

MIR: Music Information Retrieval/Research

- Introduction,
- Applications,
- What is “digital sound”?
- Code for **demonstrative** purposes
 - Code samples to give a sense of what researchers create*

<https://www.youtube.com/watch?v=YgYV-7-ohxQ>

MIR: Music Information Retrieval/Research

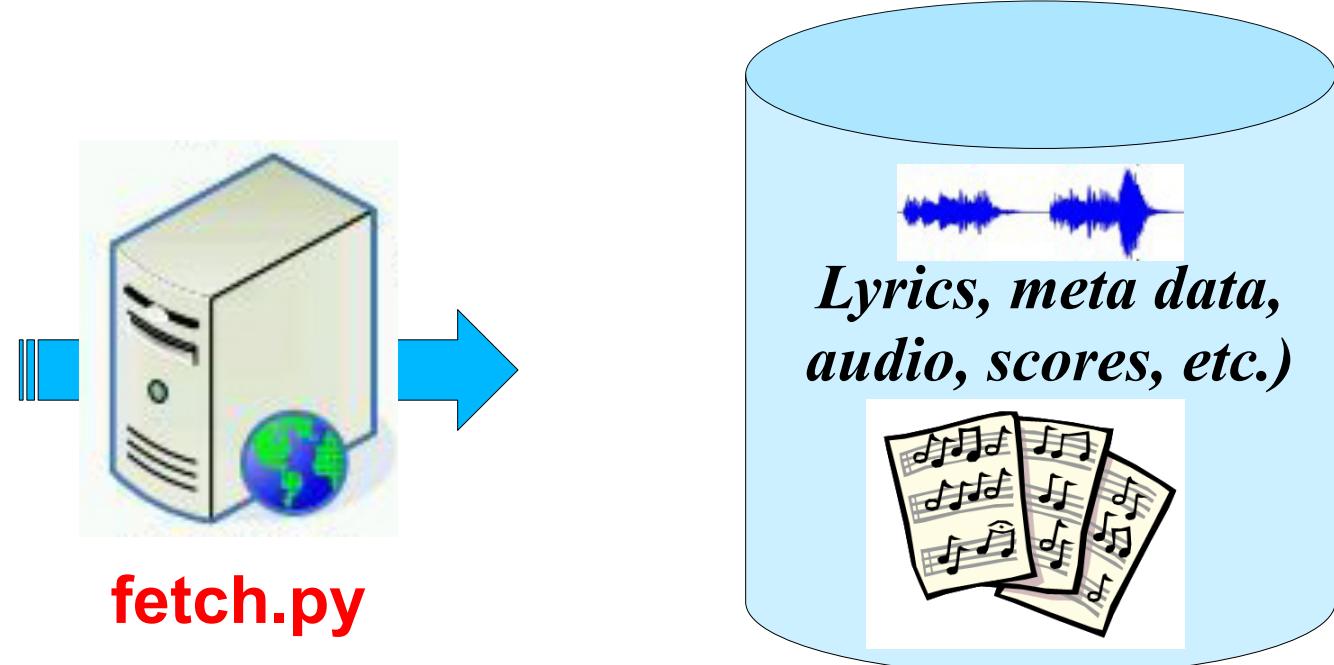
Müzik Bilgi Erişim

Developing computer based tools for:

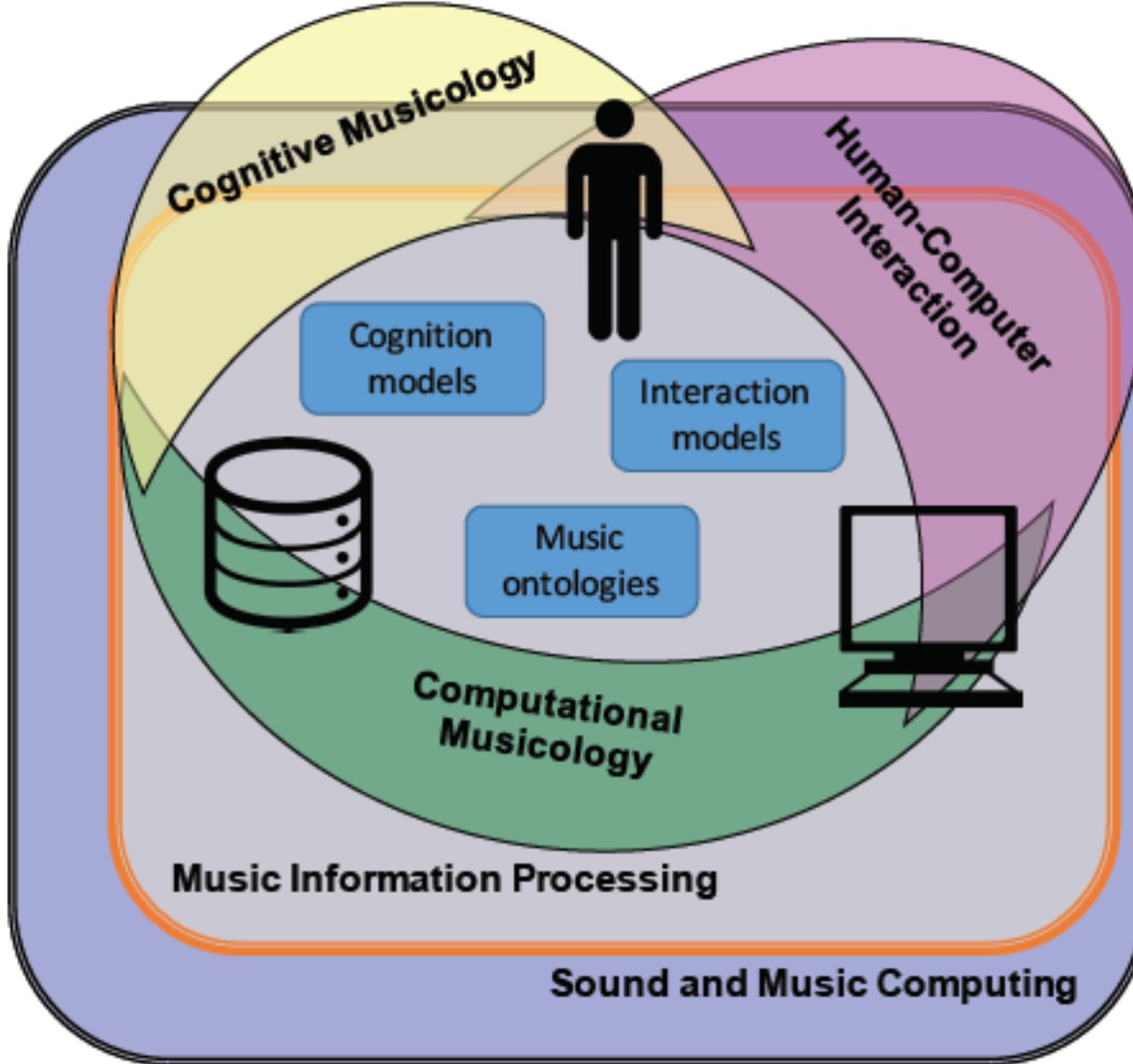
- *Extracting, accessing, representing, ... information from music data*
- *Facilitating query of items in large databases.*

User query interface:

- Text
- Humming/singing
- Recording example
- Notation
- ...



MIR is a multi-disciplinary field



Pattern recognition

Musicology

Signal processing

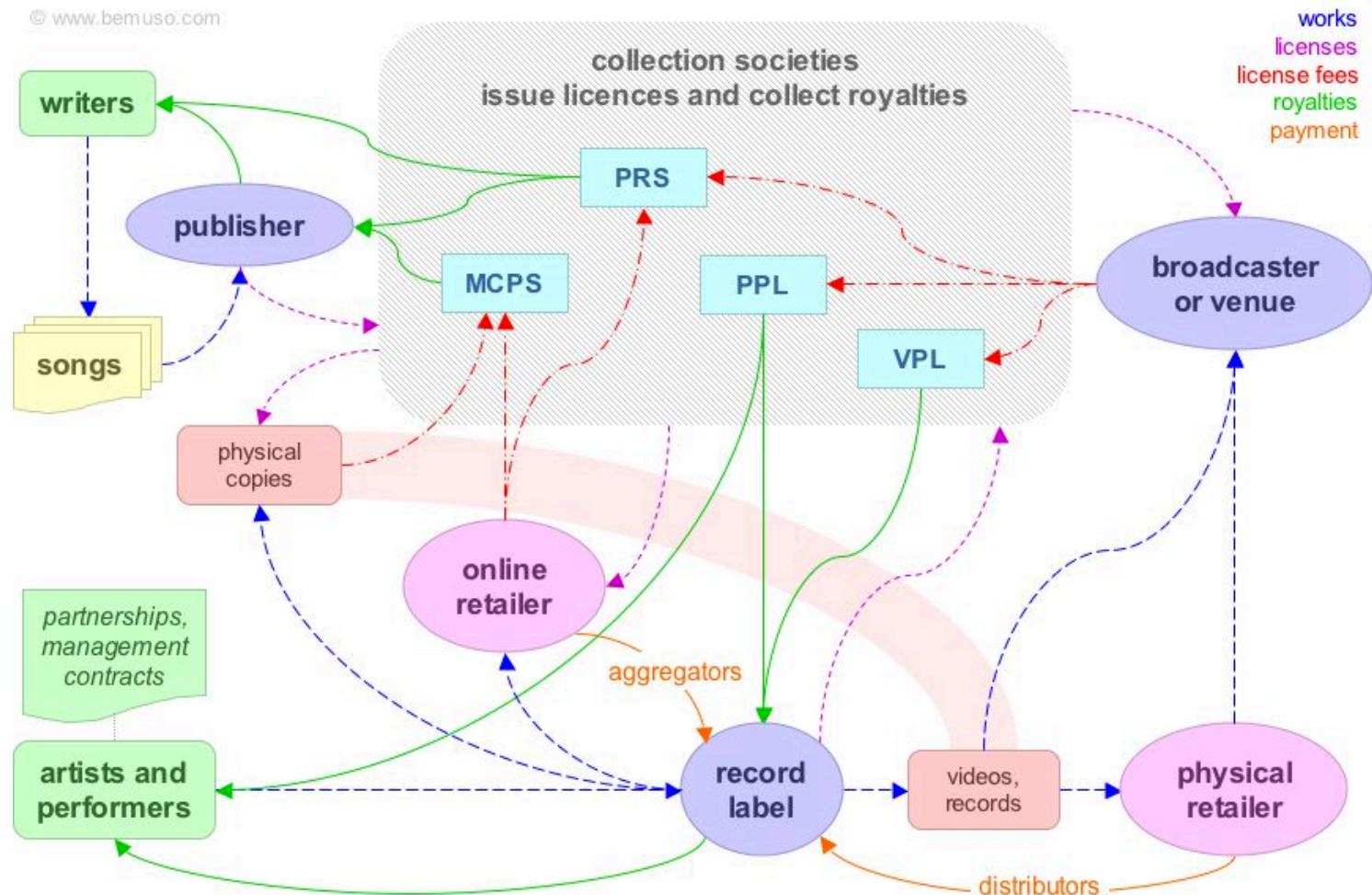
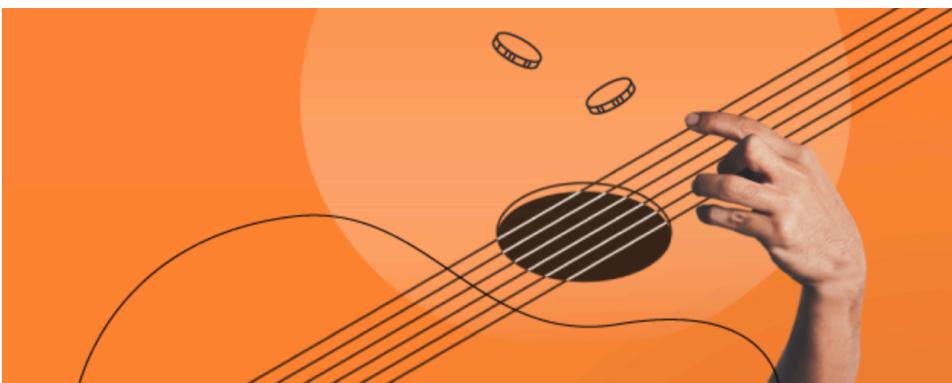
Music perception and cognition

Hardware (recording, interfaces, etc.)

Computing, programming

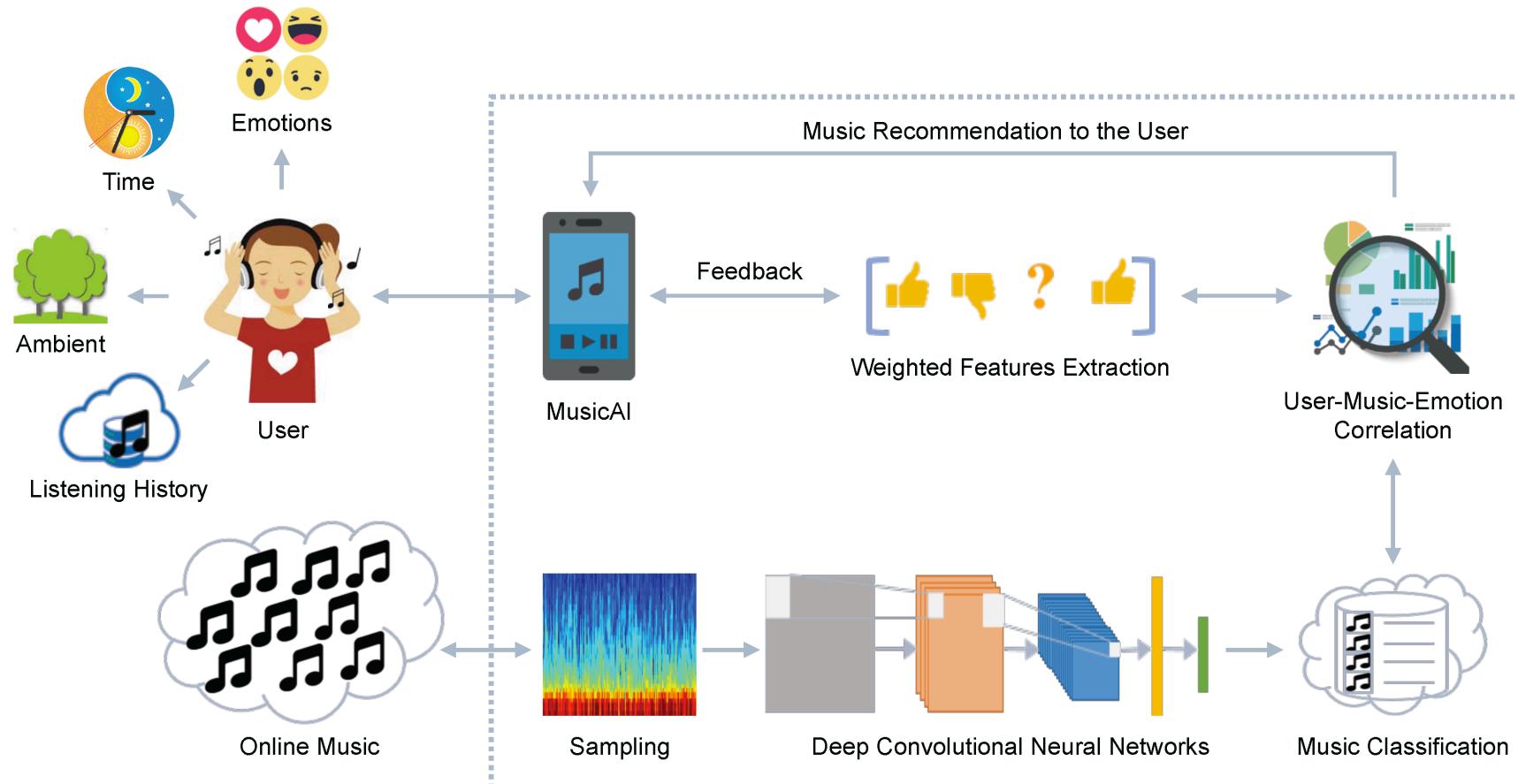
Popular MIR Applications

- Royalty rights tracking
 - *Monitoring broadcast*



Popular MIR Applications

- Recommendation systems
- Music discovery



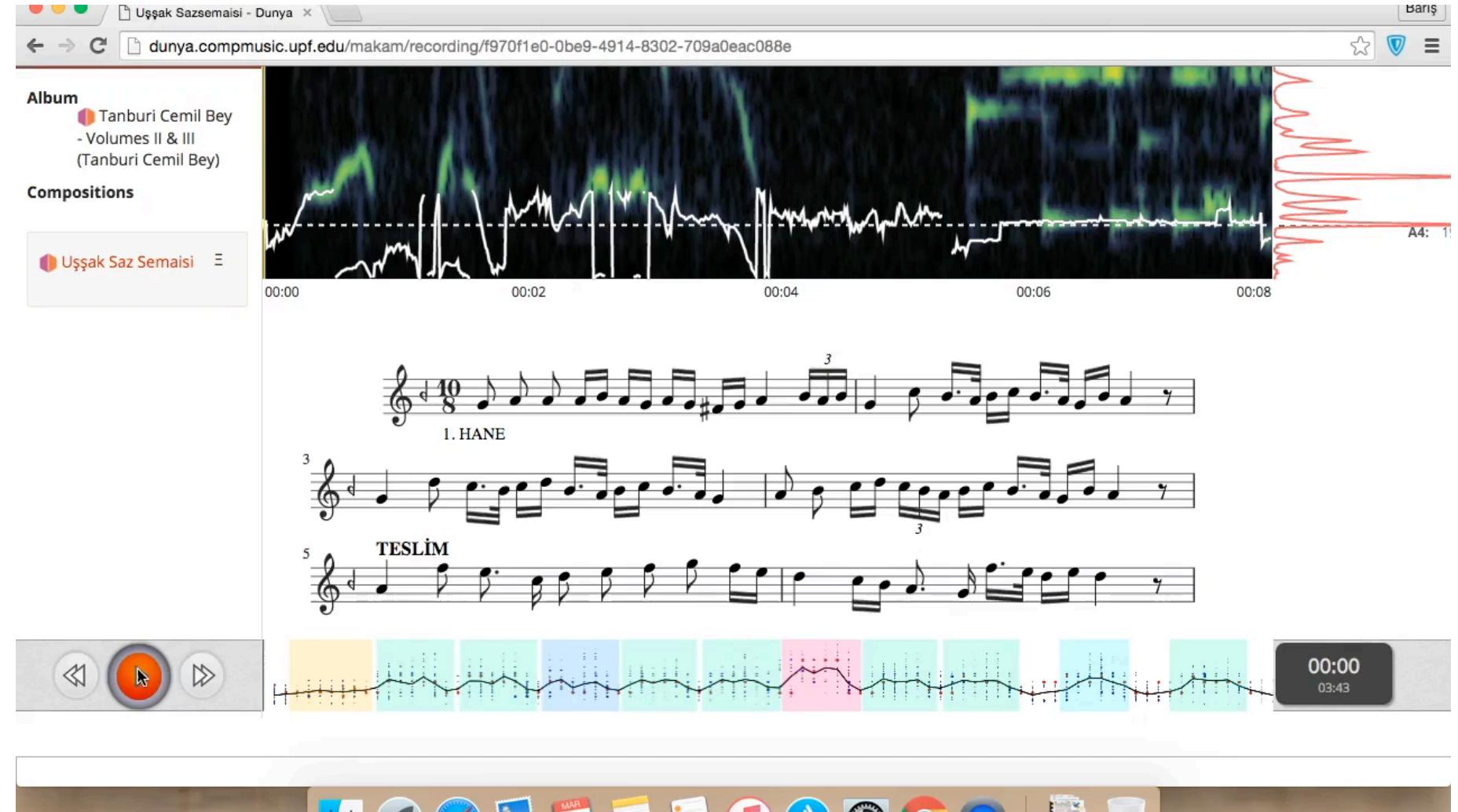
Popular MIR Applications

- Audio-score alignment, score following



Sertan Şentürk · 1st
Lead Data Scientist - R&D at Kobalt
London Area, United Kingdom · [Contact info](#)

Phd, 2017
@MTG-UPF, Barcelona



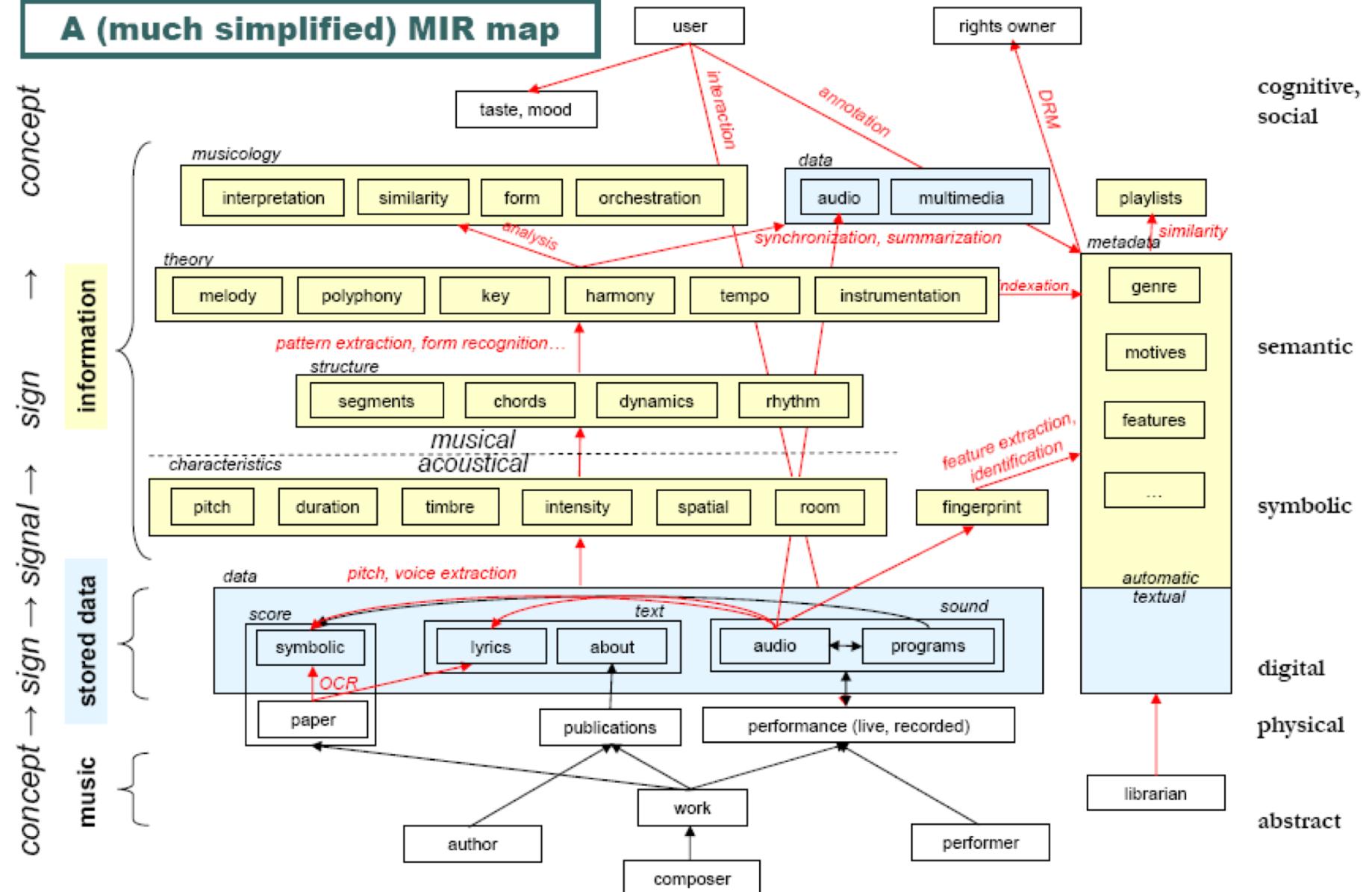
Popular MIR Applications

- Automatic chord detection

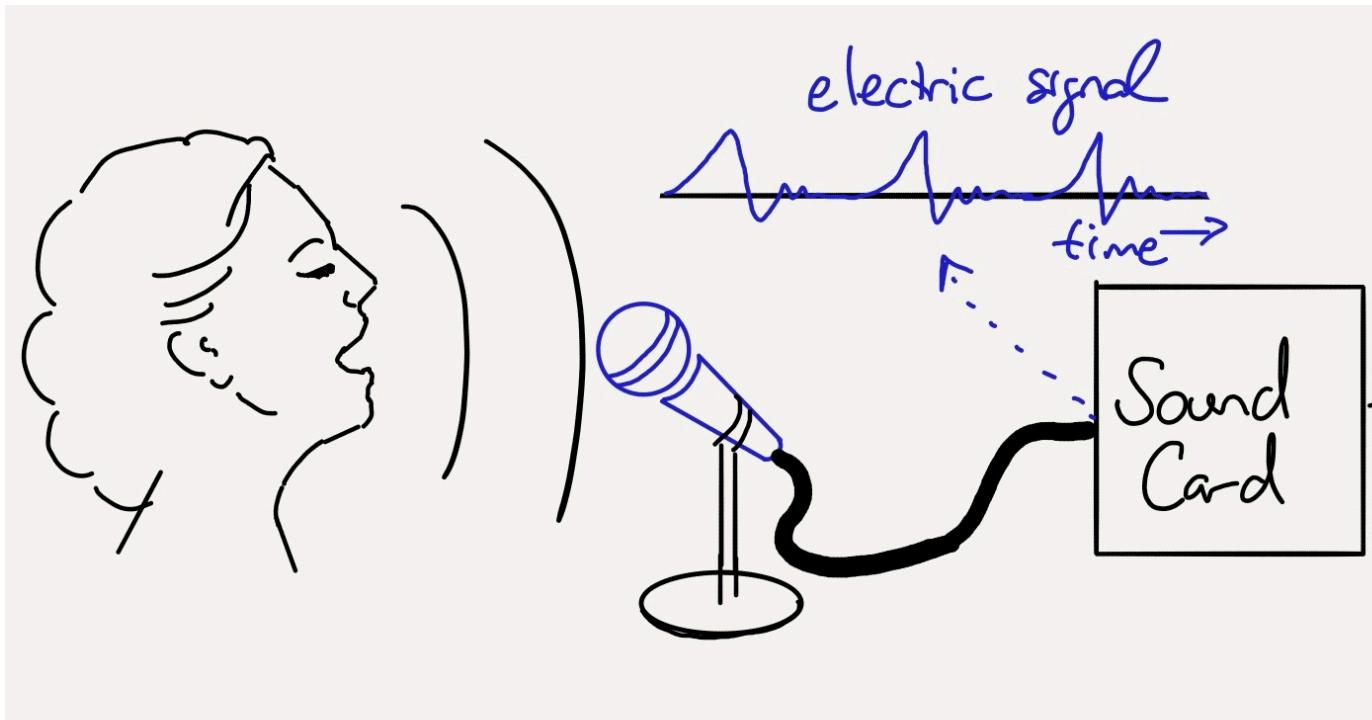
The screenshot shows a web browser displaying the chordify.net website. The URL in the address bar is `chordify.net/chords/queen-we-are-the-champions-queenofficial`. The page title is "Queen - We Are The Champions (Official Video)". Below the title, it says "by: Queen & David Bowie". There are navigation buttons for "DIAGRAMS" (selected), "OVERVIEW", and "IMPROVE". A dropdown menu "Problem with the chords? ▾" is open. On the right, there is a "SIMPLIFY CHORDS" toggle switch. Below these controls are playback controls (rewind, play, forward), volume, loop, tempo, capo, transpose, midi, and print buttons. A chord progression diagram is shown at the bottom, starting with a C major chord. To the right of the main content area, there is a video player showing a clip from the "Queen - We Are The Champions" music video. Below the video, there is a "YouTube" logo. At the bottom of the page, there is a section titled "Similar to Queen - We Are The Champions (Off" with a thumbnail for "Queen – Bohemian Rhapsody (O...".

<https://chordify.net/>

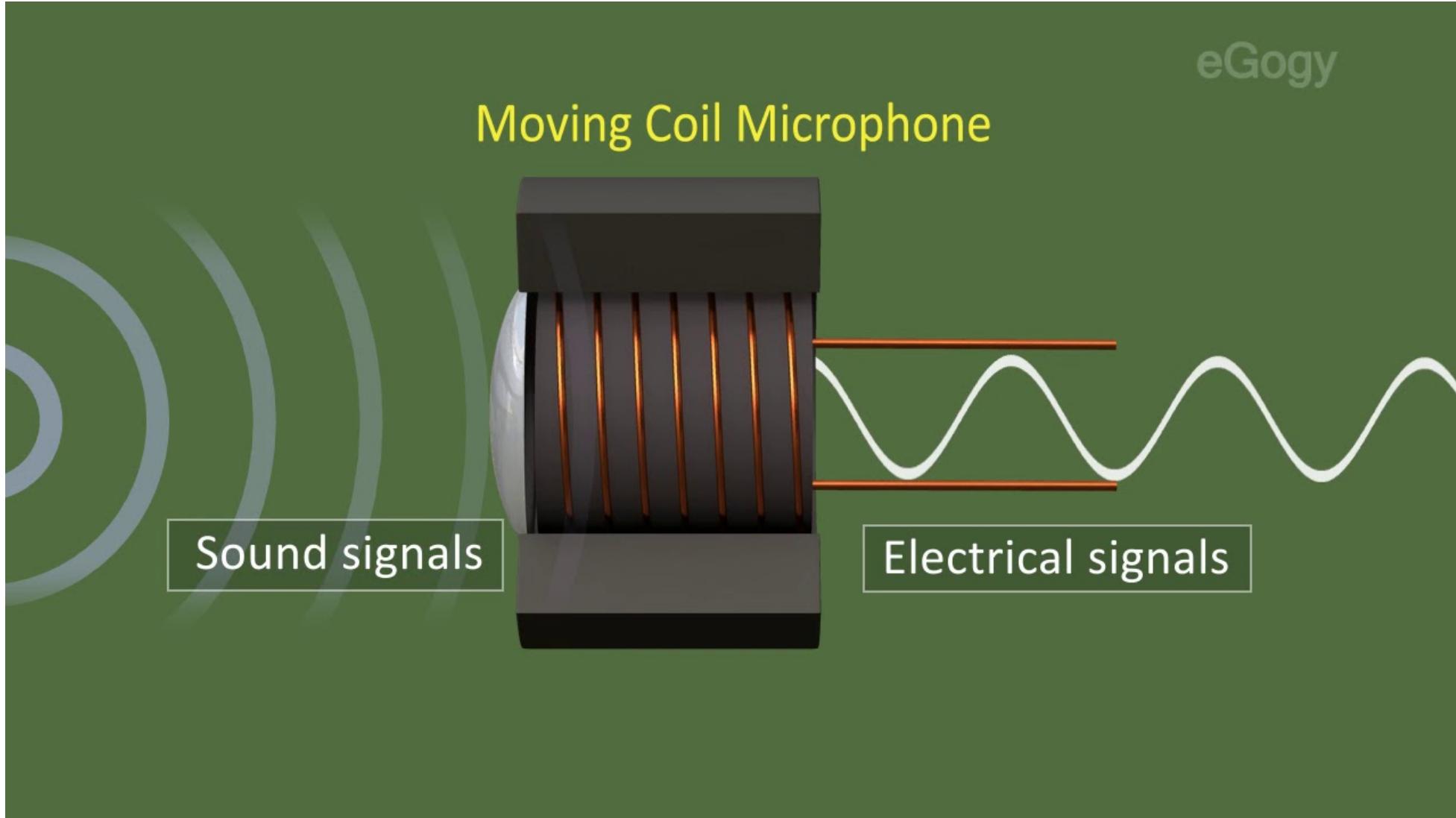
A (much simplified) MIR map



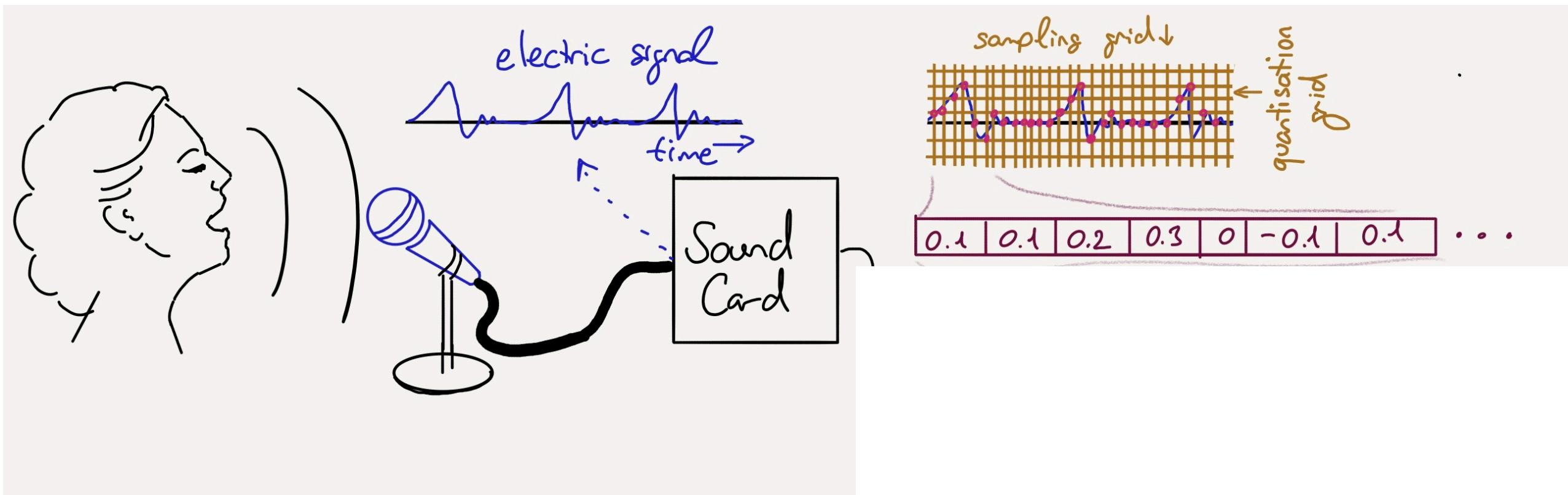
Baby steps towards audio signal processing in Python

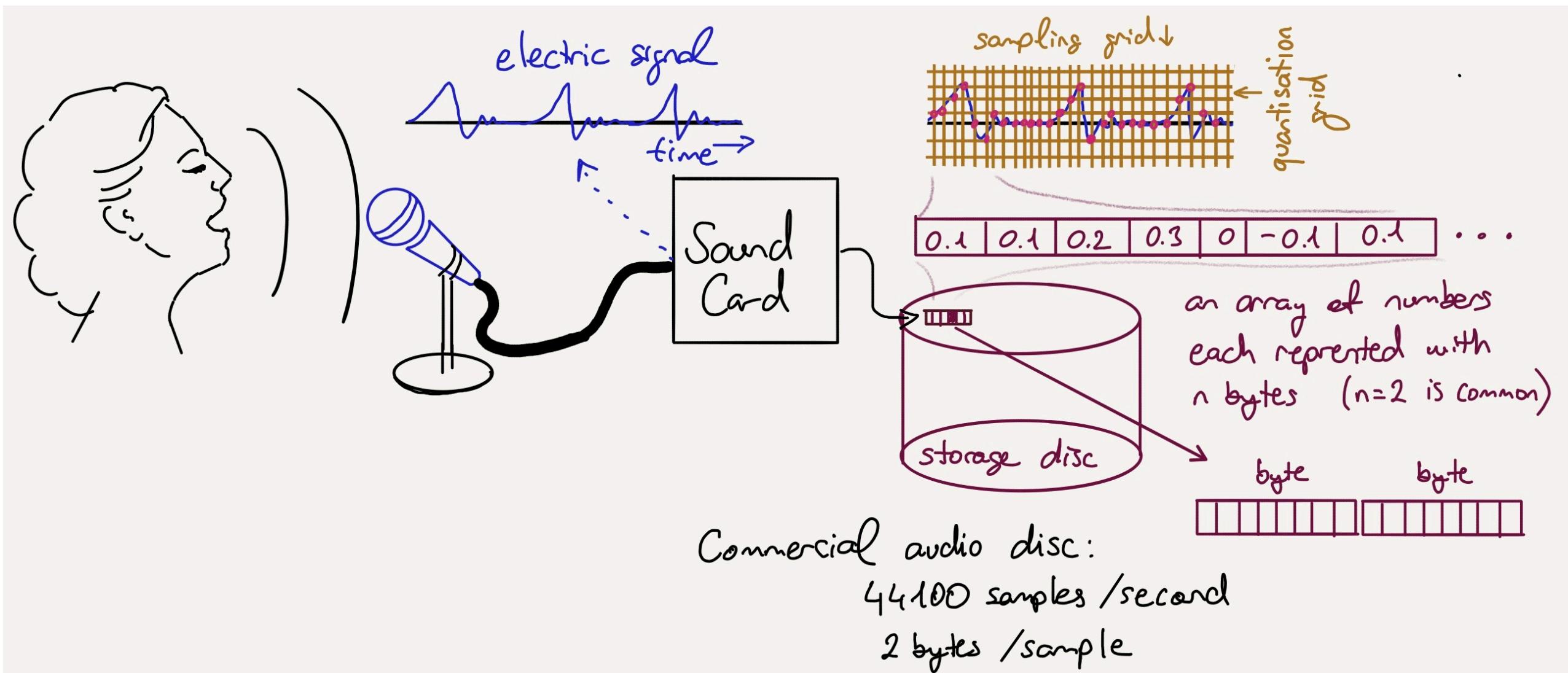


Microphones, loudspeakers; how do they work?

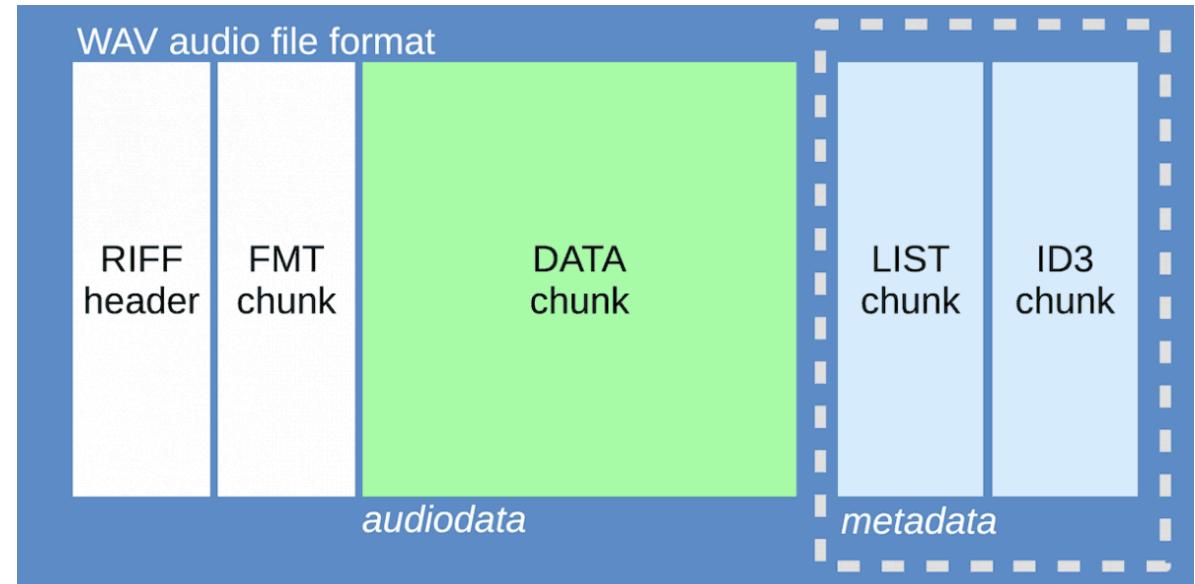


<https://www.youtube.com/watch?v=Y585z2XRFFs&list=RDecPUTGDX5cw&index=10>





Storing audio data(series of numbers) in files



To read audio data from audio files,
we need functions designed for that specific format

Let's see some sample code



DigitalAudio_sampleCode.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved



Comment



Share



+ Code + Text



RAM



Disk



Editing



▼ Reading digital audio files



Audio file reading functions are available in various libraries. This example uses [Sms-tools](#), the accompanying code resource for [Xavier Serra's Coursera course](#). We would first need to clone the repo to start using it.



13s

```
#We need to clone the repo to start using it.  
!git clone https://github.com/MTG/sms-tools.git  
import sys  
import os  
import numpy as np  
import matplotlib.pyplot as plt  
import IPython  
  
sys.path.append('sms-tools/software/models/')  
from utilFunctions import wavread, wavwrite
```

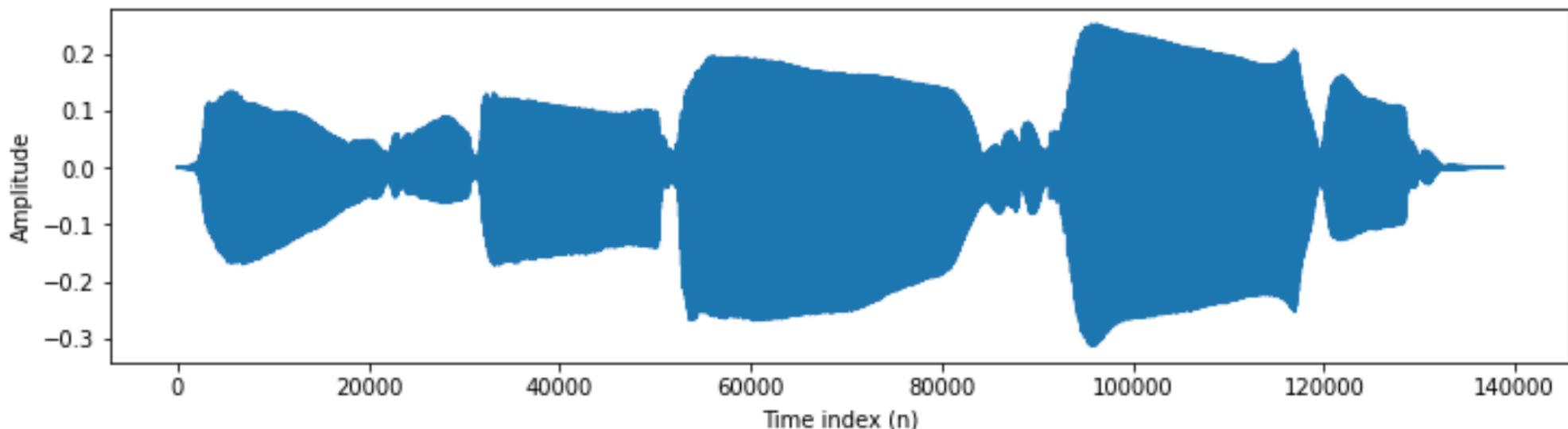
Let's see some sample code

Reading an audio file and plotting the signal loaded



```
fs, x = wavread('sms-tools/sounds/sax-phrase-short.wav')
fig = plt.figure(figsize=(12,3))
plt.plot(x)
plt.xlabel('Time index (n)'); plt.ylabel('Amplitude');
IPython.display.Audio(x, rate=fs)
```

▶ 0:00 / 0:03



<https://drive.google.com/file/d/1iyV5KTUWx4p68ByDoXjA1sn97XjgKkGD/view?usp=sharing>

Let's see some sample code

Print a few elements of the audio

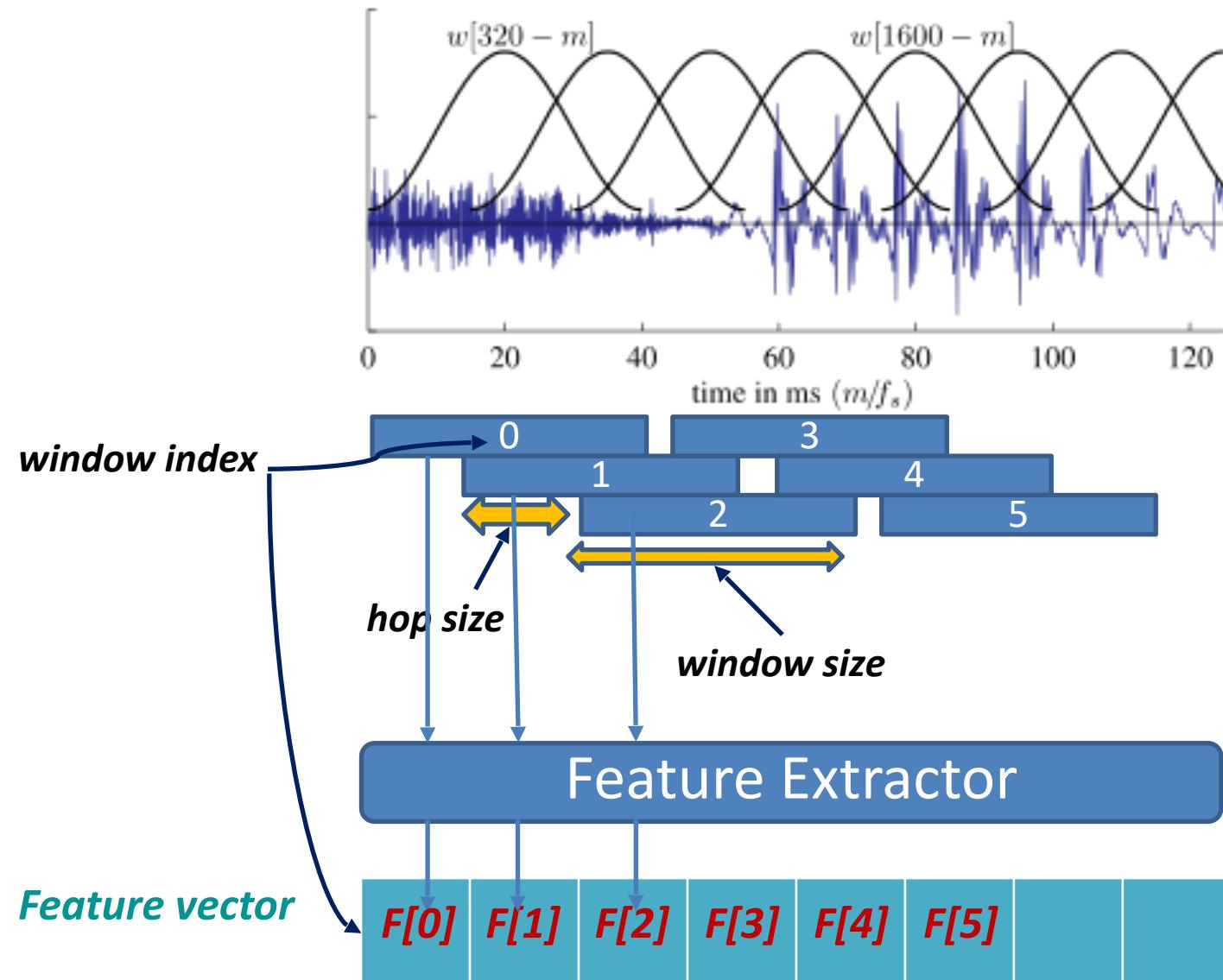
```
      start  stop  
[17] print(x[50000:50020])
```

```
[-0.12344737 -0.10858486 -0.09179968 -0.07345805 -0.05505539 -0.03827021  
 -0.02462844 -0.01327555 -0.00412       0.00332652  0.00900296  0.01361126  
  0.02035585  0.0283517   0.03482162  0.03927732  0.04412977  0.0516068  
  0.05865658  0.06576739]
```

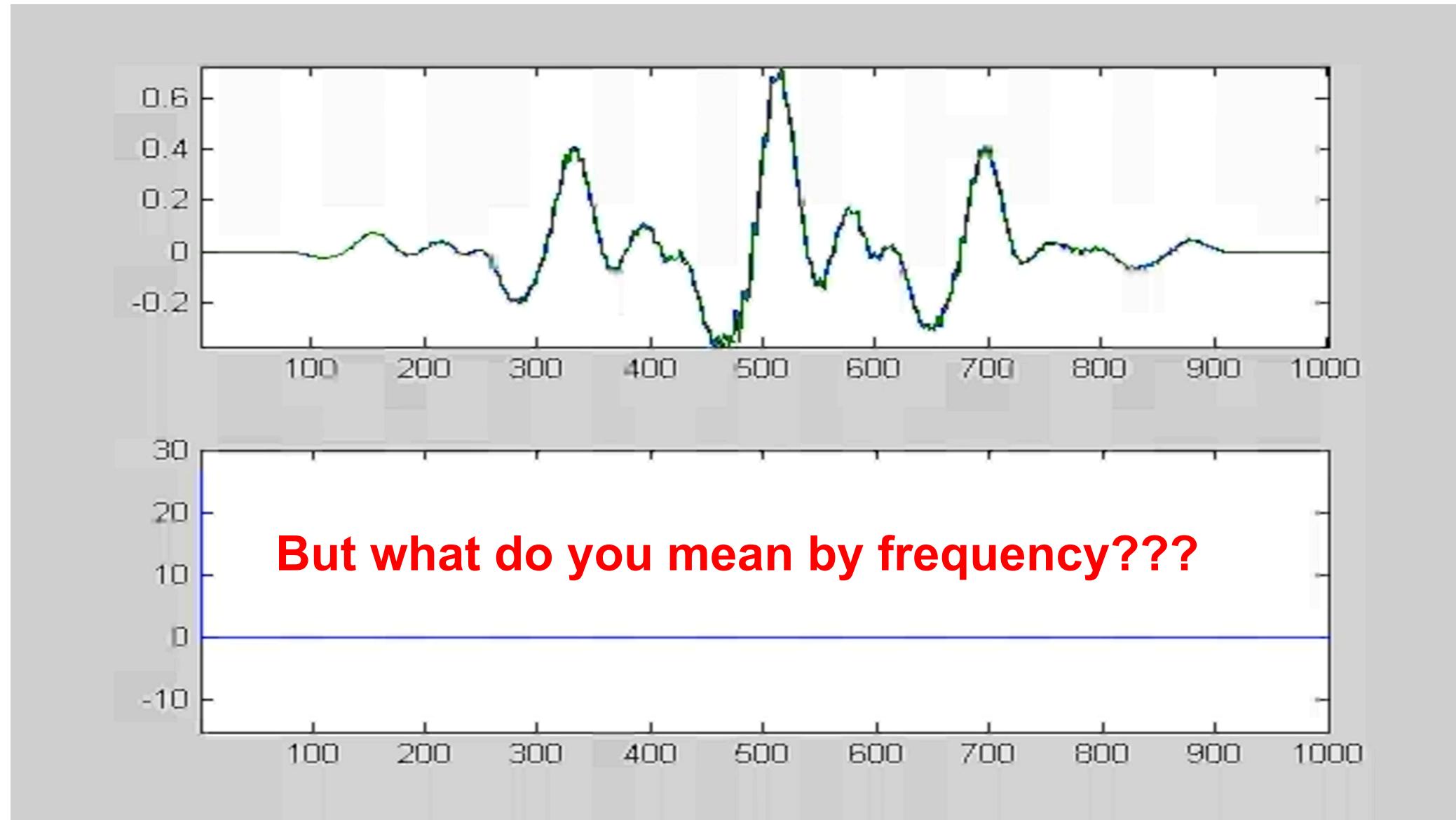
OK, fine,

But how do we extract any information from these series of numbers?

Chopping signal into frames and extracting information



Autocorrelation based fundamental frequency/period estimation



But what do you mean by frequency???

Frequency (of a signal) : the number of repetitions in 1 second duration.

Fundamental frequency extraction using the autocorrelation

The function below performs an auto-correlation based fundamental frequency estimation as explained in the lecture.

The [autocorrelation](#), r_k , of a signal x at lag k is defined as:

$$r_k = \sum_n x[n]x[n + k]$$

which is basically a dot product of the signal with its shifted version



```
def estimateF0_autoCorr(x_win, fs, minF0, maxF0):
    '''F0 detection on a single frame using autocorrelation
    Parameters
    -----
    x_win : numpy.array
        Windowed signal frame
    fs,minF0,maxF0 : int
        Sampling rate, minimum and maximum F0 limits

    Returns
    -----
    f0 : float
        Estimated f0 in Hz
    ...
```



Fundamental frequency extraction using the autocorrelation

```
def estimateF0_autoCorr(x_win, fs, minF0, maxF0):

    f0 = 0
    minT0 = int(fs/maxF0)
    maxT0 = int(fs/minF0)
    maxValAC = -1; T0 = -1
    for k in range(minT0, maxT0):
        x_win_shifted = np.hstack((np.zeros(k),x_win[:-k]))
        autoCorr = np.dot(x_win,x_win_shifted)
        if autoCorr > maxValAC:
            T0 = k
            maxValAC = autoCorr
    f0 = float(fs) / T0
    return f0
```

Fundamental frequency extraction using the autocorrelation

```
from scipy.signal import get_window

# Analysis parameters
minF0 = 50 #in Hz
maxF0 = 2000 #in Hz
windowSize = 4096
hopSize = 1024

w = get_window('blackman', windowSize)
startIndexes = np.arange(0, x.size - windowSize, hopSize, dtype = int)
numWindows = startIndexes.size

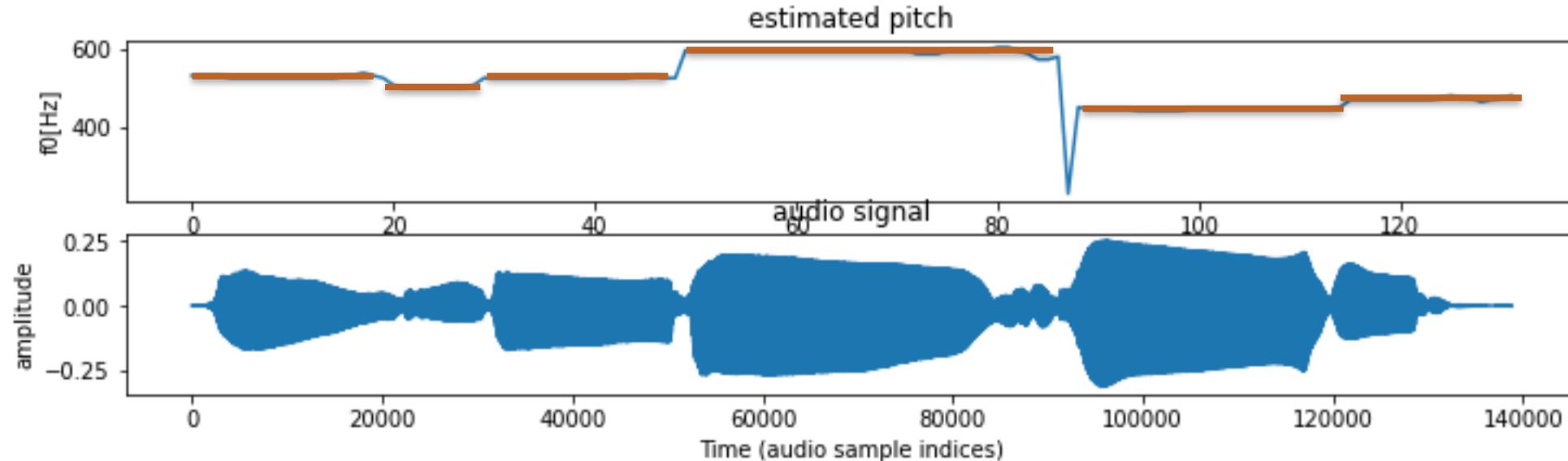
#F0 estimation for each window
f0 = []
for k in range(numWindows):#framing/windowing
    x_win = x[startIndexes[k] : startIndexes[k] + windowSize] * w#window applied here
    f0.append(estimateF0_autoCorr(x_win, fs, minF0, maxF0))
f0 = np.array(f0)
```

Fundamental frequency extraction using the autocorrelation

```
fig = plt.figure(figsize=(12,3))
plt.subplot(2,1,1)
plt.plot(f0)
plt.title('estimated pitch')
plt.ylabel('f0[Hz]')
plt.xlabel('Time (frame indices)')
plt.subplot(2,1,2)
plt.plot(x)
plt.title('audio signal')
plt.ylabel('amplitude')
plt.xlabel('Time (audio sample indices)')

Text(0.5, 0, 'Time (audio sample indices)')
```

One potential next step is to estimates the notes played in this recording: automatic transcription



Looks interesting, where do I find more material?

← → ⌂ youtube.com/c/MusicTechnologyGroup

YouTube Premium TR

Ara

Ana Sayfa

Kesfet

Abonelikler

Originals

/YouTube Music

Kitaplık

 MTG website

MusicTechnologyGroup
1,31 B abone

ABONE OLUNDU

ANASAYFA VİDEOLAR OYNATMA LİSTELERİ TOPLULUK KANALLAR HAKKINDA


We are the Music Technology Group
MusicTechnologyGroup • 1,3 B görüntüleme • 1 yıl önce
Presentation of the Music Technology Group (MTG), a research group of the Universitat Pompeu Fabra in Barcelona, part of its Department of Information and Communication Technologies. The group...

Yüklemeler ► TÜMÜNÜ OYNAT

Looks interesting, where do I find more material?

<https://ccrma.stanford.edu/>

Looks interesting, where do I find more material?

← → ⌂ coursera.org/learn/audio-signal-processing

☆ 🔑 🎓 🔒 🗝️ 🗐️

coursera Explore ▾ What do you want to learn? Q Enterprise Students Log In Join for Free

Browse > Computer Science > Software Development

Audio Signal Processing for Music Applications

Offered By
Universitat Pompeu Fabra of Barcelona
Stanford University

★★★★★ 4.8 266 ratings | 👍 96%

 Xavier Serra [+1 more instructor](#)

Enroll for Free
Starts Aug 17

45,151 already enrolled

<https://www.coursera.org/learn/audio-signal-processing>