Acoustic Correlates of Timbre

Johanna Devaney
Brooklyn College and Graduate Center, CUNY

https://github.com/jcdevaney/TOSS2025

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)**
  - Audio files
  - Colab notebooks
  - PDFs of papers for further reading
  - Copy of Sonic Annotator for easy installation
  - Copy of these slides

Overview

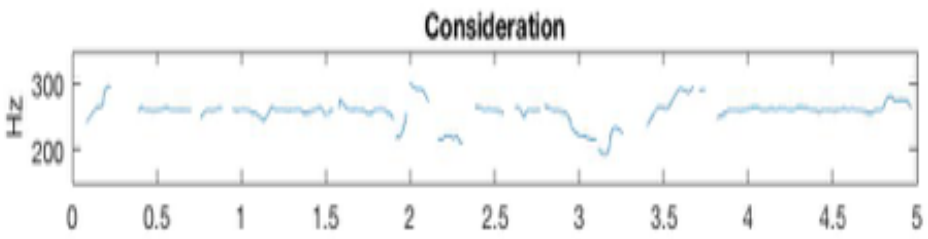
"Note"-level analysis

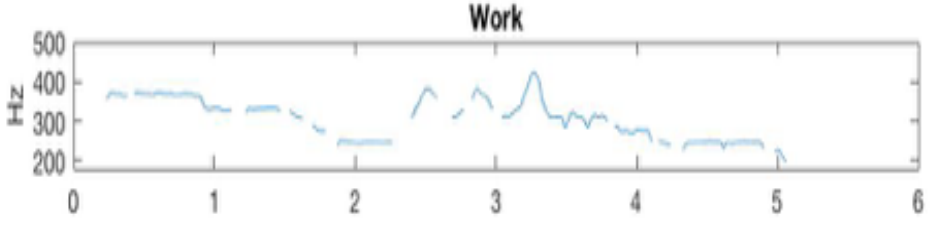
- **Note-based analysis can be useful for Western art music**
  - Particularly for notated instrumental music
  - Adopted for studying non-Western art music traditions
- **Fails to capture all of the nuance in the performance**
  - Particularly for vocals (e.g., transitions between notes)
- **Rooted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

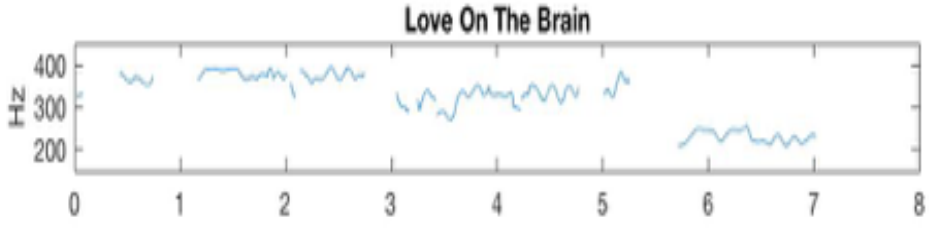
Example Analysis

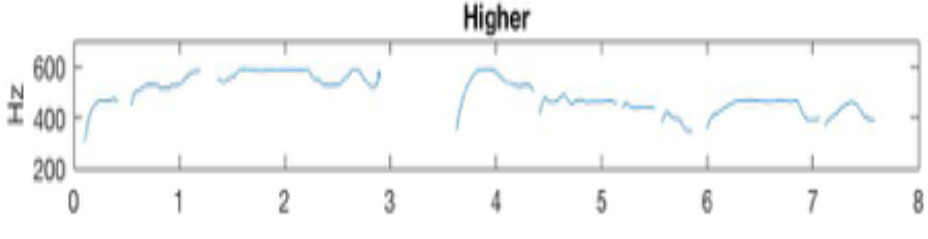
Speech-Like Emotional/Raw

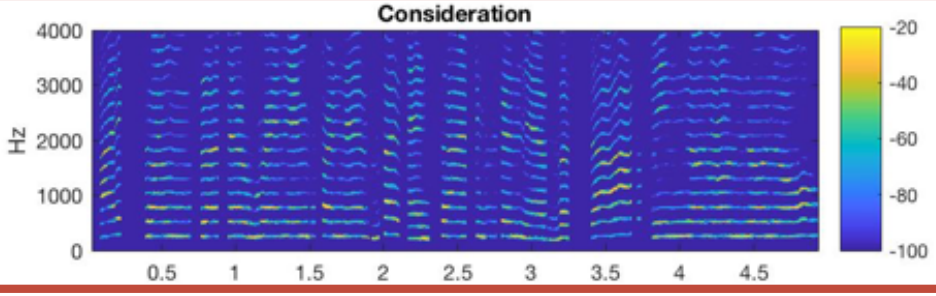
Pitch and Timbre in Rihanna's *Anti* album

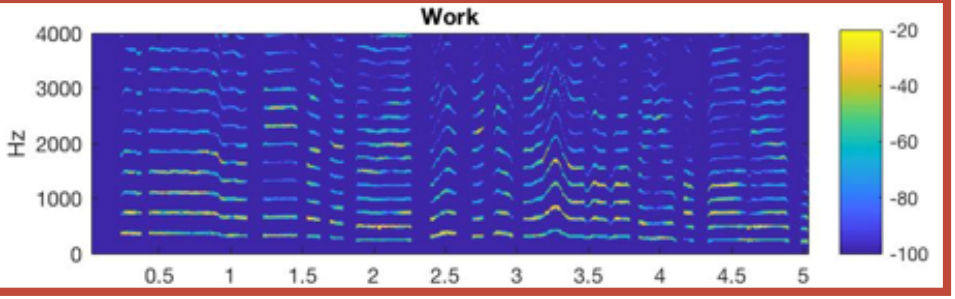


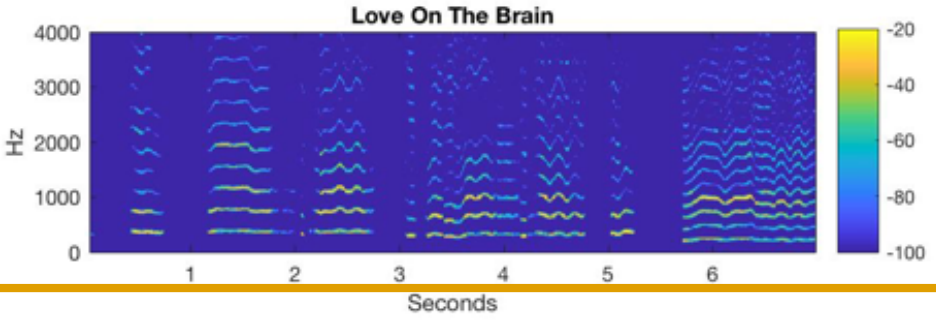


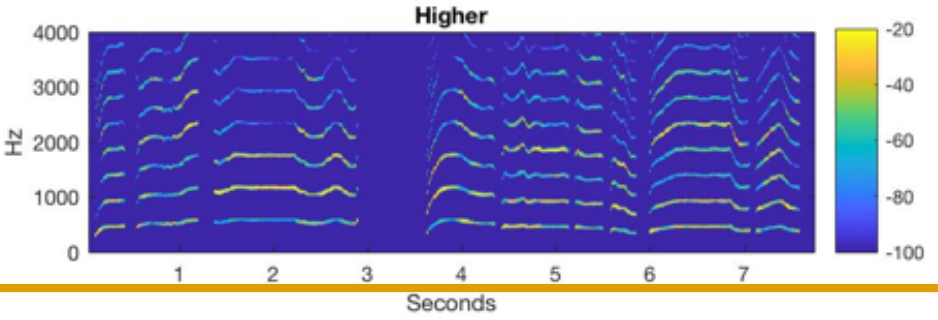








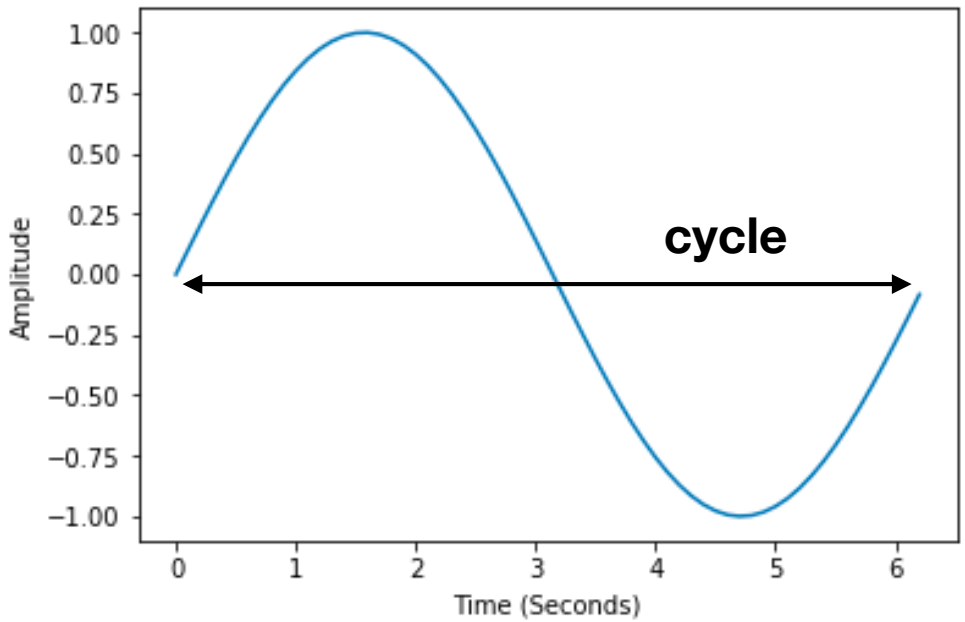




Signal Processing Basics

Frequency

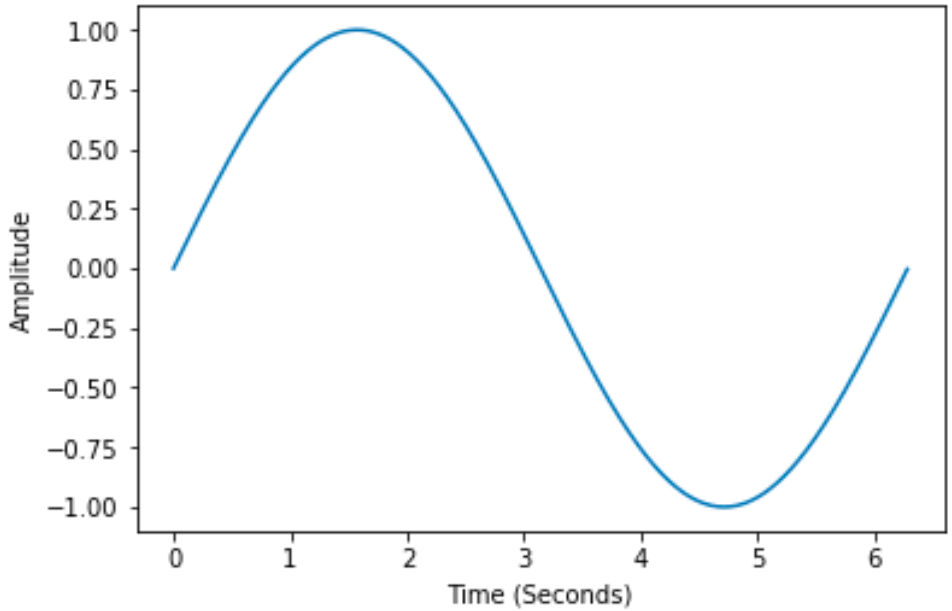
- **periodic acoustic waveform**
  - airs pressure varying with a repeated pattern
- **cycle**
  - one repetition of a waveform



Signal Processing Basics

Amplitude vs Decibels

- **amplitude**
  - amount of air pressure change



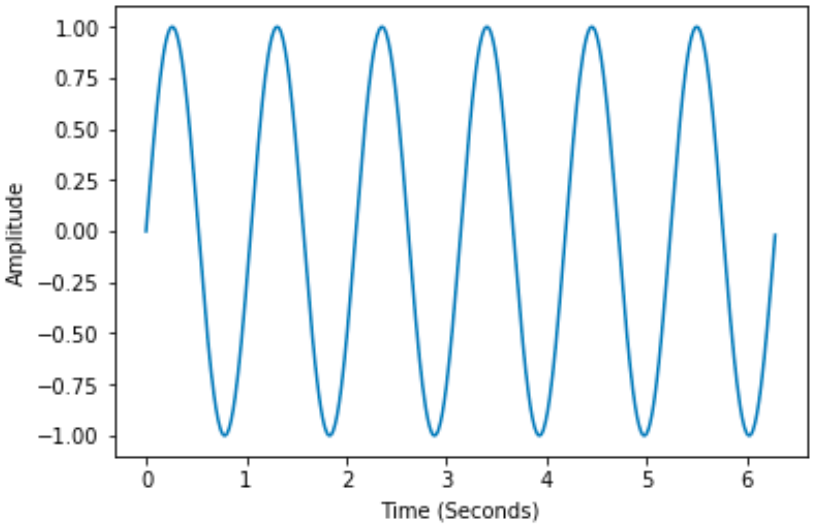
- **decibels**
  - scaling of amplitude values

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

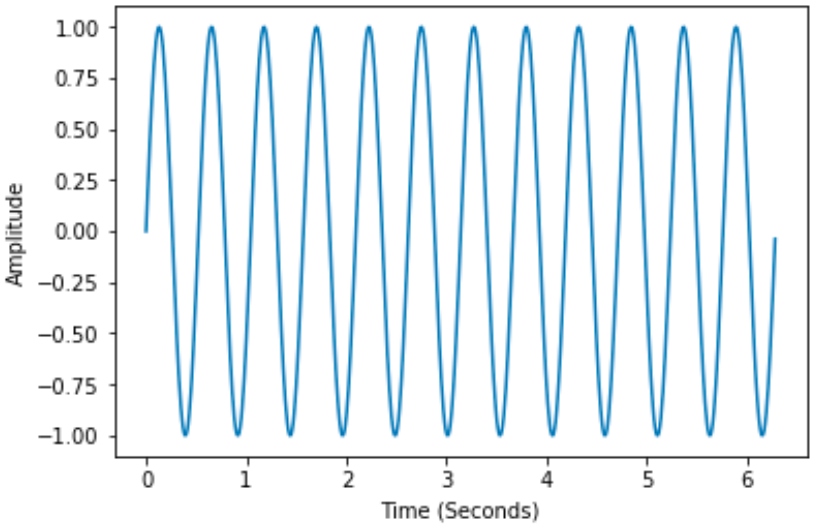
Signal Processing Basics

Frequency

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



approx. 1 Hz

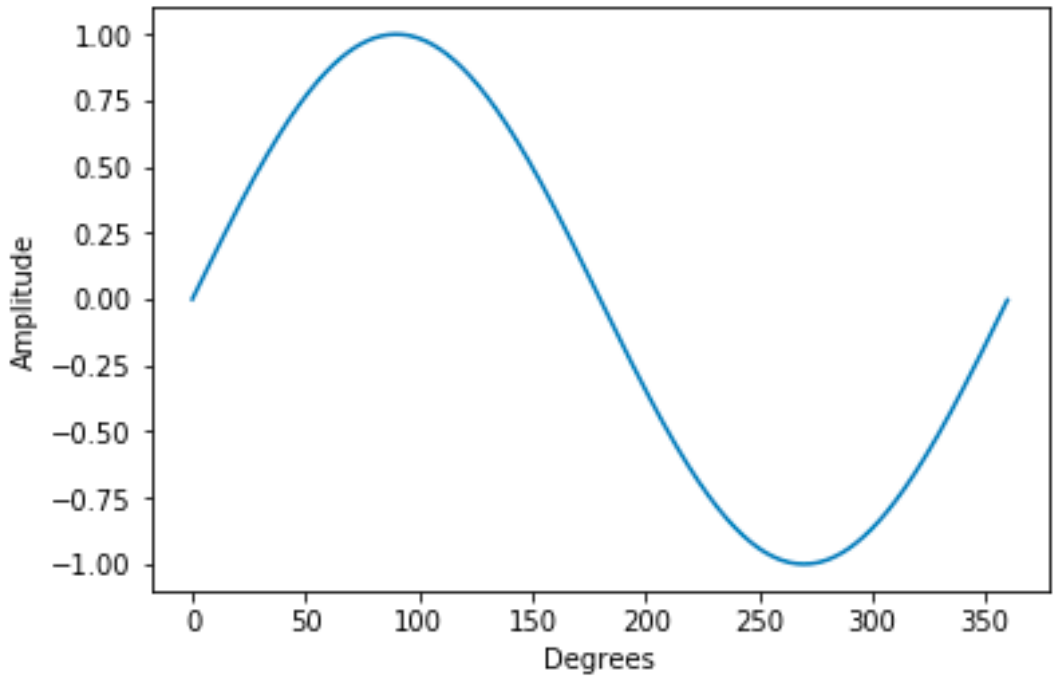


approx. 2 Hz

Signal Processing Basics

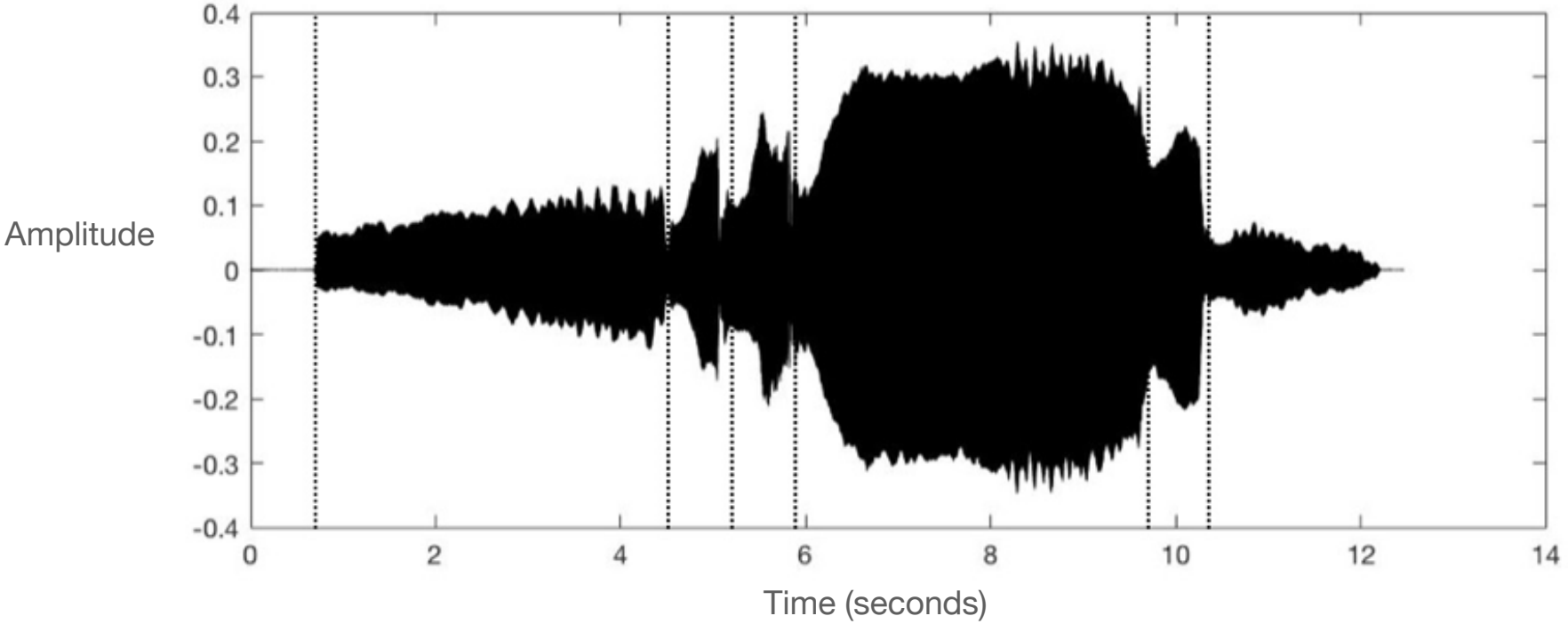
Phase

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

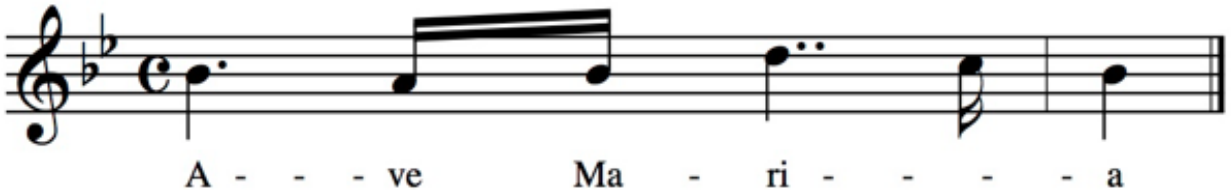


Signal Processing Basics

Time-domain representation

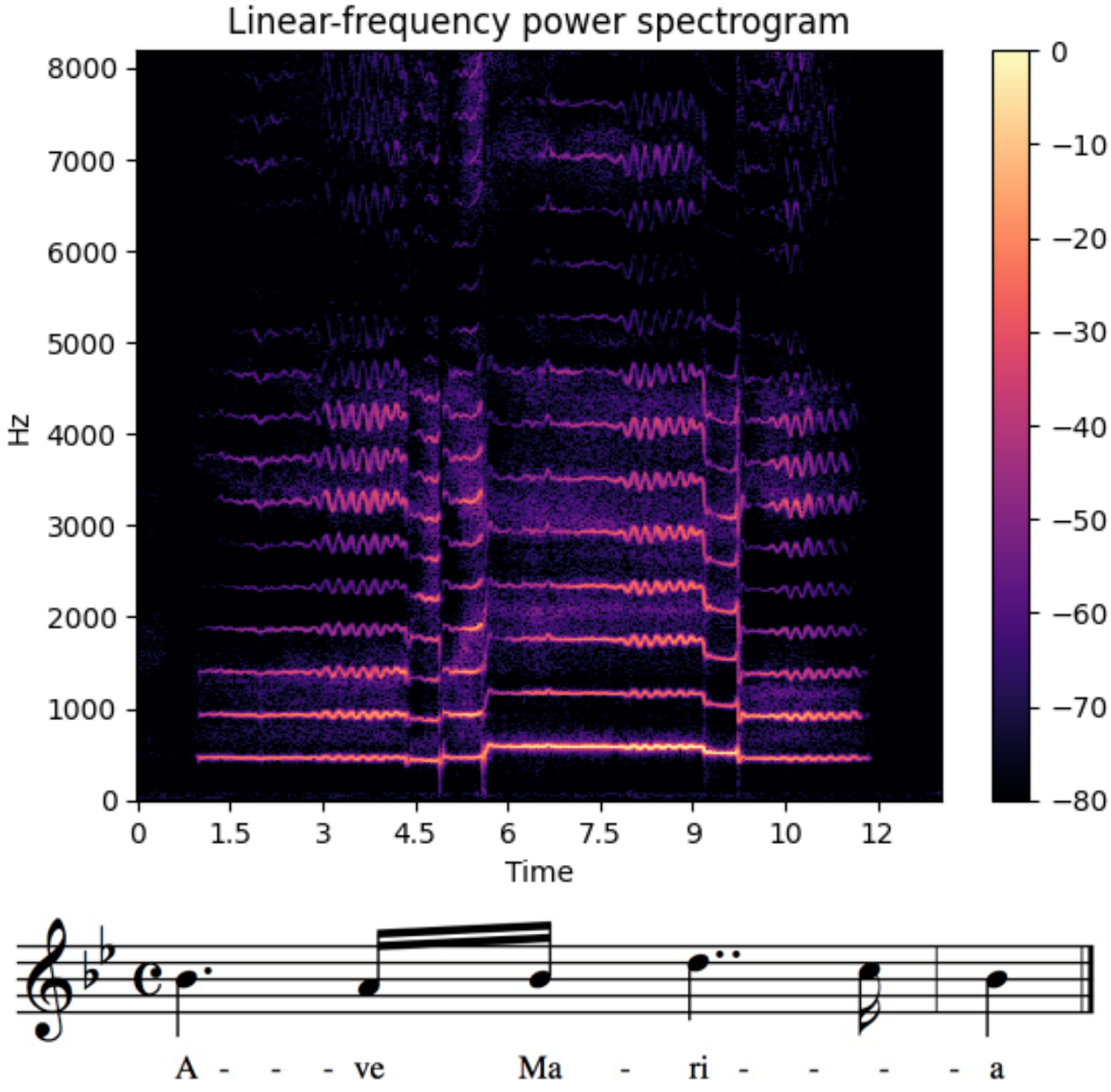


Time domain representation of an audio recording of the opening phrase of Schubert's "Ave Maria". The vertical dotted lines indicate the boundaries of the notes in the score below



Signal Processing Basics

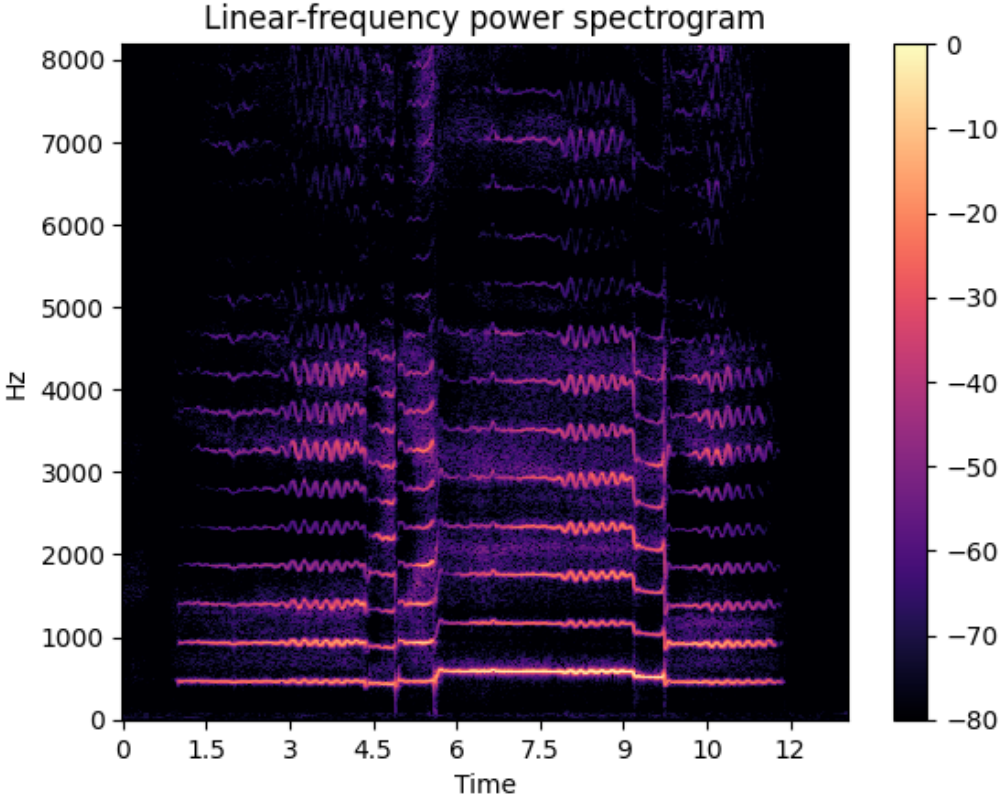
Linear- Frequency Spectrogram



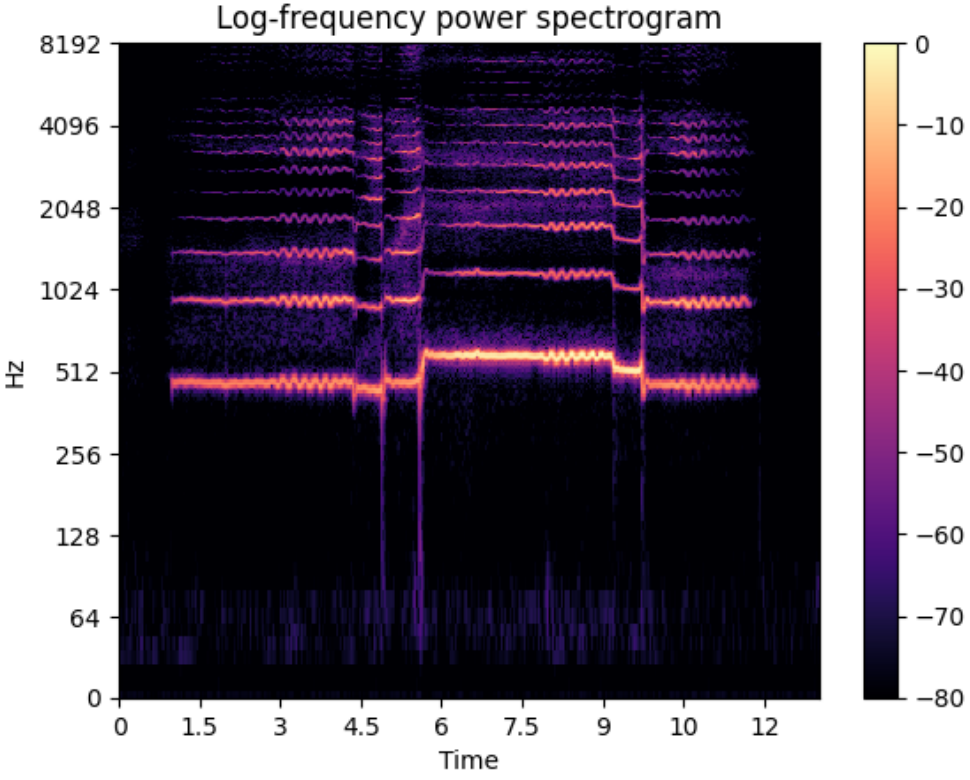
Signal Processing Basics

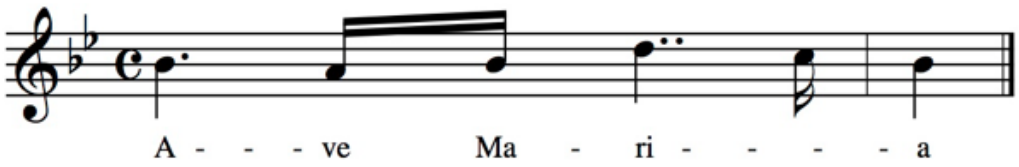
Log-Frequency Spectrogram

Spectrograms often default to a linear spacing of frequencies



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

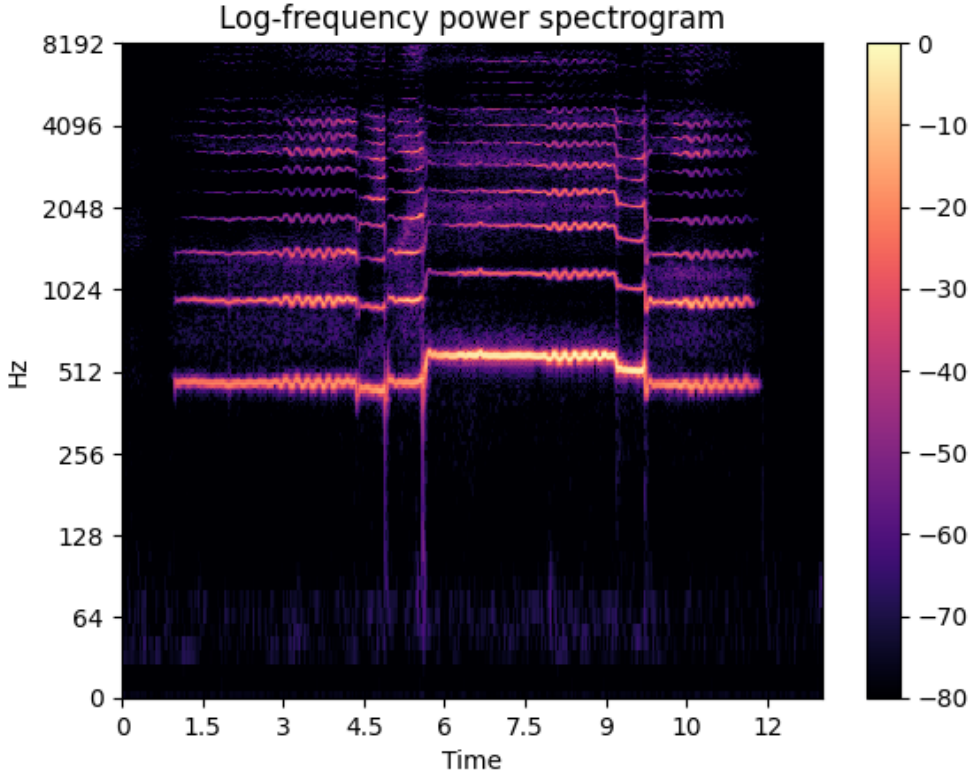


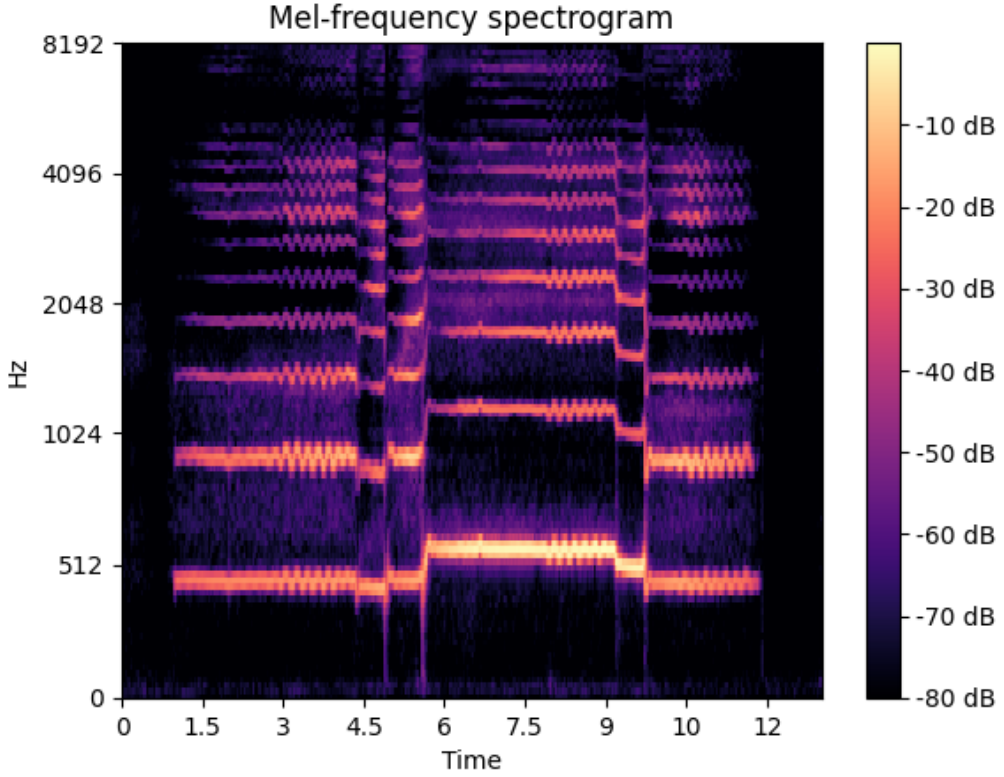


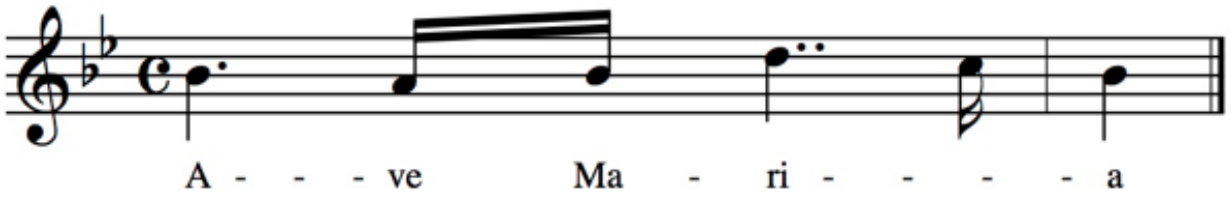
Log-spacing in spectrograms can help observe harmonic and timbral characteristics

Signal Processing Basics

Mel Scale Spectrogram



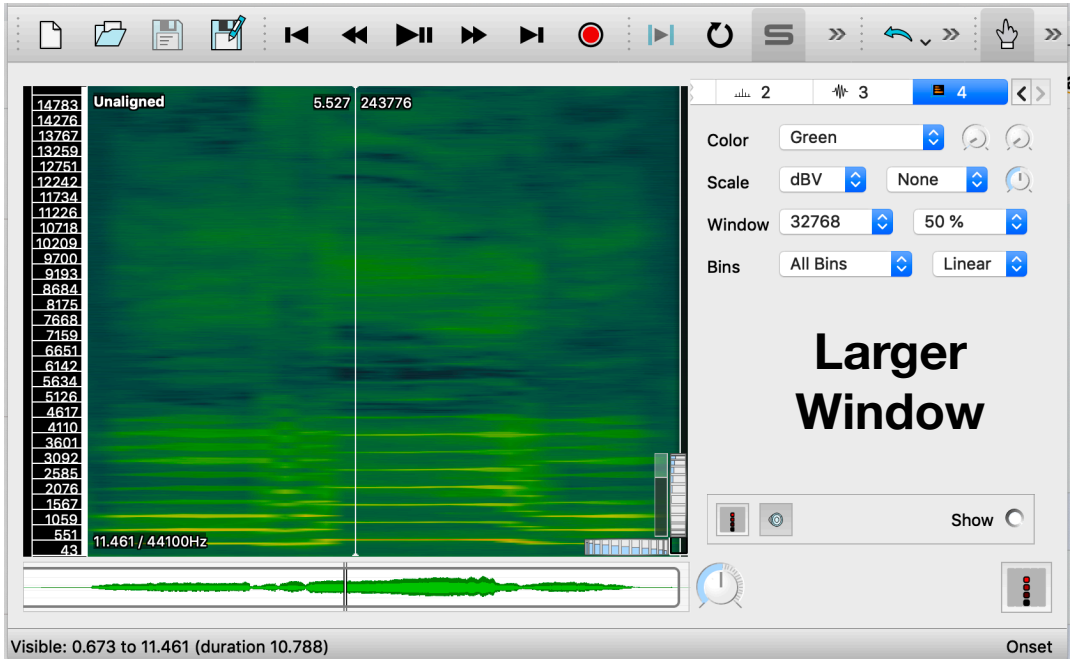




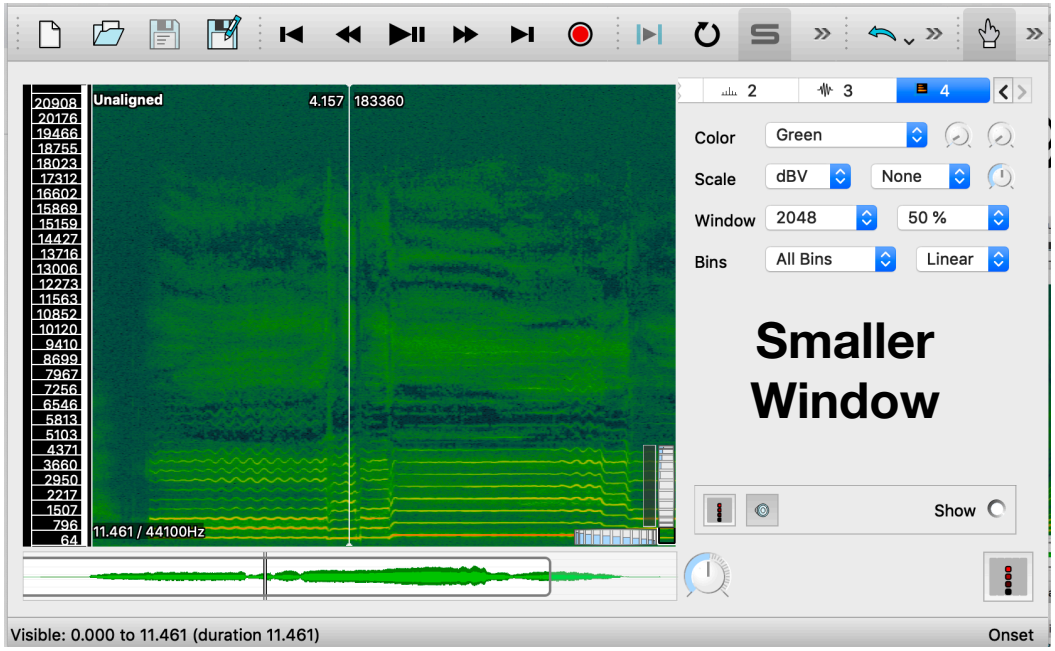
Signal Processing Basics

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 analyzed



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)

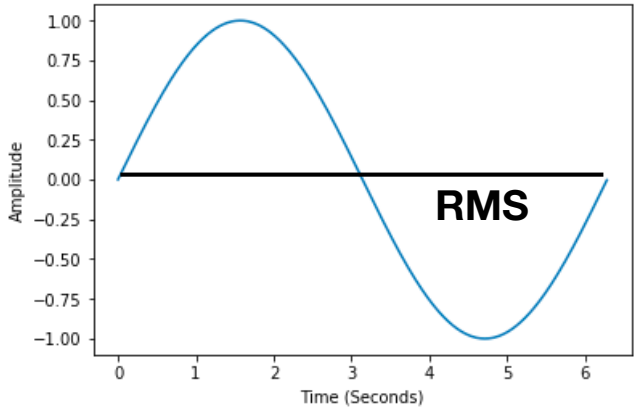
Acoustic Descriptors

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)**

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

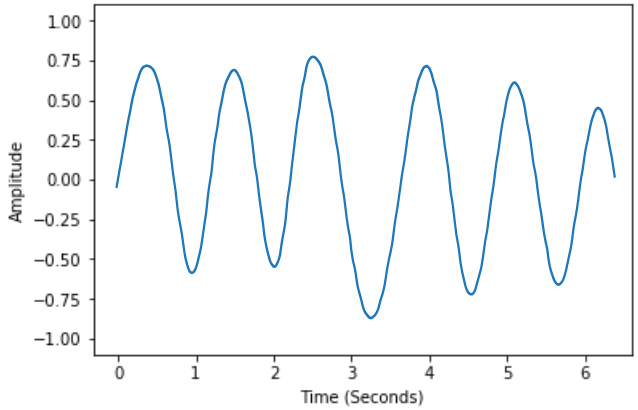
Where:

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

https://en.wikipedia.org/wiki/Root_mean_square

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.

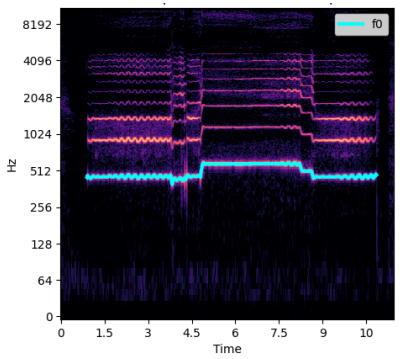
$$\text{Shimmer}_{\text{dB}} = \frac{1}{N-1} \sum_{i=1}^{N-1} 20 \log_{10} \left(\frac{A_{i+1}}{A_i} \right)$$

Where:

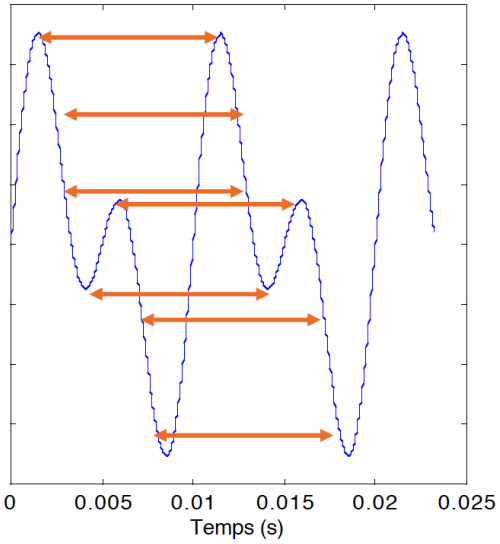
- A_i is the peak amplitude of the i-th period,
- N is the total number of periods.

Acoustic Descriptors

Fundamental Frequency (F0)



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



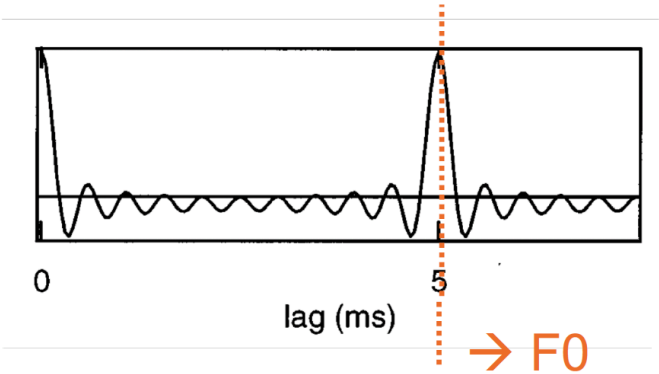
"shift and compare"

$$A_t(\tau) = \sum_{W} x(t)x(t-\tau)$$

product

time lag

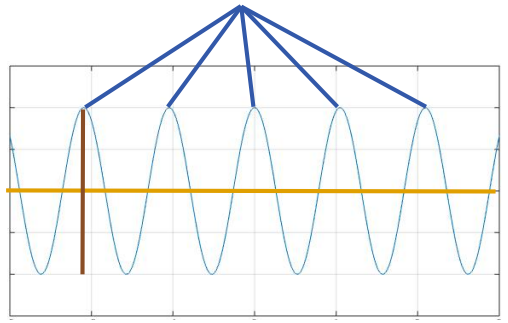
integration window



https://ismir2018.ismir.net/pages/events-tutorial-06.html

Acoustic Descriptors

F0-related Descriptors



Mean Pitch

Vibrato Depth

Vibrato Rate

- **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**

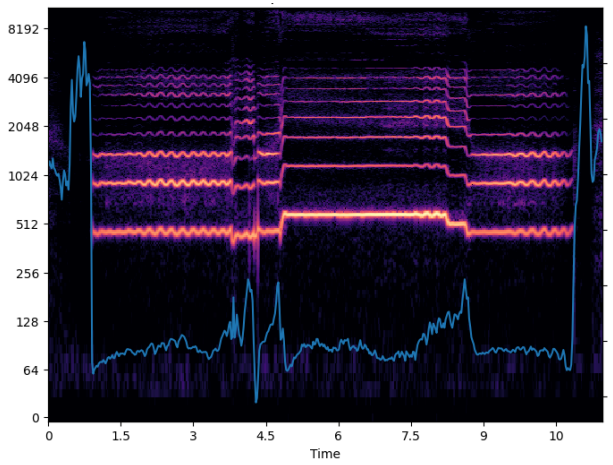
$$\text{Jitter} = \frac{1}{N-1} \sum_{i=1}^{N-1} \frac{|T_{i+1} - T_i|}{\frac{1}{N}\sum_{j=1}^{N} T_j}$$

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**

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

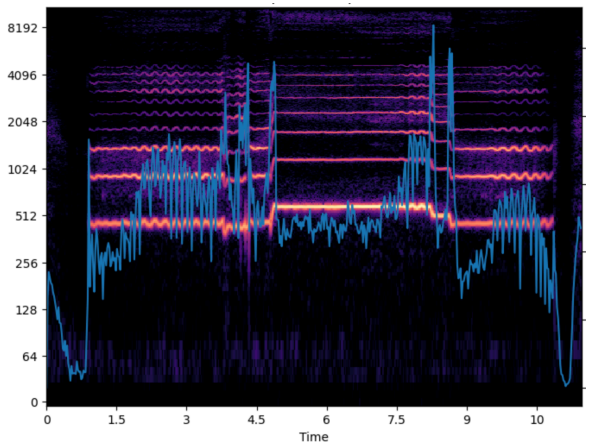
Where:

- 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.

Peeters et al. (2011)

Acoustic Descriptors

Spectral Variation/Flux



- **Changes in the spectrum over time**
- **Can be estimated by calculating**
  - Euclidian distance between spectral frames
  - cosine similarity between spectral frames

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

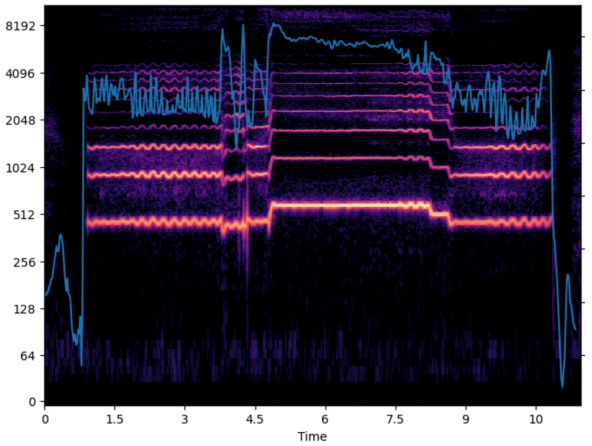
Where:

- $X_t(k)$ 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.

Peeters et al. (2011)

Acoustic Descriptors

Harmonics-to-Noise Ratio



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

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

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

$$NoiseEnergy(E_N(t_m)) = E_T(t_m) - E_H(t_m)$$

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

Where:

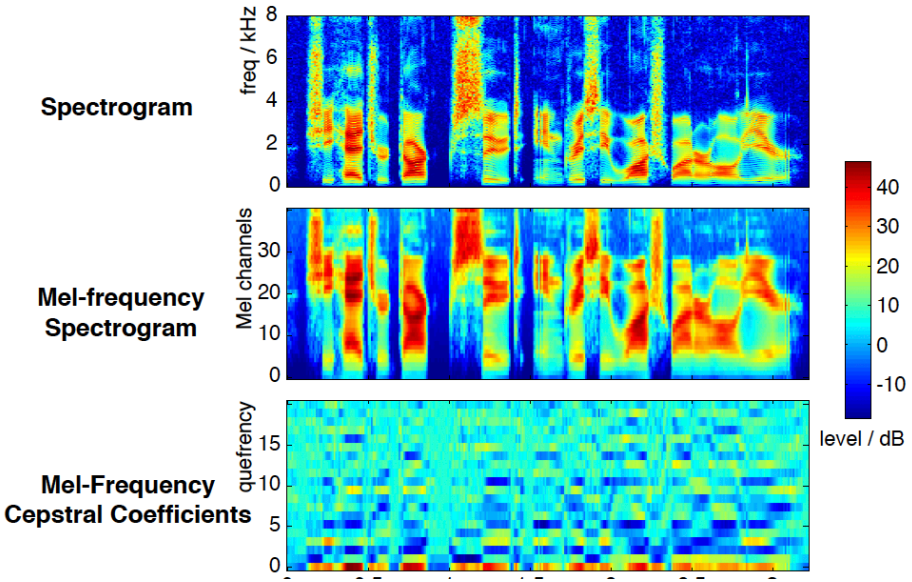
- $x_k(t_m)$ is the amplitude at the k-th frequency bin,
- $a_h(t_m)$ is the amplitude of the h-th harmonic partial,
- K is the number of frequency bins,
- H is the number of harmonic partials.

Peeters et al. (2011)

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

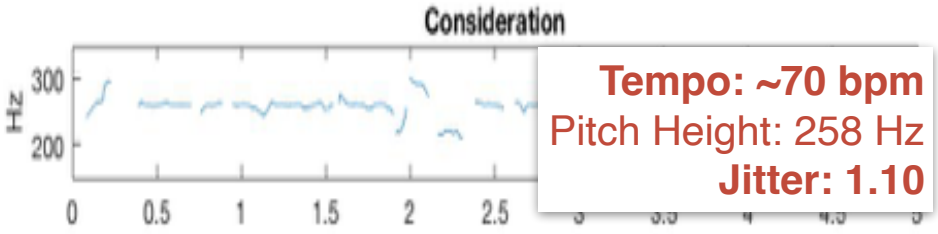
Pitch and Timbre in Rihanna's *Anti* album

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

Example Analysis

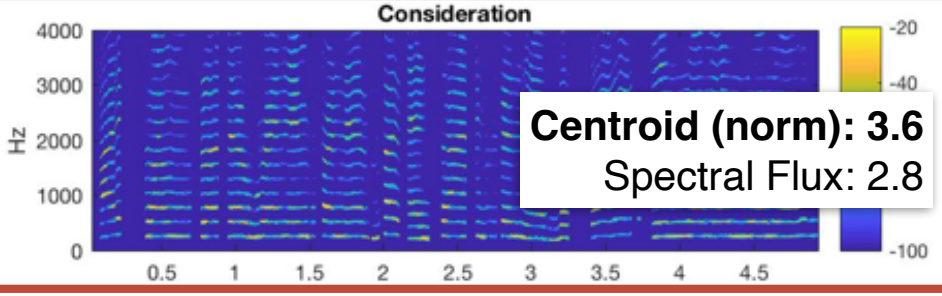
Speech-Like Emotional/Raw

Pitch and Timbre in Rihanna's *Anti* album

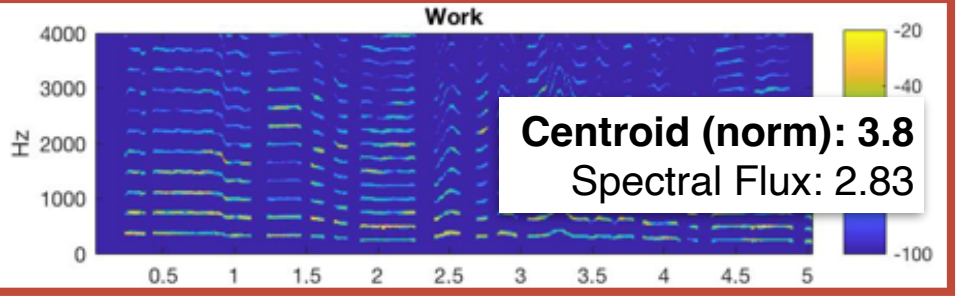


Tempo: ~70 bpm
Pitch Height: 258 Hz
Jitter: 1.10

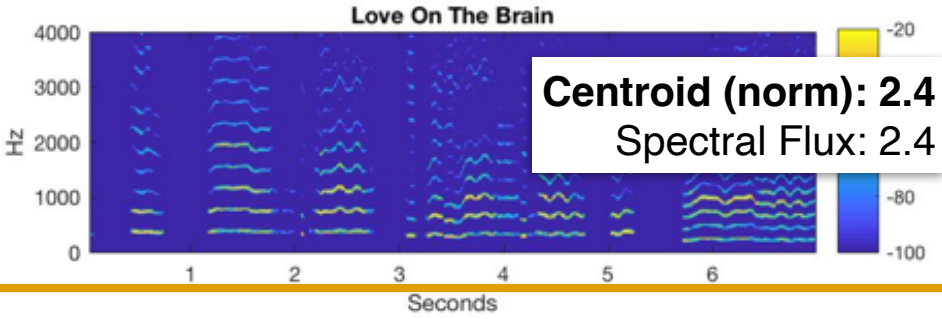
Tempo: ~55 bpm
Pitch Height: 319 Hz
Jitter: 1.45



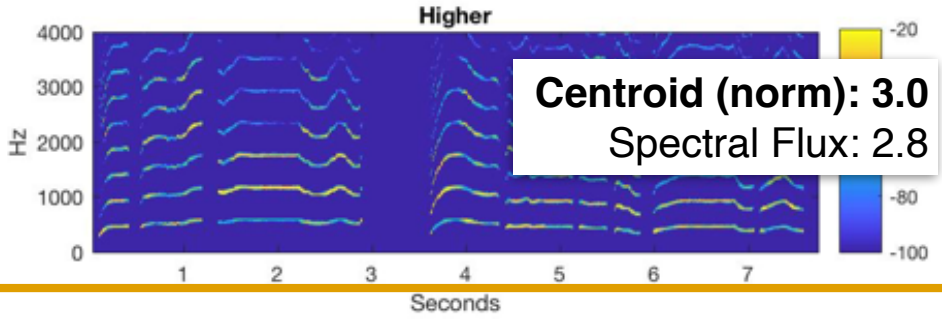
Centroid (norm): 3.6
Spectral Flux: 2.8



Centroid (norm): 3.8
Spectral Flux: 2.83



Centroid (norm): 2.4
Spectral Flux: 2.4



Centroid (norm): 3.0
Spectral Flux: 2.8

GUI-Based Tools

Overview

- **Audacity** 
- **Sonic Visualiser** 
- **Tony** TONY

Audacity

Overview

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

Sonic Visualiser

Overview

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

Tony

Overview

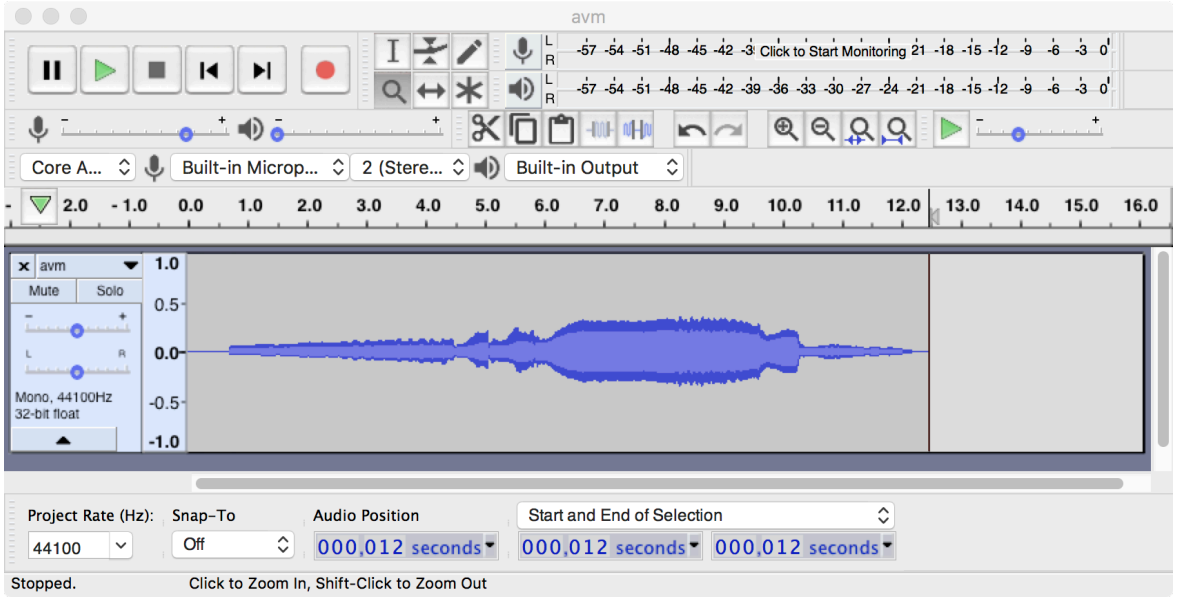
TONY

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

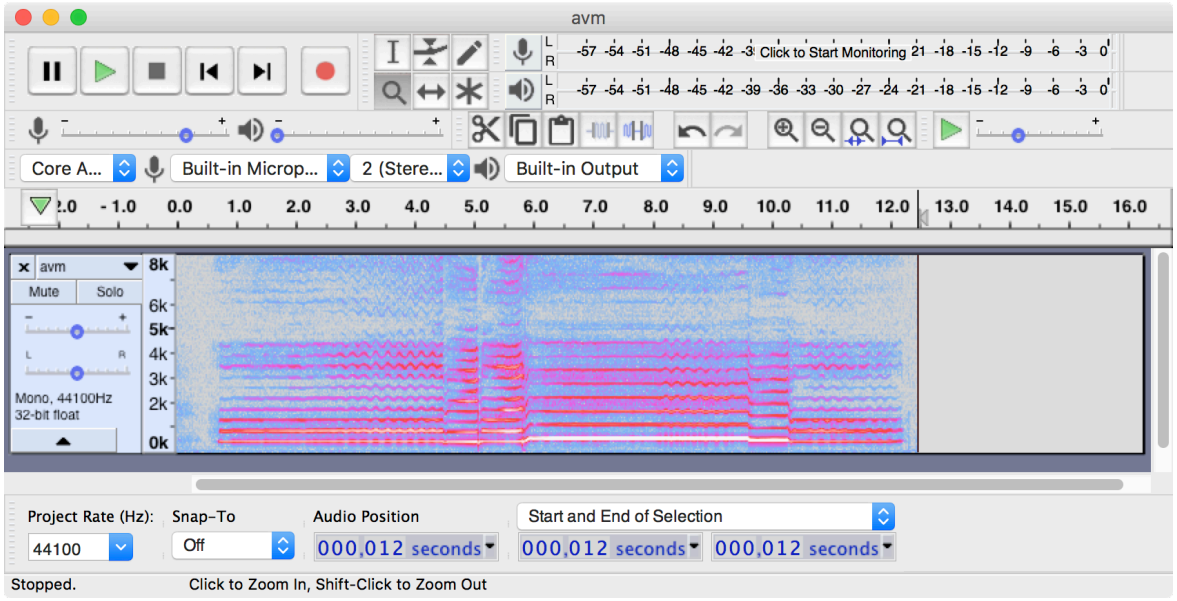
Audacity

Representations

Time-domain

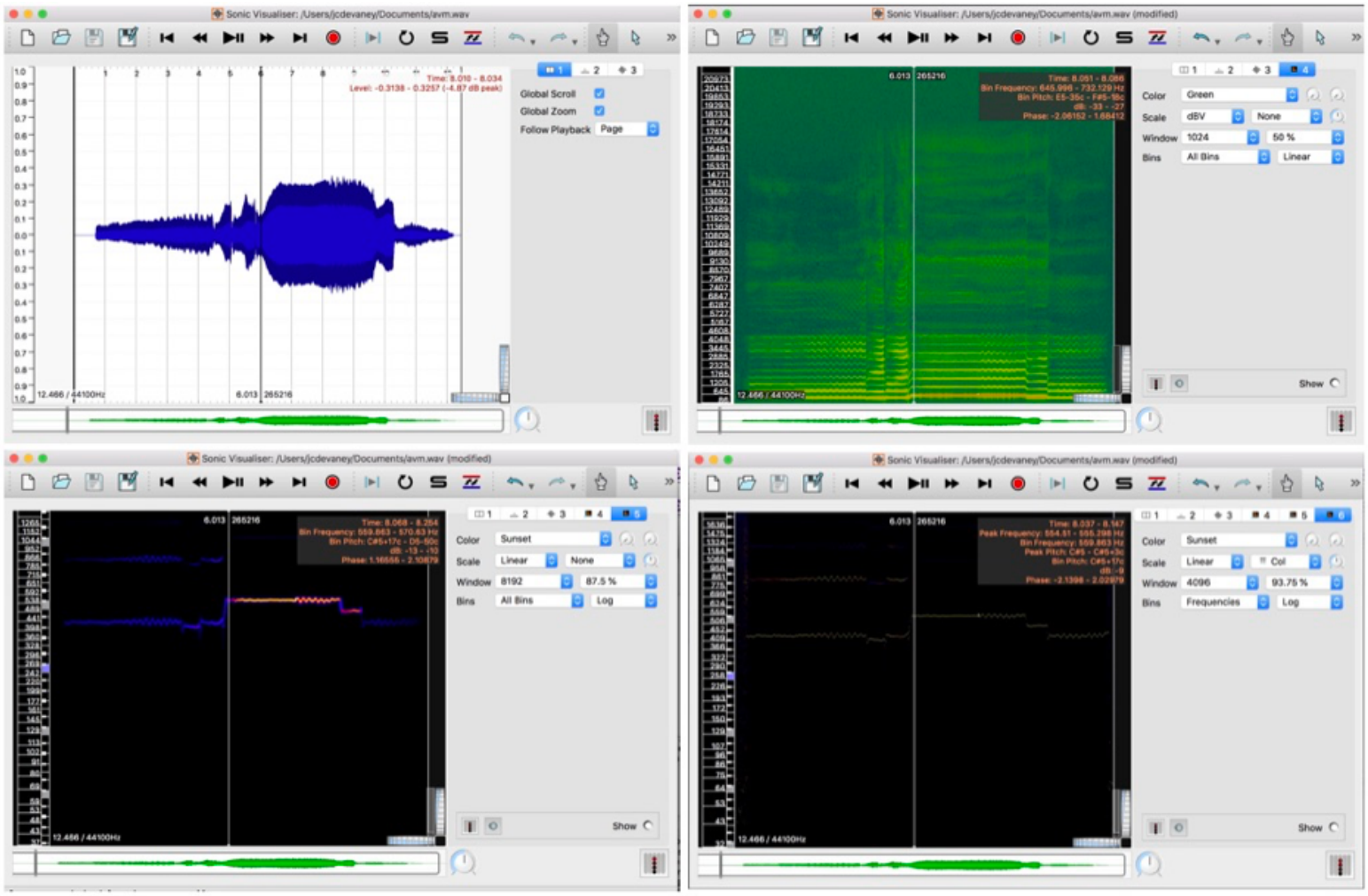


Frequency-domain



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.

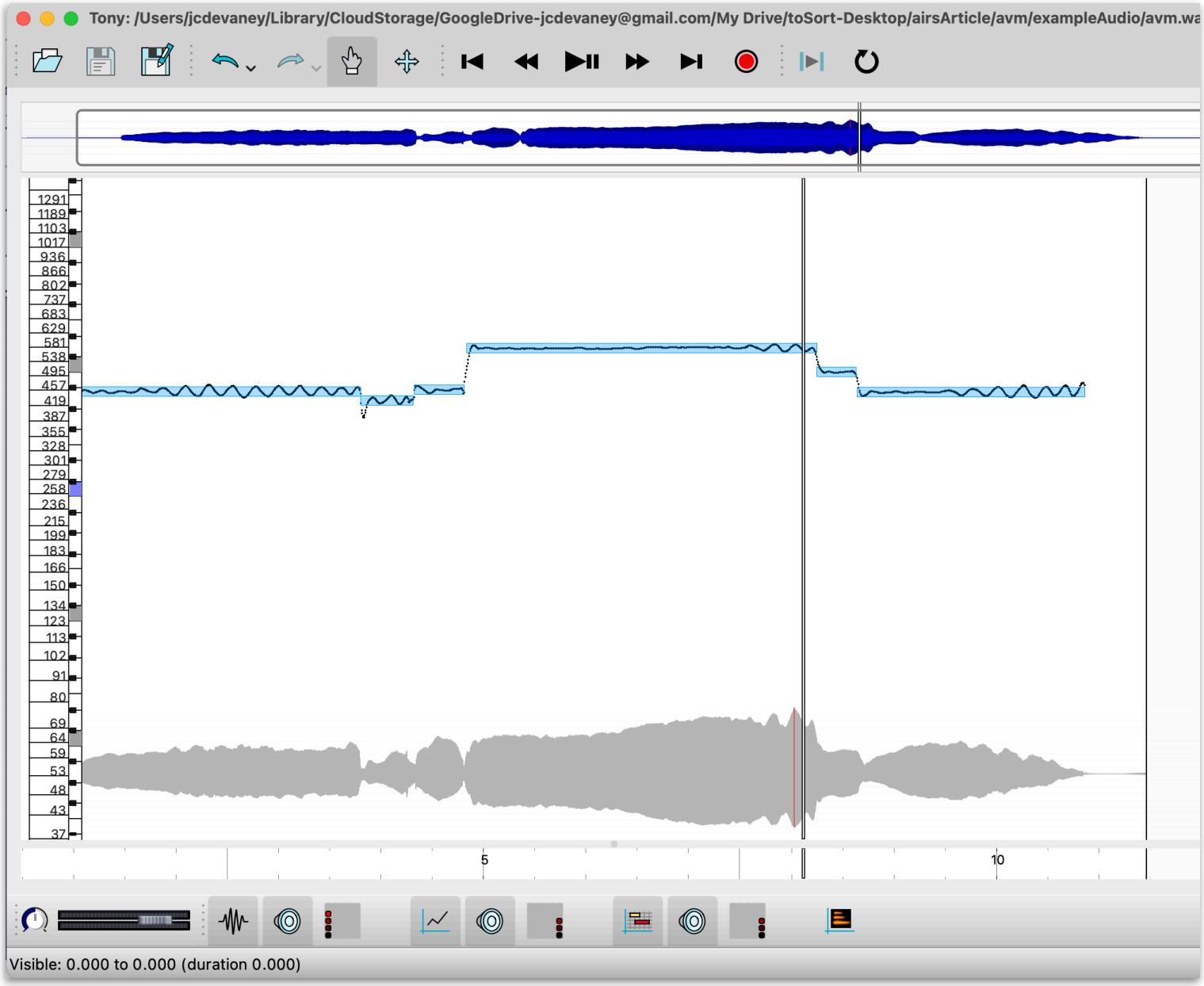
Tony

Representations

TONY

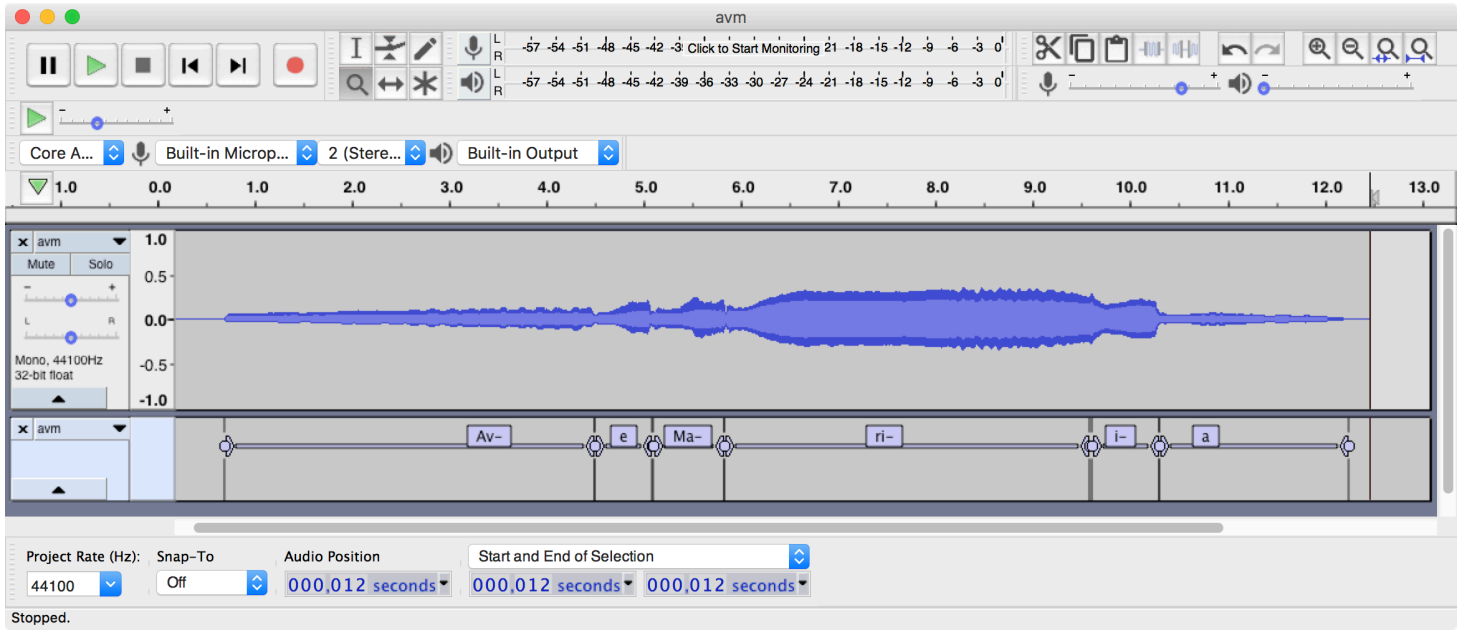
Frequency-domain

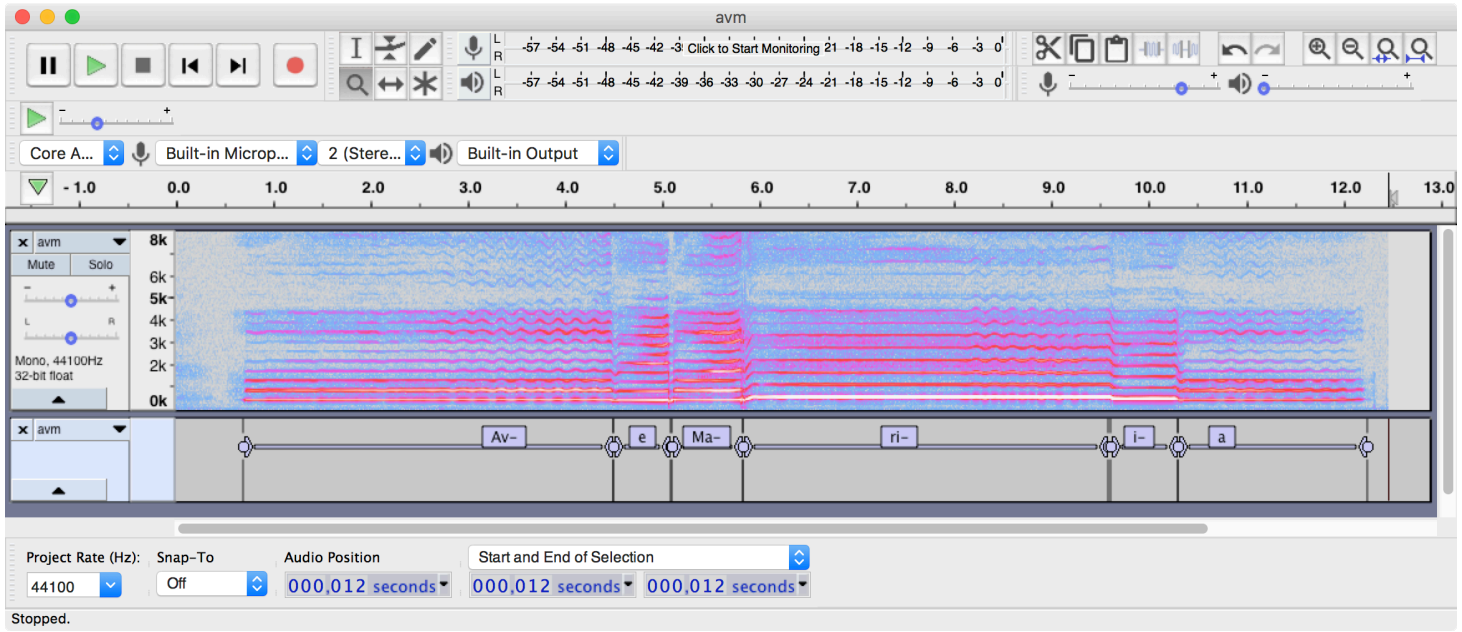
Time-domain



Audacity

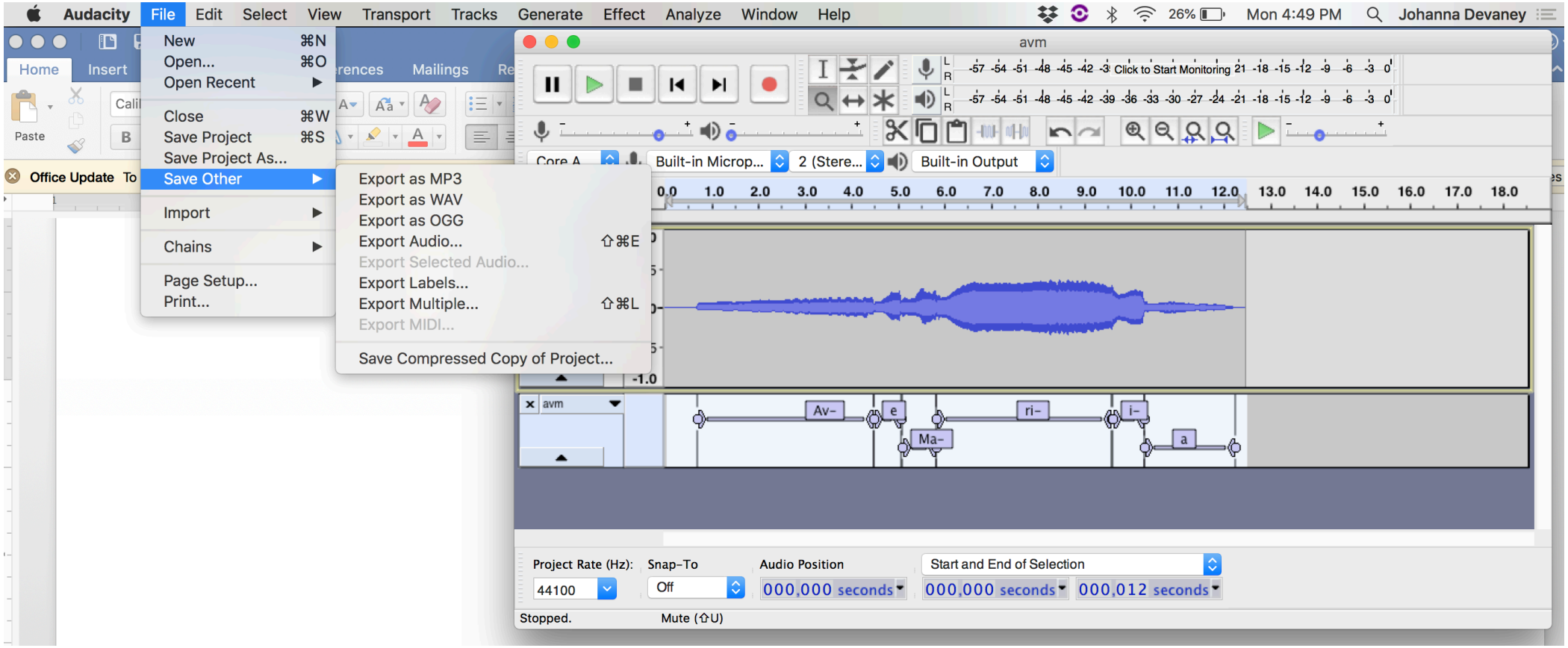
Annotations





Audacity

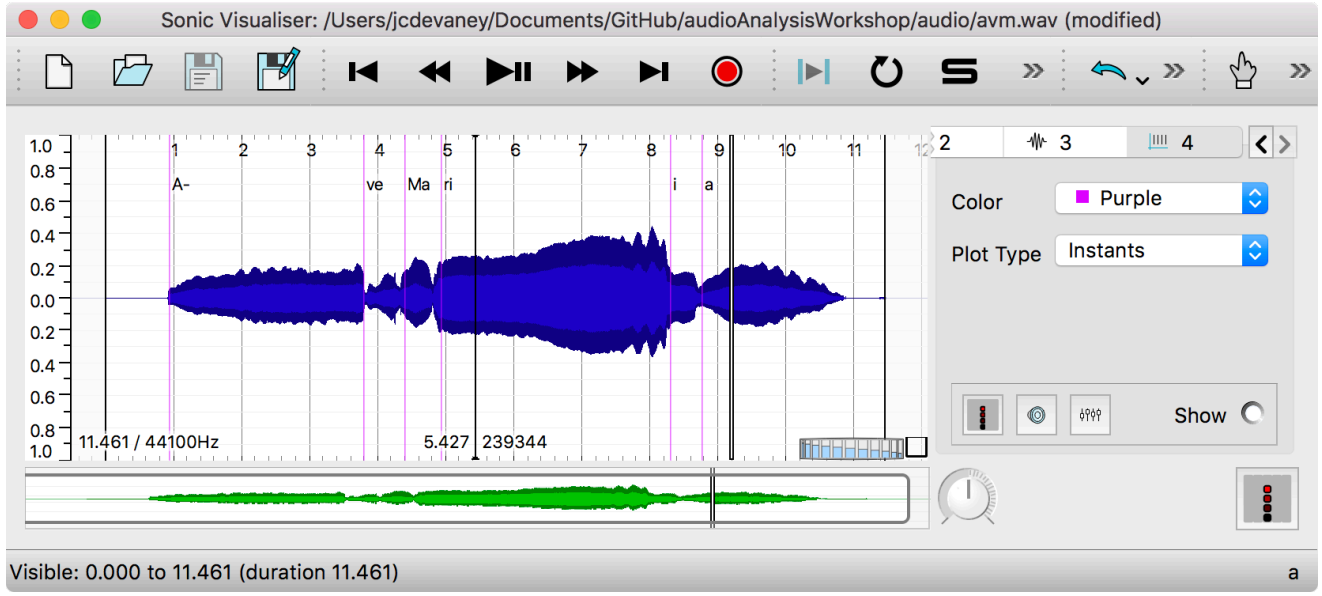
Annotations



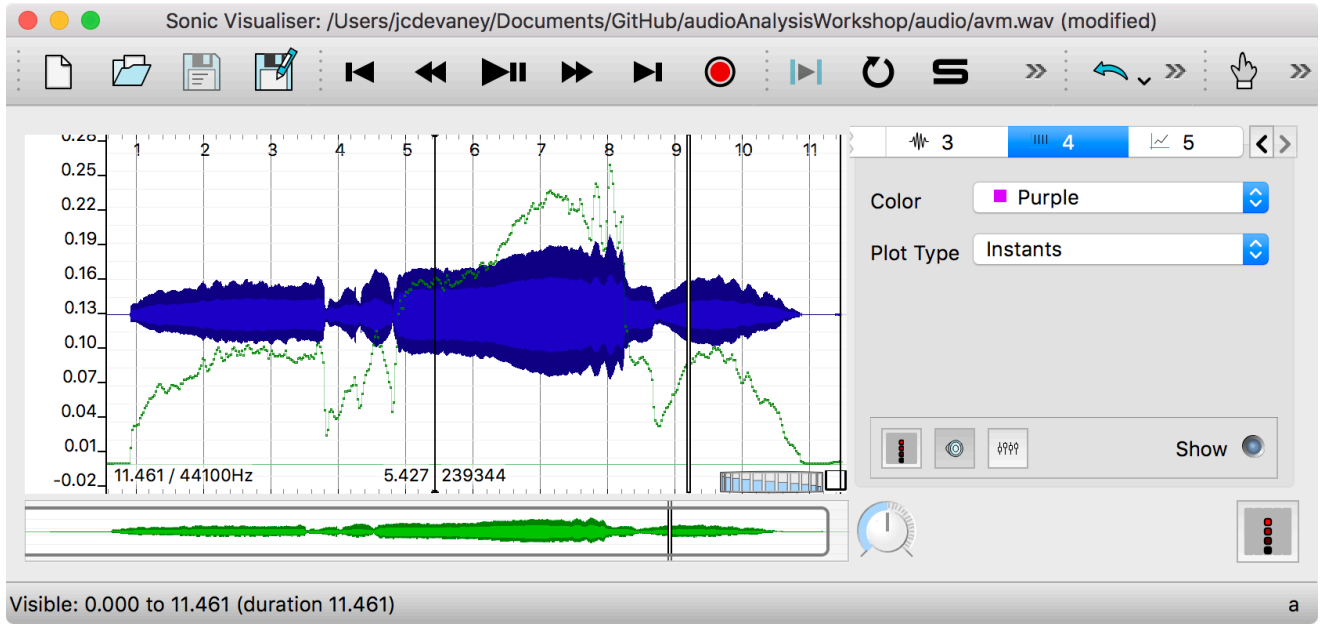
Sonic Visualiser

Annotations

Time Instants
(time points)



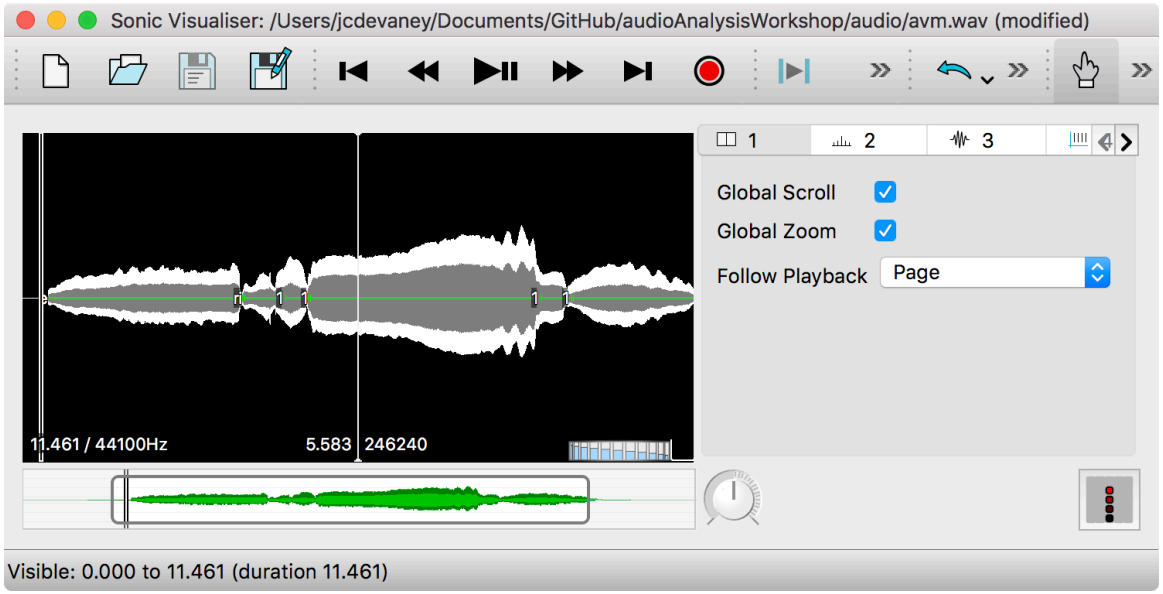
Time Values
(time points + values)



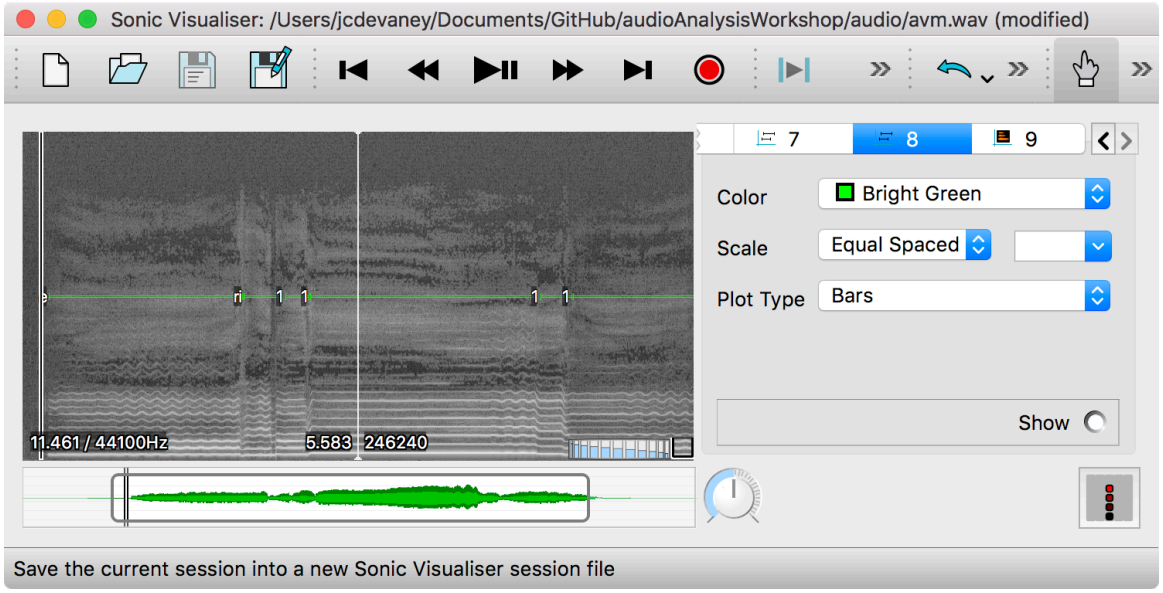
Sonic Visualiser

Annotations

Regions (Time-domain overlay)

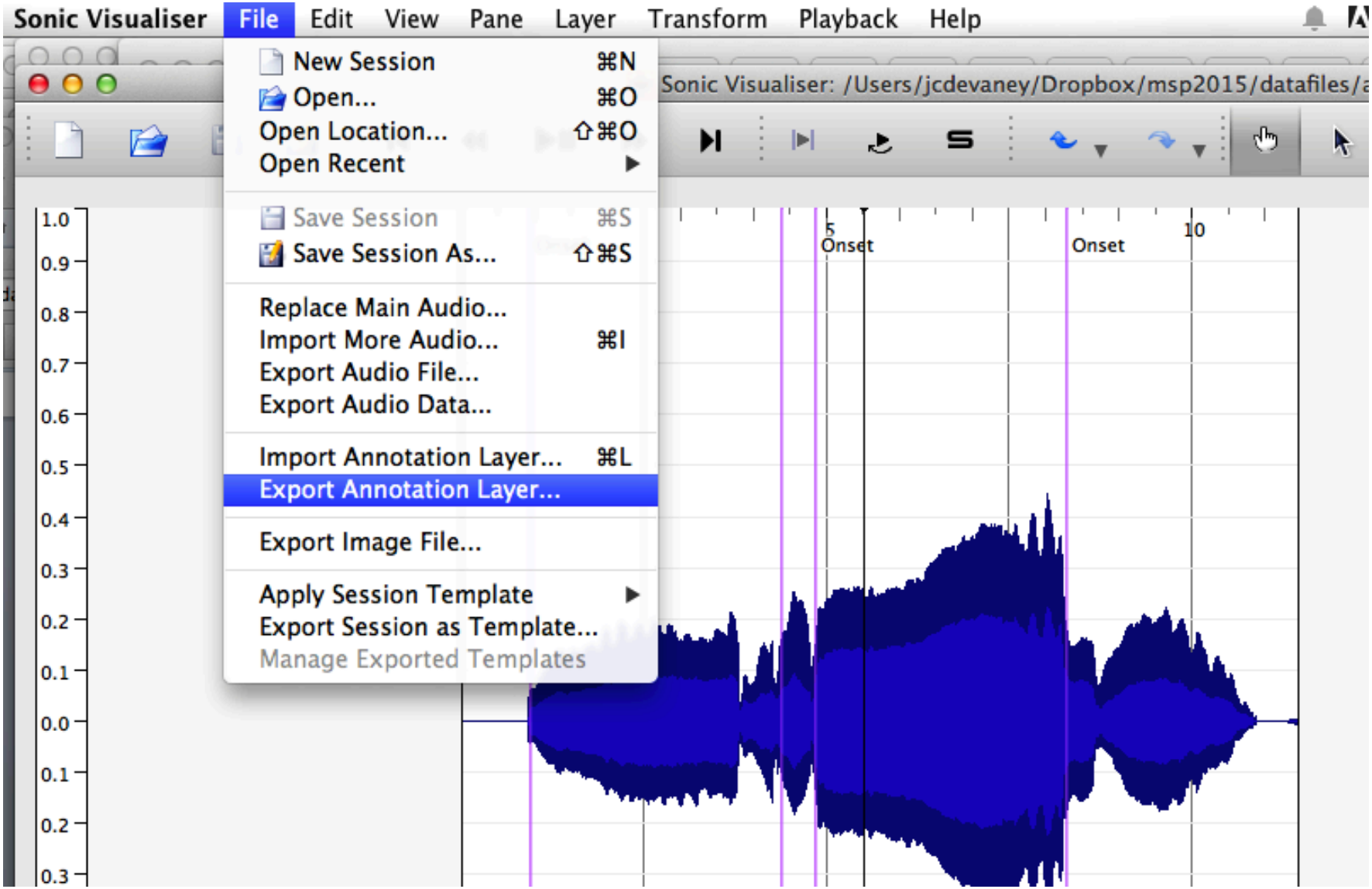


Regions (Frequency-domain overlay)



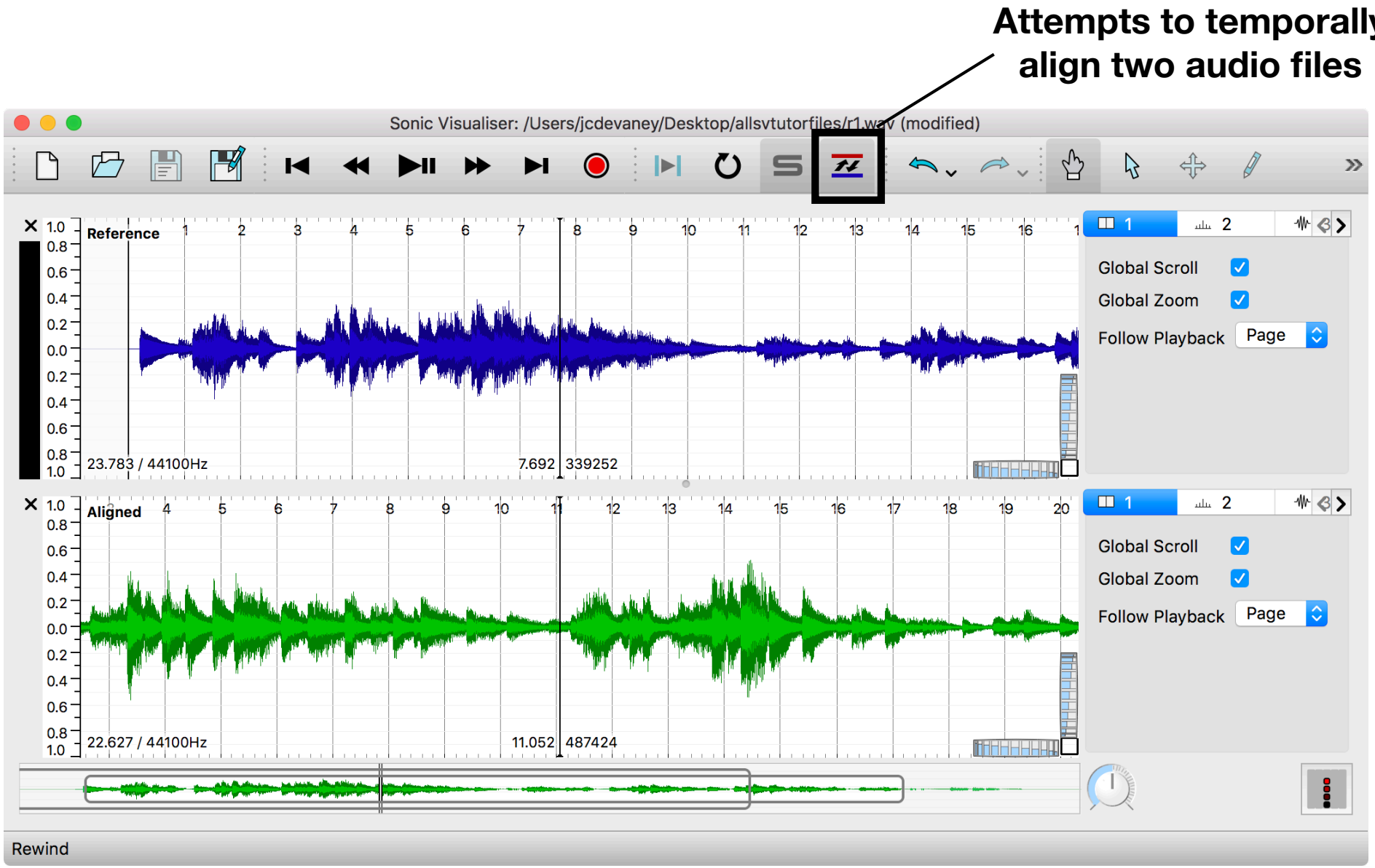
Sonic Visualiser

Annotations



Sonic Visualiser

Time Alignment

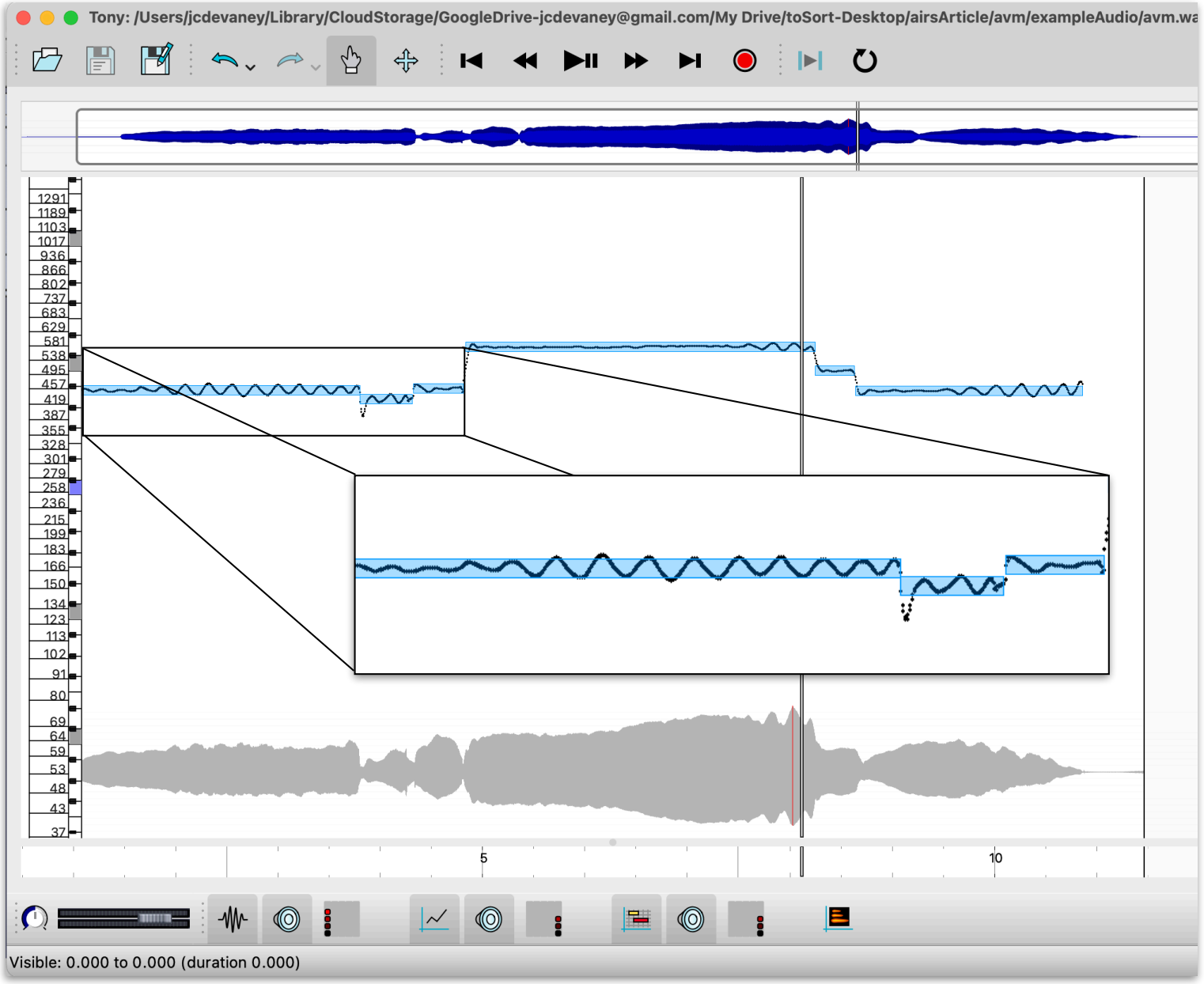


Tony

Annotations

TONY

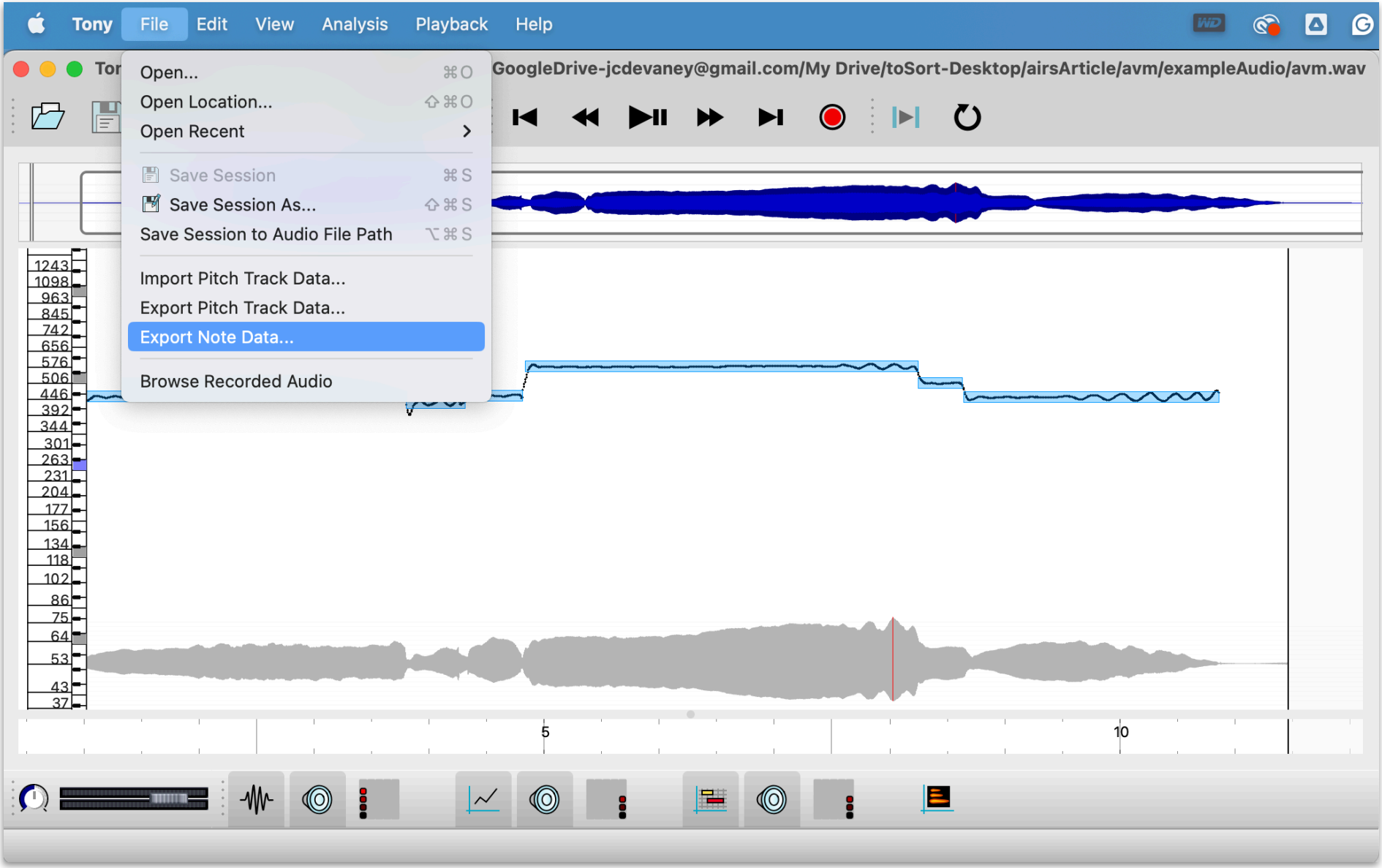
Automatically note segmentation estimation can be manually adjusted



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

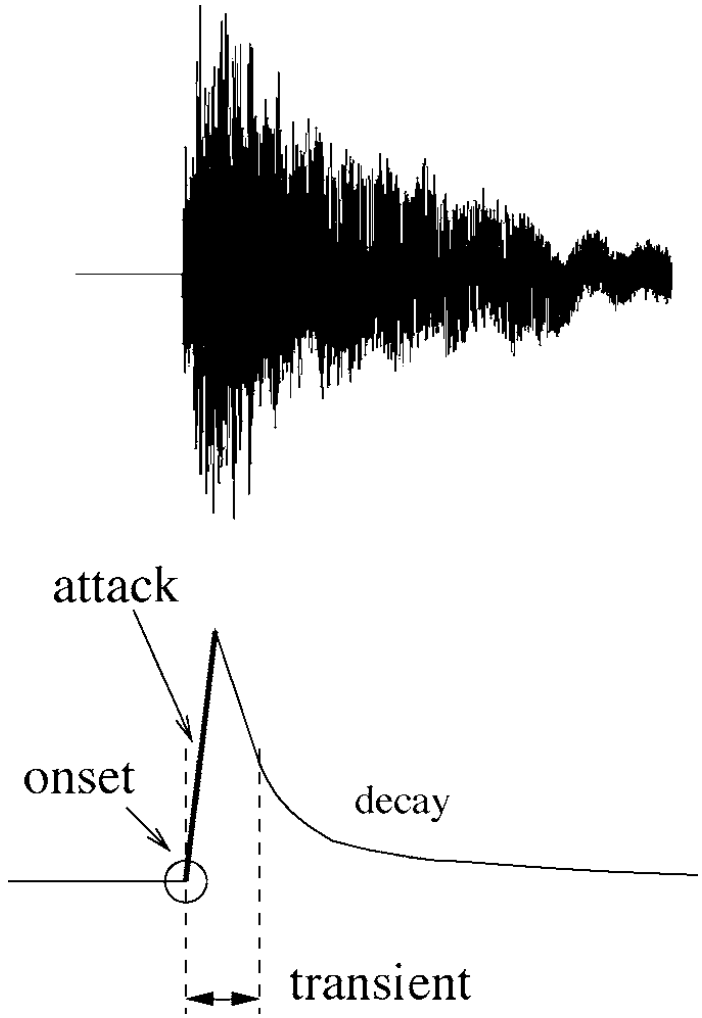


Fig. 1. "Attack," "transient," "decay," and "onset" in the ideal case of a single note.

Bello et al. (2015)

Onset Detection

Theory

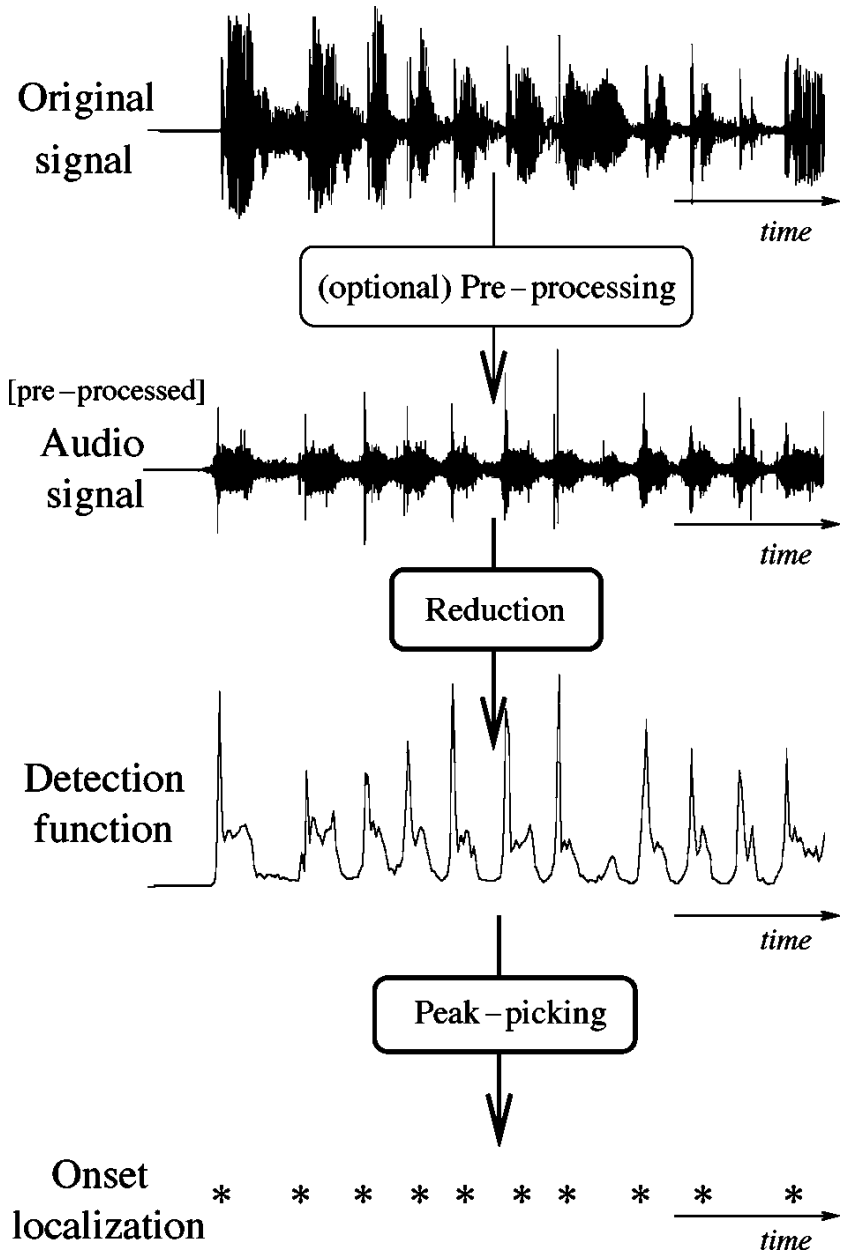


Fig. 2. Flowchart of a standard onset detection algorithm.

Bello et al. (2015)

Onset Detection

Theory

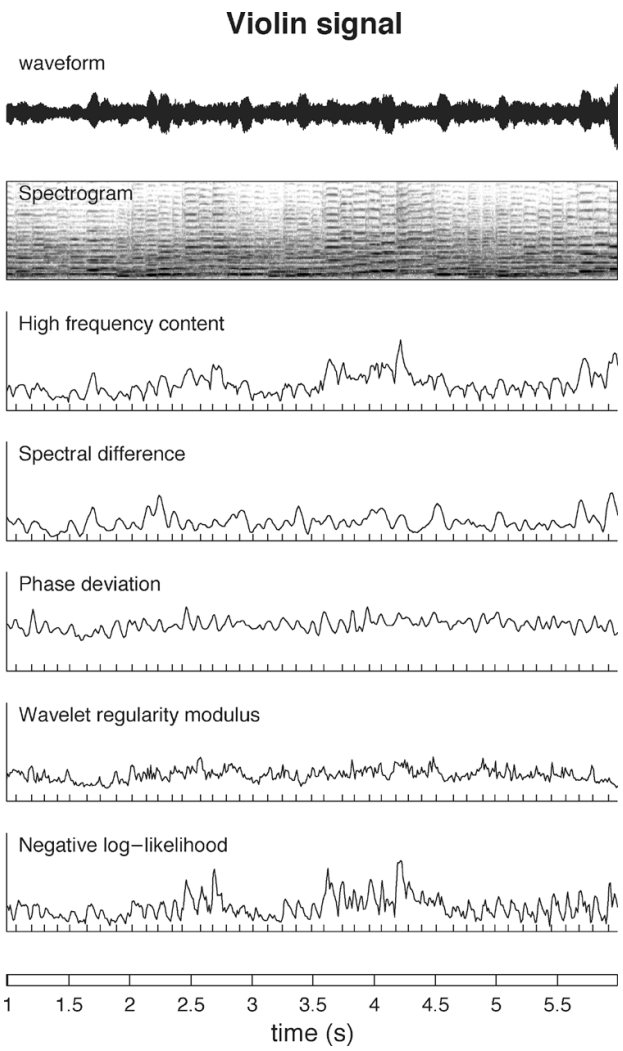


Fig. 4. Comparison of different detection functions for 5 s of a solo violin recording. From top to bottom: time-domain signal, spectrogram, high-frequency content, spectral difference, spread of the distribution of phase deviations, wavelet regularity modulus, and negative log-likelihood using an ICA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

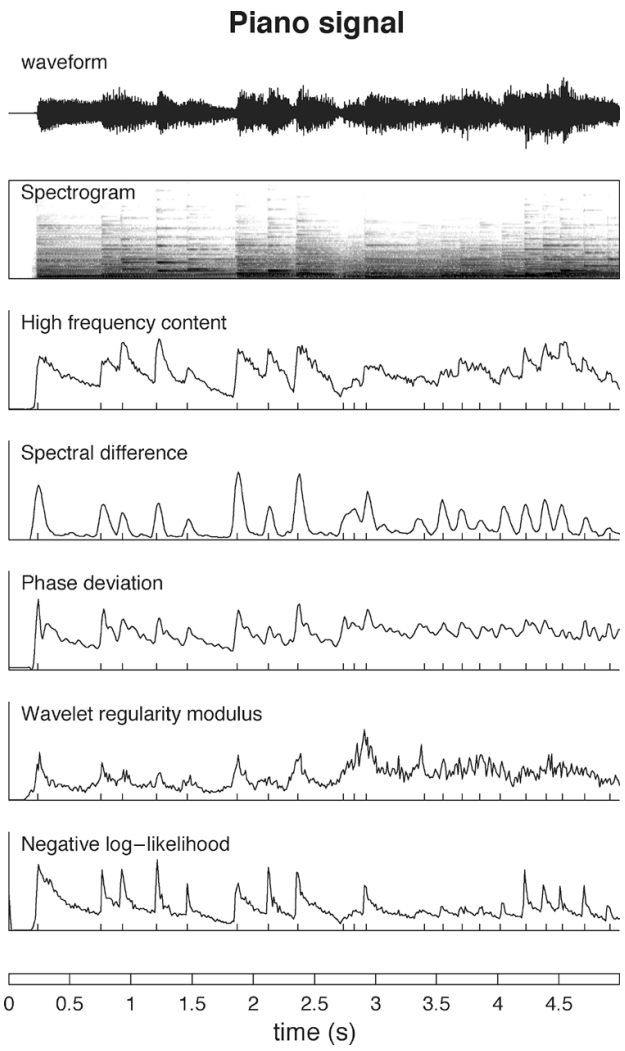


Fig. 5. Comparison of different detection functions for 5 s of a solo piano recording. From top to bottom: time-domain signal, spectrogram, high-frequency content, spectral difference, spread of the distribution of phase deviations, wavelet regularity modulus, and negative log-likelihood using an ICA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

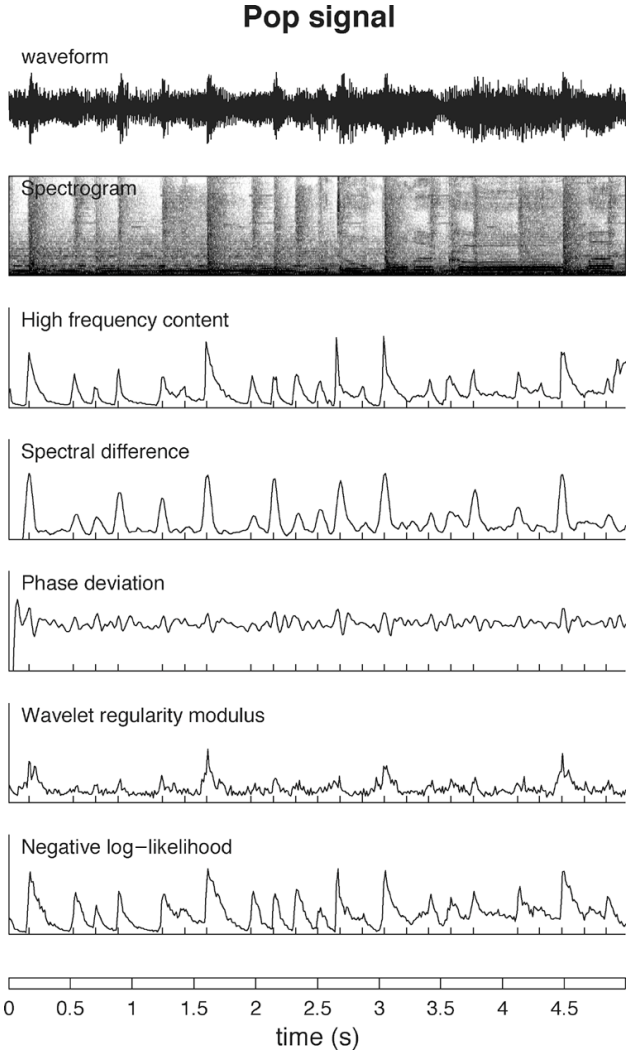


Fig. 6. Comparison of different detection functions for 5 s of a pop song. From top to bottom: time-domain signal, spectrogram, high-frequency content, spectral difference, spread of the distribution of phase deviations, wavelet regularity modulus, and negative log-likelihood using an ICA model. All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

Bello et al. (2015)

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

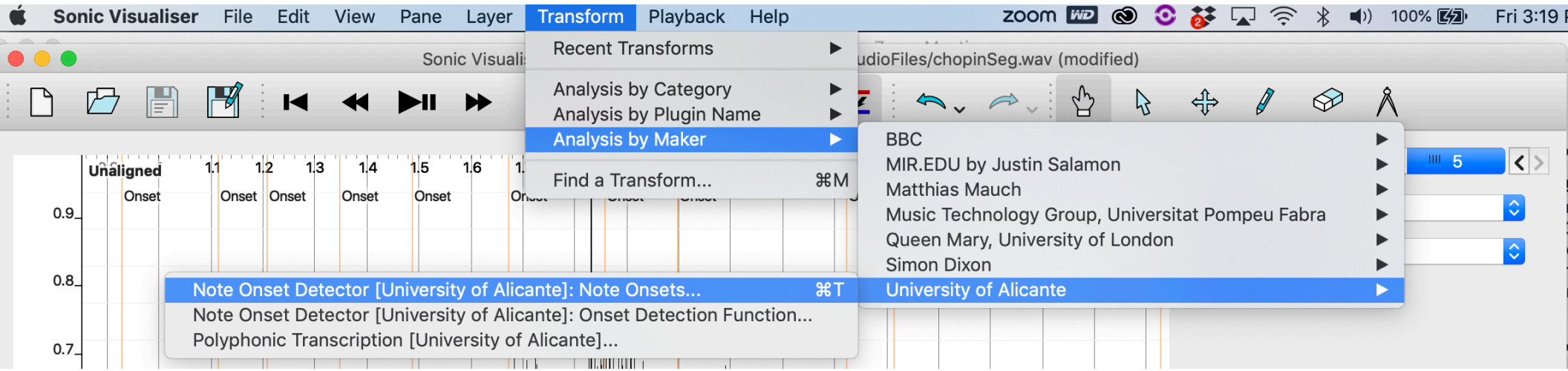
Sonic Visualiser

- **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…)**



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/XXX -t rms.n3 -w csv

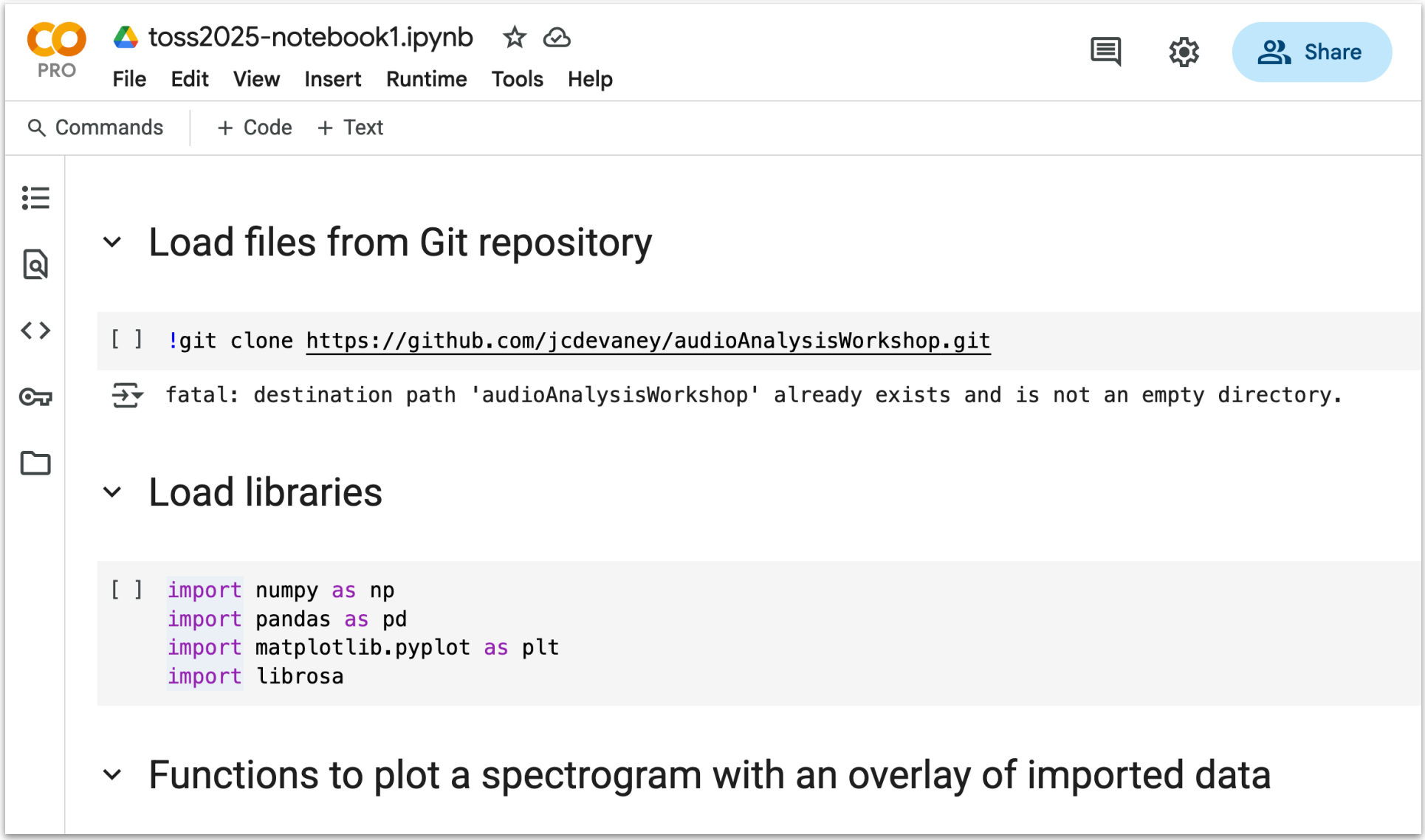
Python-based Coding

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

Visualizing Imported Data

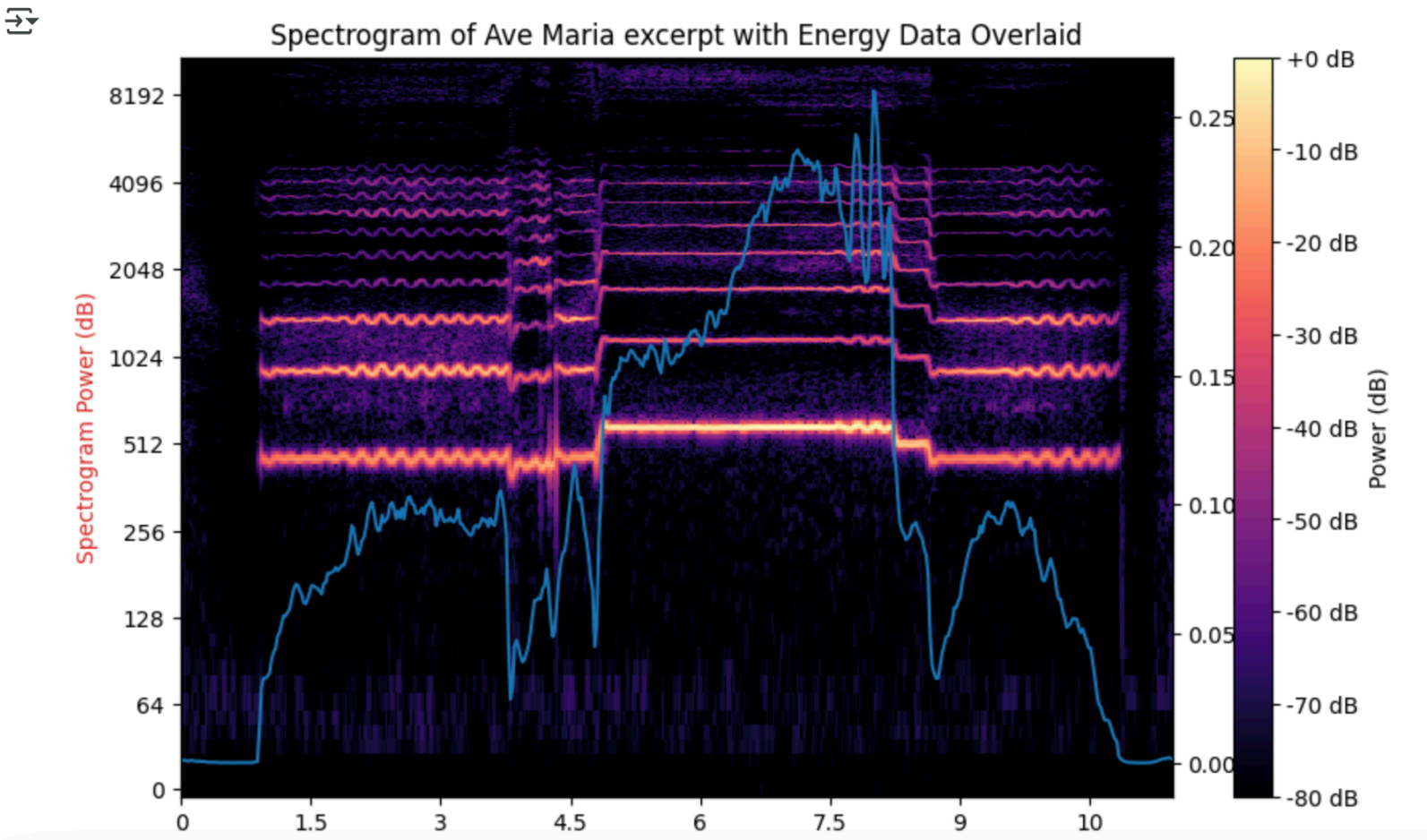


toss2025-notebook1.ipynb

Google Colab

Visualizing Imported Data

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



toss2025-notebook1.ipynb

Google Colab Exercise

Visualizing 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

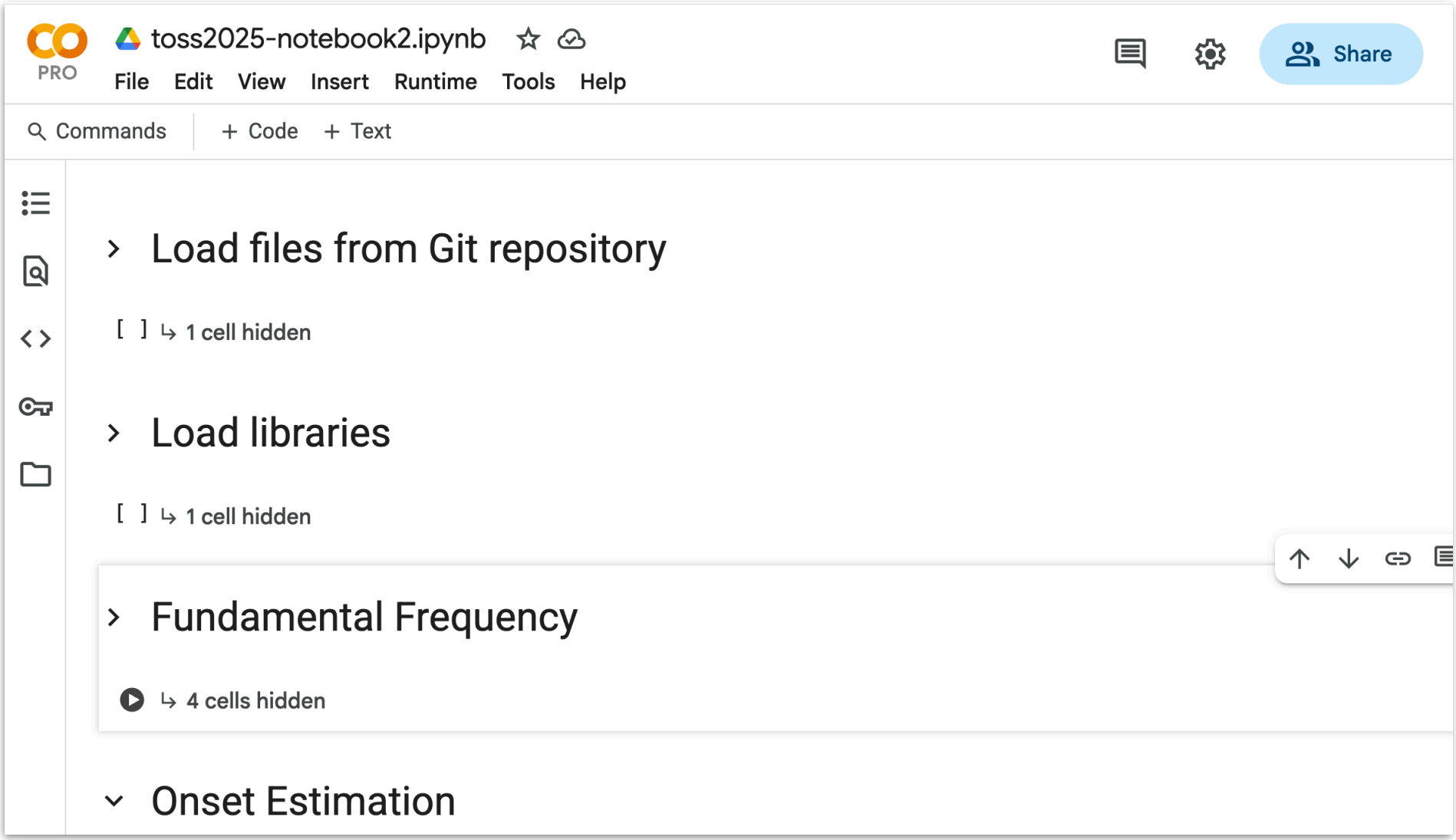
librosa

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.

librosa

Example



toss2025-notebook2.ipynb

librosa Exercise

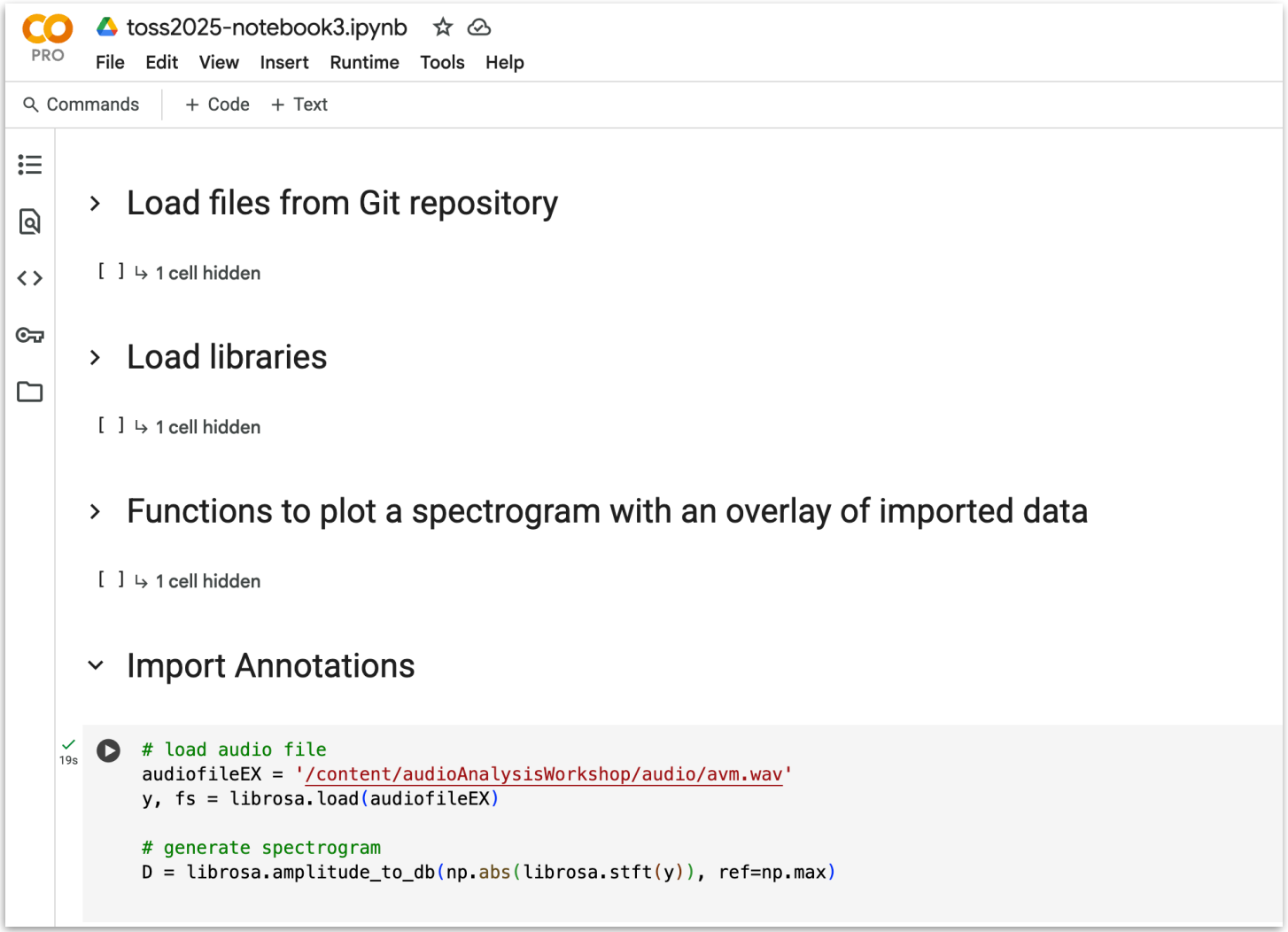
Estimate Acoustic Descriptors

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

toss2025-notebook2.ipynb

librosa

Import Annotations
Calculate Note-wise Summaries



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**

toss2025-notebook3.ipynb

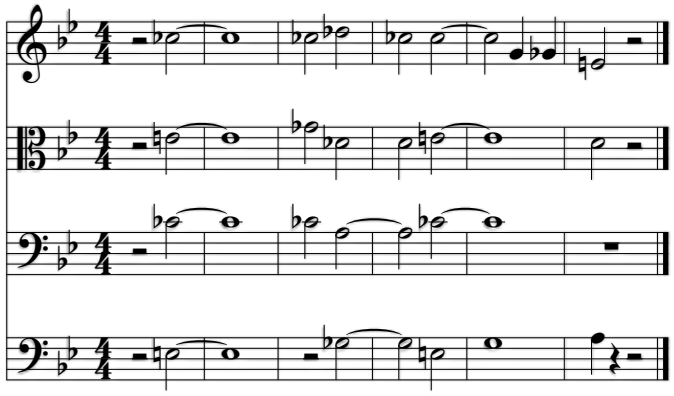
pyAMPACT

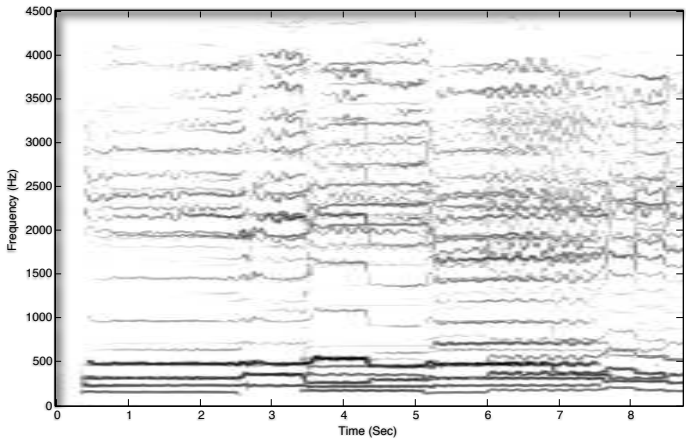
Overview

- **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

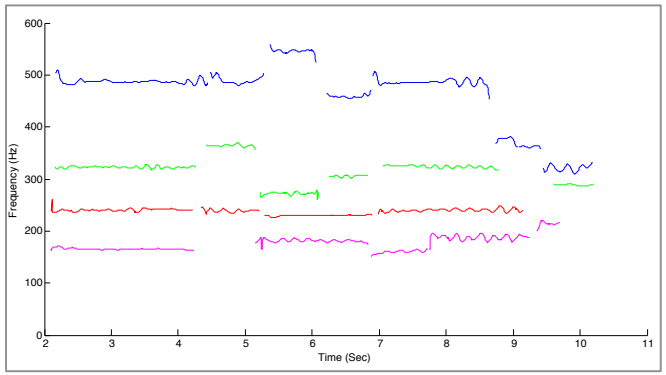
Score-Audio Alignment



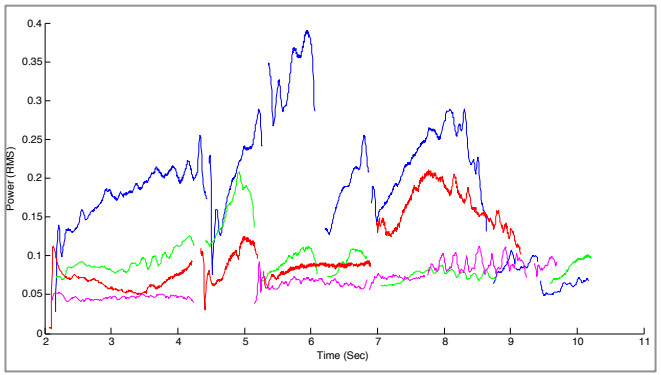


Alignment

Note-wise f_0 estimations

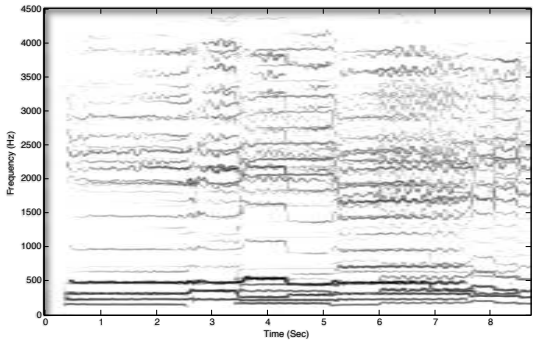


Note-wise power estimations

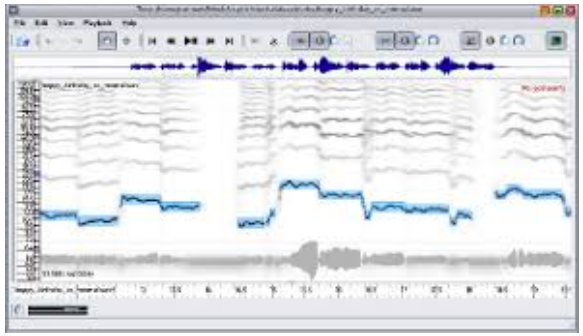


pyAMPACT

Annotation-based Usage



Source Separation

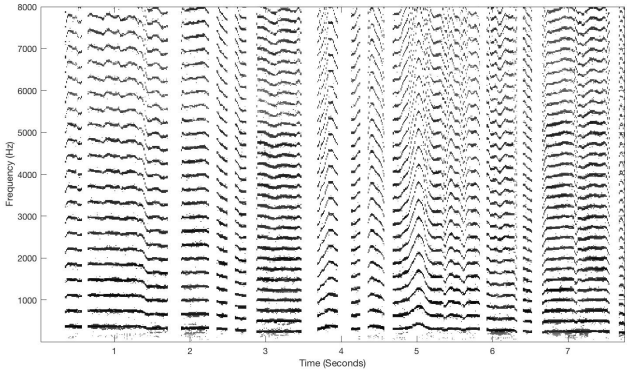


Tony

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

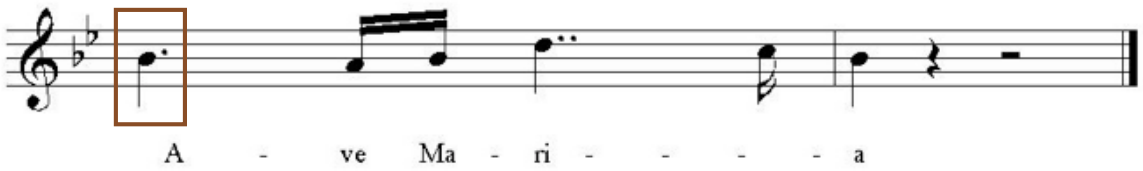
Also Extensible to Other Descriptors

pyAMPACT

Data Export (In Development)

```xml
<?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>
    </music>
</mei>
```

```xml
<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>
```

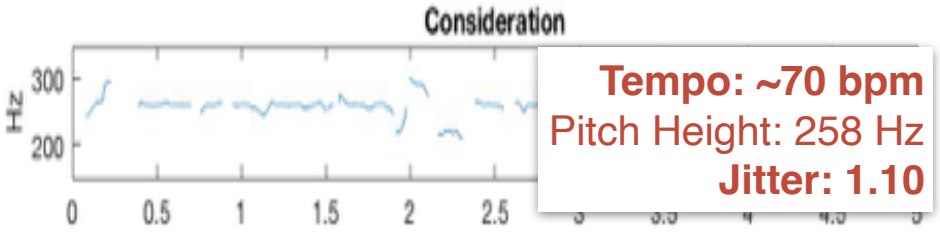


pyAMPACT Example

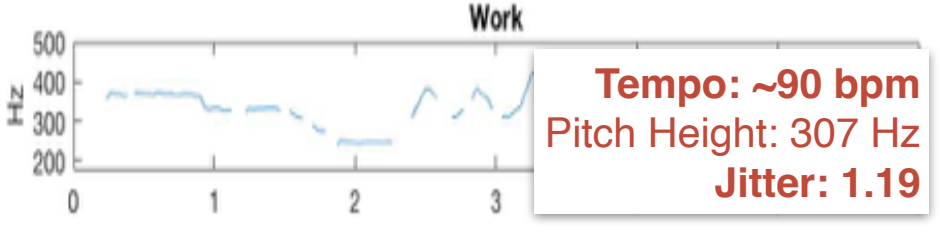
Pitch and Timbre in Rihanna's *Anti* album

Speech-Like

Emotional/Raw

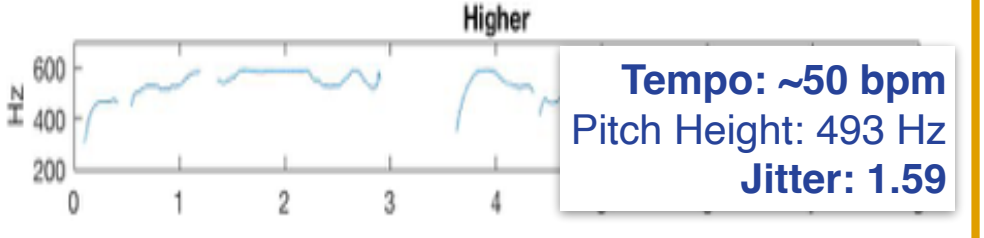


Tempo: ~70 bpm
Pitch Height: 258 Hz
Jitter: 1.10

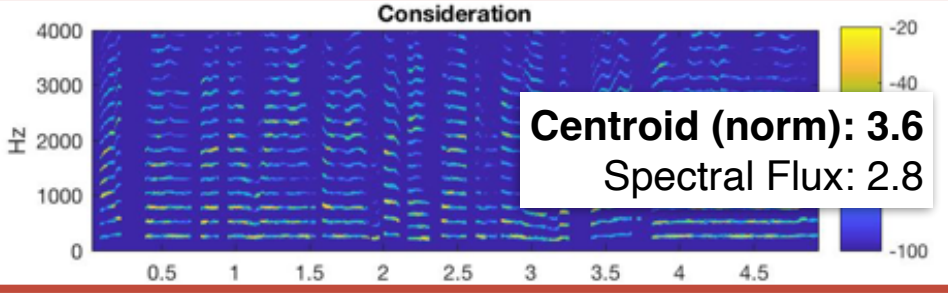


Tempo: ~90 bpm
Pitch Height: 307 Hz
Jitter: 1.19

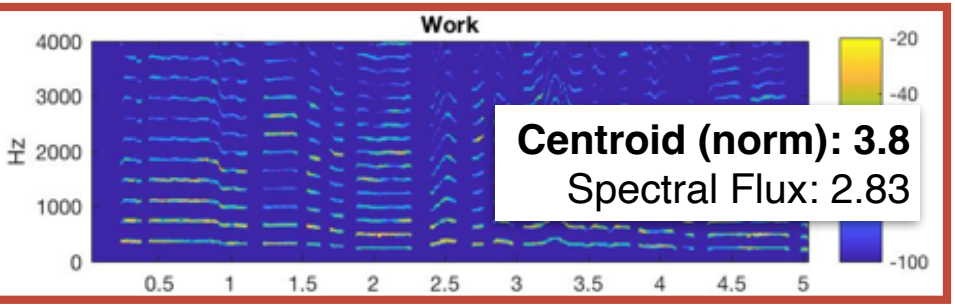
Tempo: ~55 bpm
Pitch Height: 319 Hz
Jitter: 1.45



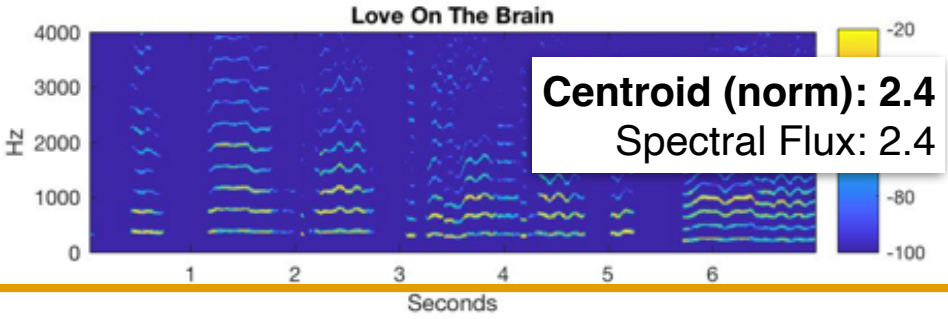
Tempo: ~50 bpm
Pitch Height: 493 Hz
Jitter: 1.59



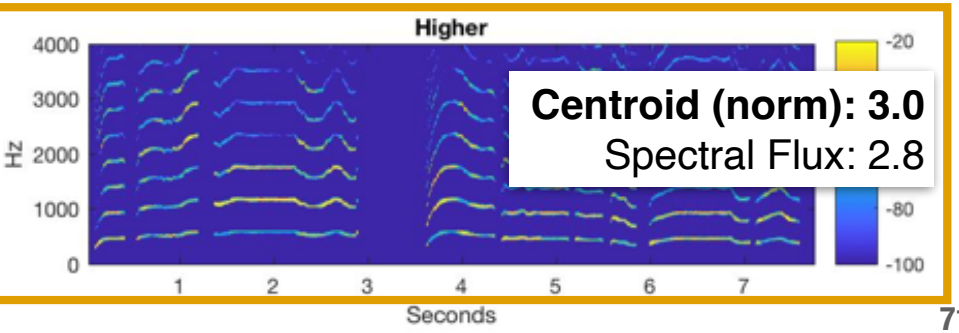
Centroid (norm): 3.6
Spectral Flux: 2.8



Centroid (norm): 3.8
Spectral Flux: 2.83



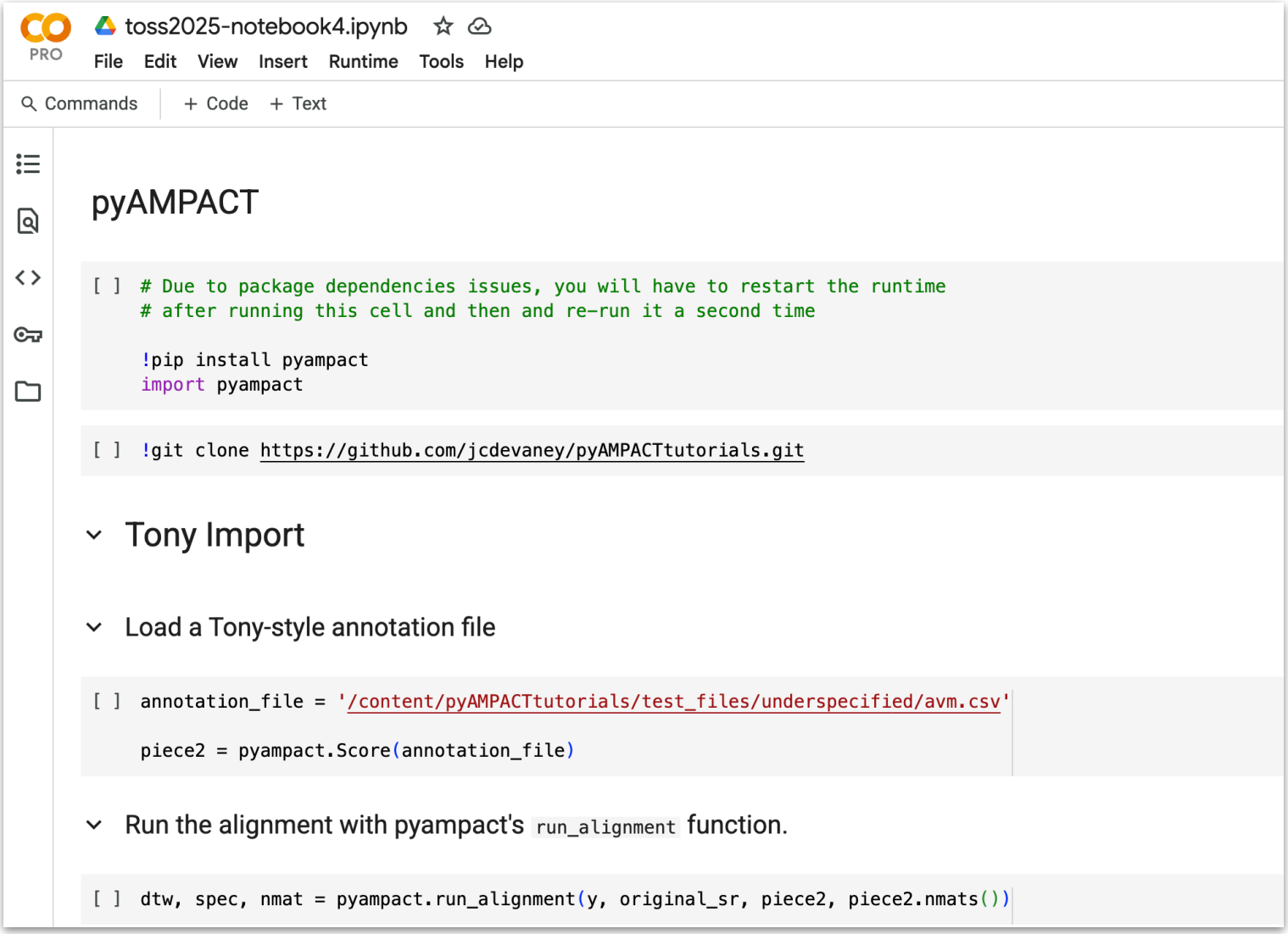
Centroid (norm): 2.4
Spectral Flux: 2.4



Centroid (norm): 3.0
Spectral Flux: 2.8

pyAMPACT

Example



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

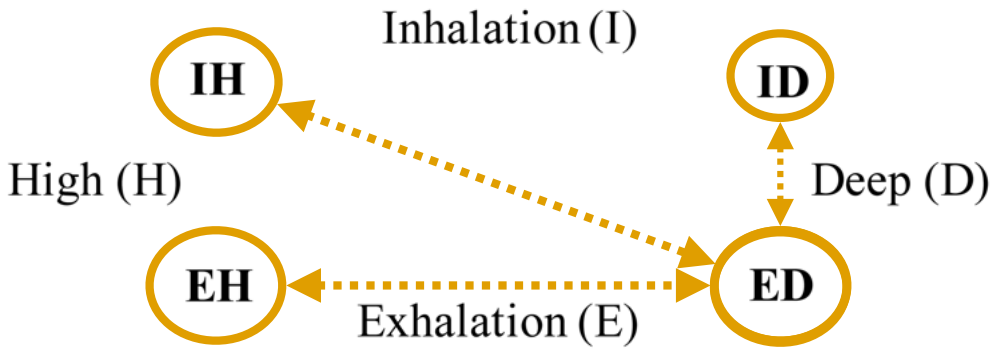
Other Research Examples

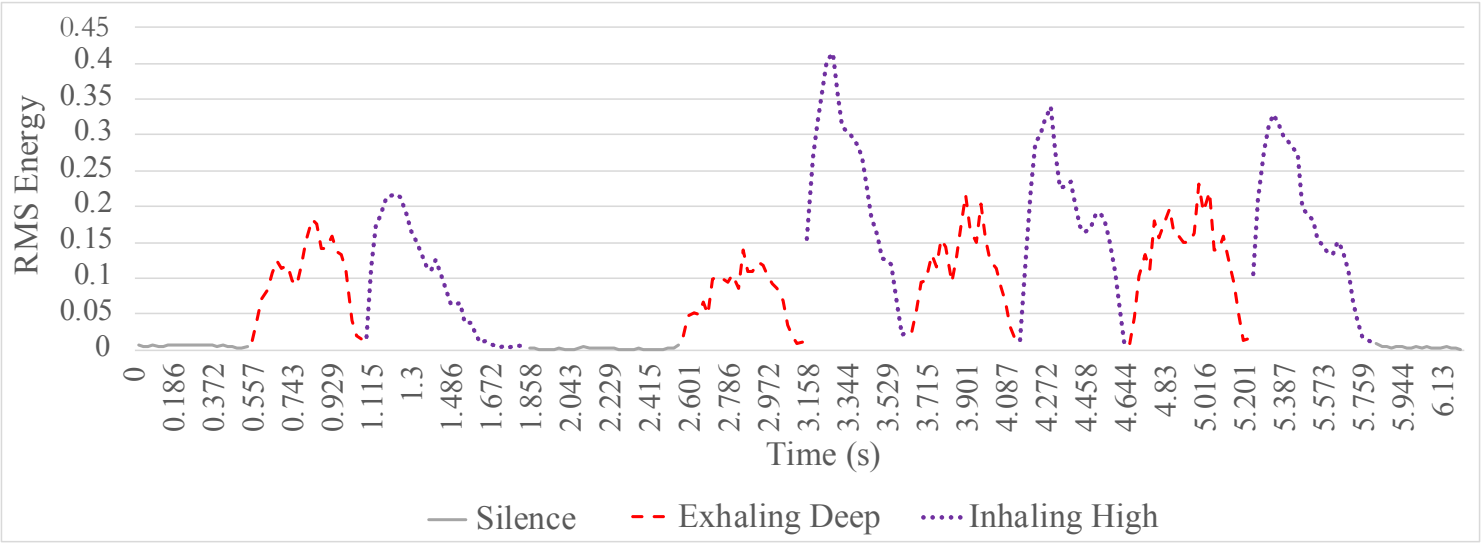
Recent Work by Former Advisees

- **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)





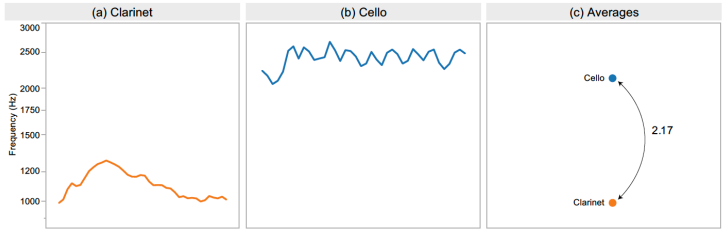
Hardman (2022)

Relative Brightness (Spectral Centroid Quotient)

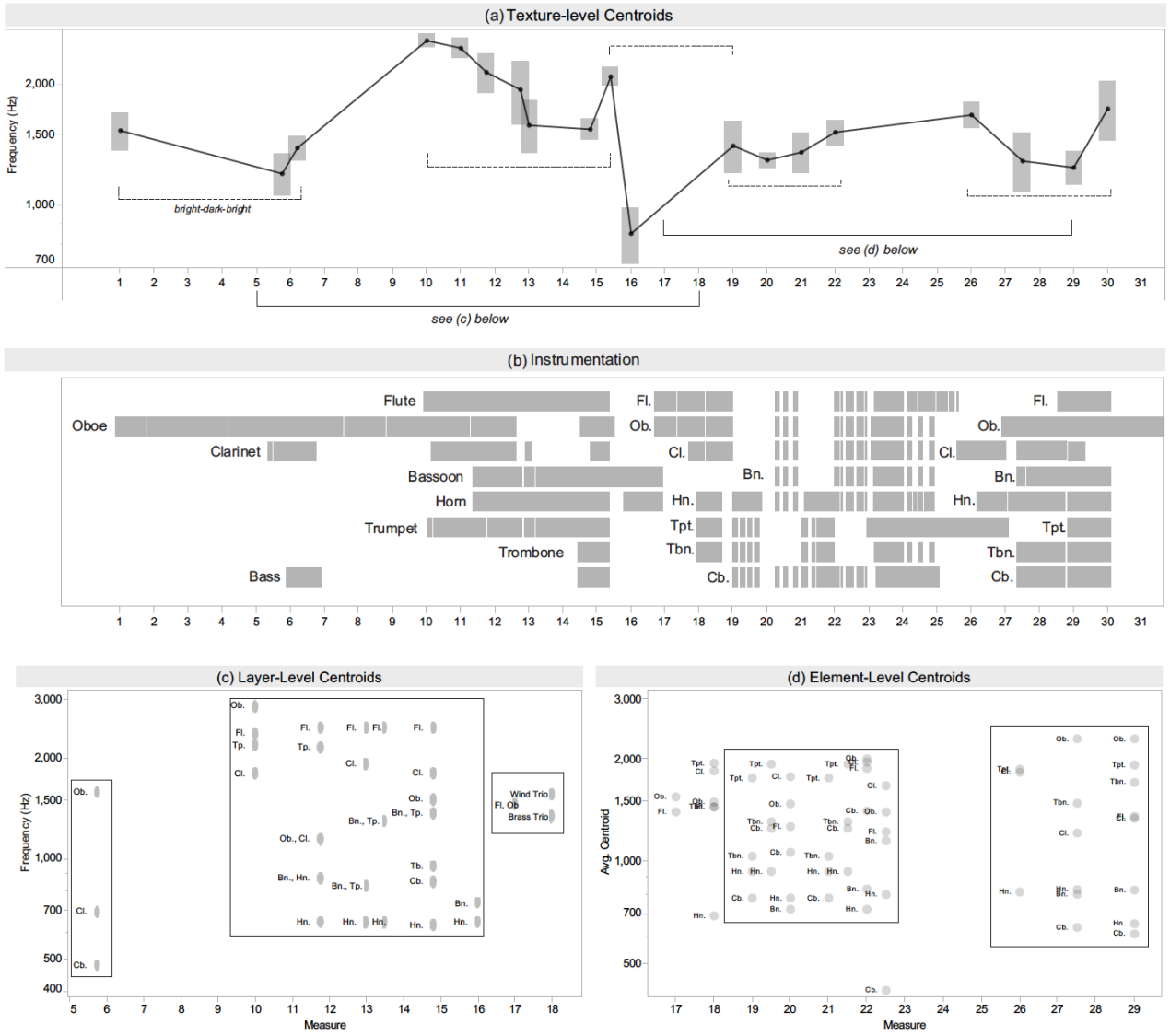
Stephen Spencer (Hunter College)

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

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



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

	Visualize	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)**
  - 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)**
  - Audio files
  - Colab notebooks
  - PDFs of papers for further reading
  - Copy of Sonic Annotator for easy installation
  - Copy of these slides

Thank you!