语音合成

Matlab 大作业报告

陈子熠

2024 年 7 月 21 日

目录

1 语音预测模型

1.1

给定

$$e(n) = s(n) - a_1 s(n-1) - a_2 s(n-2)$$

假设 $e(n)$ 是输入信号，$s(n)$ 是输出信号，等式两边同时取 Z 变换，得上述滤波器的传递函数：

$$H(z) = \frac{S(z)}{E(z)} = \frac{1}{1 - a_1 z^{-1} - a_2 z^{-2}}$$

故 $b = 1$，$a = [1, -a_1, -a_2]$。

对于任意系统函数，每一对共轭极点对应一个衰减的正弦信号的特征响应。令该对共轭极点为 $|p\_i|e^{\pm j\Omega}$，幅角为 Ω，则对应的共振峰频率为 $\Omega/2\pi f$，其中 f 为采样频率。

据此，可以计算出共振峰频率：

```
function formants = sys_formant_cal(a, T)
    % Calculate the formant frequencies of a given system
    % a [array]: denominator coefficients of the system
    % T [float]: sampling period
    % return [array]: formant frequencies
    poles = roots(a);
    poles = poles(imag(poles) > 0);
    formants = sort(atan2(imag(poles), real(poles)) / (2 * pi * T));
end
```

令 $a = [1, -a_1, -a_2]$，$T = 1/8000$，得到共振峰频率约为 $1000Hz$。

用 zplane 绘制零极点图：

```
zplane(b, a);
```

零极点图如图 1a。可见，原点处有一二阶零点，右半平面单位圆内有两个共轭极点。由此可知，该系统是一个带通滤波器，且系统稳定。

用 freqz 绘制频率响应：

```
freqz(b, a);
```

频率响应如图 1b。可见，确实是一个带通滤波器，且共振峰频率约为 $0.25 \times \pi\ rad/sample$。在 8000 Hz 的采样频率下，即 $1000Hz$，与理论计算相符。

用 impz 绘制单位样值响应：

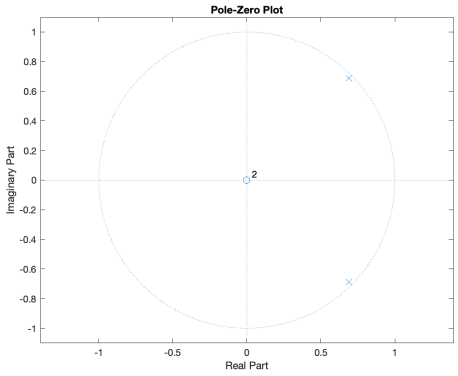
```
impz(b, a, 200);
```

单位样值响应如图 2a。

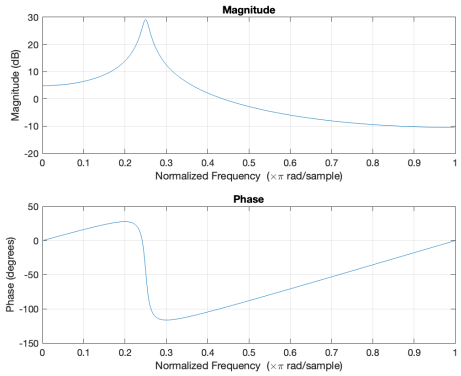
用 filter 绘制单位样值响应：

```
i = [1, zeros(1, 199)];
o = filter(b, a, i);
figure;
stem(o);
xlabel('n (samples)');
ylabel('Amplitude');
title('Impulse Response of using Filter');
```

单位样值响应如图 2b。两种方式得到的单位样值响应一致。

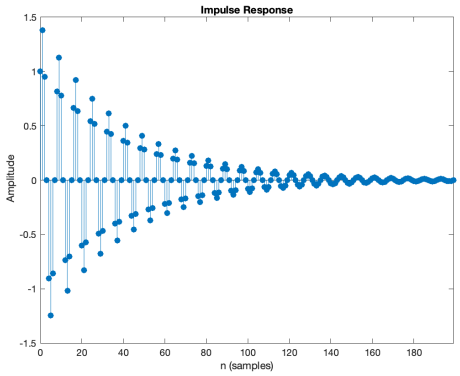


(a) 用 zplane 绘制的零极点图

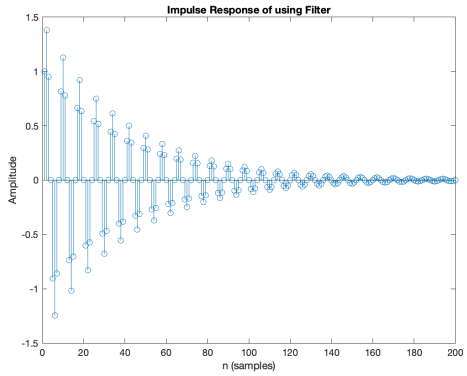


(b) 用 freqz 绘制的频率响应

图 1: 零极点图与频率响应



(a) 用 impz 绘制的单位样值响应



(b) 用 filter 绘制的单位样值响应

图 2: 单位样值响应

1.2

该程序的基本流程如下：

i 定义帧长、窗口大小等参数。

ii 加载音频文件，计算相关配置。

iii 初始化合成的语音信号、合成滤波器的初始状态等。

iv 依次处理每帧语音。首先用线性预测法计算系统系数，接着用 `filter` 函数根据系统函数计算激励，然后利用逆系统滤波重建语音信号。

v 同时，可以根据基音周期及合成激励的能量合成激励，并用合成激励和 `filter` 函数产生合成语音。

vi 此外，还可以通过更改或保持基音周期、共振峰频率、合成激励的长度等参数，调整合成语音的速度、音调等特征。

vii 在第 27 帧，观察预测系统的零极点图。

viii 最后试听并画出所合成的语音，并保存文件。

1.3

第 27 帧时绘制预测系统的零极点图：

```
if n == 27
    B = [1, zeros(1, P)];
    zplane(A, 1);
end
```

零极点图如图 3。可见，原点处有一十阶极点，单位圆内有 5 对共轭零点。由于声道模型是预测模型的逆系统，因此其零极点图与预测模型的零极点图相反。

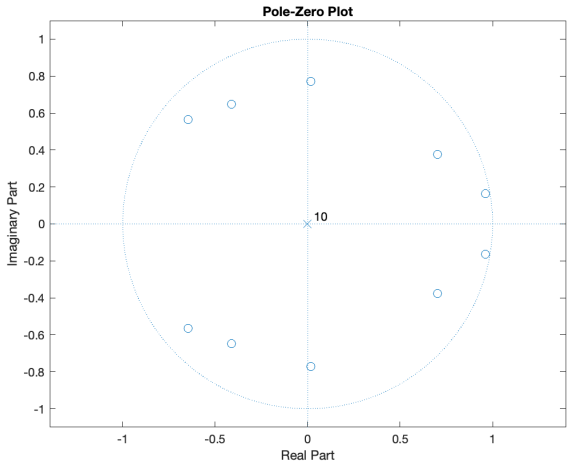


图 3: 第 27 帧时预测系统的零极点图

1.4

用 `filter` 计算激励信号：

```
[exc((n - 1) * FL + 1 : n * FL), zi_pre] = filter(A, 1, s_f, zi_pre);
```

注意到，这里记录了滤波器的状态，在下一帧作为初始状态使用，以保证连续性。

1.5

用 `filter` 重建语音信号：

```
[s_rec((n - 1) * FL + 1 : n * FL), zi_rec] = filter(1, A, exc((n - 1) * FL + 1 : n * FL), zi_rec);
```

同样，这里记录了滤波器的状态，在下一帧作为初始状态使用，以保证连续性。同时，重建系统是预测系统的逆系统，因此只需将 `filter` 的参数互换即可。

1.6

试听程序：

```
function sig_sound(s, fs)
    % Play the sound of the signal
    % s [array]: signal
    % fs [float]: sampling frequency
    sound(s / max(abs(s)), fs);
    pause(length(s) / fs);
end
```

注意这里对信号进行了归一化处理，以保证播放时的音量合适，不超过最大音量。

通过运行 `sig_sound([s; exc; s_rec], 8000);`，可以听到原始语音、合成激励、合成语音。语音内容均为“电灯比油灯进步多了”。其中，原始语音和合成语音的声音均清晰、自然，且无法区分。而合成激励的声音则音量较低，且有明显的噪音、颗粒感。

为研究三者的区别，定义了绘制波形的函数，该方法在后续的实验中也会经常使用。主要逻辑如下（省略了部分辅助功能，如保存或显示图片等）：

```
function sig_plot_t(ss, t, titles, save_prefix)
    % Plot the signals in time domain
    % ss [cell]: signals
    % t [array]: time
    % titles [cell]: titles of the signals
    % save_prefix [str][optional]: prefix of the saved images
    max_y = max(cellfun(@max, cellfun(@abs, ss, 'UniformOutput', false)));
    figure;
    for i = 1 : length(ss)
        subplot(length(ss), 1, i);
        plot(t, ss{i});
        title(titles{i});
        ylabel('Amplitude');
        ylim([-max_y, max_y]);
    end
    xlabel('Time (s)');
end
```

同时，定义绘制频域的函数，该方法在后续的实验中也会经常使用。主要逻辑如下（省略了部分辅助功能，如保存或显示图片等）：

```
function sig_plot_f(SS, f_max, titles, save_prefix)
    % Plot the signals in frequency domain
    % SS [cell]: signals
    % f_max [int]: maximum frequency
    % titles [cell]: titles of the signals
    % save_prefix [str][optional]: prefix of the saved images
    figure;
    for i = 1 : length(SS)
        subplot(length(SS), 1, i);
        plot(abs(SS{i}(1 : f_max)));
        title(titles{i});
        ylabel('Magnitude');
    end
    xlabel('Frequency (Hz)');
end
```

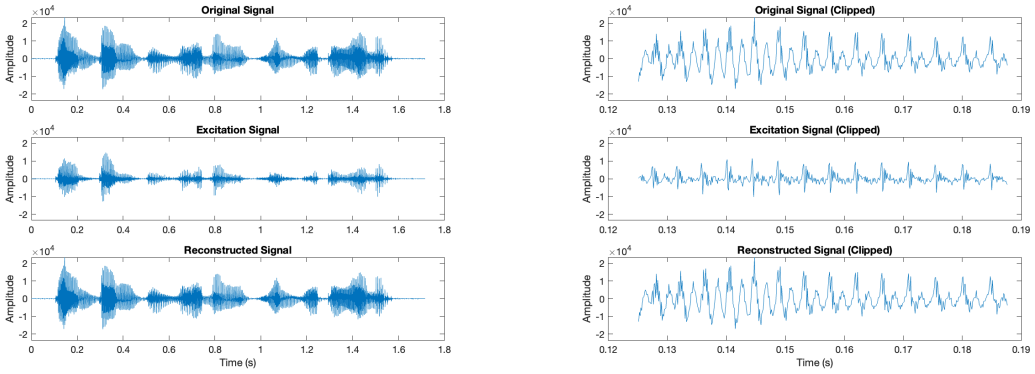
调用该函数如下（之后的调用不再重复给出）：

```
% plot the signals in time domain
t = [1 : L] / 8000;
titles = {'Original Signal', 'Excitation Signal', 'Reconstructed
↪  Signal'};
sig_plot_t({s, exc, s_rec}, t, titles, './report/asserts/1_6');
% plot clipped signals in time domain
start_s = 1000;
end_s = 1500;
s_clip = s(start_s : end_s);
exc_clip = exc(start_s : end_s);
s_rec_clip = s_rec(start_s : end_s);
t = [1 : (end_s - start_s + 1)] / 8000 + start_s / 8000;
titles = {'Original Signal (Clipped)', 'Excitation Signal (Clipped)',
↪  'Reconstructed Signal (Clipped)'};
sig_plot_t({s_clip, exc_clip, s_rec_clip}, t, titles,
↪  './report/asserts/1_6_clipped');
% plot the signals in frequency domain
titles = {'Original Signal Spectrum', 'Excitation Signal Spectrum',
↪  'Reconstructed Signal Spectrum'};
sig_plot_f({fft(s), fft(exc), fft(s_rec)}, 8000, titles,
↪  './report/asserts/1_6');
```

整体波形如图 4a，局部波形如图 4b，频谱如图 5。

可见，原始语音、合成语音的波形和频谱基本一致，与听感一致。这是由于预测模型与重建模型互为逆系统，经过预测和重建后，语音信号能够保持不变。

而合成激励的波形和频谱则有明显差异。例如幅度较小、高频噪声较多，与听感一致。



(a) 整体波形　　(b) 局部波形

图 4: 原始语音、合成激励、合成语音波形

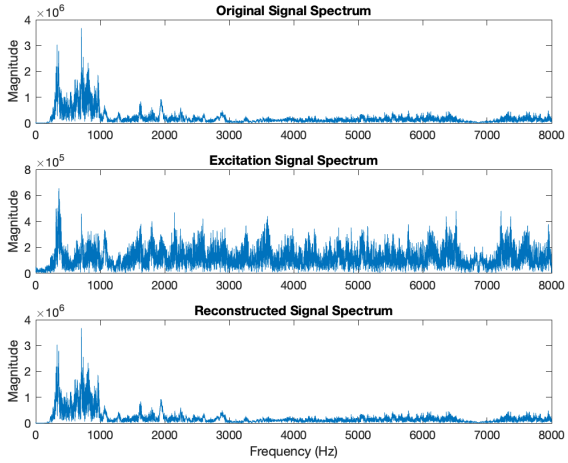


图 5: 原始语音、合成激励、合成语音频谱

这表明在语音生成模型中，声门脉冲串含有丰富的谐波。

2　语音合成模型

2.1

对单位样值串 $x(n)=\sum_{i=0}^{NS-1}\delta(n-iN)$，采样频率 $sr=8000Hz$，基音频率 $f_0=200Hz$，则基音周期 $N=sr/f_0=40$。若持续时间 $t=1s$，则 $NS=f_0\times t=200$。

生成该基音周期固定的单位样值串，该方法在后续的实验中也会经常使用：

```
function e = digit_sig_gen_const(PT, N)
    % Generate unit impulse signal
    % PT [int]: period
    % N [int]: length of the signal, including the zero padding
    % return [array]: unit impulse signal
    e = zeros(N, 1);
    e(1 : PT : N) = 1;
end
```

生成并试听 200 Hz 和 300 Hz 的单位样值串：

```
e200 = digit_sig_gen_const(8000 / 200, 8000);
e300 = digit_sig_gen_const(8000 / 300, 8000);
sig_sound([e200; e300], 8000);
```

可以听到，300 Hz 的音调更高，约高半个八度。这与理论八度数 $\log_2(\frac{300}{200}) \approx 0.585$ 相符。

2.2

利用循环生成基音周期按指定规律变化的单位样值串：

```
function e = digit_sig_gen_addon(N, seg_N, func)
    % Generate unit impulse signal with variable period
    % N [int]: length of the signal, including the zero padding
    % seg_N [int]: length of each segment
    % func [function]: function to generate the period of each
    ↪  segment
    % return [array]: unit impulse signal with variable period
    e = zeros(N, 1);
    idx = 1;
    while idx <= N
        e(idx) = 1;
        seg_idx = floor(idx / seg_N);
        PT = func(seg_idx);
        idx = idx + PT;
    end
end
```

其中 func 指示了基音周期随段序号的变化规律，在本例中，$func = 80+5\mathrm{mod}(seg\_idx, 50)$。

生成并试听基音周期按指定规律变化的单位样值串：

```
e_addon = digit_sig_gen_addon(8000, 80, @(x) 80 + 5 * mod(x, 50));
sig_sound(e_addon, 8000);
```

听上去音调的确是逐渐变化的，且变化规律符合预期。

2.3

将此基音周期按指定规律变化的单位样值串作为合成激励，输入 1.1 中的滤波器：

```
s_addon = filter(1, [1, -1.3789, 0.9506], e_addon);
sig_sound(s_addon, 8000);
```

听上去，相比原信号，合成信号更沉闷，不再刺耳，类似通过一个管道发出的声音。

做出原信号、合成信号的波形和频谱如图 6 和 7。

可见，合成信号的波形变得更加平滑，与听感一致。同时，频谱在 1000 Hz 附近有明显的峰值，与预测模型的共振峰频率相符。同时，频谱以 4000 Hz 为对称轴，与离散信号频谱周期性的特点相符。

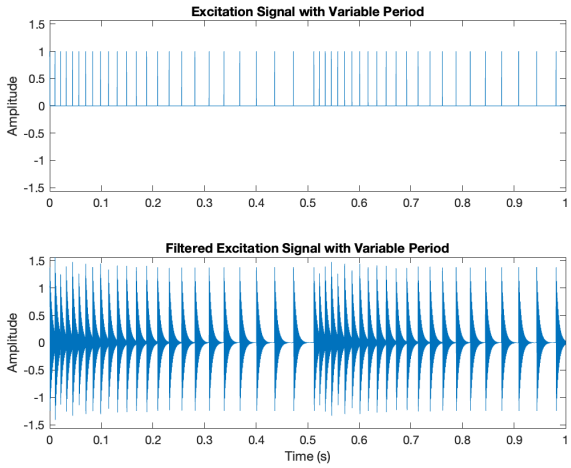


图 6: 原始语音、合成语音波形

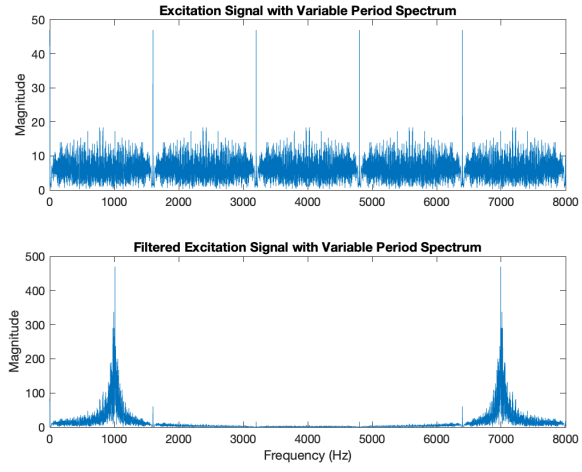


图 7: 原始语音、合成语音频谱

2.4

合成激励与语音：

```
exc_syn((n - 1) * FL + 1 : n * FL) = G * digit_sig_gen_const(PT, FL);
[s_syn((n - 1) * FL + 1 : n * FL), zi_syn] = filter(1, A, exc_syn((n
↪  - 1) * FL + 1 : n * FL), zi_syn);
```

其中，合成激励的基音周期为 PT，合成语音的增益为 G。zi_syn 记录了滤波器的状态，在下一帧作为初始状态使用，以保证连续性。

试听并画出时域波形和频谱如图 8 和 9：

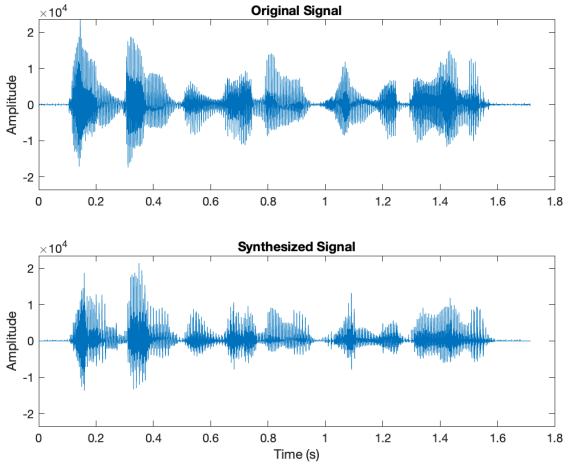


图 8: 合成语音波形

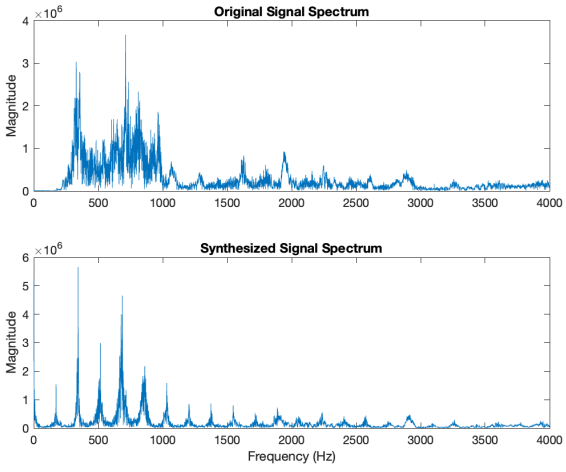


图 9: 合成语音频谱

相比原信号，合成信号内容相近，音调、语速等特征也相近，但存在一定失真，颇有种机械合成的感觉。

波形上，两者变化趋势相仿，但并不完全相同，存在一定失真。频谱上，原信号的频率分量更丰富，而合成信号的频率分量更单一。

3 变速不变调

3.1

激励的长度增加一倍，即帧长增加一倍 `FL_v = 2 * FL`：

```
exc_syn_v((n - 1) * FL_v + 1 : n * FL_v) = G * digit_sig_gen_const(PT, FL_v);
[s_syn_v((n - 1) * FL_v + 1 : n * FL_v), zi_syn_v] = filter(1, A, exc_syn_v((n - 1) * FL_v + 1 : n * FL_v), zi_syn_v);
```

试听并画出时域波形和频谱如图 10 和 11：

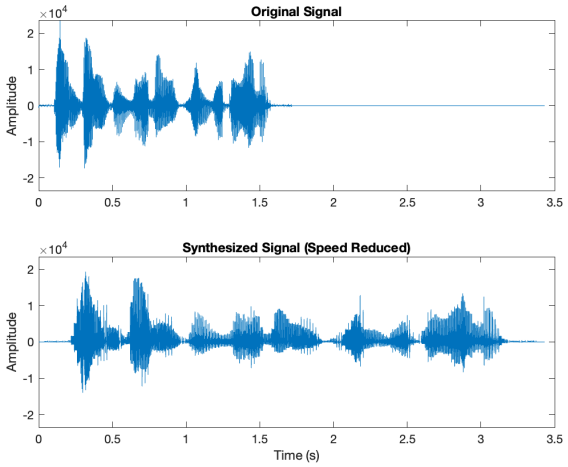


图 10: 降速不变调合成语音波形

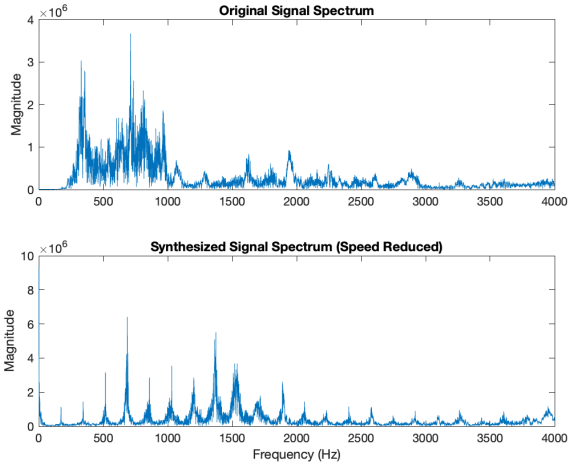


图 11: 降速不变调合成语音频谱

相比原信号，降速不变调合成语音实现了变速不变调的效果，音调不变，语速变慢。

波形上，合成语音近似原信号在时间轴上拉长了一倍，与预期相符。频谱上，合成语音在更高的频率上出现了更多峰值，这是因为时间上的拉伸会导致频谱的压缩，但音调不变

意味着基本频率和谐波结构应该保持不变。为了实现这一点，新的频谱成分需要分布在更宽的频率范围内，以保证原来的音调特征，补偿低频区域频谱的压缩效应。

4 变调不变速

4.1

改变系统的共振峰频率：

```
function rot_a = sys_rot_gen(a, angle)
    % Generate the rotated system
    % a [array]: denominator coefficients of the system
    % angle [float]: angle of rotation in radians
    % return [array]: denominator coefficients of the rotated system
    poles = roots(a);
    real_poles = poles(imag(poles) == 0);
    imag_poles = poles(imag(poles) > 0);
    imag_poles = imag_poles .* exp(1i * angle);
    all_poles = [imag_poles; conj(imag_poles); real_poles];
    rot_a = poly(all_poles) * a(1);
end
```

这里将一、二象限的复极点按指定角度旋转，然后用 `poly` 函数生成新的系统系数。同时，保持实极点不变。

将 1.1 中的系统共振峰频率增加 150 Hz，相当于将一、二象限的复极点逆时针旋转 $\frac{150}{8000} \times 2\pi$：

```
rot_A = sys_rot_gen(A, 150 * 2 * pi / 8000);
```

做出该系统的频率响应如图 12。

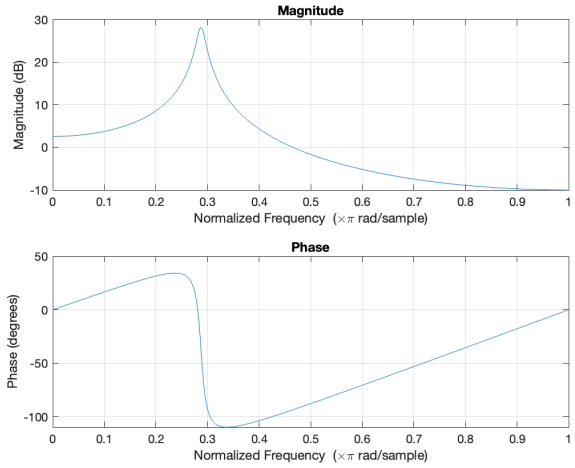


图 12: 共振峰频率增加 150 Hz 的系统频率响应

可见，共振峰频率约为 $0.2876 \times \pi\ rad/sample$，即 1150 Hz，与预期相符。

4.2

合成激励和语音：

```
exc_syn_t((n - 1) * FL + 1 : n * FL) = G *
    digit_sig_gen_const(round(PT / 2), FL);
[s_syn_t((n - 1) * FL + 1 : n * FL), zi_syn_t] = filter(1, rot_A,
    exc_syn_t((n - 1) * FL + 1 : n * FL), zi_syn_t);
```

其中，合成激励的基音周期为 `PT / 2`，合成语音的增益为 `G`，系统系数为 `rot_A`。`zi_syn_t` 记录了滤波器的状态，在下一帧作为初始状态使用，以保证连续性。

试听并画出时域波形和频谱如图 13 和 14：

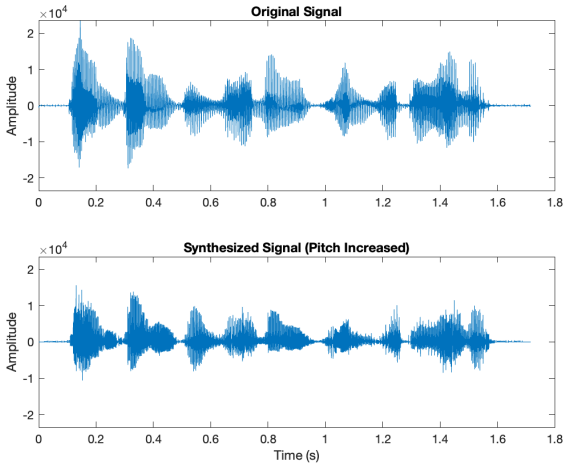


图 13: 提升音调不变速合成语音波形

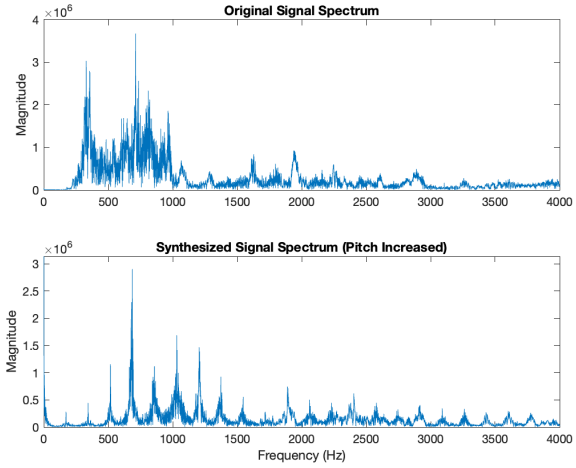


图 14: 提升音调不变速合成语音频谱

相比原信号，提升音调不变速合成语音实现了变调不变速的效果，音调提升，语速不

变。

波形上，合成语音变得更密集，但整体形状与原信号相似，与变调不变速的预期相符。频谱上，合成语音在更高的频率上出现了更多峰值，这印证了在音调提升的同时语速不变的现象。

5 实验总结

本次实验主要研究了语音合成的基本原理和方法，通过实现语音预测模型和语音合成模型，实现了语音的合成。在此基础上，进一步研究了变速不变调和变调不变速的方法，实现了这两种效果。

在实验中，我学会了如何用滤波器合成激励和语音，如何调整系统系数以实现变速不变调和变调不变速。同时，我学会了如何绘制波形和频谱图，如何试听合成语音，如何分析合成语音的特征。

除此以外，本次实验锻炼了我的 Matlab 编程能力，提高了我的信号处理和数字信号处理的实践能力。通过实验，我更加深入地理解了语音合成的原理和方法，对信号与系统的应用有了更深的认识。