华中科技大学光学与电子信息学院

《信号与系统》课程

工程设计问题设计报告

题目：卷积混响系统设计与实验验证

报告撰写说明

1. 按照参考模板的内容和格式撰写报告
2. 理论模型部分须结合本课程知识分析问题、建立模型
3. 程序设计部分应给出设计思路、主要流程图和关键函数的说明；结果分析不能只是简单给出结论，应结合具体问题，对关键参数或算法在不同取值条件下对结果的影响情况进行分析和总结。如果可能，还应进行误差分析
4. 组内互评分 A、B、C、D 四个贡献等级，最终评价应区分出前三个等级

目 录

1 问题描述

混响是声源发声停止后声音继续存在的声学现象。比如室内，声音发出后被墙壁等障碍物阻碍并反射，会产生混响现象。混响现象在自然界中非常常见，只要有声音传播和障碍物存在，就会产生混响。不同材质、室内户外、周围的物体、空间的大小等都会影响混响的发生。

在现实的环境中，比如会议室、歌剧院、森林、甚至管道中，用麦克风记录下实际的声音混响效果，然后保存为各种场景的 IR（单位冲激响应）文件。后期采用混响卷积系统，导入 IR 文件，就能模拟实际环境的混响效果。

卷积混响在音频处理领域有着广泛的应用。通过卷积混响，声音信号可以产生更加真实和自然的混响效果，从而提升声音的听觉体验。

本项目要求：

1. 设计用 MATLAB 设计一个卷积混响系统，可以选择不同场景的 IR 文件，利用卷积对声源信号模拟对应场景的混响效果。

2. 设计并制作一个实验采集装置，记录并保存 3 个不同混响效果场景的 IR 文件。在此基础上，采用项目设计的卷积混响系统，对比演示声音混响效果。

3. 设计一类卷积混响方法和实验系统，在产生混响效应的情况下，赋予声音不同的材质感。如金属、木器、玻璃等音质的混响效果。

2 理论模型

2.1 原理分析与设计思路

从物理学角度看，混响是声波在室内遇到障碍物后反射、折射和散射，形成的一系列连续的回响。当声源停止发声后，由于声音的多次反射，听音者会听到一组不断重复且密集的多重反射声，这些声音逐渐混合、衰减到无声的状态，这种现象就被称为混响。

从信号声学特性看，混响是室内声源停止发声后，由于房间边界面或障碍物使声波多次反射或散射产生延续的现象。它是室内有界空间的一个重要声学特性，对建筑物的音质有重要影响。

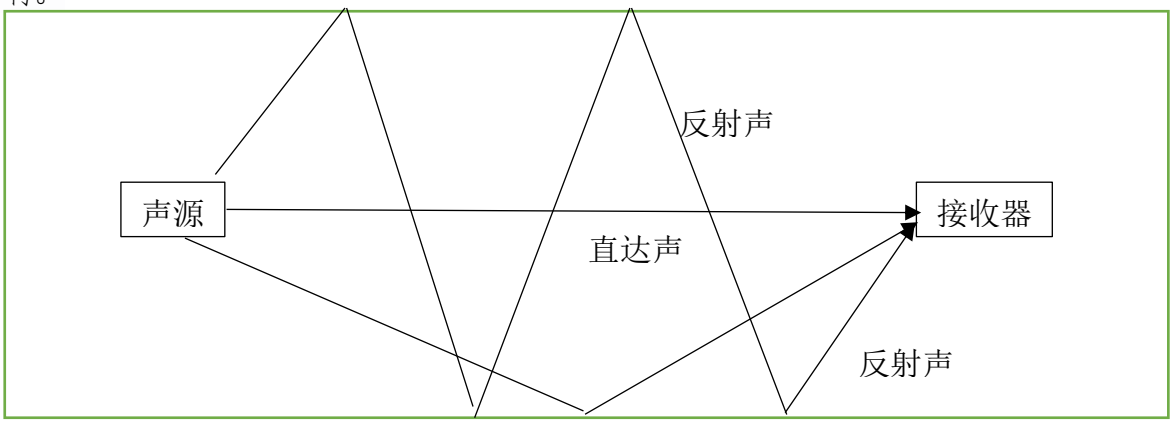


图 1. 混响原理图

脉冲响应是指在一个声学空间中，录制一个短而尖锐的声音（理论为单位冲激信号）所得到的反射脉冲。这个脉冲响应包含了该声学空间的混响特征，即直达声和后续各个反射声之间的时间和能量关系。

卷积是一种被广泛应用的数学处理方式，其应用范围包括统计学、图像处理和音频处理等领域。在音频处理中，卷积计算被用来模拟现实中的混响效果。具体来说，就是将输入信号（需要加入混响的声音）和脉冲响应进行卷积计算，得到的结果就是输入信号获取了脉冲响应中关于空间中的声学特性后的信号，即卷积混响。

在本问题中，我们分别利用拍手声音和 MATLAB 制作的单位冲激信号近似作为脉冲信号，分别在光电大楼楼梯、光电大楼 C117 以及寝室播放并录制单位脉冲响应，通过对比与已知语音信号卷积混响的效果，滤波调节优化单位脉冲响应信号，最终得到卷积混响后的语音信号。

同时，我们利用 MATLAB 中 APP Designer 功能设计了卷积混响系统的软件界面，通过绘制原声、IR 文件和混响语音的时域和频域波形，直观反映声音的混响效果。

在赋予声音材质感时，我们录制了敲击木器、金属和玻璃的单位冲激响应，通过滤波预处理消除杂波，与原语音卷积混响，赋予声音 3 种不同的材质感。

根据混响特性（如不同频率的衰减速度、密度、频率的特性等）不同，一般将混响分为以下几种类型：$rooms$，$halls$，$plates$，$ambience$，$spaces$，$chambers$。它们的特征如下：

1. $rooms$类型：自然界中最为常见，特性为边界感明显，可以给声音加上明显的空间感，使得声音产生距离。
2. $halls$类型：类似于声音墙体反射效果，声音扩散充分，延音明显。加在声音上一般不太影响声源的清晰度，但会在声源背后产生一个边界不明显的大空间。
3. $plates$类型：非自然的混响类型。其原型是让声音进入金属板传导产生的反射回馈。特性是很亮，早期反射密度高，尾音衰减快。听上去的感觉如果$decay\ time$短，会有点像比较亮且边界感不明显的$rooms$；$decay\ time$长则有点像尾音不太一样的比较亮的$halls$。
4. $ambience$类型：环境混响，也可翻译为氛围混响。这种类型一般可以达到一些常规几种类型达不到的特殊效果比如特别大的空间、特别长的尾巴，或者特殊空间感。
5. $spaces$类型：较为特殊的声学空间，声学性质各异。
6. $chambers$类型：与$halls$类型比较类似，其原型的声学空间往往更大，边界感更加不明显。

我们录制的 IR 文件中，C117 混响属于$chambers$类型的IR文件，楼道混响属于$spaces$类型的IR文件，寝室混响属于$rooms$类型的IR文件，而三种不同的材质效果中，金属质感属于$plates$类型的 IR 文件，木器和玻璃质感属于$ambience$类型的IR文件。

2.2 数学模型

由图 1 可知，假如声源和接收器在特定的声学空间中，接收器接收的信号包括直达声外，还包含相当多的反射声。如果声学空间非常大，可以近似不存在反射声，仅能接收直达声；如果空间反射强度非常小，空间的墙壁会吸收反射声的大部分能量，从而也可以近似反射声，因此我们重点关注声学空间的空间大小和反射强度两个要素。

假设声源输入信号为$x(t)$，我们将所有经过一次反射的声音进行叠加，该叠加结果总的效果为时域上延时为m，幅度存在在通常情况下对低频频率分量衰减较少、对高频分量衰减较多的反射衰减$r(t)$，其傅里叶变换为：

$$R(j\omega) = A(j\omega)e^{-jm\omega} \tag{1}$$

从而有：

$$y(t) = x(t) + \sum_{n=1}^{\infty} x(t) * r(t)^n \tag{2}$$

得到声学空间的系统函数为：

$$H_{ideal}(j\omega) = 1 + \sum_{n=1}^{\infty} R(j\omega)^n \tag{3}$$

查阅资料可知，我们构建的模型正是梳状滤波器混响模型的简化版本[1]。其优化模型可以见 1962 年提出的 Schroeder 混合模型[2]，它开启了人工混响研究的大门。更加先进的方法可见 1991 年的反馈延迟网络（FDN）的数字混响方法[3]和 1995 年 Reilly 提出的用于人工混响的脉冲响应卷积技术[4]，这与我们的卷积混响存在一定关联。

因此，我们录制的单位冲激响应$h(t)$，其频谱就是对$H_{ideal}(j\omega)$的近似模拟。受实际录音设备条件限制，我们只能录入$R(j\omega)$的前若干项（从工程应用角度出发，这通常已经足够了），同时会存在一定噪声。在利用卷积混响赋予声音不同的材质感中，噪声往往会产生较大的干扰而无法产生预期效果，因此我们需要针对单位冲激响设计相应滤波器进行优化，优化后的冲激响应作为软件使用的IR文件。

3 程序设计

3.1 编程思路

整体设计思路如下：利用线性时不变系统特性，通过在不同场景下录制单位冲激响应，与输入信号进行卷积运算，并借助滤波器对输出效果进行调整预处理。处理后的单位冲激响应作为软件设计中应用的IR文件。

关于单位冲激响应的预处理部分，我们使用$audioread()$函数读取音频原声信号和录制的

单位冲激响应信号，利用 MATLAB 自带的卷积函数*conv*()在时域和频域上使用*plot*()函数绘制两个信号及卷积后信号的时域和频域波形。通过单位冲激响应信号频谱特征及混响音频效果，我们根据实际情况选用低通滤波器以削弱录制时可能存在的噪声，使混响后的音频更加契合实际环境效果。

关于软件设计部分，我们使用 MATLAB App Designer 进行编程，以面向对象的编程思路制作了一个具有混响功能的 app。

根据功能需要，我们创建了 app.path0，app.sound，app.type，app.texture，app.reverb，app.stop，app.A1，app.A2，app.A3，app.B1，app.B2，app.B3 共 12 个组件，并通过对应的回调函数实现功能。

app.path0 是编辑文本类组件，用户可在其中输入音源文件存储路径，而这可视作一条 字符串作为其 Value。

app.sound是按钮类组件，用户单击按钮后，运行其回调函数，在回调函数中，会先通过audioread函数读取app.path0.Value路径下的音源，再利用sound函数播放音源，利用plot函数在A1 B1处绘制声音的时域与频域图像。

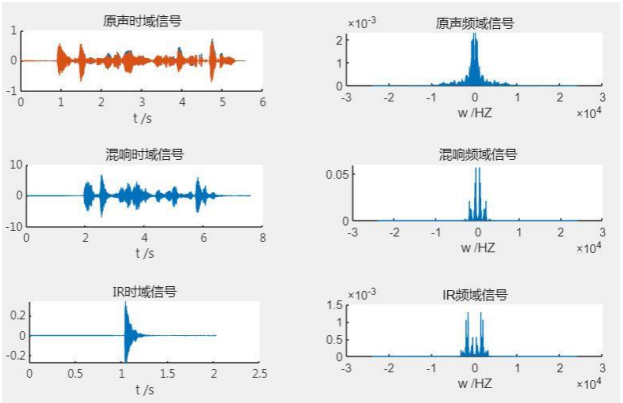
app.type是列表框类组件，我们预设了原声，寝室，楼道，C117 四种混响效果，用户选择某种效果后，该组件的Value将变成对应IR文件的路径，由于原声不存在IR文件，其对应的 Value 是一个空字符串。

app.texture是列表框类组件，我们预设了无，金属，玻璃，木头四种声音质感，用户选择某种质感后，该组件的Value将变成对应 IR 文件的路径，由于“无”不存在IR文件，其对应的Value是一个空字符串。

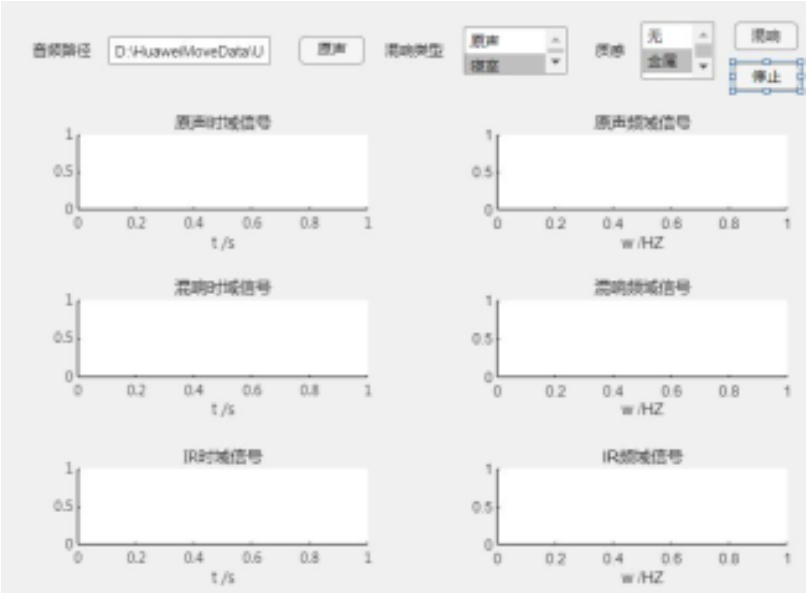
app.reverb是按钮类组件，用户单击按钮后，运行回调函数，在回调函数中，会先通过audioread函数读取app.path0.Value路径下的音源至x与fs0，其中fs0为采样频率，读取app.type.Value路径下的IR文件至h1与fs，读取app.texture.Value路径下的IR文件至h2，先将h1与h2卷积赋给 h，然后考虑到由于fs0与fs有可能不相等，我们要对x与h中采样率更高的数据再采样，使二者采样率一致，然后再将x与h卷积赋予y，y即为混响后的声音，再利用sound函数播放音源，利用plot函数在A2 B2处绘制单位冲激响应的时域与频域图像，在A3 B3处绘制混响声音的时域与频域图像。

app.stop是按钮类组件，用户单击按钮后，运行回调函数，利用clear sound函数中止发声。

app.A1，app.A2，app.A3，app.B1，app.B2，app.B3是坐标区类组件，可显示信号的时域与频域图像。



以上通过使用回调函数实现十二个组件的相互联系与各自功能，我们成功制作了一个允许用户输入各种音源，均可输出混响后声音的软件。



3.2 主要流程图及说明

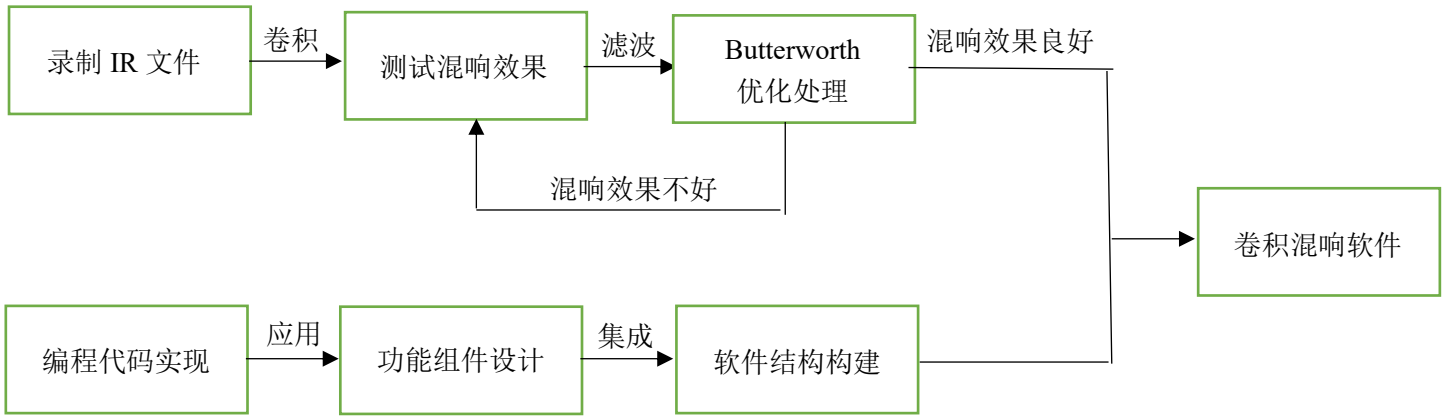


图 2. 卷积混响软件系统流程图

在录制 IR 文件中，我们采用两种方式，分别用拍手声音近似单位冲激信号和 MATLAB 设计输出单位冲激信号的代码 impulse.m，在不同现实环境下录制单位冲激响应。

在测试混响效果中，我们采取评价标准如下：首先判断卷积混响后声音是否有底噪、底噪是否严重；是否有多次回声、回声是否严重；是否存在啸叫现象、啸叫是否严重。声音存在底噪的主要原因在于向实际环境输入信号并非良好的单位冲激信号，同时环境存在一定底噪；声音存在干扰回声的主要原因在于录入的 IR 文件包含了除单位冲激响应之外的杂声，这些杂声的最大幅值与单位冲激响应的最大幅值近似在 1 个数量级之内，使得杂声与原声卷积后的噪声较为明显；声音存在啸叫的主要原因是录入 IR 文件存在正反馈调节，同时 IR 文件幅频可能较大。其次，我们通过 Bricasti.IR.Library 和 York University 声学实验室发布提供的 IR 文件资源，将录制 IR 文件与二者在频域视角上进行对比，确认噪声分布范围，便于 Butterworth 滤波处理。

在软件结构构建上，根据功能需要，我们创建了app.path0，app.sound，app.type，app.texture，app.reverb，app.stop，app.A1，app.A2，app.A3，app.B1，app.B2，app.B3共12个组件，并通过对应的回调函数实现功能。通过使用回调函数实现十二个组件的相互联系与各自功能，我们成功制作了一个允许用户输入各种音源，均可输出混响后声音的软件。

3.3 结果分析

- **滤波优化**

我们选用 5 阶低通 butterworth 滤波器（通带截止频率 Wp=8000Hz，阻带截止频率 Ws=1.1*8000=8800Hz），其频响特性如下：

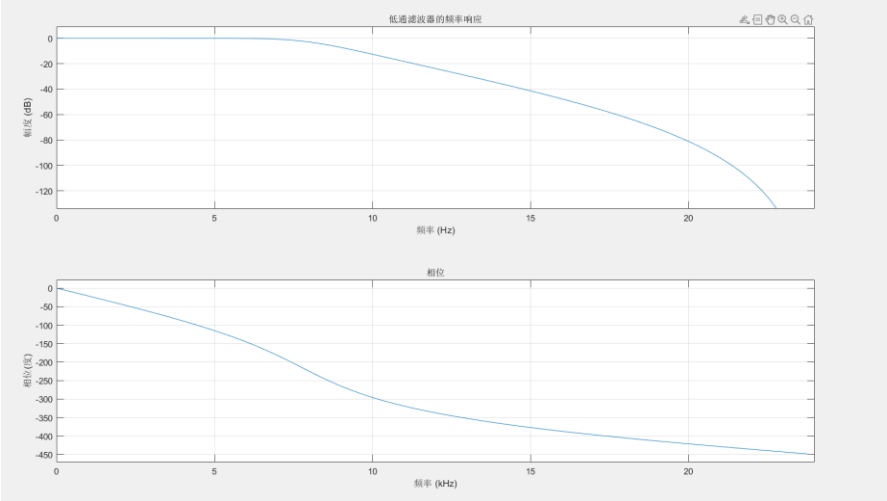


图 3.　低通滤波器频率响应特性曲线

以楼道的单位冲激响应为例，绘制其原时域波形和频谱特性，及滤波后的时域波形和频谱特性曲线如下：

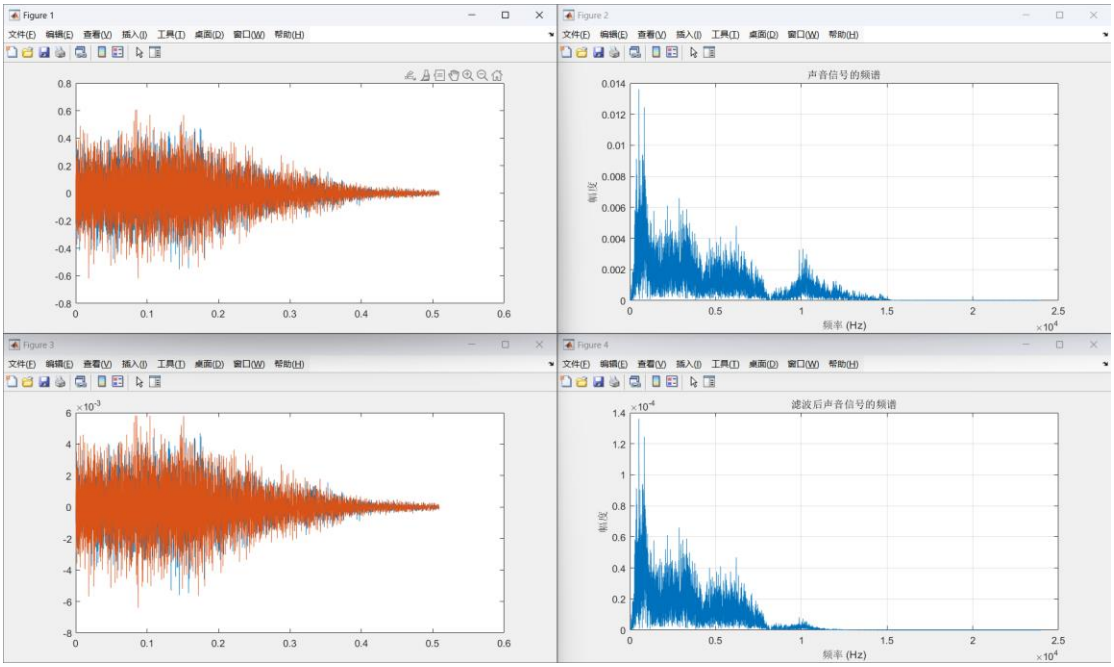


图 4.　楼道 IR 文件滤波前后时域与频域波形图

在滤波优化处理的过程中，我们发现：引起混响效果严重失真的主要因素在于 IR 文件时域波形的幅值和频谱的高频分量。当时域波形幅值过大时，使得底噪对于人耳更为明显，听起来会有很大的啸叫和回声。当频谱高频分量过多时，会引起较严重的底噪。基于此，对于 C117 的 IR 文件及寝室 IR 文件，由于其高频分量很少，我们仅做了幅值上的调整，在此未作展示。

- **运行结果**

在 matlab 上运行后播放原声音，同时绘制原声音时域与频域波形：

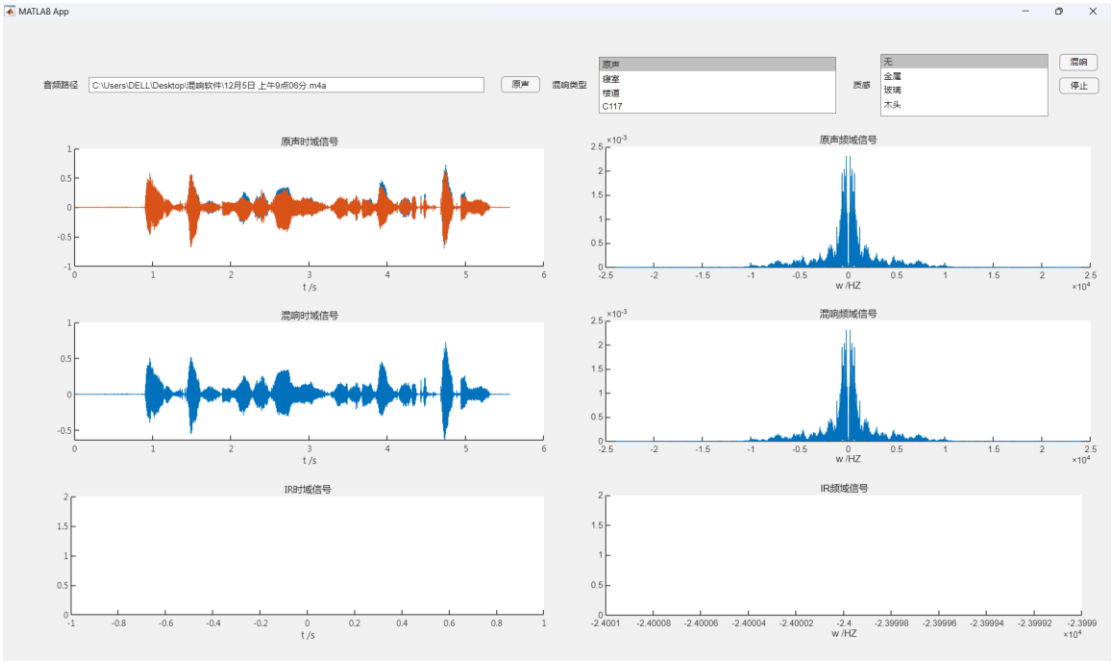


图 5.　原声时域与频域波形图

将原声音分别与寝室、楼道、C117 分别进行卷积，得到时域和频域波形：

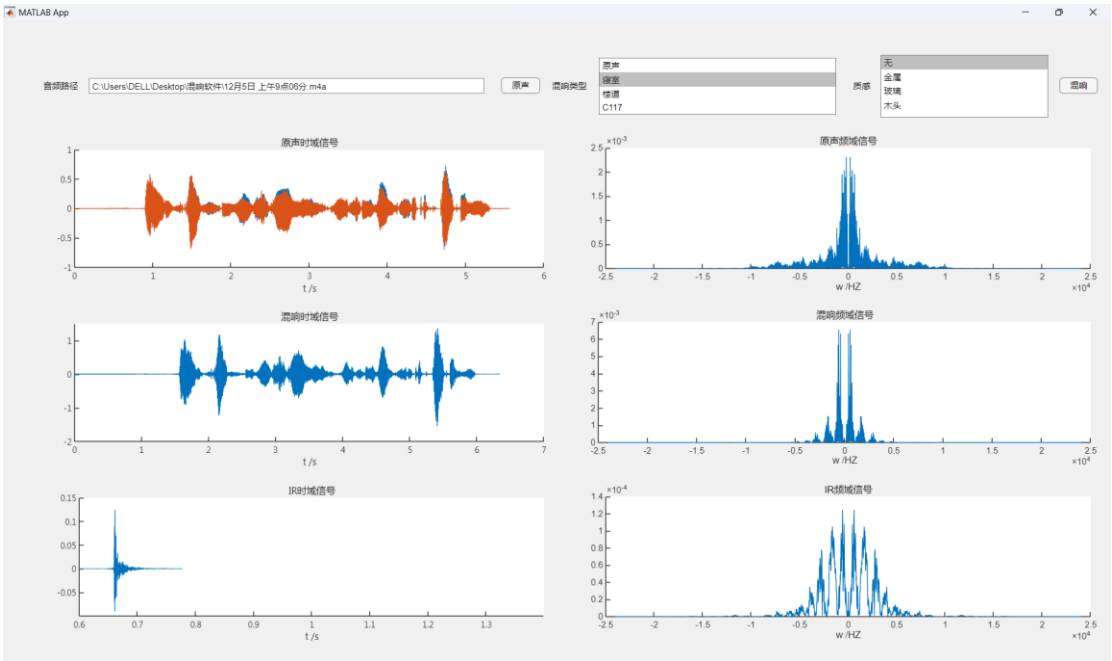


图 6.　寝室混响音频时域与频域波形图

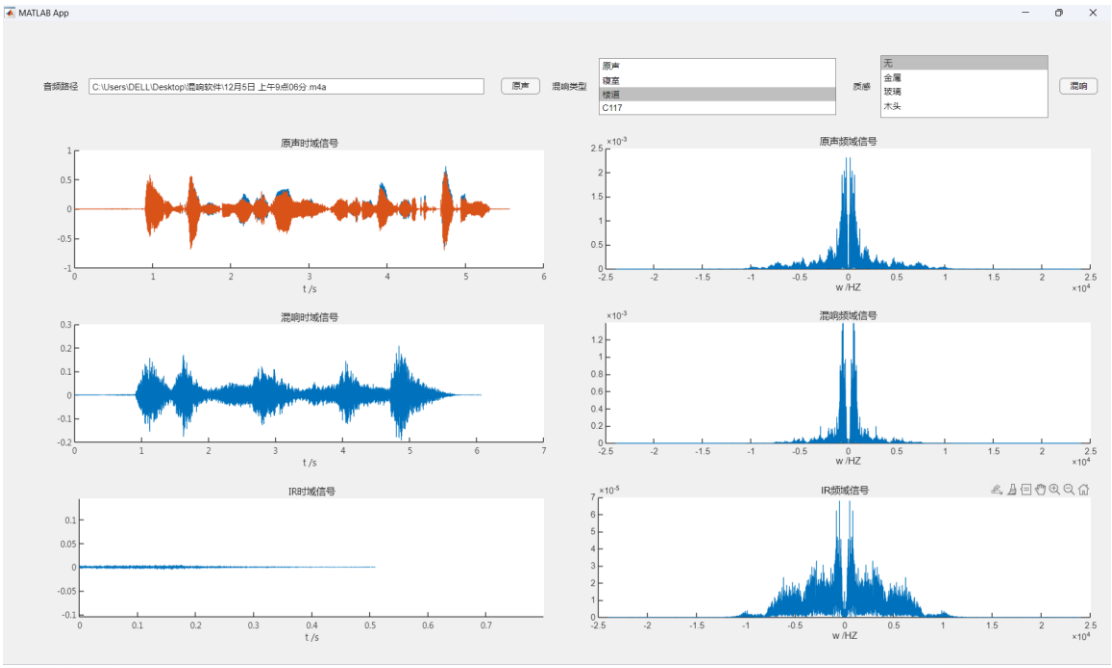


图 7.　楼道混响音频时域与频域波形图

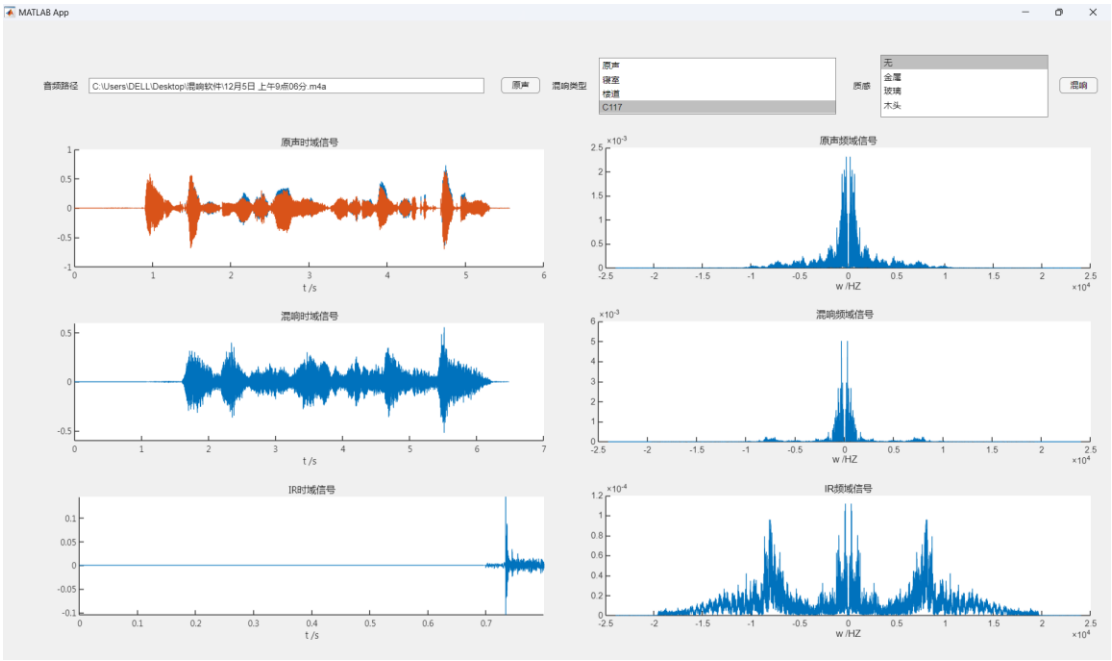


图 8.　C117 混响音频时域与频域波形图

将原声音分别金属、玻璃、木器 IR 文件分别进行卷积，得到时域和频域波形：

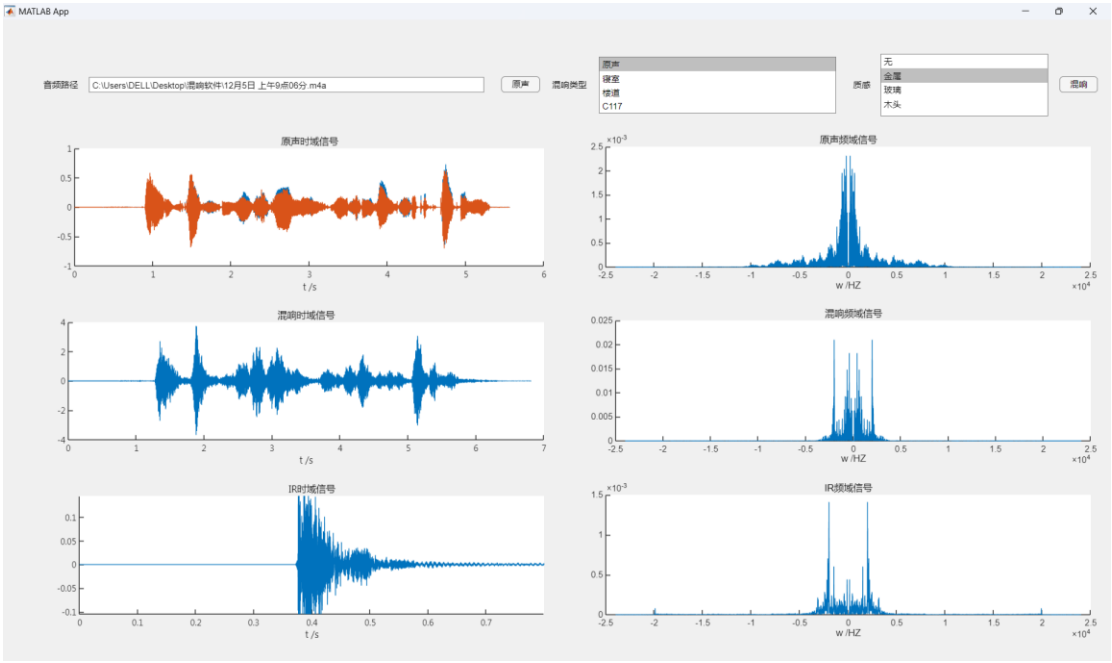


图 9.金属质感音频时域与频域波形图

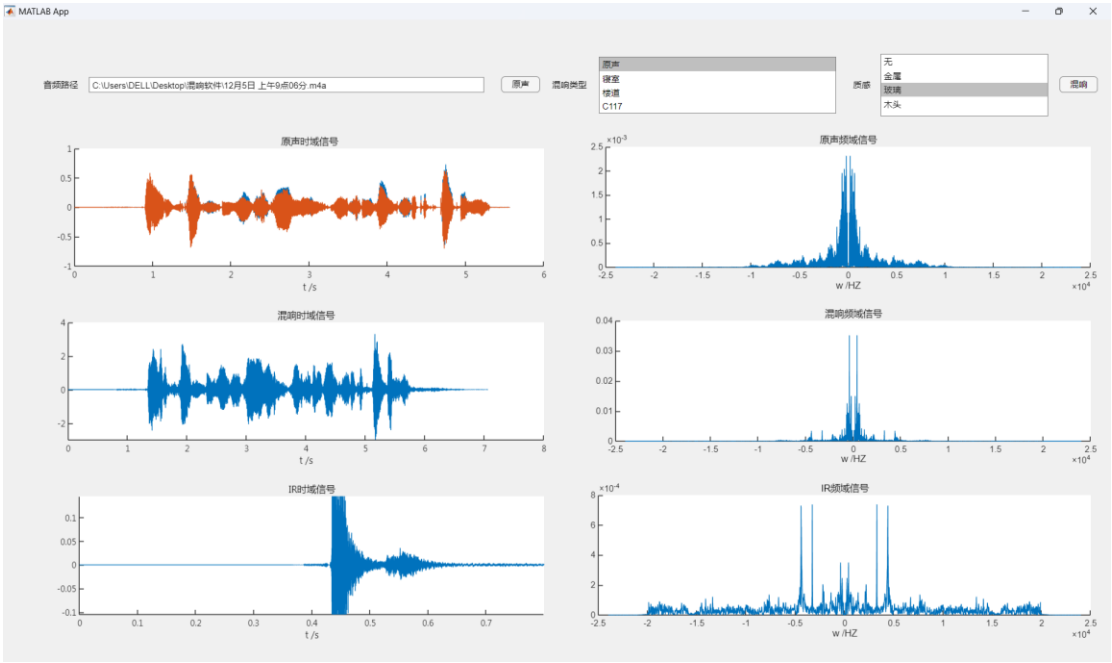


图 10.玻璃质感音频时域与频域波形图

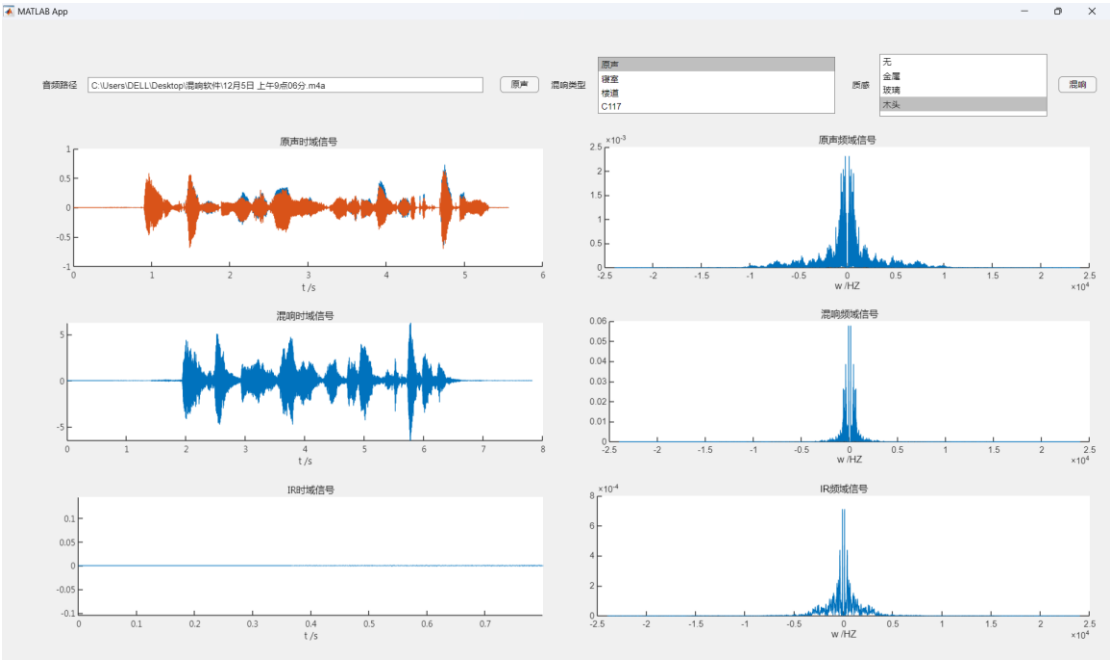


图 11.木头质感音频时域与频域波形图

- **结果验证**

我们将录制的 IR 文件与 Bricasti.IR.Library 和 York University 的 IR 文件资源进行对比，得到时域与频域波形如下：

1. P084 - Small Tiled.wav（理想类寝室，Bricasti.M7.IR.Library）与 IR 寝室 2.wav 相对比

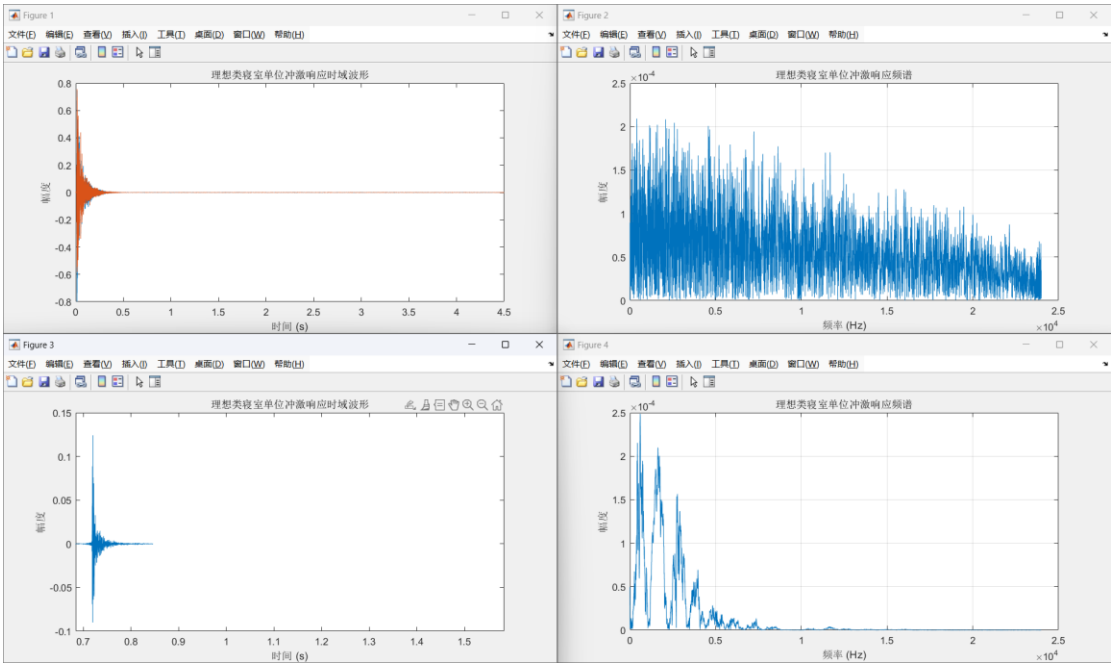


图 12.寝室与类理想寝室音频时域与频域波形图

二者在频谱上衰减规律基本相同，而衰减速率略有不同，主要原因在于声

学空间的空间大小和反射强度可能存在一定差异。

2. stairwell_ortf.wav（理想类楼道，York University）与 stair.wav 相对比

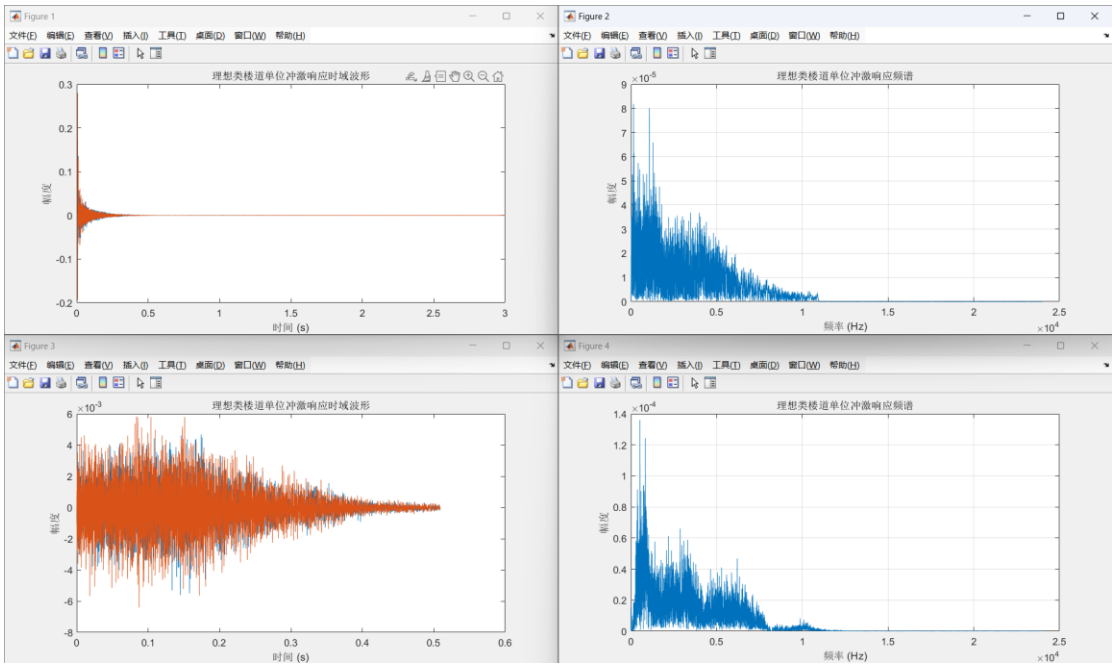


图 13.楼道与类理想楼道音频时域与频域波形图

二者在频谱上较为接近，均在频率约 2000Hz 左右时高频分量得到了骤减，然后频谱随频率增加逐渐衰减至 0，约 12000Hz 附近无明显波形。

3. P056 - Medium Chamber.mp3（理想类 C117，Bricasti.M7.IR.Library）与 C117.mp3 相对比

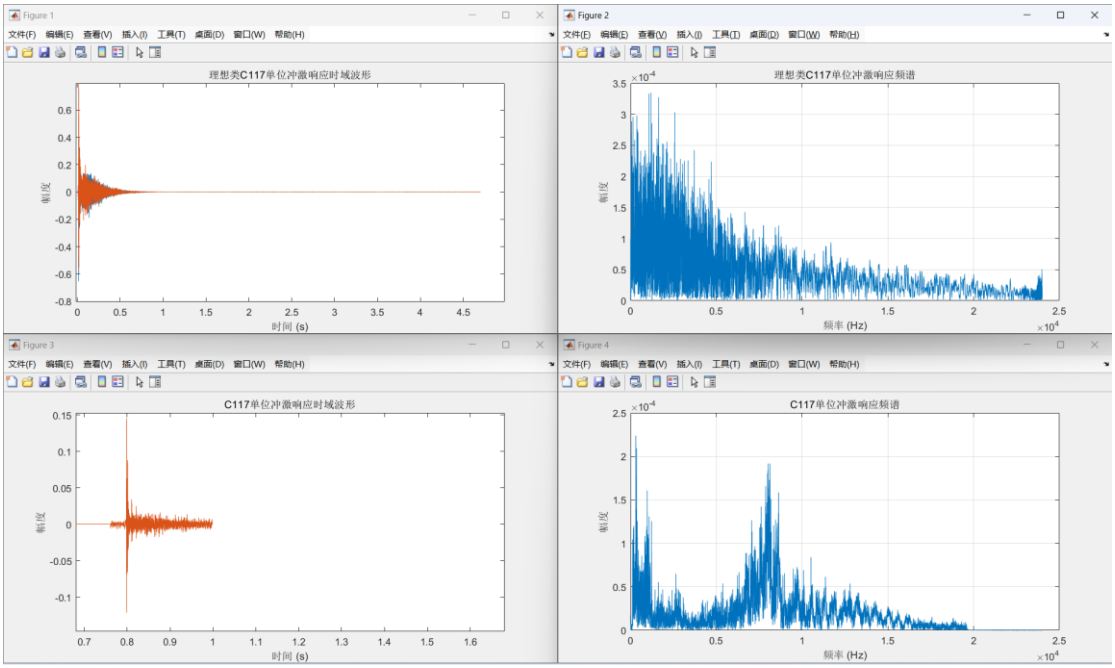


图 14.C117 与理想类 C117 音频时域与频域波形图

二者在频域上均存在多个峰值，其中我们采集的 C117 的 IR 文件两主峰值频率相差约为 8000Hz，而理想类 C117 相差值较小。由于采样地点不同，频谱上存在一定差异是合理的。

4 组内互评

5 总结与体会

参考文献

[1] Schroeder M R. Logan B F. Colorless artificial reverberation[J]. Journal of the Acoustical Society ofAmerica,1961,AU-9(6):209-214.
[2] Schroeder M R. Natural-sounding artificial reverberation[J].Journal of the Audio Engineering Society,1962,10(3):219-233.
[3] Jot J M, Chaigne A. Method and system for artificial spatialization of digital audio signals[J].Journal ofthe Acoustical Society ofAmerica, 1996, 100(5):2901.
[4]Reilly A,McGrath D.Convolution processing for realistic reverberation[C].Convention of Audio Engineering Society, Paris, France, Feb.1995:3977.

附录 MATLAB 程序主要代码

impulse.m

```
clear;clc;
% 设置采样频率和采样点数
fs = 44100; % 采样频率
duration = 1; % 信号持续时间
t = 0:1/fs:duration-1/fs;

% 生成单位冲激信号
impulse_signal = zeros(size(t));
impulse_signal(1) = 1;

sound(impulse_signal, fs);

impulse_duration = 0.05;
pulse_width = round(impulse_duration * fs); % 计算脉冲宽度的样本数
impulse_signal_visible = [ones(pulse_width, 1); zeros(length(t)-
pulse_width, 1)];
impulse_signal_visible = impulse_signal_visible(1:length(t)); % 截取到与 t 相
同的长度

figure;
plot(t, impulse_signal_visible);
xlabel('时间 (秒)');
ylabel('幅值');
title('单位冲激信号波形');
grid on;
```

Stair_clap1.m

```
clear;clc;
[x,fs0]=audioread('12 月 5 日 上午 9 点 06 分.mp3');
[h1,fs]=audioread('clap1_裁剪.wav');

%时域波形
t=(0:length(h1)-1)/fs;
figure;
plot(t,h1);
title('原 IR 文件的时域波形');
xlabel('时间 (s)');
ylabel('幅度');

%频谱
Fs = 48000;
t0 = 0:1/Fs:1-1/Fs;
```

```
Y = fft(h1);% 计算信号的傅里叶变换
P2 = abs(Y/length(h1));% 计算双边频谱并转换为单边频谱
P1 = P2(1:length(h1)/2+1);
P1(2:end-1) = 2*P1(2:end-1);
f = Fs*(0:(length(h1)/2))/length(h1);
figure;
plot(f, P1);
title('原 IR 文件的频谱');
xlabel('频率 (Hz)');
ylabel('幅度');
grid on;

% 滤波器 时域/频域
Wp = 8000; % 通带截止频率
Ws = 1.1 * Wp; % 阻带截止频率
N = 5; % 滤波器阶数（阶数越高，过渡带越窄，但相位失真也可能增加）
[b, a] = butter(N, Wp / (Fs / 2), 'low');
y = filter(b, a, h1).*0.01;

figure;
plot(t,y);
title('滤波后 IR 文件的时域波形');
xlabel('时间 (s)');
ylabel('幅度');

Y_filter=fft(y);
P2_filter = abs(Y_filter/length(h1));
P1_filter = P2_filter(1:length(h1)/2+1);
P1_filter(2:end-1) = 2*P1_filter(2:end-1);
f_filter = Fs*(0:(length(h1)/2))/length(h1);
figure;
plot(f_filter, P1_filter);
title('滤波后 IR 文件的频谱');
xlabel('频率 (Hz)');
ylabel('幅度');
grid on;

% 查看滤波器的频率响应
figure;
freqz(b, a, 1024, Fs);
title('低通滤波器的频率响应');
xlabel('频率 (Hz)');
ylabel('幅度 (dB)');
grid on;
```

```
y1=conv(x(:,1),h1(:,1));
y2=conv(x(:,2),h1(:,2));
y_fin=[y1 y2];
sound(y_fin,fs0);
```

app1.mlapp

```
classdef app1 < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure   matlab.ui.Figure
        Button_3   matlab.ui.control.Button
        ListBox_2  matlab.ui.control.ListBox
        Label_3    matlab.ui.control.Label
        Button_2   matlab.ui.control.Button
        ListBox    matlab.ui.control.ListBox
        Label_2    matlab.ui.control.Label
        Button     matlab.ui.control.Button
        path0      matlab.ui.control.EditField
        Label      matlab.ui.control.Label
        A3         matlab.ui.control.UIAxes
        B3         matlab.ui.control.UIAxes
        B2         matlab.ui.control.UIAxes
        A2         matlab.ui.control.UIAxes
        B1         matlab.ui.control.UIAxes
        A1         matlab.ui.control.UIAxes
    end

    % Callbacks that handle component events
    methods (Access = private)

        % Button pushed function: Button
        function ButtonPushed(app, event)
            clear sound
            p0=app.path0.Value;
            [x, fs0] = audioread(p0);
            sound(x,fs0);
            t = (0:length(x)-1) / fs0;  % 时间向量
            plot(app.A1,t,x);
            if size(x, 2) > 1
            signal = x(:, 1);  % 取第一个通道
            end
            N = length(signal);  % 信号长度
```

```
    signal_fft = fft(signal);  % 傅里叶变换
    f = (-N/2:N/2-1)*(fs0/N);  % 使用负频率范围
    magnitude = abs(fftshift(signal_fft)) / N;  % 归一化幅度并居中
    plot(app.B1,f, magnitude);  % 只绘制频率范围为正的部分
end

% Button down function: A1
function A1ButtonDown(app, event)

end

% Button pushed function: Button_2
function Button_2Pushed(app, event)
   clear sound
    p0=app.path0.Value;
    pir=app.ListBox.Value;
    pir2=app.ListBox_2.Value;
    [x, fs0] = audioread(p0);
    if pir
    [h1,fs]=audioread(pir);
    else
        h1=1;
        fs=fs0;
    end
    if pir2
    [h2,fs2]=audioread(pir2);
    else
        h2=1;
        fs2=fs0;
    end
    h=conv(h1(:,1),h2(:,1));
     t0 = (0:length(h)-1) / fs0;% 时间向量
     plot(app.A3,t0,h);
      N0 = length(h);  % 信号长度
    h_fft = fft(h);  % 傅里叶变换
    f = (-N0/2:N0/2-1)*(fs0/N0);  % 使用负频率范围
    magnitude0 = abs(fftshift(h_fft)) / N0;  % 归一化幅度并居中
    plot(app.B3,f, magnitude0);

    h=h(1:round(fs/fs0):end,:);
    if size(x, 2) > 1
    signal = x(:, 1);  % 取第一个通道
    end
    y=conv(signal,h(:,1));
```

```
            sound(y,fs0);
            t = (0:length(y)-1) / fs0;  % 时间向量
            plot(app.A2,t,y);

            N = length(y);  % 信号长度
            y_fft = fft(y);  % 傅里叶变换
            f = (-N/2:N/2-1)*(fs0/N);  % 使用负频率范围
            magnitude = abs(fftshift(y_fft)) / N;  % 归一化幅度并居中
            plot(app.B2,f, magnitude);

        end

        % Button pushed function: Button_3
        function Button_3Pushed(app, event)
            clear sound
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 705 540];
            app.UIFigure.Name = 'MATLAB App';

            % Create A1
            app.A1 = uiaxes(app.UIFigure);
            title(app.A1, '原声时域信号')
            xlabel(app.A1, 't /s')
            ylabel(app.A1, ' ')
            zlabel(app.A1, 'Z')
            app.A1.FontName = 'Microsoft Tai Le';
            app.A1.ButtonDownFcn = createCallbackFcn(app, @A1ButtonDown,
true);
            app.A1.Position = [56 307 285 117];

            % Create B1
            app.B1 = uiaxes(app.UIFigure);
            title(app.B1, '原声频域信号')
            xlabel(app.B1, 'w /HZ')
```

```
zlabel(app.B1, 'Z')
app.B1.Position = [400 307 286 117];

% Create A2
app.A2 = uiaxes(app.UIFigure);
title(app.A2, '混响时域信号')
xlabel(app.A2, 't /s')
ylabel(app.A2, ' ')
zlabel(app.A2, 'Z')
app.A2.FontName = 'Microsoft Tai Le';
app.A2.Position = [56 170 285 117];

% Create B2
app.B2 = uiaxes(app.UIFigure);
title(app.B2, '混响频域信号')
xlabel(app.B2, 'w /HZ')
zlabel(app.B2, 'Z')
app.B2.Position = [400 170 286 117];

% Create B3
app.B3 = uiaxes(app.UIFigure);
title(app.B3, 'IR 频域信号')
xlabel(app.B3, 'w /HZ')
zlabel(app.B3, 'Z')
app.B3.Position = [400 32 286 117];

% Create A3
app.A3 = uiaxes(app.UIFigure);
title(app.A3, 'IR 时域信号')
xlabel(app.A3, 't /s')
ylabel(app.A3, ' ')
zlabel(app.A3, 'Z')
app.A3.FontName = 'Microsoft Tai Le';
app.A3.Position = [56 32 285 117];

% Create Label
app.Label = uilabel(app.UIFigure);
app.Label.HorizontalAlignment = 'right';
app.Label.Position = [52 462 53 22];
app.Label.Text = '音频路径';

% Create path0
app.path0 = uieditfield(app.UIFigure, 'text');
app.path0.Position = [120 462 133 22];
```

```
            app.path0.Value = 'D:\HuaweiMoveData\Users\86135\Desktop\信号大作
业\12 月 5 日 上午 9 点 06 分.m4a';

            % Create Button
            app.Button = uibutton(app.UIFigure, 'push');
            app.Button.ButtonPushedFcn = createCallbackFcn(app,
@ButtonPushed, true);
            app.Button.Position = [277 462 53 23];
            app.Button.Text = '原声';

            % Create Label_2
            app.Label_2 = uilabel(app.UIFigure);
            app.Label_2.HorizontalAlignment = 'right';
            app.Label_2.Position = [342 462 53 22];
            app.Label_2.Text = '混响类型';

            % Create ListBox
            app.ListBox = uilistbox(app.UIFigure);
            app.ListBox.Items = {'原声', '寝室', '楼道', 'C117'};
            app.ListBox.ItemsData = {'', 'IR 寝室 2.wav', 'stair.wav',
'C117.mp3'};
            app.ListBox.Position = [412 454 85 38];
            app.ListBox.Value = 'IR 寝室 2.wav';

            % Create Button_2
            app.Button_2 = uibutton(app.UIFigure, 'push');
            app.Button_2.ButtonPushedFcn = createCallbackFcn(app,
@Button_2Pushed, true);
            app.Button_2.Position = [634 473 53 23];
            app.Button_2.Text = '混响';

            % Create Label_3
            app.Label_3 = uilabel(app.UIFigure);
            app.Label_3.HorizontalAlignment = 'right';
            app.Label_3.Position = [515 462 29 22];
            app.Label_3.Text = '质感';

            % Create ListBox_2
            app.ListBox_2 = uilistbox(app.UIFigure);
            app.ListBox_2.Items = {'无', '金属', '玻璃', '木头'};
            app.ListBox_2.ItemsData = {'', '金属 0.m4a', '玻璃.m4a', '木
头.m4a'};
            app.ListBox_2.Position = [558 450 60 46];
            app.ListBox_2.Value = '金属 0.m4a';
```

```
            % Create Button_3
            app.Button_3 = uibutton(app.UIFigure, 'push');
            app.Button_3.ButtonPushedFcn = createCallbackFcn(app,
@Button_3Pushed, true);
            app.Button_3.Position = [634 441 54 22];
            app.Button_3.Text = '停止';

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = app1

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```