信号与系统课程实践报告

**题目：歌曲人声消除**

**班级：1802015**

**学号：18029100040**

**姓名：吴程锴**

# 内容与要求

本文通过歌曲立体声和滤波的知识实现对歌曲中人声的消除并较好地保留背景音乐。使用设计仿真程序，对不同歌曲进行测试，从而验证并实现了其可行性。

# 思路与方案

## 立体声消除人声

### 基本原理

假设有两个信号



当我们分别去听这两个信号时，听到的声音是完全一样的。当我们把这两个信号相加，得到一段新的音频信号



得到的就是一条幅值为零的直线，听不到任何声音。

### 通过左右两声道的音频消除人声

由2.1.1我们可以知道，两个振幅、频率完全相同的音频信号相位相差时，人耳听到的两者声音没有区别，并且把这两个振幅、频率完全相同且相位相差的两个音频信号相加后，声音被消除了，根据这个原理，我们可以消除歌曲中的人声。

对于一般的立体声歌曲，人能够感受到立体的原因是左右两个声道的伴奏在频率、振幅和相位上有区别，而人声在左右两个声道是基本一致的。

假设人声信号为，左声道的伴奏为，右声道的伴奏为，则左右声道的音频分别为



我们将左声道减去右声道得到新的左声道音频信号，再把右声道减去左声道得到新的右声道音频信号



对于单声道输出设备，新的左右声道的信号会相互消除，所以当使用单声道输出设备时，可以令，使声音正常输出。

## 设计带阻滤波器消除人声

### 基本原理

消除声音的另一种方法是把音频信号在某一段频率的能量使用带阻滤波器降为零，从而使原信号中某一段频率的声音消失。

### 通过设计带阻滤波器消除歌曲中的人声

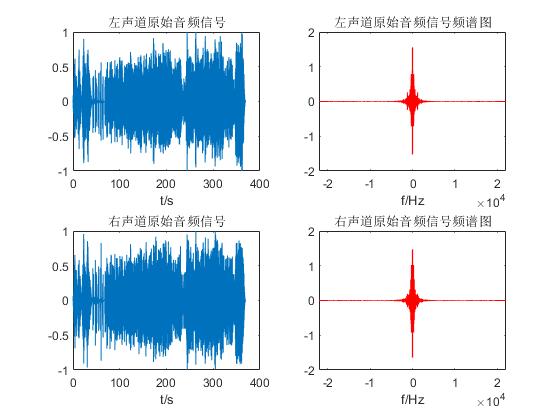
经过查阅的资料，男歌声的频率大约在之间，而女歌手的频率大约在之间，我们可以设计两个带阻滤波器，分别滤除这两个频率范围内的音频信号来消除人声。

# 成果及展示

使用设计仿真程序，完整代码见附录一。

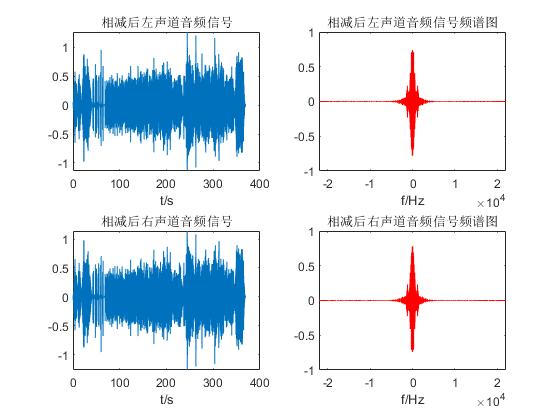
## 导入音频

使用中的函数导入一段音频，并差分左右声道，分别进行离散傅里叶变换，得到的左右声道时域和频域的结果如下图所示。



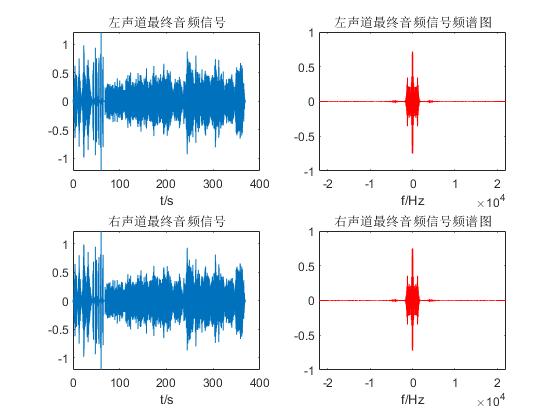
## 左右声道相减，消去人声

先左声道减去右声道得到新的左声道音频信号，再把右声道减去左声道得到新的右声道音频信号。得到新的左右两声道的音频信号的时域与频域结果如下图所示。



## 使用带阻滤波器消去人声

使用中的函数设计两个阶的带阻滤波器，信号的截至频率分别为和。再用函数对最新得到的左右声道音频信号使用刚刚得到的带阻滤波器进行滤波，滤波后得到的最终左右声道时域和频域的结果如下图所示。



## 最终效果

运行程序，对不同的歌曲进行人声消除，效果都十分显著，人声几乎消失，达到了要求。

# 总结与感想

本文通过左右声道相减和带阻滤波器的方式对歌曲中的人声进行消除，并用进行实现，实现了实践的要求。

通过本次实践，我对音频的原理有了更深的理解，对滤波器的使用有了经验，通过把书本中的知识运用到实际问题中，不仅对知识有了更加深刻的理解，还对信号与线性系统这门课有了更浓的兴趣。

# 参考资料

[1] 百度文库，<https://wenku.baidu.com/view/e2a1f44f647d27284b735194.html>；

[2] 百度文库，<https://wenku.baidu.com/view/7d64b3585acfa1c7aa00cc57.html>；

# 附录

##### 附录一

1. clc,clear
2. close all
3. %读取文件
4. music='music2';
5. [Sound,Fs] = audioread([music,'.mp3']); %读取音频 Fs采样频率
6. T=1/Fs; %采样间隔
7. N=length(Sound);
8. t=(0:N-1)\*T;    %时间 单位秒
9. % f=(0:N)\*Fs/length(Sound);   %频率 单位赫兹
10. f=(-N/2:N/2-1)\*Fs/N;    %频率
12. SoundL\_t=Sound(:,1);   %左声道
13. SoundR\_t=Sound(:,2);   %右声道
14. SoundL\_f=T\*fftshift(fft(SoundL\_t,N));
15. SoundR\_f=T\*fftshift(fft(SoundR\_t,N));
17. figure(1)
18. subplot(221)
19. plot(t,SoundL\_t);
20. title('左声道原始音频信号')
21. xlabel('t/s')
22. subplot(222)
23. plot(f,SoundL\_f,'r');
24. title('左声道原始音频信号频谱图')
25. xlabel('f/Hz')
26. subplot(223)
27. plot(t,SoundR\_t);
28. title('右声道原始音频信号')
29. xlabel('t/s')
30. subplot(224)
31. plot(f,SoundR\_f,'r');
32. title('右声道原始音频信号频谱图')
33. xlabel('f/Hz')
35. %参数设置
36. a1=1;
37. a2=-1;
38. b1=-1;
39. b2=1;
40. %双声道人声相互抵消
41. NewSoundL\_t=a1\*SoundL\_t+a2\*SoundR\_t;
42. NewSoundR\_t=b1\*SoundL\_t+b2\*SoundR\_t;
43. NewSound(:,1)=NewSoundL\_t;
44. NewSound(:,2)=NewSoundR\_t;
45. NewSoundL\_f=T\*fftshift(fft(NewSoundL\_t,N));
46. NewSoundR\_f=T\*fftshift(fft(NewSoundR\_t,N));
48. figure(2)
49. subplot(221)
50. plot(t,NewSoundL\_t);
51. title('相减后左声道音频信号')
52. xlabel('t/s')
53. subplot(222)
54. plot(f,NewSoundL\_f,'r');
55. title('相减后左声道音频信号频谱图')
56. xlabel('f/Hz')
57. subplot(223)
58. plot(t,NewSoundR\_t);
59. title('相减后右声道音频信号')
60. xlabel('t/s')
61. subplot(224)
62. plot(f,NewSoundR\_f,'r');
63. title('相减后右声道音频信号频谱图')
64. xlabel('f/Hz')
66. %滤波
67. BandStop1=fir1(200,2\*[150,600]/Fs,'stop');
68. SoundTemp=filter(BandStop1,1,NewSound);
69. BandStop2=fir1(200,2\*[1600,3600]/Fs,'stop');
70. SoundFinal=filter(BandStop2,1,SoundTemp);
72. SoundFinalL\_t=SoundFinal(:,1);
73. SoundFinalR\_t=SoundFinal(:,2);
74. SoundFinalL\_f=T\*fftshift(fft(SoundFinalL\_t,N));
75. SoundFinalR\_f=T\*fftshift(fft(SoundFinalR\_t,N));
77. figure(3)
78. subplot(221)
79. plot(t,SoundFinalL\_t);
80. title('左声道最终音频信号')
81. xlabel('t/s')
82. subplot(222)
83. plot(f,SoundFinalL\_f,'r');
84. title('左声道最终音频信号频谱图')
85. xlabel('f/Hz')
86. subplot(223)
87. plot(t,SoundFinalR\_t);
88. title('右声道最终音频信号')
89. xlabel('t/s')
90. subplot(224)
91. plot(f,SoundFinalR\_f,'r');
92. title('右声道最终音频信号频谱图')
93. xlabel('f/Hz')
95. %播放音乐
96. player = audioplayer(SoundFinal, Fs);
97. play(player);