UC Davis
EEC 201
Final Report

Project B: Vocoder

Ryan Bunk

Hui-Ying Siao

1. Introduction

The overall objective of this project is to design and implement a vocoder that encodes information using the LPC algorithm. The vocoder should be able to compress the input bit stream to less than 16 kbits per second, save the information, and replay the input information as audio. Specifically, the vocoder should be implemented in Matlab, with an interactive GUI to operate the vocoder.

2. Methods

In this project, we experimented with many possible implementations of the LPC vocoder. While we used the default LPC function in Matlab to handle the generation of gain and LPC coefficients, the pitch detection and decoding required some experimentation to get correct results. The pitch detection proved the most technically difficult aspect, but two methods produced usable results. These two methods were STFT(Short Time Fourier Transform) and Cepstral Analysis[1][2][3].

2.1 STFT(Short Time Fourier Transform)

Fundamentally, the data to be transformed is broken up into windowed frames, and each frame analyzed independently. The general equation for STFT is shown below:

$$\mathbf{STFT}\{x[n]\}(m,\omega) \equiv X(m,\omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n}$$

However, this approach alone was insufficient for pitch detection even with small noise or harmonics, so a more elegant approach was required. We found that by multiplying the fourier transform by a nonlinear function, the peaks could be more easily characterized versus the raw output. Examples of these nonlinear transformations are shown below.

$$T(x[n]) = e^{FFT(x[n])}$$

or

$$T(x[n]) = (FFT(x[n])^{2a},$$

where a is an integer constant. By then normalizing the signal and detecting the first peak above a threshold, the pitch could be reliably detected. Examples of this are shown below with the spectrogram shown for the fundamental pitch range. The spectrum was generated with words spoken in pinyin shi 1-4, ni 1-4. The difference of the tones in chinese were easily picked up by the STFT algorithm.

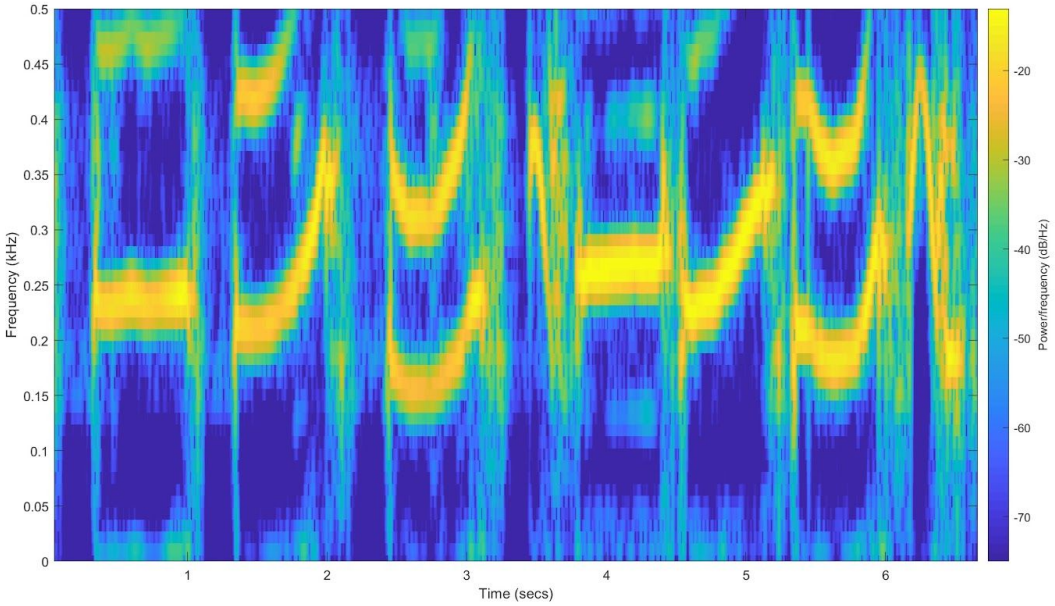

Figure 1, Spectrogram focused on fundamental pitch of words spoken in Chinese with four tones (pinyin shi 1-4, ni 1-4).

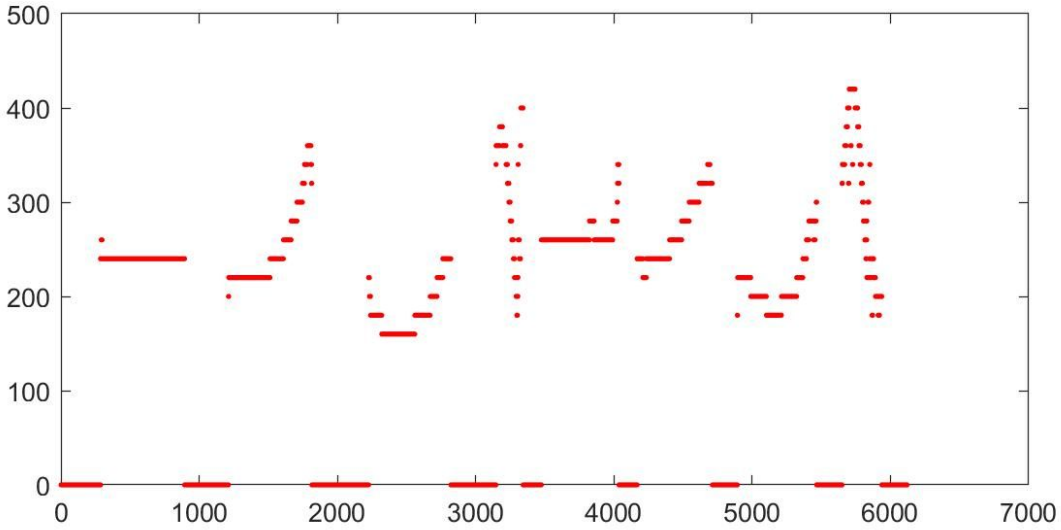

Figure 2, Detected fundamental pitch using STFT algorithm described with window of 2048 samples and overlap of 2000 samples.

2.2 Cepstrum

As a second method, a cepstral analysis-based pitch detector was created. A ceptrum is the inverse Fourier transform of the logarithm of the signal in time domain. It is used to find the fundamental frequency of human speech. In Cepstrum method, the vocal excitation and the vocal tract are separated, so as the method can work more effective. The equation describing this method is shown below:

$$C(x[n]) = ifft(\, log(\, fft(x[n])\,)^2)$$

When analyzed in time domain, peaks corresponding to multiples of the fundamental pitch are found. Simply analzying the first peak position produces very error-prone results, however, so the difference between the first and second peak was used instead to find the frequency. A plot of the cepstrum output is shown below, with peaks tagged.

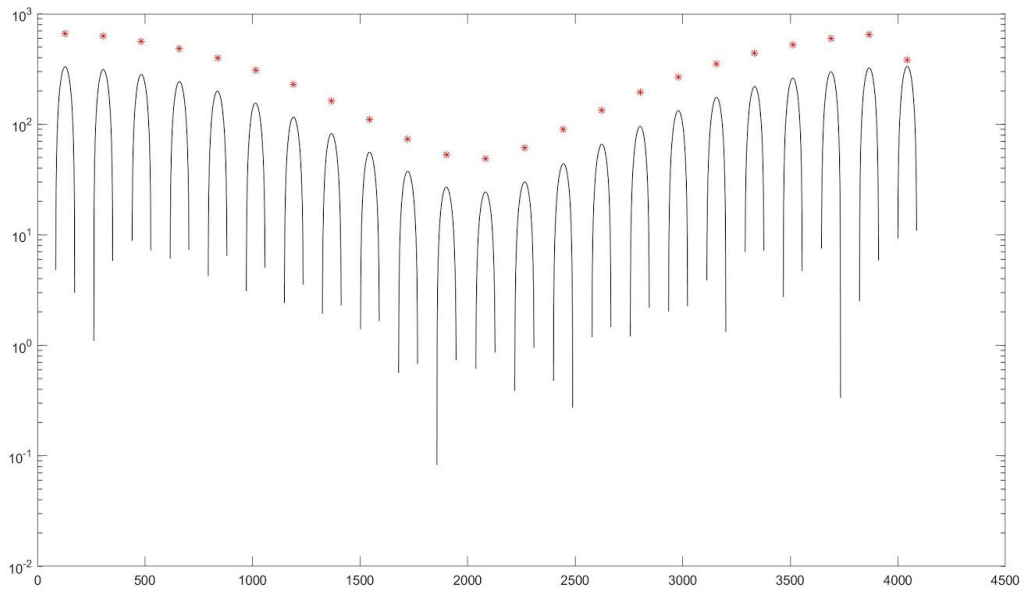

Figure 3, cepstrum of sinusoidal tone input

The pitch detection of the same chinese words is shown below:

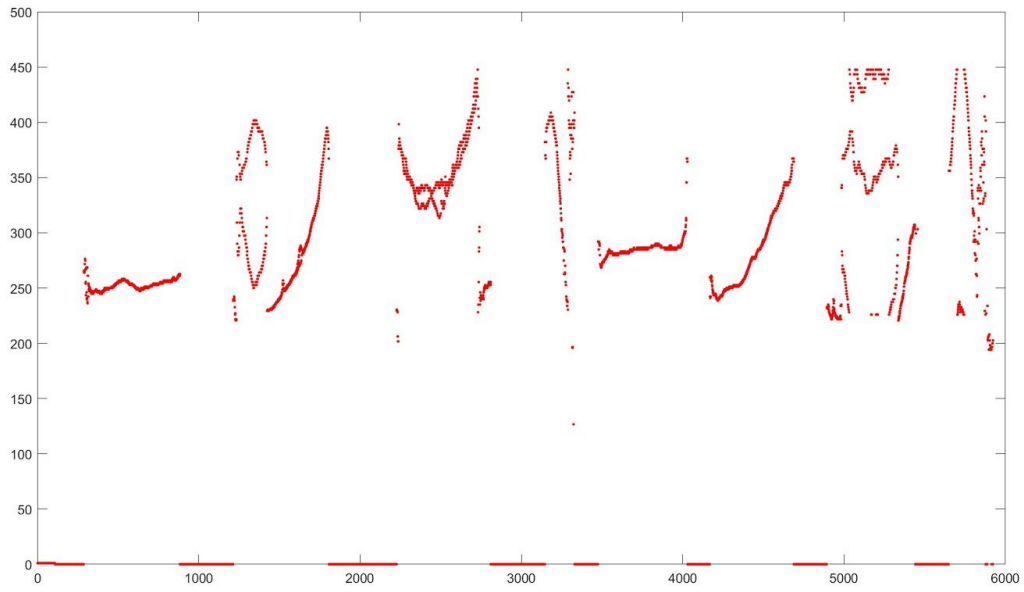

Figure 4, pitch detection of chinese words using cepstral analysis

It is immediately clear the resolution is improved at the cost of a higher error rate.

2.3 Synthesizer

To produce the synthesizer, two separate pulse and white noise generators were created. To generate pulses in matlab with controllable frequency, with high levels of harmonic content, a sinc function was used. This is shown below:

$$p(t) = sinc(k * cos(\omega t))$$

By adjusting the input ω, the pulse train frequency could be varied. The time-scaling term k was experimented with to find the optimal pulse width.

3. Results

To objectively compare the results of the two encoder/decoder, the spectrogram of the outputs was compared. The spectrograms are shown below:

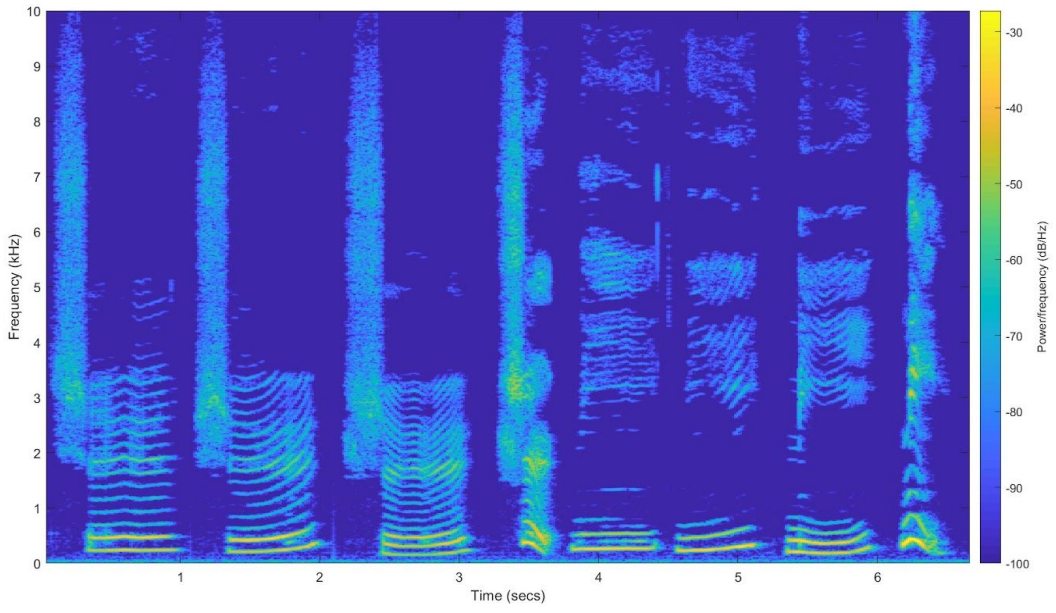

Figure 5, spectrogram of signal as-recorded

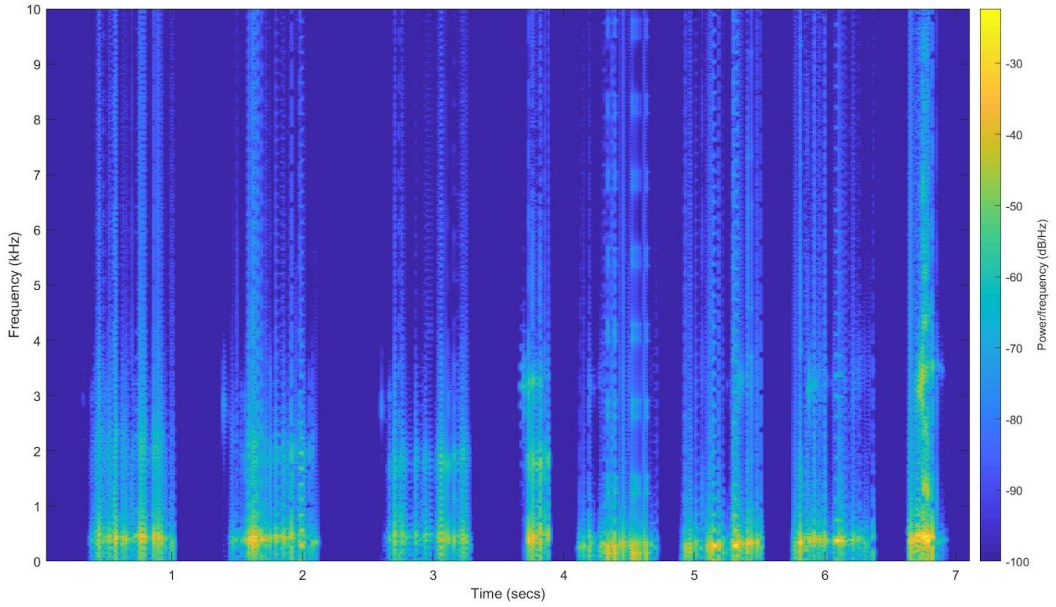

Figure 6, spectrogram of signal encoded using cepstral analysis

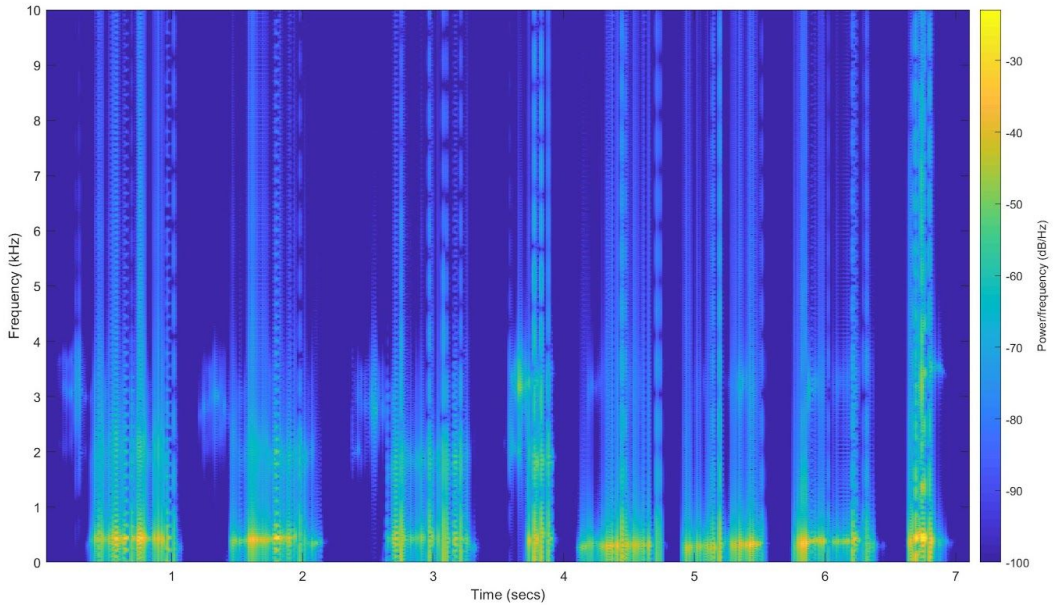

Figure 7, spectrogram of signal encoded using STFT analysis

As seen in the spectrograms, neither of the generated signals produced a spectrogram resembling the original, though the accuracy of detecting voiced sounds, unvoiced sounds, or silence is obvious. The unvoiced sound appears odd in figures 6 and 7, as the output was low and high-pass filtered to maximize intelligibility and reduce synthesis noise.

That said, from an intelligibility point of view, it was agreed that the cepstral analysis method produced more understandable results for a given bit rate. It would appear that the high frequency resolution from cepstral analysis is superior to the low glitches and accuracy of the STFT method.

Additionally, the effect of LPC coefficients and window width was explored. It was found that, even for tens of LPC coefficients, the output could be understood if the window width and overlap was large enough.

4. Conclusion

Comparing two different methods used in this project for the LPC, the cepstrum method is slightly better than the STFT method. In general, it was found that intelligibility was more strongly affected by both time and frequency resolution than the accuracy of the analysis. Intelligible outputs could be produced using a small number of LPC coefficients as well as cepstral analysis, which would have the highest error rate but highest time and frequency resolution.

5. Code source

All the code can be found in the team website:

https://github.com/ryansgithubaccount/SiaoBunk_Vocoding

6. References

[1] IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, VOL. ASSP-24, NO. 5, OCTOBER(1916)

[2] IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, VOL. ASSP-29, NO. 2, APRIL(1981)

[3] IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING, VOL. 7, NO. 3, MAY(1999)