

Acoustic Correlates of Timbre

Johanna Devaney
Brooklyn College and Graduate Center, CUNY

<https://github.com/jcdevaney/TOSS2025>

Overview

1

Signal Processing Basics

2

Acoustic Descriptors

3

GUI-based Tools

4

Code-Based Tools

5

Summary

6

Overview

Schedule

- ▶ **Signal Processing Basics/Acoustic Descriptors**
- ▶ *Break*
- ▶ **GUI-based tools**
- ▶ *Break*
- ▶ **Python-based tools**
- ▶ **Summary**



Summary

Resources

- ▶ **GitHub Repo ([https://github.com/jcdevaney/
TOSS2025](https://github.com/jcdevaney/TOSS2025))**
 - Audio files
 - Colab notebooks
 - PDFs of papers for further reading
 - Copy of Sonic Annotator for easy installation
 - Copy of these slides

Overview

“Note”-level analysis

- ▶ **Note-based analysis can be useful for Western art music**
 - Particularly for notated instrumental music
 - Adopted for studying non-Western art music traditions
- ▶ **Fails to capture all of the nuance in the performance**
 - Particularly for vocals (e.g., transitions between notes)
- ▶ **Rooted in the idea is that the performer adds expressivity to the musical score**
 - The musical score is the work/“composer as genius”

Overview

1

Signal Processing Basics

2

Acoustic Descriptors

3

GUI-based Tools

4

Code-Based Tools

5

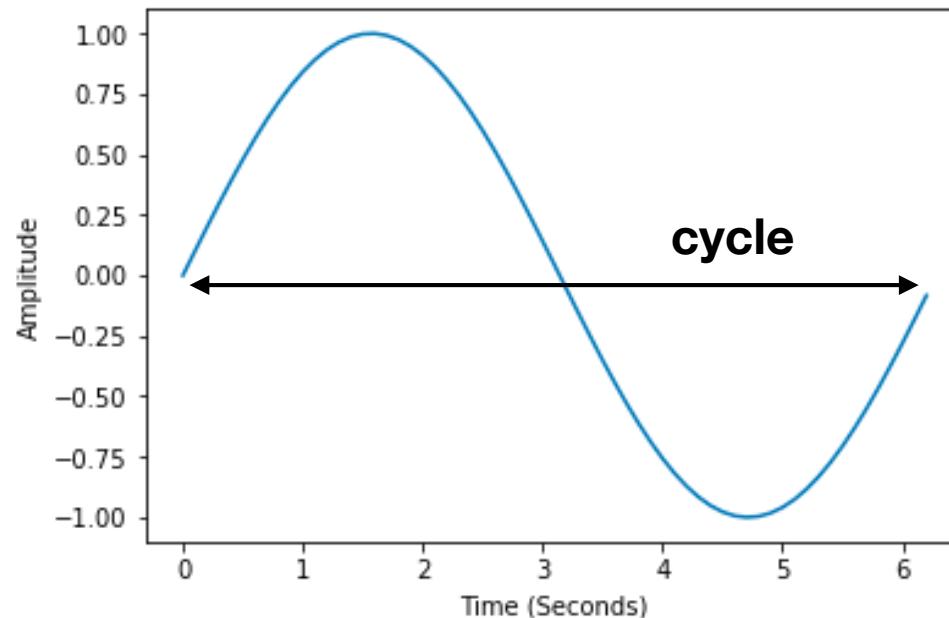
Summary

6

Signal Processing Basics

Frequency

- ▶ **periodic acoustic waveform**
 - air pressure varying with a repeated pattern
- ▶ **cycle**
 - one repetition of a waveform

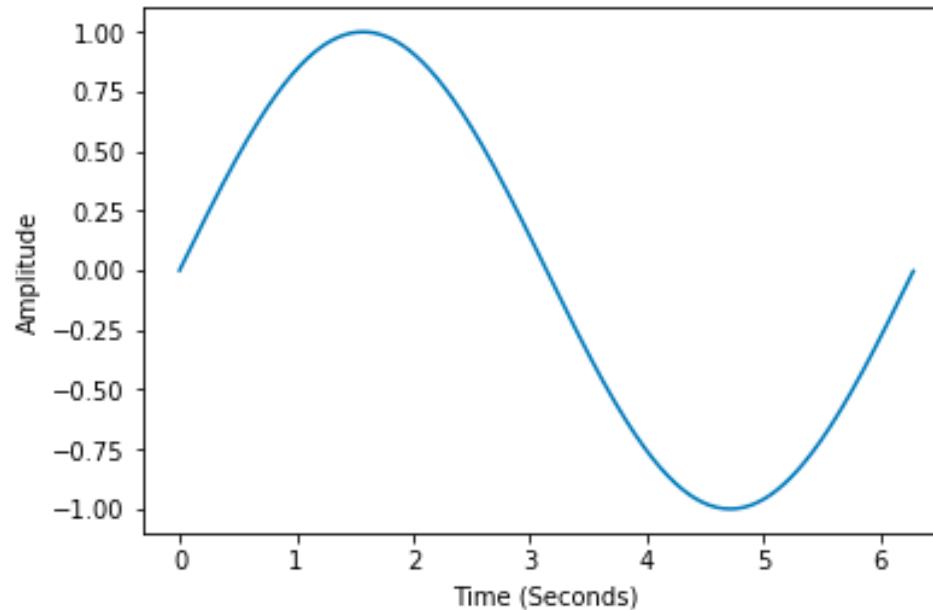


Signal Processing Basics

Amplitude vs Decibels

- ▶ **amplitude**

- amount of air pressure change



- ▶ **decibels**

- scaling of amplitude values

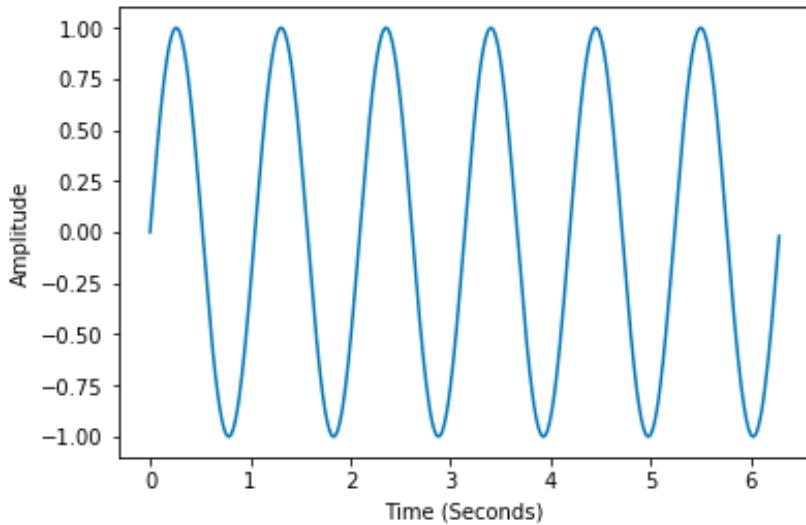
$$dB = 10 * \log_{10}(level/reference)$$

Signal Processing Basics

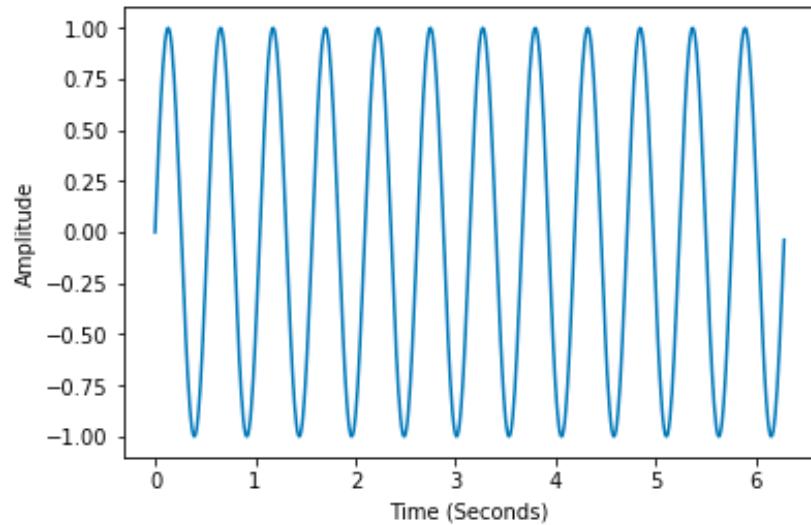
Frequency

- ▶ **frequency**

- number of cycles per second of wave traveling through the air, measured in hertz



approx. 1 Hz



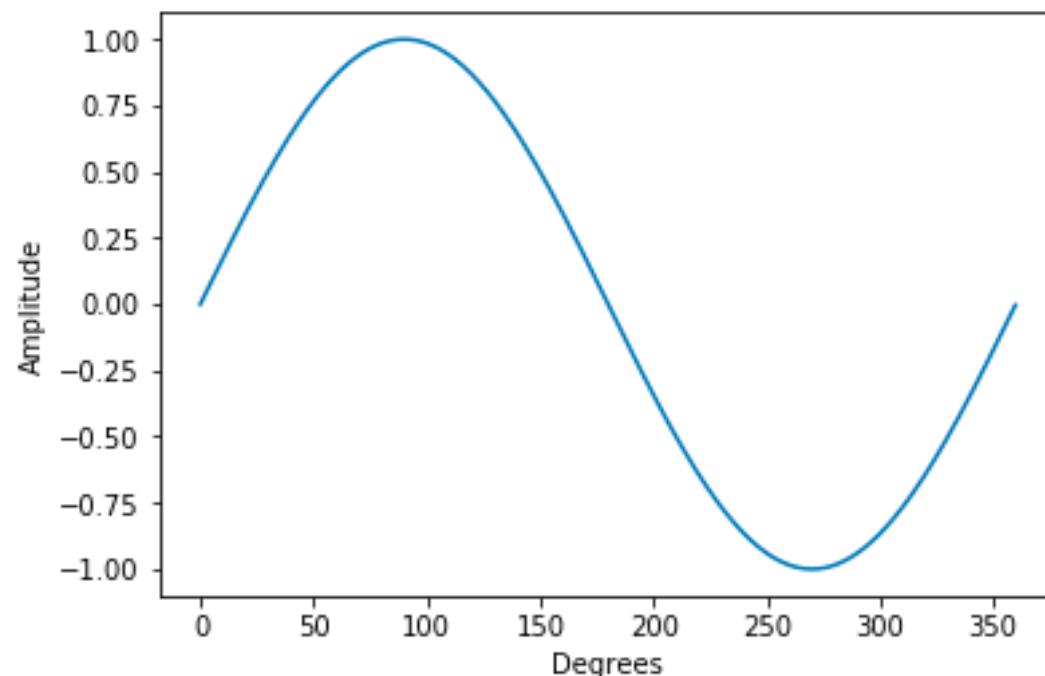
approx. 2 Hz

Signal Processing Basics

Phase

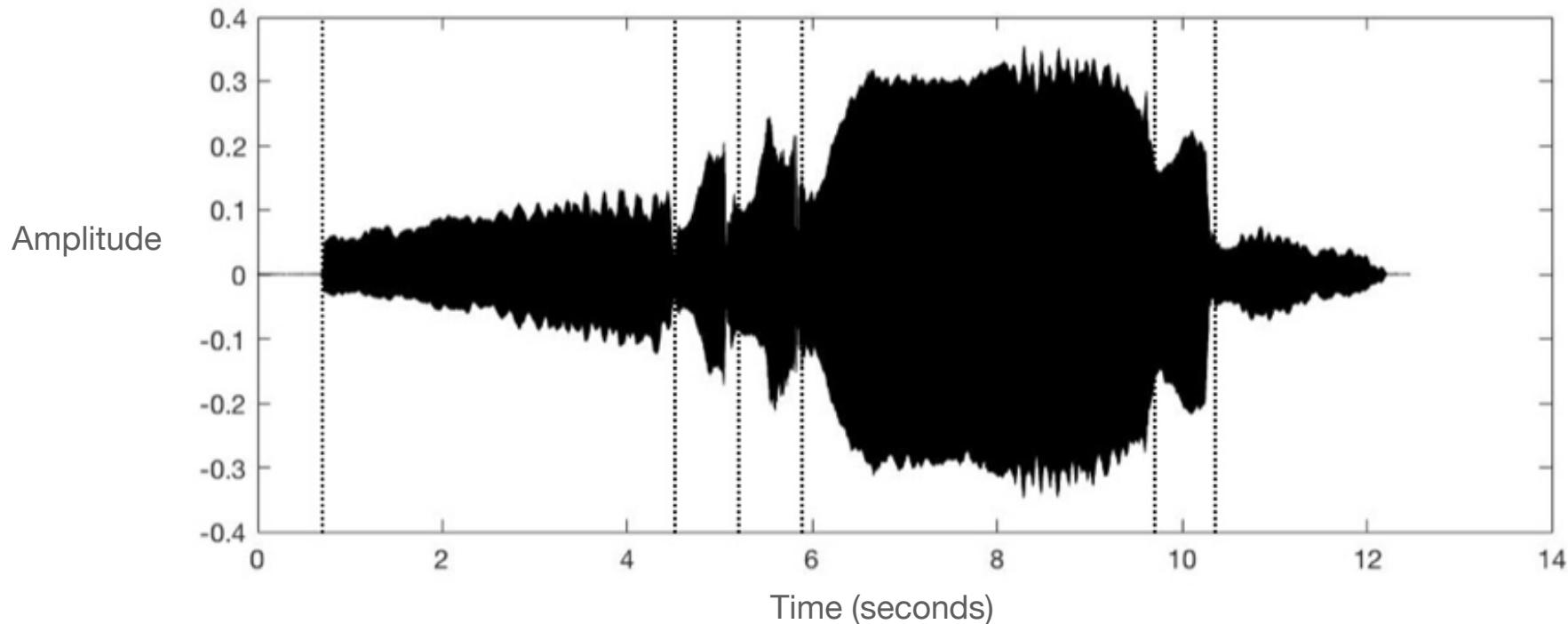
- ▶ **phase**

- starting point is initial phase, measured in degrees across the entire cycle
- measured by degrees (360 degrees = 2π)



Signal Processing Basics

Time-domain representation

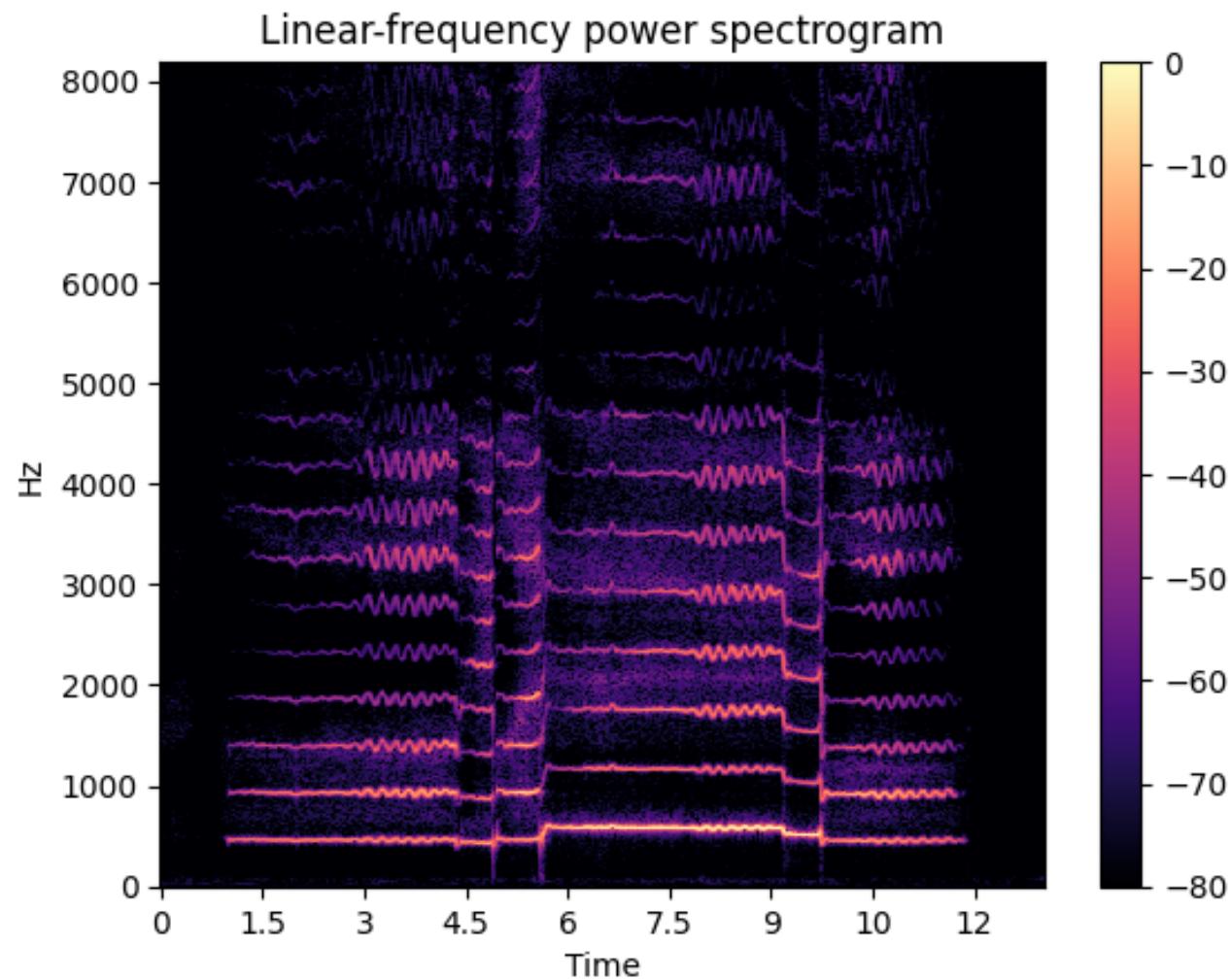


Time domain representation of an audio recording of the opening phrase of Schubert's "Ave Maria". The vertical dotted lines indicate the boundaries of the notes in the score below



Signal Processing Basics

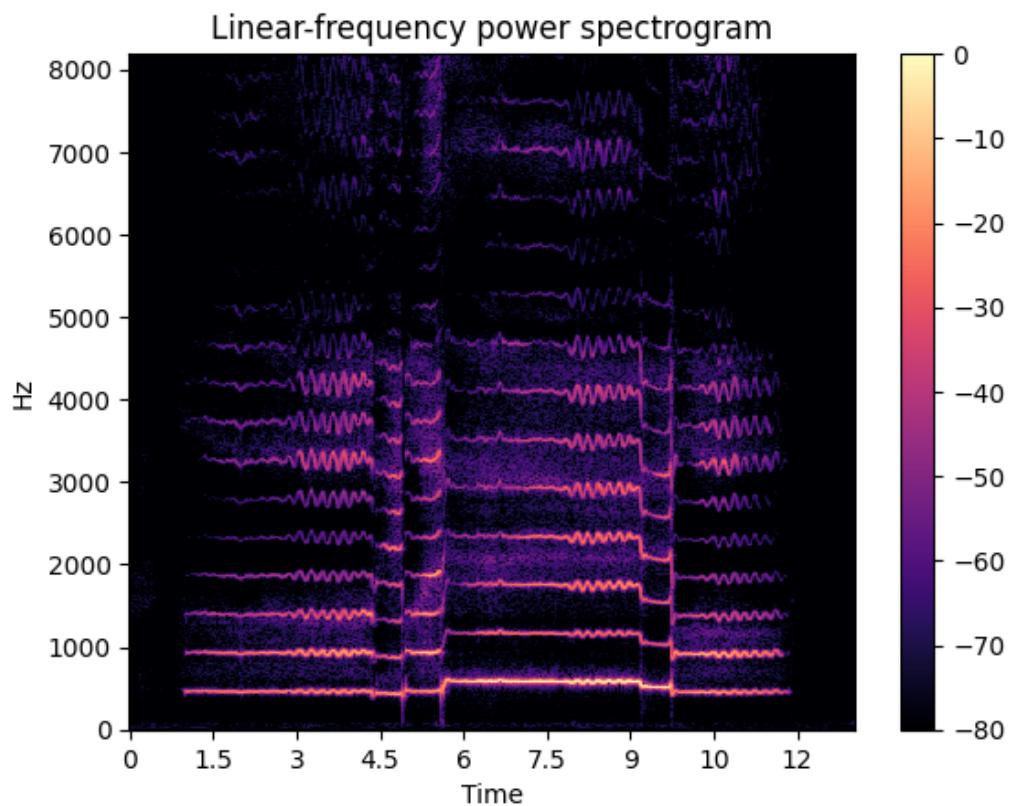
Linear- Frequency Spectrogram



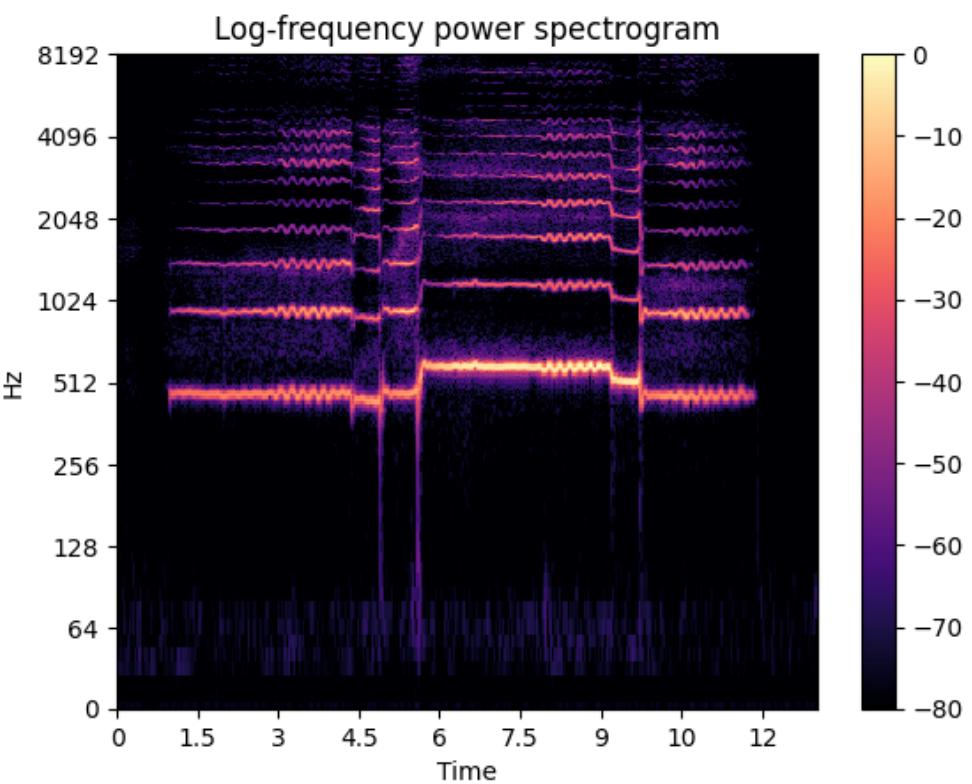
Signal Processing Basics

Log-Frequency Spectrogram

Spectrograms often default to a linear spacing of frequencies



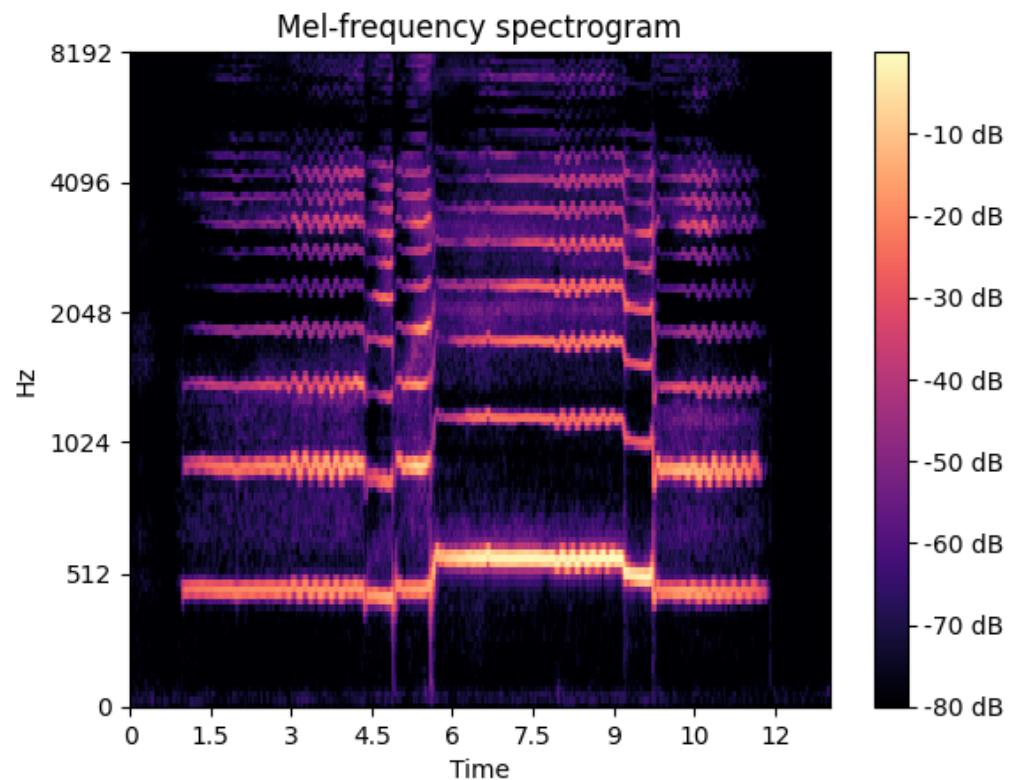
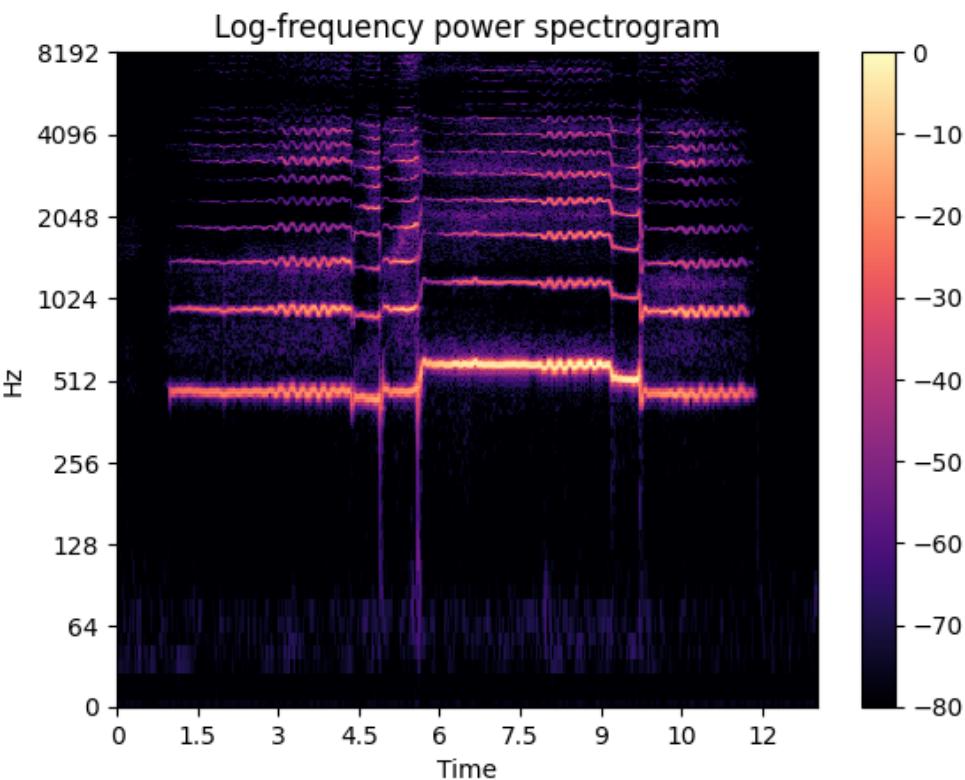
Humans hear frequencies logarithmically (a doubling of frequency is perceived as an octave)



Log-spacing in spectrograms can help observe harmonic and timbral characteristics

Signal Processing Basics

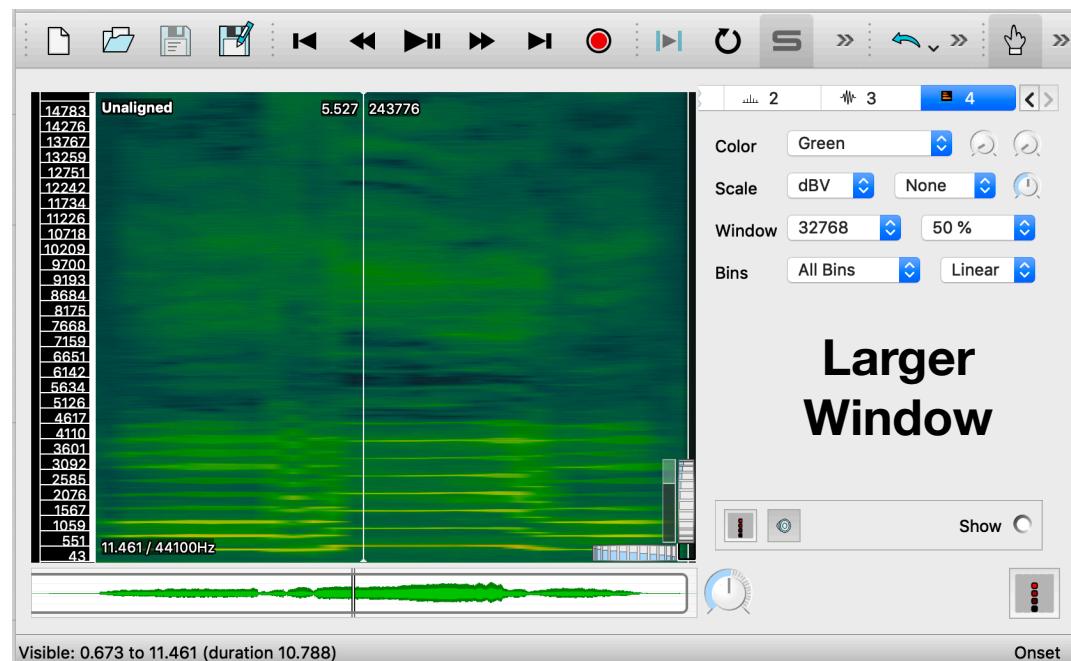
Mel Scale Spectrogram



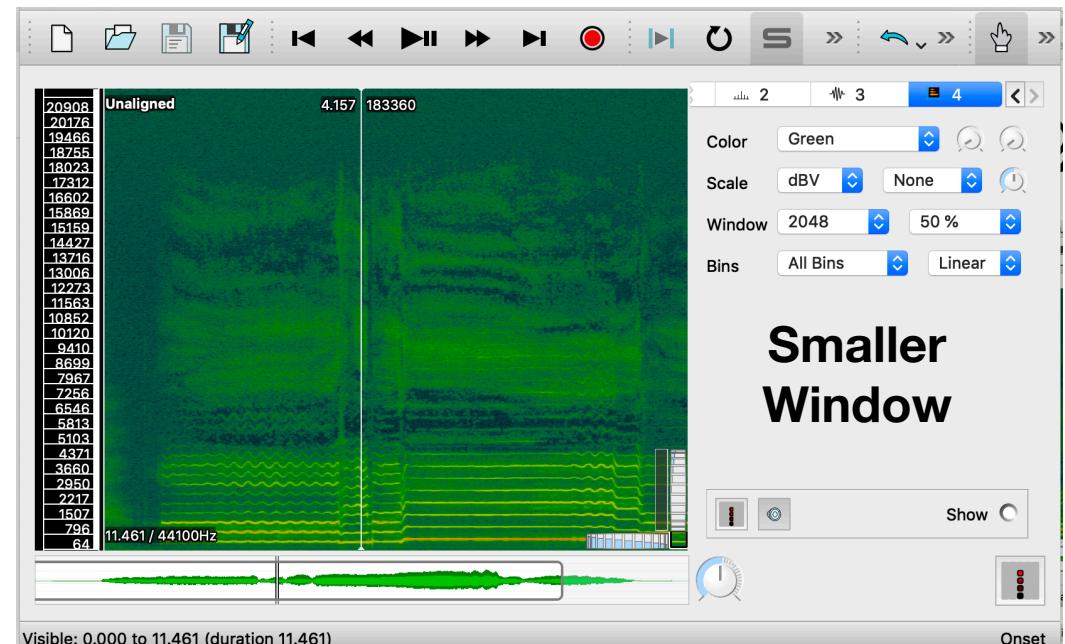
Signal Processing Basics

Spectrogram settings

A spectrogram is a series of spectral analysis (fast Fourier transforms) laid out in temporally overlapping “tiles”



Each “tile” has a window size, which determines how much of the audio signal analysed



Overview

1

Signal Processing Basics

2

Acoustic Descriptors

3

GUI-based Tools

4

Code-Based Tools

5

Summary

6

Acoustic Descriptors

Overview

- ▶ **Energy**

- Root mean square energy (RMS), Decibels (dB), shimmer

- ▶ **Pitch**

- Fundamental Frequency (F0), F0 summary descriptors (mean pitch, vibrato rate, vibrato depth, jitter)

- ▶ **Timbre**

- Spectral centroid, Spectral flux/variation, Harmonics-to-noise (HNR) ratio, Mel-frequency cepstral coefficients (MFCCs)

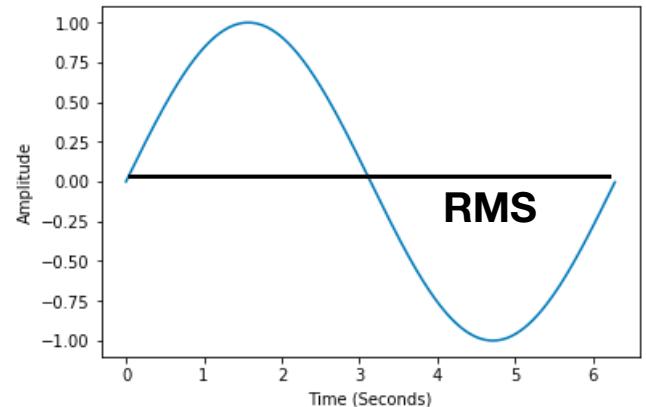
Acoustic Descriptors

Overview

	Height	Stability	Other
Energy	RMS (loud–soft)	Shimmer	
Pitch	Mean Pitch (higher–low)	Jitter	Vibrato Rate and Depth
Timbre	Spectral Centroid (bright–dark)	Spectral Flux/ Variation	Harmonic-to-Noise ratio, Mel Frequency Cepstral Coefficients

Acoustic Descriptors

Root Mean Square Energy (RMS)



- ▶ **Average amount of power in the audio signal (over time)**

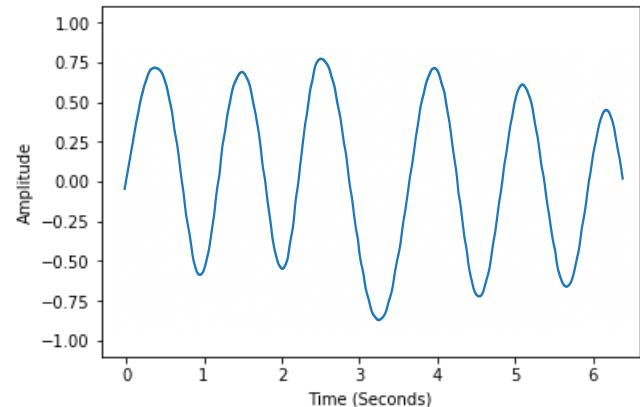
$$xRMS = \sqrt{\frac{1}{n}(x_1^2 + x_2^2 + \dots + x_n^2)}$$

Where:

- x is a series of amplitude values.
- n is the number of values in the series.

Acoustic Descriptors

Other Power-related Descriptors



- ▶ **Shimmer: period-to-period amplitude variations due to irregular vocal fold vibration**

$$\text{Shimmer} = \frac{1}{N-1} \sum_{i=1}^{N-1} \frac{|A_{i+1} - A_i|}{\frac{1}{N} \sum_{j=1}^N A_j}$$

Where:

- A_i is the peak amplitude of the i -th period,
- N is the total number of periods.

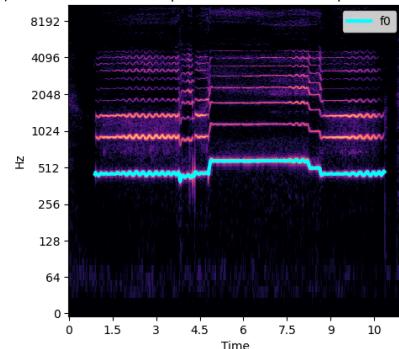
$$\text{Shimmer}_{\text{dB}} = \frac{1}{N-1} \sum_{i=1}^{N-1} 20 \log_{10} \left(\frac{A_{i+1}}{A_i} \right)$$

Where:

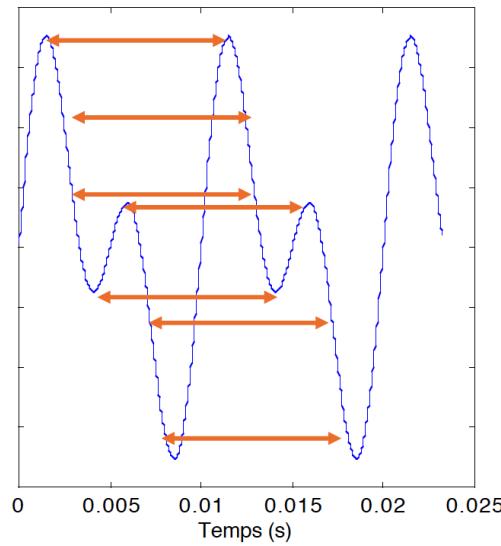
- A_i is the peak amplitude of the i -th period,
- N is the total number of periods.

Acoustic Descriptors

Fundamental Frequency (F0)

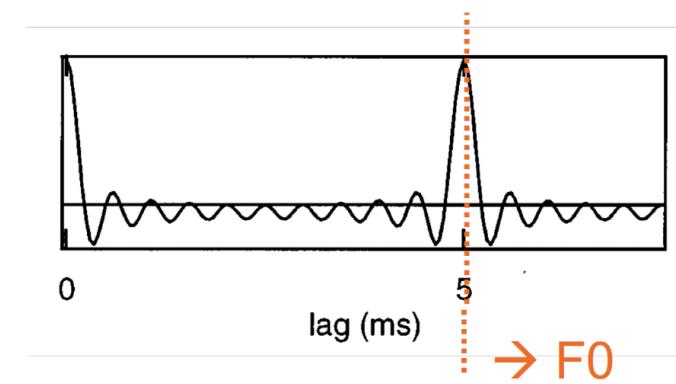


- ▶ **Lowest frequency component of a complex sound**
- ▶ **Autocorrelation approach:**



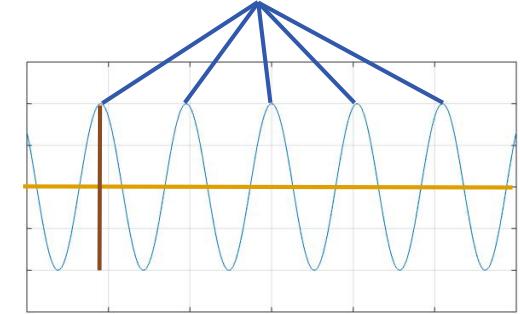
$$A_t(\tau) = \sum_W x(t)x(t - \tau)$$

"shift and compare"
product
time lag
integration window



Acoustic Descriptors

F0-related Descriptors



- ▶ **Mean pitch: simple or weighted average of F0**
- ▶ **Vibrato: regular, pulsating changes in F0**
- ▶ **Jitter: period-to-period F0 variations due to irregular vocal fold vibration**

$$\text{Jitter} = \frac{1}{N-1} \sum_{i=1}^{N-1} \frac{|T_{i+1} - T_i|}{\frac{1}{N} \sum_{j=1}^N T_j}$$

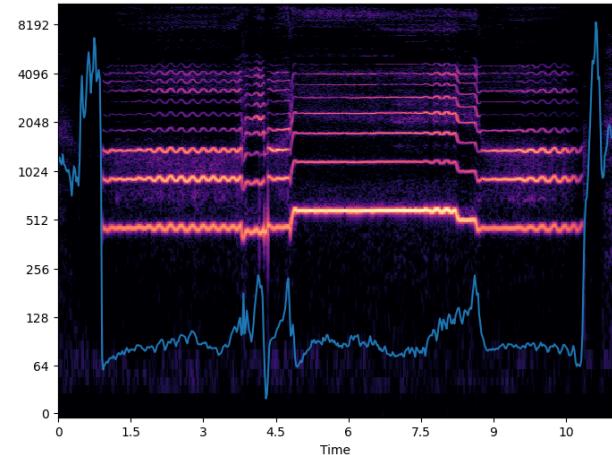
Where:

- T_i is the duration (period) of the i -th pitch cycle,
- N is the total number of cycles.

Acoustic Descriptors

Spectral Centroid

- ▶ **Spectral center of gravity**
- ▶ **Relates to perceptual sense of brightness**
- ▶ **Weighted average of frequencies present in signal (weighted by power)**



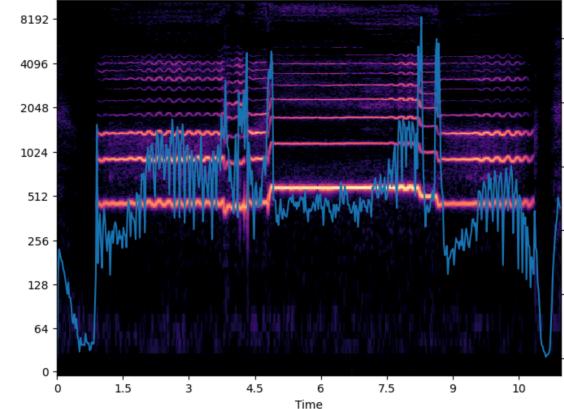
$$\text{Spectral Centroid} = \frac{\sum_{k=0}^{K-1} f_k \cdot X(k)}{\sum_{k=0}^{K-1} X(k)}$$

Where:

- f_k is the frequency (in Hz) of the k -th bin,
- $X(k)$ is the magnitude of the Fourier transform at bin k ,
- K is the total number of frequency bins.

Acoustic Descriptors

Spectral Variation/Flux



- ▶ **Changes in the spectrum over time**
- ▶ **Can be estimated by calculating**
 - Euclidian distance ("as the crow flies") between spectral frames
 - Cosine similarity between spectral frames

$$\text{Spectral Flux}_{\text{Euclidean}} = \sum_{k=0}^{K-1} (X_t(k) - X_{t-1}(k))^2$$

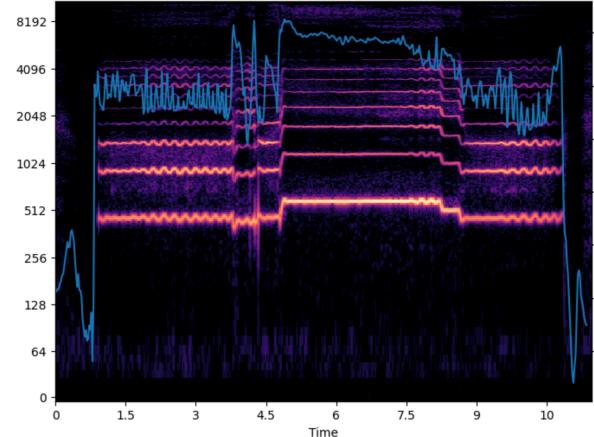
Where:

- $X_t(k)$ is the magnitude of the Fourier transform at bin k in frame t ,
- $X_{t-1}(k)$ is the magnitude of the Fourier transform at bin k in the previous frame,
- K is the total number of frequency bins.

Acoustic Descriptors

Harmonics-to-Noise Ratio

- ▶ **Relative amount of noise in a signal**
- ▶ **Ratio of harmonic energy to noise in a periodic signal**



$$\text{TotalEnergy}(E_T(t_m)) = \sum_{k=1}^K x_k^2(t_m)$$

$$\text{HarmonicEnergy}(E_H(t_m)) = \sum_{h=1}^H a_h^2(t_m)$$

$$\text{NoiseEnergy}(E_N(t_m)) = E_T(t_m) - E_H(t_m)$$

$$\text{Noisiness}(t_m) = \frac{E_N(t_m)}{E_T(t_m)}$$

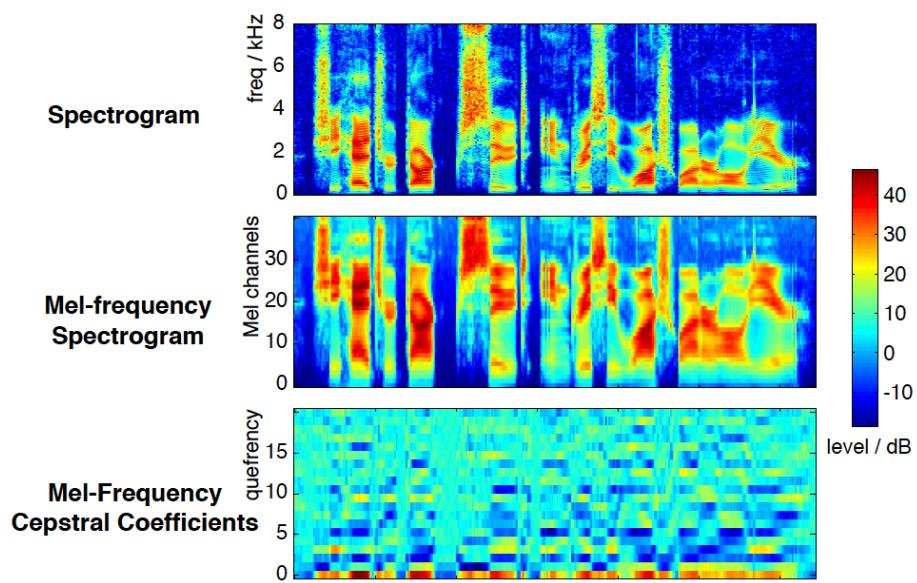
Where:

- $x_k(t_m)$ is the amplitude at the k -th frequency bin,
- $a_h(t_m)$ is the amplitude of the h -th harmonic partial,
- K is the number of frequency bins,
- H is the number of harmonic partials.

Acoustic Descriptors

Mel-Frequency Cepstral Coefficients (MFCCs)

- ▶ **Summary description of spectral envelope**
- ▶ **Useful for highlighting formants**
- ▶ **Can be hard to interpret**



Recipe

- ▶ Calculate the Short Time Fourier Transform (STFT, which we covered last class in our discussion of spectrograms)
- ▶ Convert the spectrum amplitudes to the log scale
- ▶ Map the log frequencies of the spectrum obtained from the STFT onto the mel scale, using an overlapping window function to smooth the signal
- ▶ Take the Discrete Cosine Transform (DCT) of the mel log-amplitudes



Example Analysis

Pitch and Timbre in Rihanna's *Anti* album

► **Vocal Production**

- “But this particular album, she was really, really specific about every single line. If there was anything that she heard that she felt wasn't perfection in her eyes, she wanted to change. It was a lot, and I respect that. It's hard, because I want to work quick and get stuff done and make it excellent, but now she's like, ‘Listen, let's up the quality level.’” Kuk Harrell



Example Analysis

Pitch and Timbre in Rihanna's *Anti* album

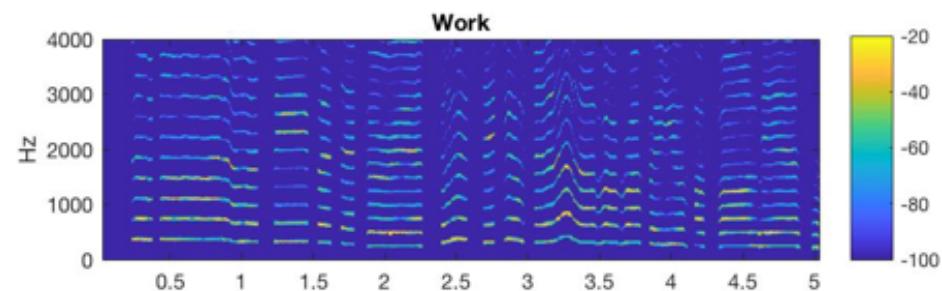
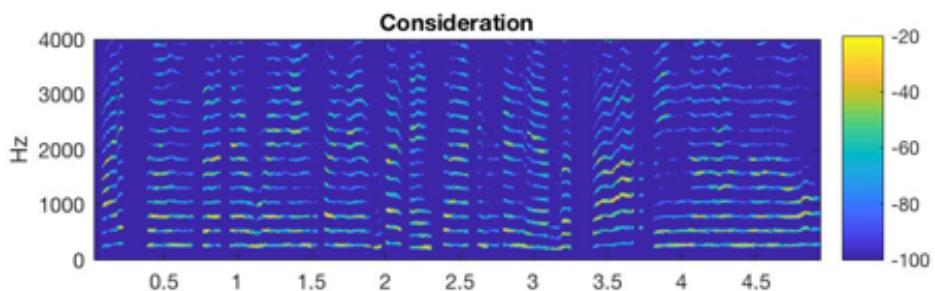
- *New York Times* declared “Rihanna, Blissfully Adrift, Juggles Styles on ‘Anti’”
- *Pitchfork* noted “her grainy, mesmerizing voice is paramount here, the sun in *ANTI*’s universe, the thing everything else orbits”
- *Independent* argued that “this album shows Rihanna hitting back at anyone who ever said her voice could only do certain things and showing them she can do anything she wants to”
- *Rolling Stone* lauded her “show-stopping vocal performance”



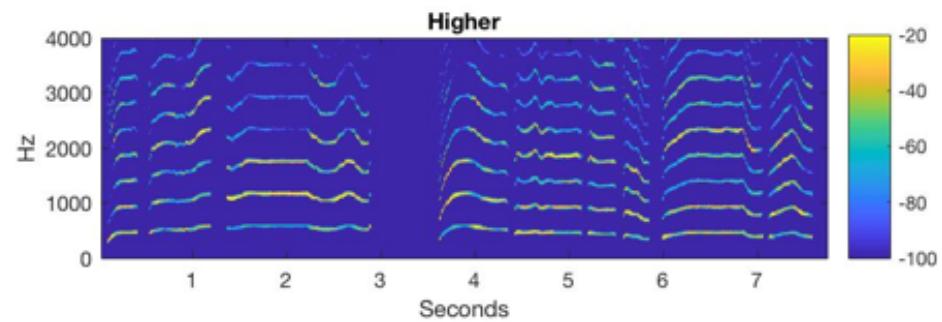
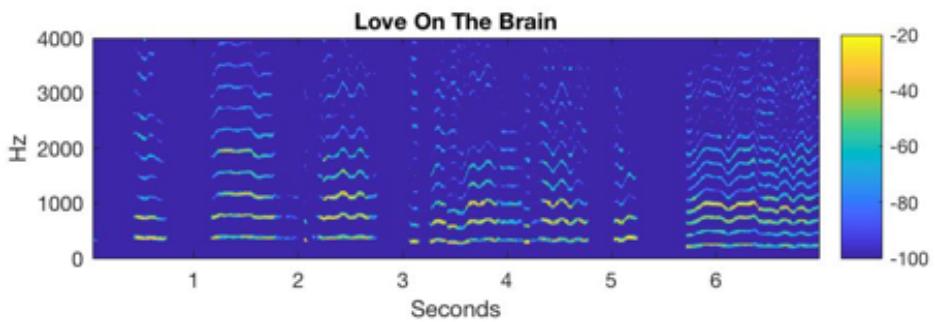
Example Analysis

Pitch and Timbre in Rihanna's *Anti* album

Speech-Like



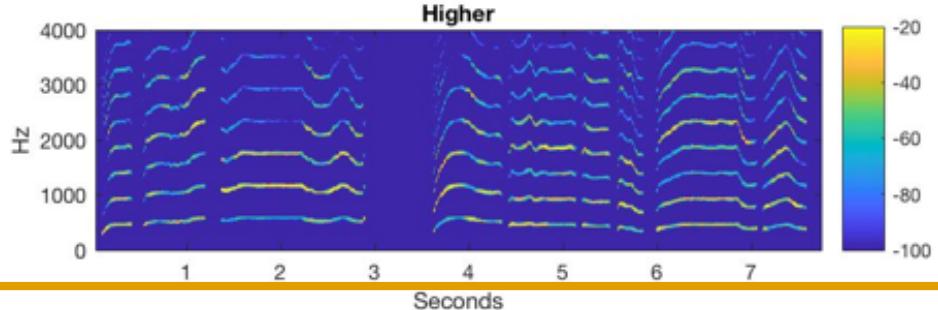
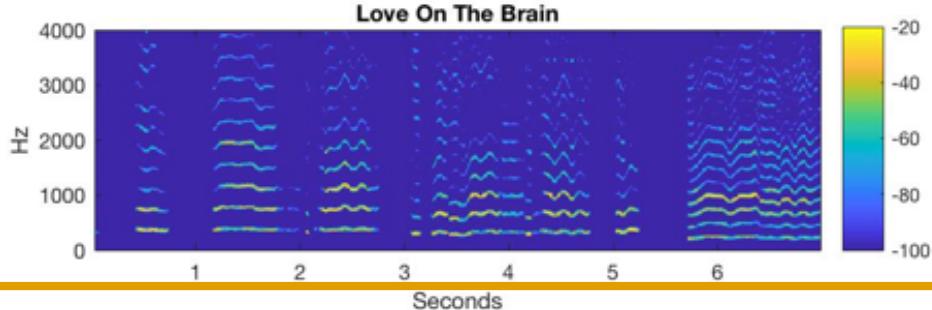
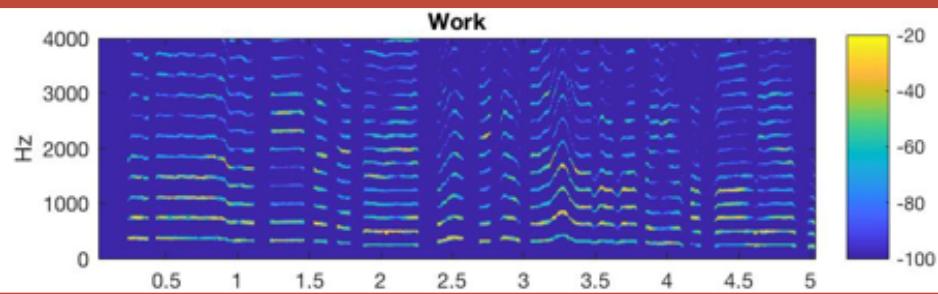
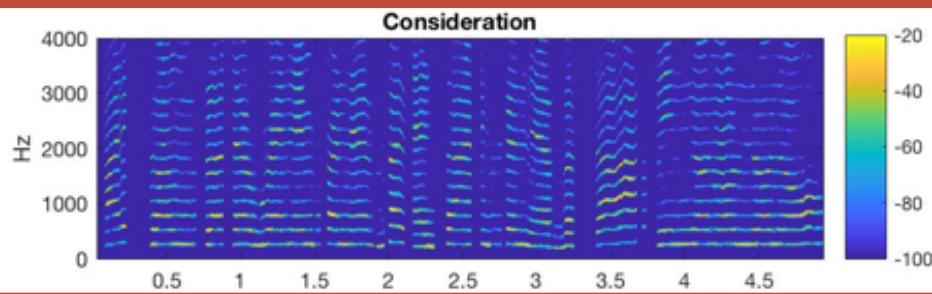
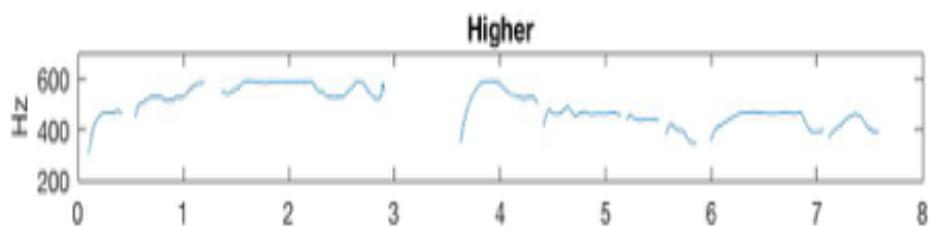
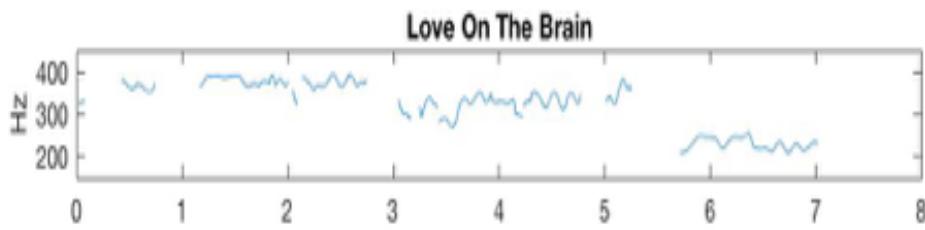
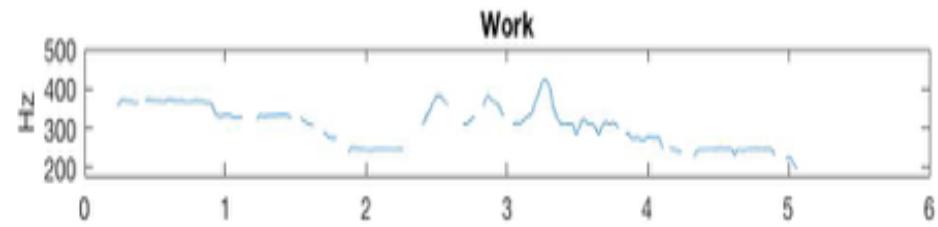
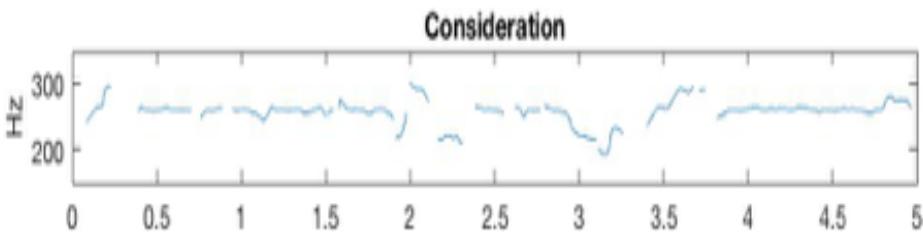
Emotional/Raw



Example Analysis

Pitch and Timbre in Rihanna's *Anti* album

Speech-Like Emotional/Raw



Example Analysis

Pitch and Timbre in Rihanna's *Anti* album



	Measure of Relative Height	Measures of Stability/Instability
Pitch	Mean Pitch <i>average fundamental frequency (F_0)</i>	Jitter <i>average of frame-to-frame changes in F_0</i>
Spectrum	Normalized Spectral Centroid <i>pitch-normalized average of spectral energy</i>	Spectral Flux <i>average of frame-to-frame changes in spectral energy</i>

Overview

1

Signal Processing Basics

2

Acoustic Descriptors

3

GUI-based Tools

4

Code-Based Tools

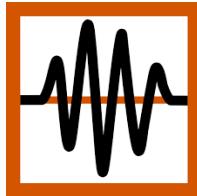
5

Summary

6

GUI-Based Tools

Overview

- ▶ **Audacity** 
- ▶ **Sonic Visualiser** 
- ▶ **Tony** 



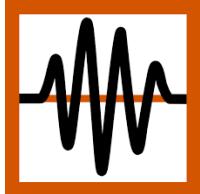
Audacity

Overview

- ▶ **Digital audio recording and editing software**
- ▶ **Can be used to visualise and annotate audio**
- ▶ **Can run a range of plug-in formats**

Sonic Visualiser

Overview



- ▶ **Digital audio analysis software**
- ▶ **Can be used to visualise and annotate audio**
- ▶ **Can run Vamp plug-ins**

Tony

Overview

TONY

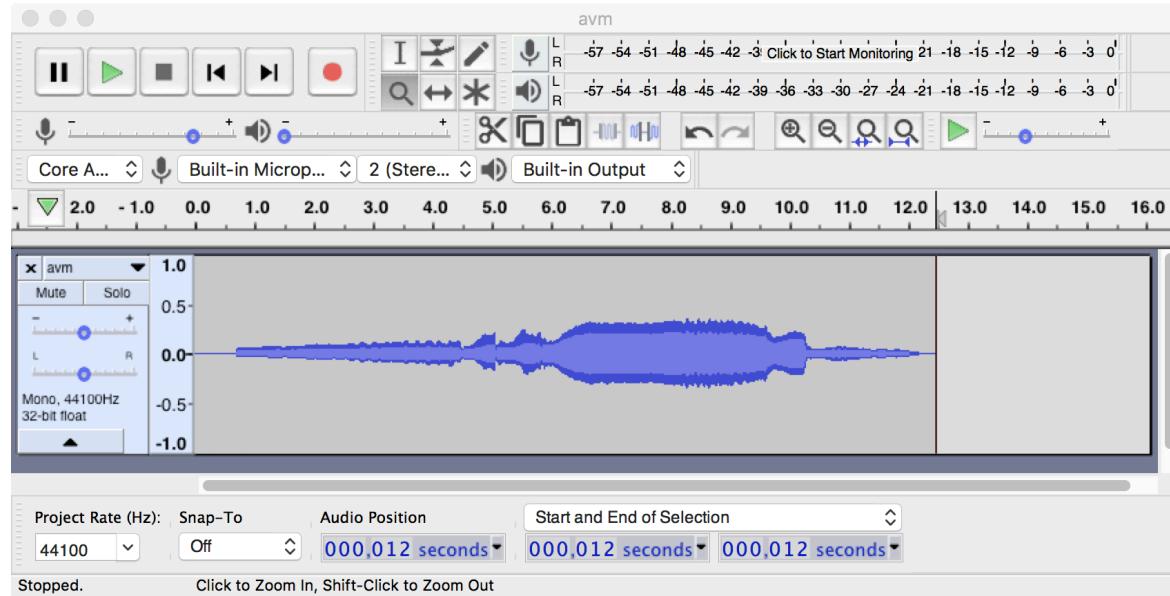
- ▶ **Melody annotation software**
- ▶ **Automatically estimates note onsets and offsets and fundamental frequency**
- ▶ **Automatically estimates can be manually adjusted**

Audacity

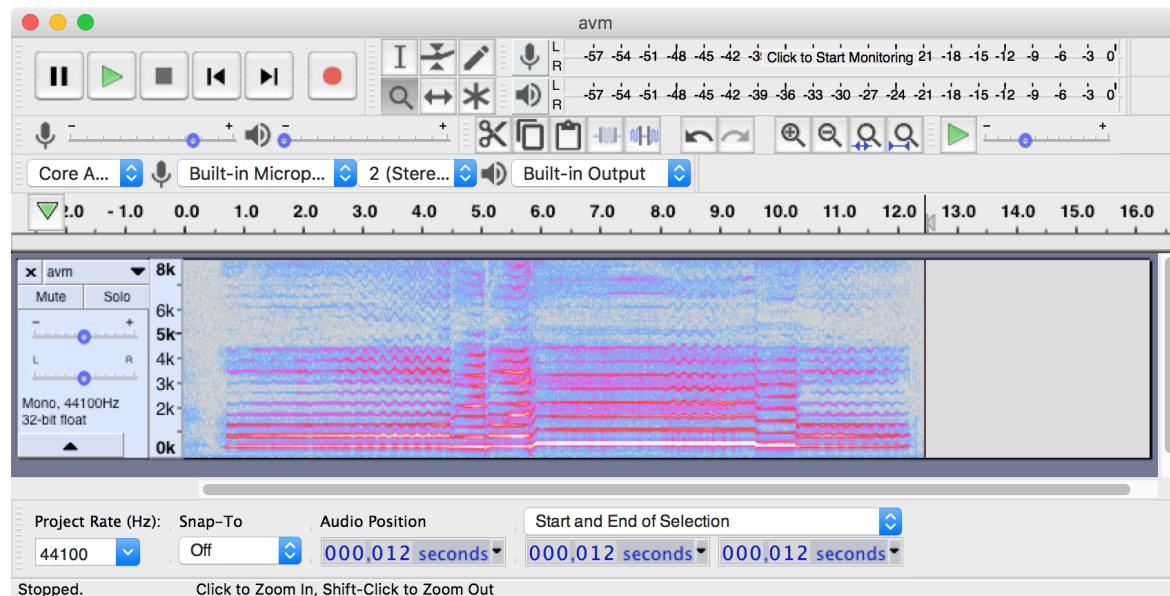


Representations

Time-
domain

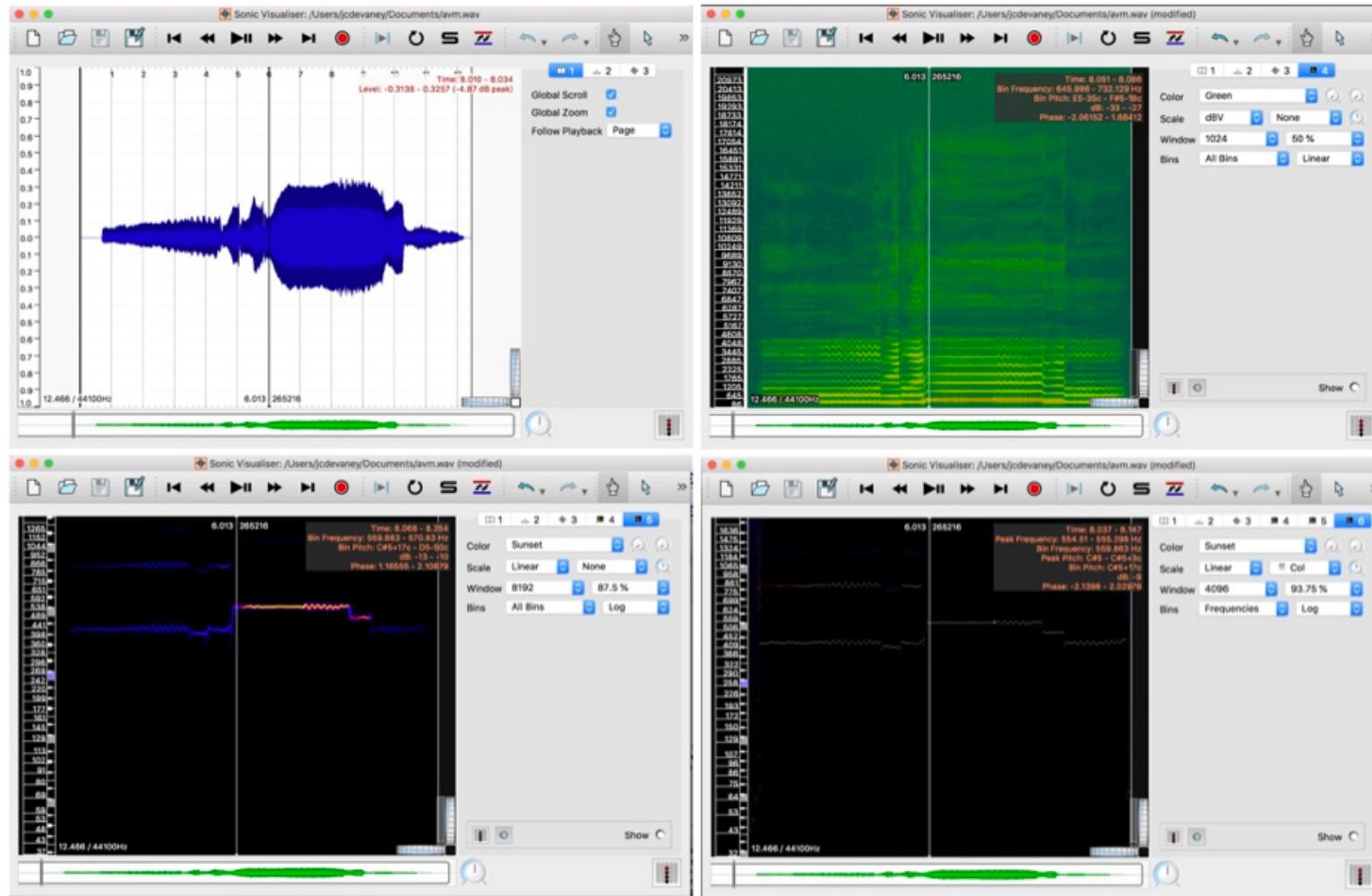
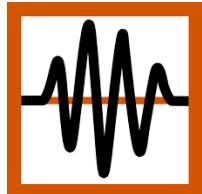


Frequency-
domain



Sonic Visualiser

Representations



The top-left panel is a time-domain waveform, the top-right panel is a spectrogram, the bottom-left panel is a melody-range spectrogram, and the bottom-right spectrogram is a peak-frequency spectrogram.

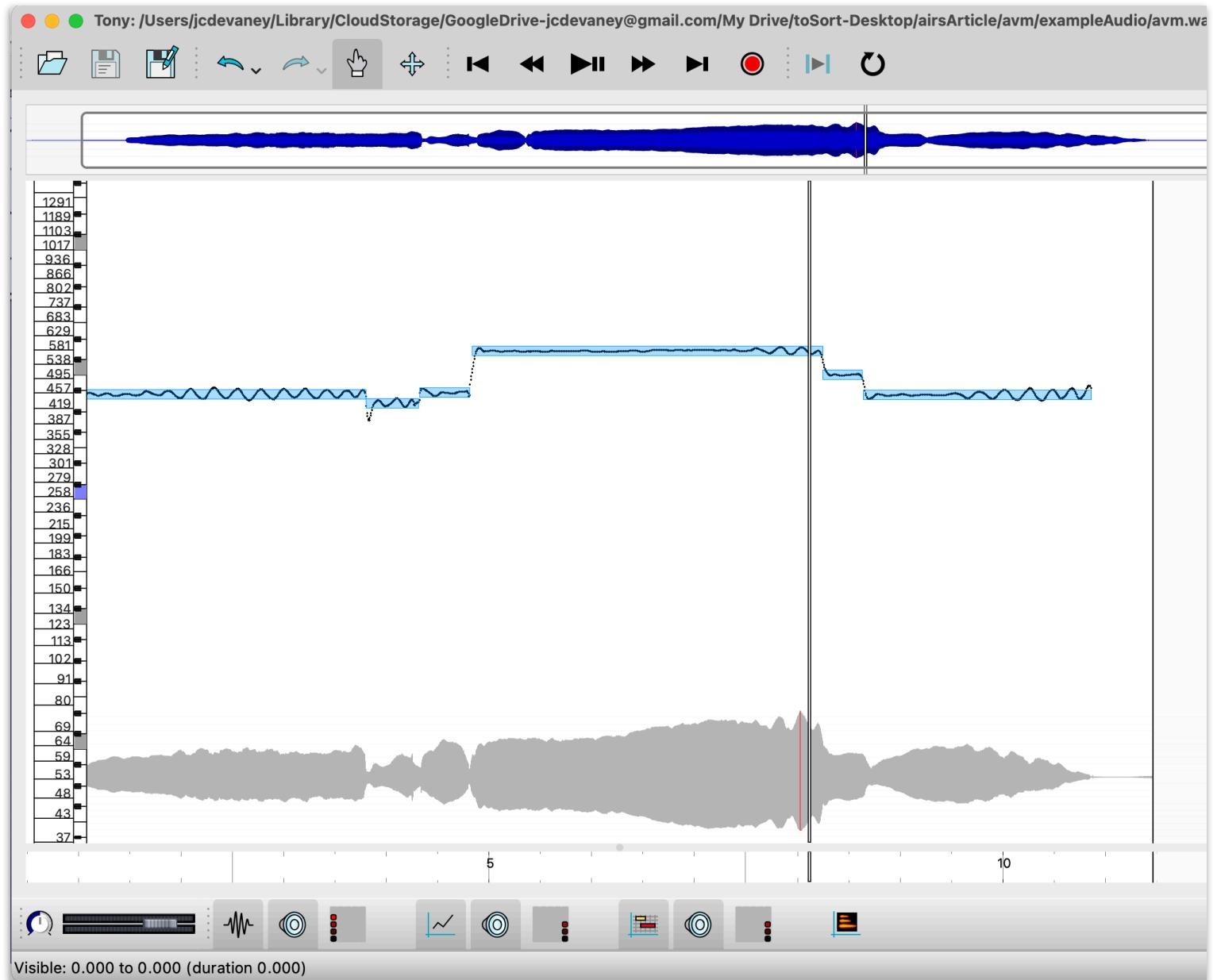
Tony

Representations

TONY

Frequency-domain

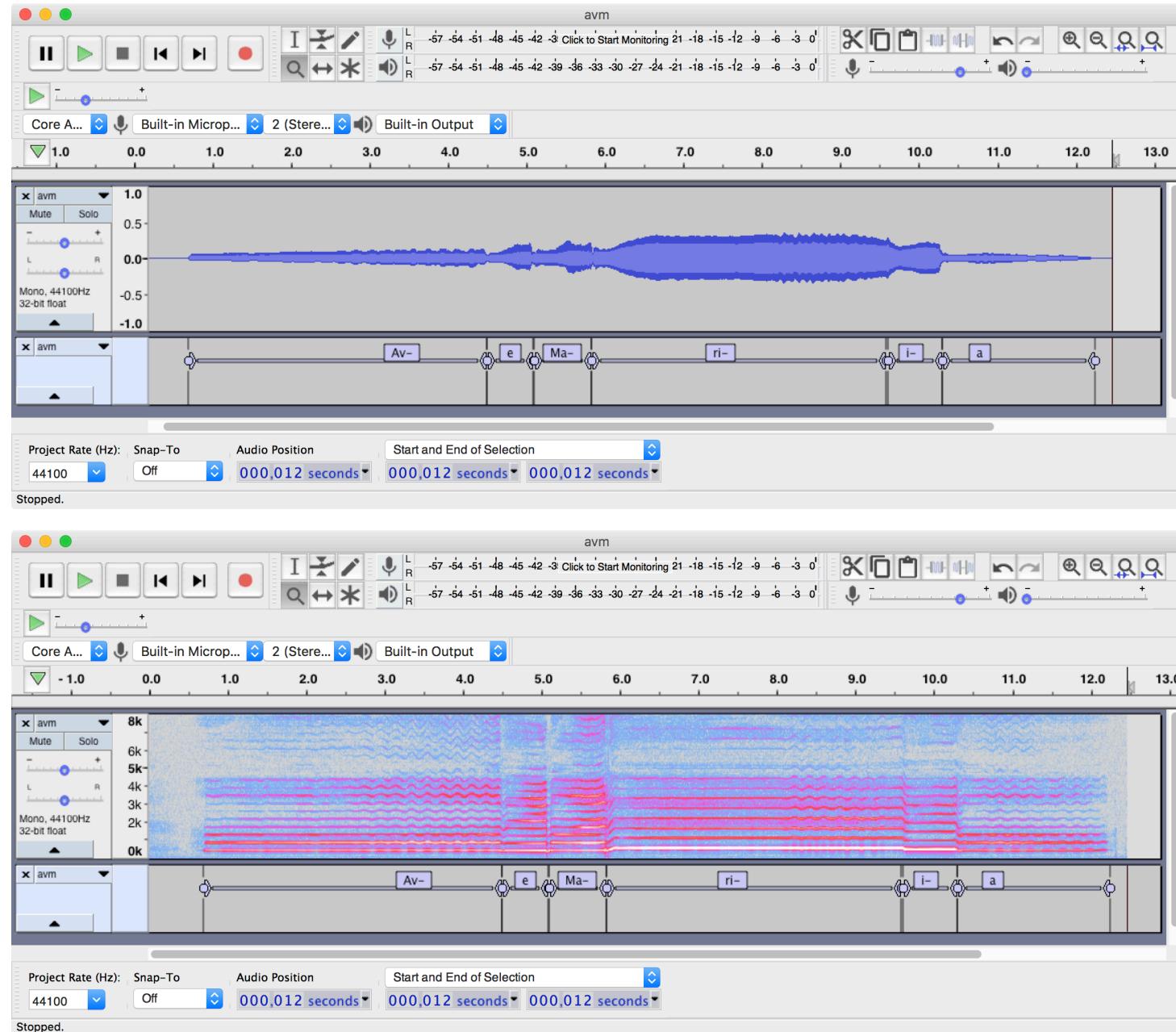
Time-domain





Audacity

Annotations



Audacity

Annotations



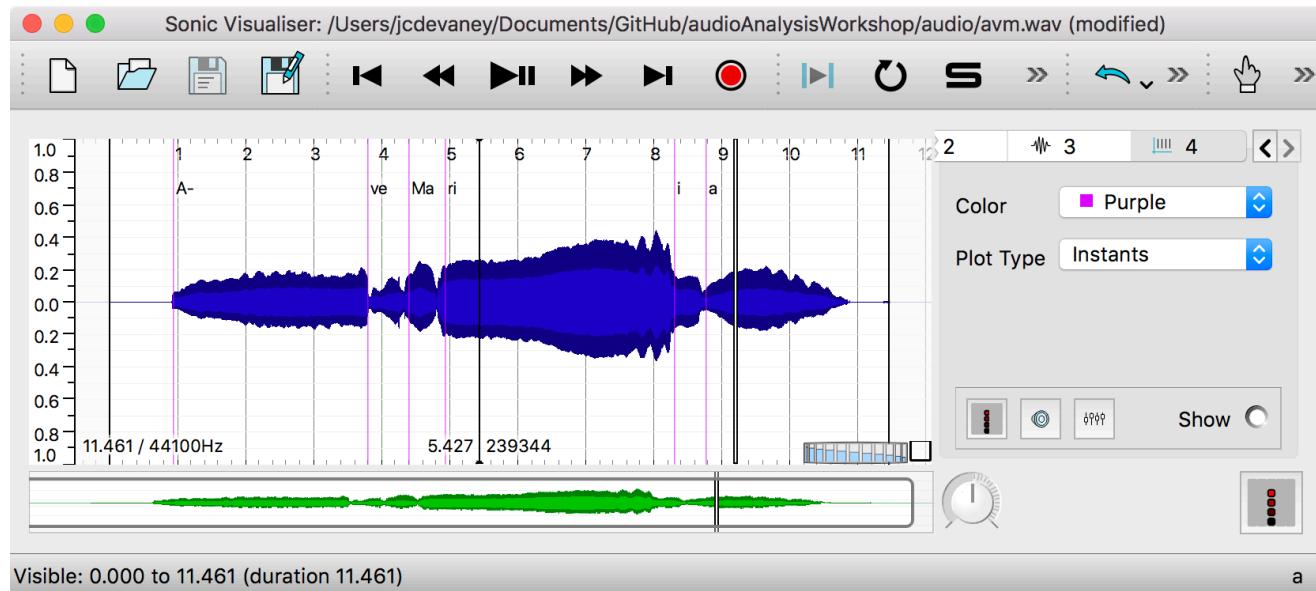
The screenshot shows the Audacity application window. The menu bar includes File, Edit, Select, View, Transport, Tracks, Generate, Effect, Analyze, Window, and Help. The toolbar contains various editing tools like selection, cut, copy, paste, and zoom. The main workspace displays an audio waveform for a file named "avm". The transport controls at the top center include play, stop, and record buttons. Below the waveform is a track list showing a single track named "avm" containing several labeled regions: "Av-", "e", "Ma-", "ri-", "i-", and "a". The bottom of the window shows project settings: Project Rate (Hz) set to 44100, Audio Position set to 000,000 seconds, and a status message indicating the project is stopped.

Sonic Visualiser

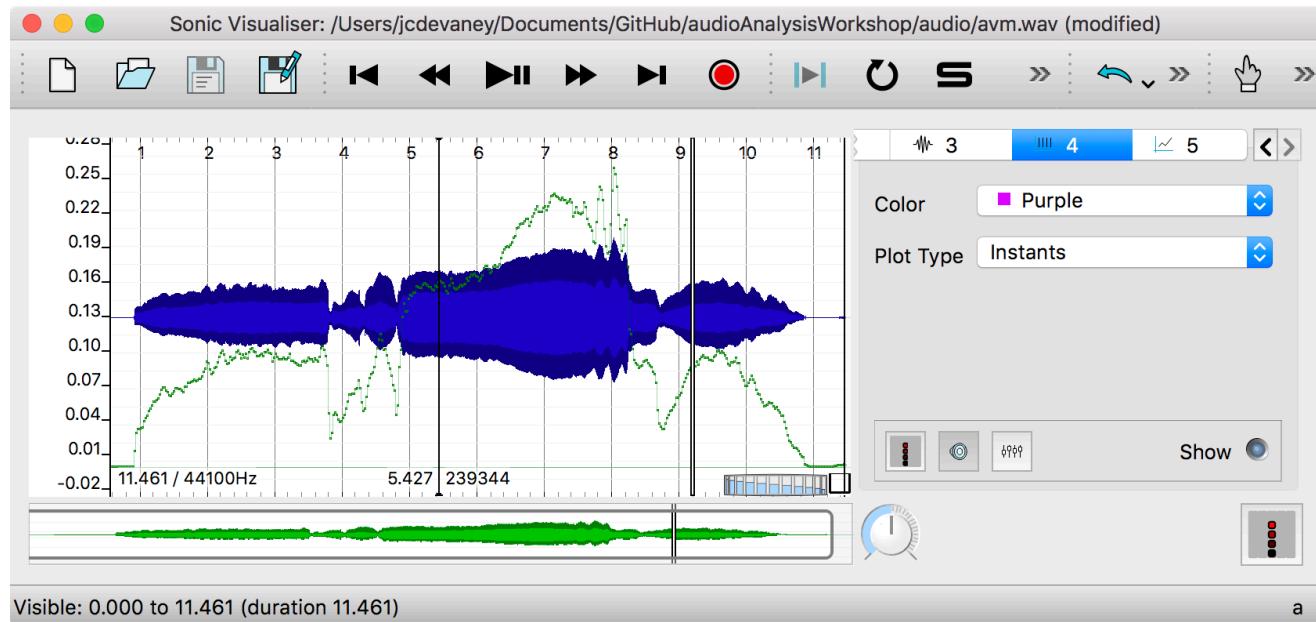


Annotations

Time Instants
(time points)



Time Values
(time points + values)

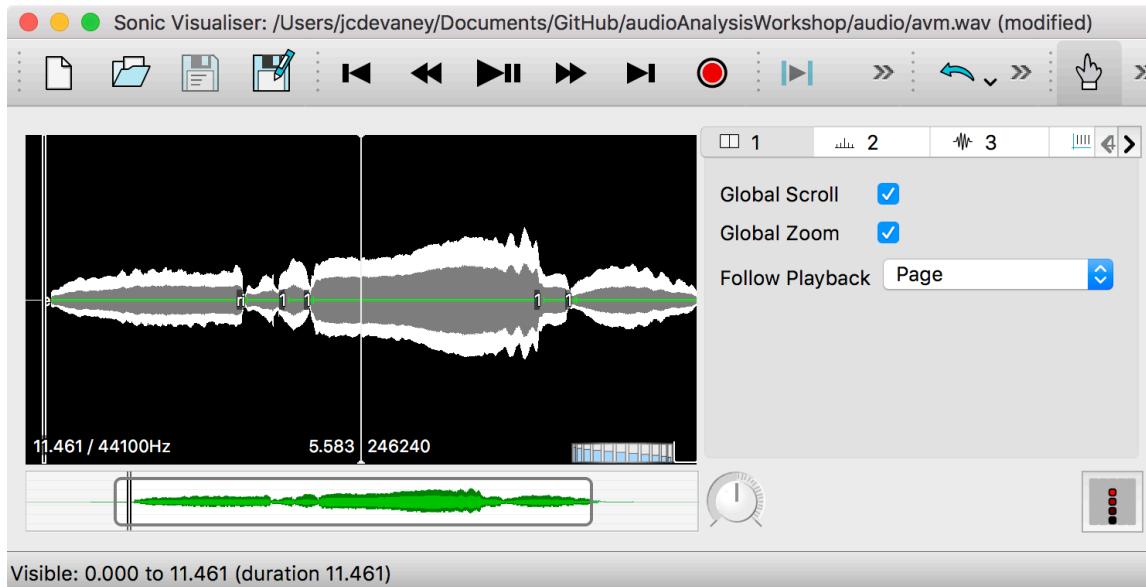


Sonic Visualiser

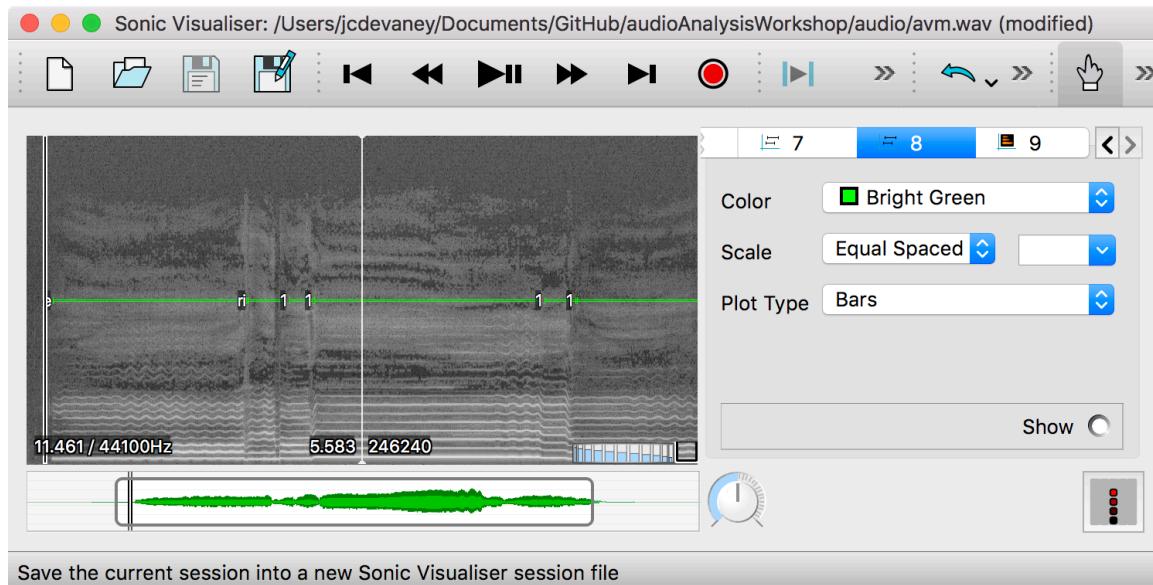


Annotations

Regions
(Time-domain
overlay)



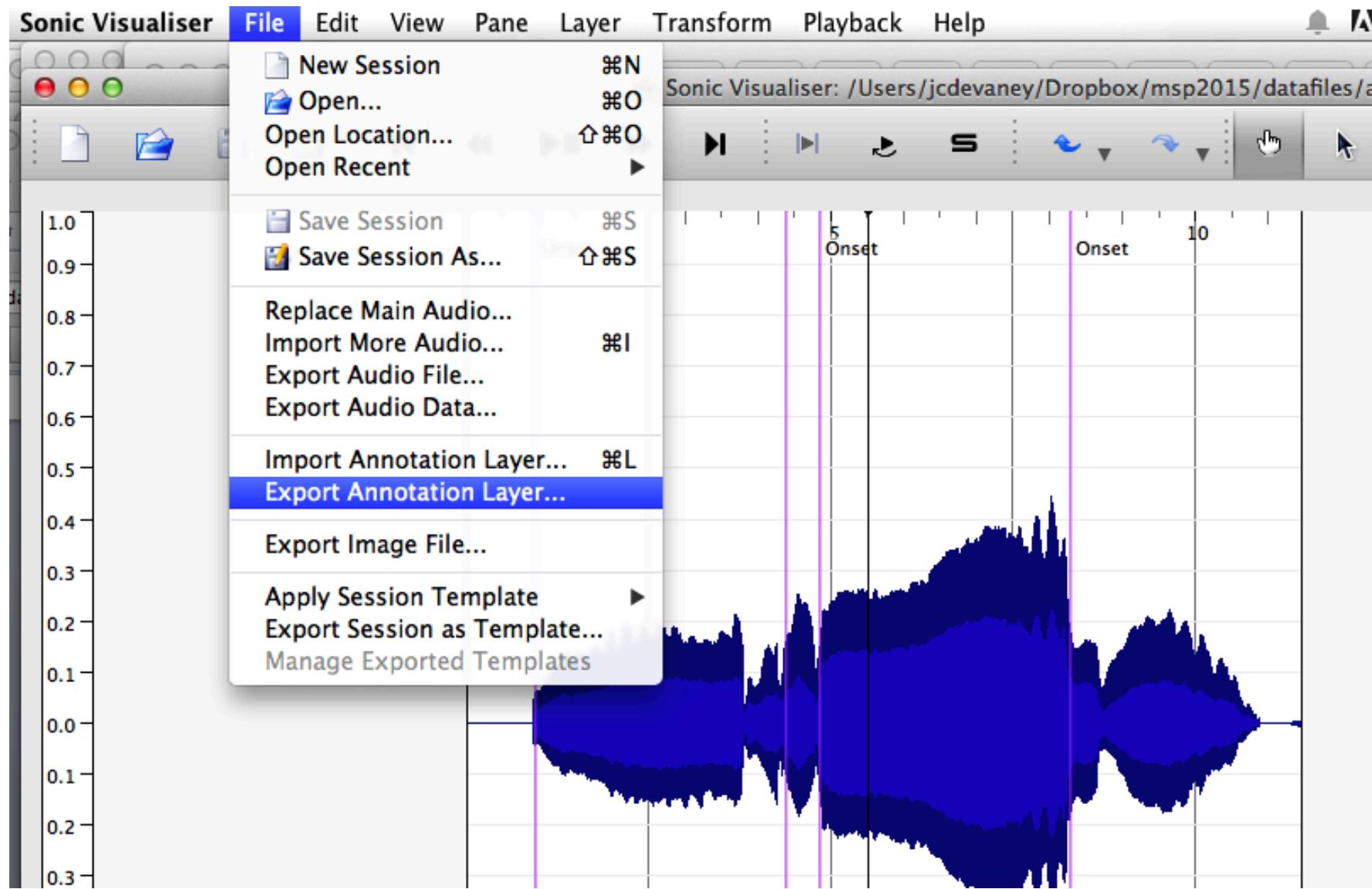
Regions
(Frequency-
domain
overlay)



Sonic Visualiser



Annotations

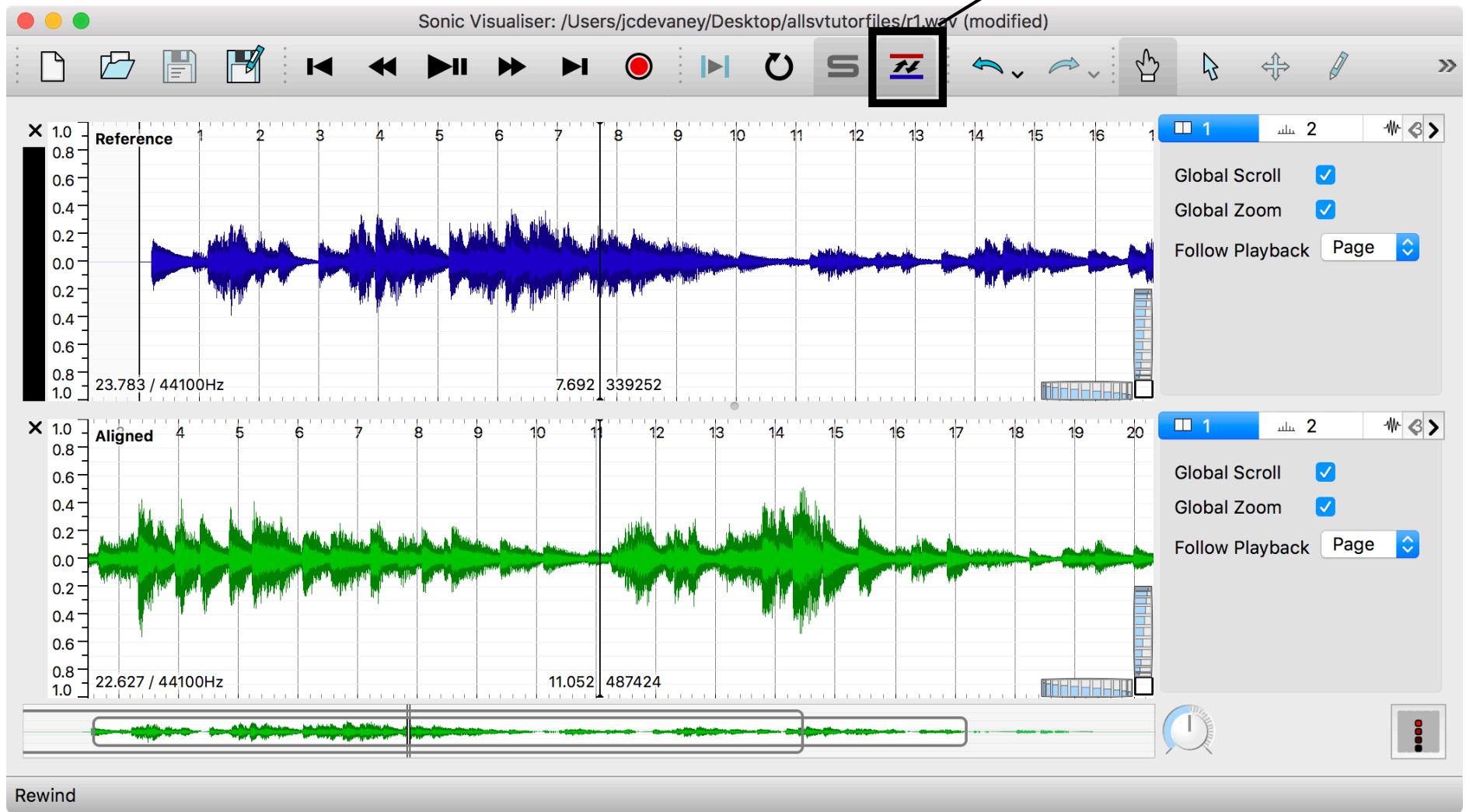


Sonic Visualiser



Time Alignment

Attempts to temporally align two audio files

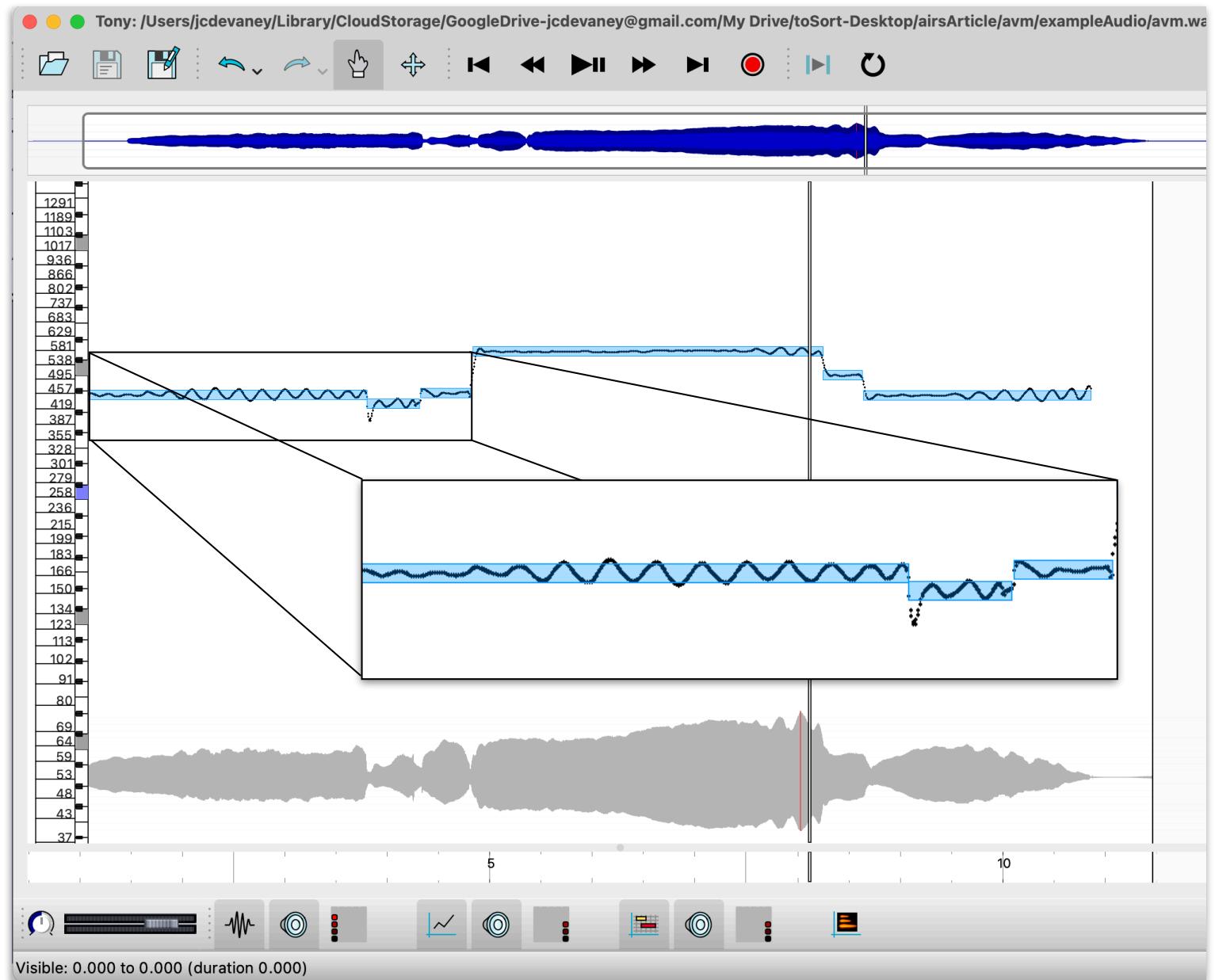


Tony

Annotations

TONY

Automatically note segmentation estimation can be manually adjusted

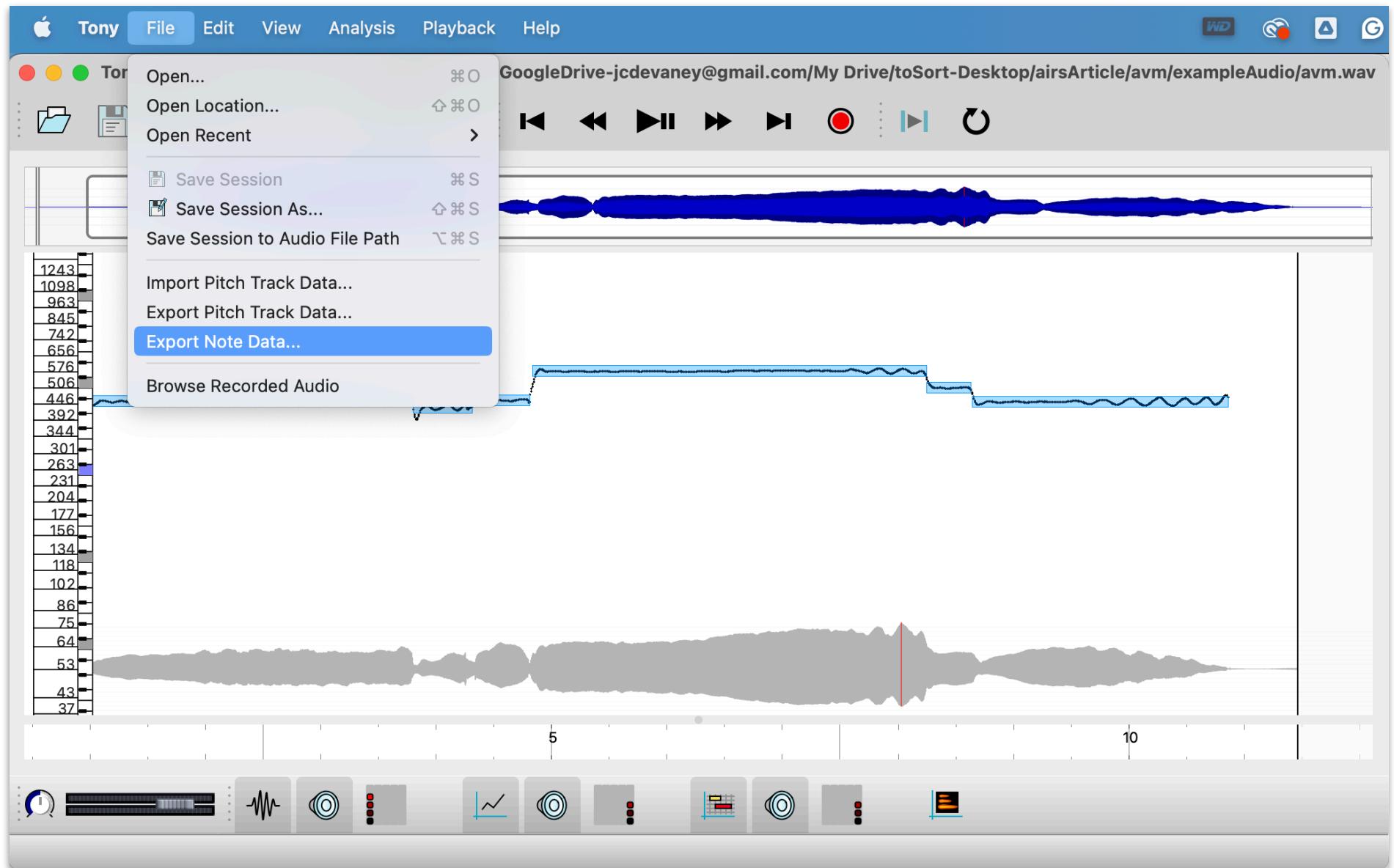


Visible: 0.000 to 0.000 (duration 0.000)

Tony

Annotations

TONY



GUI-Annotation Exercise

Audacity, Sonic Visualiser, and Tony

- ▶ **Open up `avm.wav` in each program**

- In Audacity annotate labels for each note and export the labels to a text file (File > Save Other > Export Labels...)
- In Sonic Visualiser annotate both the time instants (onsets) and regions (onsets and offsets for each note), and export both set of labels (File > Export Annotation Layer...)
- In Tony examine the automatic note estimation and adjust as needed, then export the labels (File > Export Note Data...)

Onset Detection

Theory

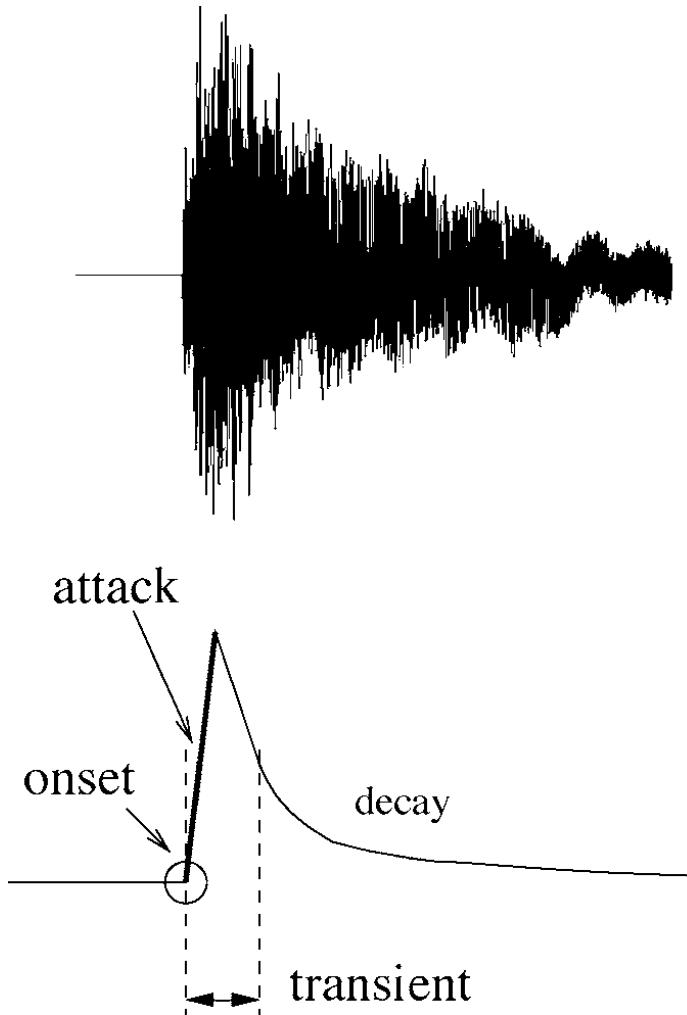


Fig. 1. “Attack,” “transient,” “decay,” and “onset” in the ideal case of a single note.

Onset Detection

Theory

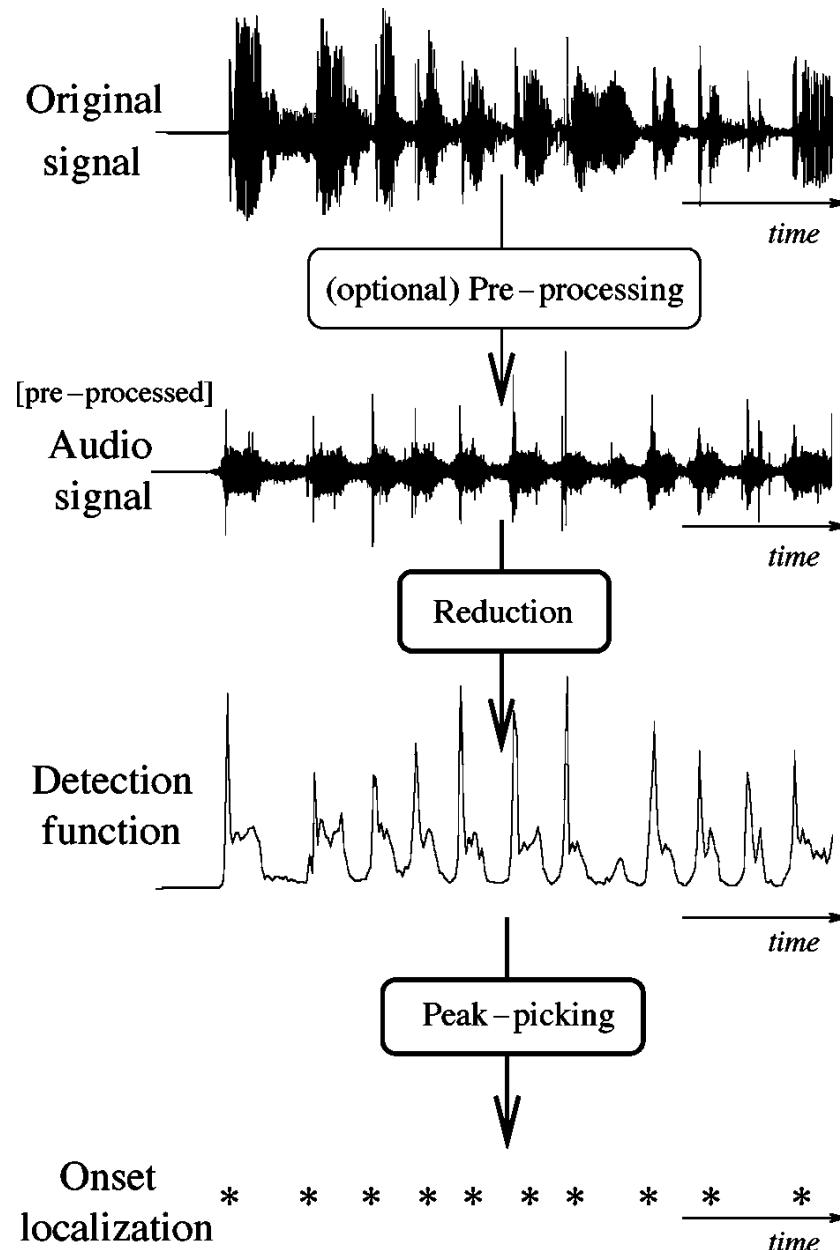


Fig. 2. Flowchart of a standard onset detection algorithm.

Onset Detection

Theory

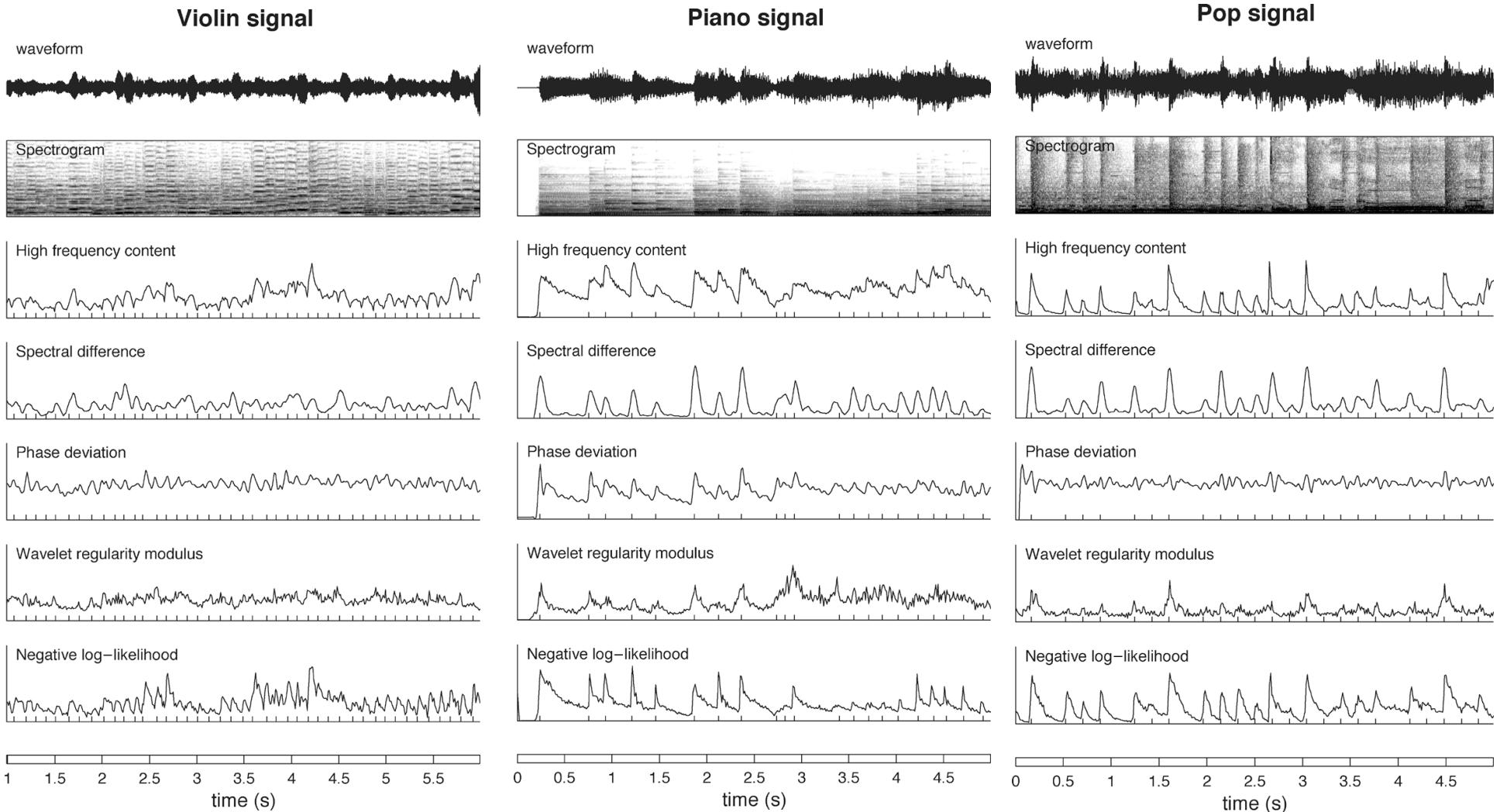


Fig. 4. Comparison of different detection functions for 5 s of a solo violin recording. From top to bottom: time-domain signal, spectrogram, high-frequency content, spectral difference, spread of the distribution of phase deviations, wavelet regularity modulus, and negative log-likelihood using an ICA model. All detection functions have been normalized to their maximum value.

Fig. 5. Comparison of different detection functions for 5 s of a solo piano recording. From top to bottom: time-domain signal, spectrogram, high-frequency content, spectral difference, spread of the distribution of phase deviations, wavelet regularity modulus, and negative log-likelihood using an ICA model. All detection functions have been normalized to their maximum value.

Fig. 6. Comparison of different detection functions for 5 s of a pop song. From top to bottom: time-domain signal, spectrogram, high-frequency content, spectral difference, spread of the distribution of phase deviations, wavelet regularity modulus, and negative log-likelihood using an ICA model. All detection functions have been normalized to their maximum value.

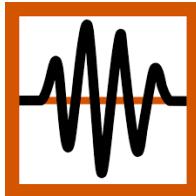
Onset Detection



Sonic Visualiser

- ▶ **University of Alicante: signal approach to reduction**

- pre-processing one-semitone filter bank
- outputs
 - onset detection function
 - note onset estimates
- parameters
 - sensitivity
 - processing option: audio frames per block (window size), window increment (hop)

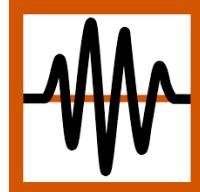


Onset Detection

Sonic Visualiser

- ▶ **Queen Mary: probabilistic approach to reduction**
 - outputs
 - onset detection function
 - smoothed detection function
 - note onset estimates
 - parameters
 - program: general, soft onsets, percussive onsets
 - function type: high-frequency content, spectral difference, phase deviation, complex domain, broadband energy rise
 - processing options: window size, window increment (hop), window shape (Hann, Hamming, Blackman, Blackman-Hanning, Nuttall, Gaussian, Parzen, triangular, rectangular)

GUI-Onset Detection Exercise



Sonic Visualiser Plugins

- ▶ **Open up `kingsLoop.wav`, `chopinSeg.wav`, `avm.wav` and run the following onset plugins**
- University of Alicante (Onset Detection Function and Note Onsets) and play to hear onset locations
- Queen Mary, University of London (Onset Detection Function and Note Onsets) and play to hear onset locations)
- ▶ **Export the labels (File > Save Other > Export Labels...)**

The screenshot shows the Sonic Visualiser interface with the Transform menu open. The 'Analysis by Maker' option is highlighted. A submenu lists several makers:

- BBC
- MIR.EDU by Justin Salamon
- Matthias Mauch
- Music Technology Group, Universitat Pompeu Fabra
- Queen Mary, University of London
- Simon Dixon
- University of Alicante

Overview

1

Signal Processing Basics

2

Acoustic Descriptors

3

GUI-based Tools

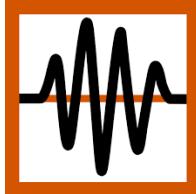
4

Code-Based Tools

5

Summary

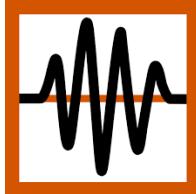
6



Sonic Annotator

Overview

- ▶ **For each plugin you will need to create a .n3 file, using the -s flag to specify a plugin. This allows you to batch process audio files with a particular plugin with particular settings (although for this assignment just use the default values)**
 - E.g., `./sonic-annotator -s vamp:bbc-vamp-plugins:bbc-energy:rmsenergy > rms.n3`



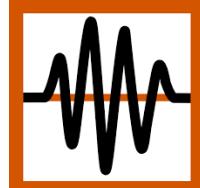
Sonic Annotator

Overview

- ▶ You will then need to put all of the audio file in the same directory and run each .n3 file on the directory using the -r flag to specify the directory, -t flag to specify the .n3 file, and the -w flag to specify csv (which writes to a CSV file)
 - E.g., ./sonic-annotator -r ~/Documents/GitHub/toss2025/audio -t rms.n3 -w csv

Sonic Annotator Exercise

Export Plugin-Estimated Data



- ▶ **Create your own version of rms.n3**
 - `./sonic-annotator -s vamp:bbc-vamp-plugins:bbc-energy:rmsenergy > rms.n3`
- ▶ **Run rms.n3 on all of the example files in audio**
 - `./sonic-annotator -r ~/Documents/GitHub/toss2025/audio -t rms.n3 -w csv`

Python-based Coding



Overview

- ▶ **Google Colab allows for a range of Python libraries to be imported (and, if need be, installed)**
- ▶ **Data can be imported from Git repositories**
- ▶ **We will first import/visualise data from Sonic Visualiser/Annotator**
- ▶ **Then we will use librosa and pyAMPACT to estimate acoustic descriptors in Python**

Google Colab



Visualising Imported Data

The screenshot shows a Google Colab notebook interface. At the top, there's a navigation bar with a 'PRO' badge, a file icon, the notebook title 'toss2025-notebook1.ipynb', and icons for saving and sharing. Below the title, a menu bar includes File, Edit, View, Insert, Runtime, Tools, and Help. A search bar and command buttons for Code and Text are also present. The main area contains a sidebar with icons for file operations like clone, copy, paste, and delete. The notebook content is organized into sections:

- Load files from Git repository**

```
[ ] !git clone https://github.com/jcdevaney/audioAnalysisWorkshop.git
```

↳ fatal: destination path 'audioAnalysisWorkshop' already exists and is not an empty directory.
- Load libraries**

```
[ ] import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import librosa
```
- Functions to plot a spectrogram with an overlay of imported data**

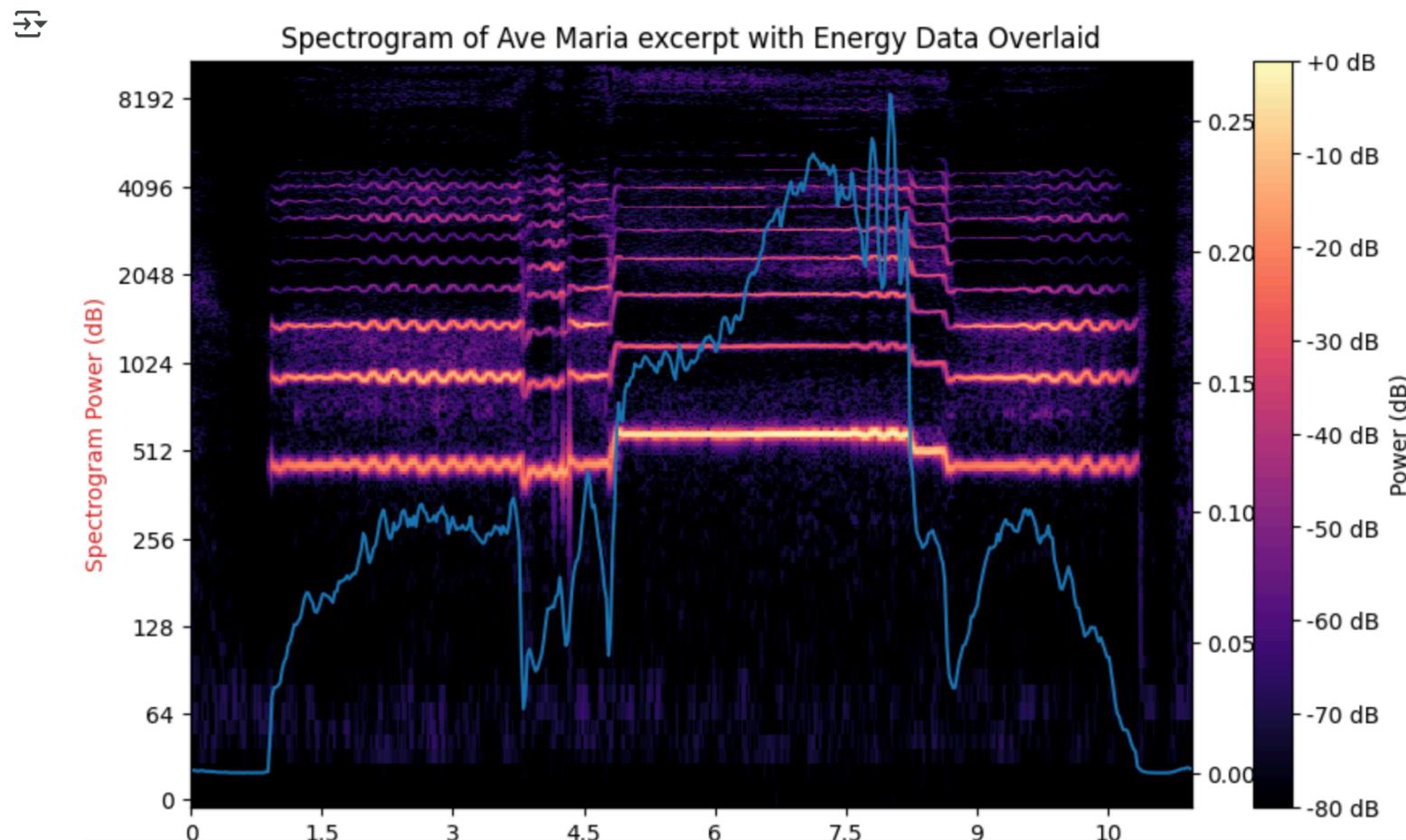
toss2025-notebook1.ipynb

Google Colab



Visualising Imported Data

```
[6] # plot the imported CSV data over a spectrogram of the corresponding audio
audiofileEX = '/content/audioAnalysisWorkshop/audio/avm.wav'
titleEX = 'Spectrogram of Ave Maria excerpt with Energy Data Overlaid'
overlayPlot(audiofileEX,csvEx,titleEX)
```



Google Colab Exercise



Visualising Audio Data

- ▶ **Using the `overlayPlot` function, plot the rms csv files we generated with Sonic Annotator for**
 - `chopinSeg.wav`
 - `kingsLoop.wav`
 - `mozartSeg.wav`

`toss2025-notebook1.ipynb`

librosa

Overview



- ▶ **Python-library of audio signal processing tools**
- ▶ **Robust audio import, representation generation, and audio feature extraction**
- ▶ **Limited number of timbral descriptors available**
- ▶ **Other libraries can be used to generate additional timbral descriptors**
 - parselmouth, pyACA, PyTimbre, etc.

Some Other Libraries

Acoustic Descriptors

	Alicante	AMPACT	Audio	BBC Plugins	BeatRoot	Cepstral Pitch	Chordino	Chroma Toolkit	Essentia	INESC Porto	JAudio	LIBROSA	LibExtract	Madmom	MARSyas	Maurka	MELODIA	MIR-EDU	MISEA	MIR Toolbox	OnsetDSS	GMM Plugins	PYIN	Sementino	Silvert	Timbre Toolkit	Tony	
Low-level Features																												
RMS	X		X					X	X	X	X					X	X											
Intensity	X	X						X								X												
F0	X	X			X			X			X	X	X						X		X					X		
Chroma						X	X	X			X			X				X	X									
MFCCs	X	X						X	X	X	X		X			X	X	X	X									
Synchronicity								X			X							X								X		
Spectral Desc	X	X	X					X	X	X	X		X	X		X	X	X								X		
Zero-crossing								X	X	X	X		X			X	X	X								X		
Onset Detect Func	X	X	X					X		X										X								
High-Level Features																												
Melody								X							X													
Chords						X		X				X										X						
Key							X				X			X					X	X								
Tuning						X	X	X																				
Onsets	X	X	X	X				X		X		X	X	X	X	X	X	X	X	X	X					X		
Offsets	X																										X	
Beats		X		X				X	X	X	X	X	X					X	X									
Downbeats														X													X	
Tempo			X					X		X		X	X	X				X	X									
Notetracking	X	X	X					X				X									X	X	X	X	X	X		
Segments								X										X	X	X	X							
Machine Learning																												
PCA									X			X																
SOM														X			X											
KNN										X			X								X							
GMM														X	X							X						
SVM											X			X														
NN												X		X														
HMM														X														
CRF														X														

librosa

Example



CO PRO toss2025-notebook2.ipynb ⭐ ☁

File Edit View Insert Runtime Tools Help

Commands + Code + Text

☰ 🔎 ↻ 🔑 📂

> Load files from Git repository
[] ↴ 1 cell hidden

> Load libraries
[] ↴ 1 cell hidden

> Fundamental Frequency
▶ ↴ 4 cells hidden

⌄ Onset Estimation

↑ ↓ ⏪ ⏩

toss2025-notebook2.ipynb

librosa Exercise

Estimate Acoustic Descriptors



- ▶ **Estimate and plot the librosa onset detection estimates for**
 - chopinSeg.wav
 - kingsLoop.wav
 - mozartSeg.wav

librosa

Import Annotations
Calculate Note-wise Summaries



CO PRO toss2025-notebook3.ipynb ⭐

File Edit View Insert Runtime Tools Help

Commands + Code + Text

> Load files from Git repository

[] ↴ 1 cell hidden

> Load libraries

[] ↴ 1 cell hidden

> Functions to plot a spectrogram with an overlay of imported data

[] ↴ 1 cell hidden

▼ Import Annotations

✓ 19s ⏪ # load audio file
audiofileEX = '/content/audioAnalysisWorkshop/audio/avm.wav'
y, fs = librosa.load(audiofileEX)

generate spectrogram
D = librosa.amplitude_to_db(np.abs(librosa.stft(y)), ref=np.max)

toss2025-notebook3.ipynb

librosa Exercise

Calculate Note-wise Descriptors



- ▶ **Following from the example for F0, calculate the mean spectral centroid for each note in `avm.wav`**

pyAMPACT

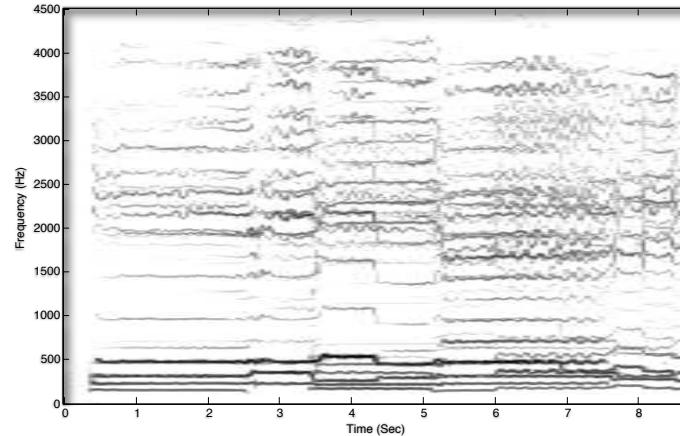
Overview



- ▶ **pyAMPACT (Python-based Automatic Music Performance Analysis and Comparison Toolkit) links symbolic and audio music representations to facilitate “note”-level estimation of performance data from audio**
 - This can be done with annotations instead of a fully-specified musical score
- ▶ **pyAMPACT’s also supports importing aligned analytic annotations from a range of formats (e.g., Humdrum, Dezrann)**

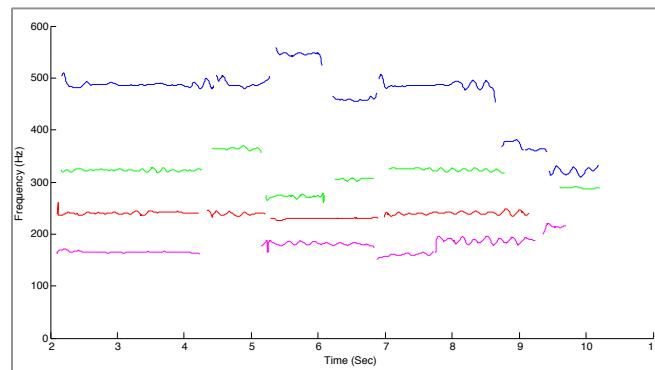
pyAMPACT

Score-Audio Alignment

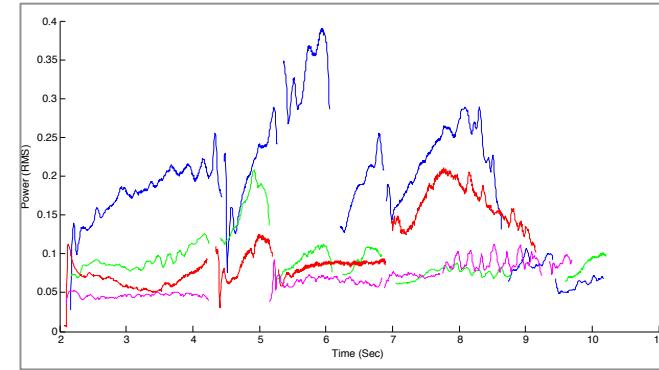


Alignment

Note-wise f_0 estimations

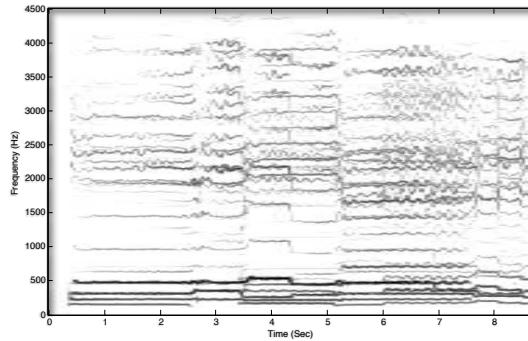


Note-wise power estimations

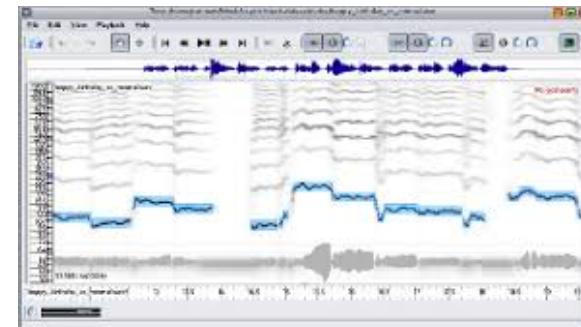


pyAMPACT

Annotation-based Usage



Source
Separation

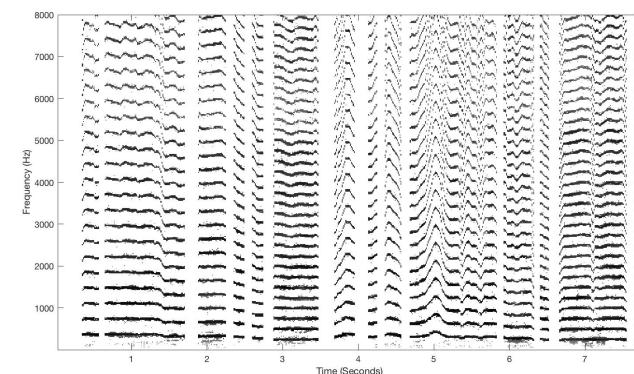


Tony

Mauch et al. (2015)

AMPACT

Spectral Reconstruction



pyAMPACT

Acoustic Descriptors



Currently Implemented

Timing: duration, inter-onset-interval

Pitch: perceived pitch, vibrato rate and depth, jitter, slope and curvature of fundamental frequency

Loudness: perceived loudness, shimmer

Timbre: spectral centroid, slope, flatness, and flux, harmonics to noise ratio, MFCCs

Also Extensible to Other Descriptors

pyAMPACT



Data Export (In Development)

```
<?xml version="1.0" encoding="UTF-8"?>
<mei xmlns="http://www.music-encoding.org/ns/mei">
  <meiHead>
    <fileDesc>
      <titleStmt>
        <title/>
      </titleStmt>
      <pubStmt/>
    </fileDesc>
  </meiHead>
  <music>
    <body>
      <mdiv>
        <score>
          <scoreDef>
            <staffGrp>
              <staffDef clef.shape="G" clef.line="2" n="1" lines="5"/>
            </staffGrp>
          </scoreDef>
          <section>
            <measure>
              <staff n="1">
                <layer>
                  <note pname="b" oct="4" dots="1" dur="2" tie="i"/>
                  <note pname="b" oct="4" dur="8" tie="t"/>
                  <note pname="a" oct="4" dur="8"/>
                </layer>
              </staff>
            </measure>
            <measure>
              <staff n="1">
                <layer>
                  <note pname="b" oct="4" dur="4"/>
                  <note pname="d" oct="5" dots="1" dur="2"/>
                </layer>
              </staff>
            </measure>
            <measure>
              <staff n="1">
                <layer>
                  <note pname="c" oct="5" dur="4"/>
                  <note pname="b" oct="4" dots="1" dur="2" tie="i"/>
                </layer>
              </staff>
            </measure>
          </section>
        </score>
      </mdiv>
    </body>
  </mei>
```

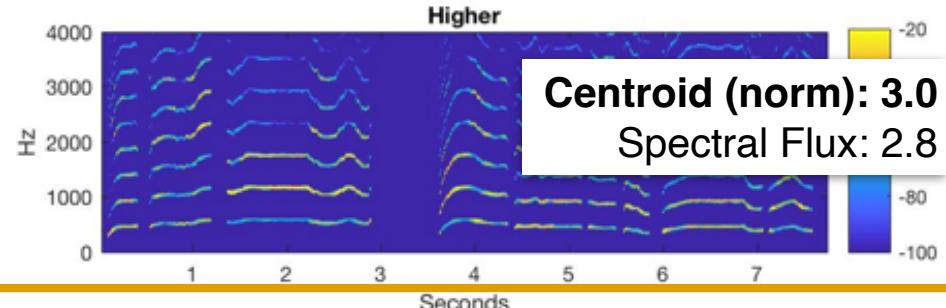
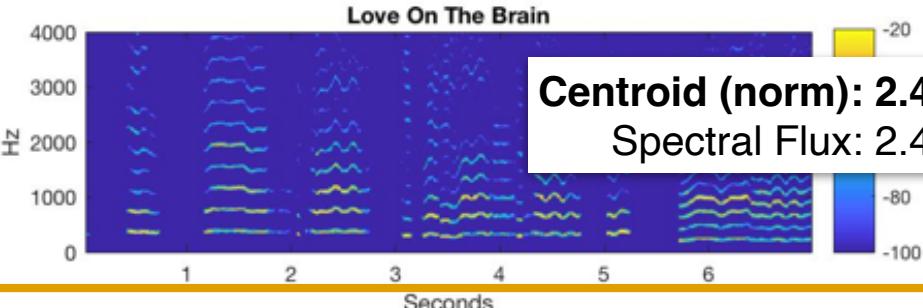
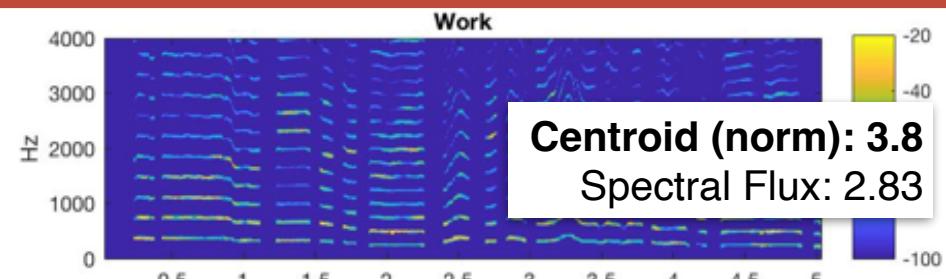
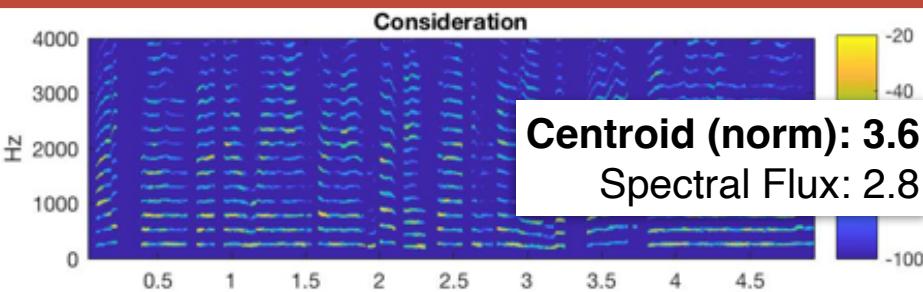
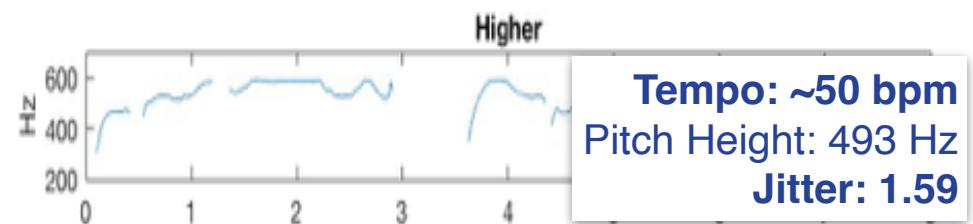
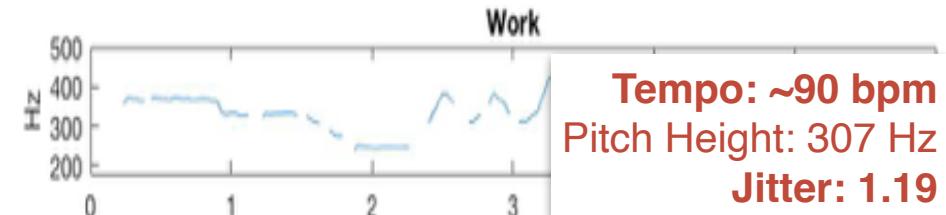
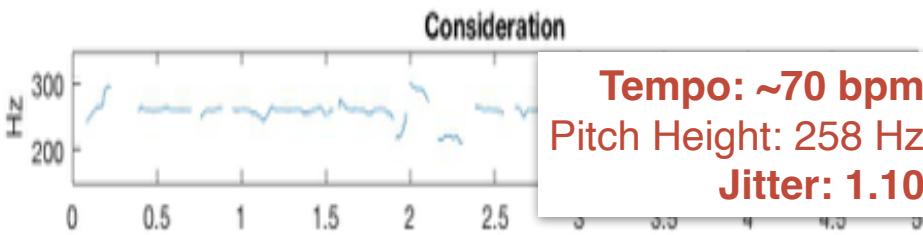
A - ve Ma - ri - - - - a

```
<performance>
  <recording>
    <avFile mimetype="audio/aiff" target="aveMaria.aiff"/>
    <when absolute="00:00:00.00" xml:id="when_1" data="#note_1">
      <extData>
        <![CDATA[
          {"offset": "00:00:02.9005",
           "pitch": "455.98",
           "slopeF0": "-0.10373",
           "curvatureF0": "<-0.15858",
           "vibdepth": "4.4811",
           "vibrate": "26.911",
           "jitter": "0.26913",
           "loudness": "23.484",
           "shimmer": "18.751",
           "specFlat": "0.00088866",
           "specSlope": "-0.0051845",
           "specCent": "0.028675",
           "specFlux": "0.00088866",
           "HNR": "0.10361",
           "contF0": [454.3737606, 454.7165531, 455.2337513, 455.4622624, 456.0605954],
           "contPWR": [8.1059e-05, 0.00013102, 0.00013133, 0.00020036, 0.00025492],
           "contSpecFlat": [0.18883, 0.093385, 0.06207, 0.074795, 0.058076],
           "contSpecSlope": [-0.0042241, -0.0049742, -0.0052679, -0.0051505, -0.0052442],
           "contSpecCent": [0.069648, 0.037643, 0.025114, 0.030121, 0.026127],
           "contSpecFlux": [0.063614, 0.063614, 0.027968, 0.0035322, 0.0019758],
           "contHNR": [0.079702, 0.12358, 0.054247, 0.054466, 0.099262]
        ]]>
      </extData>
    </when>
  </recording>
</performance>
```

pyAMPACT Example

Pitch and Timbre in Rihanna's Anti album

Speech-Like
Emotional/Raw





pyAMPACT

Example

CO PRO toss2025-notebook4.ipynb ⭐ ☁

File Edit View Insert Runtime Tools Help

Commands + Code + Text

pyAMPACT

```
[ ] # Due to package dependencies issues, you will have to restart the runtime
# after running this cell and then and re-run it a second time

!pip install pyampact
import pyampact

[ ] !git clone https://github.com/jcdevaney/pyAMPACTtutorials.git
```

▼ Tony Import

▼ Load a Tony-style annotation file

```
[ ] annotation_file = '/content/pyAMPACTtutorials/test\_files/underspecified/avm.csv'
```

```
piece2 = pyampact.Score(annotation_file)
```

▼ Run the alignment with pyampact's `run_alignment` function.

```
[ ] dtw, spec, nmat = pyampact.run_alignment(y, original_sr, piece2, piece2.nmats())
```

toss2025-notebook4.ipynb

pyAMPACT Exercise



Import Labels and Estimate Acoustic Descriptors

- ▶ **Use TONY to annotate the vocal line in `somClipVocals.wav` (or another vocal audio file of your choice)**
- ▶ **Export the data as a CSV file and upload it to your Google Colab notebook**
- ▶ **Run the pyAMPACT code to open the audio and symbolic files and calculate the acoustic descriptors**

`toss2025-notebook4.ipynb`

Overview

1

Signal Processing Basics

2

Acoustic Descriptors

3

GUI-based Tools

4

Python-Based Tools

5

Summary

6

Other Research Examples

Recent Work by Former Advisees

- ▶ **Kristi Hardman**

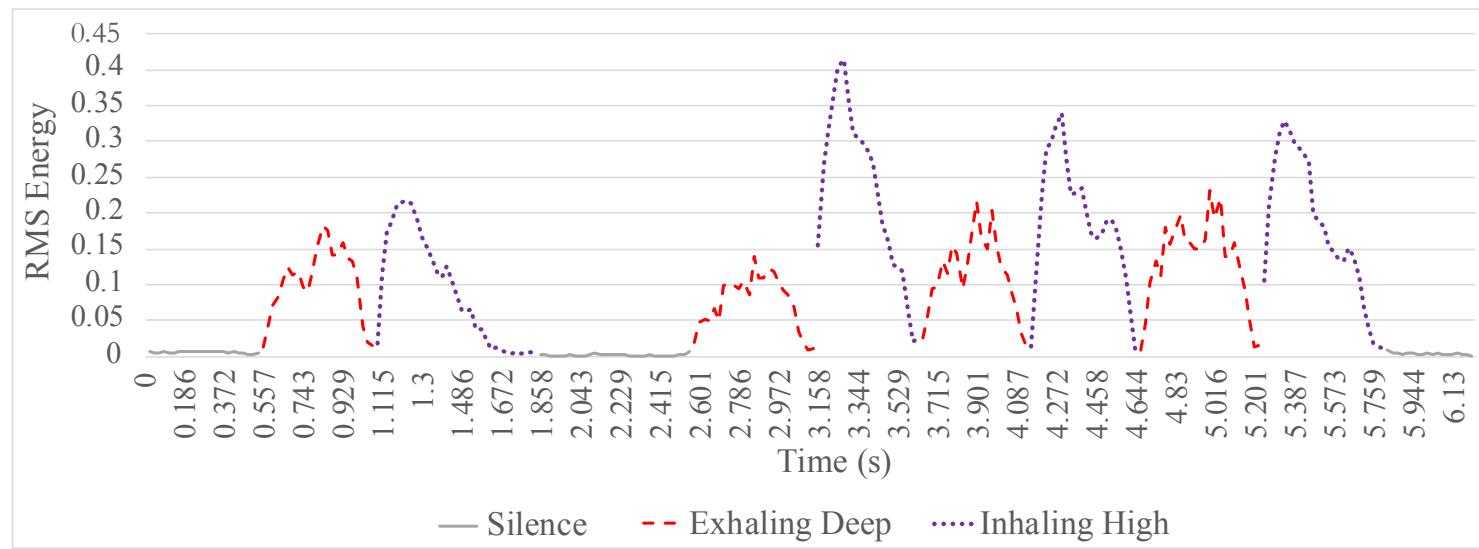
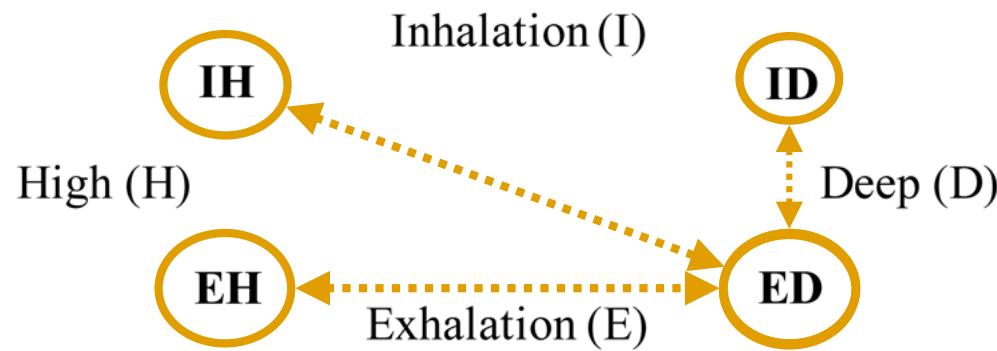
- Vocal Timbre
- Tagaq's katajjaq-inspired vocals

- ▶ **Stephen Spencer**

- Relative Brightness
- Spectral Centroid Quotient

Vocal Timbre (Tagaq's *katajjaq*-inspired vocals)

Kristi Hardman (UNC Charlotte)

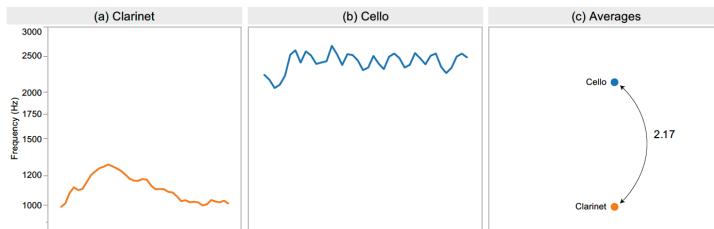


Relative Brightness/Spectral Centroid Quotient

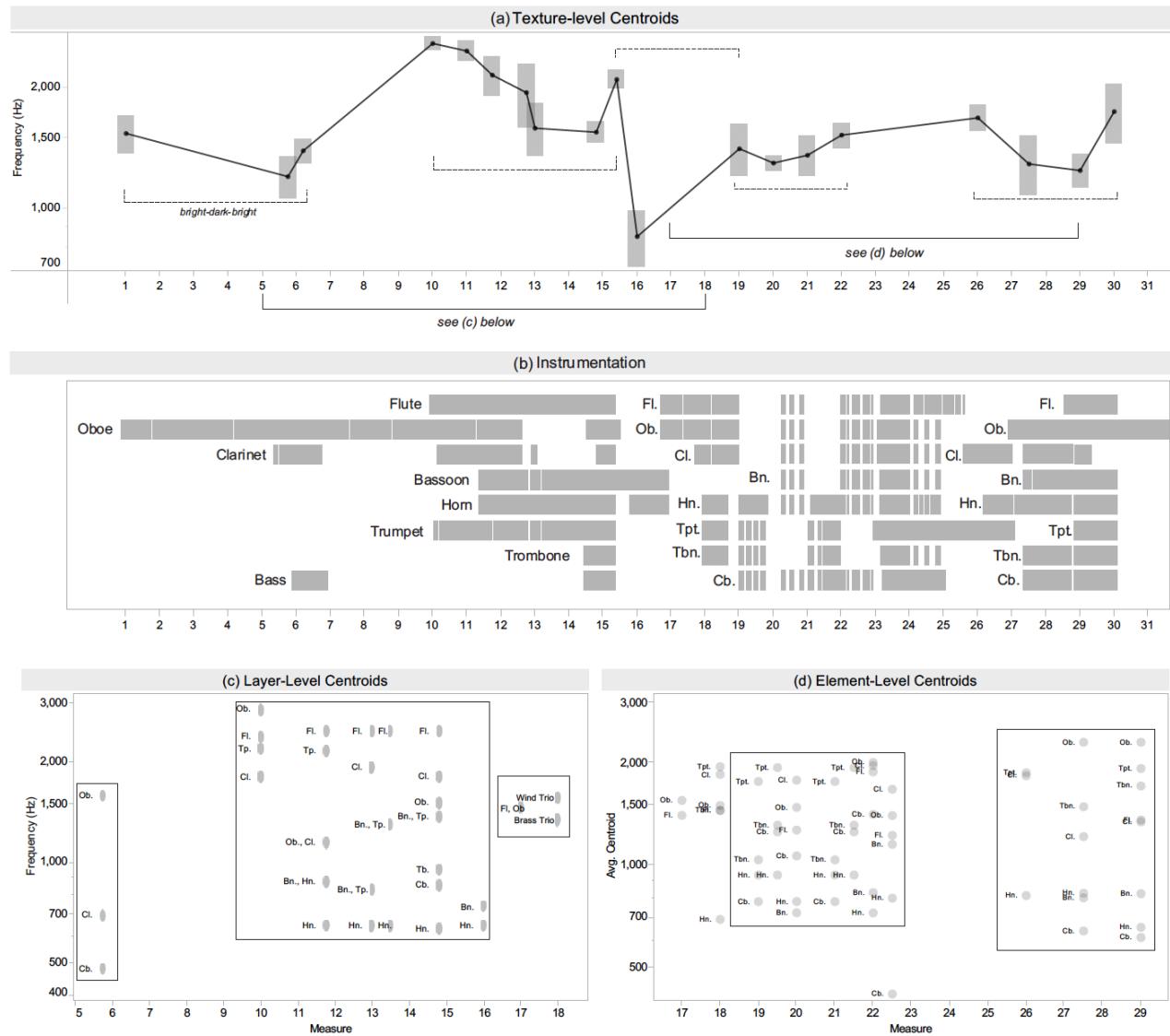
Stephen Spencer (Hunter College)

Spectral centroid quotient (SCQ) quantifies the brightness relationships between two or more sounds

$$\text{SCQ} = \frac{\text{SC}_a}{\text{SC}_b}$$



Three "textural levels" for measuring brightness in post-tonal music: element, layer, and texture



Summary

Acoustic Descriptors

► **Energy**

- Root mean square energy (RMS), Decibels (dB), shimmer

► **Pitch**

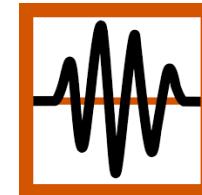
- Fundamental Frequency (F0), F0 summary descriptors (mean pitch, vibrato rate, vibrato depth, jitter)

► **Timbre**

- Spectral centroid, Spectral flux/variation, Harmonics-to-noise (HNR) ratio, Mel-frequency cepstral coefficients (MFCCs)

Summary

GUI-Based Tools



TONY

	Visualise	Annotate	Plugins
Audacity	Time-domain Frequency-domain	✓	Wide range
Sonic Visualiser	Time-domain Frequency-domain	✓	Vamp
TONY	Time-domain Frequency-domain	✓ (melody)	Built-in estimates of note onsets and offsets and fundamental frequency

Summary

Python-based Tools



▶ **librosa**

- Robust audio import, representation generation, and audio feature extraction
- Limited number of timbral descriptors available

▶ **pyAMPACT**

- Builds on librosa and other libraries to link symbolic and audio music representations to facilitate “note”-level estimation of performance data from audio
 - Also supports linking/alignment of other time-series data (e.g., musical analyses or continuous response data)



Summary

Resources

- ▶ **GitHub Repo ([https://github.com/jcdevaney/
TOSS2025](https://github.com/jcdevaney/TOSS2025))**
 - Audio files
 - Colab notebooks
 - PDFs of papers for further reading
 - Copy of Sonic Annotator for easy installation
 - Copy of these slides



Summary

Resources

- ▶ **GitHub Repo ([https://github.com/jcdevaney/
TOSS2025](https://github.com/jcdevaney/TOSS2025))**
 - Audio files
 - Colab notebooks
 - PDFs of papers for further reading
 - Copy of Sonic Annotator for easy installation
 - Copy of these slides

Thank you!