REAL TIME AUDIO ANALYTICS AND RECOGNITION

Dr TIAN Jing

tianjing@nus.edu.sg

Module objective

Module: Audios sensing and analytics

Knowledge and understanding

- Understand the fundamentals of audio signal processing, feature extraction and representation for audio analytics

Key skills

- Design, build, implement intelligent audio analytics methods

Major reference

- [Introduction] T. Virtanen, M. Plumbley, and D. Ellis, "**Computational analysis of sound scenes and events**," https://cassebook.github.io
- [Advanced] T. Giannakopoulos, ***Multimodal Information Processing & Analysis***, https://github.com/tyiannak/multimodalAnalysis
- [Practical] ***PyAudioAnalysis***, A Python library for audio feature extraction, classification, segmentation and applications**,** https://github.com/tyiannak/pyAudioAnalysis
- ***Deep learning for audio with Python***, https://github.com/musikalkemist/DeepLearningForAudioWithPython
- ***Audio processing for machine learning***, https://github.com/musikalkemist/AudioSignalProcessingForML

Overview

Objective: Extract high-level descriptions from raw audio signals (sounds) using either signal analysis to extract features and representations, or machine learning (supervised or unsupervised) to discover patterns.

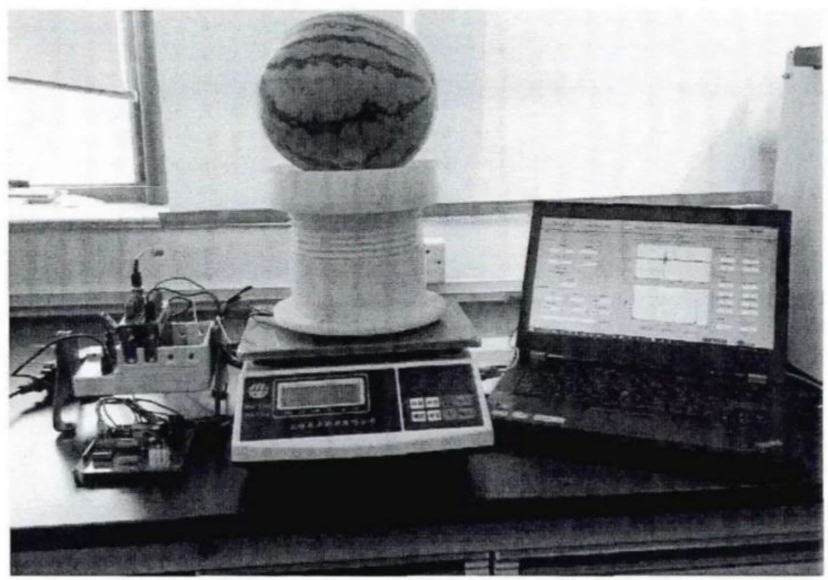
Applications	**Audio**	**Speech**	**Music**
Automatic Speech Recognition (ASR)			
Virtual Assistants			
Music Search			
Surveillance			
Environmental Monitoring			
Recommendation			

Applications

Reference
https://www.youtube.com/watch?v=hXJQm32cvs8
https://www.youtube.com/watch?v=d-JMtVLUSEg

Emotion-aware dialogs

Urban sound monitoring

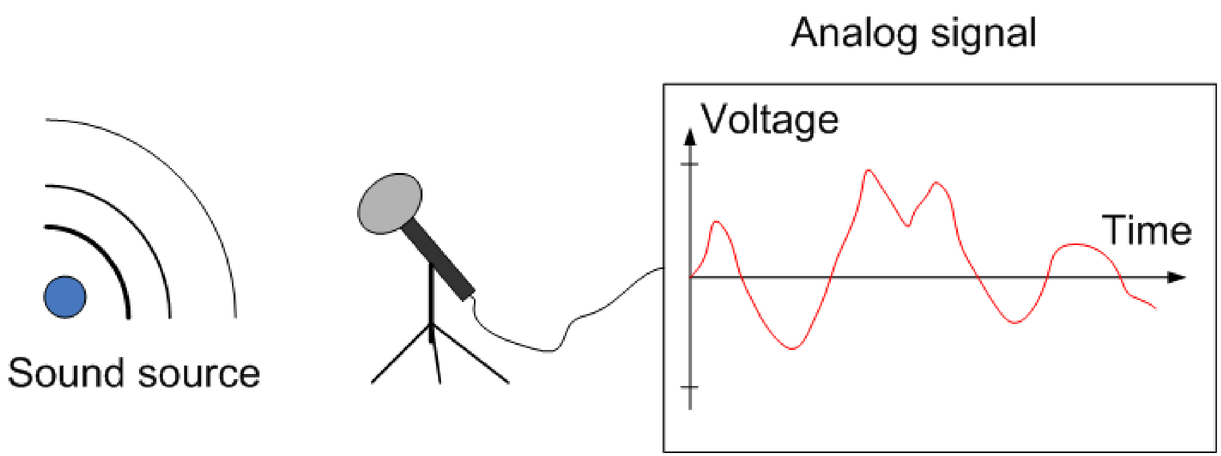


How to pick a good watermelon? Knock the melon using your phone!

Photo: https://segmentfault.com/a/1190000019956365

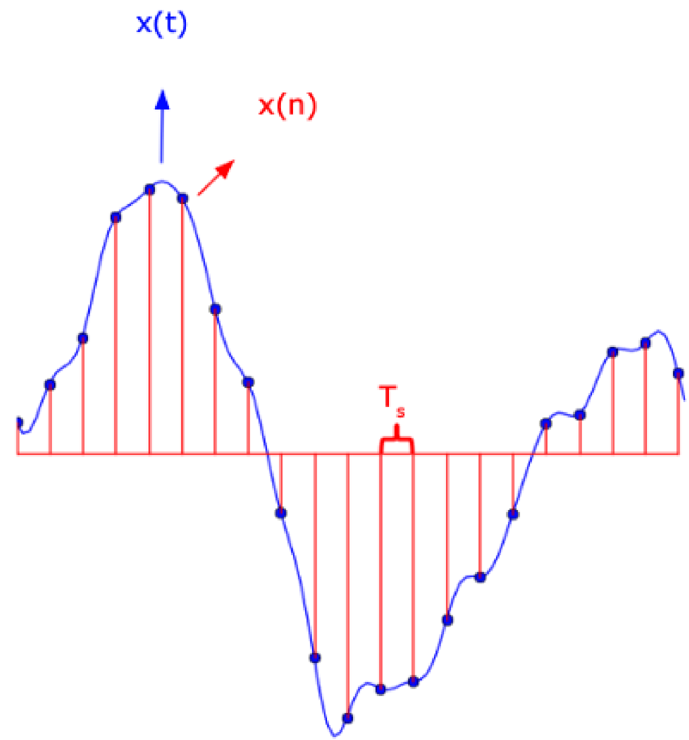
Audio signal

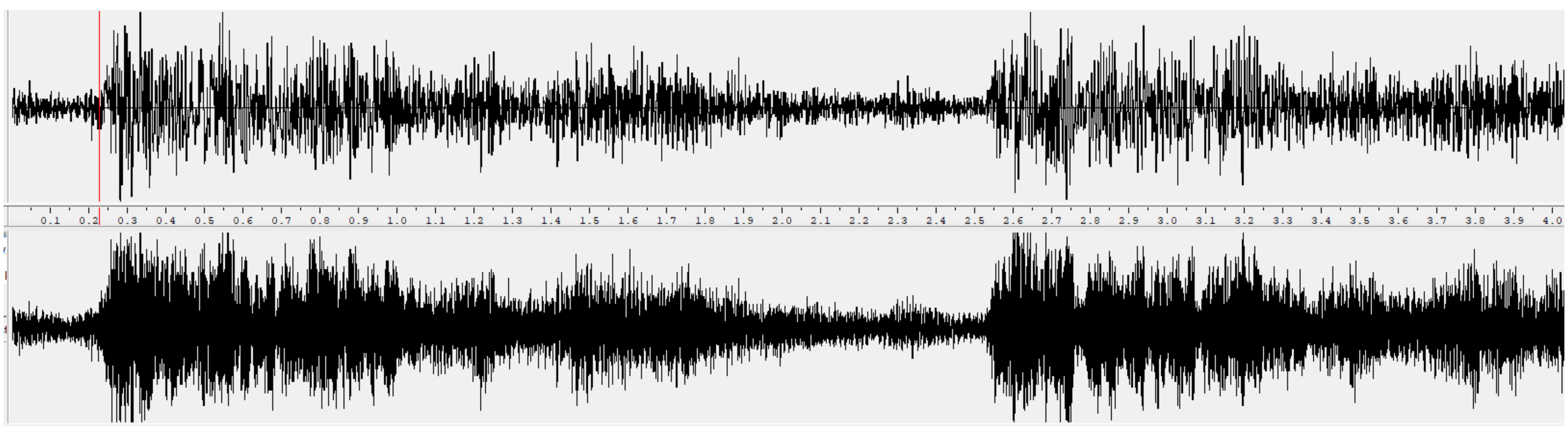
- Sound (physics): A travelling vibration (wave) through a medium (e.g. air) transfers energy (particle to particle) until “perceived” by our ears
- Amplitude: Loudness
- Frequency: Vibrations per second



Audio signal

Sampling rate: Convert the time-varying continuous signal $x(t)$ to a discrete sequence of real numbers $x(n)$. The interval between two successive discrete samples is the sampling period (T_s). We use the sampling frequency ($f_s = 1/T_s$) as the attribute that describes the sampling process. For example: CD recordings 44.1 kHz, telephone 8 kHz, PC and smartphone 16 kHz.





Low

High

Audio signal

Sampling resolution (quantization): Represent each real number, $x(n)$, of the sequence of samples with an approximation from a finite set of discrete values. We usually call this bit resolution property of the quantization procedure "sample resolution" and it is measured in bits per sample.

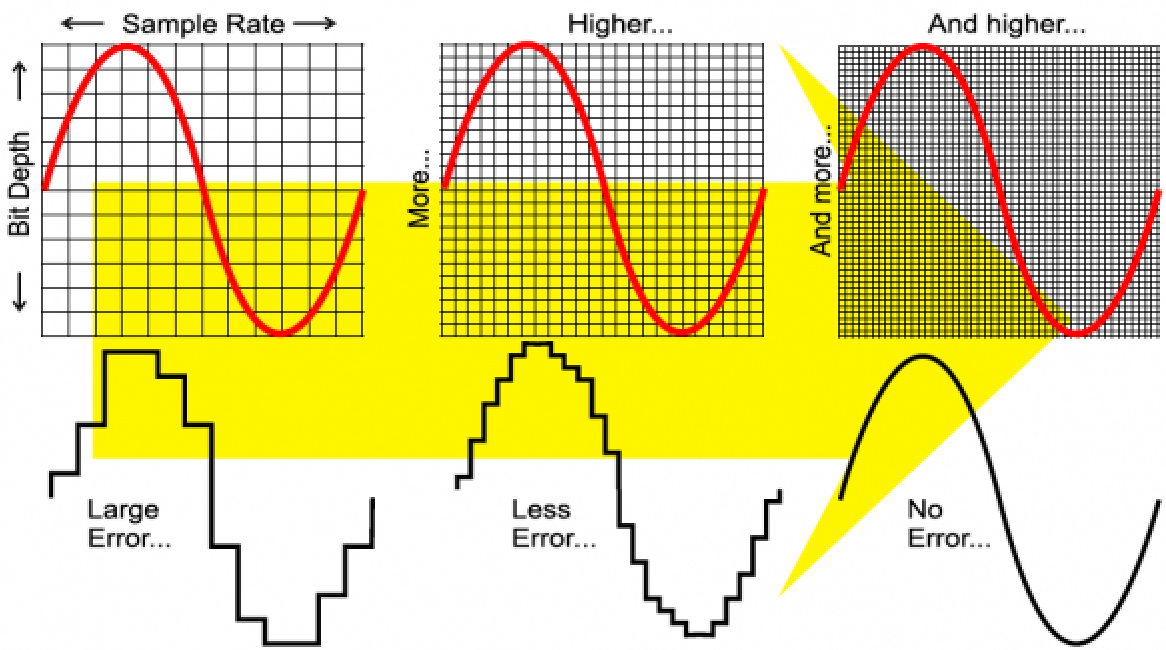


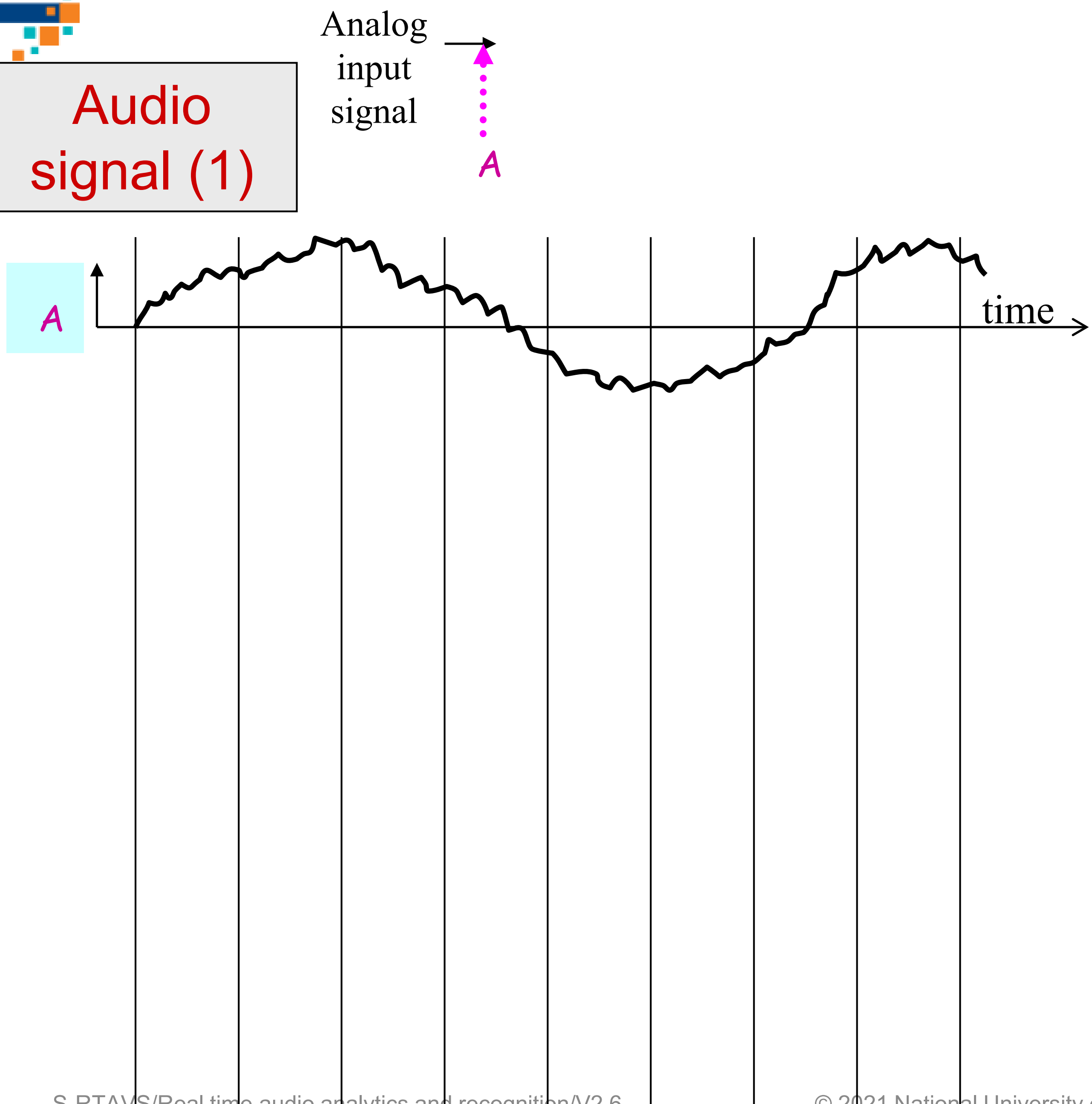
Photo: https://www.izotope.com/en/learn/digital-audio-basics-sample-rate-and-bit-depth.html

Example: Two methods to represent $(0V, 6V)$ using 2 bits and 3 bits, respectively.

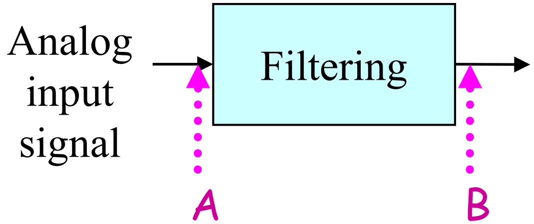
Binary	Voltage (v)
00	0
01	2
10	4
11	6

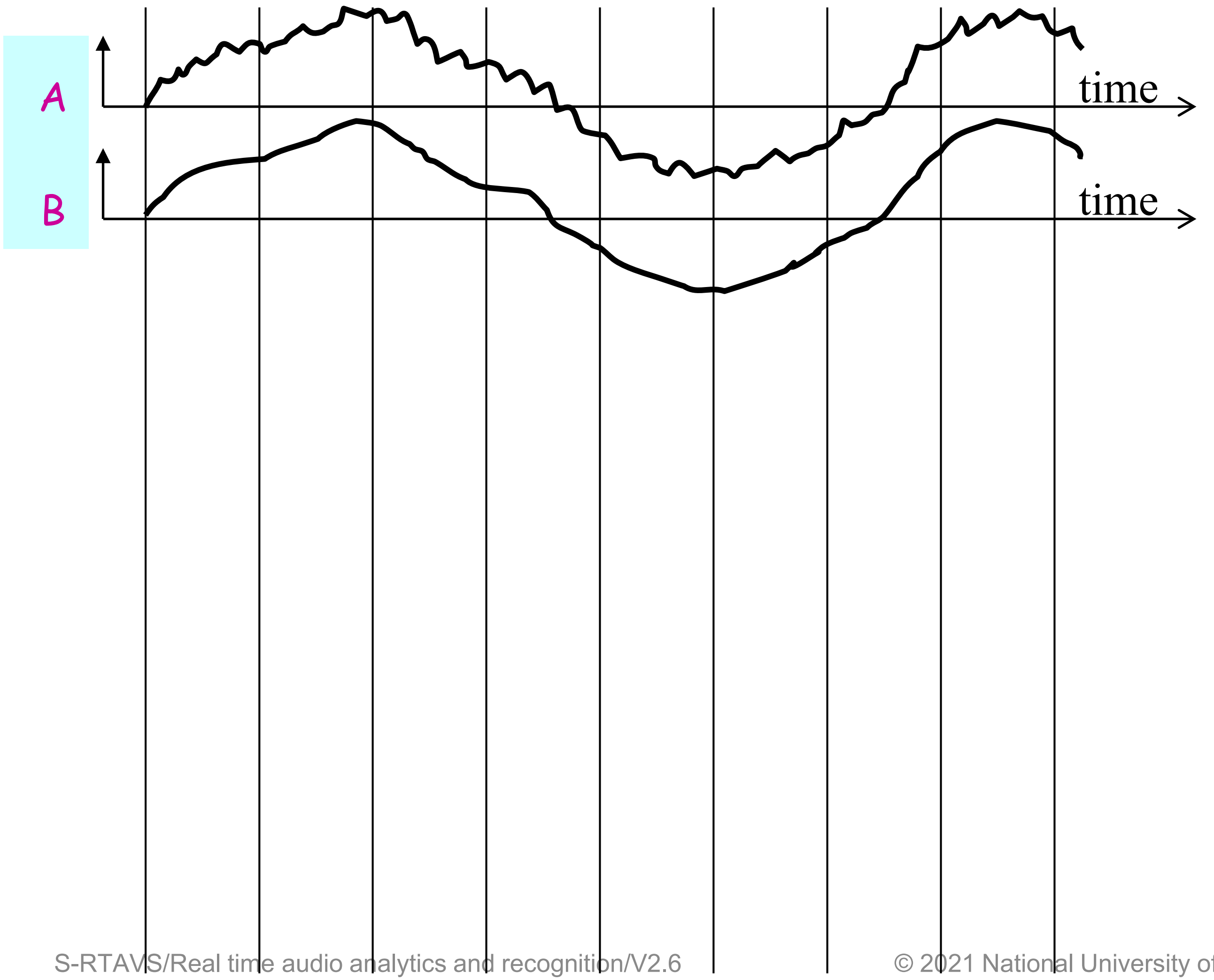
Binary	Voltage (v)
000	0
001	0.86
010	1.72
011	2.58
100	3.44
101	4.3
110	5.16
111	6

Audio signal (1)

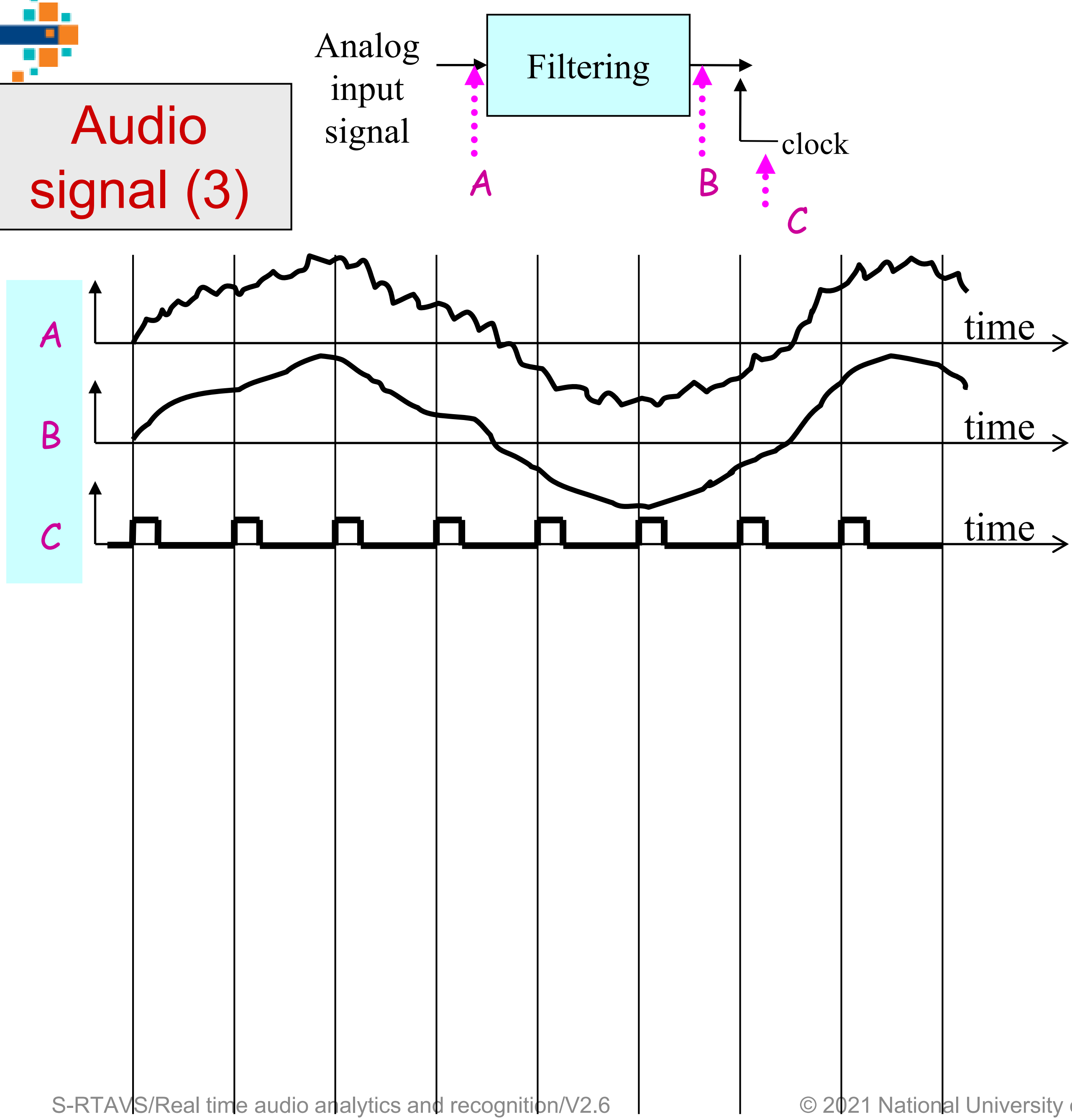


Audio signal (2)

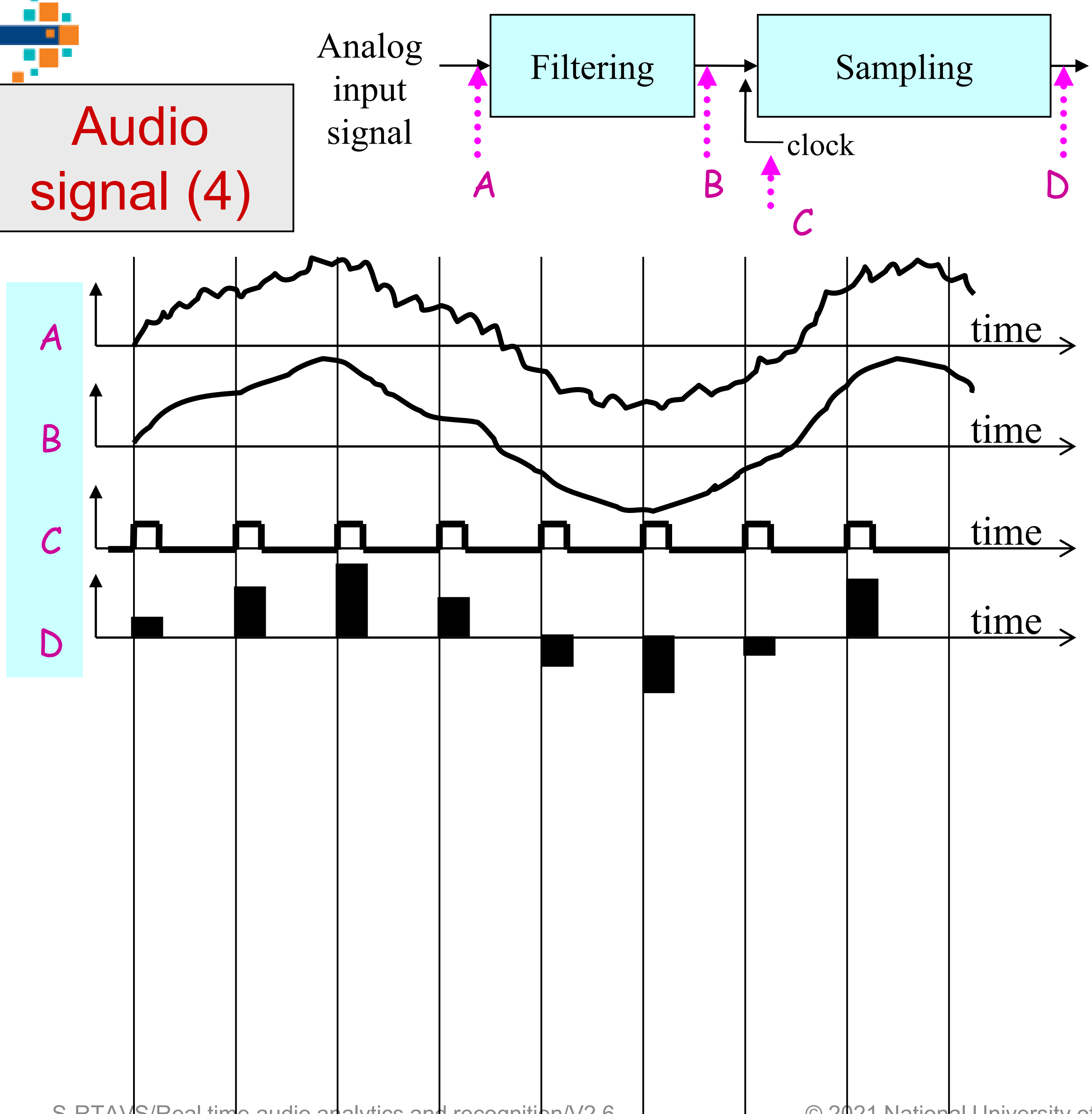




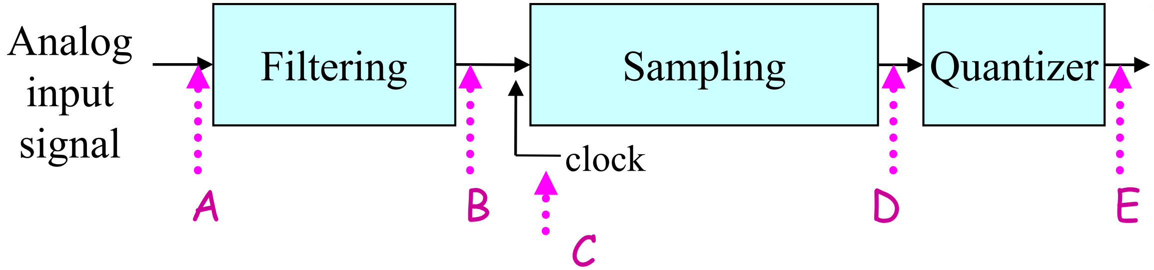
Audio signal (3)

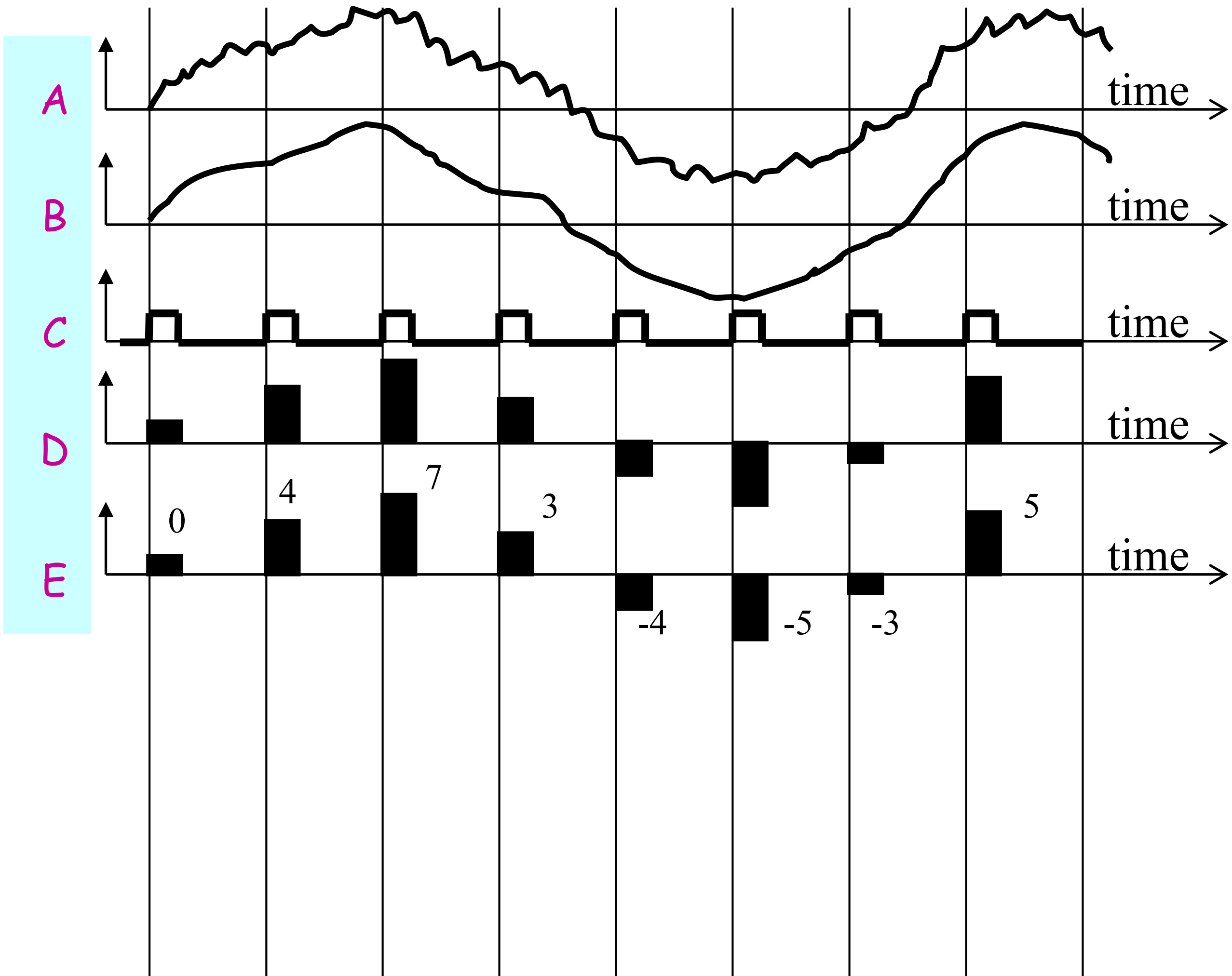


Audio signal (4)

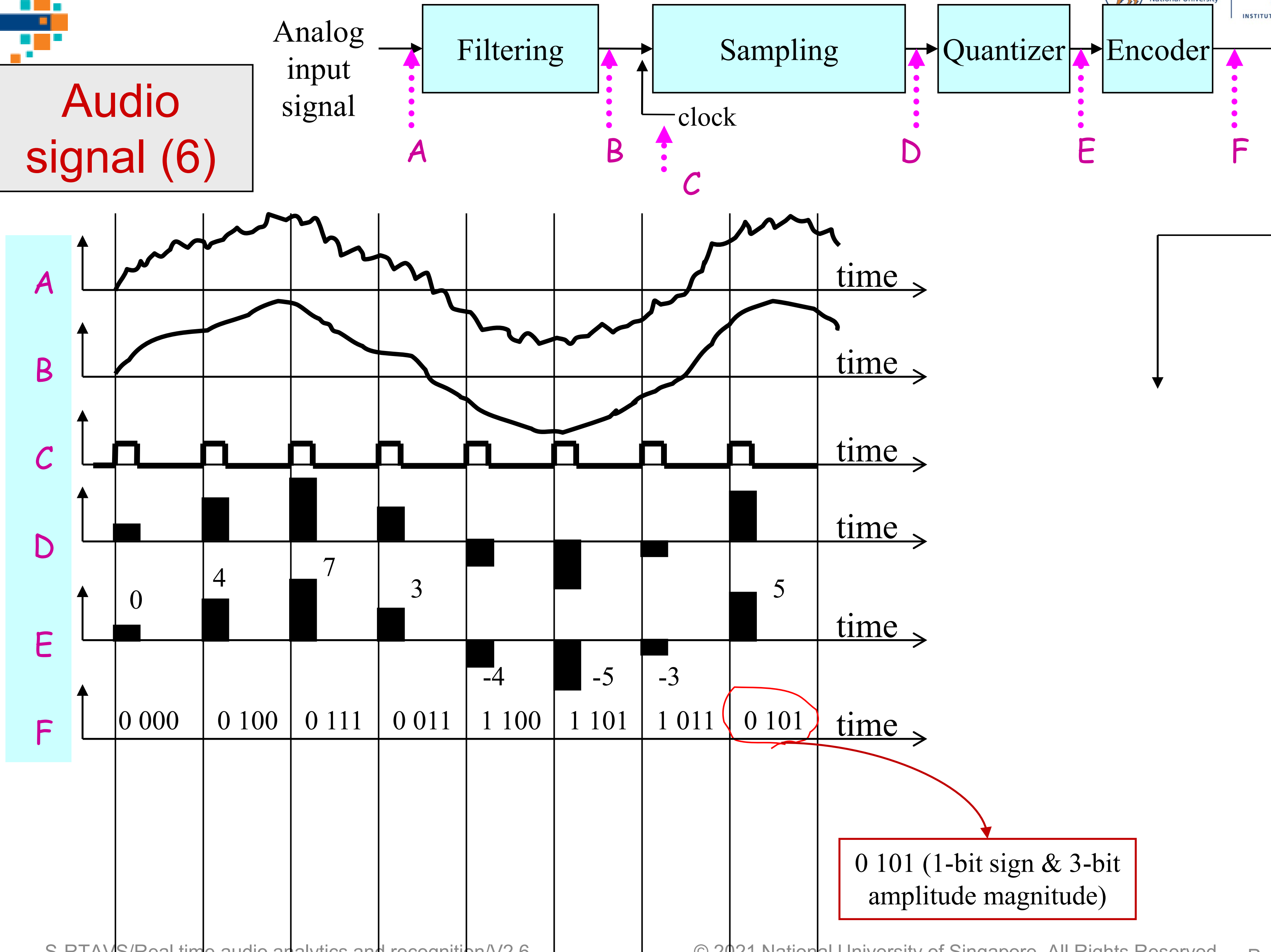


Audio signal (5)

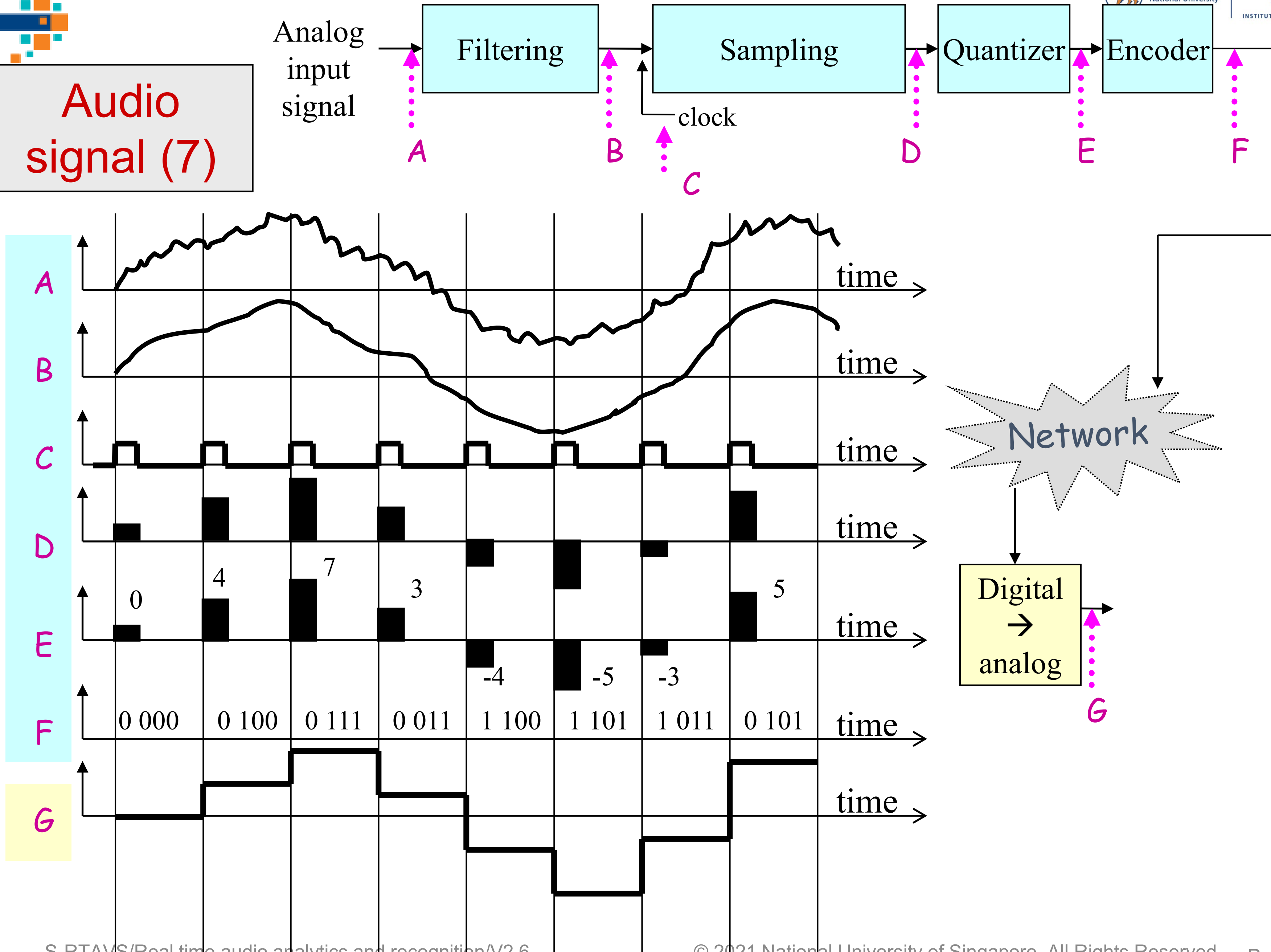




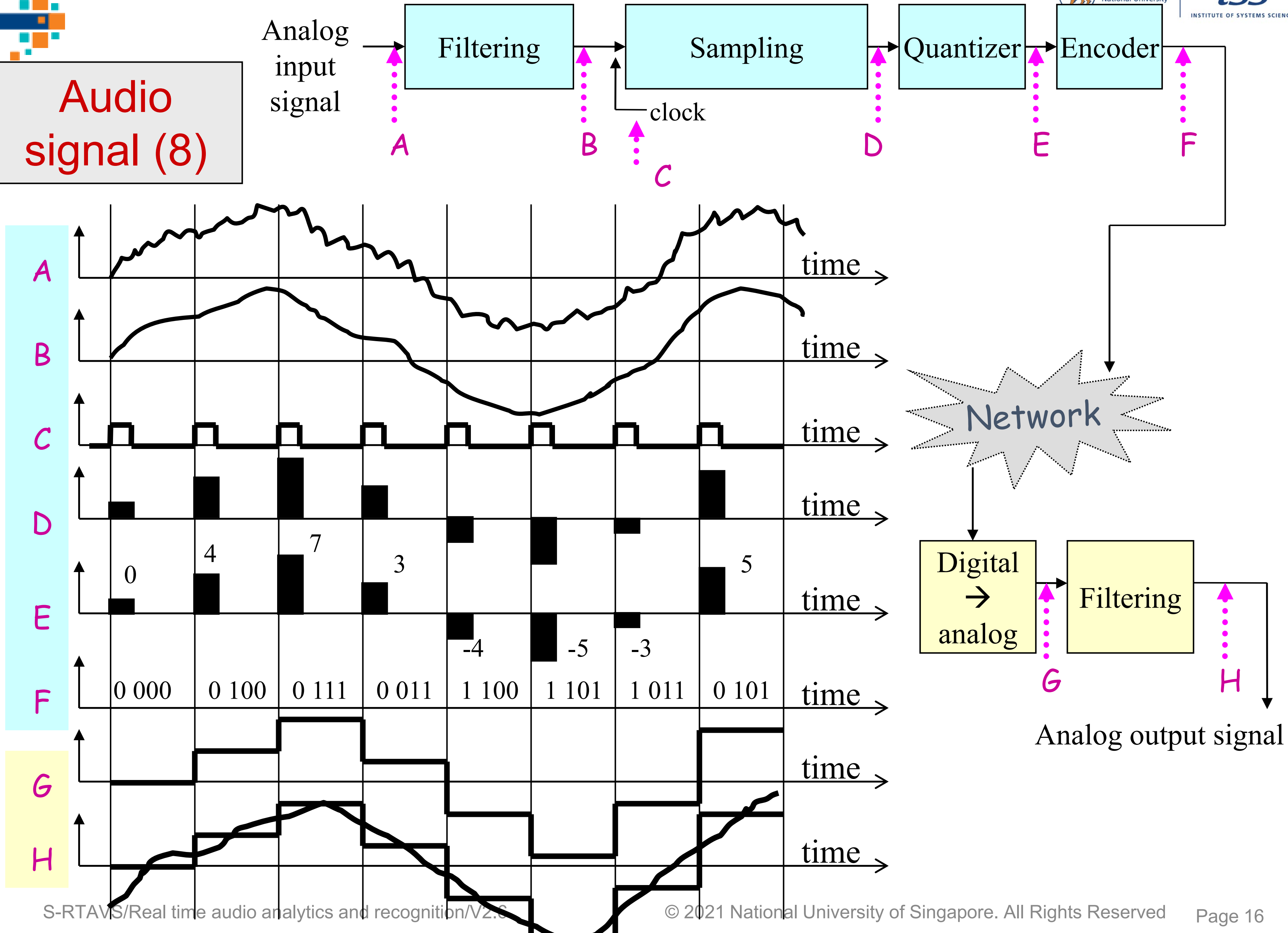
Audio signal (6)



Audio signal (7)



Audio signal (8)



Audio features: Time-domain features

Extract a segment (called *frame*) from the audio sequence $\{x(n)\}$, where $n = 1,2,\cdots,W$, that is covered by the window length W

- **Energy:** Energy of the signal $E = \frac{1}{W}\sum_{n=1}^{W}|x(n)|^2$
- **Zero Crossing Rate:** Rate of sign changes during the frame

$$Z = \frac{1}{2W}\sum_{n=1}^{W}\left|sign\big(x(n)\big) - sign\big(x(n-1)\big)\right|$$

$$sign\big(x(n)\big) = \begin{cases} 1 & x(n) \geq 0 \\ -1 & x(n) < 0 \end{cases}$$

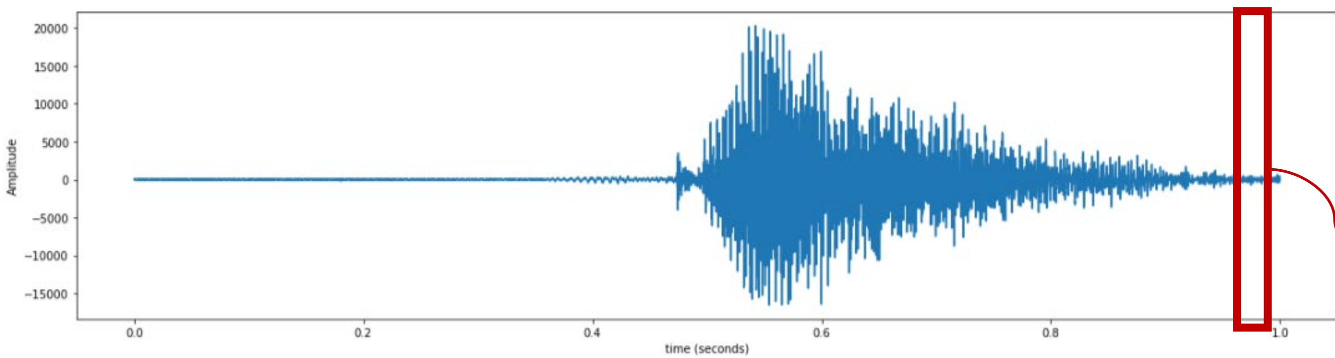
It measures smoothness of a signal is to calculate the number of zero-crossing within a segment of that signal. A voice signal oscillates slowly. A 100 Hz signal will cross zero 100 per second, whereas an unvoiced fricative can have 3000 zero crossing per second.

Reference: https://wiki.aalto.fi/display/ITSP/Zero-crossing+rate

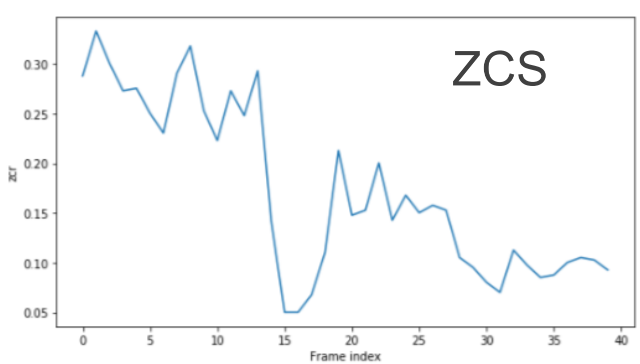
Audio features: Time-domain features

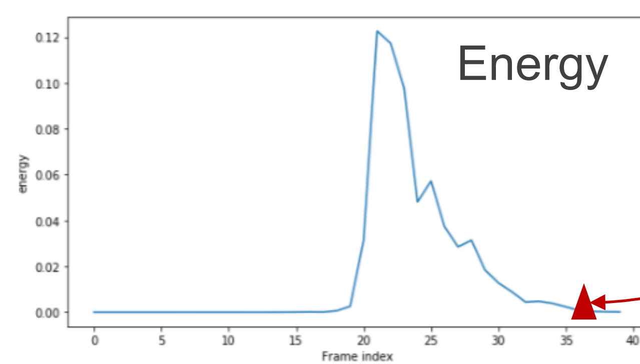
```
# Load pyAudioAnalysis library
from pyAudioAnalysis import ShortTermFeatures
from pyAudioAnalysis import audioBasicIO

# Load audio file, get audio data array s, sampling rate fs
fs, s = audioBasicIO.read_audio_file("data/go_male.wav")

# Perform feature extraction
[f, fn] = ShortTermFeatures.feature_extraction(s, fs, int(fs * 0.025), int(fs * 0.025))
```

```
def stFeature_extraction(signal, sampling_rate, window, step):
    """
    This function implements the shor-term windowing process.
    For each short-term window a set of features is extracted.
    This results to a sequence of feature vectors, stored in a np matrix.
    ARGUMENTS
        signal:        the input signal samples
        sampling_rate: the sampling freq (in Hz)
        window:        the short-term window size (in samples)
        step:          the short-term window step (in samples)
```



- Apply sliding window (size and step)
- Perform feature extraction





Recap: Fourier analysis

Signal (Fourier domain)

Apply all signal values

Signal (time domain)

Basis (sinusoid functions)

$$\text{Forward: } F(u) = \sum_{x=0}^{N-1} f(x) e^{\frac{-i2\pi ux}{N}}, \text{where } u = 0,1,\cdots,N-1$$

$$\text{Inverse: } f(x) = \frac{1}{N} \sum_{u=0}^{N-1} F(u) e^{\frac{i2\pi ux}{N}}, \text{where } x = 0,1,\cdots,N-1$$

Note: $e^{ix} = \cos x + i \sin x$; $e^{i\pi} = \cos \pi + i \sin \pi = -1$, Reference: https://en.wikipedia.org/wiki/Euler%27s_identity

Example

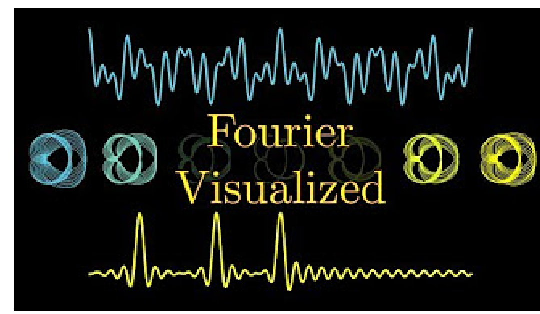
- Signal: $f(x) = [2, 3, 4, 4]$
- Fourier coefficients: $\mathrm{F}(u) = [13, (-2+i), -1, (-2-i)]$, where i is the imaginary unit

$$F(0) = \sum_{x=0}^{3} f(x) e^{\frac{-i2\pi 0x}{4}} = 2+3+4+4 = 13$$

$$F(1) = \sum_{x=0}^{3} f(x) e^{\frac{-i2\pi x}{4}} = 2e^{0} + 3e^{-i\pi/2} + 4e^{-i\pi} + 4e^{-i3\pi/2} = -2+i$$

$$F(2) = \sum_{x=0}^{3} f(x) e^{\frac{-i4\pi x}{4}} = 2e^{0} + 3e^{-i\pi} + 4e^{-i2\pi} + 4e^{-i3\pi} = -1$$

$$F(3) = \sum_{x=0}^{3} f(x) e^{\frac{-i6\pi x}{4}} = 2e^{0} + 3e^{-i3\pi/2} + 4e^{-i3\pi} + 4e^{-i9\pi/2} = -2-i$$



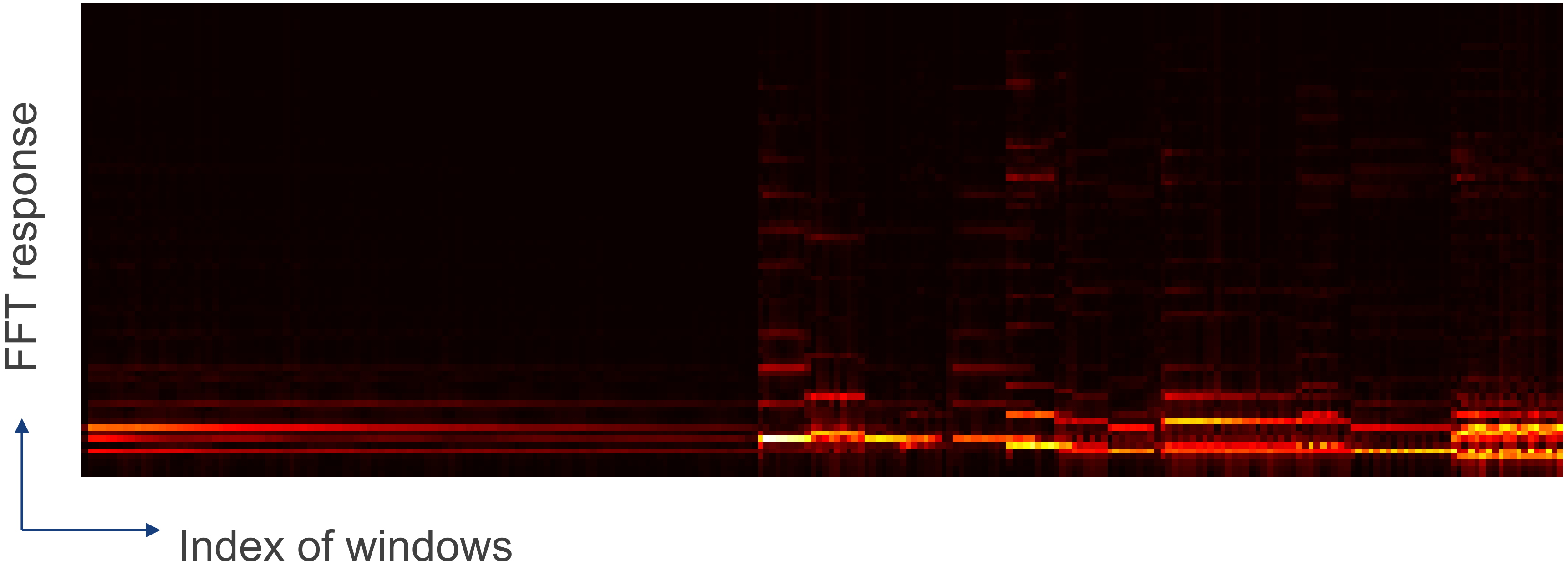
What is the Fourier Transform? A visual introduction
A 20-minute video tutorial on https://www.youtube.com/watch?v=spUNpyF58BY
4.4 millions views since Janurary 2018

Frequency representation: Spectrogram

Spectrogram: Time-Frequency 2D representation

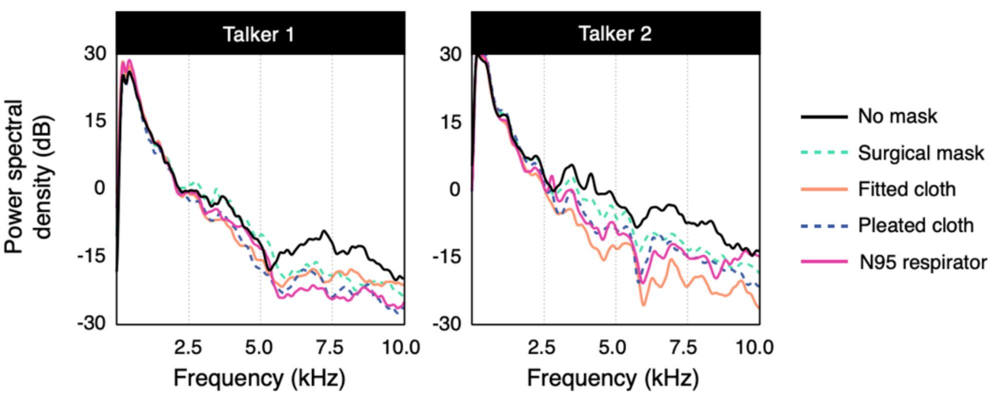
- Step 1, Windowing: Signal broken into (non)overlapping short-term windows
- Step 2, FFT: Fast Implementation of the *Discrete Fourier Transform* (DFT)

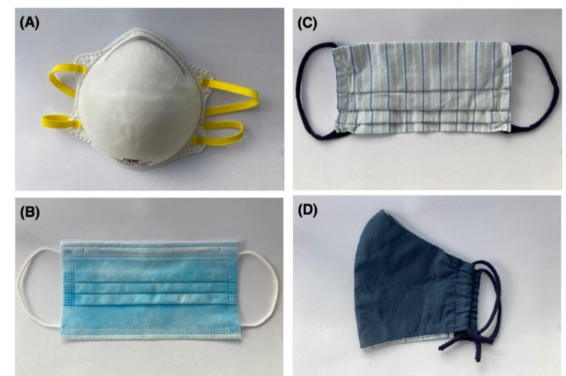
Online demo: https://lecture-demo.ira.uka.de/spectrogram-demo



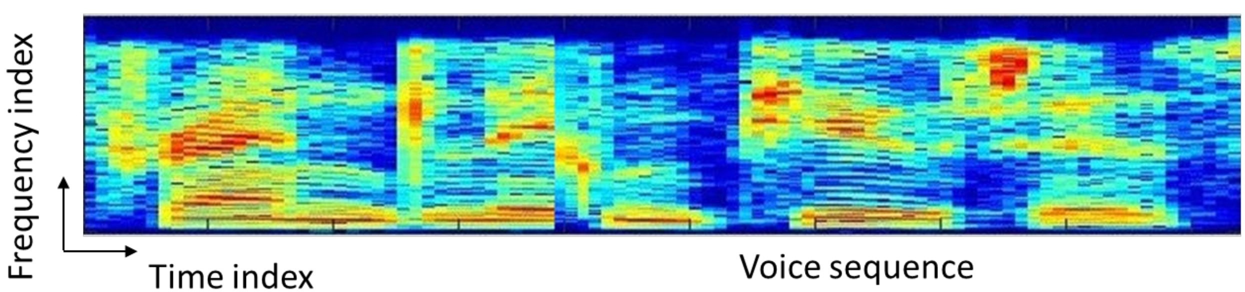
Frequency representation: Spectrogram

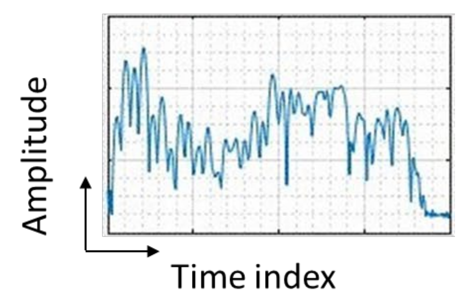
Face masks may cause a reduction in speech intelligibility. According to a recent study, there are small differences compared to the no-mask condition at lower frequencies and significant decreases at higher frequencies.





Question: Given a spectrogram shown below, how the spectrogram will change if the talker wears a mask?



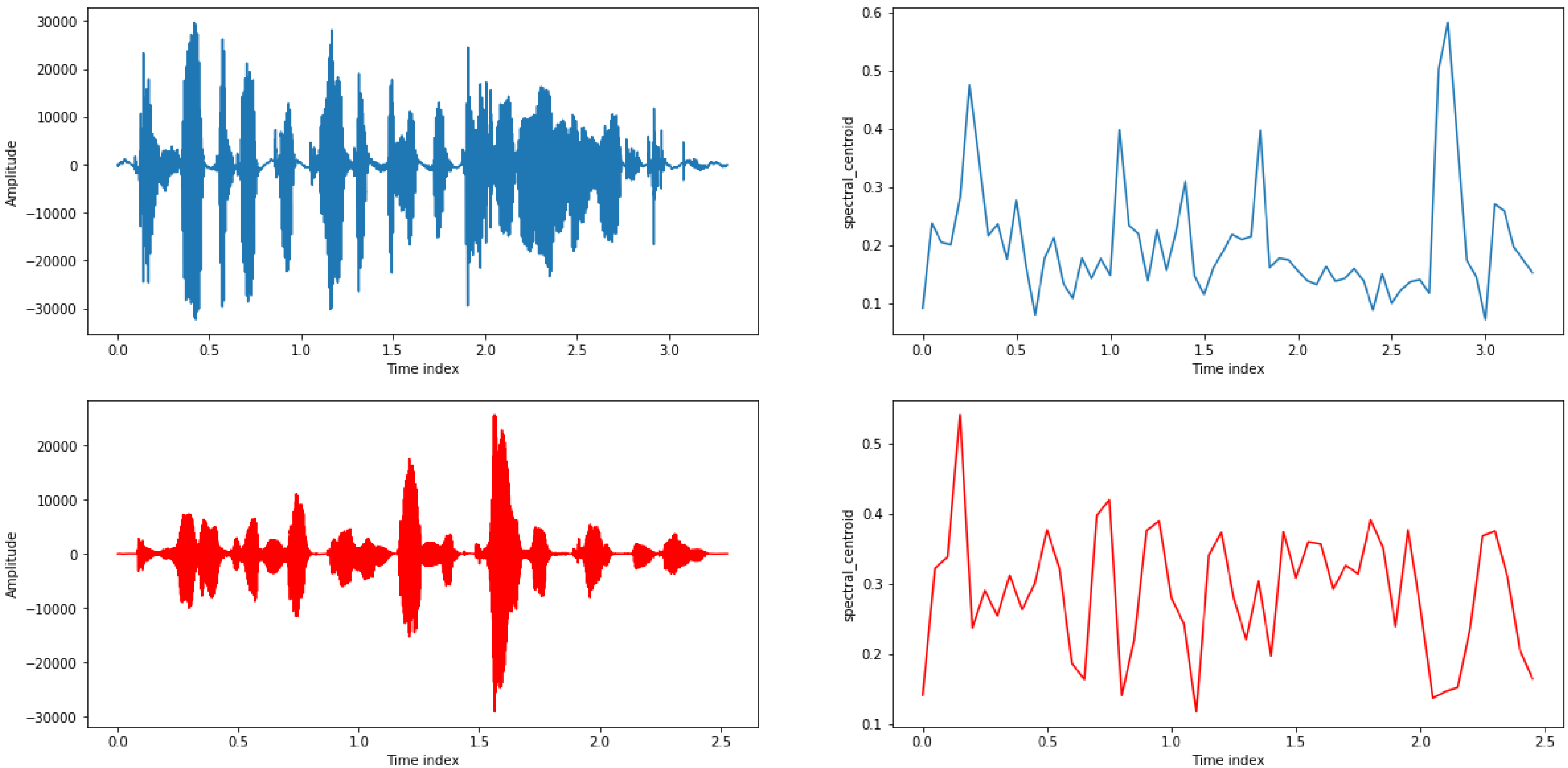


Reference: Effects of face masks on speech recognition in multi-talker babble noise, PLOS ONE, 2021, https://journals.plos.org/plosone/article/comments?id=10.1371/journal.pone.0246842

Audio features: Frequency-domain features

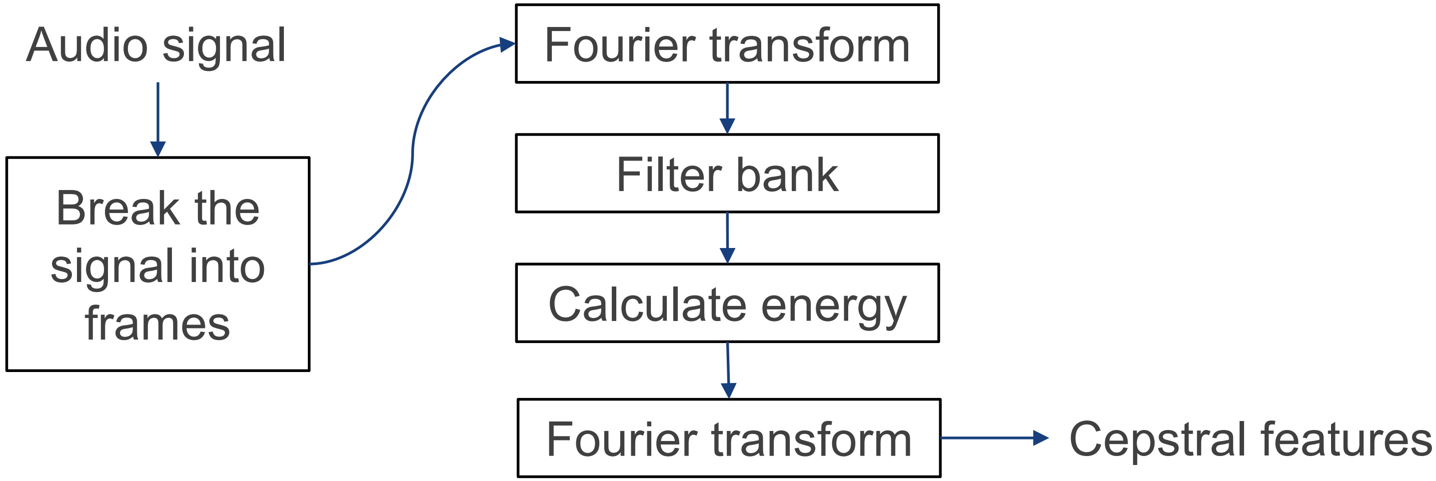
Spectral centroid: Center of gravity of the spectrum. It is calculated as the weighted mean of the frequencies present in the signal, $Centroid = \frac{\sum_{u=0}^{N-1} f(u)x(u)}{\sum_{u=0}^{N-1} x(u)}$, where $x(u)$ represents the weighting factor (frequency amplitude) of bin number u, and $f(u)$ represents the center frequency of that bin.

Example: Comparison between male and female audio sequence.



Audio features: Cepstral domain

- *Mel-Frequency Cepstral Coefficients* (MFCC)
  - Compute Fourier transform
  - Apply Mel-scale filter bank (see next slide)
  - Compute the power of the output of each filter
  - Compute MFCCs as the Fourier transform coefficients of the mel-scaled log-power spectrum
- Usually select the first 13 MFCCs (considered to carry sufficient discriminative information especially for speech classification tasks)



Reference: https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd

Frequency representation: Mel scale

Mel scale relates perceived frequency, or pitch, of a pure tone to its actual measured frequency. Humans are much better at discerning small changes in pitch at low frequencies than they are at high frequencies. Incorporating this scale makes our features match more closely what humans hear.

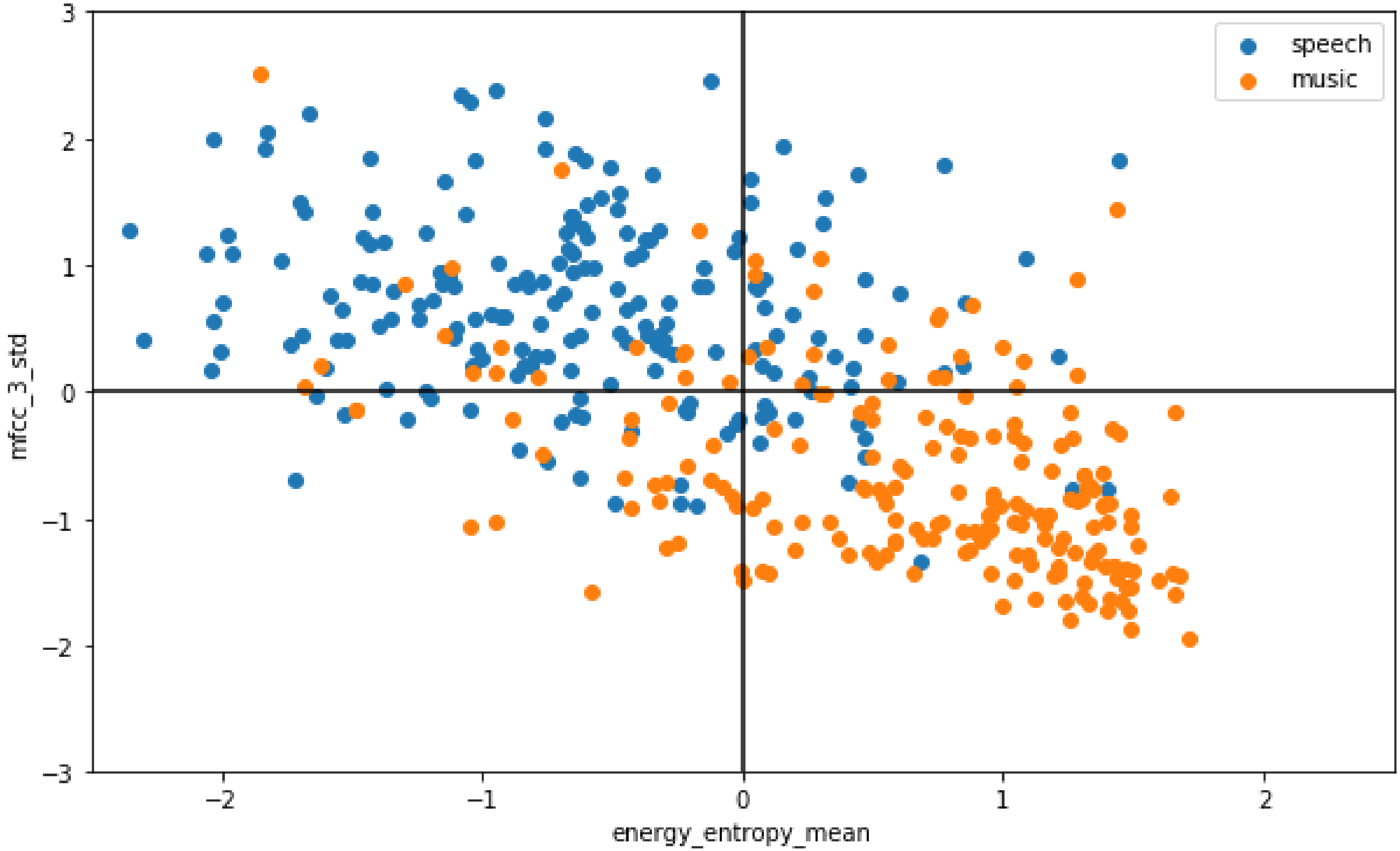
Thresholds(change in frequency)				
Freq	2 Hz	5 Hz	10 Hz	20 Hz
250 Hz	0.7	1	1	1
500 Hz	0.4	0.8	0.9	1
1000 Hz	0.6	0.8	1	0.9
2000 Hz	0.5	0.4	0.9	1
3000 Hz	0.5	0.5	0.6	1

```
# Generate two sounds and record whether it can be differentiated by human or not
def play_sound(freq, duration, fs):
    t = np.arange(0, duration, 1.0/fs);  x = 0.5*np.cos(2 * np.pi * t * freq)
    wavfile.write("temp.wav", fs, x); os.system("start temp.wav")

freqs = [500, 1000, 2000]; thres = [5, 10, 20]; n_exp = 4; fs = 16000

answers = [[] for i in range(len(freqs))]
for i_f, f in enumerate(freqs):
    for t in thres:
        answers[i_f].append(0)
        for i in range(n_exp):
            sequel = randint(1, 2)
            if sequel == 2:
                play_sound(f, 0.5, fs); time.sleep(0.5); play_sound(f+t, 0.5, fs)
            else:
                play_sound(f+t, 0.5, fs); time.sleep(0.5); play_sound(f, 0.5, fs)
            ans = int(input('Frequency=%d Hz, Threshold=%d Hz, Experiment=%d/%d, 
Which was higher frequency (1/2):' % (f, t, i+1, n_exp)))
            if ans == sequel: answers[i_f][-1] += 1
```

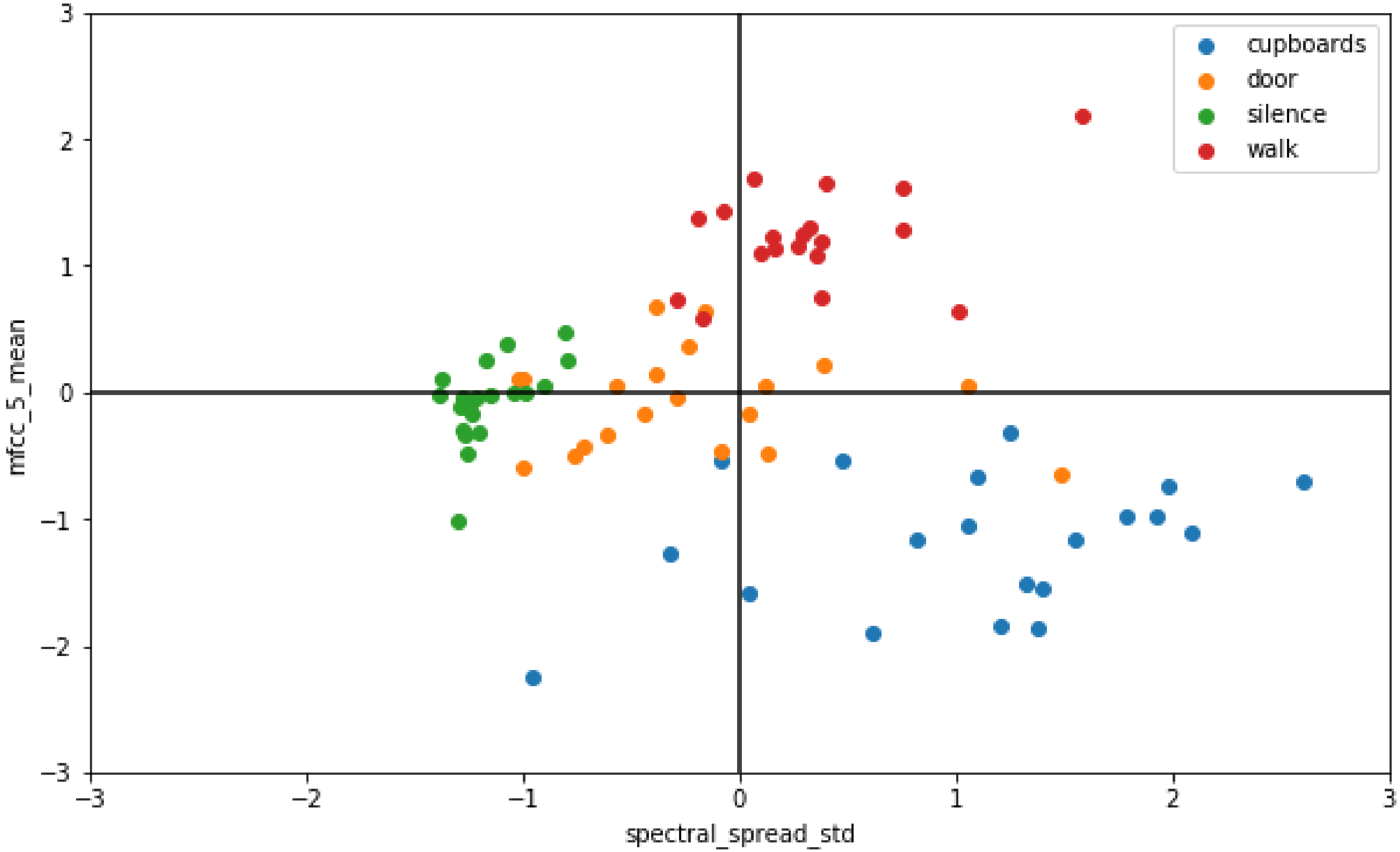

Reference

- Speech Processing for Machine Learning, https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html
- Mel Frequency Cepstral Coefficient (MFCC) tutorial, http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/

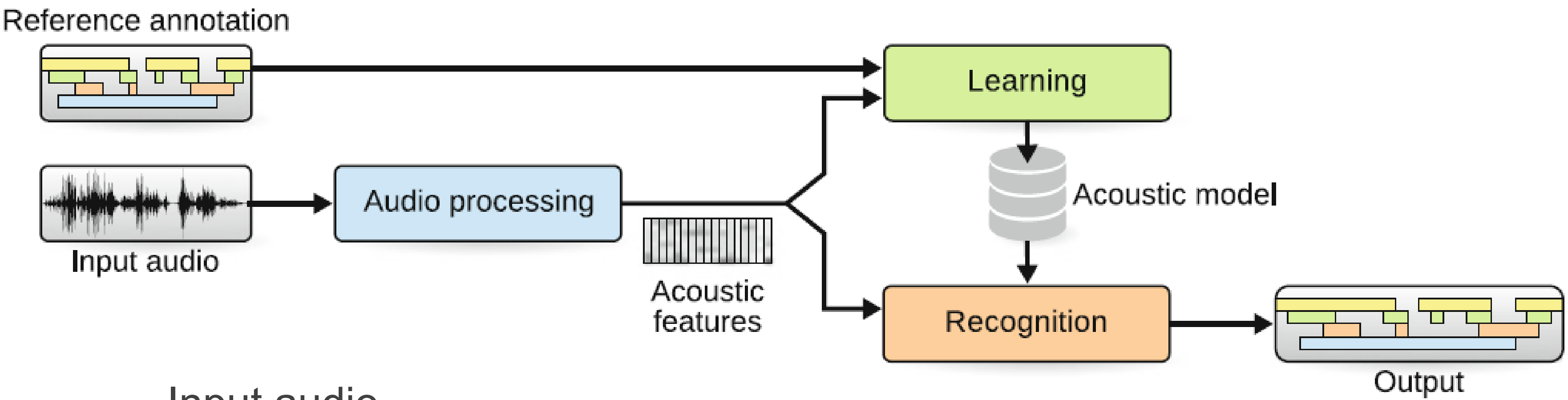
Use case: Speech vs music



Use case: Audio events



Overview of audio analytics system



Input audio

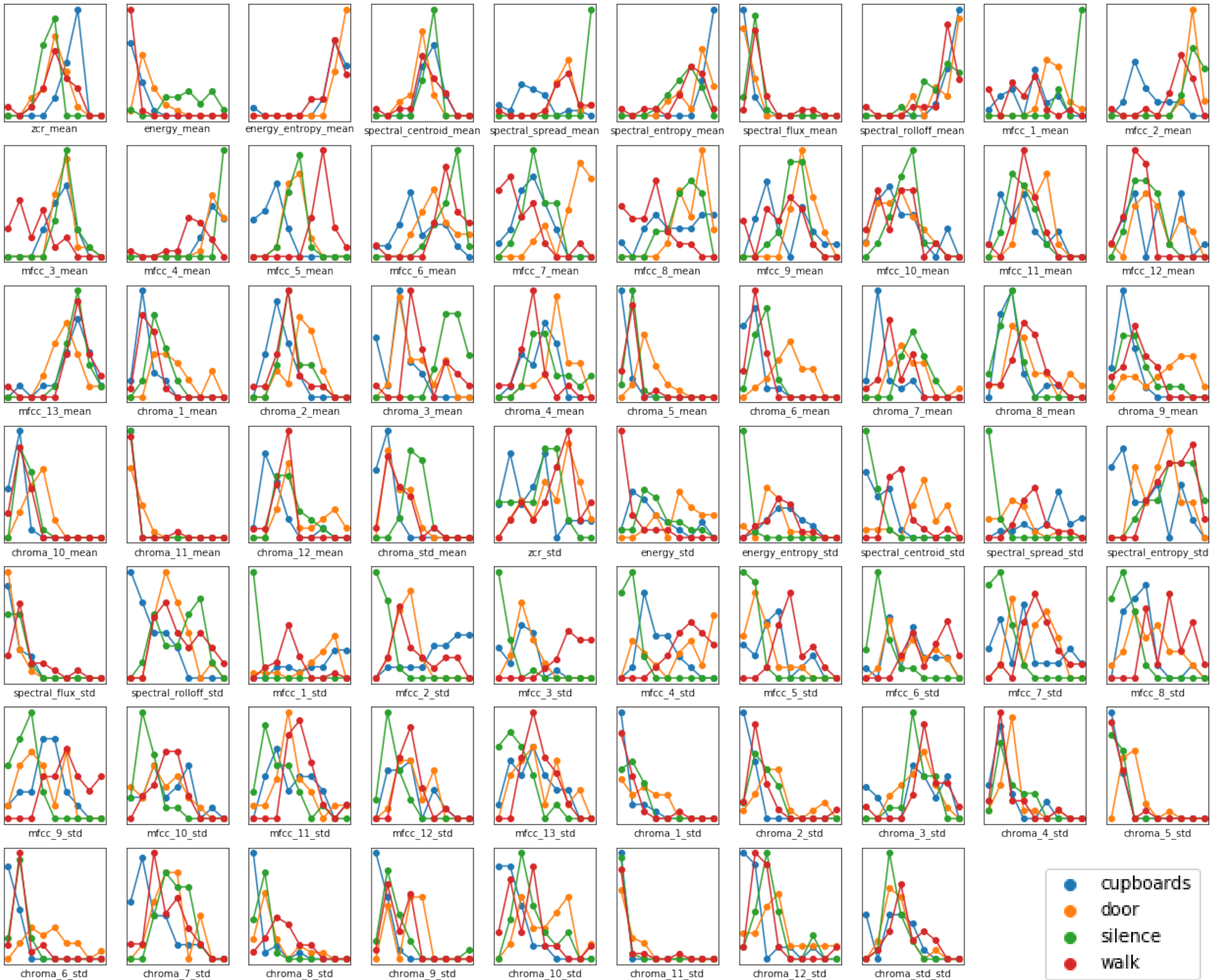
Extract segment (frame) → Time-domain features

Transformation (e.g., Fourier transform) → Frequency-domain features → Further stack into Spectrogram

Further filtering and transformation → Cepstral features

Reference: T. Virtanen, M. Plumbley, and D. Ellis, "Computational analysis of sound scenes and events," https://cassebook.github.io/

Use case: Audio events



Sense making from audio

- Acoustic Scene Classification
  - Classify a recording into one of the predefined acoustic scene classes.
- Unsupervised Detection of Anomalous Sounds for Machine Condition Monitoring
  - Identify whether the sound emitted from a target machine is anomalous.
- Sound Event Localization and Detection
  - Given a multichannel audio input, the goal of a sound event localization and detection (SELD) method is to output all instances of the sound labels in the recording, its respective onset-offset times, and spatial locations in azimuth and elevation angles
- Sound Event Detection and Separation in Domestic Environments
  - Provide not only the event class but also the event time boundaries given that multiple events can be present in an audio recording.
- Urban Sound Tagging with Spatiotemporal Context
  - How spatiotemporal metadata can aid in the prediction of urban sound tags for recordings from an urban acoustic sensor network.
- Automated Audio Captioning
  - Generate the textual description (i.e. the caption) of the audio signal input.

Reference: Detection and Classification of Acoustic Scenes and Events, http://dcase.community/

Typical benchmark datasets

	Dataset name	Type	Classes	Examples	Size (min)
Sound scenes	Dares G1	rec	28	123	123
	DCASE 2013 Scenes	rec	10	100	50
	LITIS Rouen	rec	19	3026	1513
	TUT Sound Scenes 2016	rec	15	1170	585
	YouTube-8M	col	4716	>7M	>27M
Environmental sounds	ESC-10	col	10	400	33
	ESC-50	col	50	2000	166
	NYU Urban Sound8K	col	10	8732	525
	CHIME-Home	rec	7	6137	409
	Freefield1010	col	7	7690	1282
	CICESE Sound Events	col	20	1367	92
	AudioSet	col	632	>2M	>340k
Sound events	Dares G1	rec	761	3214	123
	DCASE 2013 Office Live	rec	16	320	19
	DCASE 2013 Office Synthetic	syn	16	320	19
	TUT Sound Events 2016	rec	18	954	78
	TUT Sound Events 2017	rec	6	729	92
	NYU Urban Sound	col	10	3075	1620
	TU Dortmund Multichannel	rec	15	1170	585

Rec: Recorded
Col: Collected from available repositories
Syn: Produced synthetically

Reference: T. Virtanen, M. Plumbley, and D. Ellis, “Computational analysis of sound scenes and events,” https://cassebook.github.io/

What we have learnt

- Audio signal representation
- Various audio feature extraction methods
  - Time-domain features
  - Frequency-domain features
  - Cepstral features
- Audio classification for sound activities

Thank you!

Dr TIAN Jing
Email: tianjing@nus.edu.sg