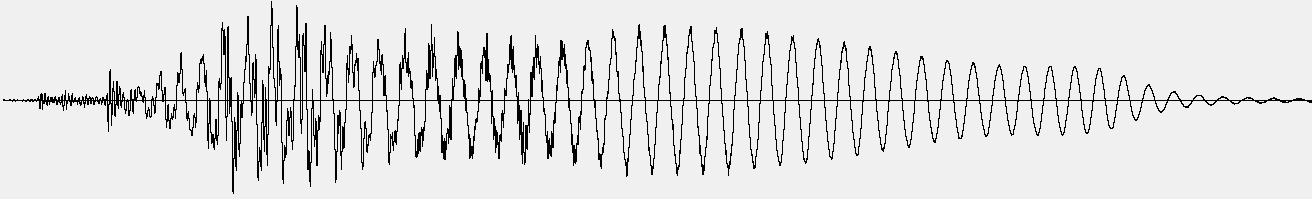
**Mini Project 2 – Spectrogram**

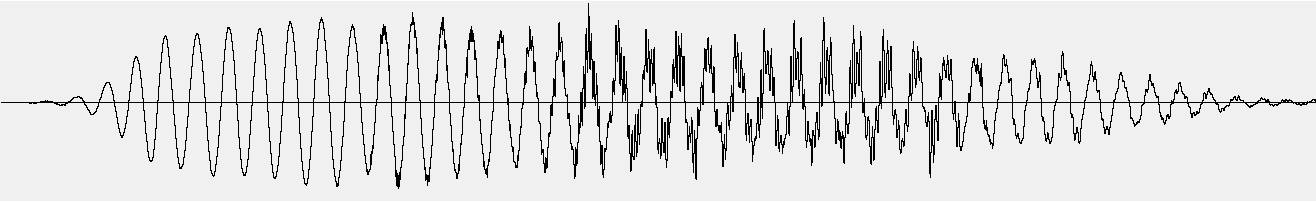
國立臺北大學 通訊工程學系 110學年度 多媒體訊號處理 (U3329)

1. **Objectives**
2. Be familiar with Discrete Fourier Transform (DFT).
3. Understand how a spectrogram is generated.
4. Mastery over the C language.
5. **Background**

Fig.1 shows waveforms of a compound vowel ‘ai’ and a syllable ‘ya.’ The compound vowel ‘ai’ and the syllable ‘ya’ have similar sound components ‘a’ and ‘i’, but are in a different order over time. It can be observed that the waveform at the beginning of ‘ai’ and the one at the end of ‘ya’ are very similar; that is, they may have similar harmonic components, since they all are the sound ‘i’ (vice versa for the case of the end of ‘ai’ vs. the beginning of ‘ya.’ These two waveforms are different speech segment. But, if you take Fourier transform for ***whole segments of ‘ai’ and ‘ya,’*** they may have very similar frequency responses in Discrete-Time Fourier Transforms (DTFTs), indicating that we cannot distinguish the two sounds ‘ai’ and ‘ya’ if the analysis window is too long to make Fourier analysis lose ***time resolution*** because speech signals change over time.



(a)



(b)

Fig.1: (a) waveform of a compound vowel ‘ai’ and (b) waveform of a syllable ‘ya.’ Note that ‘ai’ consists of two sounds ‘a’ and ‘i’ in sequence and ‘ya’ consists of two sounds ‘i’ and ‘a’ in sequence.

To catch the change in frequency components over time, as shown in Fig. 2, ***Short-Time Fourier Analysis*** or ***Short-Time Fourier Transform (STFT)*** is adopted to analyze a speech signal by decomposing the speech signal into a series of short segments, referred to as ***analysis frames***, and analyze each one independently. The waveform in each short segment (analysis frame) could be stationary since waveforms in a shorter window look more periodic and have more similar harmonic components. Also, each analysis frame could catch a different waveform to observe signal changing over time. We, therefore, can analyze phonemes and their transition by spectrogram.

A spectrogram is a visual representation of a ***spectrum sequence***, capturing spectrum changes with time. Spectrograms are sometimes called sonogram, spectral waterfalls, voiceprints, or voicegrams. In Digital Signal Processing (DSP), we can regard spectrogram, , as a function of time (frame index) and frequency bin, i.e.

 (1)

where *m* represents frame index; *k* represents frequency bin index. Note that  in Eq. 1 is in the same form with DFT.  is a short-time signal with a length of *N* samples, indicating taking DFT of a discrete-time signal of length *N*. We usually call the parameter *N* ‘*FFT window length*.’ Note that FFT stands for Fast Fourier Transform, which is efficient implementation of DFT (you’ll learn the FFT in some advanced courses). In mathematics,  is expressed by

 (2)

where  denotes a time-shifted signal of original input speech, ; *M* represents *frame interval* in sample;  is a window function, a mathematical function with zero-valued outside of some chosen interval. Therefore, we can regard  as *m*-th frame signal of length *N*. There are many types of window functions, such as rectangular window (Eq. 3), Hamming window (Eq. 4), and Hanning window (Eq. 5).

Fig.2: Overview of obtaining spectrogram. The process in sequence includes framing, taking window, zero padding, and DFT or FFT.

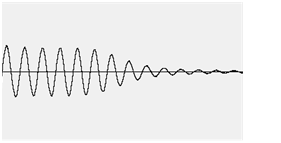
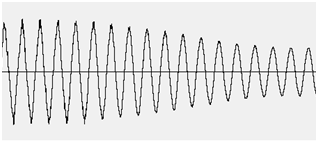
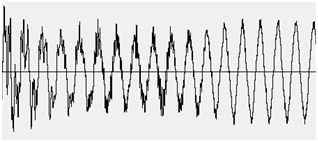
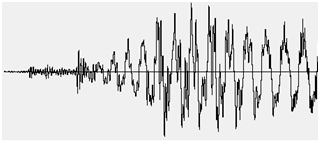
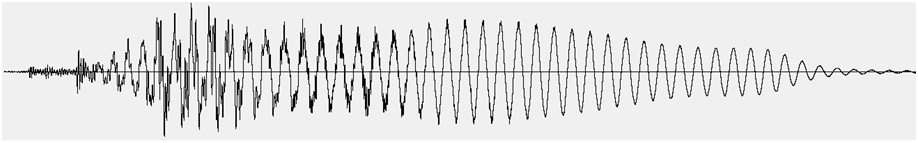




0

P-1

*n*





0

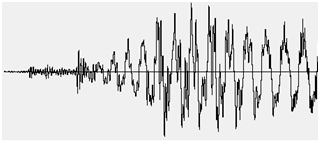
P-1

0

M

2M

3M





0

P-1

N-1









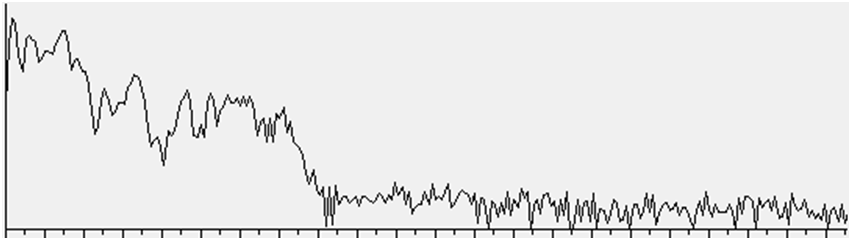


**framing**

**Taking window**

**Zeros padding**

**DFT or FFT**



0

N/2-1





*k*



frame 0

frame 1

frame 2

frame 3

Rectangular:  (3)

Hamming:  (4)

Hanning:  (5)

For analysis of speech, we usually choose Hamming window as the window function in Eq. 2. Note that in Eqs 3-5, a window function has zero value outside of an interval . We, therefore, define *P* as ***analysis window size***, regarding how many sample points are excerpted from the analyzed signal  for an analysis window. Thus, the value *P* must be equal to or smaller than the FFT window length *N*. A analyzed signal is said to be zero-padded as *P*<*N*  because  is zero for . Since we only have non-zero samples for , only *P* points excerpted from  are informative for frequency analysis. As FFT window length *N* is set to be greater than analysis window size *P*, the *N*-*P* zero-padded samples in  just let the spectrum of *m*-th frame  has a smoother outline extrapolating more frequency bins.

1. **Parameters of spectrogram by DFT**

* Analysis Window Types

***Why not using a rectangular window for spectrogram?*** Assume we have a pure sinusoidal signal to be analyzed by spectrogram, e.g.

 (6)

and the corresponding Fourier transform is

 (7)

But that spectrogram analyzes the windowed signal  , which is a distorted version of the original pure sinusoidal signal . Also, let’s recall the convolution theorem, i.e.

 (8)

From Eq. (9), we know that the DTFT  depends on the DTFT  (DTFT of the window function ). So, let’s check the  if  is a rectangular function of length *P*:

 (9)

 Is in a form of a sinc function. **So….Think about the rest by yourself. I talked too much!**

* Window Size *N*, frequency bin index *k*, and analysis bandwidth

The parameter *N* is the number of points for DFT analysis (sometimes called ‘***window size***’). Since each frequency bin index *k* in  corresponds to a frequency component of Hz of continuous-time signal (: sample rate), larger *N* means more points analyzed and more sophisticated frequency resolution. We usually define the term  (Hz) as ***analysis bandwidth***.

* Frame Interval *M*

The parameter *M* is the ***frame interval*** regarding the rate of DFT analysis on the time axis. Smaller *M* means more detailed time resolution for spectrogram. Conventionally, *M* is set to be equal or smaller than *N*, indicating each analysis frame  could have some overlapping samples to the neighboring analysis frames (e.g. ..., ,…). Typically, the ***window size*** is set to be 20ms~30ms, i.e., *N*=160 to 240 points for an 8kHZ sample-rate speech signal, or *N*=320 to 480 points for a 16kHz sample rate signal. ***Frame interval*** is usually set to be 5 to 20ms, i.e., M=40 to 160 points for an 8kHZ sample-rate speech signal, or *M*=80 to 320 points for a 16kHz sample rate signal.

1. **Implementations**
   1. Tools needed: Wavesurfer [3] and Visual Studio Code.
   2. Spectrogram by ***DFT***

Steps:

* + 1. Recording

Please record your ISOLATED five basic vowels, i.e. [ɑ], [i], [u], [ɛ], and [ɔ], one time for each sample rate of 16,000 Hz or 8,000 Hz. Hence, there are two WAVE files recorded. Sample encodings are all Lin16, i.e. linear quantized level represented by 16bits (per sample) and channels are all set to be Mono. Note that ISOLATED means you have to pronounce each vowel separately. You can find apparent silence (or short pause) between vowels. After recording, two isolated-vowel WAVE files are obtained. Please name the two files vowel-16k.wav and vowel-8k.wav.

* + 1. Generate the 8 sine waves (\*.wav) that satisfy the following specifications:
       1. Sample rates: 16,000 Hz or 8,000 Hz
       2. The corresponding 8 continuous time signals are ; where *f* = 50, 200, 55, 220 Hz; .
       3. Please name the 8 wave files: cos\_050Hz-16k.wav, cos\_220Hz-16k.wav…cos\_050Hz-8k.wav…cos\_220Hz-8k.wav
    2. Save the spectrograms  for the WAVE files vowel-16k.wav, vowel-8k.wav, cos\_050Hz-16k.wav, cos\_220Hz-16k.wav…cos\_050Hz-8k.wav…, and cos\_220Hz-8k.wav with the following settings, to files in ascii:

Setting 1:

Analysis window size = 5ms

Analysis window type = rectangular

DFT/FFT window size = 8ms

Frame interval = 5ms

Setting 2:

Analysis window size = 5ms

Analysis window type = hamming

DFT/FFT window size = 8ms

Frame interval = 5ms

Setting 3:

Analysis window size = 20ms

Analysis window type = rectangular

DFT/FFT window size = 32ms

Frame interval = 10ms

Setting 4:

Analysis window size = 20ms

Analysis window type = hamming

DFT/FFT window size = 32ms

Frame interval = 10ms

Therefore, 40 ascii files are saved: vowel-16k.{Set1~Set4}.txt, vowel-8k.{Set1~Set4}.txt, cos\_050Hz-16k.{Set1~Set4}.txt…cos\_220Hz-16k.{Set1~Set4}.txt… cos\_050Hz-8k.{Set1~Set4}.txt, …and cos\_220Hz-8k.{Set1~Set4}.txt

***Note that you should use DFT to compute Eq. 1.***

* + 1. Show the spectrogram datum in vowel-16k.{Set1~Set4}.txt, vowel-8k.{Set1~Set4}.txt, cos\_050Hz-16k.{Set1~Set4}.txt …and cos\_220Hz-8k.{Set1~Set4}.txt by Matlab Image plot (imagesc). The results would look like the ones shown in the spectrogram by the wavesurfer. Please show the frequency in Hz on the y-axis and frame index on the x-axis.
    2. Compare the results by Settings 1-4 and discuss the differences and their significances.
    3. Calculate how many multiplications and additions are executed for Settings 1-4.
  1. Requirements
     1. Please write a report to record every step you make in the implementation as mentioned above steps with word descriptions and figures excerpted from your screen.
     2. Upload your report with your C programs processing the implementations. Note that adding comments on your C codes is necessary.

References:

1. Xuedong Huang, Alex Acero, Hsiao-Wuen Hon (2001). *Spoken Language Processing: a guide to theory, algorithm, and system development*, pages 274-281. Prentice Hall
2. https://www.speech.kth.se/wavesurfer/