抑制PAPR的SLM算法

# 背景分析

OFDM技术具有众多优良性能，如抗多径、抗干扰能力强等；但是OFDM技术的一大缺陷是峰均功率比（Peak to Average Power Ratio，PAPR）较高，PAPR较高的信号容易超出放大器的线性范围，从而带来失真。

目前已经存在各种各样的PAPR抑制算法，首要任务就是看懂目前常用的PAPR抑制算法，然后，将PAPR算法移植到OFDM-LFM信号中。

OFMD（Orthogonal Frequency Division Multiplexing）作为一种调制技术，其原理是将信道分成多个正交的子信道，通信信号可以在各个子信道上进行传输。该调制技术已经成为4G的关键技术，广泛应用于ADSL、DAB、DVB、HDTV、WLAN等应用场景。但是，OFDM具有明显的缺陷：其中最为突出的问题就是峰均功率比（Peak to Average Power Ratio）过高。因为OFDM信号是由许多个字信号在时域叠加，会造成最大功率与平均功率的过大，即带来过高的PAPR。过高的PAPR对A/D和D/A转换器、信号放大器等具有较高的线性要求。除此之外，过高的PAPR会带来非线性失真，导致符号间干扰，从而增大了误码率。因此，采用OFDM信号必须要解决PAPR过高的问题。

OFDM-LFM与OFDM的区别：

OFDM与OFDM-LFM信号都是多个子载波在时域的叠加，区别在于OFDM的子载波的频率是个单值，OFDM-LFM的子载波是一段频率，且是线性。

# PAPR的定义

PAPR(Peak Average Power Ratio,峰均功率比)的表达式：

dB

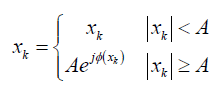
其中，表示信号的峰值功率；表示信号的均值功率，dB是PAPR的单位。

# 对PAPR问题的解决方法

目前对PAPR问题的解决方法主要三种：（1）预失真技术（2）编码技术（3）概率技术。

### 裁剪算法（Clipping Algorithm）

基本原理：



当信号的幅值低于阈值A时，不做处理；当信号的幅值高于阈值A时，那么就利用阈值A作为新的幅值。裁剪算法的阈值通过CR(Clipping Ratio)来决定，表达式为



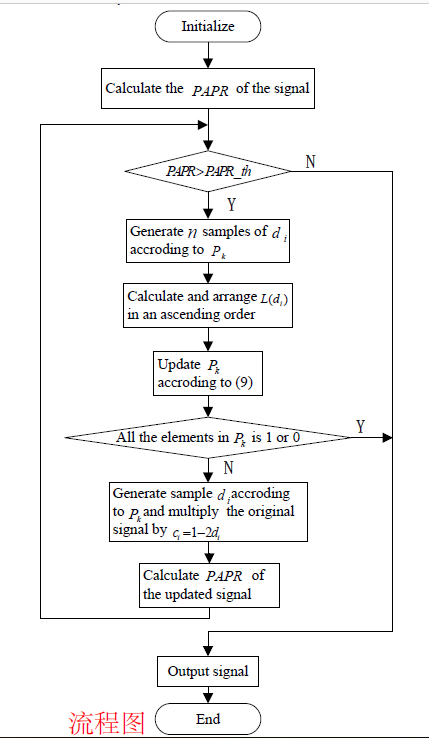
其中，Pin表示信号裁剪之前的平均功率。

裁剪算法可以有效地降低OFDM系统的PAPR值，但是同时也引入了非线性失真，增大了系统的误码率。

### SLM算法（Selective Mapping）-选择性映射算法

SLM算法是一种利用概率来解决PAPR问题的方法。该方法利用随机的M序列与原始的OFDM信号相乘，通过不断地判断，直至选择出合适的序列。

SLM算法在不失真的情况下，可以降低OFDM系统的PAPR，但是这种算法的复杂度比较高，需要多次的IFFT变换。



### CE-SLM算法

鉴于SLFM算法和Clipping Algorithm的缺陷，可以将两种算法结合在一起，得到一种联合算法：CE-SLM-Clipping。文献[1]中，重点阐述了该联合算法并对该算法做了仿真分析。

# 抑制PAPR的三种主要方法

降低PAPR的技术主要分为三种：限幅类技术、编码类技术和概率类技术。

## 限幅类技术

限幅类技术采用的非线性过程会引起带内信号的畸变和带外噪声，导致系统性能下降。其具体方法包括限幅滤波、峰值加窗和峰值抵消。

## 编码类技术

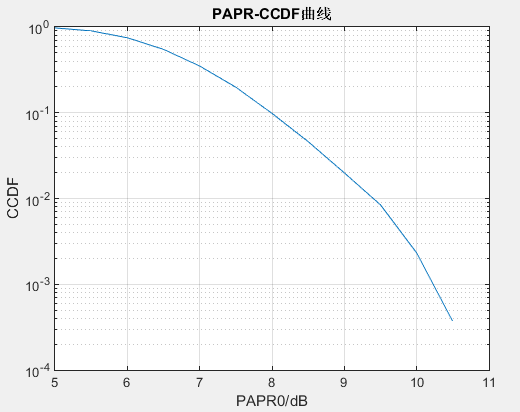
编码类技术在减小了PAPR的同时也降低了数据传输速率，编解码器复杂，计算量大，具体方法有格雷补码序列和线性块编码。

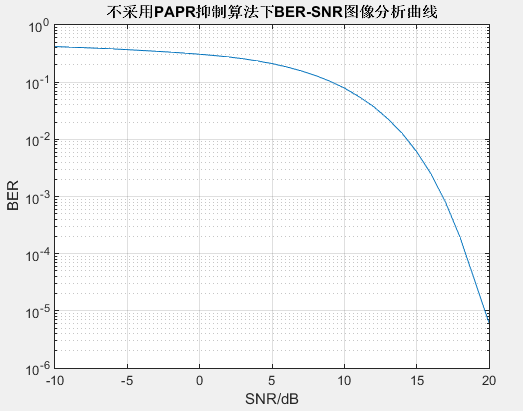
## 概率类技术

概率类技术并不着眼于降低信号幅度的最大值，而是降低峰值出现的概率，但是将带来一定的信息冗余，具体方法包括选择映射（SLM）、部分传输序列（PTS）、冲激整形（PS）等。

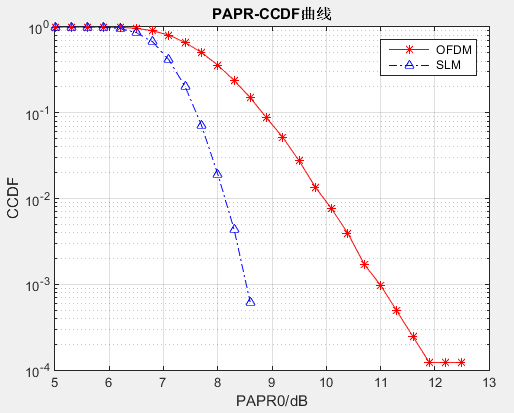
# 仿真分析

## 设计16QAM调制与解调、IFFT与FFT、过采样、高斯噪声、串并转换等。

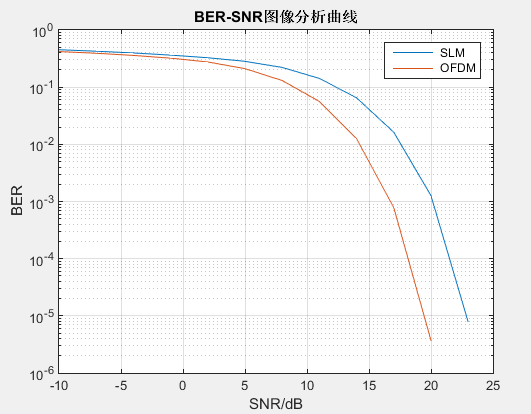




## SLM算法的仿真分析



从上图看出，采用SLM算法可以较好地降低PAPR值，但是，导致系统的复杂度大大升高。通过分析，计算复杂度主要是集中在IFFT模块，因此设计新的算法降低计算复杂度是一个重要的研究方向。



采用SLM算法以高昂的计算复杂度换取了低的PAPR，但是同时导致误码率升高。采用SLM算法，需要发送side information，致使通信传输速率有略微降低。在接收端，side information接收错误，将导致较大的误码率。因此，不传输side information的情况下，实现SLM算法具有较大的研究价值。

附录1.

matlab代码：

% 该程序功能改进：

% 在QAM\_2\_OFDM\_PAPR.m基础上,添加了过采样功能

% 在添加信噪比修改了

close all;clear;clc

%信号比特数据n

n = 1024\*128\*16; n\_16 = n/4;

%MQAM

M = 16; k = log2(M);

%载波个数

N = 64;

%过采样率

L = 1; %over\_samp

%生成随机二进制比特流(行向量）

x = randi([0 1],1,n);

%转换矩阵维度，每k个bit一组

x\_4 = reshape(x,length(x)/k,k);

%变成16进制

x\_16 = bi2de(x\_4,'right-msb');%right-msb参数表示第一列为最低位置

%用16QAM调制器对信号进行调制

y = modulate(modem.qammod(M),x\_16);

%y输出的y为一个复数向量

colums\_num = length(y)/N;

y\_para = reshape(y,N,colums\_num);

%-------添加过采样功能-------------

y\_para\_oversample = [y\_para(1:N/2,:);zeros((L-1)\*N,colums\_num);y\_para(N/2+1:end,:)];

%-----------对QAM调制后的数据进行IFFT并计算PAPR---------

LN = L\*N;

x\_ofdm = zeros(LN,colums\_num);%x\_ofdm 存放调制后的数据

papr = zeros(1,colums\_num);

for i = 1:colums\_num

x\_ofdm(:,i) = ifft(y\_para(:,i),LN);%默认也是LN点ifft

p\_av = 1/N\*sum(abs(x\_ofdm(:,i)).^2);%这里先算成N

p\_peak = max(abs(x\_ofdm(:,i)).^2);

papr(i) = 10\*log10(p\_peak/p\_av);

end

papr\_baseS = 5:0.5:15;

percent\_papr = zeros(1,length(papr\_baseS));

i = 0;

for papr\_base = papr\_baseS

i = i + 1;

percent\_papr(i) = length(find(papr > papr\_base))/length(papr);

end

figure

semilogy(papr\_baseS,percent\_papr),grid on

xlabel('PAPR0/dB');ylabel('CCDF');title('PAPR-CCDF曲线');

%----------PAPR分析结束---------------

%-------下面进行发送到信道，加噪声，fft、去过采样、dQAM,计算误比特率----------

%并变串

y\_Serial\_ifft = reshape(x\_ofdm,1,LN\*length(x\_ofdm));

j\_index = 0;EbNos = -10:1:30;

rates = zeros(1,length(EbNos));

for EbNo= EbNos

%snr = EbNo+10\*log10(k)-10\*log10(L);%信噪比

snr = EbNo;

yn=awgn(y\_Serial\_ifft,snr,'measured');%加入高斯白噪声

%串变并

colums\_num\_R = length(yn)/LN;

y\_para\_R = reshape(yn,LN,colums\_num\_R);

%fft变换

y\_fft = zeros(LN,colums\_num\_R);

for i = 1:colums\_num\_R

y\_fft(:,i) = fft(y\_para\_R(:,i),LN);

end

%---------去除过采样-------------

y\_fft\_quoversample = [y\_fft(1:N/2,:);y\_fft(end-N/2+1:end,:)];

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

yd = demodulate(modem.qamdemod(M),y\_fft\_quoversample);%此时解调出来的是16进制信号

z=de2bi(yd,'right-msb'); %转化为对应的二进制比特流

z=reshape(z,1,n);

[number\_of\_errors,bit\_error\_rate]=biterr(x,z);

j\_index = j\_index+1;

rates(j\_index) = bit\_error\_rate;

end

disp('误比特率：');rates

figure,semilogy(EbNos,rates),grid on;

xlabel('SNR/dB');ylabel('BER');title('不采用PAPR抑制算法下BER-SNR图像分析曲线')

参考文献

[1] Ce, J., Dianxia, J., Chao, Z., & Lijuan, W. (2017, January). **A Union Algorithm of PAPR Reduction for OFDM Systems**. In Measuring Technology and Mechatronics Automation (ICMTMA), 2017 9th International Conference on (pp. 339-342). IEEE.

[2] M. Kaur, P. Singh and S. Singh, "**CMA technique: A solution for minimum PAPR in OFDM**," 2015 2nd International Conference on Recent Advances in Engineering & Computational Sciences (RAECS), Chandigarh, 2015, pp. 1-4.

附录2.

% 该程序功能改进：

% 在QAM\_2\_OFDM\_PAPR.m基础上,添加了过采样功能

% 在添加信噪比修改了

% SLM算法：该代码并不是符合SLM算法

% SLM算法应该是对个符号都进行筛选，从而选择出PAPR最低的那个。

% 改进的算法见

close all;clear;clc

%信号比特数据n

n = 1024\*128\*64; n\_16 = n/4;

%MQAM

M = 16; k = log2(M);

%载波个数

N = 256;

%过采样率over\_samp

L = 1; LN = L\*N;

%SLM算法的备选信号

U = 4;%不要大于16

%生成随机二进制比特流(行向量）

x = randi([0 1],1,n);

%转换矩阵维度，每k个bit一组

x\_4 = reshape(x,length(x)/k,k);

%变成16进制

x\_16 = bi2de(x\_4,'right-msb'); %right-msb参数表示第一列为最低位置

%用16QAM调制器对信号进行调制

y = modulate(modem.qammod(M),x\_16);

%输出的y为一个复数向量

colums\_num = length(y)/N;

%串变并

y\_para = reshape(y,N,colums\_num);

%生成向量因子Q

Q = 1j.^randi([1,4],U,N);

%下面进行SLM算法抑制PAPR

y\_slm\_para = zeros(LN+1,colums\_num);%待发送的信号

paprs = zeros(1,colums\_num);

for i = 1:colums\_num

[y\_slm\_para(:,i), paprs(i)]= Func\_PAPR\_SLM\_0430(conj(y\_para(:,i)'),Q,L,M);

end

% 普通OFDM下PAPR-CCDF分析

%-------添加过采样功能-------------

y\_para\_oversample = [y\_para(1:N/2,:);zeros((L-1)\*N,colums\_num);y\_para(N/2+1:end,:)];

%-----------对QAM调制后的数据进行IFFT并计算PAPR---------

x\_ofdm = zeros(LN,colums\_num);%x\_ofdm 存放调制后的数据

paprs\_ofdm = zeros(1,colums\_num);

for i = 1:colums\_num

%注意LN点采样后需要乘以L,因为采样后的幅度变为原来的1/L

x\_ofdm(:,i) = L\*ifft(y\_para\_oversample(:,i),LN);%默认也是LN点ifft

p\_av = 1/N\*sum(abs(x\_ofdm(:,i)).^2);%这里先算成N

p\_peak = max(abs(x\_ofdm(:,i)).^2);

paprs\_ofdm(i) = 10\*log10(p\_peak/p\_av);

end

% 下面统计PAPR的CCDF,并作图

papr\_baseS = 5:0.3:13;

percent\_papr\_ofdm = zeros(1,length(papr\_baseS));

percent\_papr\_slm = zeros(1,length(papr\_baseS));i = 0;

for papr\_base = papr\_baseS

i = i + 1;

percent\_papr\_ofdm(i) = length(find(paprs\_ofdm > papr\_base))/length(paprs\_ofdm);

percent\_papr\_slm(i) = length(find(paprs > papr\_base))/length(paprs);

end

figure

semilogy(papr\_baseS,percent\_papr\_ofdm,'-r\*'); hold on;

semilogy(papr\_baseS,percent\_papr\_slm,'-.b^');

legend(' OFDM',' SLM')

grid on;xlabel('PAPR0/dB');ylabel('CCDF');title('PAPR-CCDF曲线');

%%%-------------开始SLM解调计算BER------------------------------------

y\_slm\_serial = reshape(y\_slm\_para,1,colums\_num\*(LN+1));%并变串

j\_index = 0;EbNos = -10:3:30;

rates = zeros(1,length(EbNos));

for EbNo = EbNos

%添加噪声

y\_slm\_serial\_noise = awgn(y\_slm\_serial,EbNo,'measured');

%串变并

y\_slm\_para\_noise = reshape(y\_slm\_serial\_noise,LN+1,colums\_num);

%-------------------开始解调-------------------

y\_slm\_after\_fft\_over = zeros(LN,colums\_num);

for i = 1:colums\_num

%当SNR较低时，导致q严重错误

q = round(demodulate(modem.qamdemod(M),y\_slm\_para\_noise(LN+1,i)));

if ( q>U || q<1 )

q = randi([1,4]);

end

%q\_No\_noise = demodulate(modem.qamdemod(M), y\_slm\_para(LN+1,i));

y\_slm\_after\_fft\_over(:,i) = fft(y\_slm\_para\_noise(1:LN,i),LN)./conj(Q(q,:)');

end

%---------去除过采样-------------

y\_slm\_after\_fft\_quover = [y\_slm\_after\_fft\_over(1:N/2,:); y\_slm\_after\_fft\_over(end-N/2+1:end,:)];

y\_slm\_after\_fft = demodulate(modem.qamdemod(M),y\_slm\_after\_fft\_quover);

z = de2bi(y\_slm\_after\_fft,'right-msb'); %转化为对应的二进制比特流

z = reshape(z,1,n);

[number\_of\_errors,bit\_error\_rate] = biterr(x,z);

j\_index = j\_index + 1;

rates(j\_index) = bit\_error\_rate;

end

%%-------------普通OFDM-------------

j\_index\_ofdm = 0;

rates\_ofdm = zeros(1,length(EbNos));

y\_ofdm\_serial = reshape(x\_ofdm,1,colums\_num\*LN);

for EbNo = EbNos

%添加噪声

y\_ofdm\_serial\_noise = awgn(y\_ofdm\_serial,EbNo,'measured');

%串变并

y\_ofdm\_para\_noise = reshape(y\_ofdm\_serial\_noise,LN,colums\_num);

%开始解调

y\_ofdm\_after\_fft\_over = zeros(LN,colums\_num);

for i = 1:colums\_num

y\_ofdm\_after\_fft\_over(:,i) = fft(y\_ofdm\_para\_noise(1:LN,i),LN);

end

%---------去除过采样-------------

y\_ofdm\_after\_fft\_quover = [y\_ofdm\_after\_fft\_over(1:N/2,:);y\_ofdm\_after\_fft\_over(end-N/2+1:end,:)];

y\_ofdm\_after\_fft = demodulate(modem.qamdemod(M),y\_ofdm\_after\_fft\_quover);

z = de2bi(y\_ofdm\_after\_fft,'right-msb'); %转化为对应的二进制比特流

z = reshape(z,1,n);

[number\_of\_errors,bit\_error\_rate\_ofdm] = biterr(x,z);

j\_index\_ofdm = j\_index\_ofdm + 1;

rates\_ofdm(j\_index\_ofdm) = bit\_error\_rate\_ofdm;

end

disp('误比特率：')

rates

rates\_ofdm

figure,semilogy(EbNos,rates),grid on;

hold on ;semilogy(EbNos,rates\_ofdm);

xlabel('SNR/dB');ylabel('BER');title('BER-SNR图像分析曲线');

legend(' SLM',' OFDM')