Pitch estimation in monophonic and polyphonic audio signals

Bernard, Damon

Award date:
2022

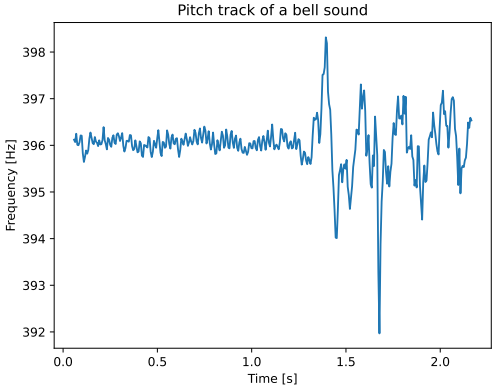
EINDHOVEN UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

BACHELOR FINAL PROJECT

Pitch estimation in monophonic and polyphonic audio signals

Author:
Damon BERNARD
d.bernard@student.tue.nl

Supervisor:
Dr. J. W. PORTEGIES
j.w.portegies@tue.nl



October 7, 2022

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

Abstract

The aim of this study is to give an overview of commonly used methods in single fundamental estimation and multiple fundamental estimation. One of the objectives is to find out what methods are state-of-the-art. It turns out that harmonic summation is used in state-of-the-art pitch correction software. Another objective is to implement the YIN algorithm, the harmonic product spectrum algorithm and the harmonic summation algorithm in software.

Contents

Glossary **4**

1 Introduction **5**

2 Single fundamental estimation **7**

2.1.1 Pitch tracking 8

2.1.2 Error metrics 8

2.2 The YIN algorithm 8

2.2.1 Step 1: Auto-correlation 8

2.2.2 Step 2: Difference function 8

2.2.3 Step 3: Cumulative mean normalised difference function 9

2.2.4 Step 4: Thresholding 9

2.2.5 Step 5: Parabolic interpolation 9

2.2.6 Step 6: Best local minimum 9

Time complexity 10

2.3 Harmonic Product Spectrum 10

Windowing the signal 11

Downsampling the log power spectrum 11

Finding the maximum 11

2.4 Extension of the Harmonic Product Spectrum 12

2.4.1 Example of using a different weight function 13

2.5 Results 14

2.5.1 YIN 14

2.5.2 HPS 15

3 Multiple fundamental estimation **17**

3.1 Harmonic Summation 17

3.1.1 Spectral whitening 17

3.1.2 Salience 18

3.1.3 Searching for a maximum of $\hat{s}$ 19

Linear search 20

Iterative search 20

Time complexity 20

3.1.4 Cancellation 21

3.2 Results 21

3.2.1 Single fundamental estimation 21

3.2.2 multiple fundamental estimation 23

Conclusion **24**

References **25**

A **29**

A.1 Frequency to fourier index 29

A.2 Fourier index to frequency 29

A.3 Frequency to delay 29

A.4 Delay to frequency 29

A.5 Time to sample index 29

A.6 Implementation 30

Keywords: Pitch estimation, YIN, harmonic summation, Harmonic product spectrum

Glossary

K	Length of the discrete Fourier transform.
T_0	The fundamental period T_0 is $\frac{1}{f_0}$. The unit is seconds. Depending of context it may be denoted in terms of sample delay.
$[\cdot] : \mathbb{R} \longrightarrow \mathbb{Z}$	Rounding operator, rounds numbers to the nearest integer.
f_0	A fundamental frequency f_0 is a frequency such that there are no other present frequencies of which f_0 is an integer multiple. The unit of f_0 is Hz. Its name is sometimes abbreviated to just fundamental.
f_N	The Nyquist frequency f_N is equal to $\frac{f_\mathrm{s}}{2}$.
f_s	The symbol f_s stands for the sample frequency. It's also called sample rate. Its value is often 44100. It describes how many samples per second are taken. The unit is Hz.
frame	A part (with smaller length) of an audio signal.
harmonic	The n-th harmonic of a frequency f is the frequency $(n+1)f$. For example, the first harmonic of 400 Hz is 800 Hz.

Chapter 1

Introduction

Digital signal processing (DSP) is an important field of study. This field has wide-ranging applications such as in computer music, seismology and medical sciences [20]. An important topic in DSP is to determine the fundamental frequency in a signal. In literature this is commonly referred to as fundamental estimation, pitch estimation and pitch detection. In computer music, fundamental estimation plays a role in correcting accidental notes on instruments to the correct tuning. Seismologists use fundamental estimation to improve the resistance of buildings against earthquakes.

Fundamental estimation algorithms can be grouped into two categories. The first category is single fundamental estimation and the other is multiple fundamental estimation. The expression f_0 is often used to denote 'fundamental'. For this reason people also sometimes write single f_0 estimation or multiple f_0 estimation. The word 'fundamental' in 'fundamental frequency' indicates that there are higher frequencies that are m integer multiples of this so-called fundamental frequency, where m is allowed to be 0. Unfortunately, fundamental frequency is often not defined mathematically in literature. It is understood rather informally as a frequency f_1 that is peaking in the frequency spectrum such that there is no peaking frequency f_2 that satisfies $f_1 = kf_2$ for some $k \in \mathbb{Z}$. It is not clear what 'peaking' means here, but it is apparent that this pertains to some threshold. We can describe this mathematically, which is done in the introduction of chapter 2.

In audio, a signal is called a *polyphonic* signal if it contains more than one different fundamental, for instance when two different notes are played on a musical instrument at the same time. A signal is otherwise said to be *monophonic*. Thus the categories of fundamental estimation can also be referred to as monophonic f_0 estimation and polyphonic f_0 estimation in the field of audio. A polyphonic signal can be turned into monophonic signals by splitting it up into smaller signals, as long as there is only one fundamental in the signal at a time. A recording where two different notes on a guitar are played *at the same time* is polyphonic. However, it cannot be split up into monophonic signals, because the notes are played too close together. It would be possible to split the recording into smaller monophonic signals if the notes were played further apart in time.

There are a vast number of algorithms to find the fundamental in monophonic signals. In the time domain, signals are analysed purely based on the signal data. However a signal can also be analysed in the frequency domain for which the Fourier transform is used. The most common type of time domain methods is autocorrelation, where a signal is cross-correlated with itself. Many different kinds and implementations of this type of method exist.

Some of the most notable autocorrelation based algorithms are YIN [5], PYIN [22], MPM [24], PRAAT [33], BaNa [2], IPTA [29], Speech Filing System (SFS) [12], RAPT [17]. In some methods autocorrelation is computed according to a (difference function) formula where the signal is subtracted from itself rather

than being multiplied with itself [30][10][19][18]. Other types of time domain methods exist, such as zero-crossing.

Time domain methods operate by performing an algorithm on the audio signal in time domain representation rather than frequency domain representation. Neural networks are also utilised for f_0 estimation, often being used as a hybrid time and frequency domain method. For example deep neural networks [9][34][31], convolutional neural networks (such as CREPE) [15] and recurrent neural networks [32]. These neural networks are often trained with time-frequency data. Methods that use frequency information are frequency domain methods. Frequency domain methods are or involve for example cepstrum analysis [27], harmonic product spectrum (HPS) [8], TEMPO [14], wavelets [23].

Current research in f_0 estimation is mainly about multiple f_0 estimation. Examples of algorithms for multiple f_0 estimation are non-negative matrix factorisation (NMF) [13], convolutional neural networks including CREPE [6][15], a method based on probabilistic latent component analysis (PLCA) and hidden Markov random fields (HMRF) [1], harmonic summation [16], Gaussian spectral modelling (GSM) and hidden markov models (HMM) [3], harmonic summation and spectral cancellation [4], maximum likelihood estimation after a STFT [36], a claimed method using wavelets [25], and a time domain method called MC-HMUSIC[35]. MC-HMUSIC models a signal with a spacio-temporal model. To determine fundamental frequency and the direction-of-arrival (DOA), MC-HMUSIC makes use of linear algebra and joint estimation. The DOA is estimated because it makes it easier to distinguish between the sources that generate the frequencies in case there are multiple sources such as a trumpet and speech.

There are many algorithms to choose from for both single and multiple f_0 estimation. But which of these are the best? One of the goals in this thesis is to find out which methods for single and multiple f_0 estimation are state of the art.

In September 2011 Peter Neubäcker, inventor of the respected pitch correction software Melodyne, patented a note tracking algorithm which uses a multiple f_0 estimation method based on Klapuri's paper on harmonic summation [16][26]. Since then, no patents in the name of P. Neubäcker have been published. This may suggest that Melodyne still uses a similar algorithm to this day.

In 2019 researchers at Google published DDSP [7] which stands for Differentiable Digital Signal Processing and it allows for astounding audio synthesis. Based on their DDSP paper, a demo project called ToneBoosters was created by Google. ToneBoosters can turn a vocal into a trumpet, violin, saxophone, etc. In order to accomplish this, the researchers made use of the convolutional neural network CREPE for multiple f_0 estimation [7].

This thesis elaborates on two single f_0 estimation methods and one multiple f_0 estimation method. The single f_0 estimation methods described are discussed due to their popularity or their simplicity. The multiple f_0 estimation method discussed is harmonic summation, because it is likely still used by state-of-the-art fundamental estimation software. We aim to formally describe these methods and to apply them to a dataset of which the fundamentals are known, in order to evaluate an algorithm's accuracy. The dataset that is used comes from the university of Iowa. This dataset can be downloaded from the website theremin.music.uiowa.edu.

Chapter 2

Single fundamental estimation

In the glossaries, a fundamental (frequency) f_0 is said to be a frequency such that there are no other present frequencies of which f_0 is an integer multiple. It is possible to read off which frequencies have a large amplitude using a Fourier transform of the signal. Fundamental frequency is usually not defined in literature. Definition 2, Definition 3 and Definition 5 are not from any references. These definitions are made up. However, it is important to have a sense of what fundamental frequency means. For multiple fundamental estimation it is convenient to define a fundamental frequency according to Definition 2.

Definition 1 (Fourier transform)**.** Let $x : \mathbb{R} \to \mathbb{R}$ be a signal. The fourier transform $X : \mathbb{R} \to \mathbb{C}$ is defined by $X(\xi) := \int_{-\infty}^{\infty} x(t)e^{-i2\pi\xi t}\ dt$.

Definition 2 (fundamental frequency)**.** Let $x : \mathbb{R} \to \mathbb{R}$ be a signal. Let $\alpha \in \mathbb{R}$ be a threshold value. Let X be the Fourier transform of x, i.e. $X(\xi) = \int_{-\infty}^{\infty} x(t)e^{-i2\pi\xi t}\ dt, \quad \forall \xi \in \mathbb{R}$. The frequency f_0 is called a fundamental frequency if for all $n \in \mathbb{N} : \left|X\left(\frac{f_0}{n}\right)\right| < \alpha$ and $|X\left(f_0\right)| > \alpha$.

One drawback of the above definition is that being a fundamental frequency then also depends on how loud the signal is. It is possible to change the definition to include a normalization step. But this is not satisfactory in case $X(\xi)$ is small for all ξ. For single fundamental estimation, the definition of a fundamental can be simplified to Definition 3, where a time representation is used rather than a frequency domain representation.

Definition 3 (fundamental frequency)**.** If the continuous signal $x(t)$ is periodic, i.e. $x(t)$ satisfies $\exists_{T \in \mathbb{R}} \forall_{t \in \mathbb{R}} : x(t) = x(t+T)$, then $f_0 \in \mathbb{R}$ is defined as $\frac{1}{T_0}$ where T_0 is the smallest $T \in \mathbb{R}$ such that $\forall_{t \in \mathbb{R}} : x(t) = x(t+T)$ holds, with t in seconds.

Signals measured from real world surroundings are discrete and usually not periodic. For discrete signals we define a fundamental according to Definition 5.

Definition 4 (Discrete Fourier transform (DFT))**.** Let $x : \mathbb{Z} \to \mathbb{R}$ be a discrete signal. The fourier transform $X : \mathbb{Z} \to \mathbb{C}$ is defined by $X[k] := \sum_{n=0}^{N-1} x[n]e^{-i2\pi\frac{nk}{N}}$.

Definition 5 (fundamental frequency)**.** Let $x : \mathbb{Z}_{[0,N]} \to \mathbb{R}$ be a discrete signal of length N. Let $\alpha \in \mathbb{R}$ be a threshold value. Let X be the discrete Fourier transform of x, i.e. $X[k] = \sum_{n=0}^{N-1} x[n]e^{-i2\pi\frac{nk}{N}}, \quad \forall k \in \mathbb{N}$. The frequency f_0 is called a fundamental frequency if $\forall n \in \mathbb{N} : \left|X\left[\text{fftindex}\left(\frac{f_0}{n}\right)\right]\right| < \alpha$ and $|X\left[\text{fftindex}\left(f_0\right)\right]| > \alpha$. For the definition of fftindex see Appendix A.1.

2.1.1 Pitch tracking

Pitch detection algorithms can be used to perform pitch tracking. Pitch tracking means to keep track of the fundamental(s) in an audio signal. This is first done by taking small parts of the audio signal. These smaller parts are called frames. After that, to each frame a pitch detection algorithm is applied. Two single fundamental estimation algorithms will be discussed here. The first algorithm is the YIN algorithm. The second algorithm is the Log Harmonic Product Spectrum algorithm.

2.1.2 Error metrics

When researchers use algorithms they may want to check how well those algorithms perform. In studies on speech, the quality of pitch detection algorithms is sometimes measured in Gross Pitch Error (GPE). It is defined by Definition 6 from Mathworks [21].

Definition 6. Gross Pitch Error (GPE) is the proportion of frames, considered voiced by both pitch tracker and ground truth, for which the relative pitch error is higher than a certain threshold. A GPE of 0 is the best and a GPE of 1 is the worst.

Example 1. Suppose that we take five frames of an audio signal. A pitch detection algorithm is applied to each frame. The estimated fundamental frequencies of the frames are respectively $440.1, 400, 745, 390$ Hz and the last frame is considered unvoiced (the algorithm determines this frame has no fundamental). Suppose their ground truths are $440, 392, 740, 392$ and 440 Hz respectively. To determine the GPE, we only consider the first 4 frames because those are considered voiced by both ground truth and the algorithm. If we take a threshold of 20%, then only 3 frames of the 4 are within 20% relative error. So the GPE is 75%.

2.2 The YIN algorithm

Auto-correlation is at the basis of many basic pitch detection algorithms. For example, the YIN and the MPM algorithm are auto-correlation based methods. The YIN algorithm is widely used in speech studies due to its accuracy and it is state of the art in single fundamental estimation. That is why we will discuss it here. How it works can be explained in 6 steps [5].

2.2.1 Step 1: Auto-correlation

Let $x[n]$ be the discrete time signal induced by sampling the continuous signal $x(t)$. Let $W \in \mathbb{N}$ be a chosen window size. A typical value for W would be for example 512, 1024 or 4096 samples. The auto-correlation $r(t, \tau) : \mathbb{Z} \times \mathbb{Z} \to \mathbb{R}$ at time index t for sample delay $\tau \in \mathbb{Z}$ is defined as

$$r(t, \tau) := \sum_{j=t+1}^{t+W} x[j]x[j + \tau].$$

In the original article on YIN, Klapuri shows that step 2 through 6 improve the GPE from 10% to 0.5% when applying the algorithm to a large database comprised of five databases.

2.2.2 Step 2: Difference function

Suppose the real fundamental period is $T \in \mathbb{R}$. Define the difference function $d(t, \tau) : \mathbb{Z} \times \mathbb{Z} \to \mathbb{R}$ by

$$d(t, \tau) := \sum_{j=t+1}^{t+W} (x[j] - x[j + \tau])^2 .$$

If $\tau \approx T$, we have $x[t] - x[t + \tau] \approx 0$ for all t by definition of the fundamental period. Hence $d(t, T) \approx 0$. So the goal is now to find a τ that minimizes d. The τ found is then an estimate for the fundamental period

T. The difference function is related to the auto-correlation in the sense that $d(t, \tau)$ can be rewritten in terms of r. That is,

$$\begin{aligned} d(t,\tau) &= \sum_{j=t+1}^{t+W} (x[j] - x[j+\tau])^2 \\ &= \sum_{j=t+1}^{t+W} x[j]^2 - 2x[j]x[j+\tau] + x[j+\tau]^2. \end{aligned}$$

Writing the sum for each term and reindexing the last sum gives:

$$\begin{aligned} d(t,\tau) &= \sum_{j=t+1}^{t+W} x[j]^2 - 2\sum_{j=t+1}^{t+W} x[j]x[j+\tau] + \sum_{j=t+1}^{t+W} x[j+\tau]^2 \\ &= \sum_{j=t+1}^{t+W} x[j]^2 - 2\sum_{j=t+1}^{t+W} x[j]x[j+\tau] + \sum_{j=t+\tau+1}^{t+\tau+W} x[j]^2 \\ &= r(t,0) + r(t+\tau,0) - 2r(t,\tau). \end{aligned}$$

2.2.3 Step 3: Cumulative mean normalised difference function

Note that $d(t, 0) = 0$. If the algorithm was to return the global minimum of d then we would find the result $\tau = 0$. This is of course not the correct lag that we are looking for. One way of getting around this issue is limiting the search range for τ, but this is not a good way of doing it. Limiting the search range for τ is like limiting the lowest and the highest frequency that the algorithm can detect.

Instead, de Cheveigné proposes the cumulative mean difference function $\hat{d}(t, \tau)$ which is defined by

$$\hat{d}(t,\tau) := \begin{cases} 1, & \text{if } \tau = 0, \\ d(t,\tau) \Big/ \left(\frac{1}{\tau}\sum_{j=1}^{\tau} d(t,j)\right), & \text{for } \tau > 0. \end{cases}$$

Using $\hat{d}$ we don't have to limit the search range for τ in order to prevent the algorithm from returning $\tau = 0$. The next step defines what we need in order to make sure that the minimum we find is small enough.

2.2.4 Step 4: Thresholding

Fix a threshold $A \in [0, 1]$. We consider the set $C := \{\tau \in \mathbb{Z}_{[0,N-W]} \mid \hat{d}(t, \tau) < A\}$ and let $m := |C|$ be the size of C.

2.2.5 Step 5: Parabolic interpolation

It is possible that the period of the original continuous signal is not a multiple of the sampling period. In that case, the estimated signal period has an error of at most $\frac{T_{\text{samp}}}{2}$ seconds with $T_{\text{samp}} = \frac{1}{f_s}$. This error can be reduced by fitting a parabola on the local minima of C. Suppose m local minima $\tau_1, \ldots, \tau_m$ are found. For each local minimum $\tau_i \in C$ define the fitted parabola p_i. Each parabola p_i is fit on 3 datapoints: $\left(\tau_i - 1, \hat{d}(t, \tau_i - 1)\right), \left(\tau_i, \hat{d}(t, \tau_i)\right)$ and $\left(\tau_i + 1, \hat{d}(t, \tau_i + 1)\right)$. Since τ_i is where the local minimum is, each parabola is upwards. Define $C' := \{p_i : i = 1, \ldots, m\}$.

2.2.6 Step 6: Best local minimum

We say that $\hat{T} := \min_{p_i \in C'} \operatorname{argmin}_{t \in \mathbb{R}} p_i(t)$. And thus the estimated fundamental is $\hat{f}_0 = \frac{1}{\hat{T}}$.

Time complexity

The components of this algorithm that take the longest time are finding the best local minimum, and taking the cumulative mean normalised difference function. Suppose the YIN algorithm is applied to a frame of size W. Fix t corresponding to where that frame starts. In step 4 the algorithm loops over $\tau = 0, \dots, N-W$ to find the local minima of C and for each τ it loops over $j = t+1, \dots, t+W$ to evaluate $d(t, \tau)$ in the expression of $\hat{d}(t, \tau)$. This gives a time complexity of $O\left((N - W)W\right)$. For pitch tracking on an audio signal of length N, the algorithm is applied $N - W$ times. So the time complexity then becomes $O\left((N - W)^2 W\right) = O(N^2)$. This time complexity can however be made $O(N \log N)$, by making use of the FFT (fast Fourier transform) [5, p. 7].

To see the YIN algorithm in action, we apply it to the sound of a bell with a fundamental frequency of 396 Hz and obtain the pitch track visible in Figure 2.1. To obtain this pitch track, the audio signal was split into frames of size 1024. A frame is shifted 256 samples to the right of the start of the previous frame. The first frame has elements $x[0], \dots, x[1023]$. The second frame has elements $x[256], \dots, x[1023 + 256]$, etc.

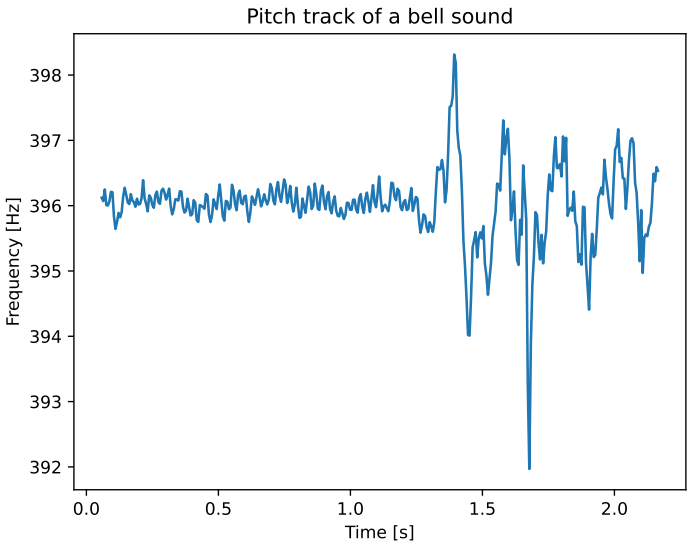


Figure 2.1: Pitch track of a bell sound using YIN

If the paramter α is chosen to be 20% in Definition 6, then the GPE is 100% for this particular sound.

2.3 Harmonic Product Spectrum

The Harmonic Product Spectrum (HPS) method is a frequency domain method. It is not state of the art,but the advantage if this method is its simplicity and efficiency. There is also the Log Harmonic Product Spectrum (LHPS) method. The only difference is that in LHPS a log power spectrum is downsampled and in HPS it is just the power spectrum that is being downsampled. Downsampling by a factor $d \in \mathbb{Z}$ is when you have a sequence defined on indices in $\{0, 1, 2, 3, \dots\}$ and take a subsequence with indices from $\{0, d, 2d, 3d, \dots\}$.

This paragraph will provide a sketch of the LHPS method. Consider the discrete signal $x : \mathbb{N} \to \mathbb{R}$. Apply windowing to the signal. Say we have applied $p \in \mathbb{Z}$ windows (each with window length W) and so we ended up with p segments. For each segment S_i where $1 \leqslant i \leqslant p$, compute the DFT, take the element-wise absolute value of the resulting DFT and then the element-wise logarithm and call the resulting sequence

$\mathcal{L}_{S_i}$. The sequence $\mathcal{L}_{S_i}$ is mathematically denoted as $\mathcal{L}_{S_i} := \log\left|\hat{S}_i\right|$ for $1 \leqslant i \leqslant p$, where $\hat{S}_i$ is $\mathrm{DFT}(S_i)$ with size K and $\log\left|\hat{S}_i\right|$ is taking an element-wise absolute value and an element-wise logarithm. Note that $\mathcal{L}_{S_i}$ has size K as well. Let $m = 3$, accordingly to [8]. The next step is to downsample $\mathcal{L}_{S_i}$ by $d \in \mathbb{N}$ for $d = 1, 2, \ldots, m$ and call the downsampled result $\mathcal{L}_{S_i,d}$. Next, we multiply $\mathcal{L}_{S_i,d}$ for $d = 1, \ldots, m$ with each other via a Hadamard product to get the estimated fundamental frequency, i.e. the estimated f_0 will be the argmax of $\prod_{d=1}^{m} \mathcal{L}_{S_i,d}$.

The LHPS algorithm is explained in more detail in the next paragraphs.

Windowing the signal

Consider a discrete signal x with elements $x[n]$. Choose a window w of window size W such as the Hamming window [28][11, p. 62] which is defined by $w : \mathbb{Z}_{[0,W]} \to \mathbb{R}$

$$w[n] := \frac{54}{100} - \left(1 - \frac{54}{100}\right)\cos\left(\frac{2\pi n}{W-1}\right).$$

The significance of the constant $\frac{54}{100}$ is that it is is a close approximation to $\frac{25}{46}$ [11, p. 62]. Using this window w, we compute frames $S_i : \mathbb{Z}_{[0,W]} \to \mathbb{R}$ by

$$S_i[n] = x[i \cdot \Delta + n] \cdot w[n]$$

where $\Delta \in \mathbb{N}$. The value of Δ depends on how far apart the start of consecutive segments should be from each other. We define the absolute log value of a sequence (or segment in this context) S as a new sequence

$$\log|S| := (\log|S[j]|)_{j=1,\ldots,W}.$$

Downsampling the log power spectrum

For each segment $S_i : \mathbb{Z}_{[0,W]} \to \mathbb{R}$, let $\mathcal{L}_{S_i} = \log\left|\hat{S}_i\right|$, where the hat denotes the DFT (see Definition 4). The downsampling or time scaling goes as follows. Say we would like to downsample $\mathcal{L}_{S_i}$ by a factor $d \in \mathbb{N}$. Then we define the downsampled result $\mathcal{L}_{S_i,d}$ by

$$\mathcal{L}_{S_i,d}[n] := \mathcal{L}_{S_i}[d \cdot n],$$

where n ranges from $n = 1$ to $n = \left\lfloor \frac{W}{d} \right\rfloor$, otherwise $\mathcal{L}_{S_i,d}$ would be undefined. For example, suppose we wanted to downsample by a factor of 2, then we would get the sequence

$$\mathcal{L}_{S_i,2} := \left(\mathcal{L}_{S_i}[2], \mathcal{L}_{S_i}[4], \ldots, \mathcal{L}_{S_i}\left[2\left\lfloor \frac{W}{2} \right\rfloor\right]\right).$$

Note that by downsampling we obtain less samples than we started with. If we downsample by a factor of 2, the length of the result will be approximately half of the length we started with (depending on if W is even or odd). Why does this help in estimating the fundamental frequency? We assume that the fundamental induces harmonics at multiples of the fundamental, i.e. $2f_0, 3f_0, \ldots$. Hence if we downsample a signal containing peaks at $2f_0, 3f_0, \ldots$ by an integer factor then we will still see a peak at the index corresponding to frequency f_0 after downsampling n times, the peak at f_0 in the downsampled result comes from the $(n-1)$-th harmonic, assuming there are at least $n - 1$ harmonics.

Finding the maximum

Finally, determine the index where the log harmonic product spectrum has the highest peak:

$$k = \operatorname*{argmax}_j \left(\prod_{d=1}^{W} \mathcal{L}_{S_i,d}[j]\right),$$

which is our estimate for the index of f_0. Using Appendix A.2, the index k can be converted to frequency f_0. Figure 2.2 shows an overview for harmonic product spectrum (HPS) method. The difference between HPS and log harmonic product spectrum (LHPS) is that the latter makes use of the log power spectrum whereas the first makes use of the regular power spectrum. The LHPS method has the property that the accuracy gets better as W gets larger, because we can downsample more. However, we would need a larger DFT size to increase W. It would thus require more time for the processor to estimate f_0 because it needs to use more computing power for the Fourier transform.

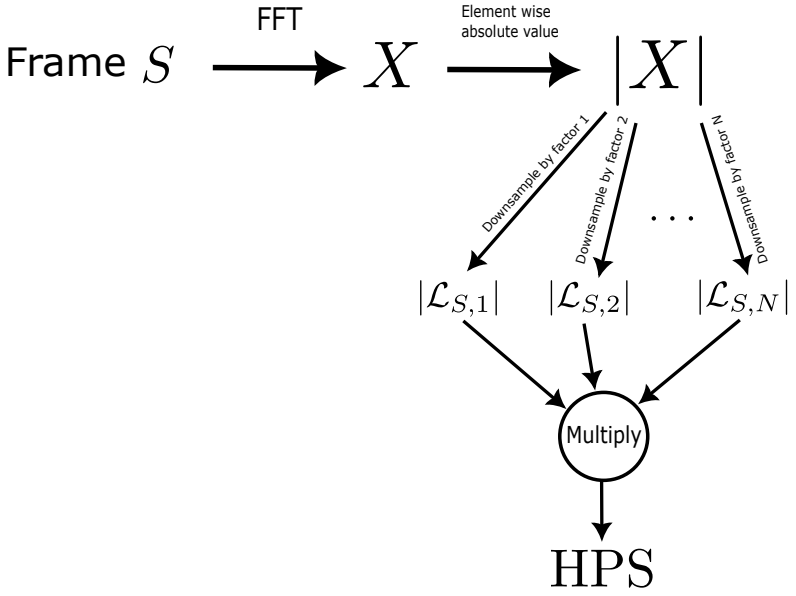


Figure 2.2: Overview of how the HPS is computed for a frame S

Figure 2.3 shows the pitch track of a vocal using HPS, with time in seconds on the horizontal axis and frequency in Hz on the vertical axis. In section 2.1.1, pitch tracking has been explained. In this case, the audio signal has been split into frames of size 4096 and the HPS algorithm has been applied to each frame, giving an f_0 estimate for each frame. It can be seen that there are some outliers. They are outliers because humans cannot normally sing above 2000 Hz. The singer was inactive in the case of these outliers, with as result that the maximum of the HPS is unpredictable because there is nothing with a pitch playing if the singer isn't singing. No ground truth is available for this vocal and frequencies have been quantised to the nearest note in the western scale.

2.4 Extension of the Harmonic Product Spectrum

The Harmonic product spectrum method can be extended to computing a convolution if the decimated spectra are summed instead of multiplied. Adding the spectra should still give a peak at the index of the sought after fundamental. In the next paragraphs we will show that the HPS can be written in terms of a convolution and that

Suppose the power spectrum $\mathcal{L}(x^{(N)})$ of a discrete signal $x^{(N)}$ of length N is given. Denote this power spectrum by $f : \mathbb{Z}_{[0,K]} \to \mathbb{R}$. The result $\mathrm{HPS} : \mathbb{Z}_{[0,\lfloor \frac{N}{2d} \rfloor]} \to \mathbb{R}$ after summing the decimated spectra at index n is

$$\mathrm{HPS}[n] = f[n] + f[2n] + \cdots + f[dn], \qquad n = 0, \ldots, \left\lfloor \frac{N}{2d} \right\rfloor.$$

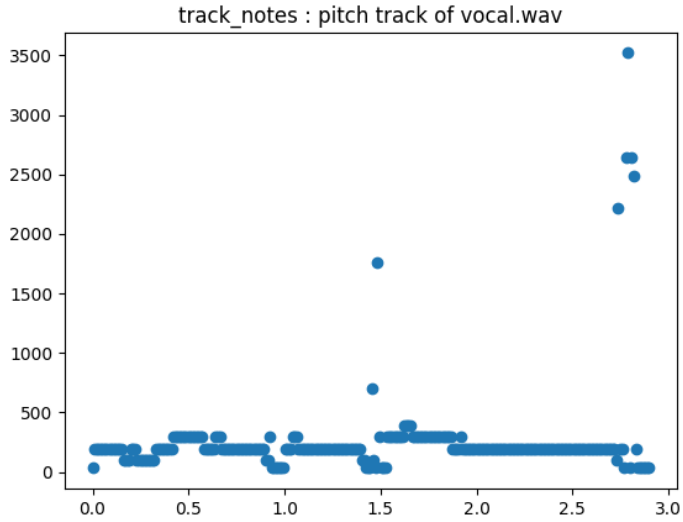


Figure 2.3: Pitch tracking a vocal by a singer using HPS

This can be written as the sum of a product if we let $g_n : \mathbb{Z}_{[0,\frac{N}{2}]} \to \mathbb{R}$ be given by $i \mapsto g_n[i]$ where $g_n[i] = \delta_n(i) + \delta_{2n}(i) + \cdots + \delta_{dn}(i)$. Note that $\delta_j(i)$ is the Dirac delta function, it equals 1 if $i = j$ and 0 otherwise. Then we can rewrite HPS as

$$\begin{aligned}\text{HPS}[n] &= f[n] + f[2n] + \cdots + f[dn] \\ &= \sum_{m=0}^{\frac{N}{2}} \left(f \cdot (\delta_n + \delta_{2n} + \cdots + \delta_{dn})\right)[m] \\ &= \sum_{m=0}^{\frac{N}{2}} \left(f \cdot g_n\right)[m].\end{aligned}$$

Now let $g : \mathbb{R}_{[0,\frac{N}{2}]} \to \mathbb{R}$ be a function given by $g(x) = \delta_1(x) + \cdots + \delta_d(x)$. Then

$$\text{HPS}[n] = \sum_{m=0}^{\frac{N}{2}} f[m] \cdot g\left(\frac{m}{n}\right), \tag{2.1}$$

which corresponds to the convolution $\int_0^{\frac{N}{2}} f(x) g\left(\frac{x}{y}\right)\, dx$.

2.4.1 Example of using a different weight function

Instead of using impulses $\delta_1, \delta_2, \ldots, \delta_d$ for g, we can use Gaussians centered at $1, 2, \ldots, d$. Fix an $a \in \mathbb{R}_{>0}$. Let $h : \mathbb{R} \longmapsto \mathbb{R}$ be defined by $h(z) = \sum_{i=1}^{d} e^{-a(z-i)^2}$. If we choose such a function h for g in Equation 2.1 we get

$$\text{HPS}[n] = \sum_{m=0}^{\frac{N}{2}} \sum_{i=1}^{d} f[m] \exp\left(-a\left(\frac{m}{n} - i\right)^2\right).$$

Since d is constant, the time complexity of computing $\mathrm{HPS}[n]$ for some n is $O(N)$ and therefore the time complexity to compute the whole harmonic product spectrum HPS directly using the above formula for $n = 1, \ldots, \lfloor \frac{N}{2d} \rfloor$ is $O(N^2)$. Computing the HPS is equivalent to computing the product

$$\begin{aligned}\mathrm{HPS} &= \begin{pmatrix} h\left(\frac{0}{1}\right) & h\left(\frac{1}{1}\right) & \ldots & h\left(\frac{\frac{N}{2}}{1}\right) \\ \vdots & & & \vdots \\ h\left(\frac{0}{\lfloor \frac{N}{2d} \rfloor}\right) & h\left(\frac{1}{\lfloor \frac{N}{2d} \rfloor}\right) & \ldots & h\left(\frac{\frac{N}{2}}{\lfloor \frac{N}{2d} \rfloor}\right) \end{pmatrix} \begin{pmatrix} f[0] \\ \vdots \\ f\left[\frac{N}{2}\right] \end{pmatrix} \\ &=: H \begin{pmatrix} f[0] \\ \vdots \\ f\left[\frac{N}{2}\right] \end{pmatrix}.\end{aligned}$$

Experimentally it can be shown that at least 60% of the entries in H are unique for most d and N. As N gets larger with d fixed, the percentage of unique entries in H get closer to 100%. Due to these unique entries in H, almost every entry in H must be computed. So there is no way to make this computation with this choice of the function h into a time complexity less than $O(N^2)$. A different choice of h may allow one to implement the computation with an improved time complexity.

2.5 Results

For this project, the algorithms have been implemented in code. It would be nice to see how these algorithms perform on a dataset. As mentioned previously, we will check the performance on the IOWA dataset.

2.5.1 YIN

The samples in the dataset are described in Table 2.1. To every sample we apply pitch tracking with the YIN algorithm.

Instrument	Number of samples
Alto Flute	35
Alto Sax	64
Bass	208
Bass Clarinet	46
Bass Flute	38
Bass Trombone	27
Bassoon	40
Bb Clarinet	46
Cello	195
Eb Clarinet	39
Flute	77
Horn	44

Instrument	Number of samples
Marimba	303
Oboe	35
Sop Sax	64
Tenor Trombone	33
Trumpet	36
Tuba	37
Vibraphone	167
Viola	200
Violin	182
Xylophone	175
bells	82

Table 2.1: Dataset description

The algorithm has been configured to look for a fundamental in a range from 30 Hz to 3000 Hz. The window length was set to 1024 and the window step size was set to 256. The GPE is the proportion of voiced frames whose estimated fundamental has a relative error of 0.05 from the ground truth.

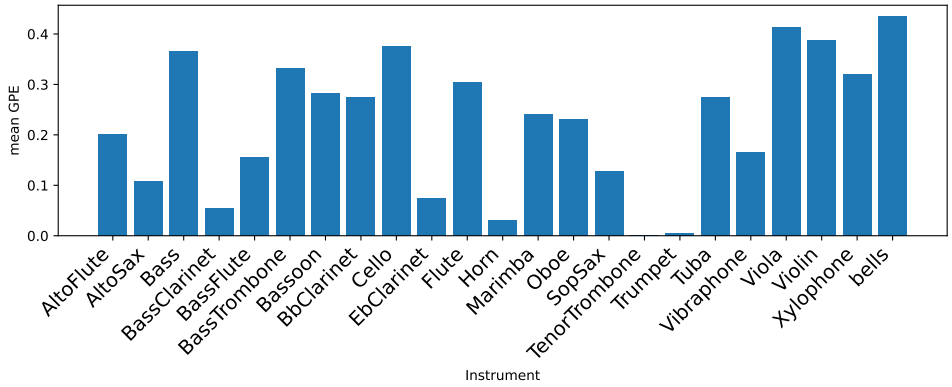


Figure 2.4: Bar plot of the mean GPE per instrument, using the YIN algorithm and the IOWA dataset

From the results visible in Figure 2.4 it becomes apparent that the YIN algorithm worked best on the bass clarinet instrument. The algorithm did the worst on bass and bells. The average GPE over all samples was 0.29.

2.5.2 HPS

For the HPS method we used a smaller subset of the IOWA dataset (see Table 3.1) than the one we used for YIN. This is because it takes a very long time to run the performance tests on the database.

Instrument	Number of samples
Alto Flute	35
Alto Sax	64
Bass	208
Bass Clarinet	46
Bass Flute	38
Bass Trombone	27
Bassoon	40
Bb Clarinet	46
Cello	195
Eb Clarinet	39
Flute	77
Horn	44

Instrument	Number of samples
Marimba	303
Oboe	35
Sop Sax	64
Tenor Trombone	33
Trumpet	36
Tuba	37
Vibraphone	139
bells	82

Table 2.2: Dataset description

The algorithm has been configured to split signals up into frames of length 1024.

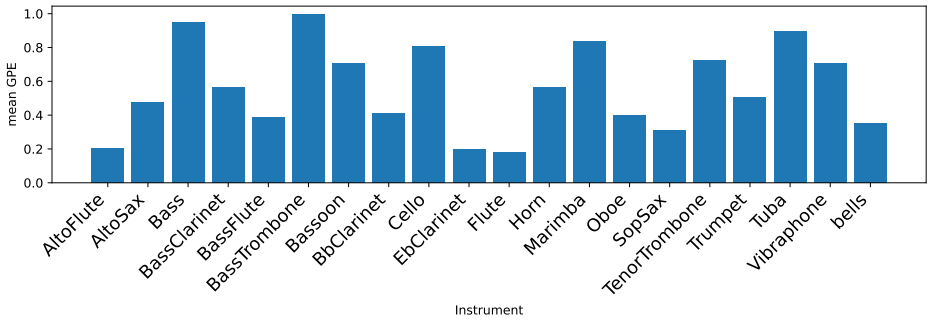


Figure 2.5: Bar plot of the mean GPE per instrument, using the HPS algorithm and the IOWA dataset

The overall GPE is higher than for the YIN algorithm. The highest GPE for YIN was around 0.4, and for HPS the highest GPE was around 1. There is no instrument that the HPS method performs really well on. In fact, there are many more that it doesn't perform well on at all. This may be the result from a frequency resolution that is too small after applying the FFT.

Chapter 3

Multiple fundamental estimation

Multiple fundamental estimation is needed when an audio signal has multiple sources, such as when there are two people singing at the same time. As mentioned previously, a fundamental frequency is defined by Definition 2 when doing multiple f_0 estimation. In this chapter, harmonic summation will be discussed. Harmonic summation is used by Melody, a state-of-the-art pitch correction software.

3.1 Harmonic Summation

First in this section, a broad overview is given of the method from Klapuri's paper on harmonic summation [16]. The method is explained in more detail starting from section 3.1.1. First spectral whitening is applied in order to suppress timbral information. Timbral information doesn't help with finding fundamental frequencies. Timbral information can be for example a blowing sound when someone plays a flute. Then the so-called salience is computed for the spectral whitened spectrum. Salience is a measure for how strong a frequency and its harmonic are, it is denoted by $s(\tau)$. In the sum, harmonics have a certain weight. The function that determines the weights for a harmonic is called g. Klapuri considered different possibilities for g. Then he optimised this weight function to get the highest accuracy in terms of frame error. The salience function $s(\tau)$ depends on g. Once the weight function g has been established, the next step is to find the argmax of $s(\tau)$. Klapuri provides an algorithm that does this quickly by means of divide and conquer. Then the process of spectral cancellation is discussed after a fundamental has been found. Using this spectral cancellation one can estimate how many sources there are in the original signal.

3.1.1 Spectral whitening

The first step in the algorithm is to spectrally whiten the spectrum X obtained from the discrete Fourier transform. The result we get is called the whitened spectrum Y.

In order to spectrally whiten X, Klapuri first defines *critical frequencies* $c_b \in \mathbb{R}$ for $b = 1, \ldots, 30$. It is around these critical frequencies that we will sum values of spectrum X. So, split the frequencies from 0 Hz up to 7 kHz into the critical band scale c_b in Hz defined by

$$c_b := 229 \cdot \left(10^{\frac{b+1}{21.4}} - 1\right).$$

How the constants in this formula were determined is unknown and the only relevant mention regarding units is that the numbers c_b are in Hz. There are also no references in the original article. The spectrum X will later be convolved with responses $H_b : \mathbb{Z}_{[0,K]} \longrightarrow \mathbb{R}$. These responses are tent functions. Define

$T : \mathbb{R}^2_{[0,K]} \longrightarrow [0,1]$ by:

$$T(b,j) := \begin{cases} j-b+1, & b-1 < j \leqslant b \\ b-j+1, & b < j < b+1 \\ 0, & \text{else} \end{cases}$$

To define the responses H_b we need a conversion operator that we will call fftindex which converts frequency in Hz to the corresponding DFT index. Define the operator fftindex as $\text{fftindex}(f_s, K, f) = \left\lfloor \frac{2f \cdot K}{f_s} \right\rceil$ where the DFT size K and the samplerate f_s may be left out as arguments if they can be derived from the context. We then let $H_b(k) := T(\text{fftindex}(c_b), k)$. For the definition of fftindex see Appendix A.1. When plotted, these H_b look like:

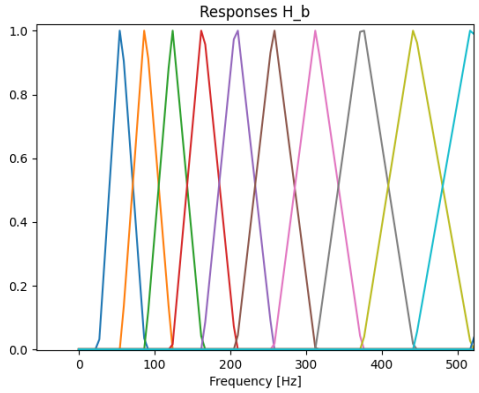


Figure 3.1: Plots of H_b for multiple values of b.

Writing out H_b explicitly gives:

$$H_b(k) = \begin{cases} 0, & \frac{f_s K}{2k} < c_{b-1}, \\ \frac{1}{c_b - c_{b-1}} \left(\frac{f_s K}{2k} - c_{b-1} \right), & c_{b-1} \leqslant \frac{f_s K}{2k} < c_b, \\ \frac{1}{c_b - c_{b+1}} \left(\frac{f_s K}{2k} - c_{b+1} \right), & c_b \leqslant \frac{f_s K}{2k} < c_{b+1}, \\ 0, & \frac{f_s K}{2k} > c_{b+1}, \end{cases}$$

for $k = 0, \ldots, K-1$. These H_b are convolved with the spectrum X to obtain σ_b. Define:

$$\sigma_b := \left(\frac{1}{K} \sum_k^K H_b(k) |X(k)|^2 \right)^{1/2}, \quad \text{for } b = 1, \ldots, 30.$$

Let $\gamma_b := \sigma_b^{\nu - 1}$ with $\nu \in \mathbb{R}_{[0,1]}$. These γ_b are linearly interpolated so that for each FFT index k we have a value $\gamma(k)$. The whitened spectrum Y can be obtained by

$$Y(k) := \gamma(k) X(k), \quad \text{for } k = 0, \ldots, K-1.$$

3.1.2 Salience

Salience measures how loud a frequency and its harmonics are in a (whitened) spectrum Y. In Klapuri's paper, the salience function[16, p. 1] is defined as

$$s(\tau_{\text{low}}, \tau_{\text{up}}) = \sum_{m=1}^{M} g(\tau_{\text{low}}, \tau_{\text{up}}, m) \max_{k \in \kappa(\tau_{\text{low}}, \tau_{\text{up}}, m)} |Y(k)|,$$

where

$$\kappa(\tau_{\text{low}}, \tau_{\text{up}}, m) = \left\{ y \in \mathbb{Z} \mid \left\lceil \frac{2mK}{\tau_{\text{up}}} \right\rceil \leqslant y \leqslant \left\lceil \frac{2mK}{\tau_{\text{low}}} \right\rceil \right\}$$

and

$$g(\tau_{\text{low}}, \tau_{\text{up}}, m) = \frac{f_s/\tau_{\text{up}} + \alpha}{m f_s/\tau_{\text{low}} + \beta}$$

with $\alpha = 27$ and $\beta = 320$. However, it is useful to think of the salience in terms of frequencies rather than delays. So define $\hat{s}(f_{\text{low}}, f_{\text{up}})$ such that $\hat{s}(f_{\text{low}}, f_{\text{up}}) = \hat{s}\left(\frac{f_s}{\tau_{\text{up}}}, \frac{f_s}{\tau_{\text{low}}}\right) := s(\tau_{\text{low}}, \tau_{\text{up}})$ by:

$$\hat{s}(f_{\text{low}}, f_{\text{up}}) := \sum_{m=1}^{M} \frac{f_{\text{up}} + \alpha}{m f_{\text{low}} + \beta} \max_{k \in \hat{\kappa}(f_{\text{low}}, f_{\text{up}}, m)} |Y(k)|,$$

with $\hat{\kappa}(f_{\text{low}}, f_{\text{up}}, m) = \left\{ y \in \mathbb{Z} \mid \left\lceil \frac{2mK f_{\text{low}}}{f_s} \right\rceil \leqslant y \leqslant \left\lceil \frac{2mK f_{\text{up}}}{f_s} \right\rceil \right\}$.

In the original article, it is claimed that the salience goes down as the iterative search algorithm in section 3.1.3 runs. However, no proof is given for this. In Lemma 1 this will be proven. This lemma is needed because then we can be sure that splitting an interval into smaller intervals will only decrease the salience and since we're only looking for the maximum of the salience, we can safely ignore those intervals.

Lemma 1. *Let f_{m} be a frequency satisfying $f_{\text{low}} < f_{\text{m}} < f_{\text{up}}$. Then we have $\hat{s}(f_{\text{low}}, f_{\text{m}}) \leqslant \hat{s}(f_{\text{low}}, f_{\text{up}})$ and $\hat{s}(f_{\text{m}}, f_{\text{up}}) \leqslant \hat{s}(f_{\text{low}}, f_{\text{up}})$.*

Proof.

$$\begin{aligned}
\hat{s}(f_{\text{low}}, f_{\text{m}}) &= \sum_{m=1}^{M} \frac{f_{\text{m}} + \alpha}{m f_{\text{low}} + \beta} \max_{k \in \hat{\kappa}(f_{\text{low}}, f_{\text{m}}, m)} |Y(k)| \\
&\leqslant \sum_{m=1}^{M} \frac{f_{\text{m}} + \alpha}{m f_{\text{low}} + \beta} \max_{k \in \hat{\kappa}(f_{\text{low}}, f_{\text{up}}, m)} |Y(k)| \\
&< \sum_{m=1}^{M} \frac{f_{\text{up}} + \alpha}{m f_{\text{low}} + \beta} \max_{k \in \hat{\kappa}(f_{\text{low}}, f_{\text{up}}, m)} |Y(k)| \\
&= \hat{s}(f_{\text{low}}, f_{\text{up}})
\end{aligned}$$

And similarly,

$$\begin{aligned}
\hat{s}(f_{\text{m}}, f_{\text{up}}) &= \sum_{m=1}^{M} \frac{f_{\text{up}} + \alpha}{m f_{\text{m}} + \beta} \max_{k \in \hat{\kappa}(f_{\text{m}}, f_{\text{up}}, m)} |Y(k)| \\
&\leqslant \sum_{m=1}^{M} \frac{f_{\text{up}} + \alpha}{m f_{\text{m}} + \beta} \max_{k \in \hat{\kappa}(f_{\text{low}}, f_{\text{up}}, m)} |Y(k)| \\
&< \sum_{m=1}^{M} \frac{f_{\text{up}} + \alpha}{m f_{\text{low}} + \beta} \max_{k \in \hat{\kappa}(f_{\text{low}}, f_{\text{up}}, m)} |Y(k)| \\
&= \hat{s}(f_{\text{low}}, f_{\text{up}})
\end{aligned}$$

□

3.1.3 Searching for a maximum of $\hat{s}$

There are two ways in which the maximum of the function $\hat{s}$ may be found. The first is by a linear search and the second is by an iterative search described in Klapuri's paper [16, p. 4]

Linear search

Fix f_{low} and f_{up} such that $f_{\text{low}} < f_{\text{up}}$. Fix $N \in \mathbb{N}$. Consider dividing the interval $[f_{\text{low}}, f_{\text{up}}]$ into $N-1$ subintervals of size $\frac{f_{\text{up}}-f_{\text{low}}}{N}$. Let $F_i = i\frac{f_{\text{up}}-f_{\text{low}}}{N} + f_{\text{low}}$. The objective is to find

$$\max_{i=0,\ldots,N-1} \hat{s}(F_i, F_{i+1}).$$

In the linear search, the function $\hat{s}$ is computed N times, so we say that it has time complexity $O(N)$.

Iterative search

The following algorithm is a slight alteration of the algorithm from the original paper, in the sense that we're using frequencies rather than delays.

Fix frequencies f_{low} and f_{up} such that $f_{\text{low}} < f_{\text{up}}$. Let $\tau_{\text{precision}} \in \mathbb{R}$, which will be used for the stopping criterion. Let I_{best} be the set $I_{\text{best}} = \{f_{\text{low}}, f_{\text{up}}\}$. Let $\mathcal{I} = \{I_{\text{best}}\}$.

The first step is to remove I_{best} from $\mathcal{I}$. Now consider the elements l_{best} and r_{best} in I_{best}, with $l_{\text{best}} < r_{\text{best}}$. Let $f_{\text{m}} = \frac{f_s}{\left(\frac{f_s}{l_{\text{best}}} + \frac{f_s}{r_{\text{best}}}\right)/2} = \frac{2l_{\text{best}}r_{\text{best}}}{l_{\text{best}}+r_{\text{best}}}$ which is the harmonic mean of l_{best} and r_{best}. Now update $\mathcal{I}$ according to:

$$\mathcal{I} \longleftarrow \mathcal{I} \bigcup \{\{l_{\text{best}}, f_{\text{m}}\}, \{f_{\text{m}}, r_{\text{best}}\}.\}$$

Due to Lemma 1 we don't need to worry about whether f_{m} is chosen well. It is only necessary for f_{m} to satisfy $l_{\text{best}} < f_{\text{m}} < r_{\text{best}}$. If the frequency we are looking for that gives the maximum is not in $\mathbb{R}_{[l_{\text{best}}, f_{\text{m}}]}$ or in $\mathbb{R}_{[f_{\text{m}}, r_{\text{best}}]}$ and another interval has a higher salience value, then we can justify ignoring the intervals $\mathbb{R}_{\{l_{\text{best}}, f_{\text{m}}\}}$ and $\mathbb{R}_{\{f_{\text{m}}, r_{\text{best}}\}}$. In the next step we redefine

$$I_{\text{best}} := \text{argmax}_{\{l,r\}\in\mathcal{I}}\, \hat{s}(l, r).$$

Suppose that $I_{\text{best}} = \{l_{\text{best}}, r_{\text{best}}\}$. Go back to step one if $\frac{f_s}{r_{\text{best}}-l_{\text{best}}} > \tau_{\text{precision}}$. Otherwise, stop the algorithm. Note that $\frac{f_s}{r_{\text{best}}-l_{\text{best}}}$ converts the frequency $r_{\text{best}} - l_{\text{best}}$ to a delay according to Appendix A.3.

The advantage of the iterative search algorithm is that it is possible to specify how small the resulting interval I_{best} should be. In the linear search, it is possible to get the same effect by choosing N big enough in order to get a smaller frequency precision that is to your liking. However, in that case the linear search would be forced to consider more intervals when searching for a maximum, whereas the iterative search may quickly converge to the correct interval.

Time complexity

In the worst case scenario, this algorithm has to halve all subintervals to a length of $f_{\text{precision}}$. Finding out how many times we need to halve the interval $[f_{\text{low}}, f_{\text{up}}]$ until we get an interval size/precision of $f_{\text{precision}}$ is equivalent to solving for x in:

$$\frac{f_{\text{up}} - f_{\text{low}}}{2^x} = f_{\text{precision}}.$$

That means it takes $\left\lceil \log_2\left(\frac{f_{\text{up}}-f_{\text{low}}}{f_{\text{precision}}}\right)\right\rceil$ number of halvings to get to the desired precision. So, an upper bound for the number of time that the iteration algorithm has to be performed is

$$\begin{aligned}\sum_{i=1}^{\left\lceil \log_2\left(\frac{f_{\text{up}}-f_{\text{low}}}{f_{\text{precision}}}\right)\right\rceil} 2^i &= 2^{\left\lceil \log_2\left(\frac{f_{\text{up}}-f_{\text{low}}}{f_{\text{precision}}}\right)\right\rceil} - 1 \\ &< 2^{\log_2\left(\frac{f_{\text{up}}-f_{\text{low}}}{f_{\text{precision}}}\right)+1} - 1 \\ &= \frac{2(f_{\text{up}} - f_{\text{low}})}{f_{\text{precision}}} - 1.\end{aligned}$$

If we let $N \in \mathbb{N}$ be $N = \left\lceil \frac{f_{\text{up}} - f_{\text{low}}}{f_{\text{precision}}} \right\rceil$ in the linear search, then the iterative search would be performed at most $\frac{2(f_{\text{up}} - f_{\text{low}})}{f_{\text{precision}}} - 1 \leqslant 2N - 1$ times. In each iteration of the iterative search algorithm, the salience function is called once. So the salience function is called at most $2N - 1$ times. This means that, just like the linear search, the iterative search algorithm is of order $O(N)$.

3.1.4 Cancellation

Using harmonic summation up till this point already allows one to perform single fundamental estimation. However, a signal may contain multiple fundamental frequencies. The following process, called cancellation, will make it possible to detect multiple fundamental frequencies present in a signal. The first step is to detect a fundamental frequency by using the iterative search algorithm. When that is done, compute the corresponding FFT index of that fundamental frequency and lower the FFT data at this FFT index.

Initialise the residual spectrum Y_R as the whitened spectrum Y. Furthermore initialise the spectrum of detected sounds Y_D such that $Y_D(k) = 0$ for $k = 0, \ldots, K-1$. Let $j = 0$ which will play the role of counting how many iterations are done.

- Increment j by one.
- Using the iterative search algorithm we obtain an estimate $\hat{f}_j$ for a fundamental frequency present in the signal. Let J be the set of multiples of the index corresponding to the index of $\hat{f}_j$, i.e. $J = \left\{ \left\lfloor \frac{mK\hat{f}_j}{f_s} \right\rfloor \in \mathbb{Z} : m \in \mathbb{Z}, \frac{mK\hat{f}_j}{f_s} \leqslant K \right\}$. For the sake of accuracy, we are using $\left\lfloor \frac{2mK\hat{f}_j}{f_s} \right\rfloor$ rather than $m \left\lfloor \frac{2K\hat{f}_j}{f_s} \right\rfloor$.
- For each index k in J, multiply it with $g(\hat{f}_j, \hat{f}_j, m)$ where the value of m is the one corresponding to the value of m used to get index k. Then set $Y_D(k)$ equal to $Y(k)g(\hat{f}_j, \hat{f}_j, m)$.
- Let d=1. Update Y_R according to $Y_R(k) = \max(0, Y(k) - dY_D(k))$.
- The fundamental estimations made until this point are $\hat{f}_i$ where $i = 1, \ldots, j$. Let $S(x) : \mathbb{Z} \to \mathbb{R}$ be defined by $S(x) = \frac{\sum_{i=1}^{x} \hat{f}_i}{x^\gamma}$ with $\gamma = 0.70$. If $S(j) > S(j-1)$, repeat the previous steps again. Otherwise, stop.

In the cancellation algorithm, d is chosen to be 1 since in the original article d is chosen to be ≈ 1, but it is not specified what it is exactly.

3.2 Results

3.2.1 Single fundamental estimation

We will apply harmonic summation to check for single fundamental estimation.

The harmonic summation algorithm has been configured to split signals up into frames of length 1024. Zeroes are then appended to increase the frequency resolution of the FFT.

Instrument	Number of samples
Alto Flute	35
Alto Sax	64
Bass	208
Bass Clarinet	46
Bass Flute	38
Bassoon	40
Bb Clarinet	46
Cello	195
Eb Clarinet	39
Flute	77
Horn	44

Instrument	Number of samples
Marimba	303
Oboe	35
Sop Sax	64
Tenor Trombone	33
Trumpet	36
Tuba	37
Vibraphone	139
bells	82

Table 3.1: Description of dataset used for harmonic summation

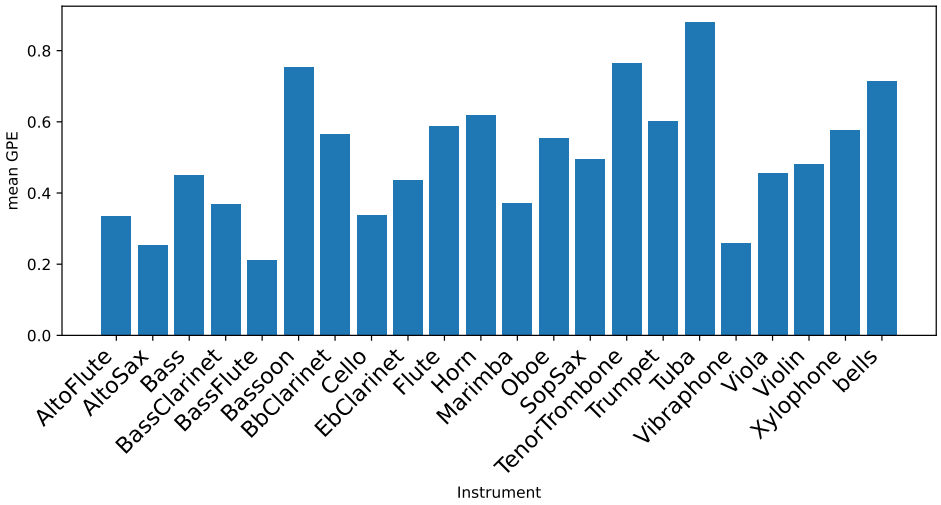


Figure 3.2: Bar plot of the mean GPE per instrument, using the harmonic summation algorithm and the IOWA dataset

The average GPE was around 0.50. After looking at the algorithm this could have been caused by the algorithm sometimes predicting a frequency that is half of the actual fundamental frequency.

We can also apply it to the vocal (see Figure 3.3) that we have applied HPS to earlier (see Figure 2.3). Apart from the outlier datapoints, Figure 3.3 is correct. This has been hand checked to be correct (by looking at the spectrum of each frame that is analysed).

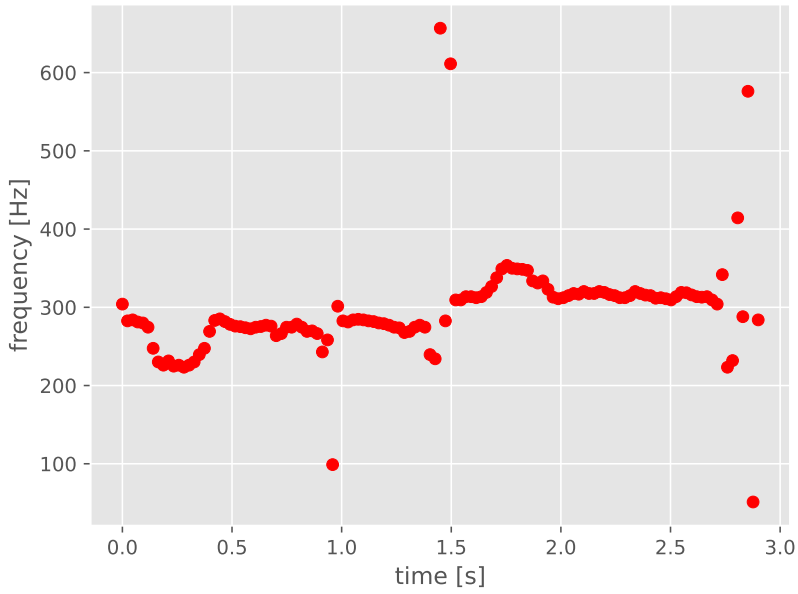


Figure 3.3: harmonic summation on vocal

3.2.2 multiple fundamental estimation

We applied harmonic summation to a sample with the notes played being C4 (261.63 Hz), E4 (329.63 Hz) and G4 (392.00 Hz). The pitch track is visible in Figure 3.4. As we can see, the algorithm correctly predicted the fundamental frequencies, apart from a few outliers.

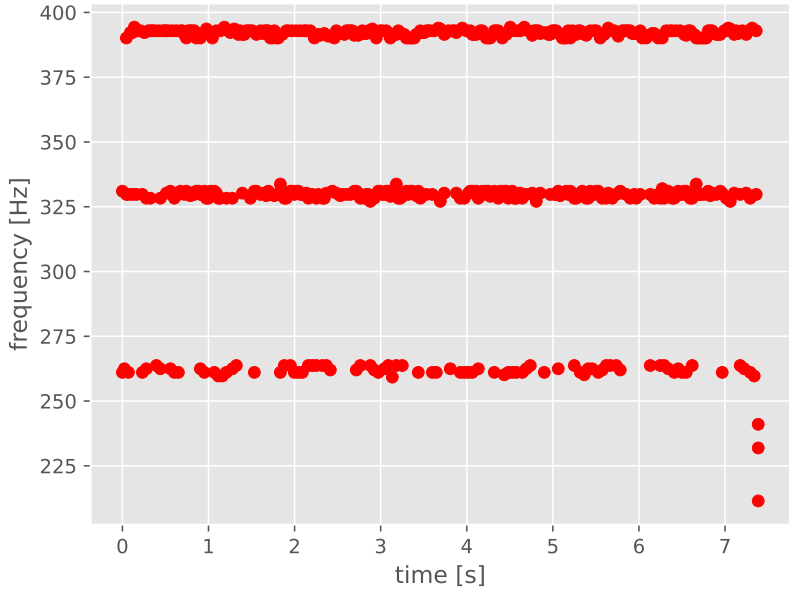


Figure 3.4: Harmonic summation applied to a sample with multiple fundamentals

Conclusion

The aims of this project were to find out which methods are state of the art, and to review some of the well known pitch detection algorithms and one advanced pitch detection algorithm. As is often the case in literature on audio, fundamental frequency is not defined explicitly. As a result of this project, the idea of fundamental frequency is formalised. The YIN algorithm Fundamental frequency

Furthermore, the methods that were discussed have been implemented in software and are available on Gitlab. The link to these implementations can be found in Appendix A.6.

During the project we found that the harmonic product spectrum (HPS) method can be generalised. The HPS can be thought of as a convolution of the power spectrum f and a weight function by $\sum_{m=0}^{\frac{N}{2}} f[m] \cdot g\left(\frac{m}{n}\right)$. It could be useful to think of this generalization rather than the HPS method, because choosing a different weight function than the one used in the HPS method allows you to take into account for the fact that harmonics may differ from integer multiples of the fundamental by some small error.

References

[1] V. Arora and L. Behera, "Multiple F0 Estimation and Source Clustering of Polyphonic Music Audio Using PLCA and HMRFs," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 2, pp. 278–287, Feb. 2015, ISSN: 2329-9290. DOI: 10.1109/TASLP.2014.2387388. [Online]. Available: http://ieeexplore.ieee.org/document/7001182/.

[2] H. Ba, N. Yang, and I. Demirkol, "BaNa: A hybrid approach for noise resilient pitch detection," *ieeexplore.ieee.org*, 2012. DOI: 10.13140/2.1.3921.6321. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6319706/?casa_token=BgfLGKChl88AAAAA:5TwBk-yXt21WleNM8jr6C3b3CNOVKs84ZLQdPY8M9QROiS1VKJpIwbZHgkmpQM-GNc60zyHt.

[3] F. J. Cañadas Quesada, N. Ruiz Reyes, P. Vera Candeas, J. J. Carabias, and S. Maldonado, "A Multiple-F0 Estimation Approach Based on Gaussian Spectral Modelling for Polyphonic Music Transcription," *Journal of New Music Research*, vol. 39, no. 1, pp. 93–107, Mar. 2010, ISSN: 09298215. DOI: 10.1080/09298211003695579. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/09298211003695579.

[4] C. Cao, M. Li, J. Liu, and Y. Yan, "MULTIPLE F0 ESTIMATION IN POLYPHONIC MUSIC (MIREX 2007)," *Citeseer*, no. Mirex, pp. 0–1, 2007. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.442.8783&rep=rep1&type=pdf.

[5] A. de Cheveigné and H. Kawahara, "YIN, a fundamental frequency estimator for speech and music.," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–30, Apr. 2002, ISSN: 0001-4966. DOI: 10.1121/1.1458024. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/12002874.

[6] H. Cuesta, B. McFee, and E. Gómez, "Multiple F0 Estimation in Vocal Ensembles using Convolutional Neural Networks," Sep. 2020. arXiv: 2009.04172. [Online]. Available: https://www.researchgate.net/publication/344180723_Multiple_F0_Estimation_in_Vocal_Ensembles_using_Convolutional_Neural_Networks%20http://arxiv.org/abs/2009.04172.

[7] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable Digital Signal Processing," Jan. 2020. arXiv: 2001.04643. [Online]. Available: http://arxiv.org/abs/2001.04643.

[8] Gareth Middleton, *Pitch Detection Algorithms - ECE 301 Projects Fall 2003 - OpenStax CNX*, 2003. [Online]. Available: https://cnx.org/contents/aY7_vV4-@5.8:i5AAkZCP@2/Pitch-Detection-Algorithms (visited on 02/21/2022).

[9] K. Han and D. L. Wang, "Neural network based pitch tracking in very noisy speech," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 22, no. 12, pp. 2158–2168, 2014, ISSN: 23299290. DOI: 10.1109/TASLP.2014.2363410. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6923432/?casa_token=uM2Y-xljw-cAAAAA:W-QON3IeSoc9OvOX37yWrRXA7JzSIBV1rxAg_bKsDRZ9fJuMPvdqYdTDhwedhJ8NQt2qtp6R.

[10] Z. Han and X. Wang, "A signal period detection algorithm based on morphological self-complementary Top-Hat transform and AMDF," *Information (Switzerland)*, vol. 10, no. 1, 2019, ISSN: 20782489. DOI: 10.3390/info10010024. [Online]. Available: https://www.mdpi.com/393830.

[11] F. Harris, "On the use of windows for harmonic analysis with the discrete Fourier transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, 1978, ISSN: 0018-9219. DOI: 10.1109/PROC.1978.10837. [Online]. Available: http://ieeexplore.ieee.org/document/1455106/.

[12] M. Huckvale, "Speech filing system," *Speech Filing System*, 2008. [Online]. Available: https://www.phon.ucl.ac.uk/resource/sfs/.

[13] D. Joho, M. Bennewitz, and S. Behnke, "Pitch Estimation using Models of Voiced Speech on Three Levels," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, vol. 4, IEEE, Apr. 2007, pp. IV–1077–IV–1080, ISBN: 1-4244-0727-3. DOI: 10.1109/ICASSP.2007.367260. [Online]. Available: https://ieeexplore.ieee.org/document/4218291/.

[14] H. Kawahara, "Straight-tempo: A universal tool to manipulate linguistic and para-linguistic speech information," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, IEEE, 1997, pp. 1620–1625, ISBN: 0-7803-4053-1. DOI: 10.1109/icsmc.1997.638234. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/638234/?casa_token=3RF8sZV_A_gAAAAA:Hs_AniVpj8096b5m7BzMr136lAzoOM6PKkHpDKM-Acj8Nh21u-qT7jkR8pJHsGANYjuBg5KL%20http://ieeexplore.ieee.org/document/638234/.

[15] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "Crepe: A Convolutional Representation for Pitch Estimation," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2018-April, IEEE, Apr. 2018, pp. 161–165, ISBN: 9781538646588. DOI: 10.1109/ICASSP.2018.8461329. arXiv: 1802.06182. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8461329/?casa_token=axgAEDlpWdEAAAAA:55EYyeUXvMWZe6hmTpZPef-TogUoqXNYmiWUwKxqdEnCAOBX71MCaN8FtsN6zJDRCRBmixUp%20https://ieeexplore.ieee.org/document/8461329/%20http://arxiv.org/abs/1802.06182.

[16] A. Klapuri, "Multiple fundamental frequency estimation by summing harmonic amplitudes," in *ISMIR 2006 - 7th International Conference on Music Information Retrieval*, 2006, pp. 216–221, ISBN: 9781550583496. [Online]. Available: https://archives.ismir.net/ismir2006/paper/000125.pdf.

[17] W. Kleijn et al., "A Robust Algorithm for Pitch Tracking," in *Speech Coding and Synthesis*, 1995, p. 518. [Online]. Available: https://www.ee.columbia.edu/$%5Csim$dpwe/papers/Talkin95-rapt.pdf.

[18] S. Kumar, "Performance Evaluation of Novel AMDF-Based Pitch Detection Scheme," *ETRI Journal*, vol. 38, no. 3, pp. 425–434, Jan. 2016, ISSN: 1225-6463. DOI: 10.4218/etrij.16.0115.0926. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.4218/etrij.16.0115.0926%20http://doi.wiley.com/10.4218/etrij.16.0115.0926.

[19] S. Kumar, S. K. Singh, and S. Bhattacharya, "Performance evaluation of a ACF-AMDF based pitch detection scheme in real-time," *International Journal of Speech Technology*, vol. 18, no. 4, pp. 521–527, Dec. 2015, ISSN: 1381-2416. DOI: 10.1007/s10772-015-9296-2. [Online]. Available: http://link.springer.com/10.1007/s10772-015-9296-2.

[20] P. Laura, V. Cortinez, L. Ercoli, and R. Rossi, "A simple method for the determination of the fundamental frequency of vibration of bones," *Medical Engineering Physics*, vol. 16, no. 5, pp. 422–424, Sep. 1994, ISSN: 13504533. DOI: 10.1016/1350-4533(90)90009-W. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/135045339090009W.

[21] Mathworks, *Pitch*, 2022. [Online]. Available: https://nl.mathworks.com/help/audio/ref/pitch.html?s_tid=mwa_osa_a.

[22] M. Mauch and S. Dixon, "PYIN: A fundamental frequency estimator using probabilistic threshold distributions," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, May 2014, pp. 659–663, ISBN: 978-1-4799-2893-4. DOI: 10.1109/ICASSP.2014.6853678. [Online]. Available: http://ieeexplore.ieee.org/document/6853678/.

[23] J. Mccullough, *Using Wavelets for Monophonic Pitch Detection*, 2005.

[24] P. McLeod and G. Wyvill, "A smarter way to find pitch," in *International Computer Music Conference, ICMC 2005*, 2005. [Online]. Available: http://www.cs.otago.ac.nz/tartini/papers/A_Smarter_Way_to_Find_Pitch.pdf.

[25] M. Nakano, K. Egashira, N. Ono, and S. Sagayama, "SEQUENTIAL ESTIMATION OF MULTIPLE FUNDAMENTAL FREQUENCY THROUGH HARMONIC-TEMPORAL CLUSTERING," *IN PROC OF THE 4TH MUSIC INFORMATION RETRIEVAL EVALUATION EXCHANGE (MIREX*, pp. 1–2, 2008. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.375.1509.

[26] P. Neubäcker, *Sound-object oriented analysis and note-object oriented processing of polyphonic sound recordings*, 2011. [Online]. Available: https://patents.google.com/patent/US8022286B2/en%20www.eurasip.org/ProceedingS/Eusipcot.

[27] A. M. Noll, "Cepstrum Pitch Determination," *The Journal of the Acoustical Society of America*, vol. 41, no. 2, pp. 293–309, Feb. 1967, ISSN: 0001-4966. DOI: 10.1121/1.1910339. [Online]. Available: https://asa.scitation.org/doi/abs/10.1121/1.1910339?casa_token=gZzc9AcbijUAAAAA:PAA5dEkLZyPWI4FACzkWd4czB8J9BXMwgnTHDpNucO1C7ww4Dn_cppo5pPMLxauy6l_bH8RCoRg%20http://asa.scitation.org/doi/10.1121/1.1910339.

[28] Scipy, *Hamming window*. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.windows.hamming.html.

[29] B. Secrest and G. Doddington, "An integrated pitch tracking algorithm for speech systems," in *ICASSP '83. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 8, Institute of Electrical and Electronics Engineers, 1983, pp. 1352–1355. DOI: 10.1109/ICASSP.1983.1172016. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1172016/?casa_token=jt7TUSqN39EAAAAA:muvxn0AY20rrvpXSzOpzKJaLRQ1MBvPxoQK7VgNm2f3hRLlMwPD9vGuV_17OnUiRbS--VBi1%20http://ieeexplore.ieee.org/document/1172016/.

[30] C. K. Un and S. C. Yang, "A Pitch Extraction Algorithm Based on LPC Inverse Filtering and AMDF," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 6, pp. 565–572, 1977, ISSN: 00963518. DOI: 10.1109/TASSP.1977.1163005. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1163005/?casa_token=8j5F1hb80wUAAAAA:IOxsAyEJXytJloB5_Iw5m5z91nTkWhQr8-Y9yXKe_zLIQXdpKQxZaOjQBMujf3HgsU1IC-jp.

[31] P. Verma and R. W. Schafer, "Frequency estimation from waveforms using multi-layered neural networks," in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 08-12-Sept, 2016, pp. 2165–2169. DOI: 10.21437/Interspeech.2016-679. [Online]. Available: https://www.isca-speech.org/archive_v0/Interspeech_2016/pdfs/0679.PDF.

[32] X. Wang, S. Takaki, and J. Yamagishi, "An RNN-based quantized f0 model with multi-tier feedback links for text-to-speech synthesis," in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2017-Augus, ISCA: ISCA, Aug. 2017, pp. 1059–1063. DOI: 10.21437/Interspeech.2017-246. [Online]. Available: https://www.isca-speech.org/archive_v0/Interspeech_2017/pdfs/0246.PDF%20https://www.isca-speech.org/archive/interspeech_2017/wang17e_interspeech.html.

[33] B. P. D. Weenink, "Praat: Doing Phonetics by Computer," *Ear Hearing*, vol. 32, no. 2, p. 266, Mar. 2011, ISSN: 0196-0202. DOI: 10.1097/AUD.0b013e31821473f7. [Online]. Available: https://www.fon.hum.uva.nl/praat/%20https://journals.lww.com/00003446-201103000-00012.

[34] S. Xu and H. Shimodaira, "Direct F0 Estimation with Neural-Network-Based Regression," in *Interspeech 2019*, vol. 2019-Septe, ISCA: ISCA, Sep. 2019, pp. 1995–1999. DOI: 10.21437/Interspeech.2019-3267. [Online]. Available: https://www.isca-speech.org/archive_v0/Interspeech_2019/pdfs/3267.pdf%20https://www.isca-speech.org/archive/interspeech_2019/xu19b_interspeech.html.

[35] J. X. Zhang, M. G. Christensen, S. H. Jensen, and M. Moonen, "Joint DOA and multi-pitch estimation based on subspace techniques," *EURASIP Journal on Advances in Signal Processing*, vol. 2012, no. 1, p. 1, Dec. 2012, ISSN: 1687-6180. DOI: 10.1186/1687-6180-2012-1. [Online]. Available: https://asp-eurasipjournals.springeropen.com/articles/10.1186/1687-6180-2012-1.

[36] Zhiyao Duan, B. Pardo, and Changshui Zhang, "Multiple Fundamental Frequency Estimation by Modeling Spectral Peaks and Non-Peak Regions," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 8, pp. 2121–2133, Nov. 2010, ISSN: 1558-7916. DOI: 10.1109/TASL.2010.2042119. [Online]. Available: http://ieeexplore.ieee.org/document/5404324/.

Appendix A

In this project, it was common to convert from and to different types of representations for frequency, such as delays and indices in the DFT. It's not always obvious how to convert from one notion to the other. Therefore, it is convenient to have these conversions listed. The purpose of this appendix is to list those conversions.

A.1 Frequency to fourier index

Let f_s be the samplerate, f be frequency in Hz, and let K be the length of the discrete Fourier transform. Let $[\cdot] : \mathbb{R} \longrightarrow \mathbb{Z}$ denote rounding to the nearest integer. Then the fourier index k is

$$k = \left[\frac{2f \cdot K}{f_s} \right].$$

Define the operator fftindex as $\text{fftindex}(f_s, K, f) = \left[\frac{2f \cdot K}{f_s} \right]$ where K and f_s may be left out as arguments if they can be derived from the context.

A.2 Fourier index to frequency

Let K be the length of the FFT. Then the frequency corresponding to the FFT index k is

$$f = \frac{k}{K} \frac{f_S}{2}.$$

A.3 Frequency to delay

Let $f \in \mathbb{R}$ be frequency. Then the corresponding delay in number of samples is $\tau = \frac{f_s}{f}$.

A.4 Delay to frequency

Let $\tau \in \mathbb{R}$ be delay (in number of samples). Then the corresponding frequency in Hz is $f = \frac{f_s}{\tau}$. Where f_s is the samplerate.
Let $\sigma \in \mathbb{R}$ be delay (in seconds). Then the corresponding frequency in Hz is $f = \sigma^{-1}$.

A.5 Time to sample index

Let x be a discrete time signal. Let $t \in \mathbb{R}$ be the time in seconds and f_s the sample rate. Let $[\cdot]$ denote the rounding operator. Then the sample index after t seconds passed is $[f_s t]$.

A.6 Implementation

The implementation of the algorithms described in this document can be found on Gitlab:
https://gitlab.tue.nl/jim-portegies/student-projects/damon-bernard.