



Music Data Mining

Music and Audio Content Analysis Using LibROSA

Topics

- Audio representations
 - Waveform, DFT, spectrogram, MFCC, CQT, and chromagram
 - Amplitude, period, frequency, frame, logarithmic scale, and abstraction
- Important musical concepts
 - Pitch, loudness, timbre, harmonics, F0, overtones, and chords
- Feature aggregation
 - Summary of audio representations
- Librosa and numpy functions for audio representations



Time Domain Representations and Features

Visualize an Audio File



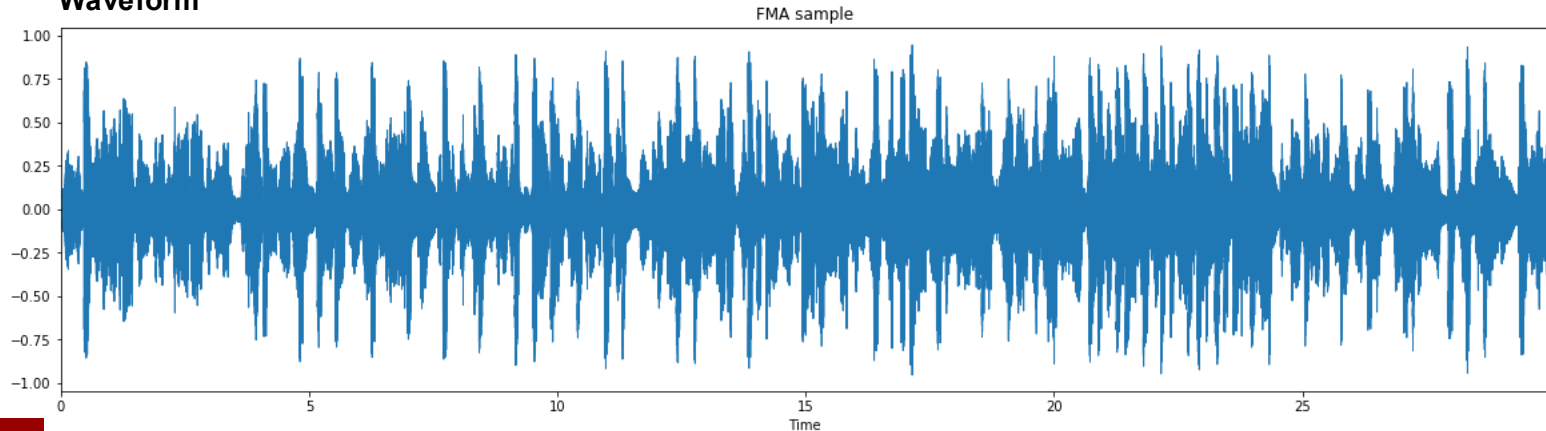
Data Excerpt (200 points)

```
[ 0. -0. 0. -0. 0. -0. 0. -0. 0. -0. -0.00001 -0. 0. -0. 0. -0. 0. -0. 0. -0.00001 0.
-0. 0. -0.00001 -0. 0. -0. -0.00001 0. -0. 0. -0. -0.00001 0.00001 0.00001 -0. 0. -0.00001 -0.00001 0. -0. -0.00001 -
0.00001 -0.00001 -0.00003 -0.00002 -0. -0.00001 -0.00003 -0.00002 -0.00001 -0.00001 0.00001 0. -0. -0.00001 -0.00002
0. -0.00002 -0.00002 -0.00002 -0.00002 -0.00002 -0.00002 -0.00003 -0.00001 0.00002 0.00001 -0.00003 -0.00008 -0.00011
-0.00011 -0.00008 -0.00009 -0.00016 -0.00025 -0.00026 -0.00018 -0.00012 -0.00013 -0.00021 -0.00025 -0.00022 -0.00012
0. 0.00001 -0.00008 -0.00015 -0.00013 -0.00004 -0.00003 0.00001 ]
```

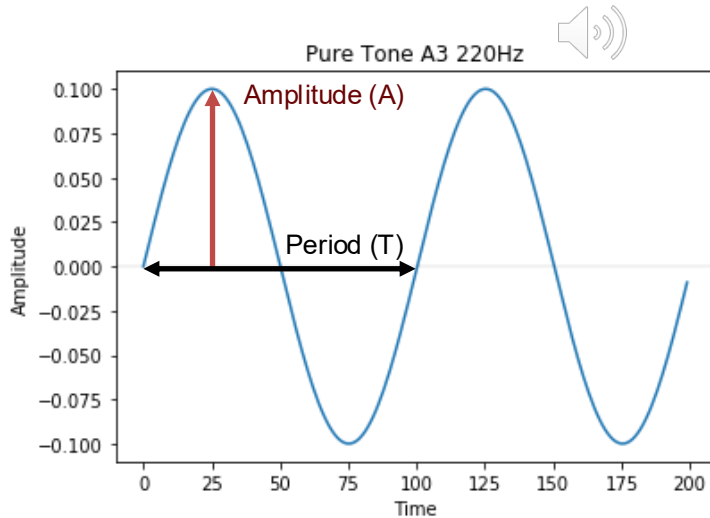
Descriptive Statistics

nobs=660984, minmax=(-0.9512512, 0.94407535), mean=0.00031254382, variance=0.027324814

Waveform



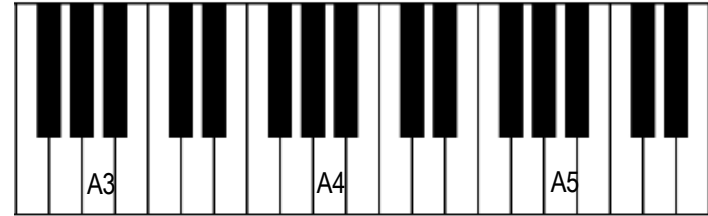
Waveform



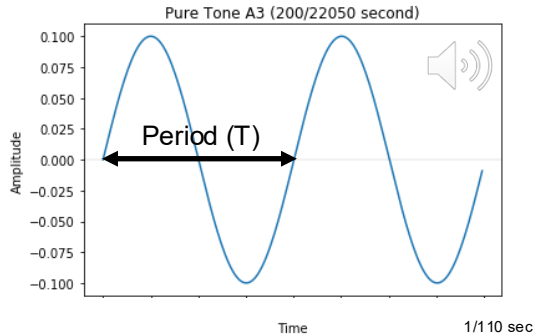
- Sinusoid: A sine wave (or a cosine wave)
- Waveform: A graph that displays amplitude over time
- Amplitude (A): The amplitude of a given wave is the value of the wave at that point.
- Periodic signal: repeats a pattern over a identical period
- Period (T): The amount of time it takes to complete one cycle of the periodic wave



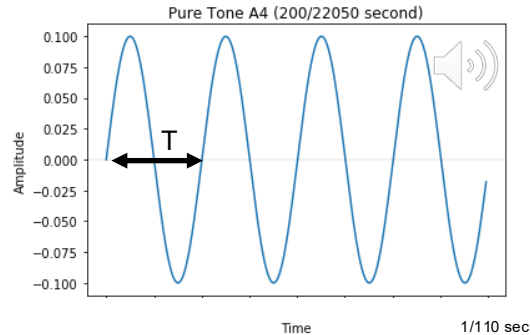
Frequency



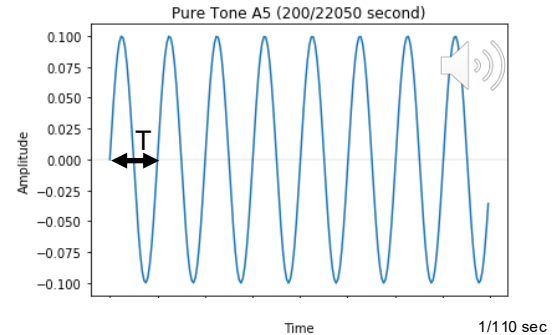
- Frequency (f): How many cycles are completed in one second (Hertz, Hz)
 - Period's reciprocal ($1/T$), related to the musical pitch



$$T = 1/220 \text{ sec}$$
$$f = 1/T = 220 \text{ Hz}$$



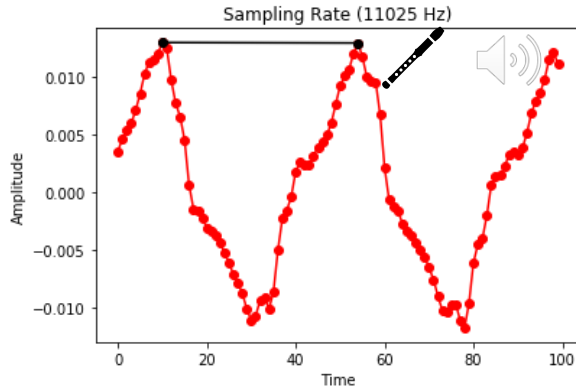
$$T = 1/440 \text{ sec}$$
$$f = 1/T = 440 \text{ Hz}$$



$$T = 1/880 \text{ sec}$$
$$f = 1/T = 880 \text{ Hz}$$



Measuring the Frequency

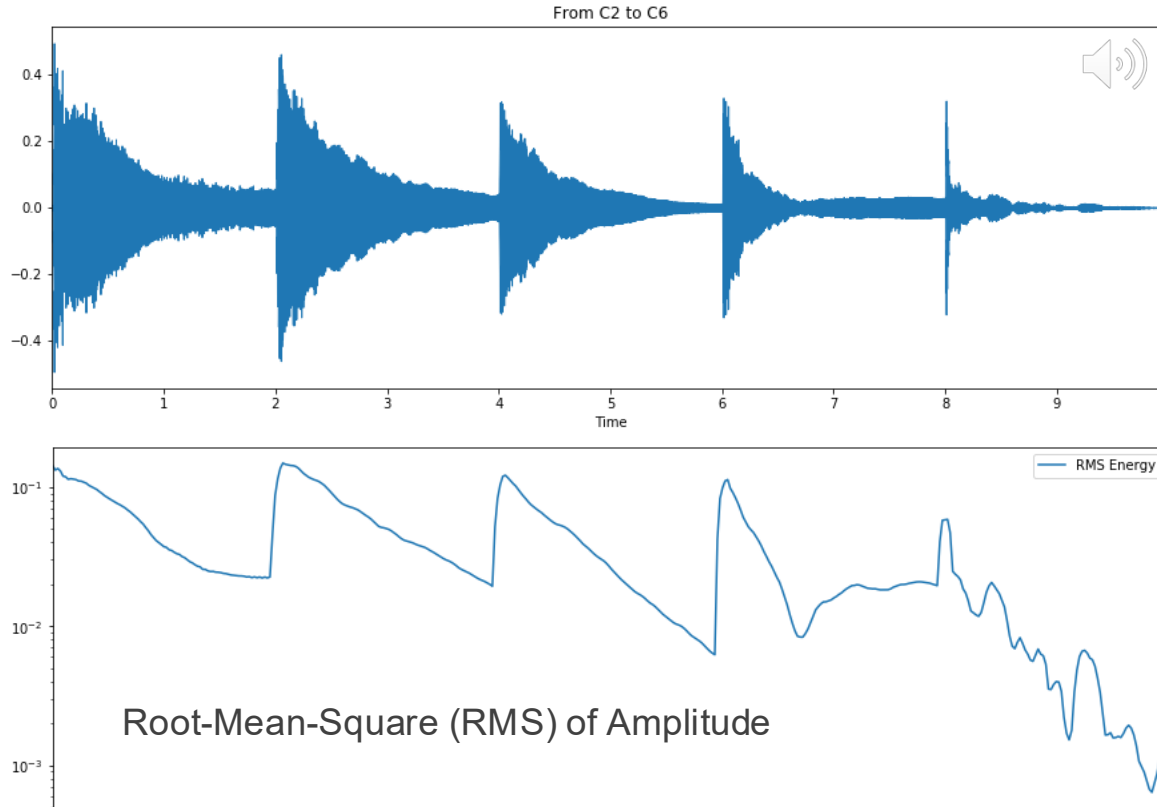


Note	Frequency (Hz)
A ₃	220.00
A [#] ₃ /B ^b ₃	233.08
B₃	246.94
C ₄	261.63
C [#] ₄ /D ^b ₄	277.18
D ₄	293.66

- Period (T): 44 points (1 point = 1/11025 second)
- $1/T = 1/44 \text{ points} = 11025/44 \text{ Hz} \approx 250 \text{ Hz}$ (about B3)



Loudness



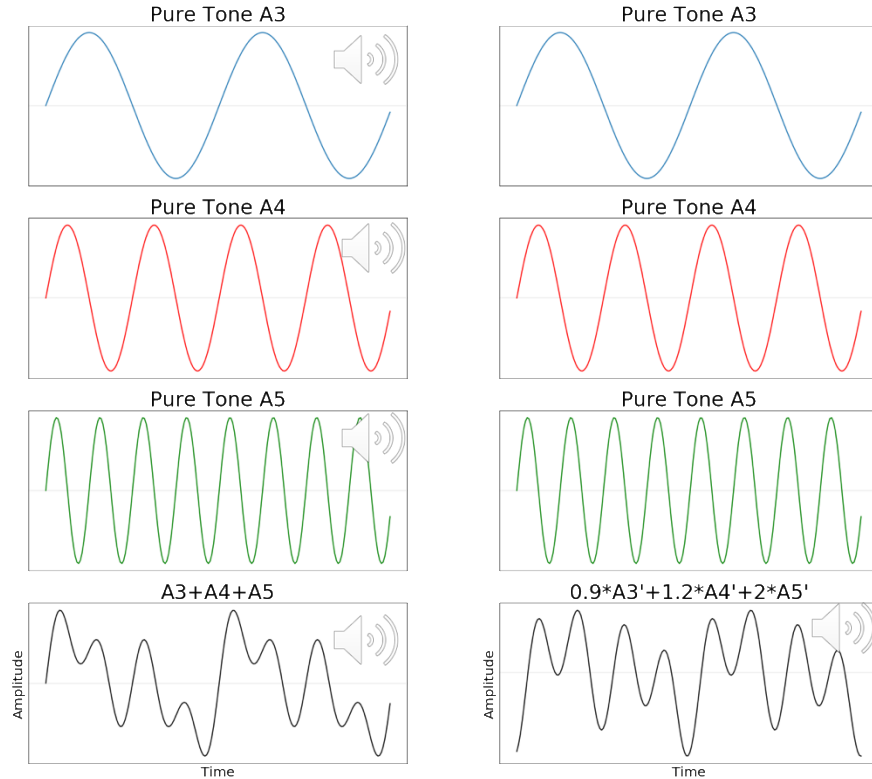
- Loudness is related to amplitude of the sound
- `librosa.feature.rms` computes root-mean-square (RMS) value for each frame
- RMS for each frame

$$x_{\text{RMS}} = \sqrt{\frac{1}{n} (x_1^2 + x_2^2 + \dots + x_n^2)}$$



Frequency Domain Representations and Features

Combination of Sinusoids

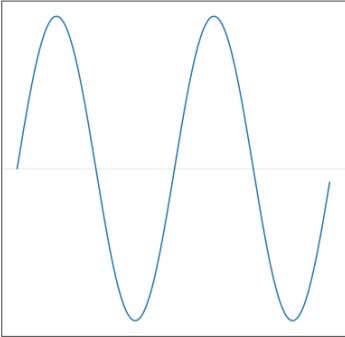


- Jean-Baptiste Joseph Fourier (1768-1830) discovered that **any periodic waveform can be expressed as an additive combination of sinusoids with varying amplitudes and phases**
 - Analysis: decompose a complex sound into a series of simple sinusoids
 - Synthesis: compose complex sounds by mixing in the right proportions a series of sinusoidal wave

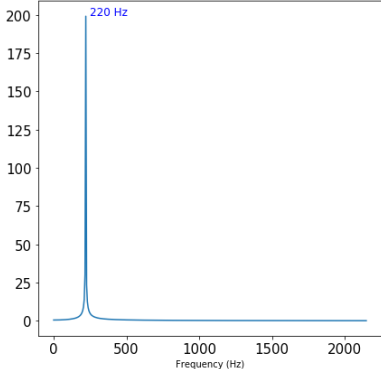


Discrete Fourier Transform (DFT)

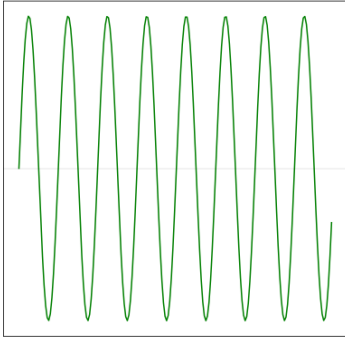
Pure Tone A3



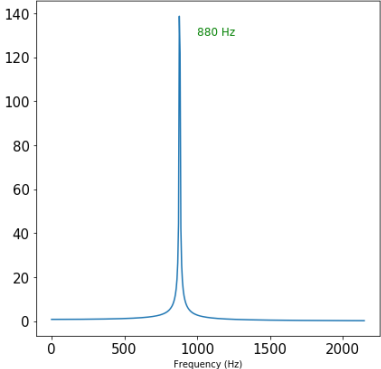
DFT of A3



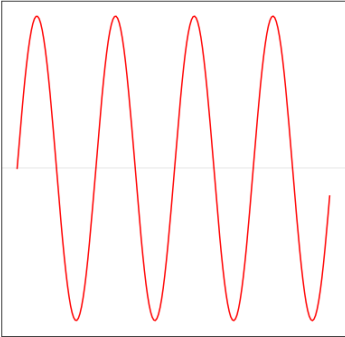
Pure Tone A5



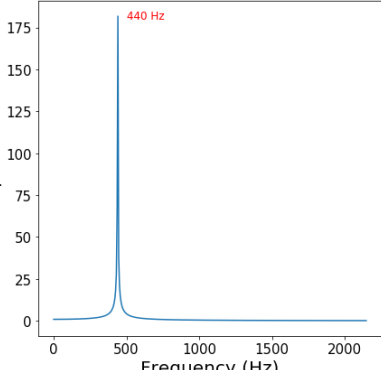
DFT of A5



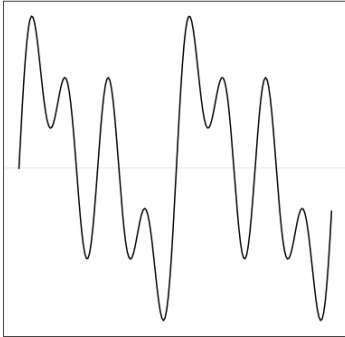
Pure Tone A4



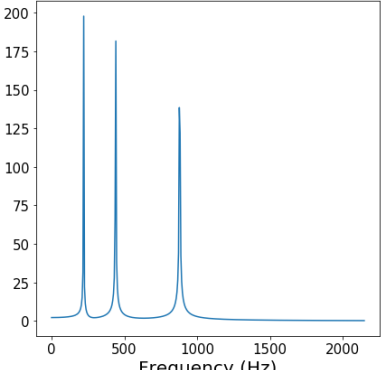
DFT of A4



A3+A4+A5



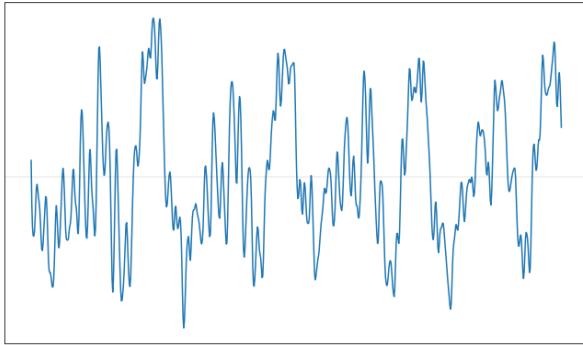
DFT of A3+A4+A5



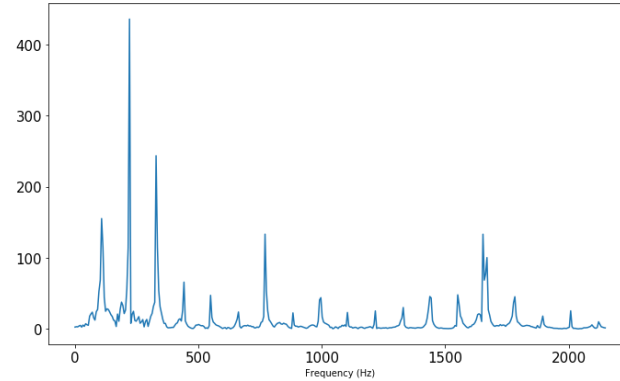
DFTs of Instruments' Recordings



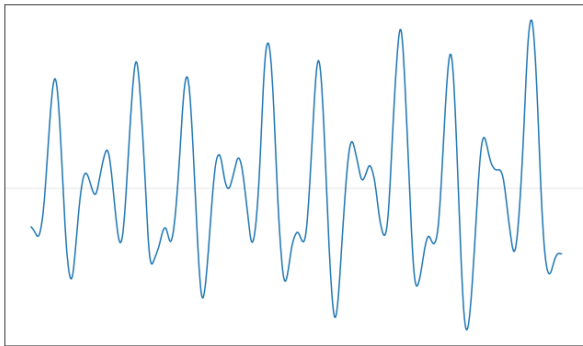
Piano A2



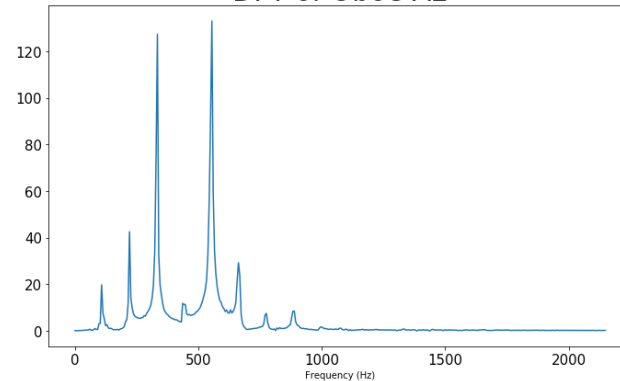
DFT of Piano A2



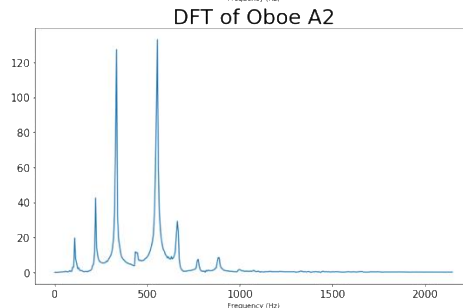
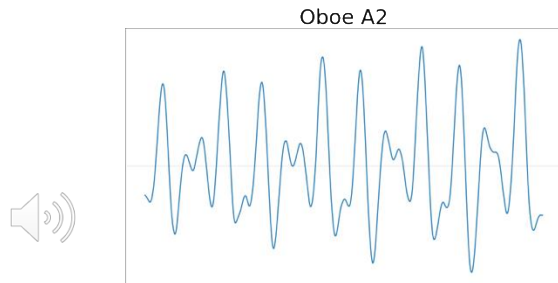
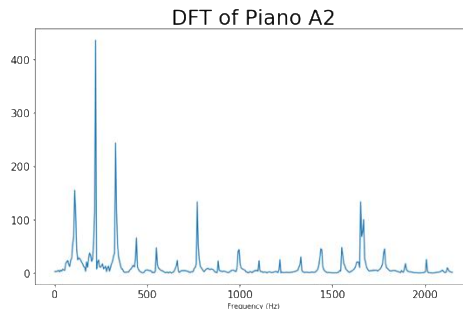
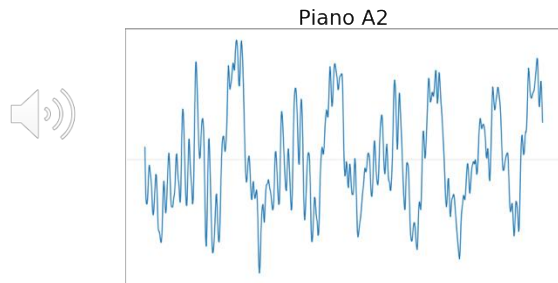
Oboe A2



DFT of Oboe A2



Timbre, Harmonics, F0, and Overtones

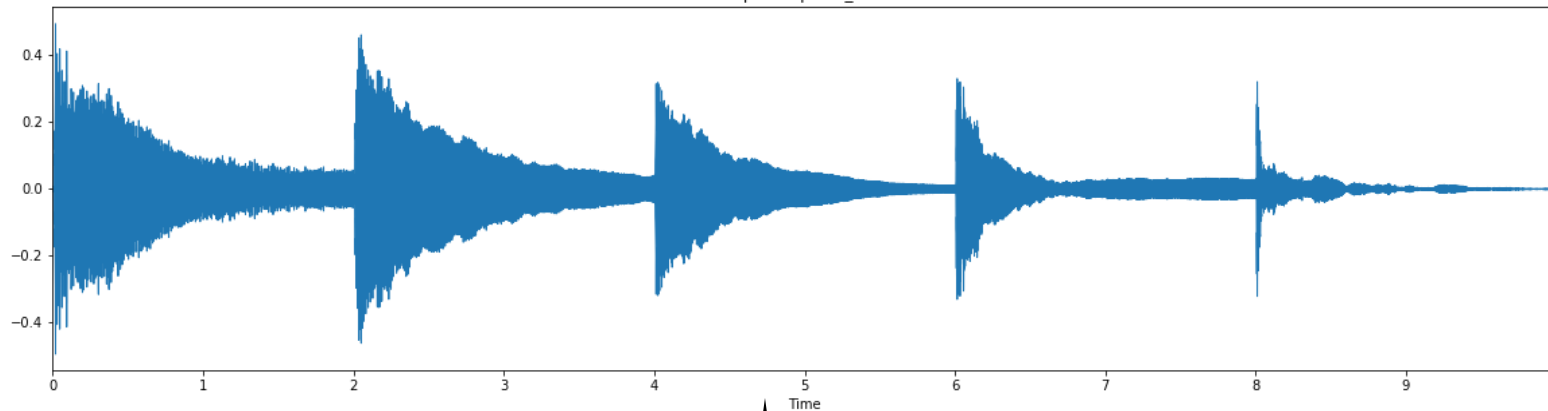


- Harmonics: Frequency peaks
 - 110, 220, 330, 440 Hz, ...
- F0 (Fundamental Frequency): The lowest frequency component among harmonics; the difference between adjacent harmonic components
 - 110 Hz
- Overtones: Harmonics except F0
 - 220, 330, 440 Hz, ...
 - At integer multiples of the F0
- Timbre: Determined by different distributions of harmonics

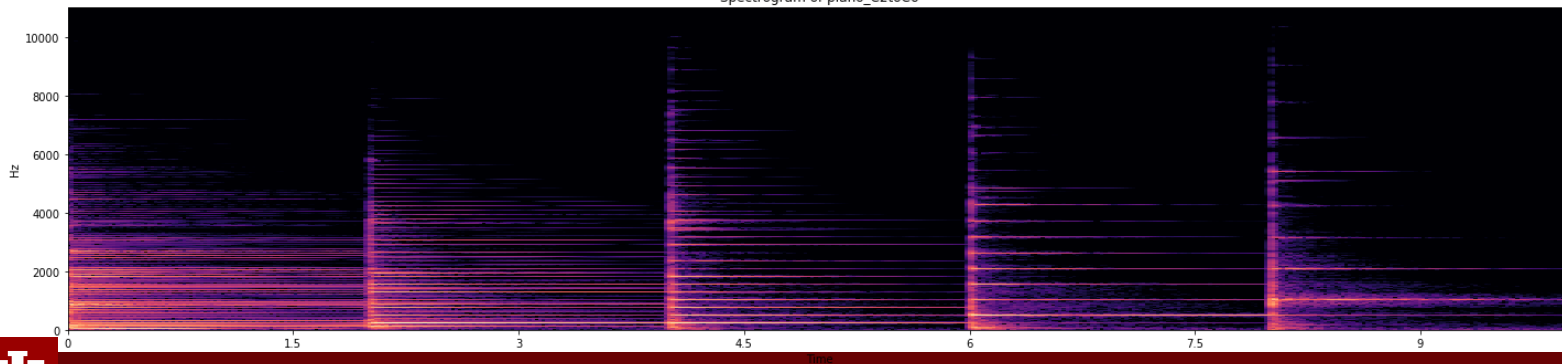
Spectrogram of Multiple Notes



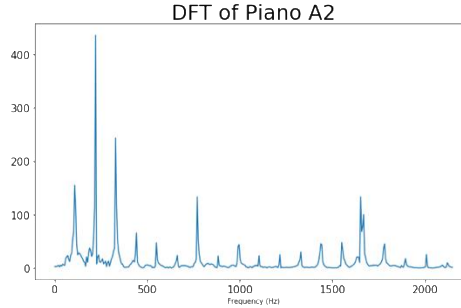
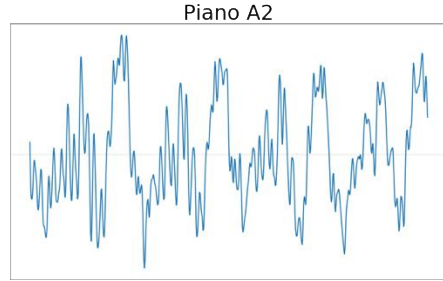
Waveplot of piano_C2toC6



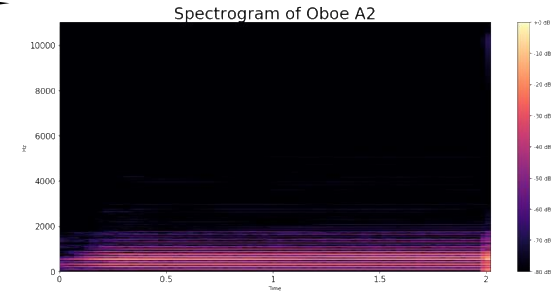
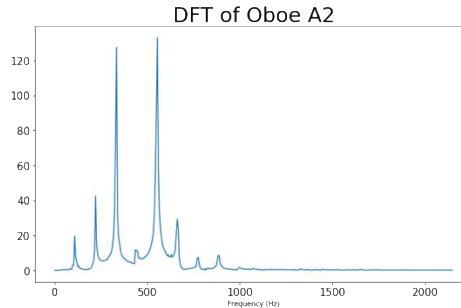
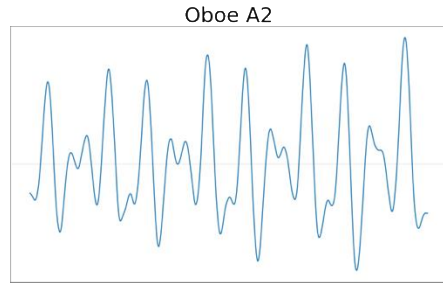
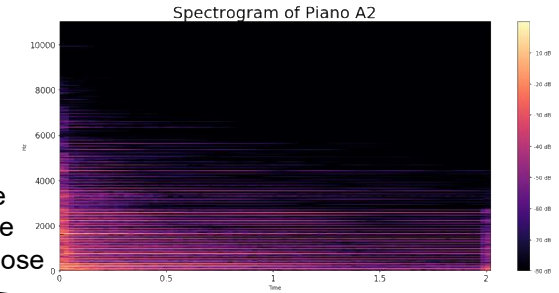
Spectrogram of piano_C2toC6



Waveform, DFT, and Spectrogram



Average
over time
and transpose



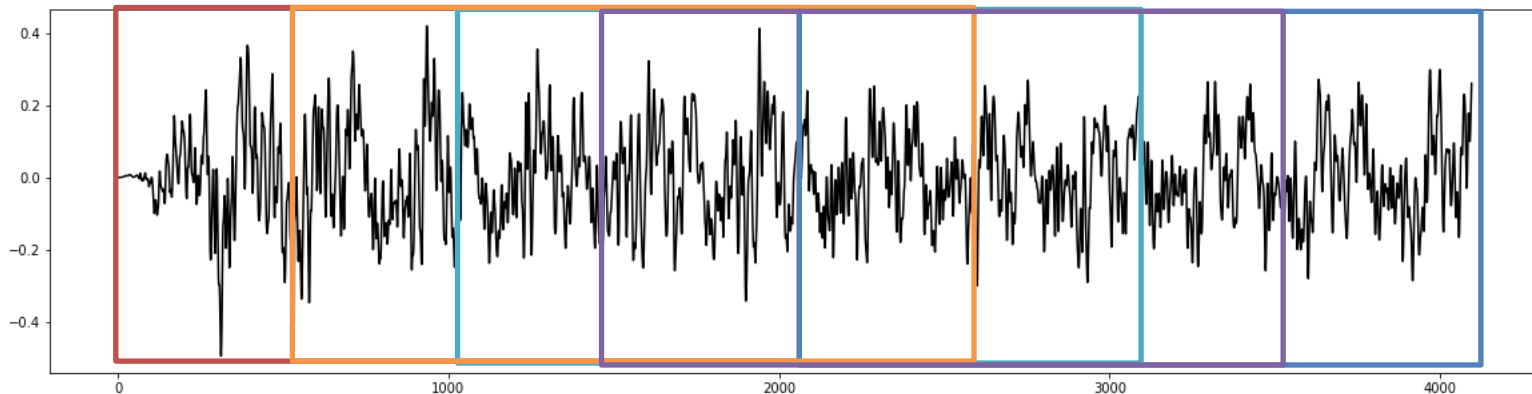
X: time
Y: amplitude

X: frequency
Y: intensity

X: time
Y: frequency
Z (Color) : energy



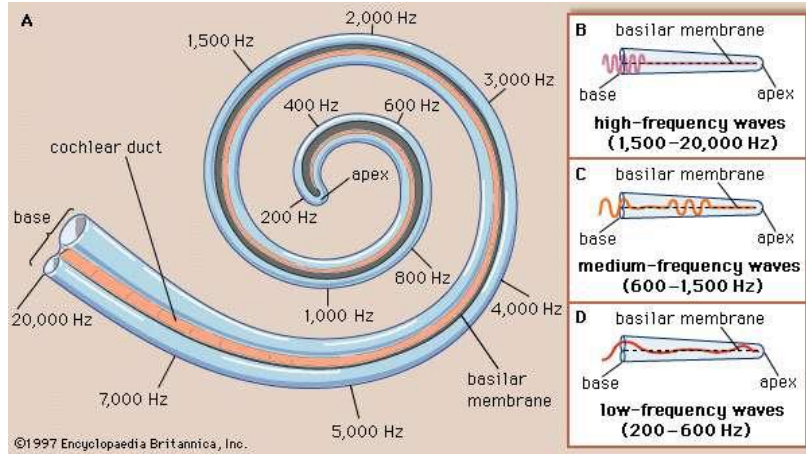
How to Calculate Spectrogram?



- Repeat the following steps to calculate spectrogram
 - Pick n_fft samples -> Apply an window -> Calculate DFT -> Move to the right by hop_length
- Important parameters
 - n_fft : length of windowed signal (default: 2048)
 - hop_length : number of audio samples between adjacent STFT columns (default: 512)

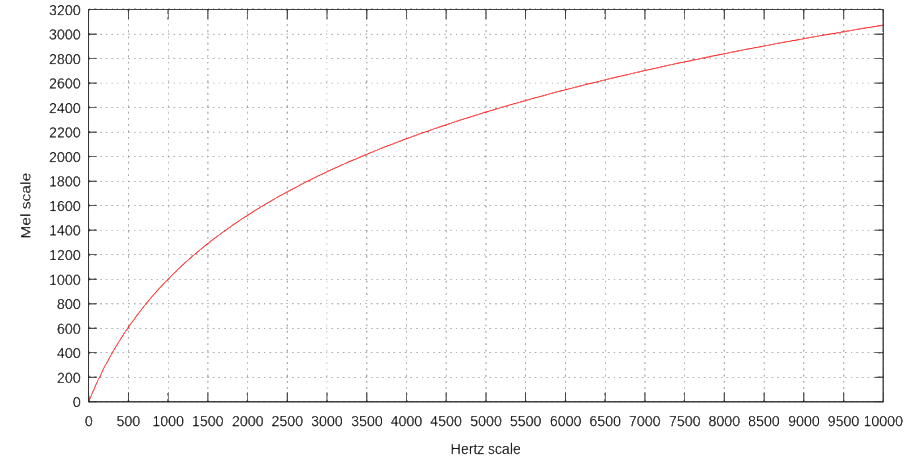


Human Ear



A1	A2	A2-A1
55Hz	110Hz	55Hz
A7	A8	A8-A7
3520Hz	7040Hz	3520Hz

The two distances are perceived equally



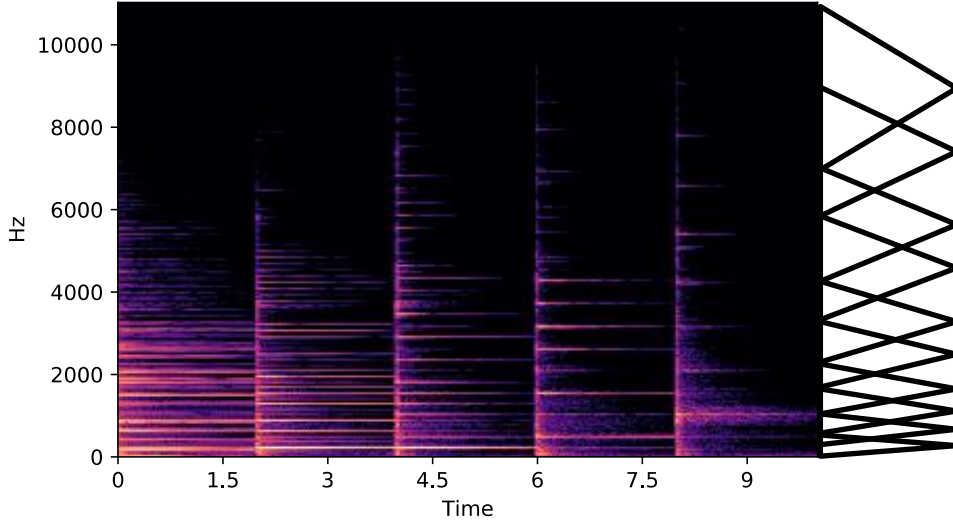
- The cochlea in the inner ear: Different frequencies excite different hair cells that are tied to nerves
- Range: 20-20,000 Hz

- The mel scale: Perceptual scale of pitches judged by listeners to be equal in distance from one another
- On a logarithmic scale



Mel Spectrogram

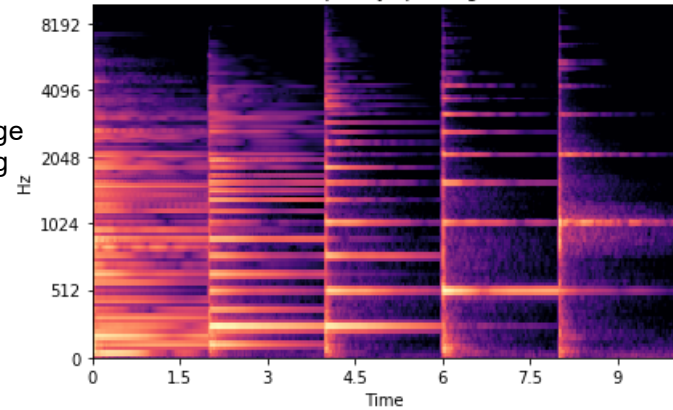
Spectrogram of Piano from C2 to C6



Weighted Average
and Log Scaling

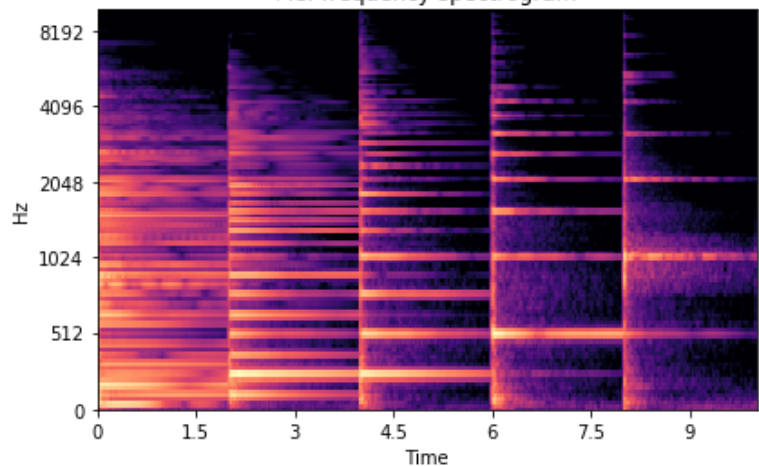


Mel-frequency spectrogram



Mel-Frequency Cepstrum Coefficients

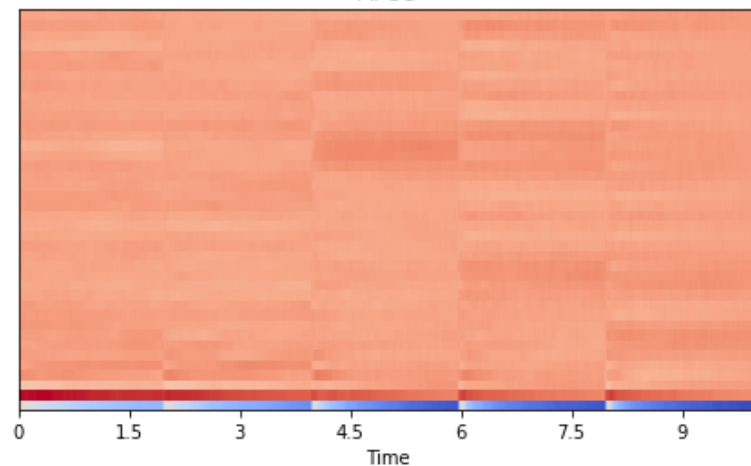
Mel-frequency spectrogram



DCT
(Discrete Cosine Transform)

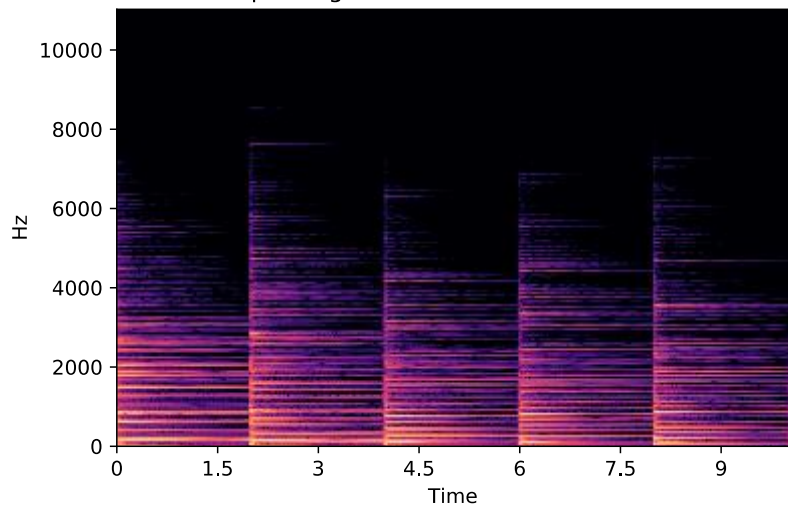


MFCC

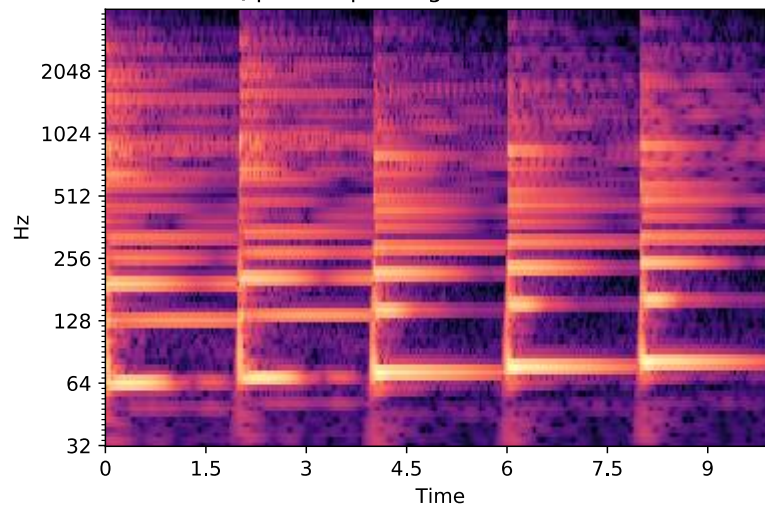


Constant Q Transform (CQT)

Spectrogram of Piano from C2 to E2



Constant-Q power spectrogram of Piano from C2 to E2

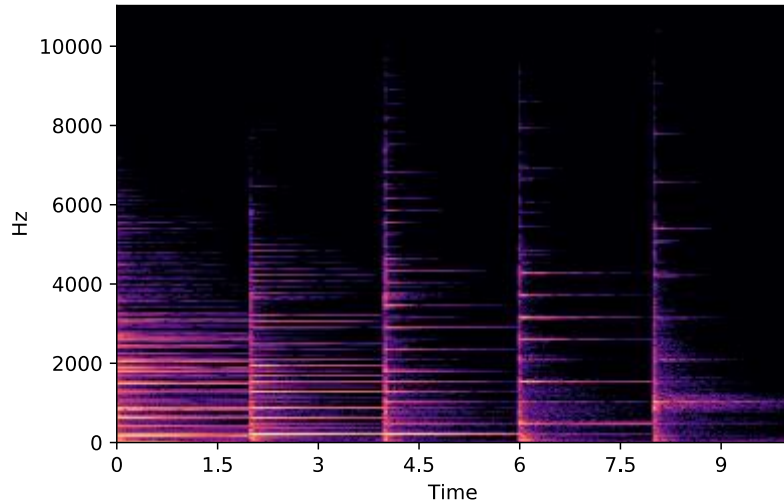


C_2	65.41
$C\#_2$	69.30
D_2	73.42
$D\#_2$	77.78
E_2	82.41

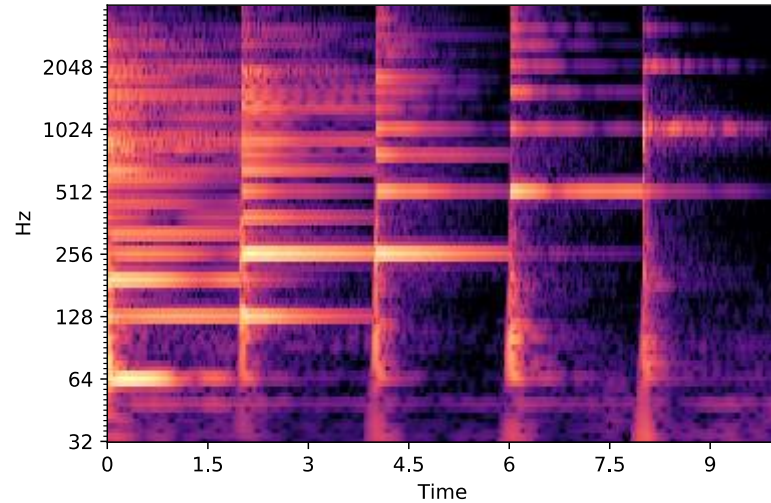


Constant Q Transform (CQT)

Spectrogram of Piano from C2 to C6



Constant-Q power spectrogram of Piano from C2 to C6

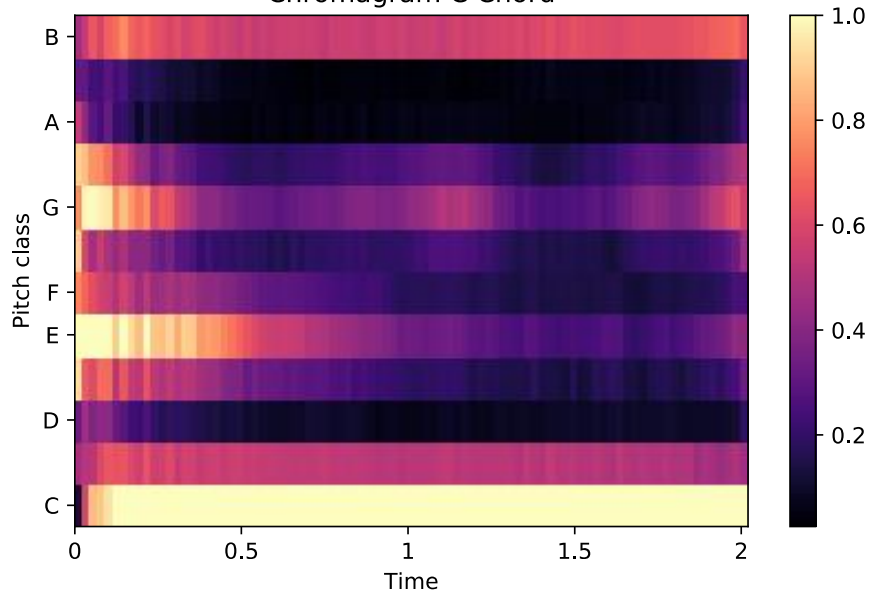


C_2	65.41
C_3	130.81
C_4	261.63
C_5	523.25
C_6	1046.50

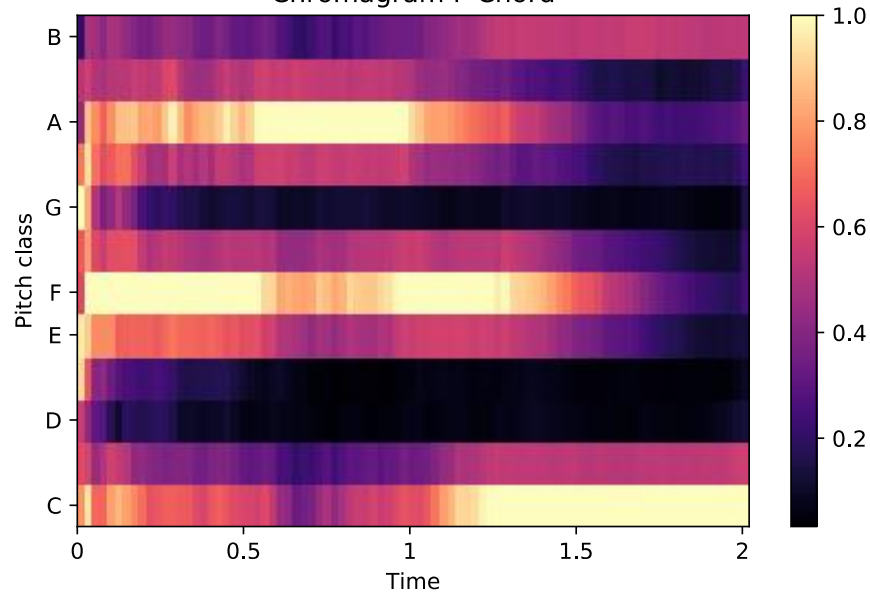


Chromagram

Chromagram C Chord

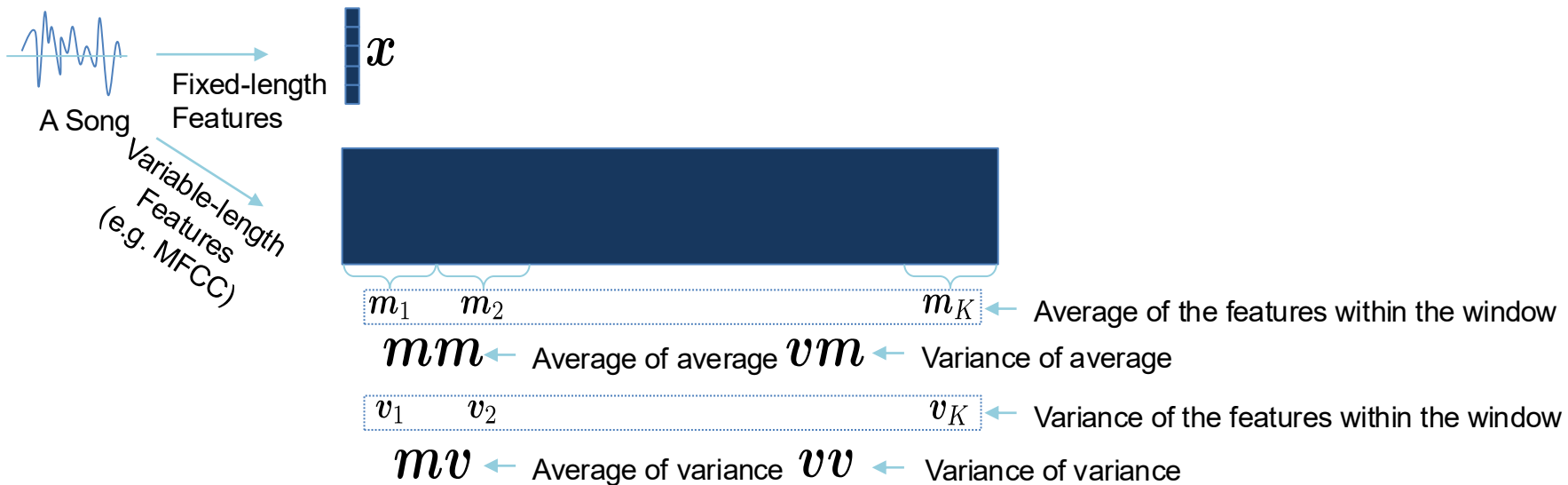


Chromagram F Chord



Audio Data as an Input to MIR Systems

Feature Aggregation (MARSYAS)



- The number of features: $|x| + |mm| + |vm| + |mv| + |vv|$

Functions

```
# load an audio file; this returns audio time series (np.ndarray) and sampling rate
y, sr = librosa.load(filename)
```

```
# ipython.display.audio generates an audio player so that you can play an audio file
ipd.Audio(y, rate=sr)
```

```
# Show some descriptive statistics of the audio data
sp.stats.describe(y)
```

```
# Visualize an Audio File using librosa.display.waveplot
librosa.display.waveplot(y, sr=sr)
plt.plot(y)
```

```
# Generate a pure tone
t = np.linspace(0, T, int(T*sr), endpoint=False) # time variable
y = 0.1*np.sin(2*np.pi*f0*t)
```

```
# find peaks inside the excerpt of the audio using scipy.signal.find_peaks
# after looking at the amplitude of peaks, set the height parameter
# in this example, the height of peaks is larger than 0.010
from scipy.signal import find_peaks
peaks, _ = find_peaks(y_slice, height=0.010)
peaks, _ = find_peaks(X_mag, distance = 220, height = 15)
```

```
# The Root-Mean Square (RMS) of the waveform magnitude within a frame indicates loudness
rms = librosa.feature.rms(y=y)
```

```
# Combine three sinusoids (y, y2, y3) after shifting them a little
y5 = 0.9*np.roll(y, 10)+1.2*np.roll(y2, 10)+2*np.roll(y3, 5)
```

```
# scipy.fft.fft computes the one-dimensional n-point discrete Fourier Transform (DFT)
# with the efficient Fast Fourier Transform (FFT) algorithm
X = sp.fft(y) Compute the one-dimensional discrete Fourier Transform
X_mag = np.absolute(X) # spectral magnitude
```

```
# Spectrogram
D = librosa.amplitude_to_db(np.abs(librosa.stft(y, n_fft=4096)), ref=np.max)
librosa.display.specshow(D, y_axis='linear', x_axis='time')
```

```
# Returns frequencies
freqs = librosa.fft_frequencies(sr=22050, n_fft=4096)
```

```
# Mel Spectrogram
S = librosa.feature.melspectrogram(y=y, sr=sr, n_mels=128, fmax=10000)
S_db = librosa.power_to_db(S, ref=np.max)
librosa.display.specshow(S_db, x_axis='time', y_axis='mel', sr=sr, fmax=10000)
```

```
# MFCC
mfccs = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=10)
librosa.display.specshow(mfccs, x_axis='time')
```

```
# CQT
CQT = librosa.amplitude_to_db(np.abs(librosa.cqt(y, sr=sr)), ref=np.max)
librosa.display.specshow(CQT, y_axis='cqt_note', x_axis='time')
librosa.display.specshow(D, y_axis='linear', x_axis='time')
```

```
# Chromagram
C = librosa.feature.chroma_cqt(y=y, sr=sr, bins_per_octave=24)
librosa.display.specshow(C, y_axis='chroma', x_axis='time')
```



More Music Signal Processing?

- Prof. Müller's Fundamentals of Music Processing
 - <https://www.audiolabs-erlangen.de/resources/MIR/FMP/C0/C0.html>
- Prof. Lerch's An Introduction to Audio Content Analysis
 - <https://www.audiocontentanalysis.org>
- LibROSA tutorial
 - <https://librosa.github.io/librosa/tutorial.html>

