

**《通信原理》报告**

实验名称： 通信原理第六章仿真

院系： 电子信息与通信学院

专业： 电子信息工程

班级： 电信1204班

姓名： 黄衎

学号： U201213468

# 实验目的

* 仿真各种基带波形及频谱
* 仿真基带信号的接收（考虑噪声）
* 画基带信号眼图

# 实验过程

## 消息码的产生与编码

AMI（Alternative Mark Inversion）码全称传号交替反转码，编码规则是将传号“1”交替地变换为“+1”和“-1”，空号“0”保持不变。

AMI编码函数如下：

function encode = AMI\_encode(msg\_code)

% AMI\_encode AMI编码

% @Author: KellyHwong

% @Update: 2015.5.14

% 输入消息码

% 输出AMI码

%从-1开始编码

flag = -1;

encode = zeros(1,length(msg\_code));

for i = 1:length(msg\_code)

if 1==msg\_code(i)

%传号交替

encode(i)=flag;

flag = -flag;

end

end

end

HDB3（High Density Bipolar of Order 3）码，全称三阶高密度双极性码，其实质是改进后的AMI码，是连“0”数目不超过3个，编码函数如下：

function hdb3\_code = HDB3\_encode(msg\_code)

% HDB3\_encode HDB3编码

% @Author: KellyHwong

% @Update: 2015.5.13

% 输入消息码

% 输出HDB3码

%先编AMI码

ami\_code = AMI\_encode(msg\_code);

zeroCounter = 0;

amiCount = 0;

for i = 1:length(ami\_code)

if 0~=ami\_code(i)

%当前AMI码的正负

amiFlag = ami\_code(i);

amiCount = amiCount + 1;

%初始化标记，第一个V与第一个非零AMI码相同，后面则要交替

if 1==amiCount

%V也要交替

hdbFlag = amiFlag;

end

zeroCounter = 0;

else

%数0的个数

zeroCounter = zeroCounter + 1;

%如果达到4个重复的0

if 4==zeroCounter && 0~=amiCount

%当前位置的HDB3码为hdbFlag

ami\_code(i) = hdbFlag;

%V的交替

hdbFlag = -hdbFlag;

%如果V与AMI码相同，则应该插入B，B符号与V相同

if ami\_code(i)~=ami\_code(i-4)

ami\_code(i-3) = ami\_code(i);

end

%重置0计数器

zeroCounter = 0;

end

end

end

hdb3\_code = ami\_code;

end

以书上P140的例子作为测试，验证了我的函数的编码效果：

msg\_code: 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 1

ami\_code: -1,0,0,0,0,1,0,0,0,0,-1,1,0,0,0,0,0,0,0,0,-1,1

hdb3\_code: -1,0,0,0,-1,1,0,0,0,1,-1,1,-1,0,0,-1,1,0,0,1,-1,1

同时写了解码函数，AMI码解码相对简单，取绝对值即可。HDB3的解码函数如下：

function ami\_code = HDB3\_decode(hdb3\_code)

% AMI\_encode AMI编码

% @Author: KellyHwong

% @Update: 2015.5.14

% 输入AMI码

% 输出消息码

% @Quote: 书上原话：从上述编码规则看出，每一个破坏脉冲V总是与前一非零

% 脉冲同极性（包括B在内）。可以很容易找到破坏点，并且断定V以及

% 前面三个符号必定是连续的3个0，由此可以解出AMI码。

amiFlag = 0;

for i = 1:length(hdb3\_code)

if 0~=hdb3\_code(i)

if 1==amiFlag

%如果符号相同了，则证明是V

if hdb3\_code(i)~=expectNext

hdb3\_code(i-3:i) = 0;

amiFlag = 0;

%符号不同则证明是AMI码

else

amiFlag = 1;

expectNext = -hdb3\_code(i);

end

end

if 0~=hdb3\_code(i)

%找到了AMI码

amiFlag = 1;

expectNext = -hdb3\_code(i);

end

end

end

ami\_code = hdb3\_code;

end

仍使用P140的例子：

hdb3\_code: -1,0,0,0,-1,1,0,0,0,1,-1,1,-1,0,0,-1,1,0,0,1,-1,1

解码得：

hdb3\_decode: -1,0,0,0,0,1,0,0,0,0,-1,1,0,0,0,0,0,0,0,0,-1,1

可以看出解出来的码就是AMI码。

## 功率谱密度的估计方法

对于离散时间信号，首先要求其频谱，在Matlab中，使用FFT来实现，FFT是DFT（离散傅里叶变换）的快速算法。

* 离散的采样信号频谱与原始连续时间信号频谱的关系



T为采样时间间隔，即，则：



式子的意义为，，表明了模拟域频率f与数字域频率w的关系。

* DFT与DTFT的关系



故：，DFT是DTFT的离散化

* 用DFT来估计功率谱

我们知道对于连续时间信号，，对于已经求得DFT的采样信号，使用下式来对的采样点作有偏估计：



## 各种基带信号的功率谱密度

第一部分实验代码如下：

%仿真各种基带波形及其频谱

clc,clear,close all;

%随机产生消息码

num\_of\_msg = 1000;

msg\_code = randi([0,1],1,num\_of\_msg);

ami\_code=AMI\_encode(msg\_code);

hdb3\_code = HDB3\_encode(msg\_code);

%码元速率

Rs = 100;

%采样频率

r=20;

Fs = r\*Rs;

%时间轴

totalTime = length(msg\_code)/Rs;

deltaTime = 1/Rs/r;

tt = 0:deltaTime:totalTime-deltaTime;

%% 使用原消息码

%单极性NRZ波形

msg\_spnrz = singlePolarNRZ(msg\_code,r);

%单极性RZ波形（占空比50%）

msg\_sprz = singlePolarRZ(msg\_code,r);

N = length(msg\_spnrz);

%画时域图像

figure();subplot(2,1,1);

plot(tt,msg\_spnrz);axis([0,1,0,2]);

title('原始信息码的单极性NRZ');xlabel('时间/s');ylabel('信号幅值');

subplot(2,1,2);

plot(tt,msg\_sprz);axis([0,1,0,2]);

title('原始信息码的单极性RZ');xlabel('时间/s');ylabel('信号幅值');

%功率谱

MSG\_SPNRZ = fft(msg\_spnrz);

MSG\_SPRZ = fft(msg\_sprz);

P\_MSG\_SPNRZ = MSG\_SPNRZ.\*conj(MSG\_SPNRZ)/N;

P\_MSG\_SPRZ = MSG\_SPRZ.\*conj(MSG\_SPRZ)/N;

omiga = 0:2\*pi/N:2\*pi;omiga = omiga(1:N);

%模拟域频率

ff=omiga\*Fs/2/pi;

figure();subplot(2,1,1);

plot(ff,abs(P\_MSG\_SPNRZ));axis([0,Fs/2,0,10]);

title('原始信息码的单极性NRZ');xlabel('频率/Hz');ylabel('功率谱');

subplot(2,1,2);

plot(ff,abs(P\_MSG\_SPRZ));axis([0,Fs/2,0,10]);

title('原始信息码的单极性RZ');xlabel('频率/Hz');ylabel('功率谱');

%% 使用AMI码

%单极性NRZ波形

msg\_spnrz = singlePolarNRZ(ami\_code,r);

%单极性RZ波形（占空比50%）

msg\_sprz = singlePolarRZ(ami\_code,r);

N = length(msg\_spnrz);

%画时域图像

figure();subplot(2,1,1);

plot(tt,msg\_spnrz);axis([0,1,-2,2]);

title('AMI的单极性NRZ');xlabel('时间/s');ylabel('信号幅值');

subplot(2,1,2);

plot(tt,msg\_sprz);axis([0,1,-2,2]);

title('AMI的单极性RZ');xlabel('时间/s');ylabel('信号幅值');

%功率谱

MSG\_SPNRZ = fft(msg\_spnrz);

MSG\_SPRZ = fft(msg\_sprz);

P\_MSG\_SPNRZ = MSG\_SPNRZ.\*conj(MSG\_SPNRZ)/N;

P\_MSG\_SPRZ = MSG\_SPRZ.\*conj(MSG\_SPRZ)/N;

omiga = 0:2\*pi/N:2\*pi;omiga = omiga(1:N);

%模拟域频率

ff=omiga\*Fs/2/pi;

figure();subplot(2,1,1);

plot(ff,abs(P\_MSG\_SPNRZ));axis([0,Fs/2,0,10]);

title('AMI的单极性NRZ');xlabel('频率/Hz');ylabel('功率谱');

subplot(2,1,2);

plot(ff,abs(P\_MSG\_SPRZ));axis([0,Fs/2,0,10]);

title('AMI的单极性RZ');xlabel('频率/Hz');ylabel('功率谱');

%% 使用HDB3码

%单极性NRZ波形

msg\_spnrz = singlePolarNRZ(hdb3\_code,r);

%单极性RZ波形（占空比50%）

msg\_sprz = singlePolarRZ(hdb3\_code,r);

N = length(msg\_spnrz);

%画时域图像

figure();subplot(2,1,1);

plot(tt,msg\_spnrz);axis([0,1,-2,2]);

title('HDB3的单极性NRZ');xlabel('时间/s');ylabel('信号幅值');

subplot(2,1,2);

plot(tt,msg\_sprz);axis([0,1,-2,2]);

title('HDB3的单极性RZ');xlabel('时间/s');ylabel('信号幅值');

%功率谱

MSG\_SPNRZ = fft(msg\_spnrz);

MSG\_SPRZ = fft(msg\_sprz);

P\_MSG\_SPNRZ = MSG\_SPNRZ.\*conj(MSG\_SPNRZ)/N;

P\_MSG\_SPRZ = MSG\_SPRZ.\*conj(MSG\_SPRZ)/N;

omiga = 0:2\*pi/N:2\*pi;omiga = omiga(1:N);

%模拟域频率

ff=omiga\*Fs/2/pi;

figure();subplot(2,1,1);

plot(ff,abs(P\_MSG\_SPNRZ));axis([0,Fs/2,0,10]);

title('HDB3的单极性NRZ');xlabel('频率/Hz');ylabel('功率谱');

subplot(2,1,2);

plot(ff,abs(P\_MSG\_SPRZ));axis([0,Fs/2,0,10]);

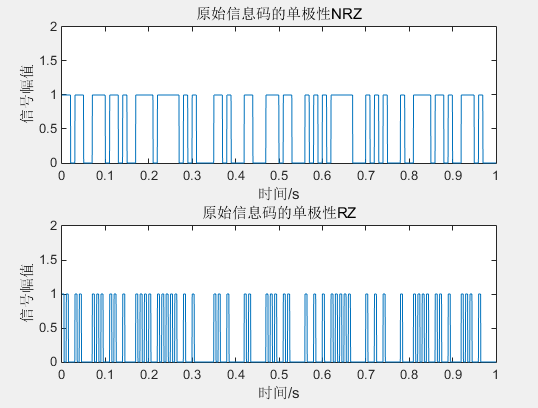
title('HDB3的单极性RZ');xlabel('频率/Hz');ylabel('功率谱');

实验首先产生一组0-1二值随机数，作为消息码，为方便对照，后面的AMI码，HDB3码均采用相同的消息码作为原码。

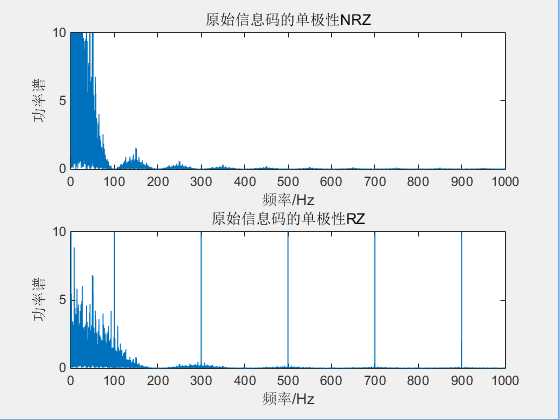
注意，消息码的总个数被设置为1000个，波特率为100Hz，即信号总时长为10s，但是为了方便比较，只画出了1s内的时域波形。

### 原信息码

NRZ和RZ（占空比50%）的时域波形：

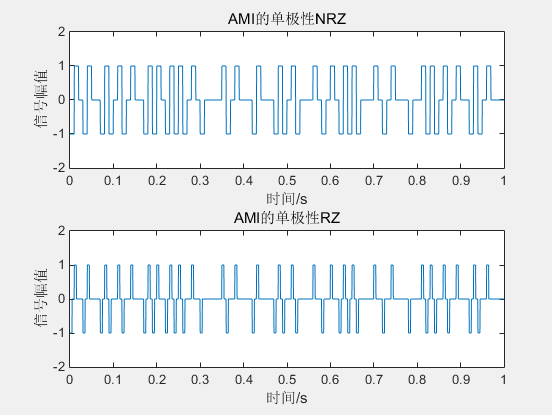


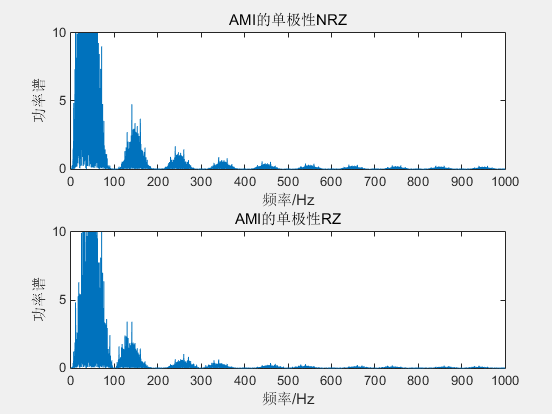
原始信息码的NRZ与RZ波形的功率谱：



实验采用的波特率为，可以看出NRZ没有稳态波的频谱（由组多单频率点的冲击组成），主瓣的带宽为100Hz；而对于RZ，则有稳态波的频率，且有定时信息（100Hz处的冲击），但是占用带宽较大，主瓣带宽约为200Hz。以上结果与书上P138的图6-3比较相符。

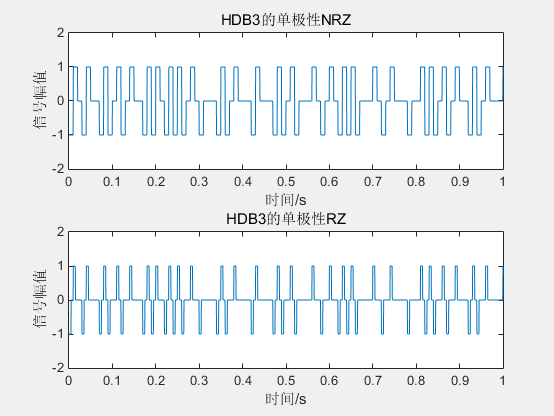
### AMI码

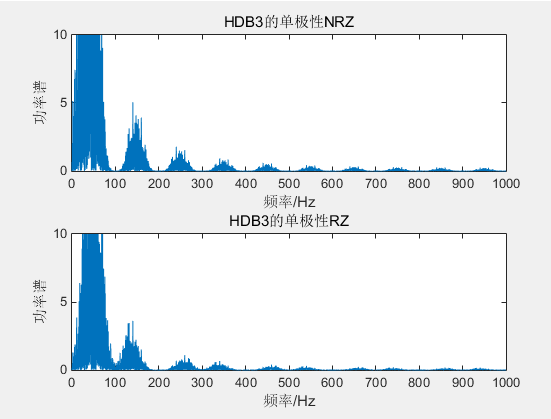




AMI码由于传号交替反转，功率谱中的直流和低频分量大幅减少。

### HDB3码





HDB3的功率谱密度与AMI比较相似，只是在主瓣的顶峰有所不同，但这里没有显示出来。

## 基带信号的接收与眼图

第二部分实验的代码如下：

%仿真基带信号的接收、眼图

clc, clear,close all;

%% 设计升余弦滚降滤波器

alpha = 0.2;% alpha在[0,1]之间

Fd=200;

%码元速率

Rs = 100;

%采样频率

Fs = 20\*Rs;

rate=Fs/Rs;

[h,a]=rcosine(Fd,Fs,'fir',alpha);

%% 随机产生消息码

num\_of\_msg = 100;%码元个数

msg\_code = randi([0,1],1,num\_of\_msg);

ami\_code=AMI\_encode(msg\_code);

hdb3\_code = HDB3\_encode(msg\_code);

%单极性NRZ波形

spnrz = singlePolarNRZ(msg\_code,rate);

%转化为双极性波形

dpnrz = spnrz\*2-1;

%% 添加噪声

mu = 0; sigma = 0.4;%sigma^2/2是噪声功率

noise = randn(1,length(dpnrz));

noise = noise\*sigma+mu;

dpnrz\_with\_noise = dpnrz + noise;

%% 通过升余弦滚降滤波器

recv = conv\_m(dpnrz\_with\_noise,1:length(dpnrz\_with\_noise),h,1:length(h));

%时间轴

totalTime = length(msg\_code)/Rs;

deltaTime = 1/Rs/rate;

tt = 0:deltaTime:totalTime-deltaTime;

figure(1);

subplot(2,1,1);

plot(tt,recv(1:length(tt)));

axis([0,1,-20,20]);

subplot(2,1,2);

plot(tt,dpnrz\_with\_noise(1:length(tt)));

% axis([0,1,-1.5,1.5]);

%% 画眼图

figure(2);

eye\_num = 3;

% 将相邻码元的波形重叠在一起

hold on;

% 注意i从2开始，即第二个码元，跳过了滤波器过渡的阶段

for i =2 : floor(num\_of\_msg/eye\_num)

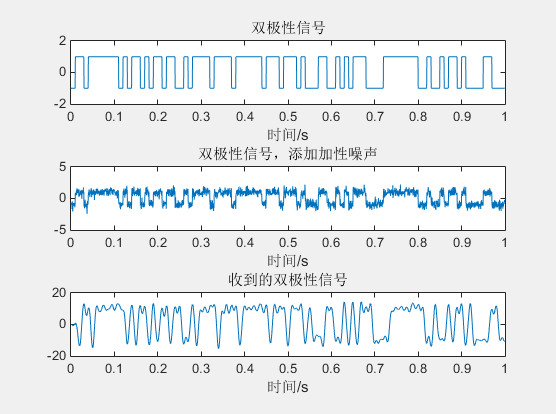
plot(tt(1:rate\*eye\_num),...

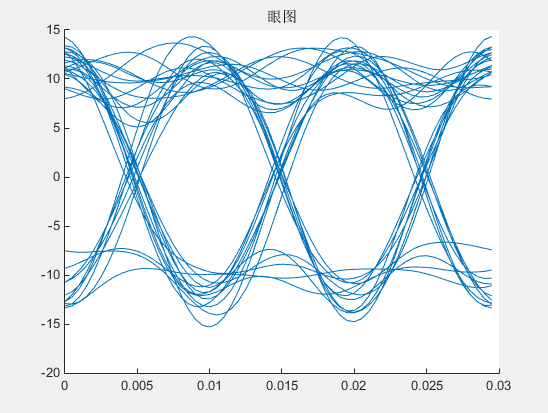
recv(rate\*eye\_num\*(i-1)+1:rate\*eye\_num\*i),...

'Color',[0 0.4470 0.7410] );

end

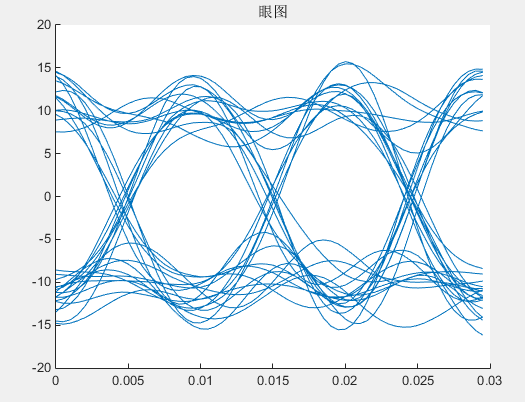
同样的，只画出1s内的时域波形，但是眼图中将除了了滤波器过渡的阶段的波形全部叠加起来。



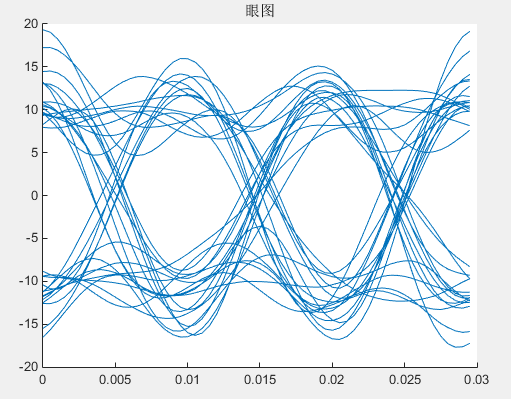


上面的图中，噪声是零均值，sigma=0.4的高斯噪声。现在可以改变sigma多做几组实验：

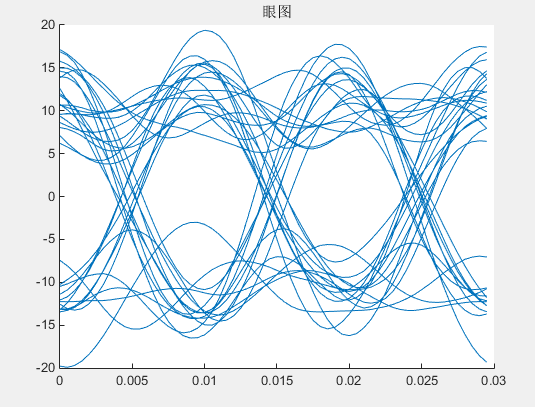
sigma=0.6：



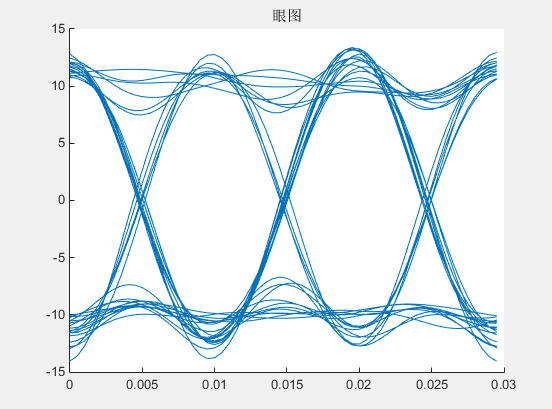
sigma=0.8：



sigma=1：



sigma=0.1：



可以发现，噪声功率越小，眼图中的线越细，“眼睛”睁得越大，噪声容限也越大，对定时误差的灵敏度越低。

# 实验感想

首先这次实验让我复习了DSP的知识，回顾了使用Matlab分析信号的方法。并且在做实验的过程中，一些不懂的地方也让我对书上的知识掌握得更加透彻，比如HDB3的编解码，在自己动手的过程中，对这种编码方式有了更深的理解。最后实验的结果与书上相应的图示比较相符的时候我也非常欣喜。

总之，此次仿真实验虽然花了我很多精力（包括复习知识和写代码），但是也很有收获，这种将理论和实践相结合的学习方法是非常好的。之后的课程如果也可以做仿真并且自己有精力的话，我也会尽量去做，因为这对理论知识的掌握、和动手能力的提高，都是有好处的，何乐而不为？

# 参考文献

【1】通信原理（第7版），樊昌信，曹丽娜，国防工业出版社

【2】数字信号处理（第3版），姚天任，江太辉，华中科技大学出版社