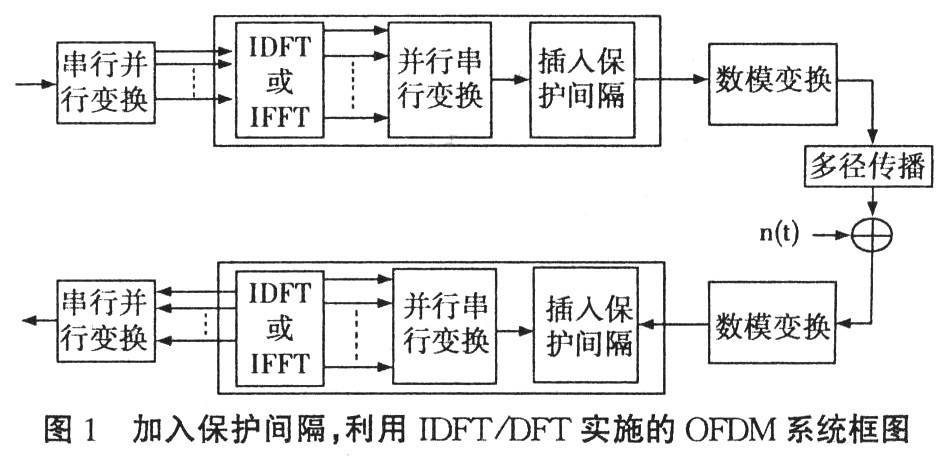
无线大作业实验报告

一、实验目的  
1、了解OFDM技术的原理与特点；  
2、使用MATLAB语言仿真实现OFDM基带信号在频率选择性衰落信道条件下的发送与接收。  
二、实验原理  
1、OFDM调制基本原理  
OFDM(Orthogonal Frequency Division Multiplexing)即正交频分复用技术，实际上OFDM是MCM(Multi Carrier Modulation)，多载波调制的一种。

OFDM技术由MCM（Multi-Carrier Modulation，多载波调制）发展而来。OFDM技术是多载波传输方案的实现方式之一，它的调制和解调是分别基于IFFT和FFT来实现的，是实现复杂度最低、应用最广的一种多载波传输方案。

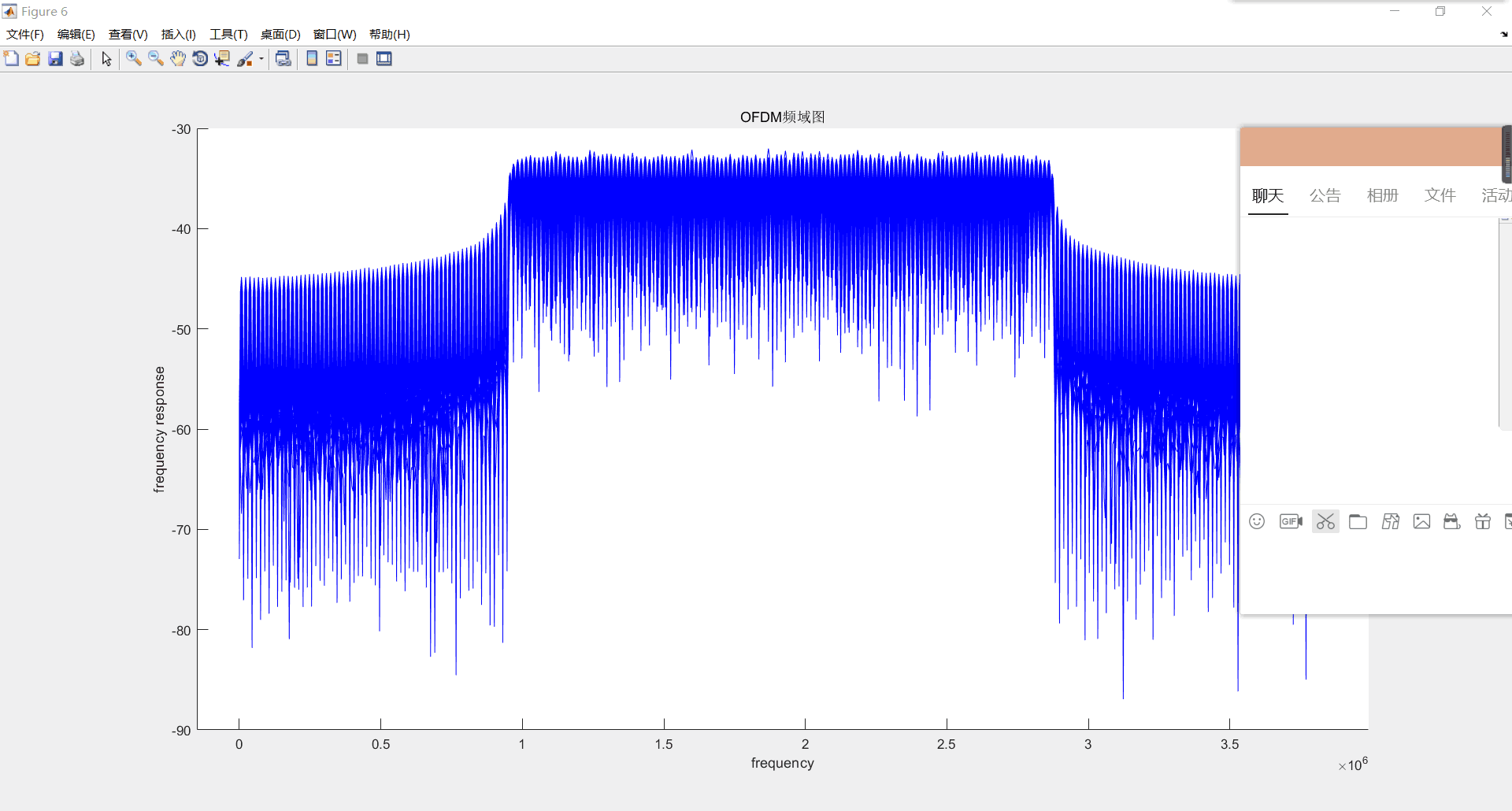
在通信系统中，信道所能提供的带宽通常比传送一路信号所需的带宽要宽得多。如果一个信道只传送一路信号是非常浪费的，为了能够充分利用信道的带宽，就可以采用频分复用的方法。

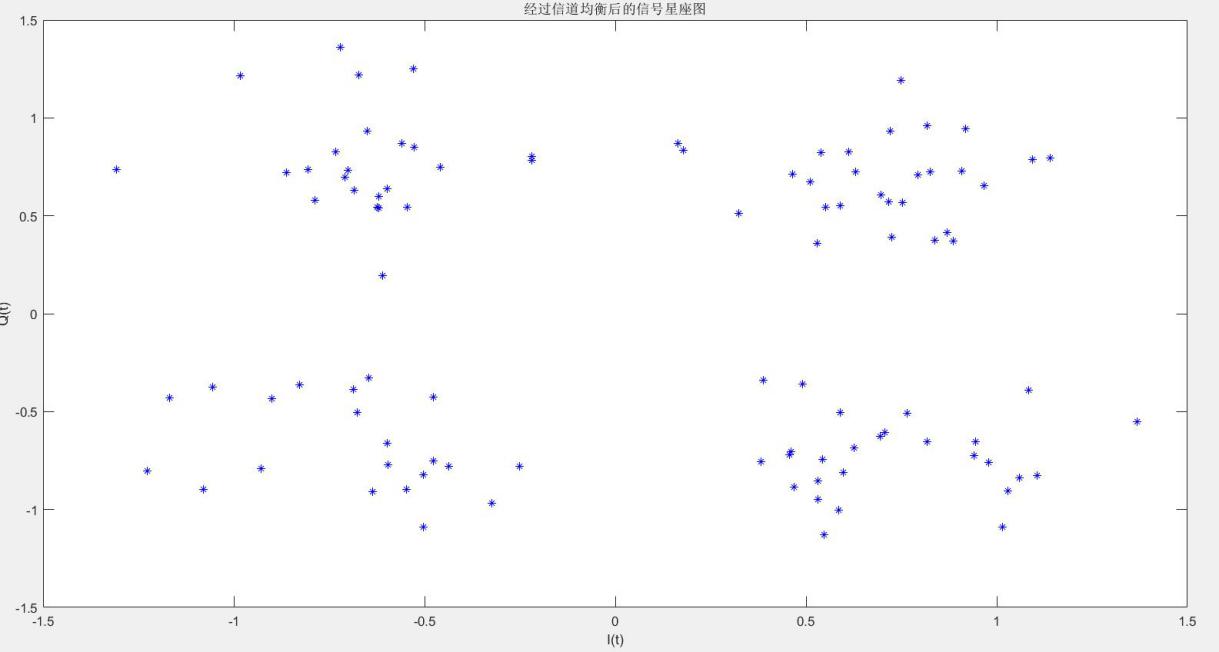
OFDM主要思想是：将信道分成若干正交子信道，将高速数据信号转换成并行的低速子数据流，调制到在每个子信道上进行传输。正交信号可以通过在接收端采用相关技术来分开，这样可以减少子信道之间的相互干扰(ISI) 。每个子信道上的信号带宽小于信道的相关带宽，因此每个子信道上可以看成平坦性衰落，从而可以消除码间串扰，而且由于每个子信道的带宽仅仅是原信道带宽的一小部分，信道均衡变得相对容易。

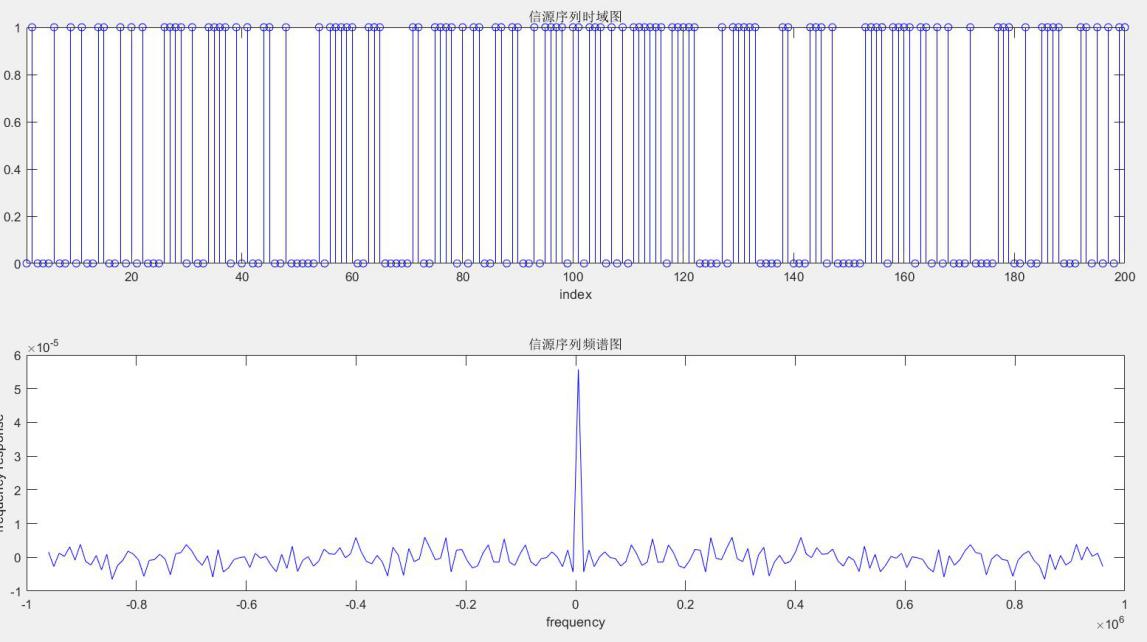
2、OFDM系统的实现模型  
利用离散反傅里叶变换(IDFT)或快速反傅里叶变换(IFFT)实现的OFDM系统如图1所示。输入已经过调制(符号匹配)的复信号经过串P并变换后,进行IDFT或IFFT和并/串变换,然后插入保护间隔,再经过数/模变换后形成OFDM调制后的信号s(t)。该信号经过信道后,接收到的信号r(t)经过模P数变换,去掉保护间隔以恢复子载波之间的正交性,再经过串/并变换和DFT或FFT后,恢复出OFDM的调制信号,再经过并P串变换后还原出输入的符号。  
从OFDM系统的基本结构可看出,一对离散傅里叶变换是它的核心,它使各子载波相互正交。设OFDM信号发射周期为[0,T],在这个周期内并行传输的N个符号为(C00，C10,...,CN),其中Cn为一般复数,并对应调制星座图中的某一矢量。比如C00\*a(0)+j\*b(0),a(0)和b(0)分别为所要传输的并行信号,若将其合为一个复数信号,很多个这样的复数信号采用快速傅里叶变换,同时也实现对正交载波的调制, 这就大大加快了信号的处理调制速度(在接收端解调也同样)。由于实际发送的是复数的实部,因此在IFFT的算法中会将处理后的信号都映射为实数,然后经过射频调制发出。  
3、OFDM系统的保护间隔(GI)和循环前缀(CP)

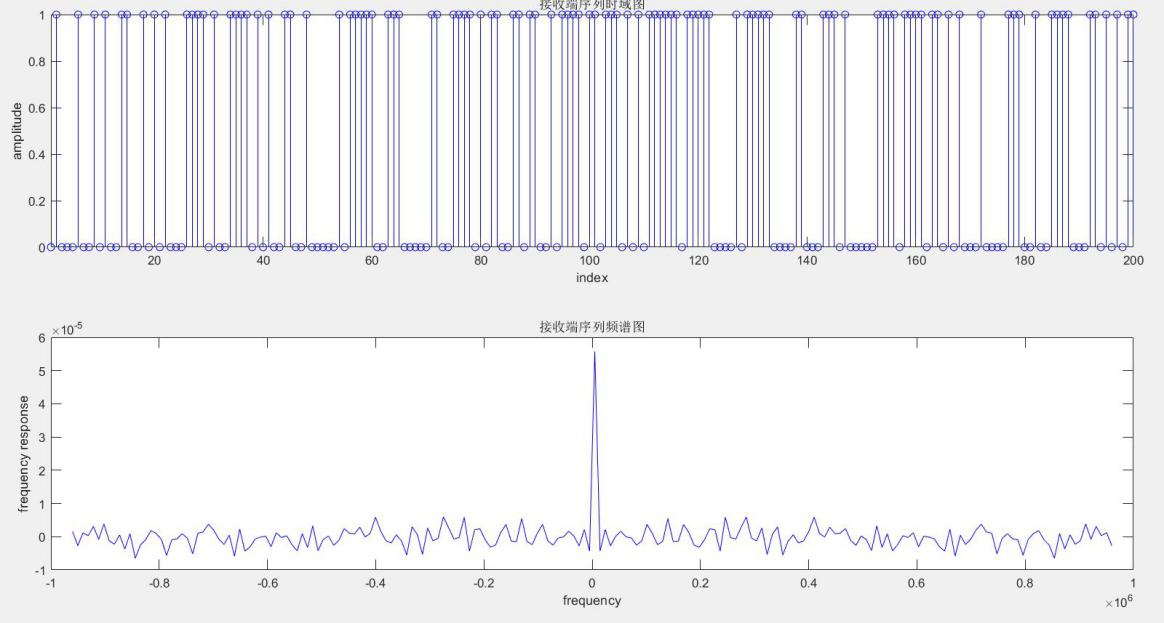
1、保护间隔  
为了保持子载波之间的正交性,在发送之前就要在每个OFDM 符号之间插入保护间隔,该保护间隔的长度TG一般要大于无线信道的最大时延扩展,才会使一个符号的多径分量不会对下一个符号造成干扰,从而有效消除码间干扰(ISI)。如果在这段保护间隔内,不插入任何信号,仅把它作为一段空闲的传输时段,那么由于多径传播的影响,就会产生子信道间的干扰(ICI),这样还是会破坏子载波之间的正交性,使得各子载波之间产生干扰。  
2、循环前缀  
为了消除多径传播造成的ICI,一种有效的方法是将原来宽度为T的OFDM符号进行周期扩展,用扩展信号来填充保护间隔,经扩充的保护间隔内的信号称为循环前缀,循环前缀中的信号与OFDM符号尾部宽度为TG的部分相同。在一个OFDM符号中,循环前缀部分携带任何信息,它和信息一起传送会带来功率和信息速率的损失,但是由于保护间隔的插入可以消除多径传播引起的ICI 影响,能更好地体现多载波传输的优越性,因此上述的损失是值得的。  
三、实验内容  
1、结合理论课讲解基于Matlab仿真OFDM信号，绘制OFDM符号星座图，时域、频域曲线；  
2、绘制发送端、接收端低通滤波器的幅频特性；  
3、分析AWGN信道条件下OFDM系统的误码率性能；

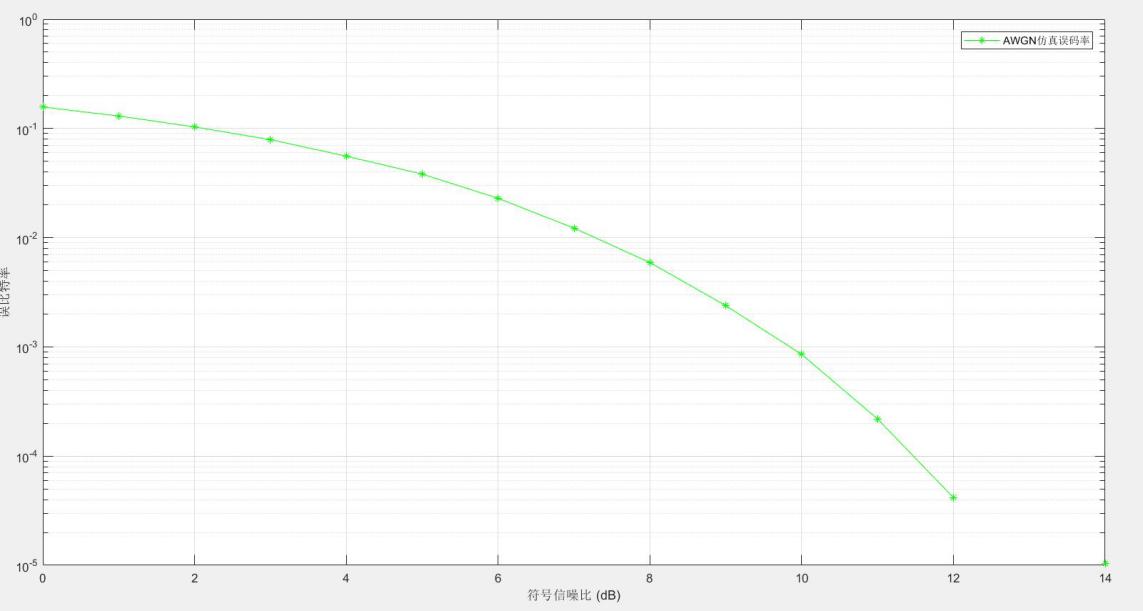
4、用简要的文字描述实验感受。  
四、实验仿真  
1、OFDM时域频域波形

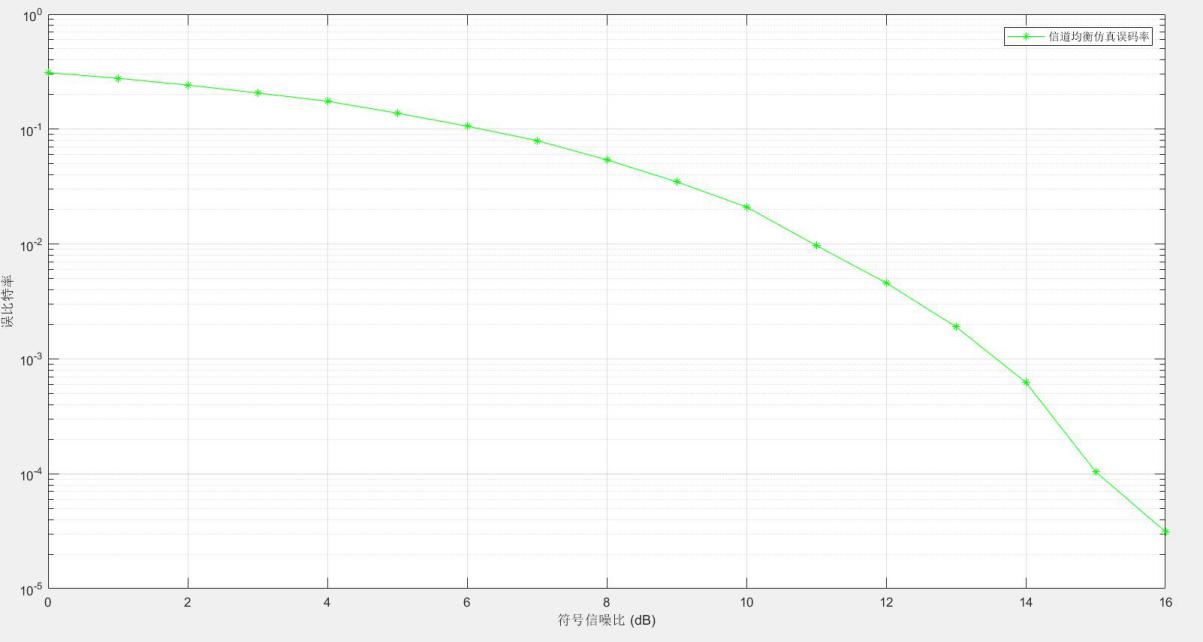
  
2、OFDM符号星座图

  
3、输入输出



  
4、AWGN信道条件下OFDM系统的误码率



1. 过多经信道叠加高斯信道条件下的OFDM系统的误码率  
   五、实验思考与结论  
   从OFDM系统的原理和仿真过程可以看出，OFDM系统频带利用率高，因为OFDM允许重叠的正交子载波作为子信道，而不是传统的利用保护频带分离子信道。同时高速数据流通过串并转换，能使得每个子载波上的信号时间比相应同速率的单载波系统上的信号时间长，采用循环前缀的方法，有效减少了ISI。  
   但是OFDM系统对频偏和相位噪声敏感，因为OFDM区分各个子载波的方法是利用各个子载波之间的正交性，而频偏和相位噪声使正交性恶化，所以会产生ICI。
2. matlab仿真程序

clc

clear

%q1:ifft点数难道不是应该等于子载波数吗？子载波数与ifft点数的关系？

%a:ifft点数等于子载波数

%q2：对矩阵进行fft？

%a:y可以是一向量或矩阵，若y为向量，则Y是y的FFT，并且与y具有相同的长度。若y为一矩阵，则Y是对矩阵的每一列向量进行FFT。

%image\_data\_generator();

%数据子载波数128

%fft点数512

snr = 0:1:20%dB 高斯信道的SNR

cdata\_number = 128;

fft\_n = 128;

%OFDM码元个数 500

%生成数据随机序列

cdata\_length = 500;

cdata = rand\_bit(cdata\_number\*cdata\_length\*2);

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

figure;

subplot(2,1,1);

time(cdata(1,1:200),"信源序列时域图",'B');

subplot(2,1,2);

frequency\_response(cdata(1,1:200),"信源序列频谱图",'B');

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

%导频长度256

%插入导频间隔为5

%生成随机的的导频序列

pilot\_inter = 5;

pilot\_length = 256;

pilot\_bit = rand\_bit(pilot\_length);

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

figure;

subplot(2,1,1);

time(pilot\_bit(1,1:200),"导频序列时域图",'B');

subplot(2,1,2);

frequency\_response(pilot\_bit(1,1:200),"导频序列频谱图",'B');

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

%进行qpsk调制 输出复数矩阵

qpsk\_bit = modulator\_QPSK(cdata);

scatterplot(qpsk\_bit);

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

figure;

subplot(2,1,1);

time(real(qpsk\_bit(1:100)),"信源序列QPSK",'B');

hold on;

time(imag(qpsk\_bit(1:100)),"信源序列QPSK",'R');

subplot(2,1,2);

frequency\_response(qpsk\_bit(1:100),"信源序列QPSK频谱图",'B');

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

%串并转换 128\*1000

paradata = reshape(qpsk\_bit,cdata\_number,cdata\_length);

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

%插入导频

inserted\_bit = insert\_pilot(paradata,pilot\_bit,pilot\_inter);

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

%逆傅里叶变换

ifft\_data = ifft(inserted\_bit,fft\_n,1)\*sqrt(fft\_n);

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

figure;

subplot(2,1,1);

time(real(ifft\_data(1:100)),"逆傅里叶变换时域图",'B');

hold on;

time(imag(ifft\_data(1:100)),"逆傅里叶变换时域图",'R');

subplot(2,1,2);

frequency\_response(ifft\_data(1:100),"逆傅里叶变换频域图",'B');

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

%生成OFDM频谱图

figure;

for i = 1:600

hold on;

frequency\_response\_deshift(inserted\_bit(:,i)',"OFDM频域图",'B');

end

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

%插入循环前缀

symbol\_cp\_pro = cyclic\_prefix(ifft\_data);

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

[row,column] = size(symbol\_cp\_pro);

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

%经过高斯信道

awgn\_err\_plot(cdata,symbol\_cp\_pro);

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

%ls多径信道

ERR1 = ls\_err\_plot(cdata,symbol\_cp\_pro,pilot\_bit);

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

%并串转换

symbol\_cp = reshape(symbol\_cp\_pro,1,[]);

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

figure;

time(real(symbol\_cp(1:160)),"加入CP时域图",'B');

hold on;

time(imag(symbol\_cp(1:160)),"加入CP时域图",'R');

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

%信道

%ts采样间隔

%fdDoppler频偏，HZ

%tau多径延时

%pdb各径功率

%---------------------------------------------多径信道---------------------

fs = 15000;

ts = 1/fs;

fd = 0;

tau = [0,50,120,200,230,500,1600,2300,5000]/(10^9);

pdb = [-1.0,-1.0,-1.0,0,0,0,-3.0,-5.0,-7.0];

chan = rayleighchan(ts,fd,tau,pdb);

chan.ResetBeforeFiltering = 0;

chan\_symbol\_s = filter(chan,symbol\_cp);

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

figure;

subplot(2,1,1);

time(real(chan\_symbol\_s(1:100)),"过多经信道时域图",'B');

hold on;

time(imag(chan\_symbol\_s(1:100)),"过多经信道时域图",'R');

subplot(2,1,2);

frequency\_response(chan\_symbol\_s(1:100),"过多经信道频谱图",'B');

%--------------------------------------------多径信道-----------------------

%------------------------------------------叠加高斯信道---------------------

SNR = 20;%单位dB

chan\_symbol\_s = awgn(chan\_symbol\_s,SNR);

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

figure;

subplot(2,1,1);

time(real(chan\_symbol\_s(1:100)),"过多经信道叠加高斯信道时域图",'B');

hold on;

time(imag(chan\_symbol\_s(1:100)),"过多经信道叠加高斯信道时域图",'R');

subplot(2,1,2);

frequency\_response(chan\_symbol\_s(1:100),"过多经信道叠加高斯信道频谱图",'B');

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

figure;

for i=1:100

plot(chan\_symbol\_s(i),'b\*');

title('QPSK调制经过高斯信道后的信号星座图');

xlabel('I(t)');

ylabel('Q(t)');

hold on;

end

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

%串并转换

chan\_symbol\_p = reshape(chan\_symbol\_s,row,length(chan\_symbol\_s)/row);

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

%去循环码

re\_symbol\_cp = re\_cyclic\_prefix(chan\_symbol\_p);

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

%fft

fft\_symbol = fft(re\_symbol\_cp,fft\_n,1)./sqrt(fft\_n);

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

figure;

subplot(2,1,1);

time(real(fft\_symbol(:,1)'),"傅里叶变换时域图",'B');

hold on;

time(imag(fft\_symbol(:,1)'),"傅里叶变换时域图",'R');

subplot(2,1,2);

frequency\_response(fft\_symbol(:,1)',"傅里叶变换频谱图",'B');

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

%去除导频

[output,pilot\_symbol] = re\_insert\_pilot(fft\_symbol,pilot\_inter);

% %----------------导频位置信道响应LS估计-----------------------------------

pilot\_qpsk = modulator\_QPSK(pilot\_bit);

pilot\_patt=repmat(pilot\_qpsk',1,100);

pilot\_esti=pilot\_symbol./pilot\_patt; % Y = H\*X + S

[r,c] = size(output);

h = zeros(r,c);

for i = 1:100

h(:,(1+(i-1)\*5):i\*5) = repmat(pilot\_esti(:,i),1,pilot\_inter);

end

output = output./h;

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

figure;

subplot(2,1,1);

time(real(output(1:100)),"接收序列QPSK",'B');

hold on;

time(imag(output(1:100)),"接收序列QPSK",'R');

subplot(2,1,2);

frequency\_response(output(1:100),"接收序列QPSK频谱图",'B');

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

figure;

for i=1:100

plot(output(i),'b\*');

title('经过信道均衡后的信号星座图');

xlabel('I(t)');

ylabel('Q(t)');

hold on;

end

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

%并串转换

output = reshape(output,1,[]);

re\_modulated =re\_modulatorQPSK(output);

figure;

subplot(2,1,1);

time(re\_modulated(1,1:200),"接收端序列时域图",'B');

subplot(2,1,2);

frequency\_response(re\_modulated(1,1:200),"接收端序列频谱图",'B');

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

% err = 0;

% for n = 1:length(cdata)

% if cdata(n) ~= re\_modulated(n)

% err = err + 1;

% end

% end

% err = err/length(cdata);

% n = 1:20;

% ERR = repmat(err,1,20);

% figure;

% semilogy(n,ERR,'-\*b')

% legend('OFDM仿真误码率')

% xlabel('符号信噪比 (dB)');

% ylabel('误符号率/误比特率');

%输入随机序列

function sr = rand\_bit(L)

sr = round(rand(1,L));

end

%输入信号进行QPSK调制

function pilot\_modulated = modulator\_QPSK(pilot)

length\_pilot = length(pilot);

ip = zeros(1,floor(length\_pilot/2));

for ii = 1:floor(length\_pilot/2)

ip(ii) = 1/sqrt(2)\*1i\*(2\*pilot(2\*ii) - 1) + 1/sqrt(2)\*(2\*pilot(1 + 2\*(ii - 1)) - 1);

end

pilot\_modulated = ip;

end

%QPSK解调

function output =re\_modulatorQPSK(input)%序列

length\_input = length(input);

output = zeros(1,2\*length\_input);

for ii = 1:length\_input

r = real(input(ii));

i = imag(input(ii));

if r>0&&i>0

output(2\*ii-1) = 1;

output(2\*ii) = 1;

end

if r>0&&i<0

output(2\*ii-1) = 1;

output(2\*ii) = 0;

end

if r<0&&i<0

output(2\*ii-1) = 0;

output(2\*ii) = 0;

end

if r<0&&i>0

output(2\*ii-1) = 0;

output(2\*ii) = 1;

end

end

end

%块状导频

%input串并转换后的传输矩阵

%pilot插入的导频序列

%pilot\_inter插入导频间隔

%output插入导频后的矩阵

function output = insert\_pilot(input,pilot\_bit,pilot\_inter)

pilot\_symbol = modulator\_QPSK(pilot\_bit);%1 128

pilot\_symbol = pilot\_symbol';%128 1

%pilot\_symbol\_tem = reshape(pilot\_symbol,128,1);

%pilot\_seq = ifft(pilot\_symbol\_tem,128)\*sqrt(128);

[N,NL] = size(input);%返回行数和列数 128 500

output = zeros(N,(NL + fix(NL/pilot\_inter)));

count = 0;

i = 1;

while i < (NL+fix(NL/pilot\_inter))

output(:,i) = pilot\_symbol;

count =count + 1;

if count \* pilot\_inter <= NL

output(:,(i + 1):(i + pilot\_inter)) = input(:,((count - 1)\*pilot\_inter + 1):count \* pilot\_inter);

else

output(:,(i + 1):(i + pilot\_inter + NL - count \* pilot\_inter)) = input(:,((count - 1)\*pilot\_inter + 1):NL);

end

i = i + pilot\_inter + 1;

end

end

%取出导频

function [output,pilot\_symbol] = re\_insert\_pilot(input,pilot\_inter)

[row,column] = size(input);%128 600

pilot\_symbol = zeros(row,column/(pilot\_inter + 1));%128 100

output = zeros(row,column - column/(pilot\_inter + 1));%128 500

for i = 1:column/(pilot\_inter + 1)%1:100

output(:,(1+(i-1)\*pilot\_inter):i\*pilot\_inter) = input(:,(2+(i-1)\*(pilot\_inter+1)):i\*(pilot\_inter+1));

pilot\_symbol(:,i) = input(:,1+(i-1)\*(pilot\_inter+1));

end

end

%插入循环前缀

function symbol\_cp = cyclic\_prefix(output\_ifft)

[row,column] = size(output\_ifft);

symbol\_cp = [output\_ifft((row-31):end,:); output\_ifft];

end

%取出循环前缀

function re\_symbol\_cp = re\_cyclic\_prefix(input)

re\_symbol\_cp = input(33:end,:);

end

%绘制OFDM频谱图

function []=frequency\_response\_deshift(data,tit,col)%qpsk\_bit 128

N=256;

data\_p = zeros(1,256);

data\_p(65:192) = data;

data\_p = awgn(data\_p,50);

data\_p = ifft(data\_p,256);

delta\_f=15\*10^3; %subcarrier spacing

fs=N\*delta\_f; %sampling frequency

ts=1/fs; %sampling period OFDM码元宽度 N\*Ts

data\_sampled=fft(data\_p,1024)\*ts;

data\_sampled = 10\*log10(data\_sampled\*10^3);

data\_ss=fftshift(data\_sampled);

len=length(data\_sampled)-1;

ff=fs/len;

f=0:ff:fs;

%stem(real(Hcentered),imag(Hcentered))

plot(f,real(data\_sampled),col)

title(tit);

xlim([-10\*delta\_f,fs+10\*delta\_f]);

xlabel('frequency');

ylabel('frequency response');

end

%绘制频谱图

function []=frequency\_response(data,tit,col)

N=128;

delta\_f=15\*10^3; %subcarrier spacing

fs=N\*delta\_f; %sampling frequency

ts=1/fs; %sampling period OFDM码元宽度 N\*Ts

data\_sampled=fft(data,1024)\*ts;

data\_ss=fftshift(data\_sampled);

len=length(data\_sampled)-1;

ff=fs/len;

f=[0:ff:fs] -fs/2;

%stem(real(Hcentered),imag(Hcentered))

plot(f,real(data\_ss),col)

title(tit);

xlabel('frequency');

ylabel('frequency response');

end

%绘制时域图

function []= time(inp,titled,col)

stem(inp,col);

title(titled);

xlabel('index');

ylabel('amplitude');

axis tight;

end

%绘制高斯信道误比特率

function [] = awgn\_err\_plot(cdata,pass\_data)

fft\_n = 128;

pilot\_inter = 5;

[row,column] = size(pass\_data);

pass\_data = reshape(pass\_data,1,[]);

snr = 0:1:20;%dB

ERR = zeros(1,21);

N = length(pass\_data);

for i = 0:20

err = 0;

pass = awgn(pass\_data,i);

pass = reshape(pass,row,length(pass)/row);

%去循环码

re\_symbol\_cp = re\_cyclic\_prefix(pass);

%fft

fft\_symbol = fft(re\_symbol\_cp,fft\_n,1)./sqrt(fft\_n);

%去除导频

[output,pilot\_symbol] = re\_insert\_pilot(fft\_symbol,pilot\_inter);

output = reshape(output,1,[]);

data = re\_modulatorQPSK(output);

for n = 1:N

if cdata(n) ~= data(n)

err = err + 1;

end

end

ERR(i+1) = err/N;

end

figure;

semilogy(snr,ERR,'-\*g')

legend('AWGN仿真误码率')

xlabel('符号信噪比 (dB)');

ylabel('误比特率');

grid on;

end

%绘制叠加了高斯信道的多径信道的误比特率

function ERR = ls\_err\_plot(cdata,pass\_data,pilot\_bit)

fft\_n = 128;

pilot\_inter = 5;

[row,column] = size(pass\_data);

pass\_data = reshape(pass\_data,1,[]);

snr = 0:1:20;%dB

ERR = zeros(1,21);

N = length(pass\_data);

%---------------------------------------------多径信道----------------------------------

fs = 15000;

ts = 1/fs;

fd = 0;

tau = [0,50,120,200,230,500,1600,2300,5000]/(10^9);

pdb = [-1.0,-1.0,-1.0,0,0,0,-3.0,-5.0,-7.0];

chan = rayleighchan(ts,fd,tau,pdb);

chan.ResetBeforeFiltering = 0;

chan\_symbol\_s = filter(chan,pass\_data);

%--------------------------------------------多径信道-------------------------------------

for i = 0:20

err = 0;

%------------------------------------------叠加高斯信道---------------------------------

pass = awgn(chan\_symbol\_s,i);

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

pass = reshape(pass,row,length(pass)/row);

%chan\_symbol\_s = reshape(chan\_symbol\_s,row,length(chan\_symbol\_s)/row);

%去循环码

re\_symbol\_cp = re\_cyclic\_prefix(pass);

%re\_symbol\_cp\_ = re\_cyclic\_prefix(chan\_symbol\_s);

%fft

fft\_symbol = fft(re\_symbol\_cp,fft\_n,1)./sqrt(fft\_n);

%fft\_symbol\_ = fft(re\_symbol\_cp\_,fft\_n,1)./sqrt(fft\_n);

%去除导频

[output,pilot\_symbol] = re\_insert\_pilot(fft\_symbol,pilot\_inter);

pilot\_qpsk = modulator\_QPSK(pilot\_bit);

pilot\_patt=repmat(pilot\_qpsk',1,100);

pilot\_esti=pilot\_symbol./pilot\_patt;

[r,c] = size(output);

h = zeros(r,c);

for m = 1:100

h(:,(1+(m-1)\*5):m\*5) = repmat(pilot\_esti(:,m),1,pilot\_inter);

end

output = output./h;

output = reshape(output,1,[]);

data = re\_modulatorQPSK(output);

for n = 1:N

if cdata(n) ~= data(n)

err = err + 1;

end

end

ERR(i+1) = err/N;

end

figure;

semilogy(snr,ERR,'-\*g')

legend('信道均衡仿真误码率')

xlabel('符号信噪比 (dB)');

ylabel('误符号率/误比特率');

grid on;

end