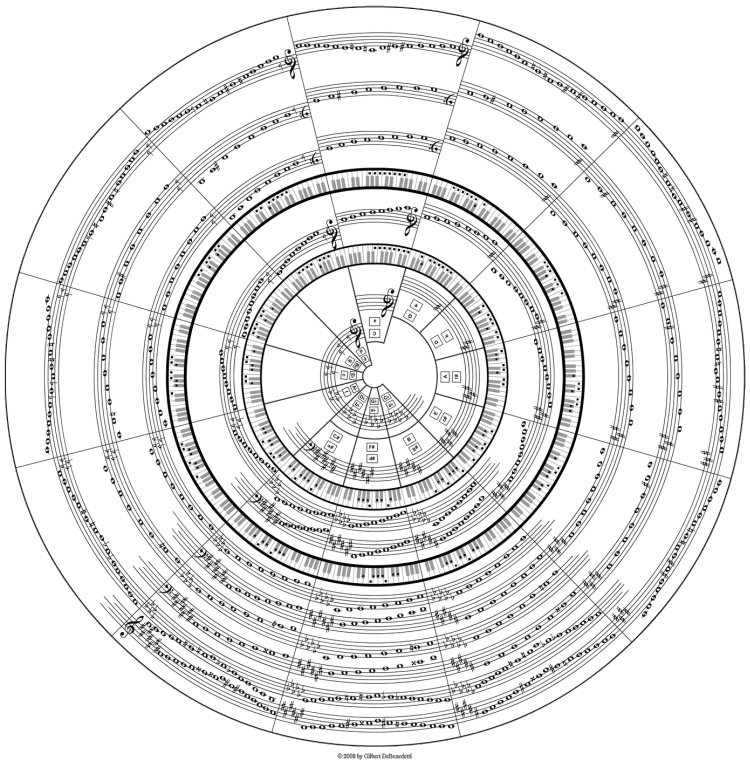
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN NORMANDIE

UNIVERSITÉ DE ROUEN NORMANDIE

Masters Thesis Project

Automatic Music Transcription

Audio Signal Processing



Rand ASSWAD
Department of Applied Mathematics
Theoretical and Applied Computer Science

Under the supervision of:
Prof. Natalie FORTIER
Prof. Jean-Philippe DUBERNARD

12 March 2020

Contents

Introduction

Music is ubiquitous ever since humans exist. Prehistoric instruments have been found and thought to be at least 40,000 years old. Music is a pilar of human civilisation; it relates to people's identities, feelings and thoughts. Hence, means of saving and sharing music are of invaluable importance. The oldest surviving notated music work *Hurrian Hymn to Nikkal* found on clay tablets dates back to 1400 BC.

Various systems were developped around the globe for *visually* representing perceived music through the use of written symbols. The modern western notation is the predominent musical notation worldwide for most music genres.

With the rise of technology, audio recordings where introduced as analog signals and eventually as digital signals, providing means for sharing and sauveguarding music *aurally*.

Music theory and musical notation have been studied for centuries, allowing humans and machines to retrieve music information from common formats. Nevertheless, *music processing* is a relatively young discipline compared to other subdomains of signal processing such as speech processing; while great results are achieved today in speech recognition, the task of retreiving music information from audio recordings is still far along.

Automatic Music Transcription (AMT) is the task of analyzing musical audio signals and producing the corresponding musical scores. This task has captured researchers interest in the late 20^{th} century and has become a wide research discipline as many of the problems in this domain remain unsolved. furthermore, strides in the domain of AMT would apply to numerous applications that can facilitate creating, sharing, and learning music.

The scope of this thesis is the domain of Automatic Music Transcription and the underlying tasks. We explore the state of the art and propose an implementation for a subset of the presented methods.

1 Background

The focus of this project is music information retrieval from music audio signals. In this section we go through the main problems in the discipline of Automatic Music Transcription, we study characteristics of musical elements, human perception of music, and basic notions of modern music theory. We also review the main characteristics of a sound wave as well as analytic tools for processing digital audio signals. Furthermore, we establish the bridge between music theory and physical properties of audio signals.

1.1 Automatic Music Transcription

> AMT is the process of converting an acoustic musical signal into some form of musical notation. (Benetos et al. 2013)

1.1.1 History and community

The interest in the task of AMT has started in the late 20^{th} century, with researchers borrowing and adapting concepts from the well-established domain of *speech-processing*. Major strides have been made in the 21^{st} century, particularly since the creation of the International Society for Music Information Retrieval **(ISMIR)** in 2000. Which have connected the community and provided a platform for sharing and learning Music Information Retrieval **(MIR)** concepts worldwide. (Müller 2015)

furthermore, the Music Information Retrieval Evaluation eXchange **(MIREX)** is an annual evaluation compaign for MIR algorithms. Since it started in 2005, MIREX has served as a benchmark for evaluating novelty algorithms and helped advance MIR Research.

1.1.2 Motivation

MIR and AMT can be of great interest for different demographics. First, most musicians stand to benefit from reliable transcription algorithms as it can facilitate their tasks in difficult cases or for the least accelerate the process.

Moreover, in many music genres such as Jazz, musical notation is rarely used, therefore the exchange formats are almost exclusively recordings of performances. AMT would play a role in democratizing no-score music for new learners and provide an easier canonical format for exchanging music.

Another use of MIR is score-following software development that include a cursor that follows real-time playing indicating the correct and incorrect notes played helping pupils practice and progress on their own more efficiently, making the task of music learning less painful.

Furthermore, MIR allows performing musicological analysis directly on recordings, gaining access to much larger databases compared to anotated music, which can also be applied for various tasks such as music recognition or melody recognition.

1.1.3 Underlying tasks

Automatic Music Transcription is divided into several subtasks where each represents a research topic that fall within the scope of Musical Information Retrieval.

The largest topic of MIR is tonal analysis, which is based on analysing spectral features of audio signals, and subsequently estimating *pitch*, melody and harmony. Despite the large interest in this topic and various techniques applied, this task remains the core problem in AMT, Exploration of main pitch analysis techniques is the first part of this project.

Another main AMT task is *temporal segmentation*, which relates consequently to rythme extraction and tempo detection in melodic sounds. (Benetos et al. 2013) This task pertains pertains to spectral features as well as signal energy. We expore this topic in the second part of this project.

Several more tasks are needed to fully transcribe a musical piece, including: loudness estimation, instrument recognition, rhythm detection, scale detection and harmony analysis. In the scope of this project, we limit our study to *pitch analysis* and *temporal segmentation*.

1.2 Physical definition of acoustic waves

Sound is generated by vibrating objects, these vibrations cause oscillations of molecules in the medium. The varying pressure propagates through the medium as a wave, the pressure is therefore the solution of the wave equation in time and space, also known as the acoustic wave equation. (Feynman 1965)

$$\Delta p = \frac{1}{c^2}\frac{\partial^2 p}{\partial t^2}$$

where p is the accoustic pressure function of time and space and c is the speed of sound propagation. The wave equation can be solved analytically with the separation of variables method, resulting in a *sinusoidal harmonic* solutions.

In audio signal processing, we are interested in the pressure at the receptor's position (listener or microphone), hence the pressure as a function of time. An audio signal is therefore defined as the deviation of pressure from the average pressure of the medium at the receptor's position.

The pressure function being harmonic, the sound signal is of the form

$$\tilde{x}(t) = \sum_{h=0}^{\infty} A_h \cos(2\pi h f_0 t + \varphi_h)$$

where

- f_0 is called the **fundamental frequency** of the signal,
- h is the harmonic number,
- A_h is the amplitude of the h^{th} harmonic,
- φ_h is the phase of the h^{th} harmonic.

In many works this formula appears in terms of the angular frequency $\omega = 2\pi f$, we denote as well $f_h = h f_0$ for $h \geq 1$.

As harmonics represent proper multiples of the fundamental frequency, $h = 0$ is excluded from the sum

$$\tilde{x}(t) = a_0 + \sum_{h=1}^{\infty} A_h \cos(2\pi h f_0 t + \varphi_h)$$

with $a_0 = A_0 \cos(\varphi_0)$.

1.3 Perception of sound and music

The human auditory system is capable of distinguishing intensities and frequencies of sound waves as well as temporal features. The inner ear is extremely sensitive to sound wave features, the brain allows further analysis of these features.

Music theory defines and studies *perceived features* of music signals. These features are based on the signal's intensity, frequency, and time patterns.

In music theory, a **note** is a musical symbol that represents the smallest musical object. The note's attributes define the *pitch* of the sound, its *relative duration* and its *relative intensity*.

1.3.1 Fundamental frequency and pitch

Sound signals are periodic, therefore by definition there exists a $T > 0$ such as

$$\forall t, \tilde{x}(t) = \tilde{x}(t + T)$$

which follows that there exists an infinite set of values of $T > 0$ that verify this property, indeed $\forall n \in \mathbb{N}, T' = nT, \tilde{x}(t) = \tilde{x}(t + T')$. We define the period of a signal as the smallest positive value of T for which the property holds. The **fundamental frequency** f_0 is defined formally as the reciprocal of the period. This definition holds for *any* periodic signal, regardless of its form.

In the case of sound wave, the *perception of the fundamental frequency* is referred to as the **pitch**. Pitch is the defined as the *tonal height* of a sound, it is closely related to the fundamental frequency however remaining a *relative musical concept* unlike the f_0 of a signal that is an absolute mathematical value. In fact, the relation between pitch and f_0 is neither bijective nor invariant.

In music theory, pitch is defined on a discrete space unlike the continuous frequency space. Moreover, human perception of frequency is logarithmic hence obtaining the *next pitch* corresponds to the multiplication of the frequency by a certain value r.

Finally, the frequency of the reference pitch A_4 is widely accepted today as $440Hz$ while in the baroque era it was around $415Hz$ and $440Hz$ was the frequency corresponding to A♯ pitch. Even in modern day, variations of the pitch frequency exist in different regions and even different orchestras!

1.3.2 Perception of intensity

Sound intensity is defined physically as the power carried by sound waves per unit area, whereas sound pressure is the local pressure deviation from the ambient atmospheric pressure caused by a sound wave. Human perception of intensity is directly sensitive to sound pressure, it is measured in terms of *sound pressure level* (SPL) which is a logarithmic measure of sound pressure P relative to the atmospheric pressure P_0 measured in decibels dB.

$$\mathrm{SPL} = 20 \log_{10} \left(\frac{P}{P_0} \right) \mathrm{dB}$$

Nevertheless, sensitivity to sound intensity is variable across different frequencies. The subjective perception of sound pressure is defined by a sound's **loudness** which is a function of both SPL and frequency ranging from quiet to loud.

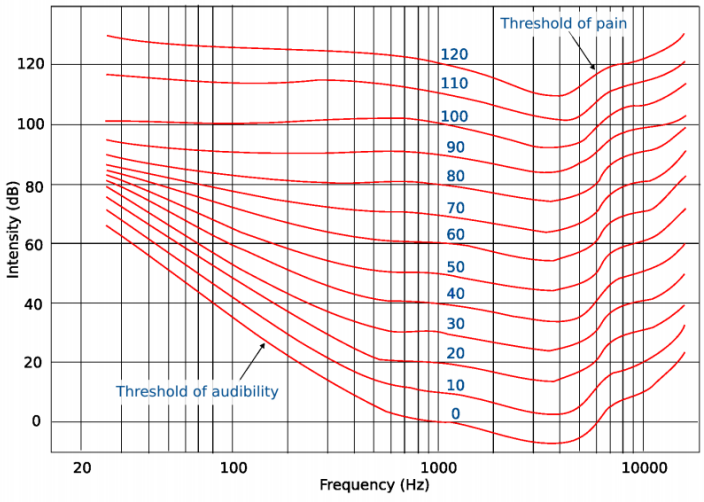


Figure 1.1: Fletcher-Munson Equal Loudness Contours, showing the perceived loudness as a function of the frequency and intensity of the stimulus. After [Fletcher and Munson, 1933, Ropshkow, 2005]

In music theory, loudness is defined by a piece's **dynamics**. Dynamics are indicators of a part's loudness *relative* to other parts and/or instruments. Dynamics markings are expressed with the italian keywords *forte* ***f*** (loud) and *piano* ***p*** (soft). Subtle degrees of loudness can be expressed by the prefixes *mezzo-* or *più*, for example ***mp*** stands for *mezzo-piano* (moderately soft) and *più* ***p*** (softer), or by consecutive letters such as *fortissimo* ***ff*** (very loud) or more letters if needed.

Music dynamics also allow expressing gradual changes in loudness, indicated as symbols or italian keywords (*crescendo* and *diminuendo*).

1.4 Audio signal processing

1.4.1 Discrete-time signals

The domain of audio signal processing deals with recorded digital/analog signals, which are discrete-time signals. The **Nyquist-Shannon sampling theorem** is the fundamental bridge between continuous-time and discrete-time signals. It establishes a sufficient condition for a sample rate that permits a discrete sequence of samples to capture all the information from a continuous-time signal. (Wikipedia 2020)

The sample rate f_s of a discrete-time signal is defined as the number of samples per second, its inverse is the time step between samples T_s.

We denote, conformely to litterature a discrete signal time frame as $x[n] = x(t_n)$ where $t_n = n \cdot T_s = \frac{n}{f_s}$.

1.4.2 Discrete Fourier Transform (DFT)

The discrete Fourier transform of N samples, with a sample rate of f_s can be obtained from its continuous definition.

$$X(f) = \int_0^{t_N} x(t) \cdot e^{-2\pi j f t} \mathrm{d}t \tag{1}$$

$$= \lim_{f_s \to \infty} \sum_{n=0}^{N-1} x(t_n) \cdot e^{-2\pi j f t_n} \tag{2}$$

$$= \lim_{f_s \to \infty} \underbrace{\sum_{n=0}^{N-1} x[n] \cdot e^{-2\pi j f \frac{n}{f_s}}}_{X[f]} \tag{3}$$

$$= \lim_{f_s \to \infty} X[f] \tag{4}$$

The DFT of $x[n]$ is given for all frequency bins $k = 0, \ldots, K$

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-2\pi j k \frac{n}{f_s}}$$

2 Pitch analysis

2.1 Introduction

Pitch analysis is the task of estimating the fundamental frequency of a periodic signal that is the inverse of the period which is defined as "the smallest positive member of the infinite set of time shifts leaving the signal invariant" (Cheveigné and Kawahara 2002). As music signal frequencies vary through time, the pitch analysis is usually performed on a short time frame (window) allowing to express the obtained pitch as a function of time, we will consider henceforth the analysis on a single frame.

Furthermore, the physical model we have considered for the signal formula is based on physical hypotheses. In fact, we considered a signal formed by a perfectly harmonic instrument travelling in a perfectly undisturbed homogenuous medium with no other iterfering waves. Since such conditions are almost never met, we base our analysis on *imperfect conditions*. Indeed, the recorded signal represents the pressure function at the receptors position. Consequently, the recorder captures the pressure at its position from *all* surrounding stimuli, recording surrounding noise, resonance effects, and the reflected wave with a certain lag. As a result, we express the observed signal as the sum of the harmonic signal $\tilde{x}$ and the residual z. (Yeh 2008)

$$x(t) = \tilde{x}(t) + z(t)$$

Before we move on, let's consider the *harmonicity* of a sound. In the case of perfectly harmonic instrument the frequency of harmonic partials is expressed as a proper multiple of the fundamental frequency $f_h = hf_0$. However, most musical instruments are not perfectly harmonic, for example the h^{th} harmonic frequency of a vibrating string is given as

$$f_h = hf_0\sqrt{1+Bh^2} \quad \text{where} \quad B = \frac{\pi^3 E d^4}{64 l^2 T}$$

where B is the inharmonicity factor of the string, E is Young's modulus, d is the diameter of the string, l is its length and T is its tension. We refer to such signals as **quasi-periodic**. Pitch analysis therefore has to take into account the inharmonicity of a signal in the process of estimating its fundamental frequencies in order to prevent cases of false negatives (missed pitches). [source needed]

Pitch analysis deals with both monophonic and polyphonic signals, a monophonic signal is a signal produced by a single harmonic source whereas polyphonic signals have multiple sources, in the case of the latter the task is significantly harder. Nevertheless, pitch estimation methods for both single and multiple sourced harmonics can be classified into two categories: methods that estimate the *period* in the signal time domain and methods that estimate the f_0 from the harmonic patterns in the signal spectrum.

2.2 Single pitch

Single pitch estimation is based on finding the fundamental frequency of a monophonic sound. The quasi-periodic monophonic signal $\tilde{x}$ is expressed as

$$\tilde{x}(t) = \sum_{h=1}^{\infty} A_h \cos(2\pi f_0 t + \varphi_h)$$

For practical reasons, a finite number of harmonic partials H is used to approximate the signal.

$$\tilde{x}(t) \approx \sum_{h=1}^{H} A_h \cos(2\pi f_0 t + \varphi_h)$$

The estimation of f_0 can be approached in two different ways: by analysing the time function $x(t)$ or by analysing the signal spectrum $X(f)$.

2.2.1 Time domain

Time domain methods analyse the repetitiveness of the wave by comparing the signal with a delayed version of itself. This comparison is achieved using special functions that represent the pattern similarity or dissimilarity as a function of the **time lag** τ.

We will study and compare the functions that appear the most in litterature.

Autocorrelation function The autocorrelation function (ACF) comes immediately to mind. By definition, autocorrelation is the similarity function between observations. Given a discrete signal of N samples, the autocorrelation function is defined as

$$r[\tau] = \sum_{t=1}^{N-\tau} x[t]x[t+\tau]$$

The value is of the ACF is at a local maximum when the lag is equal to the signal's period or its multiples. Autocorrelation is sensitive to structures in signals, making it useful to applications of speech detection. However, in the case of music signals, resonance structures appear hence the need for a better adapted function.

Difference function The Average Magnitude Difference Function (AMDF) (Ross et al. 1974) is the average unsigned difference between $x(t)$ and $x(t+\tau)$.

$$d_{\mathrm{AM}}[\tau] = \frac{1}{N} \sum_{t=1}^{N-\tau} |x[t] - x[t+\tau]|$$

The difference function is at its local minima for lags equal to proper multiples of the signals period. AMDF is more adapted than autocorrelation for applications in music processing.

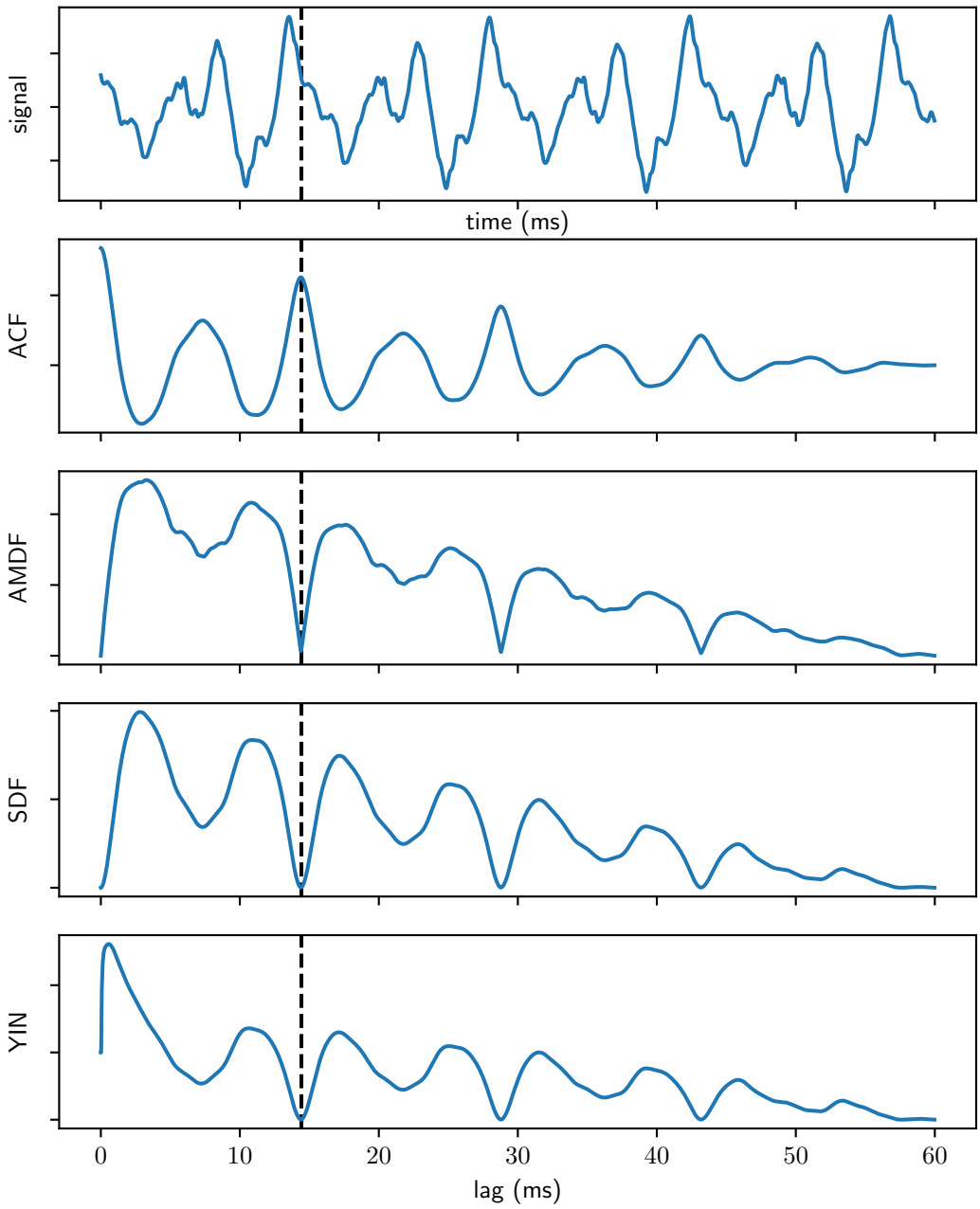
Squared difference function The Squared Difference Function (SDF) is very similar to AMDF, it accentuates however the dips at the signals period therefore indicate local extrema more clearly.

$$d[\tau] = \sum_{t=1}^{N-\tau} (x[t] - x[t+\tau])^2$$

YIN algorithm (Cheveigné and Kawahara 2002) employs the SDF as an auxiliary function for calculating the **cumulative mean normalized difference function** that divides SDF by its average over shorter lags and starts at 1 rather than 0 (in the case of SDF and AMDF); it tends to stay large at short lags and drops when SQD falls under its average.

$$d_{\mathrm{YIN}}[\tau] = \begin{cases} 1 & \text{if } \tau = 0 \\ d[\tau] / \frac{1}{\tau} \sum_{t=0}^{\tau} d[t] & \text{otherwise} \end{cases}$$

```
from muallef.io import AudioLoader
from muallef.plot import diff_functions as df

cello = AudioLoader('samples/instrument_single/cello_csharp2.wav')
cello.cut(start=2, stop=2.06)
df.time_domain_plots(cello.signal, cello.sampleRate, pitch=69.3)
```



2.2.2 Spectral domain

Fourier transform is the most adapted mathematical tool for analysing periodicity in functions. The transform produced a complex function of frequency, where the magnitude of the transform attains its local maxima at the signal's frequency and its *harmonics*.

Spectral domain methods analyse the magnitude and/or the phase of the fourier transform of the signal, which generally gives better results. Nevertheless, similar comparison functions are employed in order to get the fundamental frequency.

Spectral autocorrelation Autocorrelation measures repititive patterns, since harmonics appear at almost fixed frequency intervals, ACF allows to identify harmonic partials. (Lahat, Niederjohn, and Krubsack 1987) The autocorrelation is applied to the spectrum of the signal, that is the magnitude of the fourier transform. The function attains its local maxima at frequency shifts that are multiples of f_0, otherwise the function is attenuated since the partial peaks are not well aligned.

For a spectrum $S[f] = |X[f]|$ with K spectral bins

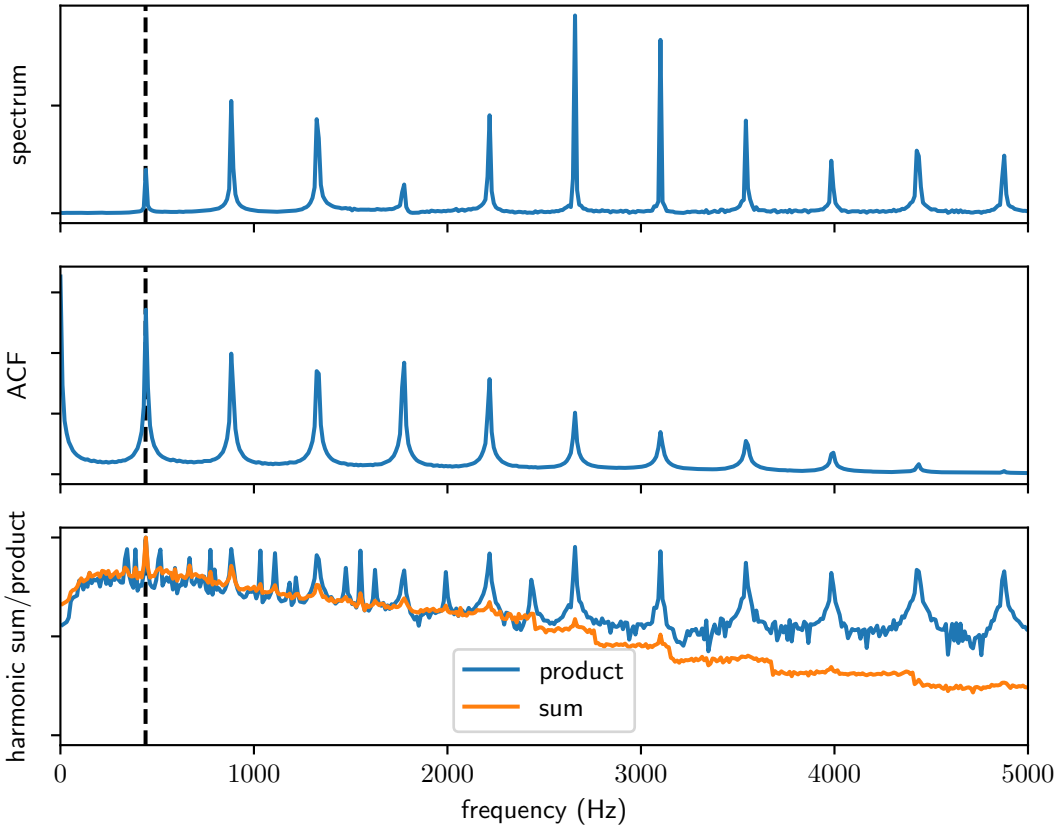
$$R[f] = \sum_{k=1}^{K-f} S[k]S[k+f]$$

Harmonic sum A *frequency histogram* represents the number of occurrences of each frequency, it does not however reflect the *amplitudes* of the harmonics of frequencies. Schroeder proposes to *weight* the contribution of each harmonic to the histogram with a monotonically increasing function of its amplitude, this is done using *log compression* where spectral harmonic bins are compressed with a logarithm. Finally, Schroeder proposed two functions of frequency that sum the compressed weighted histogram. (Schroeder 1968)

- **Harmonic sum:**
$$\Sigma(f) = \sum_{m=1}^{M} 20 \log_{10} S(nf)$$
- **Harmonic product:**
$$\Sigma'(f) = 20 \log_{10} \sum_{m=1}^{M} S(nf)$$

The sum inside the logarithm in the harmonic product can be viewed as a product because of the properties of the logarithm function.

```
oboe = AudioLoader('samples/instrument_single/oboe_a4.wav')
oboe.cut(start=0.5)
df.spectral_plots(oboe.signal[:4096], oboe.sampleRate, pitch=440)
```



Spectral YIN The spectral YIN method (Brossier 2006) is an optimized version of YIN's algorithm computed in the frequency domain. The square difference function is defined over spectral magnitudes

$$\hat{d}(\tau) = \frac{2}{N} \sum_{k=0}^{\frac{N}{2}+1} \left| \left(-e^{2\pi j k \tau / N} \right) X[k] \right|^2$$

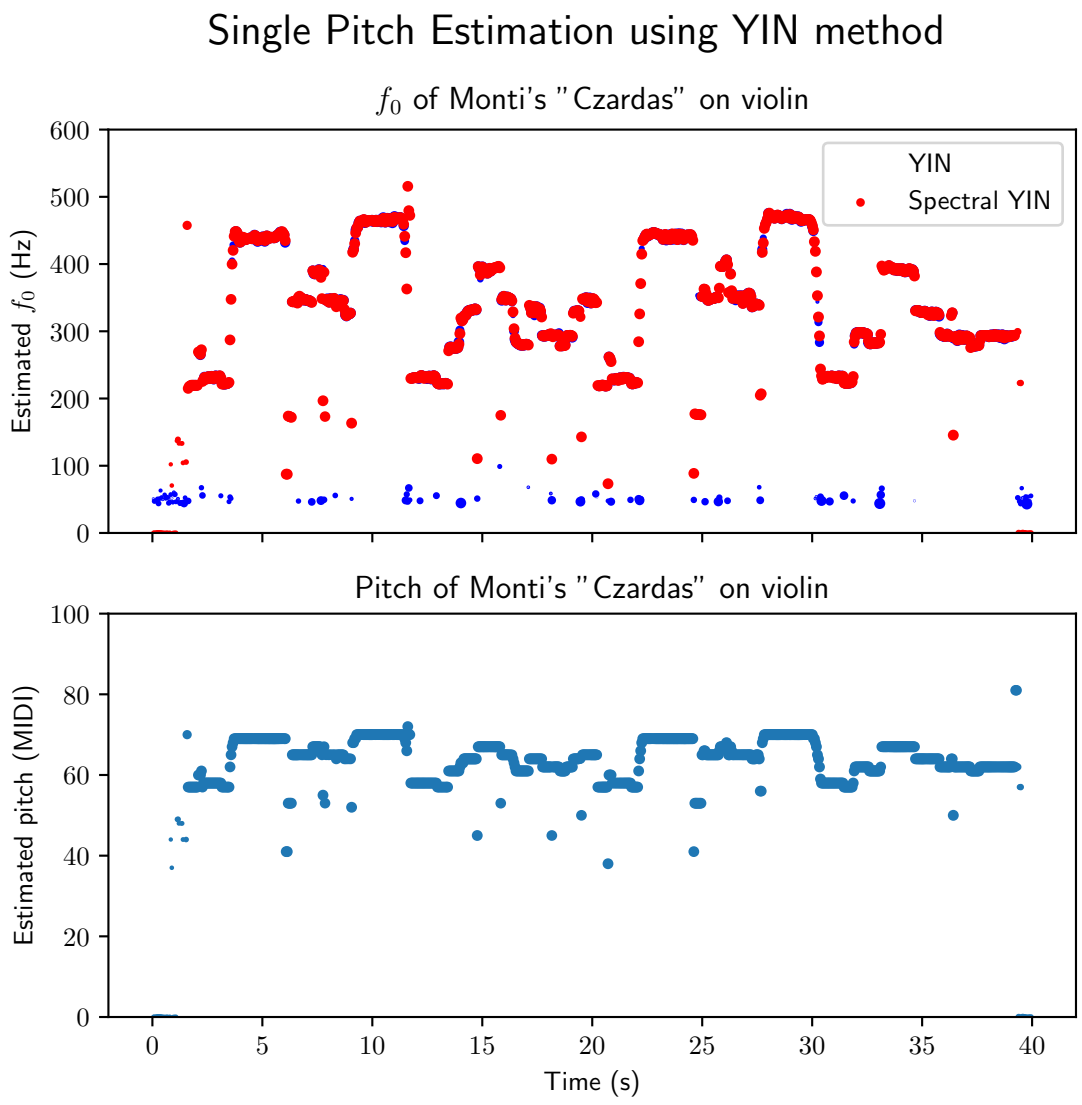
2.2.3 Application Example

I have recorded myself playing Vittorio Monti's violin piece "Czardas" which is relatively complex musically since it features tonal *glissando* (continuous slides) and is grace notes (short time notes).

We test pitch estimation using the YIN method in the time domain as well as the spectral domain.

```
from muallef.pitch import MonoPitch
from muallef.util.units import Hz_to_MIDI

czardas = AudioLoader('samples/monophonic/czardas_cut.wav')

yin = MonoPitch(czardas.signal, czardas.sampleRate, method='yin')
yin_f0 = yin()
yin_conf = yin.get_confidence(normalize=True)
yinfft = MonoPitch(czardas.signal, czardas.sampleRate, method='yinfft')
yinfft_f0 = yinfft()
yinfft_conf = yinfft.get_confidence(normalize=True)
time = czardas.time(len(yinfft_f0))

fig, ax = plt.subplots(2, 1, sharex=True)
fig.set_figheight(6)
```

```
_ = fig.suptitle("Single Pitch Estimation using YIN method", fontsize=16)
_ = ax[0].set_title("$f_0$ of Monti's \"Czardas\" on violin")
_ = ax[0].scatter(time, yin_f0, c='blue', s=10*yin_conf, label='YIN')
_ = ax[0].scatter(time, yinfft_f0, c='red', s=10*yinfft_conf, label='Spectral YIN')
_ = ax[0].set_ylim(0, 600)
_ = ax[0].set_ylabel('Estimated $f_0$ (Hz)')
_ = ax[0].legend()
_ = ax[1].set_title("Pitch of Monti's \"Czardas\" on violin")
pitch = np.round(Hz_to_MIDI(yinfft_f0))
_ = ax[1].scatter(time, pitch, s=10*yinfft_conf)
_ = ax[1].set_ylim(0, 100)
_ = ax[1].set_ylabel('Estimated pitch (MIDI)')
_ = ax[1].set_xlabel('Time (s)')
plt.show()
```



As expected, f_0 values were successfuly detected including fuzzy glissando pitches and grace notes.

2.3 Multiple pitch

In polyphonic music analysis, we are interested in detecting the fundamental frequences for concurrent signals, the signals can be produced by several instruments simultanuously.

There are generally two approaches to this problem: iterative estimation and joint estimation. In iterative estimation, the most prominent f_0 is extracted at each iteration until no additional f_0 can be estimated. Generally, iterative estimation models tend to accumulate errors at each iteration step, they are however computationally cheap. Whereas joint estimation methods evaluation f_0 combinations which leads to more accurate estimates, however the computational cost is significantly increased.(Benetos et al. 2013)

We establish the formalism of the task analogously to a single pitch harmonic signal. A multi-pitch harmonic signal $\tilde{x}(t)$ can be expressed as the sum of M harmonic signals.

$$\tilde{x}(t) = \sum_{m=1}^{M} \tilde{x}_m(t)$$

where $\tilde{x}_m(t)$ is a harmonic monophonic signal similar to signals we've seen so far. It follows that

$$x(t) \approx \sum_{m=1}^{M} \sum_{h=1}^{H_m} A_{m,h} \cos(2\pi h f_{0,m} t + \varphi_{m,h}) + z(t)$$

2.3.1 Harmonic Amplitudes Sum

A. Klapuri (2006) proposes a robust pipeline for estimating fundamental frequencies in polyphonic music signals. The method looks for f_0 that maximizes a frequency strength over candidate frequencies in a whitened spectrum.

1. **Spectral whitening:** different sources can have different timbral information in the signal spectrum. In order to detect analyse the frequencies of the different sources, Klapuri proposes suppressing the timbral information prior to detecting dominant frequencies in the spectrum. This process is done by a sequence of transformations:
   - Apply *bandpass filter* to the spectrum $X(f)$ to obtain center frequencies c_b where b is the subband index of the filtered spectrum. Each subband has a triangular power response $H_b(f)$ such that $\text{supp}\left(H_b(k)\right) = [c_{b-1}, c_{b+1}]$.
   - Calculate standard deviations σ_b within subbands

     $$\sigma_b = \left(\frac{1}{K} \sum_f H_b(f) \left| X(f) \right|^2 \right)^{1/2}$$

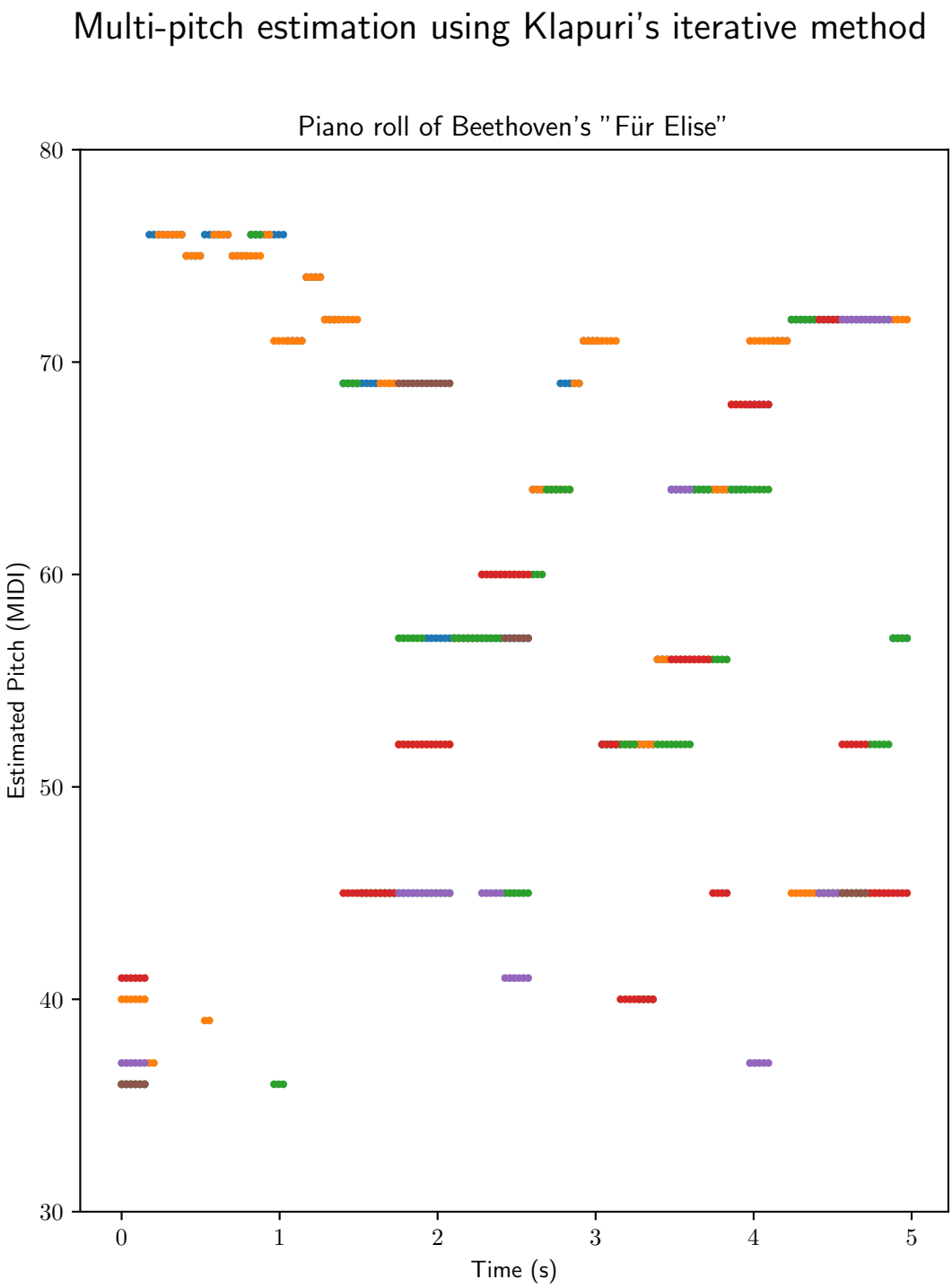
     where K is the number of frequency bins of the Fourier transform.
   - Calculate compression coefficients $\gamma_b = \sigma_b^{\nu-1}$ where ν is the whitening parameter, the proposed value is $\nu = 0.33$.
   - Interpolate $\gamma(f)$ for all frequency bins f from γ_b.
   - Finally calculate the whitened spectrum $Y(f)$ by weighting the input spectrum by the obtained compression coefficients $Y(f) = \gamma(f)X(f)$.
2. **Salience function:** strength of $f0$ candidates is evaluated using a salience function s that calculates the weighted sum of harmonic partials' amplitudes, similarly to Schroeder's function (1968).

   $$s(\tau) = \sum_{h=1}^{H} g(\tau, h) \left| Y(h f_\tau) \right|$$

   where $f_\tau = f_s/\tau$ is the f_0 candidate corresponding to the period τ and $g(\tau, h)$ is the weight of the h partial of period τ.
3. Finally the frequencies are estimated iteratively or jointly by determining $f_0 = \text{argmax}_f\, s(f)$. In iterative evaluation, the found f_0 is removed from the residual spectrum and the process is repeated until the spectrum is flat.

Application Example We test Klapuri's pipeline on Beethoven's infamous piano piece "Für Elise".

```
from muallef.pitch import MultiPitch
from muallef.util.units import Hz_to_MIDI

fur_elise = AudioLoader('samples/polyphonic/furElise.wav')
fur_elise.cut(stop=5)

klapuri = MultiPitch(fur_elise.signal, fur_elise.sampleRate, method='klapuri')
pitch = Hz_to_MIDI(klapuri())
time = fur_elise.time(pitch.shape[1])

fig, ax = plt.subplots()
fig.set_figheight(8)
_ = fig.suptitle("Multi-pitch estimation using Klapuri's iterative method", fontsize=16)
_ = ax.set_title("Piano roll of Beethoven's \"Für Elise\"")
for m in range(pitch.shape[0]):
    _ = ax.scatter(time, np.round(pitch[m]), s=5)
_ = ax.set_xlabel('Time (s)')
_ = ax.set_ylabel('Estimated Pitch (MIDI)')
_ = ax.set_ylim(30, 80)
plt.show()
```



The resulting piano-roll shows a generally decent representation of the piece as the percentage of false positives remains relatively low.

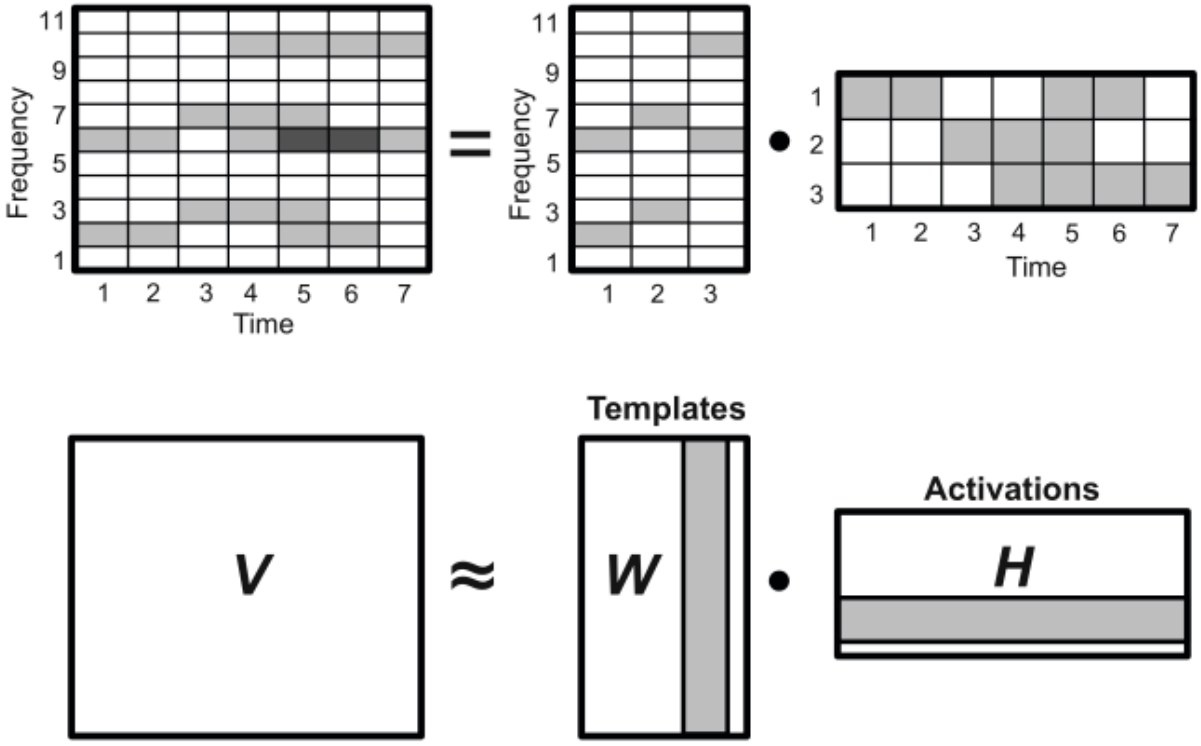
2.3.2 Spectral factorisation

Non-negative Matrix Factorisation **(NMF)** is a well-established technique applied to several problems, in (Smaragdis and Brown 2003) a method is proposed for applying NMF to the signal spectrum.

$$\boldsymbol{V} \approx \boldsymbol{W}\boldsymbol{H}$$

The method consists of factorising a non-negative matrix $\boldsymbol{V} \in \mathbb{R}_+^{K \times N}$ into the product of two non-negative matrices $\boldsymbol{W} \in \mathbb{R}_+^{K \times R}$ and $\boldsymbol{H} \in \mathbb{R}_+^{R \times N}$ where R is the factorisation rank with $R << K$, given N time frames and K spectral bins. The matrix $\boldsymbol{W}$ is the **template matrix** that extracts the features of $\boldsymbol{X}$ into R classes referred to as *templates*. The matrix $\boldsymbol{H}$ is the **activation matrix** which represents the *activation time* of each template.

In the application of over music signal spectrums, $\boldsymbol{V} = \boldsymbol{X}^\top$ where $\boldsymbol{X} \in \mathbb{R}_+^{N \times K}$ is the spectrogram of the signal which is the magnitude of the STFT of the signal. The factorisation templates correspond to *pitch classes*, where in the case of most instruments or music ensembles is less than $R = 100$. The template matrix $\boldsymbol{W}$ corresponds to spectral bases for each pitch component and the activation matrix $\boldsymbol{H}$ represents pitch activity across time.



The problem is formulated as a non-convex optimisation problem

$$(\boldsymbol{W}, \boldsymbol{H}) = \underset{\boldsymbol{W},\boldsymbol{H}>0}{\operatorname{argmin}} \|\boldsymbol{V} - \boldsymbol{W}\boldsymbol{H}\|$$

The implemented cost function $C = \|\boldsymbol{V} - \boldsymbol{W}\boldsymbol{H}\|$ is the euclidean norm L_2. The matrices $\boldsymbol{V}$ and $\boldsymbol{H}$ are decomposed into N column vectors, $\boldsymbol{V} = (v_1, \ldots, v_N)$ and $\boldsymbol{H} = (h_1, \ldots, h_N)$, which implies $\forall i \in \{1, \ldots, N\}, v_i = \boldsymbol{W}h_i$. By imposing the orthogonality constraint $\boldsymbol{H}\boldsymbol{H}^\top = I$, we obtain the **K-means** clustering property. The values of $\boldsymbol{W}$ and $\boldsymbol{H}$ can be initialized randomly and are therefore *learned* iteratively.

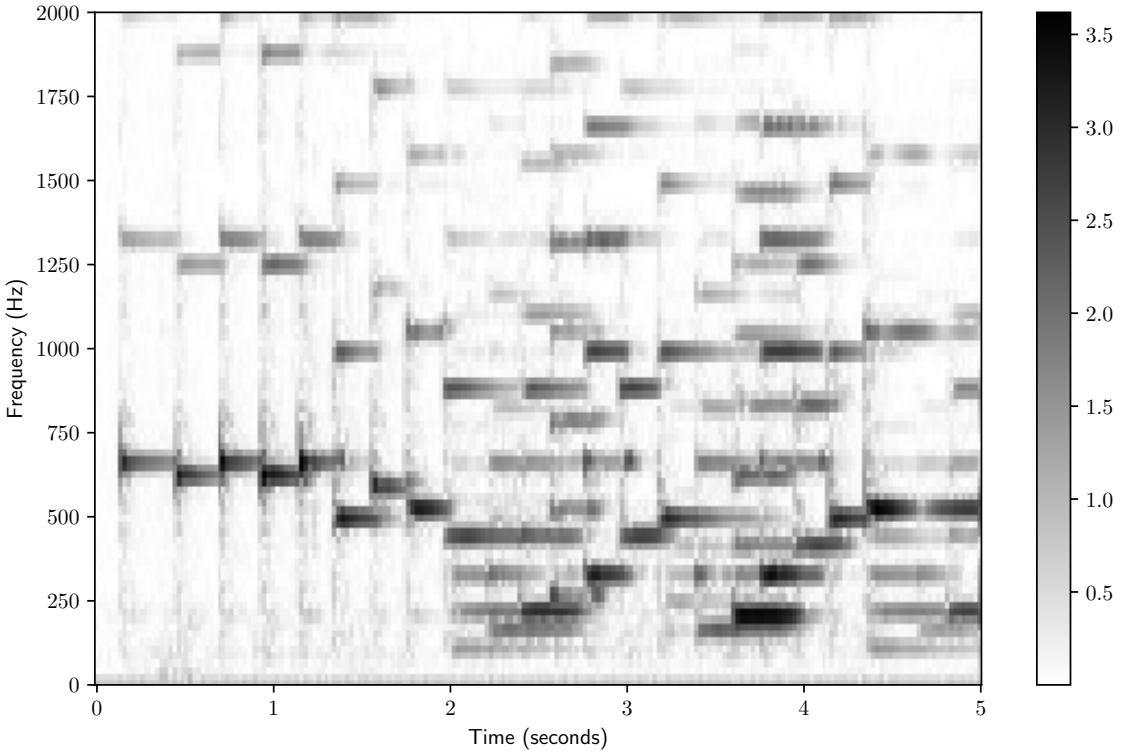
Reinforcing a sparsity constraint was proposed in (Cont 2006) for spectral factorisation since pitch templates correspond to *discrete* frequency values. Moreover, a subset of pitch templates are activated simultanuously in a musical piece, especially in the case of a piano piece.

Finally, single pitch estimation is performed on rows of $\boldsymbol{H}$.

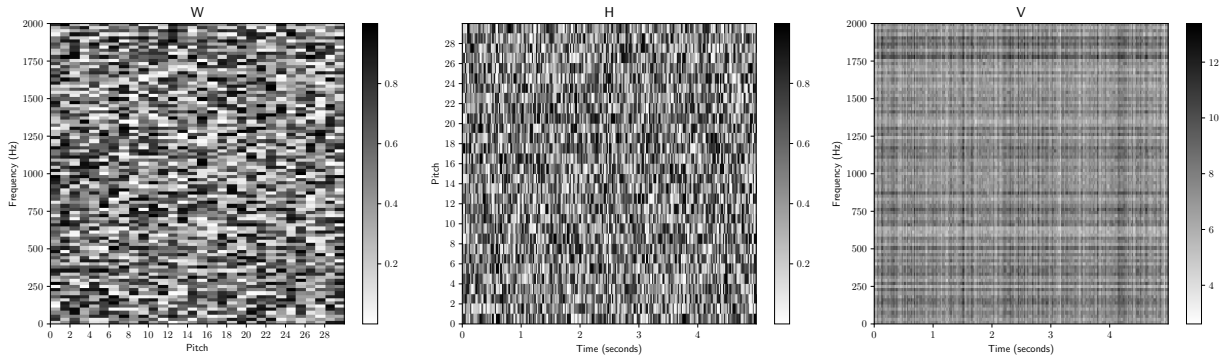
Unfortunally, our implementation of this algorithm did not give successful results, we use the API provided in for testing (Müller 2015).

```
from muallef.plot.nmf import plot_NMF_factors

import pandas as pd
import librosa
from LibFMP.B import plot_matrix
from LibFMP.C8 import NMF
```

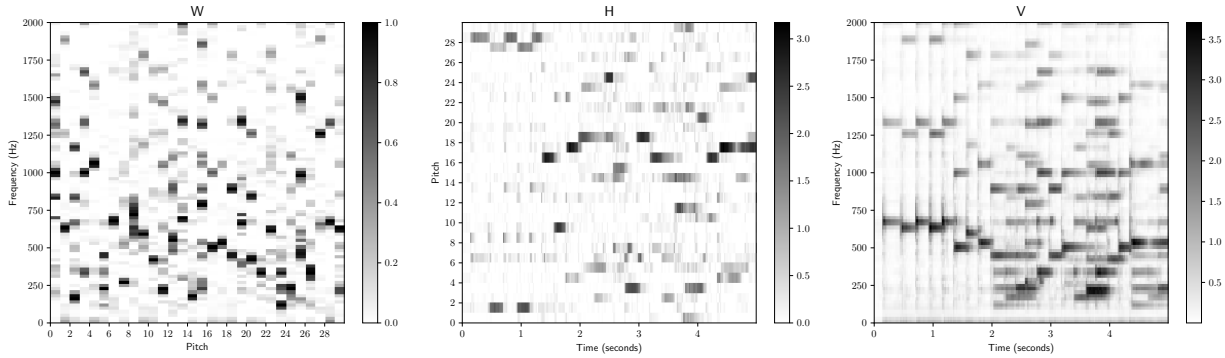
```
fs = fur_elise.sampleRate
x = fur_elise.signal
N_fft = 2048
H_fft = 1024

X = librosa.stft(x, n_fft=N_fft, hop_length=H_fft)
V = np.log(1 + np.abs(X))
freq_max = 2000

# plot input spectrogram
_ = plot_matrix(V, Fs=fs/H_fft, Fs_F=N_fft/fs, figsize=(8, 5))
_ = plt.ylim([0, freq_max])
plt.show()
```




```
K = V.shape[0]
N = V.shape[1]
R = 30

# Initialize and plot random matrices W, H
W_init = np.random.rand(K,R)
H_init = np.random.rand(R,N)
plot_NMF_factors(W_init, H_init, W_init.dot(H_init), fs, N_fft, H_fft, freq_max)

# Calculate and plot NMF decomposition
```




```
W, H, V_approx, V_approx_err, H_W_error = NMF(V, R, W=W_init, H=H_init, L=200, norm=True)
plot_NMF_factors(W, H, W.dot(H), fs, N_fft, H_fft, freq_max)
```




```
print(f"V error approximation = {V_approx_err}")
```

```
V error approximation = 3.21356984068787
```

The obtained $\boldsymbol{V}$ matrix is close to the input spectrogram, the matrices are all sparse as expected. Nevertheless, the matrices are fuzzy therefore pitch templates and their activations are not clear. The NMF factorisation as is, might not render better results than Klapuri's. In fact most of pitch templates correspond to multiple note mixtures, the results can be enhanced by initializing *pitch-informed constraints* where $\boldsymbol{W}$ is initialized to MIDI pitch classes. It can however be very useful method for separating different sound sources.

3 Temporal segmentation

3.1 Introduction

Temporal segmentation is the task of finding time boundaries of audio objects, in the case of music signals the audio objects in question are the musical *notes*.

A musical note's **onset** is defined as the time it starts, and its ending time is the **offset**.

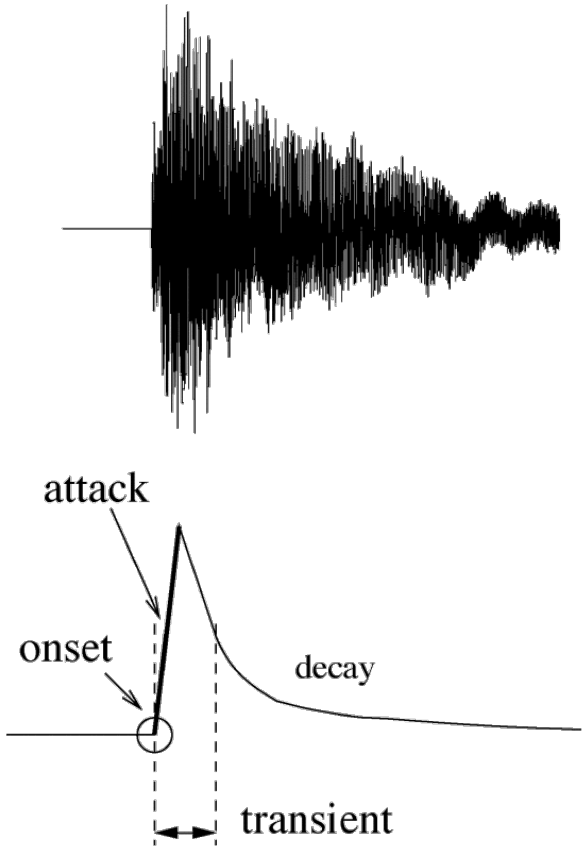


Figure 1: IEEE transactions on speech and audio processing, vol. 13, no. 5, september 2005

The signal form varies according to instruments. The onset profile in the image above corresponds to an instrument producing sudden energy bursts such as a pinched-chord instrument (piano, guitar, etc), or a percussive instrument, unlike bowed-chord instruments and wind instruments that do no exhibit such energy bursts. In both cases, the spectral flux, changes in energy, and/or harmonic distribution are analysed in order to estimate onset times.

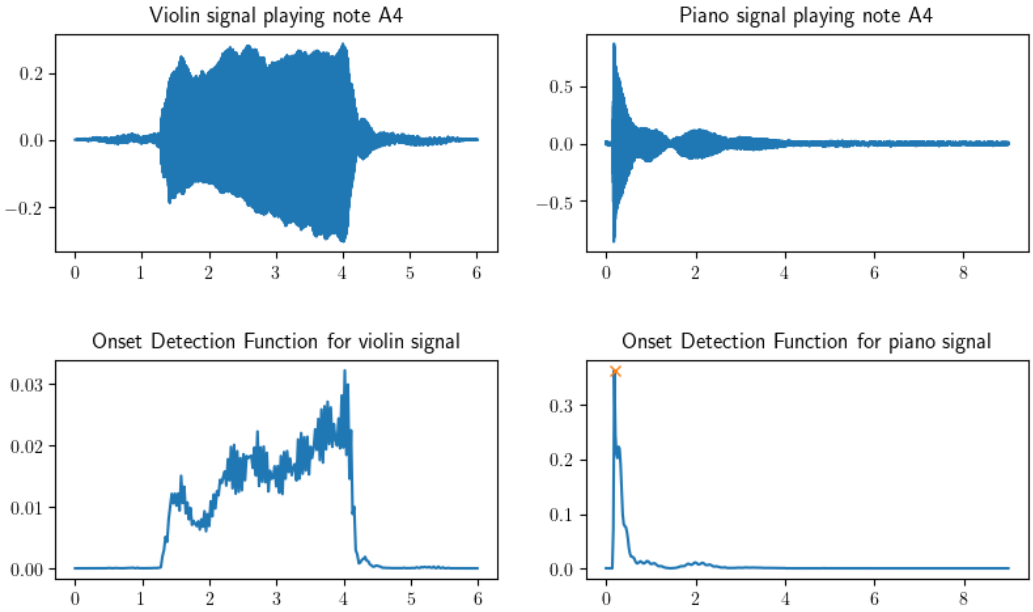


Figure 2: (left) a violin onset profile (right) a piano onset profile

The general onset estimation model is a three-step pipeline (Brossier 2006)

1. Computing an **Onset Detection Function (ODF)** that characterizes change in energy and/or harmonic content in a music signal. The change is measured in the time domain, frequency domain, phase domain, or complex domain for analysing onsets of sounds of different natures.
2. Calculate a smooth **threshold function** as ODFs tend to be sensitive to the slightest changes, therefore providing a threshold for viable onset candidates.
3. **Peak-picking** local maxima of the ODF that are greater than the calculated threshold.

3.2 Onset Detection Function (ODF)

We present a few functions that analyse different features of a musical sound that correspond to subsets of musical sources.

3.2.1 High Frequency Content (HFC)

The proposed function (Masri and Bateman 1996) favours wide-band energy bursts over changes in amplitude modulation, and accords a stronger weight to high frequency spectral bins.

$$D_{\mathrm{HFC}}[n] = \sum_{k=1}^{N} k \cdot \|X[n,k]\|^2$$

The function emphasises high frequency energy bursts, which makes it more adapted to percussive onsets than than bowed-strings or wind instruments. (Brossier 2006)

3.2.2 Phase Deviation

A different approach proposed by (Bello and Sandler 2003), where the function evaluates phase difference that can help identify *tonal* onsets as well as percussive onsets.

$$D_{\Phi}[n] = \sum_{k=0}^{N} |\hat{\varphi}[n,k]|$$

where

- $\text{princarg}(\theta) = \pi + ((\theta + \pi) mod(-2\pi))$
- $\varphi(t, f) = \arg(X(t, f))$
- $\hat{\varphi}(t, f) = \text{princarg}\left(\frac{\partial^2 \varphi}{\partial t^2}(t, f)\right)$

The phase deviation function can result in false positives as phase changes occur in noisy signals.

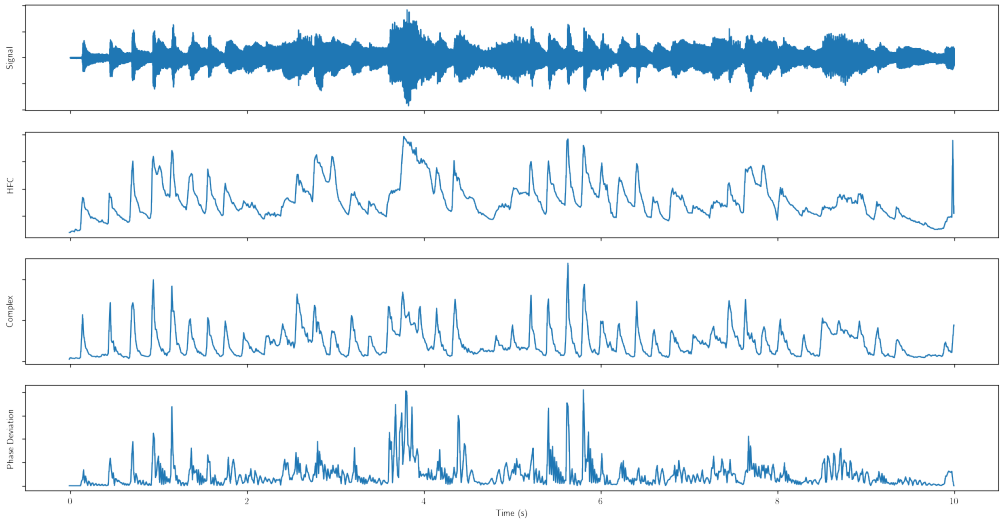
3.2.3 Complex Distance

Another ODF is presented in (Duxbury et al. 2003) that qualifies changes in both magnitude and phase in order to detect percussing *and* tonal onsets.

$$D_{\mathbb{C}}[n] = \sum_{k=0}^{N} \left\| \hat{X}[n,k] - X[n,k] \right\|^2$$

where $\hat{X}[n,k] = |X[n,k]| \cdot e^{j\hat{\varphi}[n,k]}$

The presented function combines spectral difference and phase-based approaches, by borrowing the phase deviation function from (Bello and Sandler 2003)



3.3 Thresholding & Peak-picking

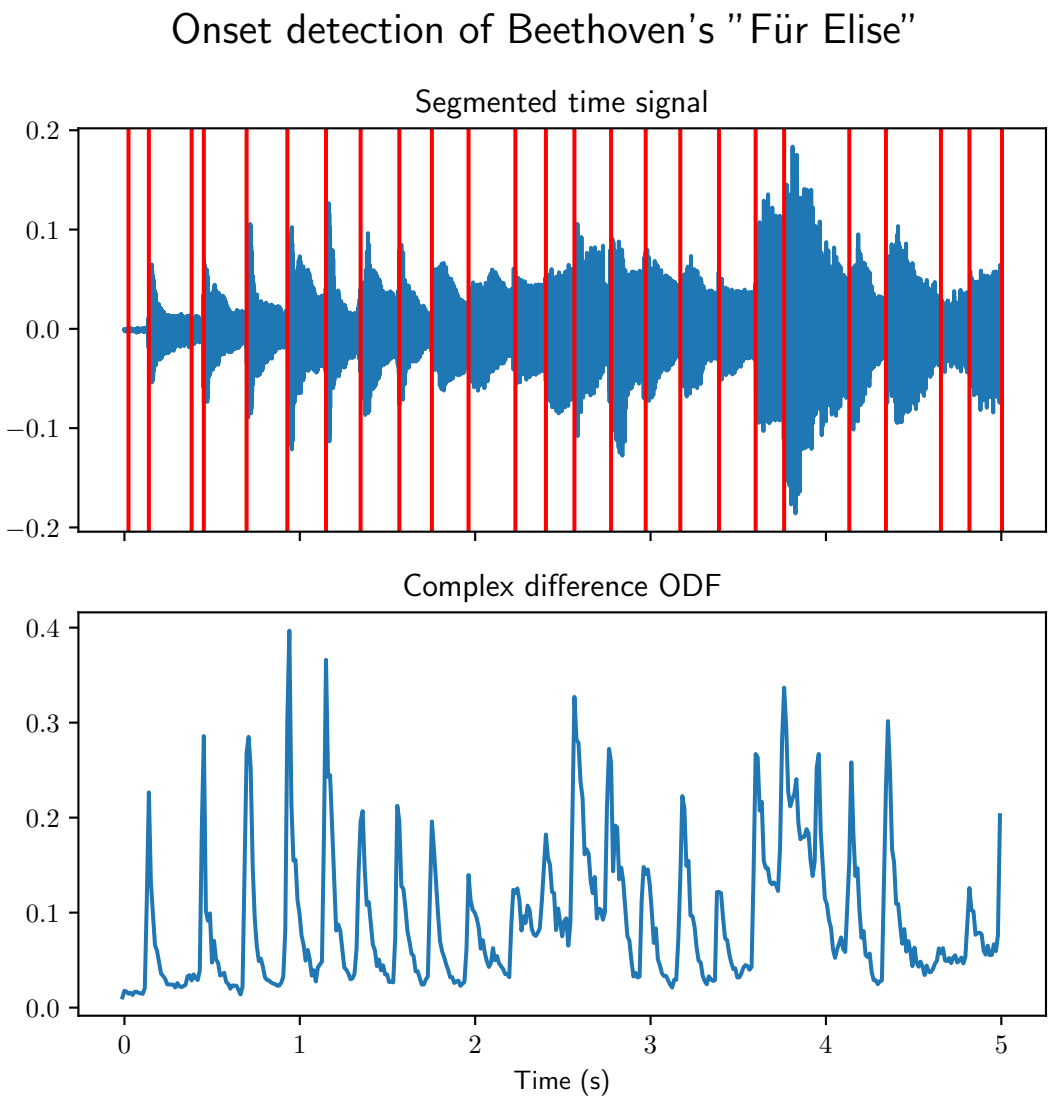
Since Onset Detection Functions are usually sensitive to the slightest perturbations, false positives belong to a subset of the local maxima of the ODF. In order to filter such values, a smoothed version of the ODF can serve as threshold for eliminating insignificant peaks.

A windowed moving average is a good threshold function, it is defined as the convolution product of the ODF with the window function. In our implementation we have used a Hann window as it limits the aliasing phenomenon in spectras.

3.4 Results

We apply our onset detection pipeline to the same audio sample of Beethoven's "Für Elise".

```
from muallef.onset import Onset

fur_elise.cut(stop=5)
fs = fur_elise.sampleRate
x = fur_elise.signal
t = fur_elise.time()

onset = Onset(x, fs, method='complex')
onsets = onset()

fig, ax = plt.subplots(2, 1, sharex=True)
fig.set_figheight(6)
_ = fig.suptitle("Onset detection of Beethoven's \"Für Elise\"", fontsize=16)
_ = ax[0].set_title("Segmented time signal")
_ = ax[0].plot(t, x)
for on in onsets:
    _ = ax[0].axvline(x=on, color='red')
_ = ax[1].set_title("Complex difference ODF")
_ = ax[1].plot(onset.onsetTime, onset.onsetFunction)
_ = ax[1].set_xlabel('Time (s)')
plt.show()
```



4 Conclusion

Great strides have been made in the field of Music Information Retrieval pertaining to Automatic Music Transcription, resulting in satisfactory results for certain underlying tasks, namely onset detection and single pitch estimation. Nevertheless, most AMT problems remain open as researchers worldwide study and apply new concepts everyday.

In the scope of this project, we have explored well-established concepts of onset detection and single pitch estimation, and succeeded in obtaining satisfactory results. We have as well explored two different approaches of multi-pitch estimation and obtained relatively coherent results with Klapuri's method, but unfortunally failed in applying Non-Negative Factorisation as we hoped.

As this is our second attempt in approaching AMT, we have been able to study closely core concepts of AMT, and deeply explore the core difficulty of AMT systems that is *Multi-pitch Estimation*. As research has lead us to studying several methods and approaches to this problem, we had to restrict the study to two algorithms that are robust, mathematically sound and appreciated by the MIR community.

I have held interest for this subject for quite some time, partly because I am a violinist myself but also because of my fondness of the employed mathematical principles. Most importantly, this project requires application of various mathematical notions as well as computer science skills hence serving as a demonstration of acquired knowledge throughout the Masters program. This open problem is more suited to a PhD thesis subject or as a full-time focus research, we have attempted to do as much as we could to accomplish with very little time.

References

Bello, J. P., and M. Sandler. 2003. "Phase-Based Note Onset Detection for Music Signals." In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, 5:V–441. https://doi.org/10.1109/ICASSP.2003.1200001.

Benetos, Emmanouil, Simon Dixon, Dimitrios Giannoulis, Holger Kirchhoff, and Anssi Klapuri. 2013. "Automatic Music Transcription: Challenges and Future Directions." *Journal of Intelligent Information Systems* 41 (December). https://doi.org/10.1007/s10844-013-0258-3.

Brossier, Paul M. 2006. *Automatic Annotation of Musical Audio for Interactive Applications.*

Cheveigné, Alain de, and Hideki Kawahara. 2002. "YIN, a Fundamental Frequency Estimator for Speech and Music." *The Journal of the Acoustical Society of America* 111 (4): 1917–30. https://doi.org/10.1121/1.1458024.

Cont, Arshia. 2006. "Realtime Multiple Pitch Observation Using Sparse Non-Negative Constraints," 6.

Duxbury, Chris, Juan Pablo Bello, Mike Davies, and Mark Sandler. 2003. "COMPLEX DOMAIN ONSET DETECTION FOR MUSICAL SIGNALS," 4.

Feynman, Richard. 1965. "The Feynman Lectures on Physics Vol. I Ch. 47: Sound. The Wave Equation." https://www.feynmanlectures.caltech.edu/I_47.html.

Klapuri, Anssi. 2006. "Multiple Fundamental Frequency Estimation by Summing Harmonic Amplitudes," 6.

Lahat, M., Russell J. Niederjohn, and David A. Krubsack. 1987. "A Spectral Autocorrelation Method for Measurement of the Fundamental Frequency of Noise-Corrupted Speech." *IEEE Trans. Acoustics, Speech, and Signal Processing.* https://doi.org/10.1109/TASSP.1987.1165224.

Masri, Paul, and Andrew Bateman. 1996. "Improved Modelling of Attack Transients in Music Analysis-Resynthesis," 4.

Müller, Meinard. 2015. *Fundamentals of Music Processing - Audio, Analysis, Algorithms, Applications.* Springer. https://www.audiolabs-erlangen.de/fau/professor/mueller/bookFMP.

Ross, M., H. Shaffer, A. Cohen, R. Freudberg, and H. Manley. 1974. "Average Magnitude Difference Function Pitch Extractor." *IEEE Transactions on Acoustics, Speech, and Signal Processing* 22 (5): 353–62. https://doi.org/10.1109/TASSP.1974.1162598.

Schroeder, Manfred R. 1968. "Period Histogram and Product Spectrum: New Methods for Fundamental-Frequency Measurement." *The Journal of the Acoustical Society of America.* https://doi.org/10.1121/1.1910902.

Smaragdis, P., and J. C. Brown. 2003. "Non-Negative Matrix Factorization for Polyphonic Music Transcription." In *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No.03TH8684)*, 177–80. New Paltz, NY, USA: IEEE. https://doi.org/10.1109/ASPAA.2003.1285860.

Wikipedia. 2020. "Nyquist–Shannon Sampling Theorem." *Wikipedia.* https://en.wikipedia.org/w/index.php?title=Nyquist%E2%80%93Shannon_sampling_theorem&oldid=941933031.

Yeh, Chunghsin. 2008. "Multiple Fundamental Frequency Estimation of Polyphonic Recordings," 153.