+Hls1(:,k);

HLs2=[HLs2,HLs1];

end

HLs=[HLs,HLs2];

end

HLs3=[];

for t=1:mod(carrier\_count,LI)

HLs1(:,1)=(Hls1(:,Np)-Hls1(:,Np-1))\*(t-1)./LI+Hls1(:,Np-1);

HLs3=[HLs3,HLs1];

end;

HLs=[HLs,HLs3];

end

%Hls1=Hls.';

%H=repmat(Hls1,symbols\_per\_carrier,1);%将导频扩展成symbols\_per\_carrier\*carrier\_count矩阵

Y1=Y2./HLs;

%-------------------------------------------------------------

%并串变换

YY=reshape(Y2.',1,N\_number/bits\_per\_symbol);

YY1=reshape(Y1.',1,N\_number/bits\_per\_symbol);

%------------------------------------------------------------

%QPSK解调

r01=zeros(1,N\_number);

r11=zeros(1,N\_number);%解调序列矩阵初始化

for i=1:1:N\_number/2

A = real(YY(i));

B = imag(YY(i))

if (A>0)

if(B>0)

r01(2\*i-1) = 1/sqrt(2);

r01(2\*i) = 1/sqrt(2);

end

end

if (A<0)

if(B>0)

r01(2\*i-1) = -1/sqrt(2);

r01(2\*i) = 1/sqrt(2);

end

end

if (A>0)

if(B<0)

r01(2\*i-1) = 1/sqrt(2);

r01(2\*i) = -1/sqrt(2);

end

end

if (A<0)

if(B<0)

r01(2\*i-1) = -1/sqrt(2);

r01(2\*i) = -1/sqrt(2);

end

end

end

for i=1:1:N\_number/2

A = real(YY1(i));

B = imag(YY1(i))

if (A>0)

if(B>0)

r1(2\*i-1) = 1/sqrt(2);

r11(2\*i) = 1/sqrt(2);

end

end

if (A<0)

if(B>0)

r11(2\*i-1) = -1/sqrt(2);

r11(2\*i) = 1/sqrt(2);

end

end

if (A>0)

if(B<0)

r11(2\*i-1) = 1/sqrt(2);

r11(2\*i) = -1/sqrt(2);

end

end

if (A<0)

if(B<0)

r11(2\*i-1) = -1/sqrt(2);

r11(2\*i) = -1/sqrt(2);

end

end

end

r01

%-------------------------------------------------------------

%计算在不同信噪比下的误比特率并作图

dif\_bit=s-r01;

dif\_bit1=s-r11;

ber\_snr=0; %纪录误比特数

for k=1:N\_number;

if dif\_bit(k)~=0;

ber\_snr=ber\_snr+1;

end

end;

ber\_snr1=0; %纪录误比特数

for k=1:N\_number;

if dif\_bit1(k)~=0;

ber\_snr1=ber\_snr1+1;

end

end

Error\_ber=[Error\_ber,ber\_snr];

Error\_ber1=[Error\_ber1,ber\_snr1];

end

BER=zeros(1,length(0:snr:N\_snr));

BER1=zeros(1,length(0:snr:N\_snr));

BER=Error\_ber./N\_number;

BER1=Error\_ber1./N\_number;

%-------------------------------------------------------------

%-------------------------------------------------------------

i=0:snr:N\_snr;

semilogy(i,BER,'-\*r');

hold on;

semilogy(i,BER1,'-ob');

hold on;

grid on;

legend('No Channel Estimation','LS Channel Estimation');

hold off

运行结果：

