乐音信号的合成与识别

班级：21 通信 2 班　　姓名：龚烨　　学号：2128410206

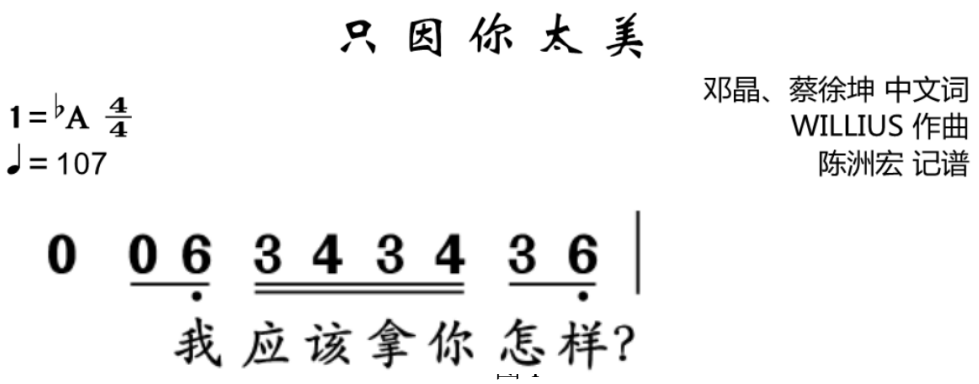
一、概述

随着社会的发展，声音信号在生活中的价值越来越大。用信号与系统分析语音可以更加有效地产生、传输、存储、获取和应用语音信息，这对于促进社会的发展具有十分重要的意义。用信号与系统分析语音可以实现语音的自动识别、机器合成以及语音感知等各种处理技术，这对于实现人机交互、智能控制、信息安全等领域有着广阔的应用前景。用信号与系统分析语音可以利用各种数学工具和方法，如傅里叶变换、拉普拉斯变换、离散时间傅里叶变换、Z 变换等，对语音信号的时域和频域特性进行深入研究，揭示语音信号的统计规律和结构特征。这也能使人们能更加有效地产生、传输、存储、获取和应用语音信息，这对于促进社会的发展具有十分重要的意义。接下来，对几段乐音信号进行处理来进行研究

二、乐曲的分析与演奏

2.1 乐曲中乐音的分析

我选择的第一个信号为歌曲《只因你太美》中的一个小节，如图 1 所示。



“我应该那你怎样”对应“la mi fa mi fa mi la”，调号“1= ♭A”对应降 A 大调，其中低音“la”对应音高 F，“mi”对应音高“C”，“fa”对应音高“♭D”。为了方便分析，我使用钢琴演奏了这一小节的旋律并保存为 audio.wav。通过 Matlab 程序可以得到整段音频的时域波形、幅度谱、相位谱。

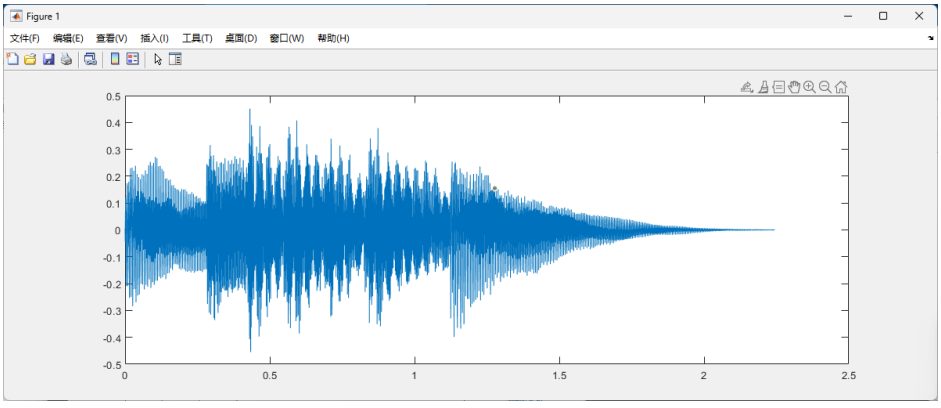


图 2 时域波形

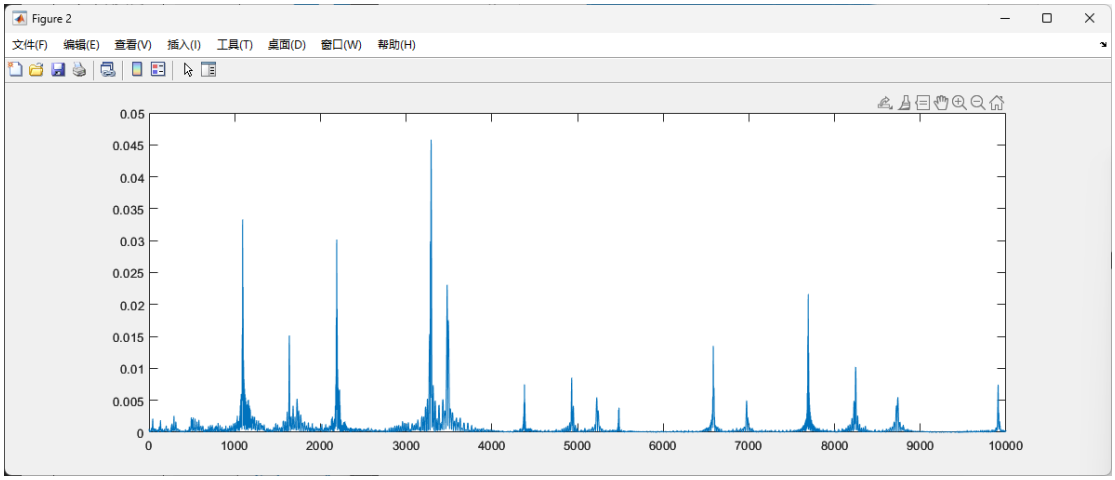


图 3 幅度谱

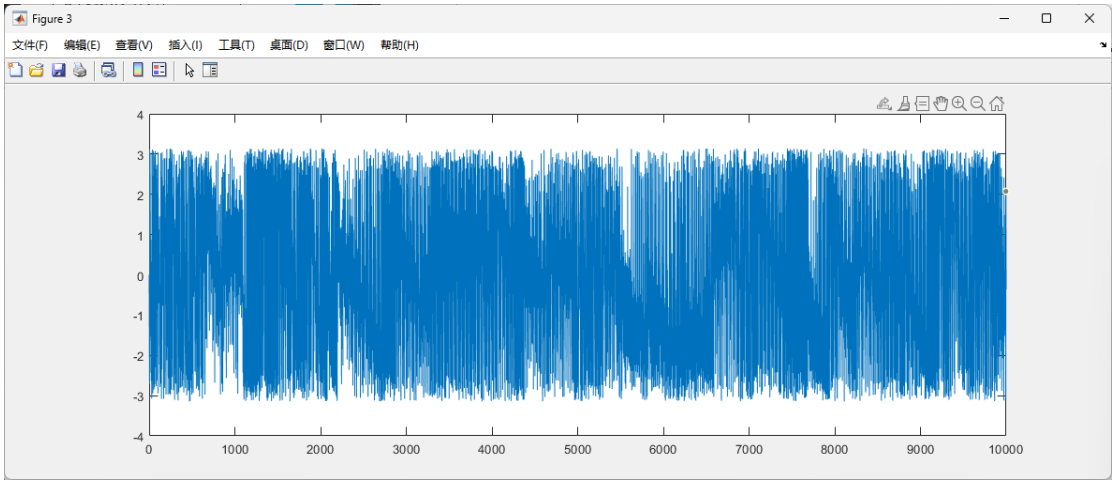


图 4 相位谱

使用的代码如下：

```
[f,Fs] = audioread('audio.wav');
close all
f1 = f(:,1); % 双声道，取其中之一
f1 = f1';
dt = 1/Fs; %采样点间隔
L = length(f);  % 信号总长度
start = 1;
ending = L;
t = (start:ending)*dt;  %实际的时间范围
w1=0:10000;
F=f1*exp(-1i*t'*w1)*dt;  % 连续傅里叶变换，用求和近似积分，加法隐含在矩阵乘法中
figure(1)
plot(t,f1);   % 时域信号
```

```
figure(2)
plot(w1,abs(F)); % 幅度谱 %横坐标为 f
figure(3)
    plot(w1,angle(F)); %相位谱
```

可以看到，由于原信号较为复杂，幅度谱难以辨认。因此，对原信号进行裁剪。第一个“la”音的长度为 266ms，因此将 L 设置为$0.266 * 44100 = 11730$。得到幅度谱如图 5 所示。

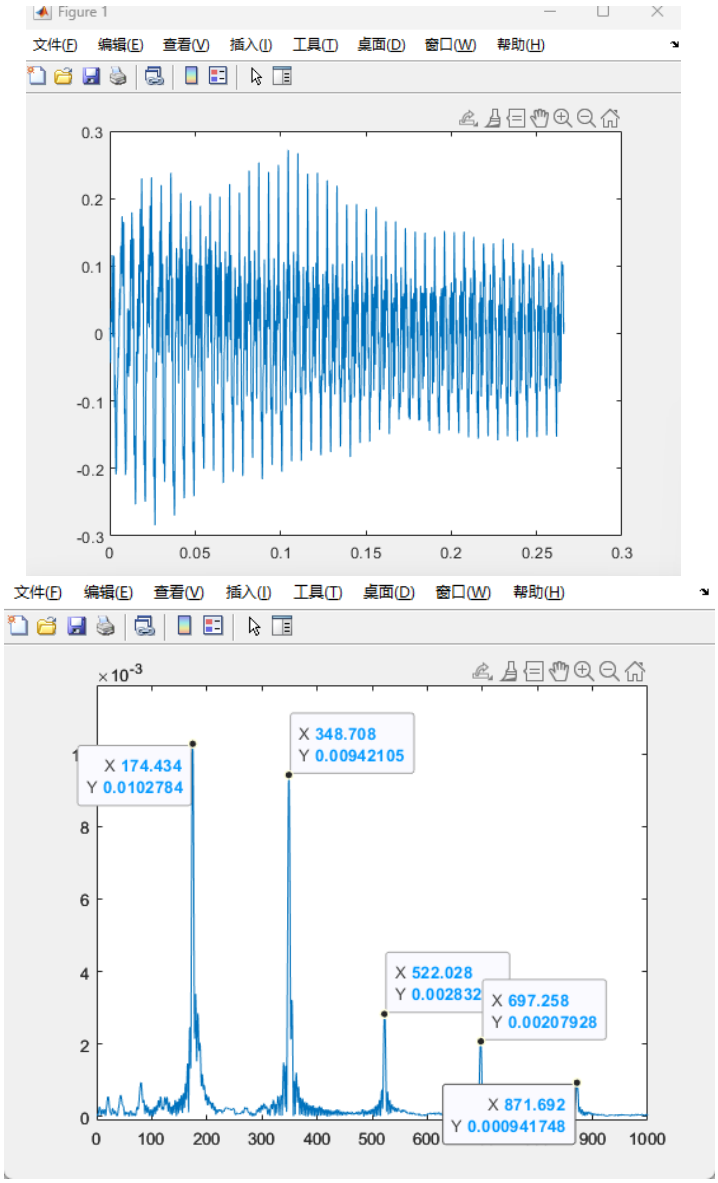


图 5

使用的代码如下：

```
[f,Fs] = audioread('audio.wav');
close all
f1 = f(1:11730,1); % 双声道，取其中之一
```

```
f1 = f1';
dt = 1/Fs; %采样点间隔
L = 11730;  % 信号总长度
start = 1;
ending = L;
t = (start: ending)*dt;  %实际的时间范围
w1=0:1000 * 2 * pi;
F=f1*exp(-1i*t'*w1)*dt;  % 连续傅里叶变换，用求和近似积分，加法隐含在矩阵乘法中
figure(1)
plot(t,f1);   % 时域信号
figure(2)
plot(w1 / 2 / pi, abs(F)); % 幅度谱 %横坐标为 f
```

可以看到，对于钢琴来说，大多由一个基波与它不同幅度的谐波组成，而基波的幅度最大，$f_1 = 174Hz$ 。

而根据公式

$$freq = 440 \times 2^{(keynum-49)/12}$$

代入 $keynum - 49 = -16$ 可以计算出 $freq \approx 174Hz$ ，与实验结果一致。

而只考虑 1000Hz 范围以内，可以看到钢琴的 F 音有 5 个分量，其幅度之比为 $A_1 : A_2 : A_3 : A_4 : A_5 \approx 103 : 94 : 28 : 21 : 9 \approx 11 : 10 : 3 : 2 : 1$。因此，只要我们根据公式算出每一个音的基波频率，再根据上面的比例，就能粗略地产生一个钢琴的声音。

2.2 乐曲中乐音的产生

通过 Matlab 中的 sound()函数，可以将一个信号以声音的形式播放出来。通过以下代码，可以产生一个采样率为 44100Hz、时长 1 秒、由 5 个分量组成的信号并播放，波形图如图 6 所示。

```
x = 0: 1 / 44100: 1;
F = 11 / 25 * sin(2 * pi * 174 * x) + ...
    10 / 25 * sin(2 * pi * 174 * 2 * x) + ...
    3 / 25 * sin(2 * pi * 174 * 3 * x) + ...
    2 / 25 *sin(2 * pi * 174 * 4 * x) + ...
    1 / 25 * sin(2 * pi * 174 * 5 * x);
plot(x, F), grid on
sound(F, 44100);
```

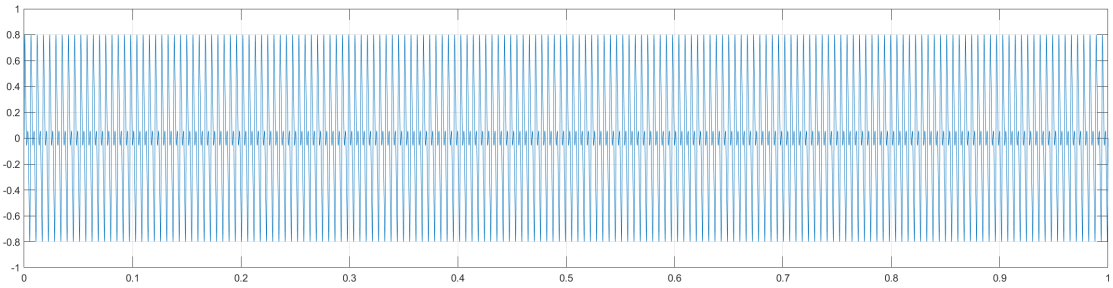


图 6

由声音可知，虽然产生的信号与钢琴原本的音色大相径庭，但是音调相同，也有部分相似之处。只要调整音调，就能产生不同音高的“钢琴”声。

通过计算，我们可以得到 $f_{1C}=262Hz$， $f_{1bD}=277Hz$，同时，原歌曲的 BPM 为 107，计算可知八分音符时长为 0.28s，十六分音符时长为 0.14s。

因此，C 和 bD 的表达式分别为：

```
C = 11 / 25 * sin(2 * pi * 262 * x) + ...
    10 / 25 * sin(2 * pi * 262 * 2 * x) + ...
    3 / 25 * sin(2 * pi * 262 * 3 * x) + ...
    2 / 25 *sin(2 * pi * 262 * 4 * x) + ...
    1 / 25 * sin(2 * pi * 262 * 5 * x);
bD = 11 / 25 * sin(2 * pi * 277 * x) + ...
    10 / 25 * sin(2 * pi * 277 * 2 * x) + ...
    3 / 25 * sin(2 * pi * 277 * 3 * x) + ...
    2 / 25 *sin(2 * pi * 277 * 4 * x) + ...
    1 / 25 * sin(2 * pi * 277 * 5 * x);
```

再通过对音符长度的计算，对这三个信号进行切割，使其符合原歌曲中每一个音符的长度。最后通过函数出来

最终得到的信号如图 8 所示。

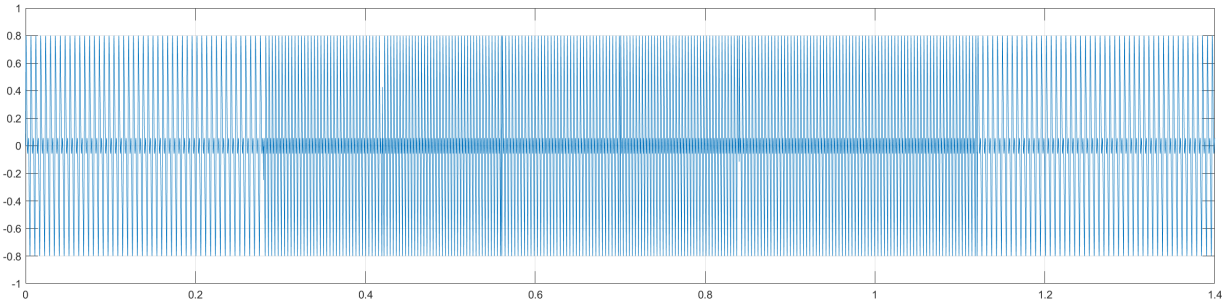


图 8

完整代码如下所示

```
x = 0: 1 / 44100: 1.4;
xx = 0: 1 / 44100: 0.28;
```

```
F = 11 / 25 * sin(2 * pi * 174 * x) + ...
    10 / 25 * sin(2 * pi * 174 * 2 * x) + ...
    3 / 25 * sin(2 * pi * 174 * 3 * x) + ...
    2 / 25 *sin(2 * pi * 174 * 4 * x) + ...
    1 / 25 * sin(2 * pi * 174 * 5 * x);
C = 11 / 25 * sin(2 * pi * 262 * x) + ...
    10 / 25 * sin(2 * pi * 262 * 2 * x) + ...
    3 / 25 * sin(2 * pi * 262 * 3 * x) + ...
    2 / 25 *sin(2 * pi * 262 * 4 * x) + ...
    1 / 25 * sin(2 * pi * 262 * 5 * x);
bD = 11 / 25 * sin(2 * pi * 277 * x) + ...
    10 / 25 * sin(2 * pi * 277 * 2 * x) + ...
    3 / 25 * sin(2 * pi * 277 * 3 * x) + ...
    2 / 25 *sin(2 * pi * 277 * 4 * x) + ...
    1 / 25 * sin(2 * pi * 277 * 5 * x);
F(x >= 0.28 & x < 1.12) = 0;
C(x < 0.28 | (x >= 0.42 & x < 0.56) | (x >= 0.7 & x < 0.84) | x >= 1.12) =
0;
bD(x < 0.42 | (x >= 0.56 & x < 0.7) | x >= 0.84) = 0;
res = F + C + bD;
plot(x, res), grid on
sound(res, 44100);
audiowrite("out.wav", res, 44100);
```

三、总结与反思

通过上面的实验，我使用 Matlab 成功地识别与合成了一段钢琴音乐。通过对最开始的钢琴信号进行傅里叶变换，可以找到基波分量的频率，从而用公式算出音高。而如果已知音高，也可以通过用正弦信号合成的方式得到类似钢琴的信号，从而合成乐音。然而，由于对谐波分量的数量选择过少，且幅度过于近似，最后合成得到的音色与钢琴相差过大，可以尝试增加谐波分量的数量，以实现合成更接近于钢琴的乐音。