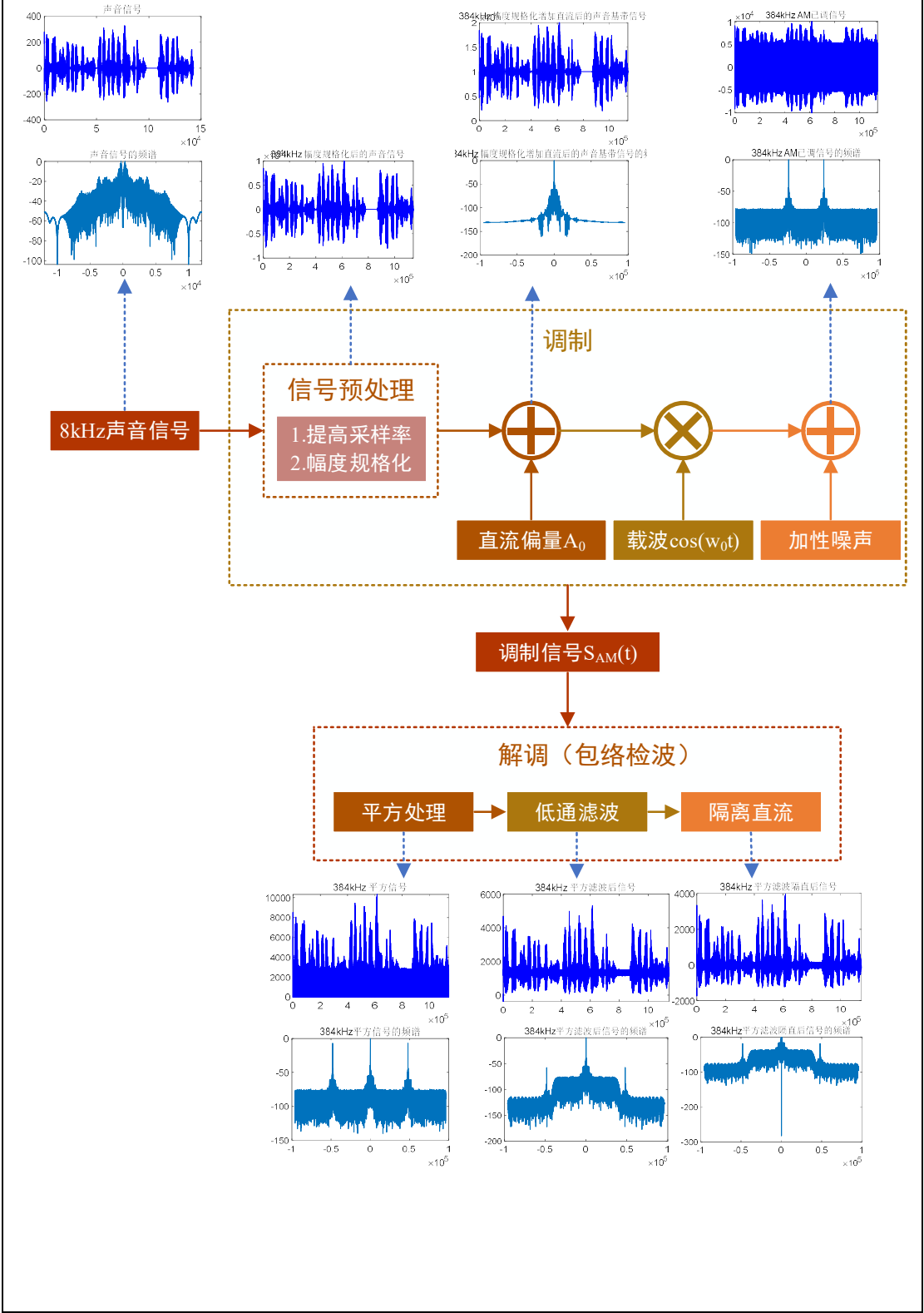


2023-2024 学年第一学期 通信原理 仿真报告

仿真实验一	
2023 年 10 月 25 日	
<div>一、仿真条件</div> <div><div>1. A00_Gen_AM_IF_pcmFile.m</div><div>AM 调制生成工具，将原始 8k 声音样本文件转成 AM 调制信号。</div></div> <div><div>2. fun_analyzeSpectrum.m</div><div>分析并显示信号的频谱</div></div> <div><div>3. fun_plot.m</div><div>显示信号的波形</div></div> <div><div>4. sound_man_8k_16bit.wav</div><div>8k 样本声音文件</div></div> <div><div>5. sound_woman_8k_16bit.wav</div><div>8k 样本声音文件</div></div>	
<div>二、仿真要求</div> <div><div>1. 研读仿真代码，理解算法结构，能够结合框图描述仿真代码的算法过程。</div><div>2. 模仿 AM 调制的仿真代码，设计生成 DSB 和 SSB 调制信号的代码。</div><div>3. 编写 AM 的解调代码，采用包络检波的方法对 AM 信号进行解调，对解调后的信号进行分析，分析调制指数（叠加的直流信号的影响）。</div><div>4. 编写 AM 的相干解调的代码，假设载存在 1%,5%的频偏的情况下，仿真相干解调的而结果。</div></div>	
<div>三、仿真实验及结论</div> <div><div>1. 仿真代码的算法过程</div><div>AM 调制的时域表达式为：</div><div>$s_{AM}(t) = [A_0 + m(t)] \cos w_c t$</div><div>AM 信号的频谱为：</div><div>$s_{AM}(w) = \pi A_0 [\delta(w + w_c) + \delta(w - w_c)] + \frac{1}{2} [M(w + w_c) + M(w - w_c)]$</div></div>	

2023-2024 学年第一学期 通信原理 仿真报告

MATLAB AM 仿真代码算法框图如下：



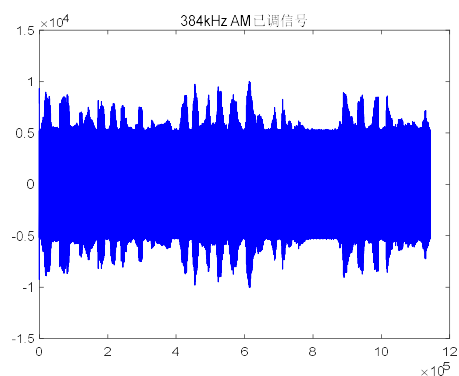
2023-2024 学年第一学期 通信原理 仿真报告

2. DSB 和 SSB 调制信号

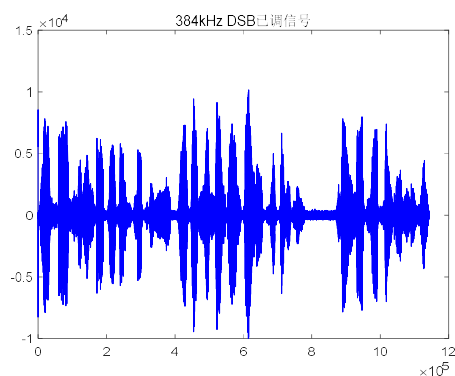
DSB 调制

双边带调制与 AM 调制相比, 无直流分量。其时域表达式为:

$$s_{DSB}(t) = m(t) \cos w_c t$$



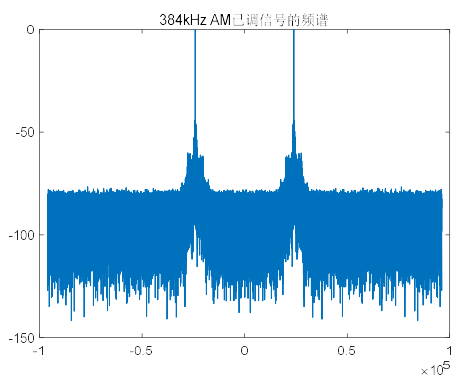
AM 调制信号



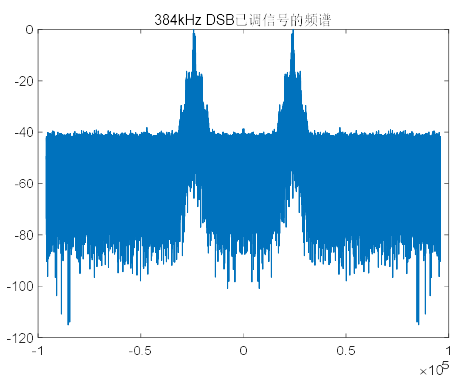
DSB 调制信号

DSB 的频谱与 AM 的谱相近, 只是没有了在 $\pm w_c$ 处的 δ 函数

$$s_{DSB}(w) = \frac{1}{2} [M(w + w_c) + M(w - w_c)]$$



AM 调制信号频谱



DSB 调制信号频谱

```
%% DSB 调制
for i=1:length(Sa)
    cCos(i,1)=cos(2*pi*f/Sampling_rate*(i-1));
    cSin(i,1)=sin(2*pi*f/Sampling_rate*(i-1));
end
Sa_IF = Sa' .* cCos;
%% 幅度规格化
max_am = max(abs(Sa_IF));
factor = IF_am/max_am;
Sa_IF = Sa_IF * factor;
```

2023-2024 学年第一学期 通信原理 仿真报告

%% 增加加性噪声并转换数据格式为整型

```
IF_noise = wgn(length(Sa_IF),1,noise);
Sa_IF = Sa_IF + IF_noise;
Sa_IF_Int16 = int16(Sa_IF);
fun_plot(Sa_IF_Int16,40,'384kHz DSB 已调信号');
fun_analyzeSpectrum(double(Sa_IF_Int16),Sampling_rate,41,'384kHz DSB 已调信号的频谱');
```

SSB 调制

SSB 信号是将 DSB 信号的一个边带滤掉而形成的，方法有滤波法和相移法。滤波法中， $H(w)$ 为单边带滤波器的传输函数，若它具有如下理想高通特性，可滤除下边带：

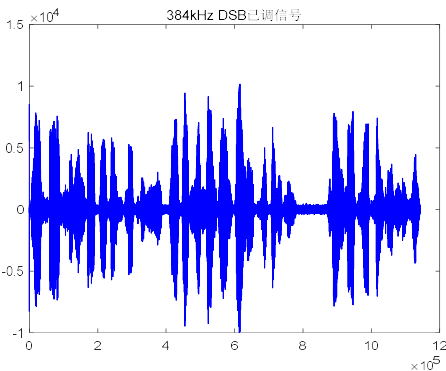
$$H(w) = H_{USB}(w) = \begin{cases} 1 & |w| > w_c \\ 0 & |w| \leq w_c \end{cases}$$

若它具有如下理想低通特性，可滤除上边带：

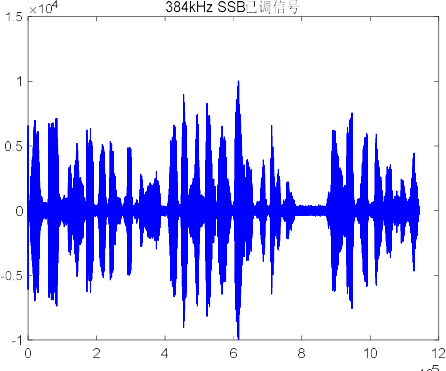
$$H(w) = H_{LSB}(w) = \begin{cases} 1 & |w| < w_c \\ 0 & |w| \geq w_c \end{cases}$$

SSB 频谱可以表示为：

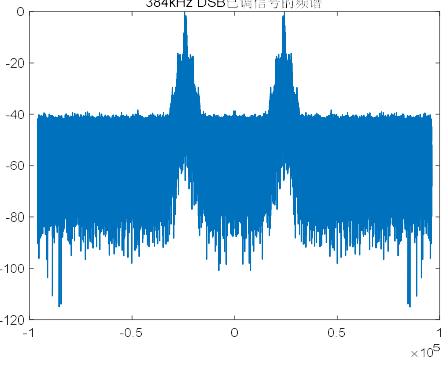
$$S_{SSB}(w) = S_{DSB}(w)H(w)$$



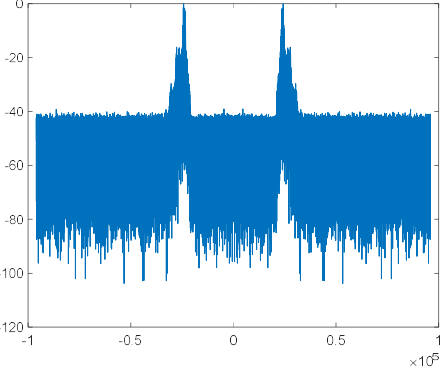
384kHz DSB已调信号



384kHz SSB已调信号



384kHz DSB已调信号的频谱



384kHz SSB已调信号的频谱

DSB 调制信号

SSB 调制信号

DSB 调制信号频谱

SSB 调制信号频谱 (高通滤波)

2023-2024 学年第一学期 通信原理 仿真报告

%% 滤波法

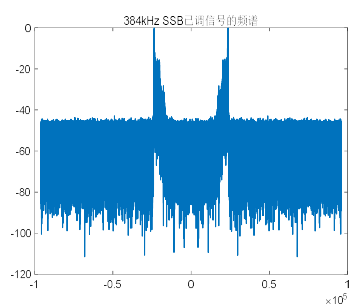
Sa_IF = Sa' .* cCos;

%% 理想高通滤波器

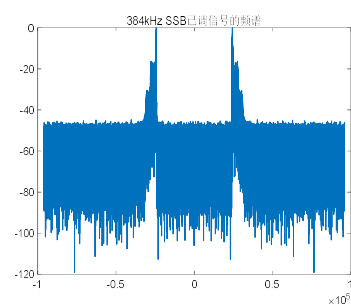
Sa_IF = highpass(Sa_IF, f, Sampling_rate);

相移法 SSB 信号的时域表达式为:

$$S_{SSB}(t) = \frac{1}{2}m(t)\cos w_c t \mp \frac{1}{2}\hat{m}(t)\sin w_c t$$



SSB (LSB) 调制信号频谱



SSB (USB) 调制信号频谱

%% SSB 调制

for i=1:length(Sa)

cCos(i,1)=cos(2*pi*f/Sampling_rate*(i-1));

cSin(i,1)=sin(2*pi*f/Sampling_rate*(i-1));

end

%% LSB

hmt = imag(hilbert(Sa'));

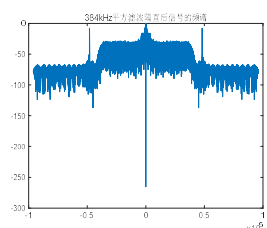
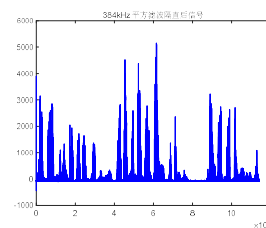
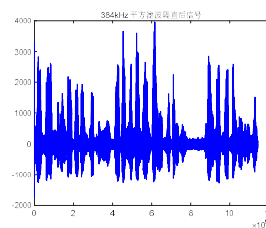
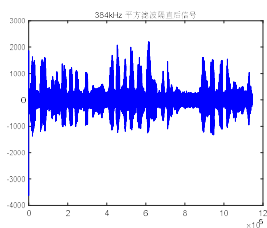
Sa_IF = 1/2*Sa' .* cCos + 1/2*hmt .* cSin;

%% USB

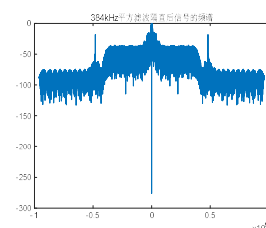
hmt = imag(hilbert(Sa'));

Sa_IF = 1/2*Sa' .* cCos - 1/2*hmt .* cSin;

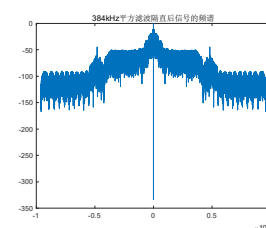
3. 采用包络检波的方法对 AM 信号进行解调，对解调后的信号进行分析，分析调制指数。



modulation_index=0.3



modulation_index=1

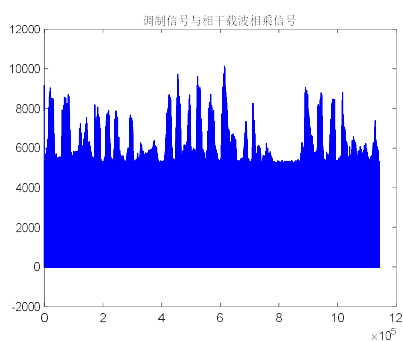
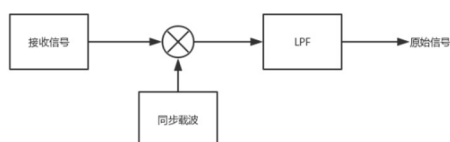


modulation_index=20

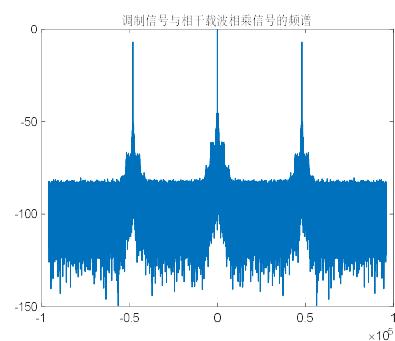
2023-2024 学年第一学期 通信原理 仿真报告

有上述三种情况可知，调制指数过大或过小都会影响最终解调的信号质量。较小的调制指数可能会导致信号过弱，使得信息传输不完整；较大的调制指数可以产生更大的幅度变化，从而提高调制信号质量，但是当 $\text{modulation_index} > 1$ 时，就会出现“过调幅”现象，包络线有时会反转相位，包络检波会发生失真。

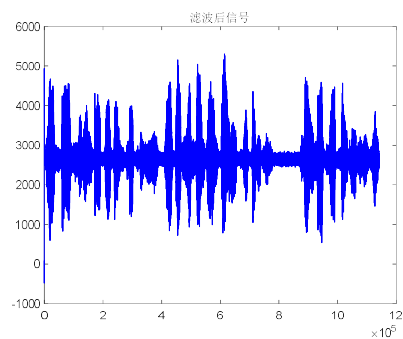
4. AM 的相干解调的代码，假设存在 1%,5%的频偏的情况下，仿真相干解调的结果。



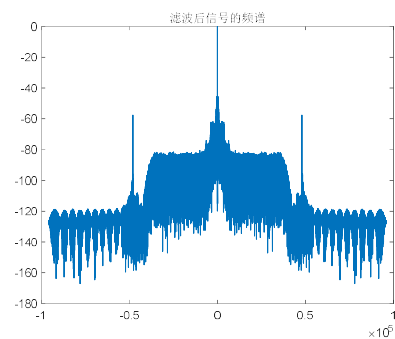
调制信号与相干载波相乘信号



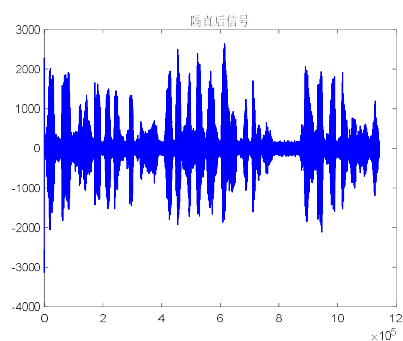
调制信号与相干载波相乘信号频谱



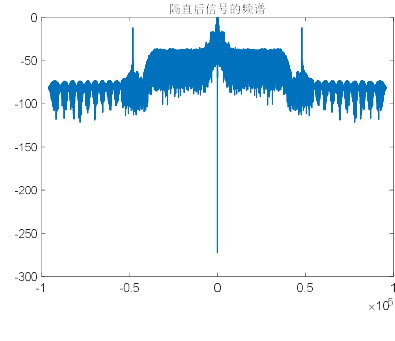
滤波后信号



滤波后信号频谱

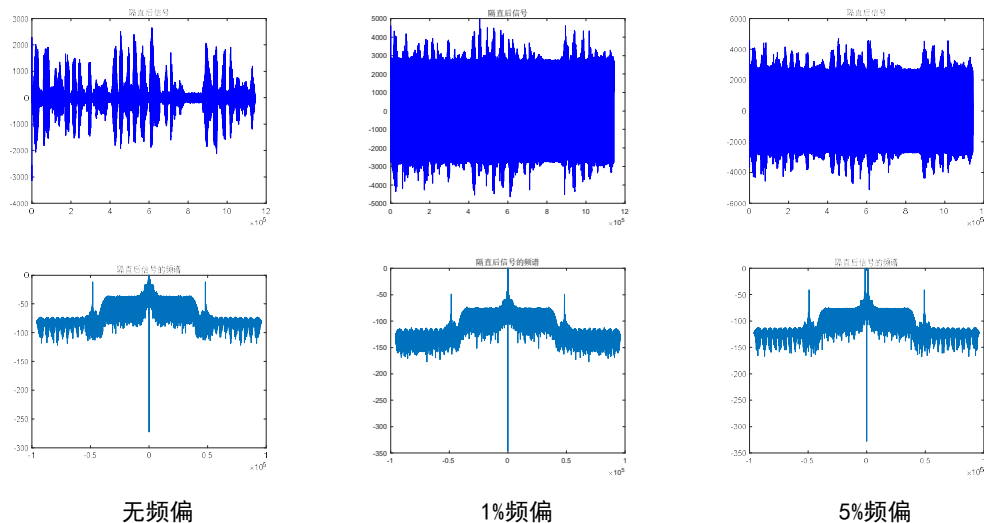


隔直后信号



隔直后信号频谱

2023-2024 学年第一学期 通信原理 仿真报告



根据上述三种情况对比, 随着频偏的增大, 信号质量明显下降, 相干解调的关键是接收端要提供一个与载波信号严格同步的相干载波, 否则可能导致严重失真。

```

%% 相干解调法
bias = 0; % 偏移量
for i=1:length(Sa_IF)
    xCos(i,1)=cos(2*pi*(1+bias)*f/Sampling_rate*(i-1));
    xSin(i,1)=sin(2*pi*(1+bias)*f/Sampling_rate*(i-1));
end
Sa_p = Sa_IF.*xCos;
fun_plot(Sa_p,50,'调制信号与相干载波相乘信号');
fun_analyzeSpectrum(double(Sa_p),Sampling_rate,51,'调制信号与相干载波相乘信号的频谱');
% 低通滤波器: 通带是采样率的 3/16; 过度带是采样率的 1/32, 抑制 40dB
FirLowPass_3_16_40dB =
[0.0115638194966750,.....,0.0115638194966750];
Sa_p_LP=filter(FirLowPass_3_16_40dB,1,Sa_p);
fun_plot(Sa_p_LP,60,'滤波后信号');
fun_analyzeSpectrum(double(Sa_p_LP),Sampling_rate,61,'滤波后信号的频谱');
Sa_p_LP_dcBlock = Sa_p_LP - mean(Sa_p_LP);
fun_plot(Sa_p_LP_dcBlock,70,'隔直后信号');
fun_analyzeSpectrum(double(Sa_p_LP_dcBlock),Sampling_rate,71,'隔直后信号的频谱');
sound(Sa_p_LP_dcBlock,audio_rate*upRate);

```