Insert here your thesis' task.

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Master's thesis

Multi-instrument music transcription

Bc. Yevhen Kuzmovych

Department of Applied Mathematics
Supervisor: Ing. Marek Šmíd, Ph.D.

January 4, 2020

Acknowledgements

THANKS (remove entirely in case you do not with to thank anyone)

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on January 4, 2020 ....

Citation of this thesis

Kuzmovych, Yevhen. *Multi-instrument music transcription.* Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2020.

Abstrakt

V několika větách shrňte obsah a přínos této práce v českém jazyce.

Klíčová slova Replace with comma-separated list of keywords in Czech.

Abstract

Summarize the contents and contribution of your work in a few sentences in English language.

Keywords Replace with comma-separated list of keywords in English.

Contents

List of Figures

List of Tables

Introduction

Problem definition

The difficulty of multiple-F0 estimation lies in the fact that sound sources often overlap in time as well as in frequency. The extracted information is partly ambiguous. Above all, when musical notes are played in harmonic relations, the partials of higher notes may overlap completely with those of lower notes. Besides, spectral characteristics of musical instrument sounds are diverse, which increases the ambiguity in the estimation of partial amplitudes of sound sources. The resulting complexity causes not only octave ambiguity but also the ambiguity in the estimation of the number of sources.

Tuning

Time signature

Chapter 1

State-of-the-art

This chapter discusses existing state-of-the-art solutions for music transcription.

Sound transcription into sheet music is a combination of several techniques that include but not limited to source instruments separation, pitch/note detection, event detection, etc.

Source separation and sound transcription to sheet music are fairly independent processes so their description and approaches may come from different sources and different projects. Therefore, implementation will also be separated.

1.1 Source separation

There were many successful attempts for music score source separation[1, 2, 3]. Performance of such projects are commonly measured according to *Source Separation campaign (SiSeC)*[4] on the standard *musdb18*[5] and *DSD100*[6] datasets.

Latest and most successful project in this field is *Spleeter*[1]. It is a project of Deezer[1]. It takes similar approaches to previous solutions by University of London and Spotify[2]. Spleeter's pre-trained models will be used in the module responsible for music source separation described in detail in the chapters 2 and 3.

Following approaches are described in [1, 2, 3].

1.1.1 Spleeters approach

The pre-trained models are U-nets[2] and follow similar specifications as in *Singing voice separation: a study on training data*[3]. The U-net is a encoder/decoder Convolutional Neural Network (CNN) architecture with skip

[1] Deezer is a French online music streaming service (deezer.com).

connections[1]. Architecture used in this approach showed a state-of-the-art results on DSD100 dataset[2] and in the last SiSeC[6].

1.1.2 U-net architecture

The U-Net shares the same architecture (shown on fig. 1.1) as a convolutional autoencoder with extra skip-connections that bring back detailed information lost during the encoding stage to the decoding stage. It has five strided[2] 2D convolution layers in the encoder and five strided 2D deconvolution layers in the decoder.

The goal of the neural network architecture is to predict the vocal and instrumental components of its input indirectly: the output of the final decoder layer is a soft mask for each source that is multiplied element-wise with the mixed spectrogram to obtain the final estimate.

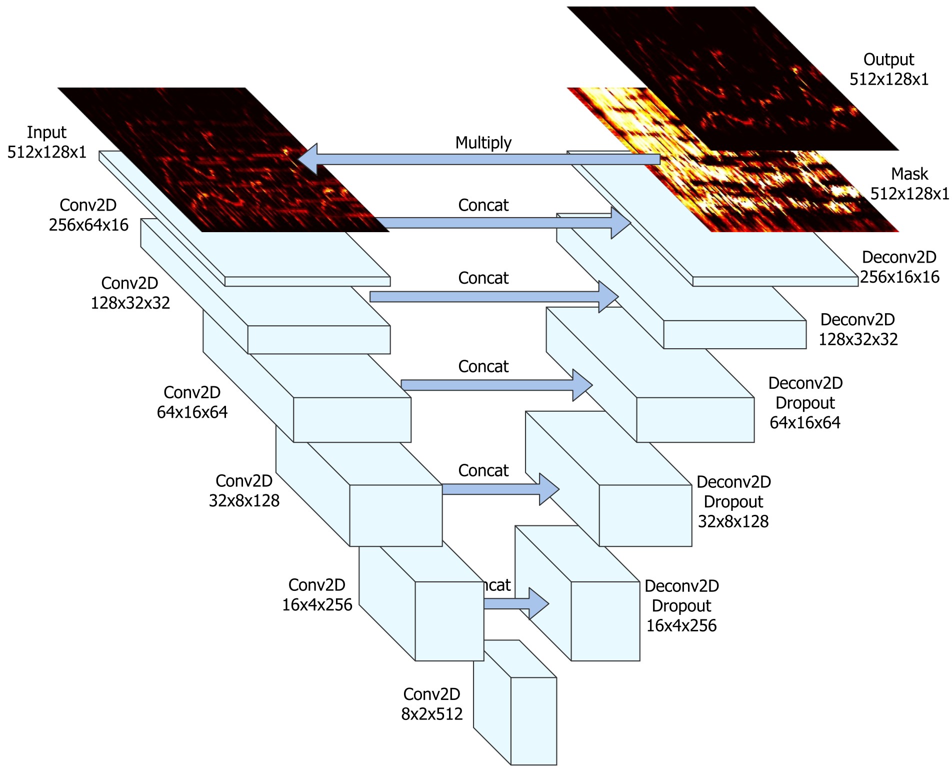


Figure 1.1: Network Architecture[2]

[2]Transposed convolutions – also called *fractionally strided convolutions* – work by swapping the forward and backward passes of a convolution. One way to put it is to note that the kernel defines a convolution, but whether it's a direct convolution or a transposed convolution is determined by how the forward and backward passes are computed.[7]

1.1.3 Data and training

Spleeter's training dataset is an internal Deezer's dataset and is not shared (for copyright reasons).

Another project with similar approach, as explained in the dedicated article[3], uses two datasets during training of the models: *MUSDB*[5] and *Bean*(private dataset).

MUSDB is the largest and most up-to-date public dataset for source separation[5]. It contains 150 songs of western music genres primarily pop/rock, some hip-hop, rap and metal songs. And each song consists of four audio tracks: drums, bass, vocal and other. Original mix (and input of the model) is produced by summing tracks of four sources (expected outputs) together.

1.2 Multi-pitch Detection

There were several projects utilizing different approaches to a problem of Automatic music transcription (AMT). Following section discuss these approaches and projects that used them.

The most important part of transcription of sound into sheet music is pitch (and subsequently note) detection. The core problem of polyphonic music transcription is multi-pitch detection.

In his Ph.D. research *Multiple fundamental frequency estimation of polyphonic recordings*[8], Chunghsin Yeh classifies multi-pitch detection systems according to their estimation type into two categories: joint and iterative. The *iterative* approach extracts the most eminent frequency per each iteration, until no other pitch can be estimated and extracted. Commonly, iterative estimators generate errors on each iteration but are much cheaper in terms of computation costs.

On the other side, the *joint* estimation models evaluate combinations of pitches at once, which leads to increase in accuracy but also in computation costs. Most of the latest approaches and state-of-the-art solutions fall into joint category. Solution in this thesis also follows this category and will be discussed in detail in chapter 2.

1.2.1 Feature-based multi-pitch detection

Most multi-pitch estimation and note tracking approaches exploit methods that come from signal processing. There is no specific model (Machine learning (ML) or other), and notes are detected using audio features that come from the input time-frequency representation (spectrogram) either in an iterative or joint way. Usually, multi-pitch estimation uses a *pitch candidate set score function* or a *pitch salience function*.

A *salience function* is a function that provides an estimation of the predominance of different frequencies in an audio signal at every time frame[9].

A *pitch candidate set score function* is function designed to evaluate the plausibility of the combination of the hypothetical sources[8].

These feature-based techniques have produced the best results in the Music Information Retrieval Evaluation eXchange (MIREX) multi-pitch and note tracking evaluations. The work by Chunghsin Yeh[8] was the best performing method in the MIREX multi-pitch and note tracking tasks. Yeh proposed a joint pitch estimation algorithm based on a pitch candidate set score function. Having a set of pitch candidates, the overlapping partials are detected and smoothed according to the spectral smoothness principle, which states that the spectral envelope[3] of a musical sound tends to be slowly changing as a function of frequency. The score function for the pitch candidate set consists of four features: harmonicity, mean bandwidth, spectral centroid, and “synchronicity” (synchrony). A polyphony inference mechanism based on the score function increase selects the optimal pitch candidate set[8].

In the following year, the best performing method for the MIREX multi-pitch estimation and note tracking tasks, Karin Dressler described in her work *Multiple fundamental frequency extraction for MIREX*[11]. A multiresolution Fast Fourier transform (FFT) (see section 2.4.2 on multiresolution FFT) analysis was used as an input time/frequency representation, where the magnitude for each spectral bin was multiplied with the bin’s instantaneous frequency. Pitch estimation is made by identifying spectral peaks and performing pairwise analysis on them, resulting on ranked peaks according to harmonicity, smoothness, the appearance of intermediate peaks, and harmonic number. Finally, the system tracks tones over time using an adaptive magnitude and a harmonic magnitude threshold.

Other notable feature-based AMT solution was introduced in the work by Pertusa and Inesta *Multiple fundamental frequency estimation using Gaussian smoothness and short context*[12]. They proposed a computationally inexpensive method for multi-pitch detection which computes a pitch salience function and evaluates combinations of pitch candidates using a measure of distance between a Harmonic partial sequence (HPS) and a smoothed HPS. Another approach for feature-based AMT was proposed in *Hybrid genetic algorithm based on gene fragment competition for polyphonic music transcription*[13], which uses genetic algorithms for estimating a transcription by mutating the solution until it matches a similarity criterion between the original signal and the synthesized transcribed signal.

More recently, Peter Grosche et al. proposed[14] an AMT method based on a mid-level representation derived from a multiresolution FFT combined with an instantaneous frequency estimation. His system also combines event (specifically start of the note) detection and tuning estimation for computing predictions. Finally, Juhan Nam et al. proposed[15] a classification-

[3] *Spectral envelope* of the sound determines the particular vowel sound produced, and is, in general, one of the important acoustic features that determine its perceived timbre[10].

based approach for piano transcription using features learned from deep belief networks[16] for computing a mid-level time-pitch representation.

1.2.2 Statistical model-based multi-pitch detection

Many approaches in the literature formulate the multi-pitch estimation problem within a statistical framework. As Valentin Emiya et al. explains in their article *Multipitch Estimation of Piano Sounds Using a New Probabilistic Spectral Smoothness Principle*[17]: given an observed frame $\boldsymbol{x}$ and a set $\boldsymbol{C}$ of all possible fundamental frequency combinations, the frame-based multi-pitch estimation problem can then be viewed as a Maximum a posteriori (MAP) estimation problem:

$$\hat{C}_{MAP} = \arg\max_{C \in \boldsymbol{C}} P(C|\boldsymbol{x}) = \arg\max_{C \in \boldsymbol{C}} \frac{P(\boldsymbol{x}|C)P(C)}{P(\boldsymbol{x})}$$

where $C = \{F_0^1, \ldots, F_0^N\}$ is a set of possible frequencies (considering tuning of an instrument), $\boldsymbol{C}$ is the set of all possible F_0 combinations, and $\boldsymbol{x}$ is the observed audio signal within a single analysis frame.

An example of MAP estimation-based transcription is the *PreFEst* system introduced by Masataka Goto in his article *A real-time music-scene-description system: predominant-F0 estimation for detecting melody and bass lines in real-world audio signals*[18], where each harmonic is modelled by a Gaussian centered at its position on the log-frequency axis. Expectation-maximisation (EM) algorithm is used to estimate MAP value. An extension of this method was proposed by Kameoka et al. in *A Multipitch Analyzer Based on Harmonic Temporal Structured Clustering*[19], which jointly estimates multiple possible frequencies, moments of start and end of the note, and dynamics. Partials are modelled using Gaussians placed at the positions of partials in the log-frequency domain and the synchronous evolution of partials belonging to the same source is modelled by Gaussian mixtures.

More recently, Peeling and Godsill, in their article *Multiple pitch estimation using non-homogeneous Poisson processes*[20], also proposed a likelihood function for multiple-pitch estimation where for a given time frame, the occurrence of peaks in the frequency domain is assumed to follow an inhomogeneous Poisson process. Also, Koretz and Tabrikian in *Maximum a posteriori probability multiple-pitch tracking using the harmonic model*[21], proposed an iterative method for multi-pitch estimation, which combines MAP and ML criteria. The predominant source is expressed using a harmonic model while the remaining harmonic signals are modelled as Gaussian interference sources[21].

1.2.3 Spectrogram factorisation-based multi-pitch detection

The majority of recent multi-pitch detection papers utilise and expand spectrogram factorisation techniques. Non-negative matrix factorisation (NMF) is a technique first introduced in their paper by Paris Smaragdis et al. [22] as a tool for music transcription. In its simplest form, the NMF model decomposes an input spectrogram $\boldsymbol{X} \in \mathbb{R}_{+}^{K \times N}$ with K frequency bins and N frames:

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

where $R \ll K, N$; $\boldsymbol{W} \in \mathbb{R}_{+}^{K \times R}$ contains the spectral bases for each of the R pitch components; and $\boldsymbol{H} \in \mathbb{R}_{+}^{R \times N}$ is the pitch activity matrix across time.

In his paper *Realtime multiple pitch observation using sparse non-negative constraints*, Cont Arshia applies NMF to AMT problem. Sparseness constraints were added into the NMF update rules, in order to find meaningful transcriptions using a minimum number of non-zero elements in $\boldsymbol{H}$. Emmanuel Vincent et al. in their article *Adaptive harmonic spectral decomposition for multiple pitch estimation*[23] incorporated harmonicity constraints in the NMF model, resulting in two algorithms: harmonic and inharmonic NMF. The inharmonic version of the algorithm is also able to support deviations from perfect harmonicity and standard equal temperament tuning. Also, Nancy Bertin et al. in their article[24] proposed a Bayesian framework for NMF, which considers each pitch as a model of Gaussian components in harmonic positions.

More recently, Ochiai et al. in his paper *Explicit beat structure modeling for non-negative matrix factorization-based multipitch analysis*[25] proposed an algorithm for multi-pitch detection and beat structure analysis. The NMF objective function is constrained using information from the rhythmic structure of the recording. It helped to improve transcription accuracy in highly repetitive recordings.

This thesis approaches AMT problem in similar fashion to spectrogram factorisation and feature-based methods. Detailed description of used methods is in chapter 2.

1.3 Note Tracking

Typically AMT algorithms compute a time-pitch representation which needs to be further processed in order to detect note events with a discrete pitch value, a time of start and end of the note. This process is called *note tracking*. Most spectrogram factorisation-based methods estimate the binary piano-roll representation from the pitch activation matrix using simple thresholding (i.e. in *Explicit beat structure modeling for non-negative matrix factorization-based multipitch analysis*[26] by Graham Grindlay and in *Adaptive harmonic spectral decomposition for multiple pitch estimation*[23] by Emmanuel Vincent). This approach will be used in the implementation of the thesis. Also, one simple and fast optimisation for note tracking is minimum duration pruning, which is applied after thresholding (idea comes from paper by Arnaud Dessein et al. *Real-time polyphonic music transcription with non-negative matrix factorization and beta-divergence*[27]). Primary idea is that output notes that have a duration smaller than a predefined threshold are removed from the final score. Similar method was also used by Juan Pablo Bello et al. in their paper *Automatic piano transcription using frequency and time-domain information*[28], where more complex rules for note tracking were used, addressing cases such as where a small gap exists between two note events. This method will also be applied as it is easy to implement.

For threshold based method, there are several issues that may appear for different instruments, primarily related to how notes are used and written for them in practice. For instance, for percussion instruments, note decay is exponential and physical duration of the note is irrelevant as it is not controlled(for most percussion instruments) by a musician. so notes may appear short and require pauses(rests) after them, even though they (rests) would not be written in sheet music. Such problems may be solved with other rule based solutions specific to each instrument that requires them or more complex approaches. Even though, in frameworks of this thesis, simple threshold-based solution was used for the note tracking task, it is important to know other approaches for this problem.

There were several more complex solutions applied to the problem of note tracking. Following are some of them, though without detailed description of the approach.

Hidden Markov models (HMMs) are frequently used at a stage of postprocessing of note tracking. In his work *A discriminative model for polyphonic piano transcription*[29], Graham Poliner proposes a note tracking method that utilizes pitch-wise HMMs, where each HMM has two states, indicating note activity and inactivity. HMM parameters (state transitions and priors) were learned directly from a ground-truth training set, while the observation probability is given by the posteriogram output for a specific pitch.

In *Polyphonic music transcription using note event modeling*[30] by Matti P Ryynanen and Anssi Klapuri, a feature-based multi-pitch detection system was combined with a musicological model for estimating musical key and note transition probabilities. Note events are described using 3-state HMMs, which model the envelope (attack, sustain, and noise/silence states) of each sound. In addition, context-dependent HMMs were employed in *Automatic transcription of recorded music*[14] for determining note events by combining the output of a multi-pitch detection system with an note-start detection system.

Finally, dynamic Bayesian networks (DBNs) were proposed by Shigeki Sagayama et al. in their paper[31] for note tracking. They used the pitch activation of an NMF-based multi-pitch detection algorithm as input. The DBN has a note layer in the lowest level, followed by a note combination layer. Model parameters were learned using MIDI files from F. Chopin piano pieces.

1.4 Tuning, time signature, key, and tempo estimation

There are several other subtasks of AMT systems that have to be resolved to be able to generate correct transcription in a form of sheet music. Also, such estimates, if properly incorporated to the system, may improve estimates of detected pitches and their durations, events. Tuning, time signature, key, and tempo estimation are such tasks.

1.4.1 Key and chord detection

Most Western music has a harmonic organisation around one key. The key is generally unchanged over whole, or at least sections of musical pieces. At one point in time, the harmony may be described by chord, which is a combinations of simultaneous or sequential notes which are perceived to belong together (and in general sound nice together, even though any combination of notes has its chord). Algorithms for key (and similarly for chord) detection use template matching or HMMs. For key detection, this theses uses the simple approach defined in section 2.9.

1.4.2 Tempo and time signature estimation

The *beats* are regularly spaced in time pulses. They are the primary unit that defines tempo and rhythm of most Western music. A number of beats per unit of time (commonly per minute in sheet music) define *tempo*. A number of beats per uniform repetitive sections in score (bars) define *time signature*. In order to interpret an audio recording in terms of such a structure (which is necessary in order to produce Western music notation), the first step is to determine the rate of the most salient pulse, which is the tempo.

Algorithms used for tempo induction include autocorrelation, Fourier transforms, and periodicity transform, which are applied to audio features such as a note-start detection function (as Fabien Gouyon and Simon Dixon describe in their article *A review of automatic rhythm description systems*[32]). The next step involves estimating the timing of the beats constituting the main pulse, a task known as beat tracking. Again, numerous approaches have been proposed, such as rule-based methods (as in *Computational models of beat induction: The rule-based approach*[33] by Peter Desain and Henkjan Honing), adaptive oscillators (as in *Resonance and the perception of musical meter*[34] by Edward W Large and John F Kolen), agent-based or multiple hypothesis trackers (as in *Automatic extraction of tempo and beat from expressive performances*[35] by Simon Dixon), and other.

Böck et. al. proposed a novel tempo estimation algorithm based on a recurrent neural network that learn an intermediate beat-level representation of the audio signal which is then feed to a bank of resonating comb filters to estimate the dominant tempo[36]. This algorithm got the best score in ISMIR 2015 Audio Tempo Estimation task. The implementation by the authors is included in the opensource *Madmom* audio signal processing library which will be used in the implementation.

The final step for rhythmic analysis consists of estimating the time signature, which indicates how beats are grouped and subdivided at respectively higher and lower metrical levels, and assigning each note-start and offset time to a position in this structure[37].

Chapter 2

Analysis and design

This chapter defines architecture of the chosen solution. It provides details of used approaches for music sources separation and models used in it, pitch extraction and signal analysis, event detection, etc.

2.1 Architecture

The implementation of the system is separated into logical parts responsible for sound data streaming, music source separation, pitch and events detection, transcription and score generation. Following diagram shows architecture of the solution. Arrows represent data flow. Dotted arrows represent flow that is optional. If given parameters (like tuning, tempo, time signature and key) are specified by user, they are not being estimated. Each rectangular block represents logical module in implementation.

Detailed description of each component is in the dedicated section following the diagram on fig. 2.1.

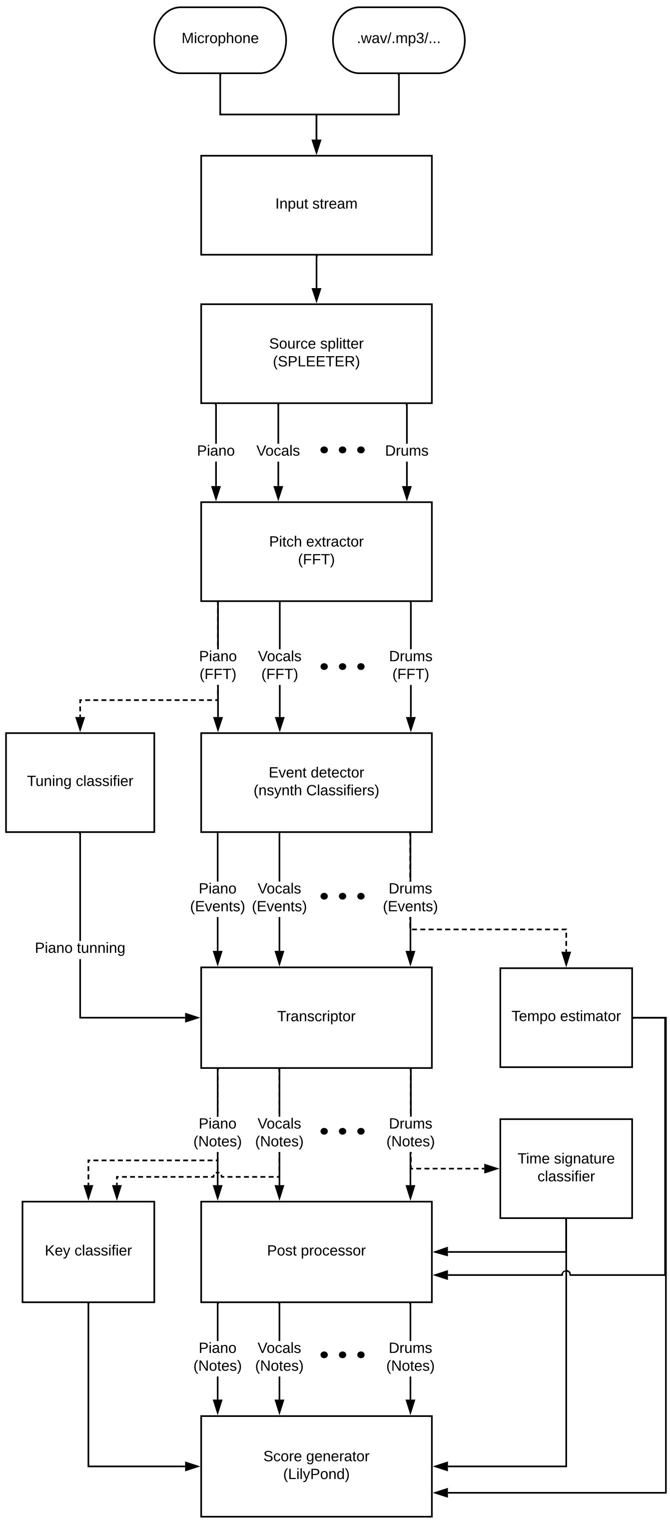


Figure 2.1: Architecture of the implementation.

2.2 Audio streaming

This implementation directly works only with Waveform Audio File Format (WAVE) (.wav/.wave). Any other format is converted to WAVE first, then processed.

2.2.1 WAVE format

WAVE is an audio file format standard, developed by Microsoft and IBM, for storing an audio bitstream on PCs. What's important for this thesis and implementation is that it stores data in chunks in Linear pulse-code modulation (LPCM) format. This format allows to perform Discrete Fourier transform (DFT) used in pitch extraction.

2.2.2 Sampling rate

LPCM mentioned above stores sampled amplitude of recorded audio at specific sampling rate (frequency, measured in Hz).

The most common sampling rate is 44.1 kHz, or 44100 samples per second. This is the standard for most consumer audio, used for formats like CDs[38].

The sampling rate determines the range of frequencies captured in digital audio. The lowest frequency a person can hear is 20 Hz. The highest frequency humans can hear are in the range of 20.000 Hz, but only young people can hear such high tones[39]. According to *Nyquist Theorem*, a signal which has a Fourier transform having only frequencies upto a certain maximum f_m, we can obtain the analog signal $f(t)$ from the sampled signal $f'(t)$ by passing the sampled signal $f'(t)$ through a low pass filter provided that the sampling frequency f_s is more than twice the maximum frequency f_m present in the signal i.e. , $f_s > 2f_m$[40]. Hence, having 44100 Hz sampling rate, we can reproduce and analyse frequencies up to 22050 Hz (assuming an ideal low pass filter). Otherwise, if recorder has a sampling rate lower than $2\times$ the highest frequency (which was not cut off by low pass filter) it causes the effect called *aliasing*, which introduces unexpected sounds in the recording that were not present in the original sound. If the sampling frequency is too low the frequency spectrum overlaps, and becomes corrupted[40].

The implementation is able to process input sound with any sampling rate, though lower sampling limits processed frequencies range to lower pitches.

2.3 Music source separation

First step of the sound processing is separation of the sound into source instruments (i.e. voice, guitar, piano, etc.)

As was mentioned in the previous chapter, this implementation uses *Spleeter* for separation of source instruments. *Spleeter* is a fast and state-of-the art music source separation tool with pre-trained models[1]. Its implementation contains three pre-trained models:

- vocals/accompaniment separation,
- 4 stems separation as in SiSeC[4] (vocals, bass, drums and other),
- 5 stems separation with an extra piano stem (vocals, bass, drums, piano and other). It is, to the author's knowledge, the first released model to perform such a separation.

Estimations for all the models is performed in a frequency domain of the sound. Meaning that sound data from time domain is converted to frequency domain using FFT, passed to the models described in section 1.1.2 about U-net architecture. Output of the model is separated tracks for each instrument and voice. To get sound of each instrument and voice in time domain (as it would be represented in WAVE), we would need to pass it through Inverse Discrete Fourier transform (IDFT). But it is not necessary, as all the subsequent processing will be performed on the sound in frequency domain.

More about FFT is in the following section 2.4 about pitch extraction.

2.4 Pitch extraction

As was mentioned in section 1.2 there are many approaches to pitch (and specifically to multi-pitch) detection. The one that is presented in this theses utilizes a combination of ideas defined in works of Matti P Ryynanen et al.[30], Arnaud Dessein et al.[27] and Paris Smaragdis et al.[22]. Solution is joint, thus estimates played notes at a given moment all at once (opposed to iterative approaches). It attempts to detect frequencies similarly to matrix factorization techniques through analysis of sound spectrogram. But instead of matrix factorization (NMF), this work attempts to detect notes' events, specifically their envelopes (more on the sound envelope in section 2.4.1), using ML models. Specification of the used data and training of the models is in the section 2.5.1.

2.4.1 Sound envelope

Sound envelope is a variation of the sound volume in time[41]. Sound envelope consists of 4 stages: attack, decay, sustain, and release (ADSR):

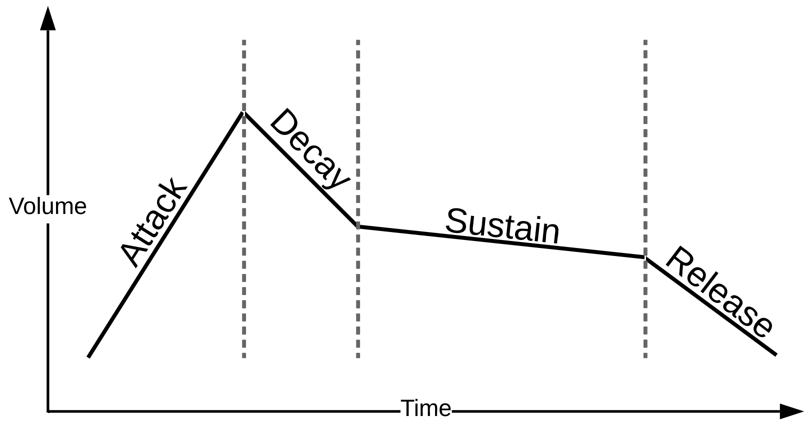


Figure 2.2: Sound envelope.

fig. 2.2 shows a theoretical simple ADSR model of sound envelope. But different instruments produce different envelopes depending on a nature of sound extraction:

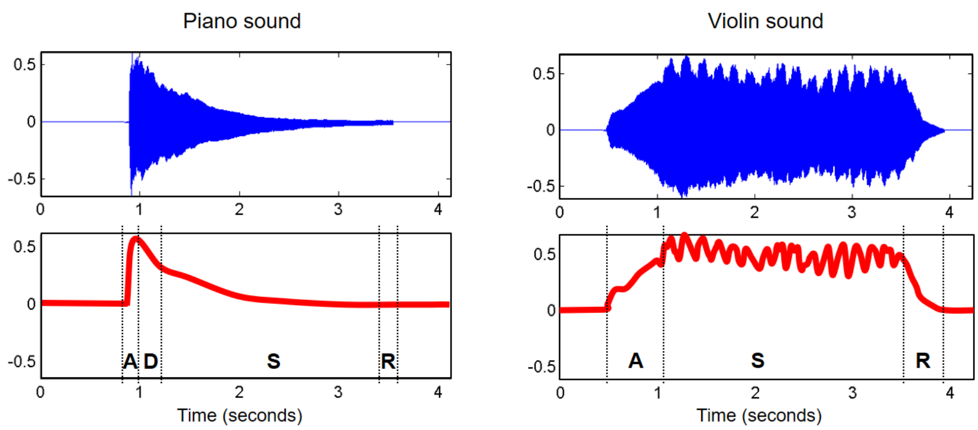


Figure 2.3: Sound envelopes of piano and violin[42].

As seen on the fig. 2.3, piano and any other instrument that produces sound by hitting, tapping or pinching of a string (like guitar, harp, bandura, balalaika, etc.), will produce similar envelope with defined attack (the moment of piano key pressing; pinching or hitting a string on guitar, etc.), decay and sustain (when piano key remains pressed or piano sustain pedal is used, etc.) and release (when piano key and sustain pedal are released, guitar strings are muted, etc.).

2.4.2 Multiresolution FFT

https://pdfs.semanticscholar.org/d55f/984d569786e1bbf945f7683361ffbbfff79a.pdf

2.5 Event detection

Note and subsequently its pitch and start are estimated by detecting its envelope. Having a sample of sound (change of volume of each pitch as determined from FFT) of duration k seconds specified by implementation, predictive model attempts to estimate whether note has been played at a given point in time by detecting its envelope that should look similar for each note of the given instrument. That means that there will be a model for each predefined instrument trained on its samples (more in section 2.5.1).

2.5.1 Data and model training

Dataset for training of the above-mentioned models was generated from the *NSynth dataset*[43]. *NSynth* is an audio dataset containing 305,979 musical notes, each with a unique pitch, timbre, and envelope. For 1,006 instruments from commercial sample libraries, there are generated four second, monophonic 16kHz audio snippets, referred to as notes, by ranging over every pitch of a standard MIDI piano (21-108) as well as five different velocities (25, 50, 75, 100, 127)[43].

NSynth contains samples for 11 different instruments: bass, brass, flute, guitar, keyboard, mallet, organ, reed, string, synth_lead, vocal. They are all stored in WAVE format and have needed metadata in JSON format alongside with them. Metadata for each sample includes instrument, note, pitch and velocity in Musical Instrument Digital Interface (MIDI) format (in the range [0, 127]), and sampling rate.

Spleeter, used for source separation, is able to separate sound only into 5 source instruments. Hence only those samples from NSynth will be used to generate models.

The way, the training dataset is generated, is completely determined by the way, the input stream is processed during transcription. Preprocessing of input stream and training data will be the same. The whole data flow is shown on fig. 2.4.

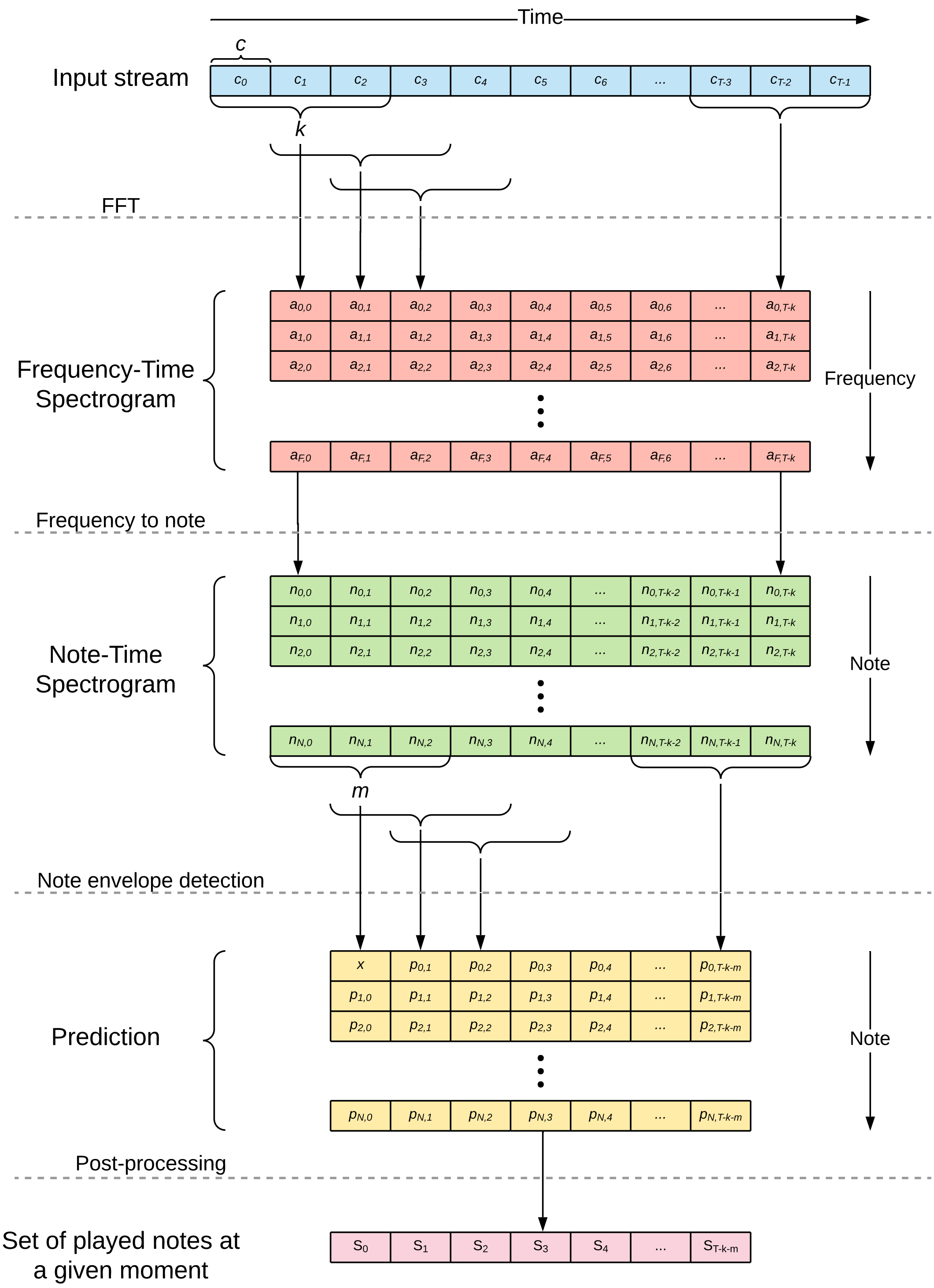


Figure 2.4: Data flow for pitch detection.

As shown on fig. 2.4, input stream (blue) is a stream of data read from input file or microphone (or any other input). It is read by chunks of size c determined by implementation. Each window of k chunks is passed through FFT to transform data to a frequency domain. Taking several chunks of data to pass it through FFT increases its accuracy, peaks of played pitches become more prominent and output becomes more robust to noise and phase shifts. Overlapping of k-sized windows allows producing more data-points per second and subsequently features for predictive model. Having T chunks and window of size k, produced spectrogram is of size $F, T-k$ where F is a number of detected frequencies.

After transforming input to time-frequency spectrogram (red), frequencies are translated to musical notes (green). Assuming equal temperament tuning and A with 440Hz frequency (actual tinning will be estimated later in the analysis), frequencies are converted to the closest note. Frequencies converted to the same note are filtered to leave only the highest volume value.

The output of previous step is passed to the model of a given instrument by window of size m. So m is a number of input features of the model. The model attempts to classify whether given window contains an envelope of a played sound that starts from a given point in time. So for $\forall i \in [0, N], j \in [0, T-k-m]$, $p_{i,j}$ (pink) shows prediction of the model for note i at a time j.

Training data is generated from NSynth dataset in a similar fashion. Positive labels are set for the pitch being played in a sample, negative for all the others. Also negative examples are generated from the same sample for played pitch but with a time shift, starting the example from or ending it somewhere on the middle of the actual sound envelope.

2.5.2 Post-processing

Duration of the note is determine by its start (start of the sample passed to the model) and its end (moment, when note's volume lowers under the predefined threshold). As was mentioned in chapter 1, implementation also utilizes several simple postprocessing ideas:

- if the duration of the note is too short, it is omitted,
- if the duration between end and next start of the same note is too short, it is combine into a single note,
- if the note played at the same time with another note but with the significant difference in volume, the quieter note is omitted.

2.6 Tuning classification

Tuning of the instrument is not a part of score transcribed into sheet music and most of the Western music follows the same tuning system. Specifically, equal temperament system with A tuned to 440Hz. But tuning estimation is an essential part for correct reproduction of the sound.

There are two primary parts to tuning estimation: detection of frequency of base note, commonly A:

Figure 2.5: Base A commonly tuned to 440Hz.

and tuning system, like equal temperament, pythagorean, meantone, etc. While tuning system in the Western music very rarely diverges from equal temperament, frequency of a base note can often be chosen to be different from 440Hz. That also often might happen for instruments that are often being retuned like guitars as tuning "by ear" by person that does not have a perfect pitch[4] is defined by tuning of the string relative to which all other strings are tuned.

Having found one played frequency F in sound (or several frequencies for better precision), it is matched to the closest note N in 440Hz-A-tuning, then real frequency of A note $f(A)$ is calculated as:

$$f(A) = F * 2^{\frac{t(N)-t(A)}{12}}$$

where $t(N)$ is an index number of the note(semitone) N (for example in MIDI representation). Now, all the other notes can be calculated in the same way.

2.7 Tempo estimation

As was mentioned in section 1.4.2, *Madmom* library will be used for the task of tempo estimation. Authors use a recurrent neural network to learn an intermediate beat-level representation of the audio signal. The output of the neural network is a beat activation function, which represents the probability of a frame being a beat position. And instead of processing the beat activation function to extract the positions of the beats, authors use it directly as a one-dimensional input to the bank of resonating comb filters. Comb filtering can be defined as "the frequency response caused by combining a sound with its delayed duplicate. The frequency response displays a series of peaks and dips caused by phase interference. The peaks and dips look like the teeth in a comb, with very narrow, deep notches where signals are attenuated."[45]. Using comb filters with different lags (delays) implementation of madmom detects at which

[4] *Perfect pitch* or *absolute pitch* is the ability to identify a note by hearing it. The ability is considered remarkably rare, estimated to be less than one in 10,000 individuals[44].

lag the beat of the sound resonates the most. Given lag would then define a tempo.

The range of possible tempo values (beats per minute (BPM)) t is limited to $1 \leq t \leq 128$ and to only whole numbers. This is decided so that the range can include loops that last from only 1 beat to 128 beats, which would correspond to a maximum of 32 bars in $\frac{4}{4}$ meter.

2.8 Time signature estimation

Problem of time signature or *meter* estimation is similar to tempo detection in a sense that the basic idea of it is finding its repetitiveness, recurrence - beat for tempo and content of a sheet music bar for time signature. That is why solutions for these problems often overlap.

Another approach for tempo as well as for time signature estimation is autocorrelation modeling. Autocorrelation modeling is used to determine the length of the bar - number of beats per each meter. Technically, time signature definition can contain any numbers for number of beats per measure (top number) and the note value that receives one beat (bottom number). But most of music peaces of western music use powers of 2 as a note value (otherwise it is called irrational measure) and rarely higher than 8 (♫). As for number of beats per bar, a 1 or values higher than 12 are considered to be the unusual time signatures. So the implementation limits estimation to these ranges.

Knowing the tempo - a number of fourth notes (♩) per minute, taking an average volume of the notes (0 if no notes are there) in all position in time of sixteenth notes(𝅘𝅥𝅯) produces the time series on which implementation models autocorrelation. Assuming that rhythmical structure of the bar and position of strong and weak[5] beats is continues through the whole peace or its significant part, the lag of modeled autocorrelation will determine the number of sixteenth notes per bar.

Autoregressive integrated moving average (ARIMA) model is used to determine the lag. ARIMA is a class of models that "explains" a given time series based on its own past values, that is, its own lags (AR part) and the lagged forecast errors (MA part), so that equation can be used to forecast future values. Specifically its simpler version AR will be used. AR is defined as follows:

$$Y_t = \alpha + \beta_1 * Y_{t-1} + \beta_2 * Y_{t-2} + \cdots + \beta_n * Y_{t-n} + \epsilon_1$$

where Y_t is value measured in time t, α is the intercept term, β_k is coefficient of the first lag, and ϵ is a noise. All of βs and α are estimated by the model. The higher the absolute value β_k - the higher the correlation between the signal

[5]Commonly, some notes per bar are *strong*(louder) and some are *weak*(quieter). This determines accents in measure. For example in $\frac{4}{4}$ time, first beat is often the loudest(strong), third is also strong, but not as strong as the first, and second and fourth are weak.

k	16 (𝅘𝅥𝅯)	8 (♫)	4 (♩)	2 (𝅗𝅥)
8	$\frac{8}{16}$	$\frac{4}{8}$	$\frac{2}{4}$	$\frac{1}{2}$
10	$\frac{10}{16}$	$\frac{5}{8}$		
12	$\frac{12}{16}$	$\frac{6}{8}$	$\frac{3}{4}$	
14	$\frac{14}{16}$	$\frac{7}{8}$		
16	$\frac{16}{16}$	$\frac{8}{8}$	$\frac{4}{4}$	$\frac{2}{2}$
18	$\frac{18}{16}$	$\frac{9}{8}$		
20	$\frac{20}{16}$	$\frac{10}{8}$	$\frac{5}{4}$	
22	$\frac{22}{16}$	$\frac{11}{8}$		
24	$\frac{24}{16}$	$\frac{12}{8}$	$\frac{6}{4}$	$\frac{3}{2}$
26	$\frac{26}{16}$	$\frac{13}{8}$		
28	$\frac{28}{16}$	$\frac{14}{8}$	$\frac{7}{4}$	
30	$\frac{30}{16}$	$\frac{15}{8}$		
32	$\frac{32}{16}$	$\frac{16}{8}$	$\frac{8}{4}$	$\frac{4}{2}$

Table 2.1: Time signature selection table.

and its copy delayed on lag k. Obviously the highest correlation of a signal is with its 0 lag. But as was mentioned above, the choice is limited to range $2 \leq k \leq 12$ with 8 as a shortest note value. So the coefficient β are estimated from 8th up to 32nd lag to determine time signatures from $\frac{2}{4}$ (which in time is equivalent to $\frac{8}{16}$) up to $\frac{12}{8}$ (which in time is equivalent to $\frac{24}{16}$) and up to $\frac{8}{4}$ or $\frac{4}{2}$ (which in time are equivalent to $\frac{32}{16}$).

Technically any score for which appropriate k was found, can be written within $\frac{k}{16}$ measure. But it is better to identify the best logical value for the number of beats per measure for simplification of reading of the score and convert the time signature to either $\frac{k/2}{8}$, $\frac{k/4}{4}$, or $\frac{k/8}{2}$.

Having found a value of k, converting is performed according to a table 2.1 as indicated by green cells for even values of k. Odd values of k are not expected as they are very unusual in measures of $\frac{k}{16}$. If they are odd time signature is left as is. Important to note that measures indicated by green cell are more common than their counterparts within a row but time signatures like $\frac{6}{8}$ and $\frac{2}{2}$ are also widely used in music. But it is hard to objectively identify which of the measures $\frac{6}{8}$ or $\frac{3}{4}$, $\frac{2}{2}$ or $\frac{4}{4}$ should be used.

2.9 Key classification

Key signature is a part of sheet music notation that simplifies it by avoiding redundant repetitive accidentals (sharps and flats) and by defining the set of used notes which in its turn defines the set of used chords, their progressions and harmonic functions in a piece of music. Detailed description of key signature is in a appendix B.3.6.

As was mentioned in section 1.4.1, there are several approaches to key classification including template matching or HMMs. The best paid solution for key detection is *Mixed In Key*[46] having 95% accuracy on their test dataset. The best free solution is *KeyFinder*[47] with accuracy of 77%. But it is implemented in C++ so is hard to incorporated into Python implementation used in this work.

A simple heuristical solution was used in the framework of this thesis. For each note that has a sharp counterpart (C, D, F, G, and A), if its semitone-higher (sharper) note appears more often than its natural note, its sharp symbol is included into a key signature.

2.10 Post processing

2.11 Transcription

2.12 Score generation

Chapter 3

Implementation

This chapter provides details of implementation, used tools, training and testing of the models.

3.1 Used tools

3.2 Music source separation

By default, this implementation uses five stems separation. Otherwise it can be defined as a parameter of analysis. If any of the output sheet music scores does not have any notes, it is ignored and does not have output sheet music.

Chapter 4

Testing

Conclusion

Possible improvements

Chords estimation, crescendo - descendo

Key note probability

Bibliography

[1] Hennequin, R.; Khlif, A.; et al. Spleeter: A Fast And State-of-the Art Music Source Separation Tool With Pre-trained Models. Late-Breaking/Demo ISMIR 2019, November 2019, Deezer Research.

[2] Jansson, A.; Humphrey, E.; et al. Singing voice separation with deep U-Net convolutional networks. October 2017. Available from: `https://openaccess.city.ac.uk/id/eprint/19289/`

[3] Pretet, L.; Hennequin, R.; et al. Singing Voice Separation: A Study on Training Data. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, doi:10.1109/icassp.2019.8683555. Available from: `http://dx.doi.org/10.1109/ICASSP.2019.8683555`

[4] Stöter, F.-R.; Liutkus, A.; et al. The 2018 Signal Separation Evaluation Campaign. 2018, `1804.06267`.

[5] Rafii, Z.; Liutkus, A.; et al. The MUSDB18 corpus for music separation. Dec. 2017, doi:10.5281/zenodo.1117372. Available from: `https://doi.org/10.5281/zenodo.1117372`

[6] Liutkus, A.; Stöter, F.-R.; et al. The 2016 Signal Separation Evaluation Campaign. In *Latent Variable Analysis and Signal Separation - 12th International Conference, LVA/ICA 2015, Liberec, Czech Republic, August 25-28, 2015, Proceedings*, edited by P. Tichavský; M. Babaie-Zadeh; O. J. Michel; N. Thirion-Moreau, Cham: Springer International Publishing, 2017, pp. 323–332.

[7] Dumoulin, V.; Visin, F. A guide to convolution arithmetic for deep learning. 2016, `1603.07285`.

[8] Chunghsin Yeh; Robel, A.; et al. Multiple fundamental frequency estimation of polyphonic music signals. In *Proceedings. (ICASSP '05). IEEE*

International Conference on Acoustics, Speech, and Signal Processing, 2005., volume 3, March 2005, ISSN 2379-190X, pp. iii/225–iii/228 Vol. 3, doi:10.1109/ICASSP.2005.1415687.

[9] A. Degani, P. M., R. Leonardi. A pitch salience function derived from harmonic frequency deviations for polyphonic music analysis. Sept. 2014. Available from: `http://www.cmap.polytechnique.fr/~bacry/MVA/getpapers.php?file=multipitch_2.pdf&type=pdf`

[10] Kumar, S.; Stephan, K. E.; et al. Hierarchical processing of auditory objects in humans. *PLoS computational biology*, volume 3, no. 6, 2007: p. e100.

[11] Dressler, K. Multiple fundamental frequency extraction for MIREX. *Eighth Music Information Retrieval Evaluation eXchange (MIREX)*, 2012.

[12] Pertusa, A.; Inesta, J. M. Multiple fundamental frequency estimation using Gaussian smoothness. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2008, pp. 105–108.

[13] Reis, G.; Fonseca, N.; et al. Hybrid genetic algorithm based on gene fragment competition for polyphonic music transcription. In *Workshops on Applications of Evolutionary Computation*, Springer, 2008, pp. 305–314.

[14] Grosche, P.; Schuller, B.; et al. Automatic transcription of recorded music. *Acta Acustica united with Acustica*, volume 98, no. 2, 2012: pp. 199–215.

[15] Nam, J.; Ngiam, J.; et al. A Classification-Based Polyphonic Piano Transcription Approach Using Learned Feature Representations. In *Ismir*, 2011, pp. 175–180.

[16] Humphrey, E. J.; Bello, J. P.; et al. Feature learning and deep architectures: new directions for music informatics. *Journal of Intelligent Information Systems*, volume 41, no. 3, 2013: pp. 461–481.

[17] Emiya , V.; Badeau , R.; et al. Multipitch Estimation of Piano Sounds Using a New Probabilistic Spectral Smoothness Principle. *IEEE Transactions on Audio, Speech, and Language Processing*, volume 18, no. 6, Aug 2010: pp. 1643–1654, ISSN 1558-7924, doi:10.1109/TASL.2009.2038819.

[18] Goto, M. A real-time music-scene-description system: predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication*, volume 43, no. 4, 2004, ISSN

0167-6393, doi:https://doi.org/10.1016/j.specom.2004.07.001, special Issue on the Recognition and Organization of Real-World Sound. Available from: `http://www.sciencedirect.com/science/article/pii/S0167639304000640`

[19] Kameoka, H.; Nishimoto, T.; et al. A Multipitch Analyzer Based on Harmonic Temporal Structured Clustering. *IEEE Transactions on Audio, Speech, and Language Processing*, volume 15, no. 3, March 2007: pp. 982–994, ISSN 1558-7924, doi:10.1109/TASL.2006.885248.

[20] Peeling, P. H.; Godsill, S. J. Multiple pitch estimation using non-homogeneous Poisson processes. *IEEE Journal of Selected Topics in Signal Processing*, volume 5, no. 6, 2011: pp. 1133–1143.

[21] Koretz, A.; Tabrikian, J. Maximum a posteriori probability multiple-pitch tracking using the harmonic model. *IEEE Transactions on Audio, Speech, and Language Processing*, volume 19, no. 7, 2011: pp. 2210–2221.

[22] Smaragdis, P.; Brown, J. C. 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)*, IEEE, 2003, pp. 177–180.

[23] Vincent, E.; Bertin, N.; et al. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech, and Language Processing*, volume 18, no. 3, 2009: pp. 528–537.

[24] Bertin, N.; Badeau, R.; et al. Enforcing harmonicity and smoothness in Bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, volume 18, no. 3, 2010: pp. 538–549.

[25] Ochiai, K.; Kameoka, H.; et al. Explicit beat structure modeling for non-negative matrix factorization-based multipitch analysis. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2012, pp. 133–136.

[26] Grindlay, G.; Ellis, D. P. Transcribing multi-instrument polyphonic music with hierarchical eigeninstruments. *IEEE Journal of Selected Topics in Signal Processing*, volume 5, no. 6, 2011: pp. 1159–1169.

[27] Dessein, A.; Cont, A.; et al. Real-time polyphonic music transcription with non-negative matrix factorization and beta-divergence. 2010.

[28] Bello, J. P.; Daudet, L.; et al. Automatic piano transcription using frequency and time-domain information. *IEEE Transactions on Audio, Speech, and Language Processing*, volume 14, no. 6, 2006: pp. 2242–2251.

[29] Poliner, G. E.; Ellis, D. P. A discriminative model for polyphonic piano transcription. *EURASIP Journal on Advances in Signal Processing*, volume 2007, no. 1, 2006: p. 048317.

[30] Ryynanen, M. P.; Klapuri, A. Polyphonic music transcription using note event modeling. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2005.*, IEEE, 2005, pp. 319–322.

[31] Raczyński, S. A.; Ono, N.; et al. Note detection with dynamic bayesian networks as a postanalysis step for NMF-based multiple pitch estimation techniques. In *2009 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, IEEE, 2009, pp. 49–52.

[32] Gouyon, F.; Dixon, S. A review of automatic rhythm description systems. *Computer music journal*, volume 29, no. 1, 2005: pp. 34–54.

[33] Desain, P.; Honing, H. Computational models of beat induction: The rule-based approach. *Journal of new music research*, volume 28, no. 1, 1999: pp. 29–42.

[34] Large, E. W.; Kolen, J. F. Resonance and the perception of musical meter. *Connection science*, volume 6, no. 2-3, 1994: pp. 177–208.

[35] Dixon, S. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, volume 30, no. 1, 2001: pp. 39–58.

[36] Böck, S.; Korzeniowski, F.; et al. madmom: a new Python Audio and Music Signal Processing Library. In *Proceedings of the 24th ACM International Conference on Multimedia*, Amsterdam, The Netherlands, 10 2016, pp. 1174–1178, doi:10.1145/2964284.2973795.

[37] Cemgil, A. T.; Kappen, B. Monte carlo methods for tempo tracking and rhythm quantization. *arXiv preprint arXiv:1106.4863*, 2011.

[38] Digital Audio Basics: Sample Rate and Bit Depth. 7 2019, [Cited 2019-11-24]. Available from: `https://www.izotope.com/en/learn/digital-audio-basics-sample-rate-and-bit-depth.html`

[39] Wendt, S. *Roots of Modern Technology: An Elegant Survey of the Basic Mathematical and Scientific Concepts*. Springer Berlin Heidelberg, 2010, ISBN 9783642120626. Available from: `https://books.google.com.ua/books?id=c8TdQmtOD-AC`

[40] Ghosh, S. *Signals and Systems*. Always Learning, Pearson Education Canada, 2005, ISBN 9788177583809. Available from: `https://books.google.com.ua/books?id=Sg6y_EBYCEsC`

[41] Dregvaite, G.; Damasevicius, R. *Information and Software Technologies: 21st International Conference, ICIST 2015, Druskininkai, Lithuania, October 15-16, 2015, Proceedings*. Communications in Computer and Information Science, Springer International Publishing, 2015, ISBN 9783319247700. Available from: https://books.google.com.ua/books?id=-pLDCgAAQBAJ

[42] Müller, M.; Zunner, T. Envelope and ADSR Model. [Cited 2019-11-28]. Available from: https://www.audiolabs-erlangen.de/resources/MIR/FMP/C1/C1S3_Timbre.html

[43] Engel, J.; Resnick, C.; et al. Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. 2017, arXiv:1704.01279.

[44] Ingmire, J. Acquiring 'perfect' pitch may be possible for some adults. May 2015, [Cited 2019-12-23]. Available from: https://news.uchicago.edu/story/acquiring-perfect-pitch-may-be-possible-some-adults

[45] Russell, R. Hearing, Columns and Comb Filtering. 2006, [Cited 2019-12-23]. Available from: http://www.roger-russell.com/columns/combfilter2.htm

[46] Vorobyev, Y. 2005, [Cited 2019-12-23]. Available from: https://mixedinkey.com/

[47] Sha'ath, I. 2011, [Cited 2019-12-23]. Available from: http://www.ibrahimshaath.co.uk/keyfinder/

[48] McGrain, M. *Music Notation: Theory and Technique for Music Notation*. Berklee guide, Berklee Press, 1990, ISBN 9780793508471. Available from: https://books.google.cz/books?id=S_y7JAZqx6QC

[49] Making Music. An Explanation of Clefs: Treble, Bass, Alto, Tenor. [Cited 2019-10-05]. Available from: https://makingmusicmag.com/explanation-clefs-treble-bass-alto-tenor/

[50] Key signature and music staff. 4 2012, [Cited 2019-10-23]. Available from: https://www.aboutmusictheory.com/key-signature.html

[51] Encyclopædia Britannica, inc. Time signature. 11 2017, [Cited 2019-10-23]. Available from: https://www.britannica.com/art/time-signature

Appendix **A**

Acronyms

WAVE Waveform Audio File Format

LPCM Linear pulse-code modulation

FFT Fast Fourier transform

DFT Discrete Fourier transform

IDFT Inverse Discrete Fourier transform

CNN Convolutional Neural Network

SiSeC Source Separation campaign

AMT Automatic music transcription

ML Machine learning

MIREX Music Information Retrieval Evaluation eXchange

HPS Harmonic partial sequence

MAP Maximum a posteriori

EM Expectation-maximisation

NMF Non-negative matrix factorisation

HMM Hidden Markov model

DBN dynamic Bayesian network

ADSR attack, decay, sustain, and release

MIDI Musical Instrument Digital Interface

BPM beats per minute

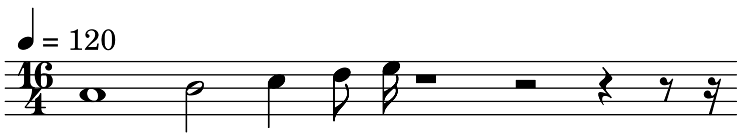
ARIMA Autoregressive integrated moving average

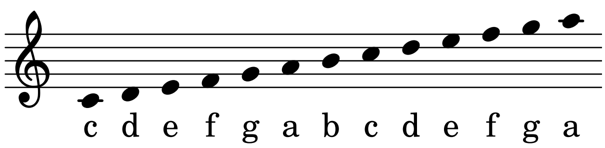
Appendix **B**

Musical notation

Music notation, when properly applied, can completely describe any musical score in a simple, concise manner. In order to achieve this, music notation must describe all definable parameters of each sound, specifically[48]:

- duration
- pitch
- dynamic
- timbre

Duration is described by time signature ($\frac{4}{4}$, $\frac{3}{4}$, $\frac{7}{8}$, etc.), tempo (primarily, beats per minute: ♩ = 120), and duration values of note-heads (𝅝, 𝅗𝅥, ♩, ♫, 𝅘𝅥𝅯, etc.) and rests (𝄼, 𝄽, 𝄾, 𝄿, etc.):

Pitch is defined by position of the note on the staff, key, accidentals (♭, ♯, ♮) and the specified clef (𝄞, 𝄢, 𝄡, etc.):



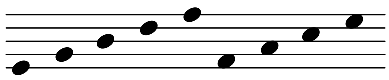
Dynamic of a sound describes its amplitude or loudness (***pp***, ***p***, ***mf***, ***f***, ***ff***, etc.), its emotional intensity and change through time.

Timbre describes specific color of a played note/sound. Timber primarily depends on the instrument played but also can define other instrumental directions (i.e. *on bell of cymbal*, etc.)

B.1 The Staff

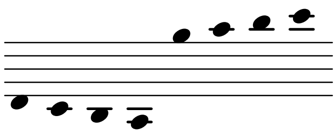
The base for all musical scores is the *staff*. All other music symbols are placed on the staff or in relation to it.

The staff consists of five horizontal lines and four spaces between the lines. Every note-head is placed on one of the lines or on one of the spaces between the lines. The higher the note-head on the staff - the higher the pitch of the produced note.



B.2 Leger Lines

Obviously, five lines and five spaces can provide only limited range of notes (precisely, eleven places to put the note-head, including just beneath the first(bottom) line and above fifth(top) line). If notes from outside this range are needed, they are placed on or between so-called *Leger lines*. These are the lines placed above or beneath the main staff only in places where they are needed, so for each note individually.



B.3 Clefs

The specified *clef* defines location of each pitch on the staff. The most commonly used clefs are the Treble and the Bass clefs[49].

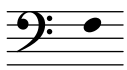
B.3.1 The Treble Clef

The *Treble Clef* (or *G clef*, because the middle curl of it encircles line on the staff that represents a G-note) is used for most high-sounding instruments (i.e. violin, guitar, ukulele, flute, clarinet, saxophone, trumpet, etc.).

As it defines second line as G, the lines on the staff, from bottom to top, are E, G, B, D, F. The spaces then are F, A, C, E. The middle C[6] goes on the first leger line below the treble staff.

B.3.2 The Bass Clef

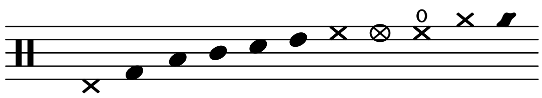
The *Base Clef* (or *F clef*, because line between two dots on the symbol represents an F-note) is used for low sounding instruments (i.e. bass guitar, cello, trombone, tuba, etc.)



As it defines fourth line as F, the lines on the staff are G, B, D, F, A, and the spaces are A, C, E, G. The middle C goes on the first leger line above the bass clef.

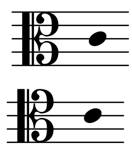
B.3.3 The Percussion Clef

The *Percussion Clef* is commonly used for drum-set notation. Each line and space represent different part of the drum kit. They are often predefined at the start of the part in so-called *key* or *legend*, or when they first appear in the score.



B.3.4 The Alto and Tenor Clefs

Alto Clef (or *C clef*, because line in the middle of the alto staff represent middle C) and The *Tenor Clef* are less often used clefs. The viola and the alto trombone are generally the only instruments that use the Alto clef. Tenor clef is occasionally used to represent the upper ranges of the cello, double bass, bassoon, and trombone.

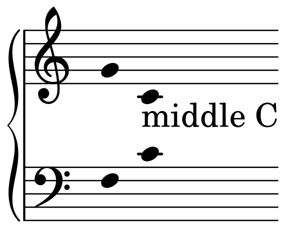


The lines of the alto staff are F, A, C, E, G, and the spaces are G, B, D, F. Similarly, for tenor clef, C is moved up one line from alto clef, making the notes on the lines D, F, A, C, E and notes in the spaces E, G, B, D.

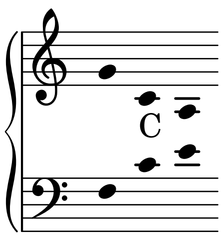
[6] *Middle C* is a commonly used reference note. It is a closest C to the middle of a standard 88 key piano (specifically, fourth C from the left). It is around 261.63 hertz.

B.3.5 The Great Staff

The *Great Staff* or the *Grand Staff* is a combination of the treble staff and the bass staff. Usually used by piano or harp musicians.

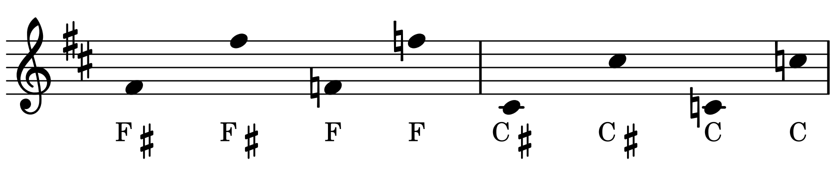


Often they also divide score into to parts played by left and right hand (i.e. on piano, treble clef part with the right hand, bass clef part with left hand). So, even if some notes belong to treble clef they may be put on leger lines above bass clef if played by left hand and vice versa.



B.3.6 Key signature

Key signature is a series of sharp symbols or flat symbols placed on the staff, designating notes that are to be consistently played one semitone higher or lower than the equivalent natural notes (for example, the white notes on a piano keyboard) unless otherwise altered with an accidental. Key signatures are generally written immediately after the clef at the beginning of a line of musical notation, although they can appear in other parts of a score, notably after a double bar[50].

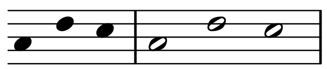


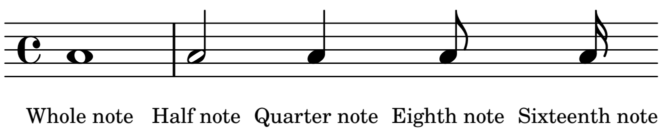
Key *D major* (defined in example above) consists of notes D, E, F♯, G, A, B, C♯. So, after the clef, notes F and C marked with a ♯, so, when they occur in a score without any accidentals, they are played one semitone higher (C♯ instead of C, etc.)

B.4 Rhythmic Description

Alongside with pitch, it is required to describe rhythm. *Rhythmic description* determines exactly when note should be played and when it should stop playing. Notationally it is defined by note-heads, stems, flags, beams, rests, and time signature.

B.4.1 Note-heads, stems, flags, beams

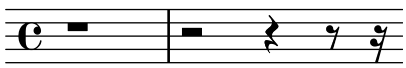
There are two types of note heads open and closed.

Stems are vertical lines attached to the side of the notes-head. Together with flags, beams, and augmentation dots they define duration value:



Two half note have the same duration as one whole note, two quoter notes have the same duration as one half note and so on.

B.4.2 Rests

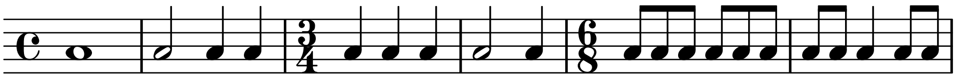
Same as for notes, we can define pauses in music - *rests*:



Whole rest, half rest, quoter rest, and so on accordingly.

B.4.3 Time signatures

Time signature is a sign that indicates the metre of a composition. Most time signatures consist of two vertically aligned numbers, such as $\frac{2}{2}$, $\frac{3}{4}$, $\frac{6}{8}$, and $\frac{11}{16}$. The top figure reflects the number of beats in each measure, or metrical unit; the bottom figure indicates the note value that receives one beat (here, respectively, half note, quarter note, eighth note, and sixteenth note). When measures contain an uneven number of beats falling regularly into two subgroups, the division may be indicated as, for instance, $\frac{3+4}{4}$ instead of $\frac{7}{4}$[51].



$^4_4$ is such a common time signature that sometimes it is specified with **C** and $^2_2$ as **¢**.

Appendix **C**

Contents of enclosed CD

```
readme.txt ........................ the file with CD contents description
exe ...................................... the directory with executables
src ..................................... the directory of source codes
    wbdcm ....................................... implementation sources
    thesis ............... the directory of LaTeX source codes of the thesis
text .......................................... the thesis text directory
    thesis.pdf ............................ the thesis text in PDF format
    thesis.ps ............................... the thesis text in PS format
```