An Introductory approach to Audio Signal Processing for Music Transcription

Diana Félix
diana.felix@ist.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

November 2018

Abstract

This thesis is focused on signal processing techniques applied to the transcription of an excerpt of a three voice J.S.Bach's fugue. The ideia is to study and apply some basic signal processing tools to an audio input signal and retrieve relevant music information that will allow to convert the signal into a symbolic representation. Music transcription is a complex task. So in order to narrow down the problem this challenge naturally imposes, it was chosen a piano solo recording of a fugue, which is a western composition technique based on a strong melodic independance. This introduces several simplifications like no need for instrument source detection or both harmonic and melodic complex analysis. The challenges faced in this task involve the computation of the spectrum of the input signal, rhythm events detection, background spectrum removal, fundamental frequencies detection and pitch detection.

Keywords: music transcription, signal processing, fugue, STFT, DFT

1. Introduction

Music transcription involves the translation of an acoustic signal into a symbolic representation, consisting on musical notes, the respective time events and the classification of the instruments used. In other words, it consists on listening to a piece of music and writing down the musical notation for that piece. It requires the retrieval of simultaneous information on several dimensions, like pitch, timing and instrument to resolve all the sound events so the goal is usually redefined as being either to notate as many of the constituent sounds as possible or to transcribe only some well-defined part of the music signal, for example, the dominant melody, the chords, bass progression or the musical key. This allows to perform automatic tasks like song identification, cover identification, genre/mood classification, score following, know as music information retrieval (MIR). When automatized, music transcription systems can assist musicians and composers to efficiently analyze compositions they only have in the form of acoustic recordings, and provide control on flexible mixing, editing, selective signal coding, sound synthesis or computer-aided orchestration. However, it also opens the door to the indept approach of concrete problems on signal processing with a wide range of applications. It provides a comprehensive set of descriptors to populate databases of music metadata, which can then be used for statistical analysis and content-based machine-learning approaches, that can also be applied to more popular fileds such as speach processing/transofrmation/recognition or physcoacoustics and perceptual research. Although monophonic music is considered a solved issue, poliphonic transcription, still poses a challenge nowadays. Much progress has been made in this area over the last decades. While newer and more sophisticated algorithms perform increasingly better, they also get considerably complex, computationally expensive and still quite depend on the type of input. More recent and efficient approaches employ machine-learning techniques and statistical models.

2. Background

2.1. The Discrete Fourier Transform

The Discrete Fourier Transform (DFT) of a signal $x(n) \in \mathbb{C}^N$, with $n \in \mathbb{Z}$ and $n = 0, 1, 2, ..., N-1$, is defined as:

$$X(w_k) = \sum_{n=0}^{N-1} x(t_n) e^{-jw_k t_n} = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N} \tag{1}$$

where k = 0, 1, 2, ..., N-1, $t_n = nT$ is the n-th sampling instant, $w_k = 2\pi k/Nf_s$ is the k-th frequency sample and $fs = 1/T$ is the sampling rate. The DFT provides a measure of the magnitude and phase of a complex sinusoid present in the signal x

on the frequency w_k and can be interpreted as the dot product of the signals x and s_k, which determines the projection coefficients of x on the complex sinusoid $\cos(w_k t_n) + j\sin(w_k t_n)$. So from the definition of dot product between two vectors x e y

$$\langle x, y\rangle = \sum_{n=0}^{N-1} x(n)\overline{y(n)} \tag{2}$$

the DFT defined in (1) can also be written as

$$X(w_k) = \langle x, s_k\rangle = \sum_{n=0}^{N-1} x(n)\overline{s_k(n)} \tag{3}$$

where s_k is the transform's kernel. With the sum of the projections of x on N-vectors s_k, it is possible to recover the original signal when $s_k{}_{k=0}^{N-1}$ is an orthogonal base of $\mathbb{C}^N$, which means that the vectors on the basis of this space are linearly independent. This is true for

$$f_k = k\frac{f_s}{N}, \qquad k = 0, 1, 2, ..., N-1 \tag{4}$$

and the DFT always has an inverse because the number of samples of the signal is always finite. The Inverse Discrete Fourier Transform (IDFT) is computed by the sum of the projections

$$x(n) = \sum_{k=0}^{N-1} \frac{X(k)}{N} s_k(n) \tag{5}$$

where $n = 0, 1, 2, ..., N-1$ and $X(k)/N$ corresponds to the projection coefficients of x in s_k. The IDFT can then be defined as

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(w_k)e^{j2\pi nk/N} \tag{6}$$

where $n = 0, 1, 2, ..., N-1$. $X(w_k)$ is then called the spectrum of x and can be interpreted as the sum of the projections of x on the orthogonal basis $s_k{}_{k=0}^{N-1}$ so its IDFT is the recovery of the original signal as a superposition of its projections on the N complex sinusoids.

2.2. Windowing

When the period of x does not correspond exactly to the roots of w_k (4) - which is the case in most of the analysis of acoustic real signals - the energy of the sinusoid is spread along all the frequency bins. This effect, designated by *spectral leakage* is equivalent to abruptly truncating the sinusoid on its edges. This effect can be reduced by dividing the signal into smaller sets of samples and applying a window function which will soften the signal decay to 0 in both edges of the window. Considering a sinusoidal signal $x(n)$

$$\begin{aligned} x(n) &= A_0\cos(2\pi k_0 n/N) \\ &= \frac{A_0}{2}e^{j2\pi k_0 n/N} + \frac{A_0}{2}e^{-j2\pi k_0 n/N} \end{aligned} \tag{7}$$

then the DFT of the windowed signal becomes

$$\begin{aligned} X(k) &= \sum_{n=-N/2}^{N/2-1} w(n)x(n)e^{-j2\pi n/k} \\ &= \frac{A_0}{2}W(k-k_0) + \frac{A_0}{2}W(k+k_0) \end{aligned} \tag{8}$$

which corresponds to the DFT of the window $w(n)$ shifted to the frequencies of $x(n)$. This is a property of the convolution theorem that states that multiplying two signals in the frequency domain corresponds to its convolution in the time domain.

The windowing technique used in the computation of the DFT determines the trade-off between temporal and frequency resolution which affects the smoothness of the spectrum and the detectability of the frequency peaks. The choice of the window function depends heavily on the size of the window's main lobe - which is a characteristic of the window refers to number of samples in the main lobe - and its relation to the side lobes amplitude. Windows with a narrower main lobe K have a better frequency resolution Δf (for a same window size M)

$$\Delta f = K\frac{f_s}{M} \tag{9}$$

but tend to have higher side lobes which is a source of *cross-talk* between the channels of the FFT. The main features of some of the most commonly used windows for audio signal processing are synthesized in Table 1.

2.3. Fast Fourier Transform

The DFT can be efficiently computed with the *Fast Fourier Transform* (FFT) algorithm which is computationally more efficient when N is a power of 2. For this reason it's frequent to add zeros at the end of the digital signal - zero-padd it - prior to the computation of the DFT to optimize the lenght of the FFT, which corresponds to a signal interpolation in the frequency domain. For processing real time varying acoustic signals is also necessary to determine its frequency distribution over the time. One way to achieve this is by applying the input signal a sliding window function with $N > M$and computing its FFT. This approach is refered as *Short-time Fourier Transform* (STFT) and can be formally defined by (10)

	Rectangular	*Hann*	*Hamming*	*Blackman*	*Blackman-Harris*
Main lobe width	$2\Omega_M$ 2 bins	$4\Omega_M$ 4 bins	$4\Omega_M$ 4 bins	$6\Omega_M$ 6 bins	$8\Omega_M$ 8 bins
Side lobe attenuation	-13 dB	-31 dB	-43 dB	-58 dB	-92 dB
Side lobe roll-off/octave	-6 dB	-18 dB	-6 dB	-18 dB	-
Degrees of freedom	1	2	3	4	5
Side lobe attenuation	100%	50%	50%	25%	20%

Table 1: Comparison of the windows Rectangular, Hamming, Blackman and Blackman-Harris.

$$X_l(k) = \sum_{n=-N/2}^{N/2-1} x(n+lH)w(n)e^{-j2\pi kn/N} \qquad (10)$$

where $l = 0, 1, 2...$, $w(n)$ is the window function, l is the frame number and H is the hop-size, which correponds to the number of sliding samples of the window between consecutive frames.

3. Implementation

For transcribing an excerpt of a piano recording, the signal processing tasks involved computing the STFT, extract the rhythmic information prior to any filtering, then remove the background noise from the spectrum, compute the fundamental frequencies spectrogram and finally detect the melodic contours from the pitch diagram.

3.1. STFT computation

For the computation of the spectrogram were considered WAV mono-channel signals sampled at 44100 Hz. The samples are converted into a floating array and normalized so that $x \in [-1, 1]$. In was used a Hamming window with M = $2^{13} = 8192$, without zero-padding to allow higher frequency resolution, and an overlap-factor of 25%, which is less than the 50% limit for this window but allows the increase of time resolution. Despite the roll-off of the side-lobes not overcoming -43 dB, in practise the Hamming window had a better performance that the Blackman-Harris windows when it comes to the compromise between time and frequency resolution and computational cost. The size of the window was dimensioned so that the minimum frequency resolution of the spectrogram Δf was around 21.5 Hz (A0), which corresponds to the lowest frequency of the piano.

3.2. Rhythmic information extraction

This task requires the presence of all spectral information, including the higher harmonics region, and for this reason it has to be performed prior to any spectral filtering. Rhythmic information extraction requires a vertical analysis of the spectogram and aims to determine the frames when a new musical note is played. This is achieved by computing a simple onset detection funcion (ODF), which determines the difference of energy between adjacent frames l on a specific frequency band [15]

$$ODF(l) = E(l) - E(l-1), \qquad l \geq 1 \qquad (11)$$

where

$$E(l) = \sum_{l=0}^{N-1} x(l)^2 \qquad (12)$$

Considering we are only interested in the onsets - when the energy increases we consider a half wave rectified ODF given by

$$ODF'(l) \begin{cases} ODF(l), & \text{if } ODF(l) > 0 \\ 0, & \text{otherwise} \end{cases} \qquad (13)$$

Three frequency bands were chosen for this analysis: the first one between 20 Hz and 200 Hz, the second between 201 Hz and 1000 Hz and the third ranging from 1001 Hz to 4000 Hz. The frequencies used to define the bands were chosed in order to comprise the frequencies produced by a piano keyboard, ranging from 27.5 Hz to 4186.01 Hz. The lower band allows the detection of lower notes, such as the bass, while the higher band provides information on notes that somehow have lower energy on the lower band - such as the missing fundamentals - but do have higher energy on the upper partials. We observe that the lower band in particular adds serveral false onsets, so the best results were obained by considering only the two highest frequency bands. We started by finding the common onsets between the two higher bands, reducing the inital 140 values to 37, and then added the values above averge on both bands (44). This increased the onsets to 52, which is very close to the real value of 50. It was assumed a lowest time resolution of one frame. We conclude we still have missing onsets (on frame 311, for example), and false detections as well, as in frame 48. Time onset, however, is a fundamental part of the audio transcription process followed here. Without accurate information on the rhythmic events, the calculation of the averaged f_0 frequencies in the next section will be compromised. For this reason, the onsets were ultimately validated by inspection.

3.3. Fundametal frequencies detection

To estimate the background spectrum it was used a simplified method similar to the spectral whitening referred in [11]. This filtering consisted on subtracting the average of the spectrum μ_k in an octave interval with step 100 bins, divide it by the standard deviation σ_k and filter out the negative amplitudes in the spectrum.

$$Y_k = \begin{cases} \frac{X_k - \mu_k}{\sigma_k}, & \text{if } X_k - \mu_k > 0 \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

Once we have the rythmic information, the second step is to determine which notes are played. In order to do so, we need to find the fundamental frequencies for each melodic line from the spectrogram. The piano however is an inharmonic instrument. This means its harmonics are not multiples of the fundamental frequency and their spacing increases with the order of the harmonic instead. One of the models that describes the behaviour of the piano partials is [4]

$$f_n = n f_0 \sqrt{\frac{1 + Bn^2}{1 + B}} \tag{15}$$

where f_n is the frequency of the n-th harmonic, f_0 is the fundamental frequency and B is the inharmonicity coefficient. B is a characteristic parameter of the instrument. It is experimentally determined and generally takes values between 10^{-3} and 1^{-6} [4]. The approach for f_0 detection followed here is based on [13]. The ideia is to determine a set of f_0 candidates selected from the peaks with magnitude threshold of -60 dB. Then the f_0 candidates are validated with the partials patterns present in the spectrogram considering the inharmonicity effect. So the current partial candidate f_{n+1} is mutiplied by a factor $f_n(n + 1/n)$ and the next partial is set as the first match within a positive varying margin Δ_{n+1}/Δ_n:

$$f_{n+1} = f_n \frac{n+1}{n} \frac{\Delta_{n+1}}{\Delta_n} \tag{16}$$

where

$$\frac{\Delta_{n+1}}{\Delta_n} = \sqrt{\frac{1 + B_{max}(n+1)^2}{1 + B_{max} n^2}}$$

corresponds to the search interval for harmonic f_{n+1}.

This margin is maximized by using an optimized inharmonicity coefficient, which experimentally was set to $B = 3$x10^{-3}, considering at least 15 partials. All candidates missing the second or third harmonics (octave or fifth) are discarted as well as the frequencies matching less than 4 partials.

Considering each musical note is not expected to change within a *tempo* frame, the frequency bins are considered valid candidates only if the number of non null points within the *tempo* frame is larger than the number of null points. After this step, the data is converted from bins to the pitches of the tempered scale resulting the pitch diagram of Figure 1.

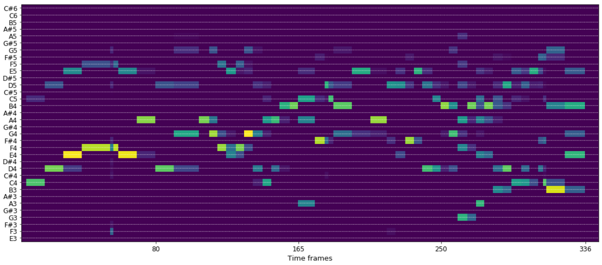

Figure 1: Averaged pitch magnitude spectrogram in linear units.

3.4. Melodic contour detection

The fundamental frequencies filtering allowed to significantly reduce the number of potencial pitch candidates from the initial spectrum. However, there are still pitches that may correspond to higher partials or sparse detections, such as echos produced by previously played notes. In order to determine which pitches are more likely to correspond to real fundamentals and which ones may eventually be discarted, the pitches along each *tempo* frame are stored in two buffers P^+ and P^-, if their magnitudes are, respectively, above or below the frame's average. One step that considerably reduces the number of potencial partials and improves the melodic contours detection is considering the octave information. From the spectrum perspective, both pitches - the fundamental and the octave - can be valid played music notes, differing only on the sum of magnitudes. So the octaves of the pitches in P^+ are flagged and added to P^- buffer for further processing. Taking advantage on the fugues horizontal independance on the melodic lines, we can expect each voice contour to span vertically along a relatively limited neighbourhood of pitches, which allows us to predict the most likely next pitch of the voice path. The ideia is to select the first unprocessed element of P^+, define a search interval spanning from a perfect forth, both above and below the reference pitch, and find the closest matching pitch on the next tempo frame. This process is repeated for every frame until a path is found and, once finished, the processed pitches are removed from P^+ and the next P^+ candidate is selected for path validation. The P^- buffer stores the secondary pitches that are used for filling up the path in case there is no P^+ candidate available. In case there are no pitches available at all - when the fundamental is missing, for example - it's allowed a one frame jump, assuming the current pitch as the next candidate.

4. Results

After computing and adding up all the voice contours, we obtain the diagram of Figure 2.

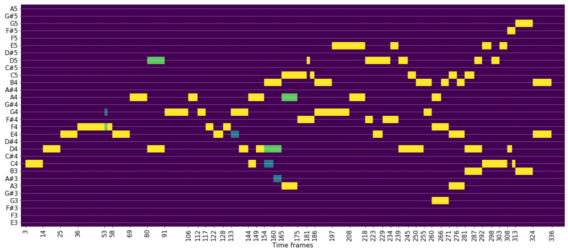

Figure 2: Melodic transcription of the first four bars of the Fugue I from J.S.Bach. Peak threshold of -60 dB, a reference interval of a perfect forth and a minimum voice lenght limit of 5 notes.

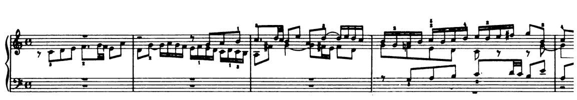

Figure 3: Score of the first four bars of the Fugue I in C Major from the *Well Tempered Clavier* BWV 846 (Book I) from J.S.Bach.

The best results were achieved with a peak threshold of -60 dB, a reference interval of a perfect fourth and a minimum voice lenght of five notes. When comparing the resulting transcription diagram of Figure 2 with the original music score (Figure 3), we conclude the algorithm performs quite satisfactorily. The approach followed here, based on the melodical structure of the fugue, revealed to be enough for the transcription task proposed. The transcription errors corresponding to missing notes are mostly due to missing fundamentals (frame 160 for example) which were never detected in the first place by the algorithm for f_0 detection because of its very low and irregular magnitudes in the original spectrogram. These results for short duration, low frequency pitches is an expected consequence of the time *vs.* frequency trade-off when dimensioning the spectrogram: the lower the frequency, the lower will be its resolution in a log-spectrogram, agravated by the fact that the fast rhythmic behavior worsens the accuracy of the onsets.

Another parameter that significantly affects the transcription results is the search margin size. If the interval is set to low, we may miss pitches along the path and if set too high we may end up loosing track of the real voice contour, by ultimately following only the high amplitude pitches P^+. This results are very senstive to changes in the margin intervals. For example, if we increase the intervals to a perfect fifth we will miss out the obvious G5 on frame 313 for B4, simply because this pitch has higher magnitude caused by the harmonics of B3 played in the previous *tempo*. As a consequence of increasing the margin size, we may end up selecting wrong pitches as well. In frame 40, for example, the played note corresponds to a D4 but the voice contour algorithm detects only its second partial which is inside the perfect fourth interval. So the algorithm ends up detecting two distinct voice lines at that frame. One way to fix this, could be to perform an octave clean up at the end of the voices computation for the frames with more than the expected pitches.

5. Conclusions

The algorithm developed did satisfactorily transcribe the excerpt of the proposed fugue recording using basic techniques of audio signal processing. The approach followed here is based on the melodical structure of the fugue, which revealed to be enough for the transcription task proposed. The solution implemented has however a quite limited scope. Firstly, the methodology followed is a simplification of the music transcription problem and is based on several previous assumptions on the data to be transcribed, so it's not generalized to other composition techniques. This method performs well for Bach's fugues, based on a voice independance and continuity assumption.

This solution also still depends a lot on static parameterization. The correct dimensioning of the reference inharmonicity parameter or the threshold for peak detection are critical for the correct detection of the fundamental frequencies and the performance of the algorithm. The onset detection, for example, is based on pre-defined frequency bands that should be dimensioned according to the type of instruments and frequency span considered for the audio signal. One critial step on the transcription in the computation of the spetrogram. One of the main challenges when dimensioning the spectrogram is the time vs. frequency trade-off. The STFT performance depends heavily on the choice of the window type and size. If a signal has a wide range of frequencies, specially when it concerns the lower frequencies, it requires a very large N to satisfy the frequency resolution requirements, which can be computationlly very inefficient and definitely compromise the time resolution. This could be improved by computing a log-spectrogram with a variable size sliding window, in order to improve the resolution on the lower pitches. Instead of implementing a pure constant-Q transform, there are other similar algorithms than allow to improve the frequency resolution using the more computationally efficient FFT function. The time-frequency trade-off also may cause that fundamental frequencies are not detected at all. So this would require some predictive analysis either on the fundamental frequencies detection step or in the melody tracking. The fundamentals can be detected by matching the upper partials patterns to a defined fundamental pitch, while the simultaneous analysis of melody and harmony would allow to predict the notes with higher probability in the harmonic context of the time frame.

So indeed music transcription is quite a challenging task, specially when dealing with requirements such as automatization, where no human interaction in the middle steps is expected, and becomes increasingly more complex with the variety of music and instruments that are to be transcribed. In this sense, the probabilistic methods and machine learning approaches may perform better on solving such problems.

References

[1] R. P. ad W. J. M. Levelt. Tonal consonance and critical bandwidth. 1965. Institute for Perception RVO-TNO, Soesterberg, Netherlands.

[2] D. Benson. Music: A mathematical offering. Web version, http://www.maths.abdn.ac.uk/bensondj/, December 2008.

[3] T. Cheng, S. Dixon, and M. Mauch. Modelling the decay of piano sounds. Centre for Digital Music, Queen Mary University of London, London, United Kingdom, February 2015.

[4] H. Fletcher. Normal Vibration Frequencies of a Stiff Piano String . *The Journal of the Accoustical Society of America*, 36, 1964. http://dx.doi.org/10.1121/1.1918933.

[5] E. Gmez. Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing*, 18(3):294304, 2006.

[6] D. J. Grout and C. V. Palisca. *Histria da Msica Ocidental.* Gradiva, 2007.

[7] C. Hausner. Design and evaluation of a simple chord detection algorithm. Master's thesis, University of Passau, Faculty of Computer Science and Mathematics, 2014.

[8] J. III. Mathematics of the Discrete Fourier Transform with audio applications, second edition. https://ccrma.stanford.edu/ jos/mdft/. Center for Computer Research in Music and Acoustics (CCRMA), Stanford University, 2007, ISBN 978-0-9745607-4-8.

[9] A. Klapuri. Automatic transcription of music, Tampere University of Technology, Department of Information Technology, 1997.

[10] A. Klapuri and M. Davy. *Signal Processing Methods for Music Transcription.* Springer, 2000.

[11] M. Matthias and D. Simon. Approximate note transcription for the improved identification of difficult chords. 2009. Queen Mary University of London, Centre for Digital Music.

[12] M. Matthias and T.Cheng. Modelling the decay of piano sounds. 2015. Queen Mary University of London, Centre for Digital Music.

[13] A. Pertusa and J. M. Iesta. Efficient methods for joint estimation of multiple fundamental frequencies in music signals. *EURASIP Journal on Advances in Signal Processing*, 2012. doi:10.1186/1687-6180-2012-27.

[14] J. P. X. Serra. Designing efficient architectures for modeling temporal features with convolutional neural networks. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017. doi:10.1109/ICASSP.2017.7952601.

[15] X. Serra and J. O. S. III. SMS tools - sound analysis/synthesis tools for music applications. https://github.com/MTG/sms-tools. Universitat Pompeu Fabra Barcelona, Stanford University, 2017.

[16] C. Yeh. *Multiple Fundamental Frequency Estimation Of Polyphonic Recordings.* PhD thesis, cole Doctorale Edite Universit Paris Vi - Pierre Et Marie Curie, 2008.