清华大学电子工程系

matlab 高级编程与工程应用

2021-2022 学年夏季学期

大作业 1

赵英竹 2022 年 7 月 31 日

---

1 简单的合成音乐

(1) 计算出《东方红》片断中各个乐音的频率，在 MATLAB 中生成幅度为 1 、抽样频率为 8kHz 的正弦信号表示这些乐音。请用 sound 函数播放每个乐音，听一听音调是否正确。用这一系列乐音信号拼出《东方红》片断，用 sound 播放你合成的音乐，听起来感觉如何？

5 对应频率为 523.25Hz，6 对应频率为 587.33Hz，2 对应频率为 392Hz，1 对应频率为 349.23Hz，低音 6 对应频率为 293.66Hz。

matlab 核心代码如下：(本题对应文件 music1_1.m)

```
for i = 1:length(sheet_notes)
    if sheet_basic(i) == 0
        fre = notes_F(sheet_notes(i));
    else
        fre = notes_F_b(sheet_notes(i));
    end
    t = [0:1/Fs:sheet_time(i)];
    apmusic = sin(2 * pi * fre *t) ;
    %sound(apmusic,Fs)
    music = [music,apmusic];
end
```

提前设置好乐谱（sheet_basic 和 sheet_notes 存音调，sheet_time 存节拍），之后根据乐曲的频率和时长生成对应的正弦波，播放乐音，拼出《东方红》片段。

生成的音乐每个音都符合对应的音高，但是有些不自然。

(2) 为了消除乐音邻接处噪声，用包络修正每个乐音，以保证在乐音的邻接处信号幅度为零。 matlab 核心代码如下：（本题对应文件 ADSR_1.m,music1_2.m,music1_2_2.m）

```
function y = ADSR_1(x)
len = length(x);
```

```
y = zeros(1,len);
for i = 1:len
    if i < len/6
        y(i) = 1.2/(x(len)/6)*x(i);
    elseif i < len/3
        y(i) = -0.2/(x(len)/6)*(x(i)-x(len)/6)+1.2;
    elseif i < 2*len/3
        y(i) = 1;
    else
        y(i) = 1- 1/(x(len)/3)*(x(i)-2*x(len)/3);
    end
end
```

```
    apmusic = sin(2 * pi * fre *t) .* ...
        exp(-3*t/sheet_time(i));
```

```
    y = ADSR_1(t);
    apmusic = sin(2 * pi * fre *t).*y;
```

在原有的正弦波形基础上乘上包络。music1_2.m 的包络是指数衰减。music1_2_2.m 的包络是 ADSR_1.m 文件中函数，包络图形如下：

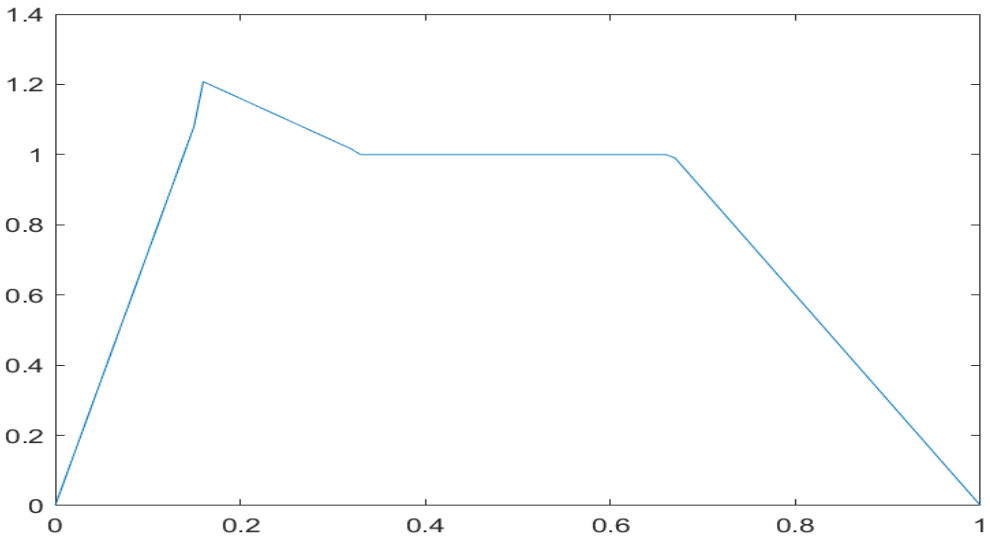

图 1: $\mathrm{ADSR}_1.m$

加包络之后的乐音基本听不到邻接处的噪声。加指数衰减包络听到的声音更清脆，加 ADSR 包络的声音更接近于长萧声。

(3) 请用最简单的方法将 (2) 中的音乐分别升高和降低一个八度。再难一些，请用 resample 函数将上述音乐升高半个音阶。

改变音乐播放的采样率实现快放和慢放，进而将音乐升高或者降低一个八度。用 resample 函数进行重采样，在音乐播放的采样率不变的情况下，采样更多的点意味着慢放，采样更少的点意味着快放，而且此时的音质损失较小。

matlab 核心代码如下：(本题对应文件 music1_3.m,music1_3_2.m)

```
sound(music,Fs/2) %低
sound(music,Fs*2) %高
```

```
music = resample(music,106,100);
sound(music,Fs)%低
music = resample(music,100,106);
sound(music,Fs)%高
```

文件 music1_3.m 中通过调整音频播放时的采样率来升高和降低一个八度,music1_3_2.m 通过重采样来升高和降低一个音阶。通过重采样后音质更加理想。

(4) 试着在 (2) 的音乐中增加一些谐波分量，听一听音乐是否更有“厚度”了

在原来音乐的基础上加上二次谐波，三次谐波，加上包络。

matlab 核心代码如下：(本题对应文件 music1_4.m)

```
t = [0:1/Fs:sheet_time(i)];
y = ADSR_1(t);
apmusic = (sin(2 * pi * fre *t) + 0.2*sin(4 * pi * fre *t) ...
    + 0.3*sin(6 * pi * fre *t)).*y;
```

选择基波幅度为 1, 二次谐波幅度 0.2, 三次谐波幅度 0.3, 加上 ADSR_1 文件中的包络，听起来确实有些像手风琴。

(5) 自选其它音乐合成

用第（4）题的谐波分量系数和第（2）题中类似萧声的包络演奏《送别》的前四个小节

matlab 核心代码如下：(本题对应文件 music1_5.m)

```
notes_F = [349.23, 392, 440, 466.16, 523.25,  587.33, 659.25,698.80];
notes_F_b = [174.61, 196, 220, 233.08, 261.63, 293.66, 329.63];
sheet_notes = [5,3,5,8,6,8,6,5,5,1,2,3,2,1,2];
sheet_time = [0.5, 0.25, 0.25, 1, 0.5, 0.25, 0.25, 1,0.5,0.25,0.25,0.5,
```

演奏出来声音有些像号声。

2 用傅里叶级数分析音乐

(6) 先用 wavread 函数载入光盘中的 fmt.wav 文件，播放出来听听效果如何？是否比刚才的合成音乐真实多了？

matlab 更新之后的版本不支持 wavread 函数，按照提示使用了 audioread 函数

matlab 核心代码如下：(本题对应文件 music2_6.m)

```
    music = audioread('fmt.wav');
    plot(music)
    sound(music,8000)
```

运行结果如图：

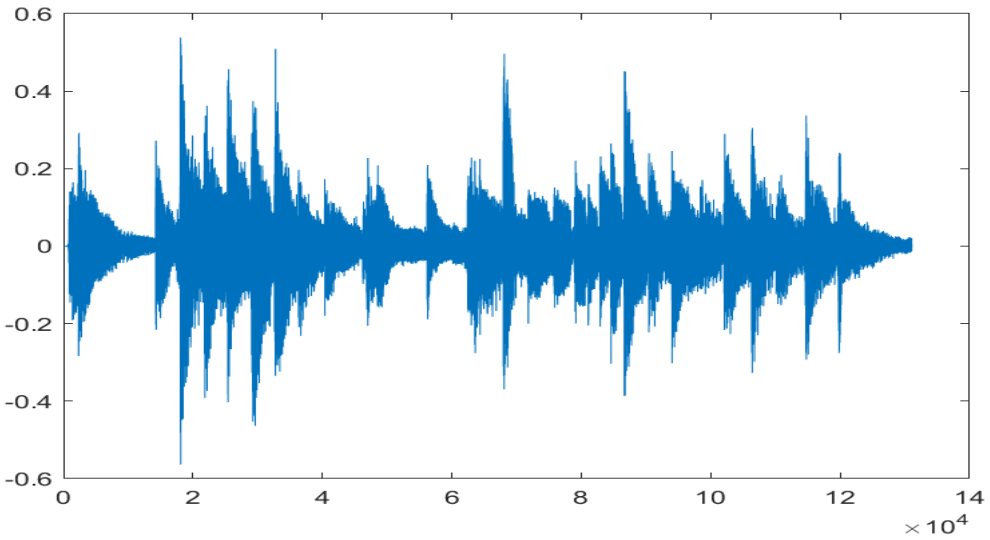

图 2: 音乐波形

播放出来的音乐是一段吉他演奏的乐曲。

(7) 从真实值 realwave 中得到待处理的 wave2proc。

题目中提到预处理过程可以去除真实乐曲中的非线性谐波和噪声，图中有多个周期，周期之间存在一些噪声误差，可以用多个周期求平均来消除。在处理之前可以通过升采样提高精度，结束后降采样恢复。
matlab 核心代码如下：(本题对应文件 music2_7.m)
计算各个峰值之间的间距以及峰值的个数，得到周期和该段音频周期的个数。

```
[pks,locs] = findpeaks(realwave,'minpeakheight',0.15);
inter = 0;
for i = 2:length(locs)
    inter  = inter + locs(i) - locs(i-1);
end
inter = round(inter/(length(locs)-1));
times = length(realwave)/inter;
```

将各个周期相加取平均得到一个周期去噪声之后的结果，然后重复十次。

```
for i = 1:243
    for j = 0:times-1
    average(i) = average(i) + realwave(i+j*inter);
    end
    average(i) = average(i) / times;
end
new_realwave = repmat(average,times,1);
```

将处理后的结果和给定的 wave2proc 画在同一张图上：

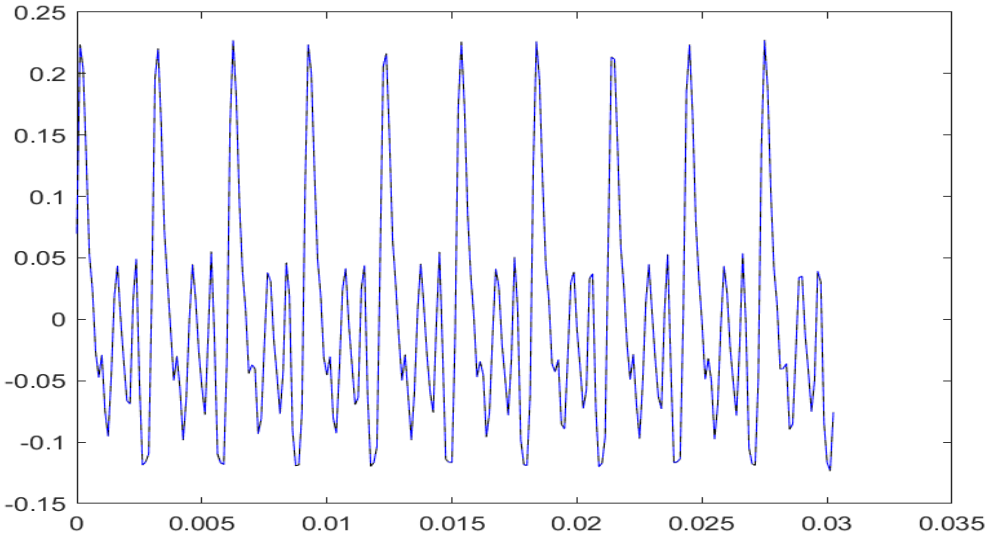

图 3: 处理后的 realwave 和 wave2proc

可以看出处理后的结果和 wave2proc 的波形比较一致。

(8) 这段音乐的基频是多少？是哪个音调？请用傅里叶级数或者变换的方法分析它的谐波分量分别是什么。

将这段音乐重复五次，之后进行傅里叶变换得到信号的频谱, 然后选取峰值和它们对应的频率，也即得到了基频和谐波分量。之后用基频在 note_dict 中查找音调。

matlab 核心代码如下：(本题对应文件 music2_8.m,note_dict.m) locs 是峰值对应的频率，pks 是对应的数值（除基频进行归一化），此部分代码参照 matlab 官方文档。

```
wave2proc = repmat(wave2proc,5,1);
y = fft(wave2proc);
f = (0:length(y)-1)*Fs/length(y);
y = abs(y);
[pks,locs] = findpeaks(y,'minpeakheight',10);
locs = (locs-1) * Fs /length(y);
pks = pks/pks(1);
```

输出结果如下：

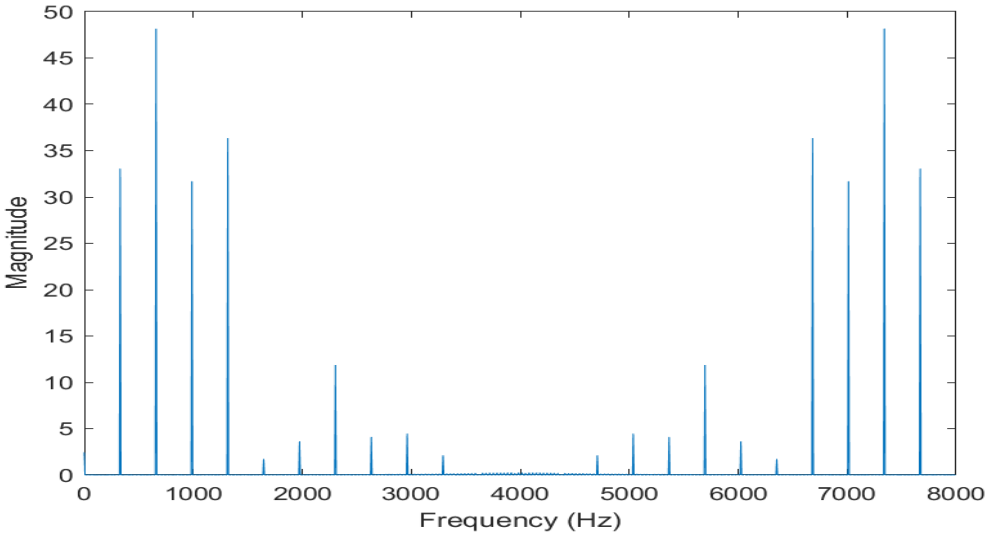

图 4: 信号的频谱

输出的音调信息：

```
>> music2_8

ans =

    'E4'

>>
```

图 5: 命令行输出

基频为 329.2Hz，是 E4 调。各谐波分量强度存在 pks 向量中，若基频强度为 1，二次谐波为 1.4572，三次谐波强度为 0.9587。不过我对这个结果有些怀疑，二次谐波和三次谐波幅度看起来有些盖过基频。

(9) 再次载入 fmt.wav，写一段程序，自动分析出这段乐曲的音调和节拍。

先手动标记出每个音起止的时间点，根据时间点算出每个音持续的时间，得到每个音的节拍。之后选取每个音的前 700 个采样值重复上一题的操作得到音调和谐波分量。

matlab 核心代码如下：(本题对应文件 music2_9.m,note_dict.m)

手动标记起止时间：

```
music_begin = [2392,14290,18100,22170,25380,29040,...
    32750,36310,40370,48570,56260,62480,68010,...
    71790,75810,79010,81190,82910,84630,86660,90470,...
```

```
94080,102200,106300,114700,119800,131072];
```

将节拍取成 0.5 的整倍数:

```
note_time = round((music_begin(i+1)-music_begin(i))/8000*2)/2;
```

最终得到的曲谱存在变量 note_sheet 中，谐波分量只取到四次。最终得到的曲谱如下：从左至右依次为音调、谐波、谐波强度、节拍（0.5 的倍数）

变量 - note_sheet

note_sheet

26x4 cell

	1	2	3	4	5	6
1	'A3'	[216.8331;...	[1;0.3508;0.3508;1]	1.5000		
2	'G3'	[194.0086;...	[1;3.4412;1.1650;1.1650]	0.5000		
3	'G3'	[194.0086;...	[1;2.5969;0.8838;0.3595]	0.5000		
4	'A3'	[216.8331;...	[1;0.8939;1.8024;1.7031]	0.5000		
5	'C4'	[262.4822;...	[1;1.9848;2.7424;1.7606]	0.5000		
6	'G3'	[194.0086;...	[1;0.2894;0.1918;0.1230]	0.5000		
7	'F3'	[171.1840;...	[1;0.3377;1.1062;0.3670]	0.5000		
8	'F3'	[171.1840;...	[1;0.6626;0.4776;0.4776]	0.5000		
9	'F3'	[171.1840;...	[1;1.1988;0.8250;0.8250]	1		
10	'B3'	[251.0699;...	[1;0.8454;0.8454;1]	1		
11	'E4'	[330.9558;...	[1;1.2997;0.9137;0.9137]	1		
12	'A3'	[216.8331;...	[1;1]	0.5000		
13	'A3'	[216.8331;...	[1;1.3835;2.1536;1.5184]	0.5000		
14	'A4'	[445.0785;...	[1;0.8380;0.8380;1]	0.5000		
15	'A3'	[216.8331;...	[1;1]	0.5000		
16	'C4'	[262.4822;...	[1;1.3234;1.2790;0.8488]	0.5000		
17	'F4'	[353.7803;...	[1;1]	0		
18	'E4'	[330.9558;...	[1;1.8588;1.1022;0.9450]	0		
19	'D4'	[296.7190;...	[1;0.7920;0.6147;0.6147]	0.5000		
20	'bA3'	[205.4208;...	[1;3.4092;1.5848;3.8381]	0.5000		
21	'B3'	[251.0699;...	[1;0.5206;0.5206;1]	0.5000		
22	'B3'	[251.0699;...	[1;1.2038;0.7505;0.7505]	1		
23	'F3'	[171.1840;...	[1;0.9391;0.9391;1]	0.5000		
24	'F3'	[171.1840;...	[1;3.9165;1.1505;1.2782]	1		
25	'G3'	[194.0086;...	[1;2.7958;0.7217;0.7217]	0.5000		
26	'bA3'	[205.4208;...	[1;0.4076;0.4076;1]	1.5000		
27						
28						

图 6: 得到的曲谱

最终结果基本分析出了该段音乐的音调和节拍，但在听这段音乐的时候，有些琶音，每个音时值较短，没能完整地分析出来。而且对于同一个音谐波分量分析也有些差异。

3 基于傅里叶级数的合成音乐

(10) 用 (7) 计算出来的傅里叶级数再次完成第 (4) 题，听一听是否像演奏 fmt.wav 的吉他演奏出来的

用计算得到的谐波强度生成正弦波，同时我仿照吉他的包络写了一个近似的版本。由于给的吉他音域较低，这里的《东方红》乐曲降了一个八度。matlab 核心代码如下：(本题对应文件 music2_10.m,ADSR_2.m) 基频强度为 1，二次谐波为 1.4572，三次谐波强度为 0.9587, 包络为 ADSR_2.m 文件

```
1       y = ADSR_2(t);
2       apmusic = (sin(2 * pi * fre *t)+1.4572*sin(4 * pi * fre *t)+0.9587*s
3           +1.0999*sin(8 * pi * fre *t)).*y;
```

ADSR_2 文件中的包络如下图：

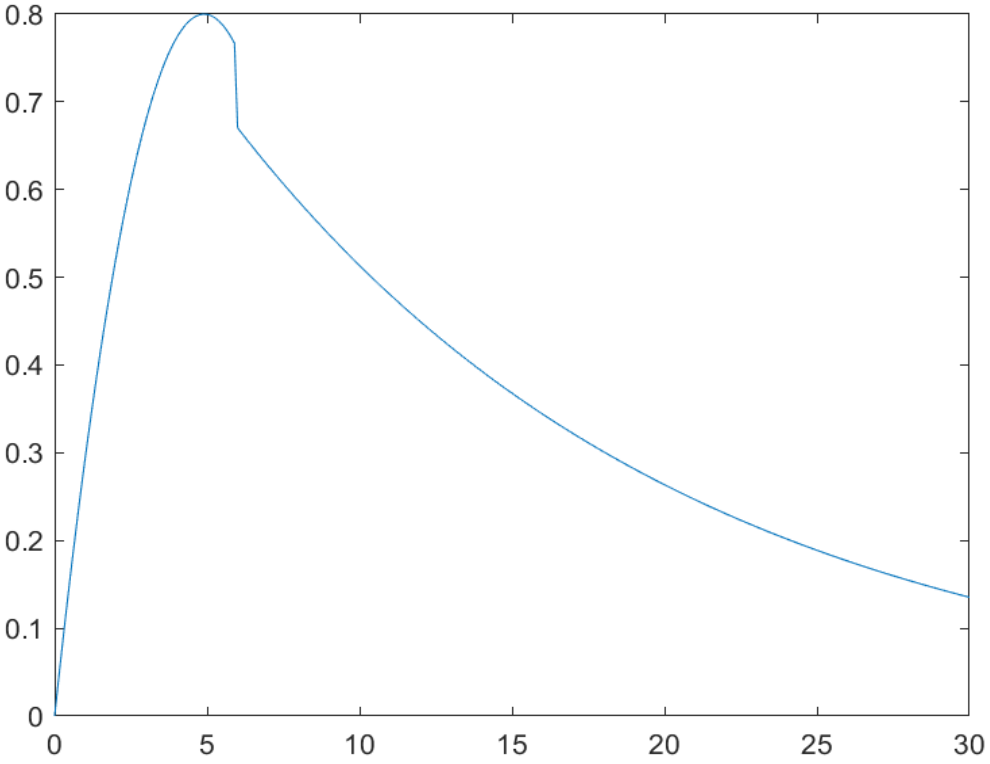

图 7: ADSR_2 中的包络图形

最终得到的音乐波形如下图:

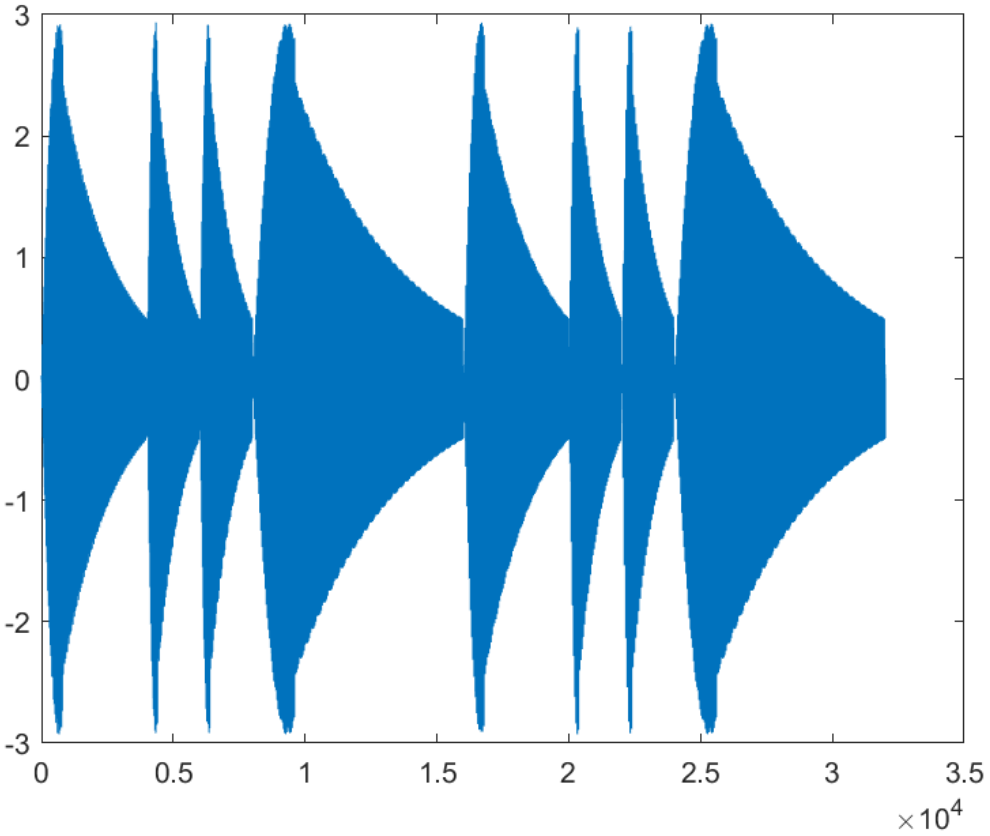

图 8: 得到的音乐

听到的音乐比最初地版本接近吉他声音，但是不是很像，可能是包络选取的问题。

(11) 已经掌握了每个音调对应的傅里叶级数，大致了解了这把吉他的特征，演奏一曲《东方红》

本题对应文件 music3_11.m,ADSR_2.m

最终得到的音乐波形如下图：

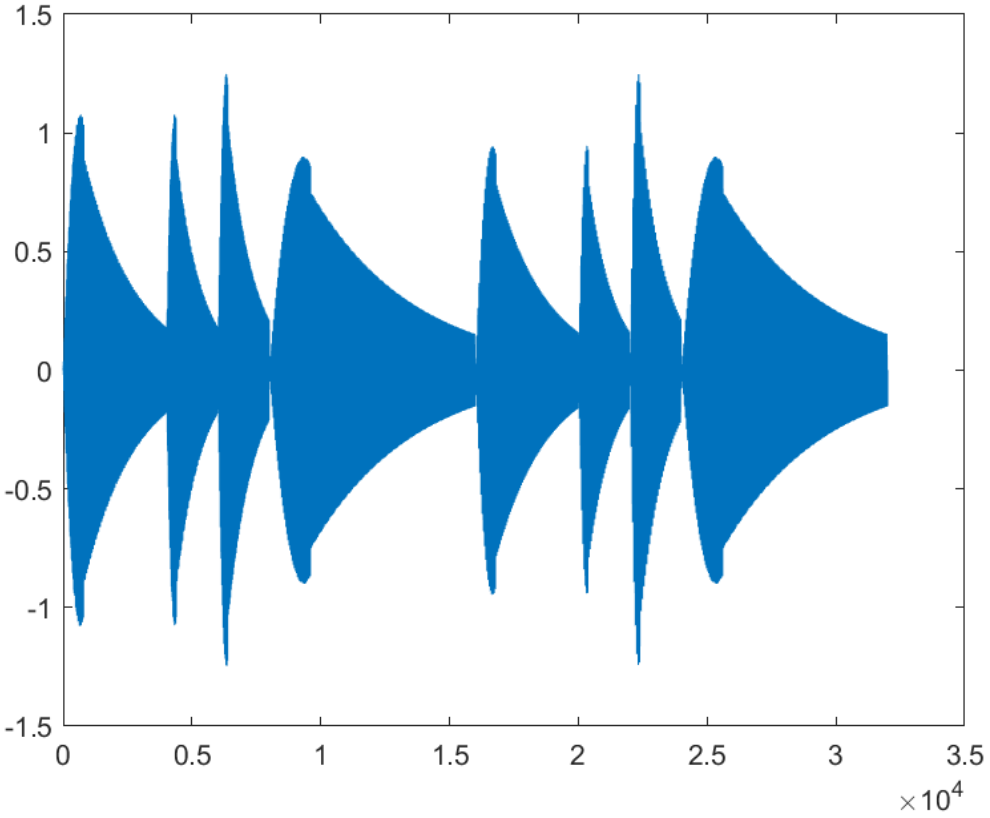

图 9: 得到的音乐波形

演奏出来的效果要优于上一问得到的结果，但仍然与真实演奏的吉他声有差距。

4 代码列表

第一题 music1_1.m
第二题 music1_2.m,music1_2_2.m,ADSR_1.m
第三题 music1_3.m,music1_3_2.m,ADSR_1.m
第四题 music1_4.m,ADSR_1.m
第五题 music1_5.m,ADSR_1.m
第六题 music2_6.m
第七题 music2_7.m
第八题 music2_8.m,note_dict.m
第九题 music2_9.m,note_dict.m
第十题 music3_10.m,ADSR_2.m

第十一题 music3_11.m,ADSR_2.m
（ADSR_1,ADSR_2 是包络，note_dict 是频率对应的音名）