**1 实验目的和要求**

* 1. **实验目的**

简要学习MATLAB的信号频谱分析方法。

**1.2 实验要求**

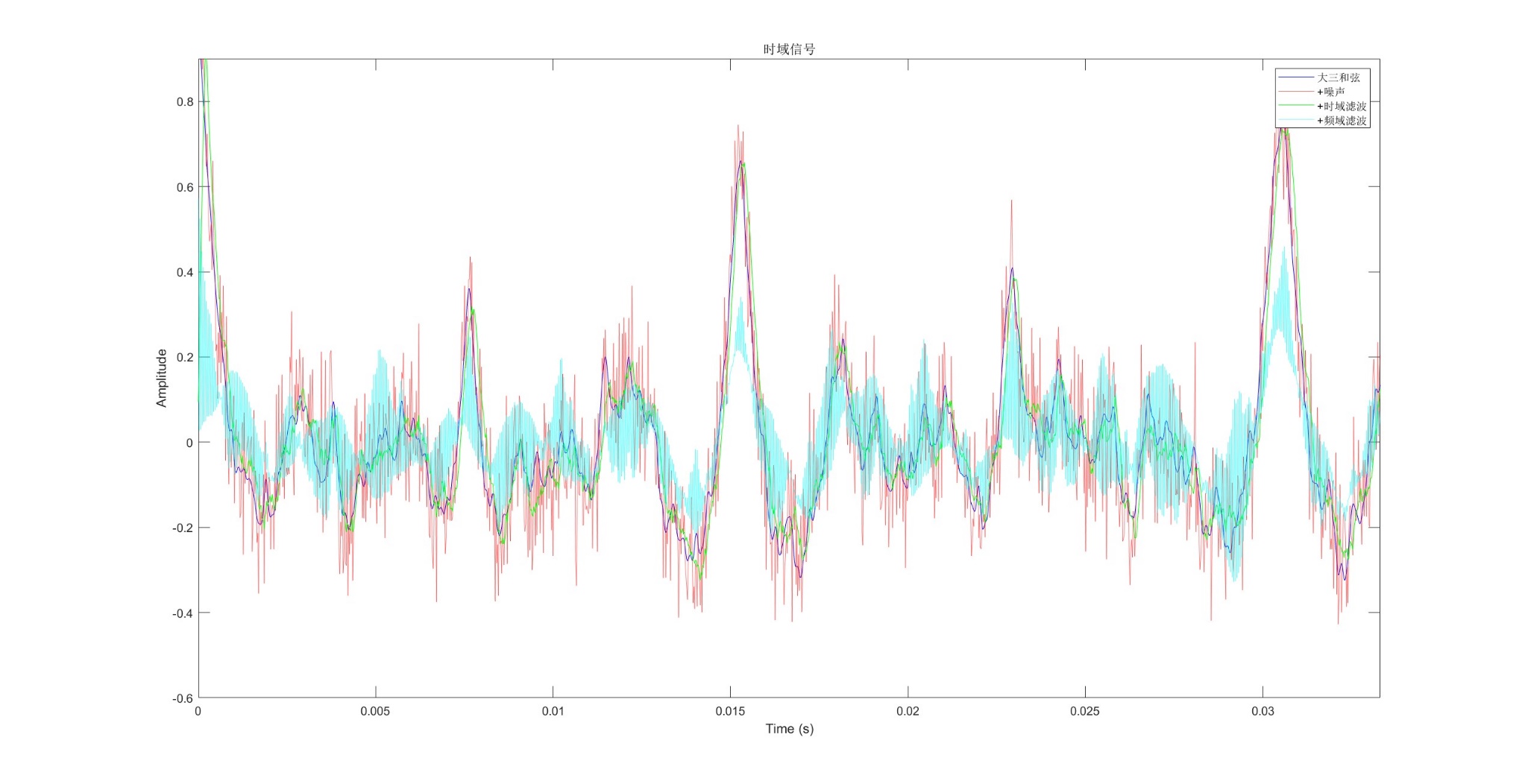
1. 参考文档《基于MATLAB 的信号频谱分析》，构建一个信号，并利用 MATLAB分析其频谱。
2. 对频谱作适当的处理（滤波，或增加一些频率成分），观察信号的变化。

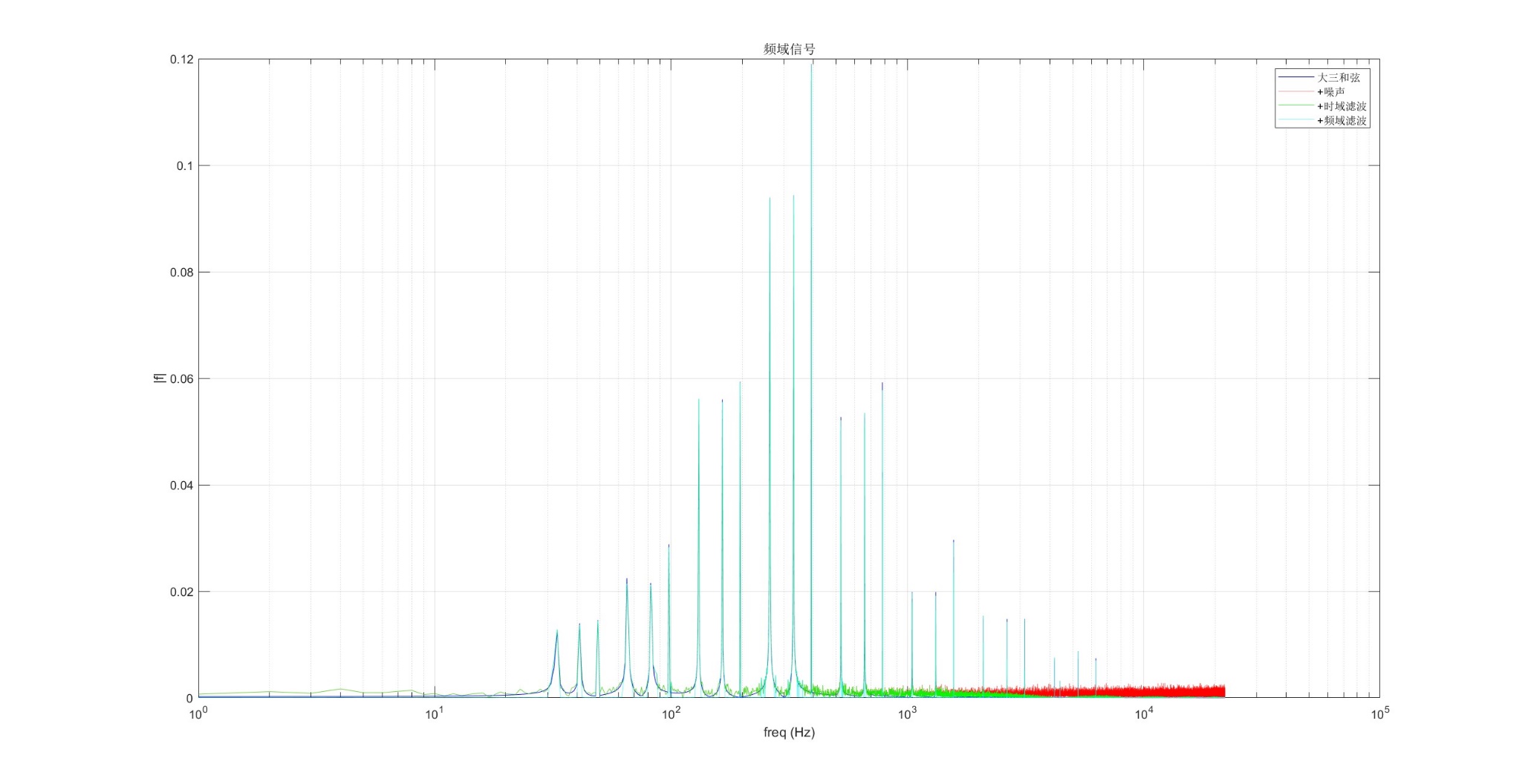
**2 实验原理**

1. 本实验构建的信号为：声音信号，C调大三和弦（由do mi so组成），采样率44.1kHz。
2. 在信号中加入白噪音，并尝试通过时域上均值滤波、频域去除底噪的两种方式降噪。
3. 均值滤波的窗口大小为：10，采用filter函数；
4. 频域去除底噪方法为，将频域中所有小于某一阈值的信号缩小到0.05倍。

**3 实验内容**

1. 构建原信号，以国际标准A4=440Hz定音，计算do mi so的音高。
2. 在原信号内加入高八度与低八度成分，使之更加饱满。具体地，每增加或降低一个八度，频率乘以或除以2，振幅减小一倍。最终原信号最大振幅约为0.8。
3. 为信号添加白噪声，噪声最大振幅为0.1。
4. 进行时域滤波与频域滤波，并计算FFT与IFFT。
5. 绘制四种信号时域与频域的图像：原信号、+噪声、+时域滤波、+频域滤波。
6. 通过sound函数播放四种信号。

**4 实验结果与分析**



**5 实验结论**

1. 时域均值滤波并不能很好地消除随机添加的白噪声；
2. 频域滤波可以去除高频的噪声，起到降噪的作用。

**6 源代码与分析**

（粘贴本次实验使用的源代码，并使用注释的方式进行适当的分析）

% 信电导

% Coursework #1

% Author: mch

% An simple program utilizing fft() & dfft() function.

Fs = 44100; % 采样率 44.1kHz

Ts = 1/Fs; % 周期

Len = 1; % 总时长 1s

N = Len / Ts;

Ns = round(N / 30);

t = (0:N-1) \* Ts;

% 在音乐里，do(1) mi(3) so(5) 构成一个和谐的"大三和弦"。

% do: C4 261.63Hz

% mi: E4 329.63Hz

% so: G4 392.00Hz

s = zeros(1, N);

% 为信号添加基波

s = sub(s, t, 1, 1);

% 为信号添加高八度与低八度的谐波

for i = 1:4 % C,E,G 5 - 8 高八度

s = sub(s, t, 2^i, 2^(-i)); % 频率 2^i 倍，同时振幅减小 2^-i

end

for i = 1:3 % C,E,G 1 - 3 低八度

s = sub(s, t, 0.5^i, 2^(-i));

end

s = s / max(s);

function r = sub(s, t, m\_freq, m\_amp)

r = s ...

+ (cos(2\*pi\*t\*m\_freq\* 261.63) ... % C do

+ cos(2\*pi\*t\*m\_freq\* 329.63) ... % E mi

+ cos(2\*pi\*t\*m\_freq\* 392.00) ... % G so

) / 3 \* m\_amp;

end

% 为信号添加白噪声

s2 = s + 0.1 \* randn(size(t));

% 为添加噪声后的信号在时域内滤波

windowSize = 10; % 卷积窗口大小为 10

b = (1/windowSize)\*ones(1,windowSize);

a = 1; % https://ww2.mathworks.cn/help/matlab/ref/filter.html

s3 = filter(b, a, s2);

% 计算频域

f\_x = Fs \* (0:(N / 2)) / N;

f = abs(fft(s) / N); % 原信号

f = f(1:N/2+1);

f(2:end - 1) = 2 \* f(2:end - 1);

f2 = fft(s2) / N; % +噪音

f2 = f2(1:N/2+1);

f2(2:end - 1) = 2 \* f2(2:end - 1);

f3 = abs(fft(s3) / N); % 时域滤波

f3 = f3(1:N/2+1);

f3(2:end - 1) = 2 \* f3(2:end - 1);

% 在噪声信号的频域内去底噪

f4 = f2;

eps = 0.003;

for i = 1:length(f4)

if abs(f4(i)) < eps

f4(i) = f4(i) \* 0.05;

end

end

% 从 f4 逆变换得到降噪信号

f4i = f4;

f4i(2:end - 1) = 0.5 \* f4i(2:end - 1);

f4j = flipud(conj(f4i(2:end)));

f4i = N \* [f4i, f4j];

s4 = real(ifft(f4i));

f2 = abs(f2);

f4 = abs(f4);

% https://ww2.mathworks.cn/matlabcentral/fileexchange/89182-get-single-sided-spectrum-from-time-signal-and-viceversa

% 绘制时域

k = figure('units','normalized','outerposition',[0 0 1 1]);

plot(t(1:Ns), s(1:Ns), 'Color', [0, 0, 1]);

hold on

plot(t(1:Ns), s2(1:Ns), 'Color', [1, 0, 0, 0.5]);

plot(t(1:Ns), s3(1:Ns), 'Color', [0, 1, 0, 1]);

plot(t(1:Ns), s4(1:Ns), 'Color', [0, 1, 1, 0.5]);

legend('大三和弦', '+噪声', '+时域滤波', '+频域滤波');

axis([0, t(Ns), -0.6, 0.9]);

title('时域信号');

xlabel('Time (s)');

ylabel('Amplitude');

% 绘制频域

k = figure('units','normalized','outerposition',[0 0 1 1]);

plot(f\_x, f, 'Color', [0, 0, 1]);

hold on

plot(f\_x, f2, 'Color', [1, 0, 0, 0.4]);

plot(f\_x, f3, 'Color', [0, 1, 0, 0.6]);

plot(f\_x, f4, 'Color', [0, 1, 1, 0.5]);

legend('大三和弦', '+噪声', '+时域滤波', '+频域滤波');

title('频域信号')

xlabel('freq (Hz)')

ylabel('|f|')

grid on

set(gca,'Xscale','log') % 设置 x 轴对数标度

% 播放大三和弦

sound(s, Fs);

pause(Len + 0.5);

% 播放干扰后的大三和弦

sound(s2, Fs);

pause(Len + 0.5);

% 播放时域滤波后的大三和弦

sound(s3, Fs);

pause(Len + 0.5);

% 播放频域滤波后的大三和弦

sound(s4, Fs);