


Chord Recognition

GEOFFROY PEETERS

IRCAM (STMS-IRCAM-CNRS-UPMC), Sound Analysis and Synthesis Team

JOHAN PAUWELS

Abstract

This chapter on chord transcription will guide the students through the process of designing a basic chord recognition tool ~~in Matlab~~  starts with a short explanation of what we mean by chord recognition, especially the generation to automatically generating lead sheets for pop and jazz music (Section 19.1). In Section 19.2, we explain the relevance of chroma features (introduced previously) for chord recognition and show the results of matching simple chord patterns to chroma data obtained from real audio. It will become obvious that frame-wise extraction leads to a strongly fragmented chord sequence, and we will show that simple temporal smoothing by low-pass filtering the chroma input improves recognition. In Section 19.3 we show how chord pattern recognition and temporal smoothing can be performed simultaneously by training and then Viterbi-decoding a hidden Markov model. Interdependency between chords and keys are then included in the hidden Markov model to perform joint chord and key recognition (Section 19.4). Section 19.5 introduces some measures for chord recognition performance and compares results between the two approaches. We close by showing what other methods have been used to tweak chroma-based chord estimation and give a summary of existing chord recognition tools.

19.1 Introduction

Chords are abstract representations of a set of musical pitches (notes) played (almost) simultaneously. Chords along with the main melody are often predominant characteristics of a music track. Well-known examples of chord reductions are the “chord sheets” where the background harmony of a music track is reduced to a succession of symbols over time (C major, C7, ...) to be played on a guitar or a piano.

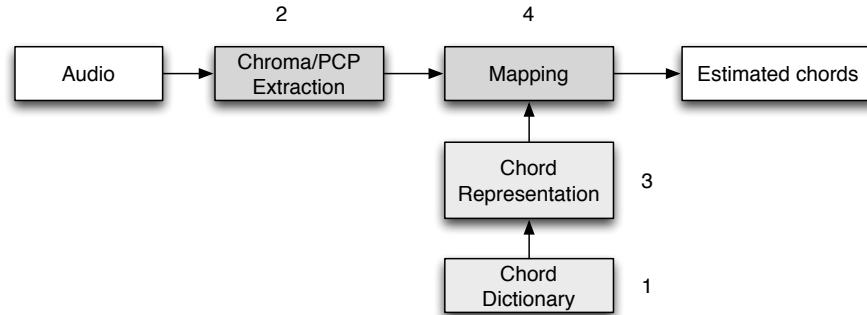


Figure 19.1: *General scheme of an automatic system for chord recognition from audio.*

In this chapter, we describe how we can automatically estimate such chord successions from the analysis of the audio signal. The general scheme of a chord recognition system is represented in Figure 19.1. It is made of the following blocks that will be described in the next sections:

1. A block that defines a set of chords that will form the dictionary over which the music will be projected (see section 19.1.1),
2. A block that extracts meaningful observations from the audio signal: chroma features or Pitch-Class-Profile extracted at each time frame (see section 19.2.1),
3. A block that creates a representation or a model of the chords that will be used to map the chords to the audio observation (see section 19.2.2),
4. The mapping of the extracted audio observations to the models that represent the various chords (see section 19.2.3).



19.1.1 Choice of the chord dictionary

When developing a chord recognition system, the first step is to choose the dictionary of chords over which the harmonic content of the music track will be projected. Examples of possible chord dictionaries are given in Table 19.1 (see also Section 3.4.4, pg. 74). One can reduce the harmonic content to the main 24 major and minor chords triads or include chords tetrads (major 7, minor 7, dominant 7) or pentads (major 9, dominant 9). Depending on this choice, the observation of the notes {c,e,g,b} in the audio can either be mapped to a CM or a CM7 label. The larger the chord dictionary, the more precise the harmonic reduction to chords, but also the more difficult the task. This difficulty does not only come from the increase in classes (chord labels can be considered as classes in a machine learning sense), but also from the equivalence between some chords (for example CM6: {c,e,g,a} has the same notes as Am7: {a,c,e,g} although not in the same order).

From a musical point of view, a single chord can be played in various ways. CM can be played in root position {c,e,g}, in first inversion {e,g,c} or second inversion

Table 19.1: Dictionary for the root notes and three possible dictionnaires for the type of chords.

Root-note	Type of the chord
c, c#, d, d# ... b	Triads: major (CM: c,e,g) , minor (Cm: c, eb , g), suspended (Csus2: c, d, g / Csus4: c, f, g), augmented (Caug: c, e, g#), diminished (Cdim: c, eb, gb)
	Tetrads: major 7 (CM7: c, e, g, b), minor 7 (Cm7: c, eb, g, bb), dominant 7 (C7: c, e, g, bb), major 6 (CM6: c, e, g, a), minor 6 (Cm6: c, eb, g, a) ...
	Pentads: major 9 (CM9: c, e g, b, d), dominant 9 (C9: c, e, g, bb, d) ...

{g,c,e} Since most current chord estimation methods rely on mapping the audio content to the twelve semi-tone classes independently of their octave positions, it is not possible to distinguish if a chord has been inverted or not.

For this last reason, chords are often estimated jointly with the local key. The root of a chord is then expressed as a specific degree in a specific key. The choice of CM6 will be favored in a C-Major key while Am7 will be favored in a A-minor key.

When estimating chords from the audio signal, we will also rely on enharmonic equivalence, i.e. we consider the note c# to be equivalent to db, and also consider the chord F#M to be equivalent to GbM.

19.2 Frame-based system for chord recognition

19.2.1 Audio signal observation: chroma or Pitch Class Profile

Chords represent a set of notes played almost simultaneously. It therefore seems natural to estimate the chords of a music track from a previous estimation of the existing pitches in its audio signal (multi-pitch estimation). However, multiple-pitch estimation is still a difficult and a very computer-time-consuming task. For this reason, most algorithms that estimate chords from an audio signal use another approach: the extraction of chroma [32] also known as Pitch-Class-Profile (PCP) [5] (see also Section 5.3.1, pg. 141).

The notion of chroma/PCP is derived from Shepard works [29] who proposes to factor the pitch of a signal into values of chroma (denoted here by $p \in [1, 12]$) and tone height (octave). For example: a4 (440 Hz) is factored as the chroma $p = 10$ at the octave 4.

The chroma/PCP representation of an audio signal is obtained by mapping the energy content of its spectrum to the 12 semitones pitch-classes (c, c#, d, d#, e, ...). More precisely, to compute the value at the chroma p , we add the energy existing at all frequencies corresponding to the possible pitches of this chroma (for example, for the chroma $p=10$, we add the energy at the frequencies corresponding to all possible octaves of the a : a0, a1, a2, a3 ...). The representation is computed at each time

frame λ . In the following we denote by $t_{\text{chroma}}[p, \lambda]$ the set of chroma vectors over time, also known as a chromagram.

Unlike multiple-pitch estimation, chroma/PCP is a mapping and not an estimation. Therefore it is not prone to errors.

19.2.1.1 Computation using the Short-Time-Fourier-Transform

Chroma/PCP can be computed starting from the Short-Time-Fourier-Transform $X_{\text{stft}}[\mu, \lambda]$ (see Section 4.5, pg. 10) where μ denotes the frequencies and λ the time frame. To compute the value of $t_{\text{chroma}}[p, \lambda]$ we simply add the energy (X_{stft}^2) at the frequencies μ of the Discrete Fourier Transform (DFT) that corresponds to the 12 semitones pitch-classes p .

$$t_{\text{chroma}}[p, \lambda] = \sum_{\mu \in p} X_{\text{stft}}[\mu, \lambda]^2 \quad (19.1)$$

To know which frequencies μ correspond to a specific p , we first convert the frequencies μ of the DFT in MIDI-scale: $m_{\mu} = 12 \log_2 \frac{\mu}{440} + 69$ (for a tuning of $a4 = 440\text{Hz}$), $m_{\mu} \in \mathbb{R}$. For example, $m_{450\text{Hz}} = 69.3891$. The value of the chroma at $p \in \mathbb{N}$ is then found by summing the energy of the spectrum at all frequencies μ that correspond to the chroma p , i.e. such that $[\text{rem}(m_{\mu}, 12) + 1] = p$ (where rem is the “remainder after division” operation and $[x]$ the “integer rounding” function). This method provides a “hard” mapping. For example the energy at $m_{452\text{Hz}} = 69.4658$ will be entirely assigned to $p=10$ while $m_{453\text{Hz}} = 69.5041$ to $p=11$.

In order to avoid this “hard” mapping a “soft” mapping is often used. In this, the energy at m_{μ} is assigned to different chroma with a weight inversely proportionnal to the distance between m_{μ} and the closest pitches. In the previous example, $m_{453\text{Hz}} = 69.5041$ will equally contribute to $m=69$ and $m=70$. For this, the summation is done through a windowing operation $h(|m - m_{\mu}|)$ where $m \in \mathbb{N}$ is one of the MIDI notes, $m_{\mu} \in \mathbb{R}$ the value of the frequency μ converted to the MIDI-scale and $|x|$ denotes the absolute value. $h(x)$ is designed such that $h(0) = 1$ and $h(1) = 0$ therefore takes the maximum value when $m_{\mu} = m$ and is equal to zero when $m_{\mu} > m + 1$ or $m_{\mu} < m - 1$. Common choice of h are the triangular, Hanning or tanh functions. We illustrate this process in Figure 19.2.

19.2.1.2 Computation using the Constant-Q-Transform

Because the frequency resolution of the Discrete Fourier Transform¹ can be too large to separate the frequencies of the lowest adjacent pitches in the spectrum², the Constant-Q Transform is often used to compute the chroma/PCP representation (see Section 4.7.2.2, pg. 125). The coefficient Q is chosen to be able to separate

¹The frequency resolution is proportionnal to the width at -6dB of the main lobe of the analysis window, which is defined by $Bw = \frac{Cw}{L}$ where Cw is the window coefficient - $Cw = 1.81$ for a Hamming window - and L the analysis window duration.

²For example, the notes c3 and c#3 are separated by only 7.8 Hz which is smaller than the frequency resolution Bw provided by a Hamming window of $L=80\text{ms}$ which is $Bw=22.62\text{ Hz}$.

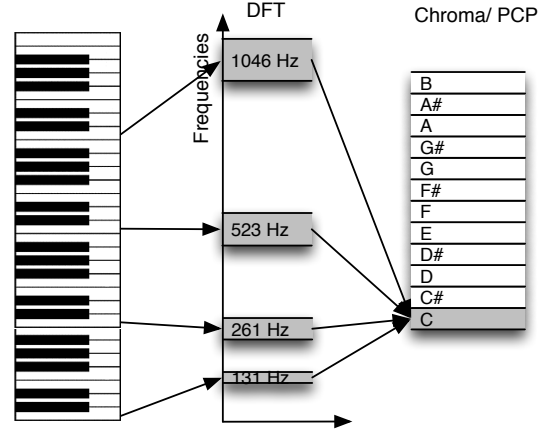


Figure 19.2: Chroma computation from the DFT.

the lowest considered pitches. For example in order to be able to separate c3 from c#3, Q is chosen such that $Q = \frac{f_k}{f_{k+1} - f_k} = \frac{130.8}{138.6 - 130.8} = 16.7$.

If the frequencies μ of the Constant-Q are chosen to be exactly the frequencies of the pitches (if $m_\mu \in \mathbb{N}$) the computation of the chroma/PCP is straightforward since it just consists in adding the values for which $\text{rem}(m_\mu, 12) + 1 = p$, without necessitating any windowing operation:

$$t_{\text{chroma}}[\lambda, p] = \sum_{o=1}^{\text{height of yellow box}} |X_{\text{CQT}}[\lambda, p + 12o]| \quad (19.2)$$

19.2.1.3 Influence of timbre on the chroma/PCP

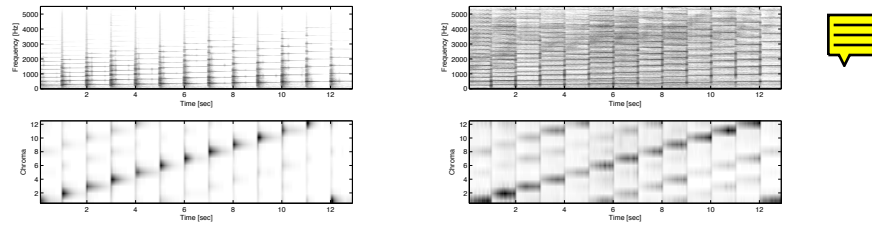
Ideally, when a musical instrument plays a pitch of c (whatever octave), we would like the chroma/PCP representation to have a single non-zero at $p = 1$: $t_{\text{chroma}}[\lambda, p] = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$. However, a musical instrument does not produce a single frequency at the pitch f_0 but an harmonic series at $f_0, 2f_0, 3f_0, 4f_0, 5f_0 \dots$. The corresponding amplitudes $(a_1, a_2, a_3, a_4, a_5 \dots)$ defines the “timbre” of the instrument. These harmonics will create components in the chroma/PCP that do not necessarily represent the pitch (see Table 19.2). For example, for a pitch c3 (130.81 Hz), its third harmonic ($3f_0 = 392.43$ Hz) has the same frequency as the note g4, its fifth harmonic ($5f_0 = 654.06$ Hz) is close to the note e5 ... When we consider the first five harmonics, the corresponding chroma/PCP vector will be $t_{\text{chroma}}[\lambda, p] = [a_1 + a_2 + a_4, 0, 0, 0, a_5, 0, 0, a_3, 0, 0, 0, 0]$. It is clear that the chroma/PCP representation depends on the timbre of the musical instrument, so it is very likely that the same note played by two different instruments results in two different chroma/PCP representations. Chroma is therefore said to be timbre-sensitive. This is illustrated on Figure 19.3.

To deal with this problem, three different strategies can be used:

define

Table 19.2: Harmonic series of the pitch c3, corresponding frequencies μ , conversion in MIDI-scale m_μ , conversion to Chroma/PCP p .

Pitch	Harmonic	Frequency μ	MIDI-scale m_μ	Chroma/PCP p
c3	f_0	130.81	48	1 (=c)
	$2f_0$	261.62	60	1 (=c)
	$3f_0$	392.43	67.01	8.01 (\simeq g)
	$4f_0$	523.25	72	1 (=c)
	$5f_0$	654.06	75.86	4.86 (\simeq e)



(a) Chromatic scale starting from c played by a piano
(b) Chromatic scale starting from c played by a violin

Figure 19.3: On each figure, the top part represents the spectrogram, the bottom the corresponding chromagram. The influence of the higher harmonics of the notes are clearly visible in the forms of extra-values in the chromagram.

1. *whiten the spectrum* (i.e., give the spectrum a flat spectral shape: $a_1 = a_2 = a_3 = a_4 = a_5 \dots$) before computing the chroma/PCP. Doing so will emphasize the presence of the higher harmonics in the chroma/PCP but will make it equal for all musical instruments, hence will make chroma/PCP timbre independent. This effect can be achieved by binarising spectral peaks that are harmonically related [34], by frequency dependent compression [27] or by applying cepstral liftering [14].
2. *remove as much as possible the presence of higher harmonics* in the spectrum before computing the chroma/PCP representations. Some suitable techniques include attenuating harmonics through the calculation of a harmonic power spectrum [14] or using a pitch salience spectrum that takes the energy of higher harmonics into consideration too [27].
3. *keep the chroma/PCP vector as it is* and consider the existence of the higher harmonics in the chords representation (see section 19.2.2).

It is also possible to combine the first approach with one of the two others.

19.2.2 Representation of the chord labels

19.2.2.1 Knowledge-driven approach

The most obvious way to represent a chord in a computer is to create a vector representing the existence or not in it of each of the 12 semitone pitches. Such a vector is often named a “chord-template”: $T_c[p]$ where c is the index of the chord. The chord templates corresponding to the chords C major, C minor and C diminished are displayed in Figure 19.4. Since these chord-templates have the same description-space as the chroma/PCP vector, it is possible to compute a distance between $T_c[p]$ and a chroma/PCP vector $t_{\text{chroma}}[\lambda, p]$ at a given time frame λ (see section 19.2.3).

As mentioned before, in order to deal with the problem generated by the existence of the higher harmonics of musical instrument sounds, it is possible to consider the existence of the higher harmonics in the chord representation. This is done by creating so-called “audio chord-templates”. For this, each existing pitch p in the chord contributes to the audio chord-template with an audio note-template. If we consider only the first five harmonics, the audio-note-template of $p = 1$ is defined by $[a_1 + a_2 + a_4, 0, 0, 0, a_5, 0, 0, a_4, 0, 0, 0, 0]$. The audio note-templates corresponding to the other p are obtained by circular permutation. To get the values of the amplitudes a_h , Izmirli [9] studied the average values of the a_h on a piano dataset. Gomez [6] proposes a theoretical model of the amplitudes in the form $a_h = 0.6^{h-1}$.

19.2.2.2 Data-driven approach

Another possibility is to learn a chord representation from a set of chord annotations through machine-learning techniques. In this case, all the extracted chroma/PCP vectors corresponding to a given annotated chord c are used to train a statistical model of this chord. Common choices for this representation are multivariate normal/Gaussian distribution model or multivariate Gaussian mixture model [18]. In the case of multivariate normal/Gaussian distribution mode, each chord c is represented by a model $\mathcal{N}_c(\mu, \Sigma)$ where μ and Σ are the mean vector and covariance matrix learned from the set of $t_{\text{chroma}}[\lambda, p]$ that belong to the specific chord c .

19.2.3 Mapping strategies between audio signal and chord labels.

The task of chord recognition consists in finding the chord expression $c(\lambda)$ with $c \in [1, C]$ (where C is the size of the chord dictionary) that best explains the succession of observations $t_{\text{chroma}}[\lambda, p]$.

Depending on the choice of the chord representation (chord templates or statistical model) and the amount of musical knowledge we want to introduce in the algorithm, various systems can be used.

19.2.4 Knowledge-driven approach

The simplest system consists in finding at each frame λ , the chord label c that minimizes a distance $d(x = t_{\text{chroma}}[\lambda, p], y = T_c[p])$. Such a distance can be a simple

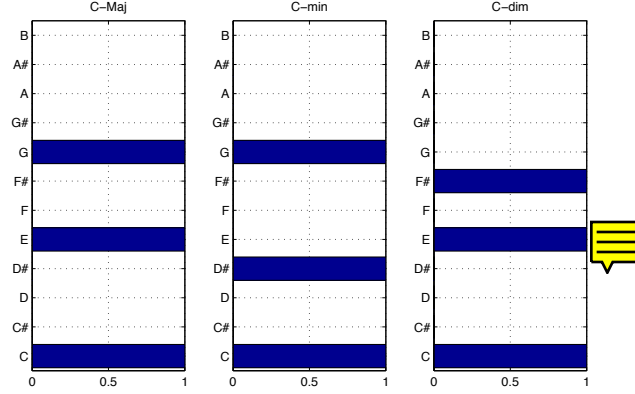


Figure 19.4: Chord templates $T_c[p]$ corresponding to the chords C major, C minor, C diminished.

Euclidean distance

$$d(\underline{x}, \underline{y}) = \sqrt{\sum_i ||x_i - y_i||^2} \quad (19.3)$$

or in order to be independent from the norm of the Chroma/PCP vector (hence independent of the local loudness of the signal), a one-minus-cosine distance:

$$d(\underline{x}, \underline{y}) = 1 - \frac{\underline{x} \cdot \underline{y}}{||\underline{x}||^2 ||\underline{y}||^2} \quad (19.4)$$

Divergences, such as Itakura-Saito or Kullback-Leibler [16] can also be used instead of distances in order to highlight additions or deletions of components p in the vectors.

Example 19.1. In Figure 19.5, we illustrate the chord recognition results obtained by using the one-minus-cosine distance. In this example, we use a dictionary of size $C = 24$ consisting of all major and minor chords. At each frame λ , we have chosen the chord $c \in C$ that has the minimal distance to $t_{\text{chroma}}[\lambda, p]$. In the figure, we display the distances between the vectors over time $t_{\text{chroma}}[\lambda, p]$ and all the chords $c \in C$ in a matrix. The chord that corresponds to the minimal distance at each frame is indicated by a circle. The top row of the Figure shows the ground-truth chord label. The audio signal corresponds to the track “I Saw Her Standing There” by The Beatles.

19.2.5 Data-driven approach

When the chords are represented by Gaussian (mixture) models instead of templates, the distance is replaced by the computation of the probability of observing a chord c given observation $t_{\text{chroma}}[\lambda, p]$: $p(c|t_{\text{chroma}}[\lambda, p])$. This posterior probability is not

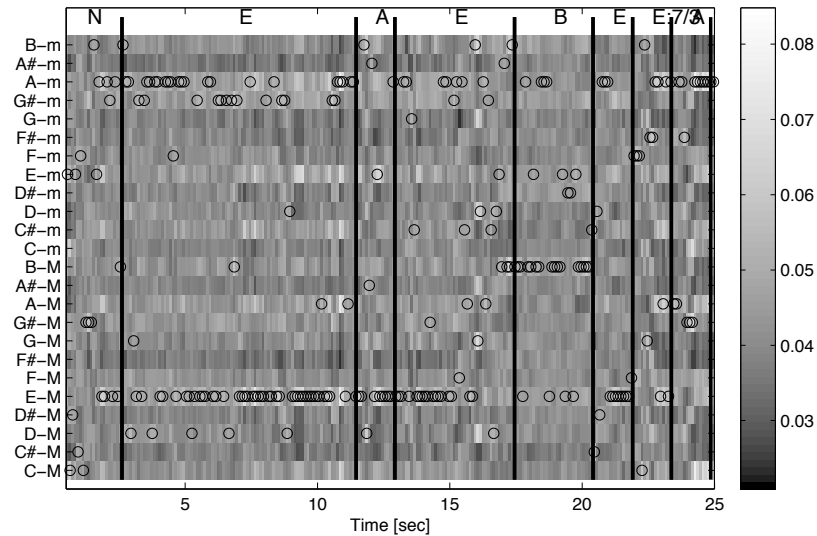


Figure 19.5: Example of frame-based chord recognition on the track “I Saw Her Standing There” by The Beatles.

calculated directly, but derived from the observation likelihood $p(c_t | \text{chroma}[\lambda, p] | c)$, which has been previously learned on a training set, through the application of Bayes’ rule (see Section 12.4, pg. 263).

19.2.6 Chord fragmentation

Whatever knowledge-driven or data-driven technique is used to define the mapping between the chroma observations and the chord representations of the system, the resulting estimation of chords over time is usually quite fragmented (visible as an unrealistically high number of jumps between chords). This is because the decision of which chord best matches the observation is taken independently at each frame, without considering adjacent frames. We however know that the frame rate of the system is higher than a realistic rate for changing chords. Therefore we expect the chord output to appear clustered over time and to have clear changes, without oscillating back and forth between the new and the old chord.

A simple process to deal with this fragmentation problem, is to apply a low-pass or median filtering over time to the output of the decision function (being a distance measure, a divergence or a likelihood).

19.3 Hidden Markov model-based system for chord recognition

Another, potentially more powerful way to achieve stability over time is the use of a hidden Markov model (HMM). In an HMM, the transition probabilities between states allow to constrain the sequence of decoded states over time. In the case of chord estimation, the chord labels form the hidden states from which we observe the Chroma/PCP features. The model is defined by

- the initial probability of the state/chord $p_{init}(c_i)$,
- the emission probability of each state/chord $p(t_{chroma}[\lambda, p]|c_i)$ and
- the transition probabilities between states/chords, i.e. the probability to transit from chord c_i at time λ to chord c_j at time $\lambda + 1$: $p_{trans}(c_j|c_i)$.

A visual representation of a simple HMM consisting of only three chords is depicted in Figure 19.6.

The initial probabilities are usually considered uniformly distributed.

For the emission probabilities, we can reuse with little or no change the same mapping strategies between chord labels and audio signal as we used in the frame-based system (by converting the distances $d(x = t_{chroma}[\lambda, p], y = T_c[p])$ to probabilities or by computing the likelihood $p(t_{chroma}[\lambda, p]|c_i)$ given the trained statistical models $\mathcal{N}_c(\mu, \Sigma)$).

The self-transition probabilities $p_{trans}(c_{j=i}|c_i)$ regulate how easy it is to change between chords and can therefore prevent fragmentation. They take on the same role as the low-pass or median filtering of the chord output. In contrast to the latter, the Viterbi algorithm used to decode the HMM does not just consider the chosen output of each frame for the temporal smoothing. It rather considers all options for each frame, also the locally suboptimal, to find the sequence of labels that optimally combines the observations and the requirement of temporal stability. Another aspect of an HMM is that the remaining transition probabilities $p_{trans}(c_{i \neq j}|c_i)$, or change probabilities, allow to encode musical rules concerning the transitions between chords. They allow to take into account the fact that music is not just a "bag of chords" but a specific temporal succession of them, i.e., certain transitions are more likely to appear than others. These combinations form the rules that are taught in music theory.

Much like the mapping between chroma observations and chord representations, these probabilities can be set manually based on a musicological knowledge or can be optimised on an annotated data set using machine learning techniques [17].

19.3.1 Knowledge-driven transition probabilities

A simple theoretical model, used in [1], that can be used to derive change probabilities, is the doubly-nested circle of fifths. It expresses the perceptual distance between all major and minor chords. A visual representation can be seen in Figure 19.7(a). It is formed by arranging major and minor chords on two concentric circles with a distance of a perfect fifth between successive roots (when moving clockwise). The perceptual distance between two chords is then calculated by counting the shortest number of steps along the circle that has to be taken to go from one chord to the other. Changing from the "major" circle to the "minor" circle counts as one step. A

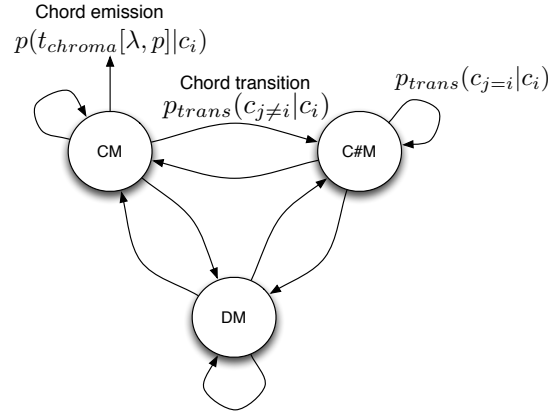


Figure 19.6: Hidden Markov model for chord estimation.

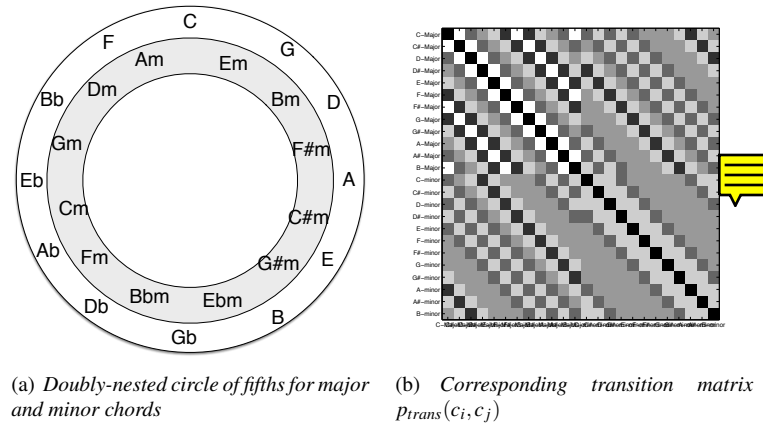


Figure 19.7: Deriving a transition matrix from a theoretic model of chord distance

drawback of this model is that it does not extend to other chord types, such as sevenths chords. A number of alternative chord distances that are more extensive in this regard are compared in [26]. The premise that all models that are based on perceptual chord distances share, is that close chords are likely to follow each other in sequence. These distances are then transformed into transition probabilities, where a small distance leads to a high probability of transition. The matrix of transition probabilities that corresponds to the doubly-nested circle of fifths can be seen in Figure 19.7(b).

Example 19.2. In Figure 19.8, we illustrate the chord recognition results obtained using a hidden Markov model for the same track as Figure 19.5. The observation probabilities of the HMM have been taken as the one-minus-cosine distance (normalized such that $\sum_i p(t_{chroma}[\lambda, p] | c_i) = 1$). The transition probabilities of the HMM

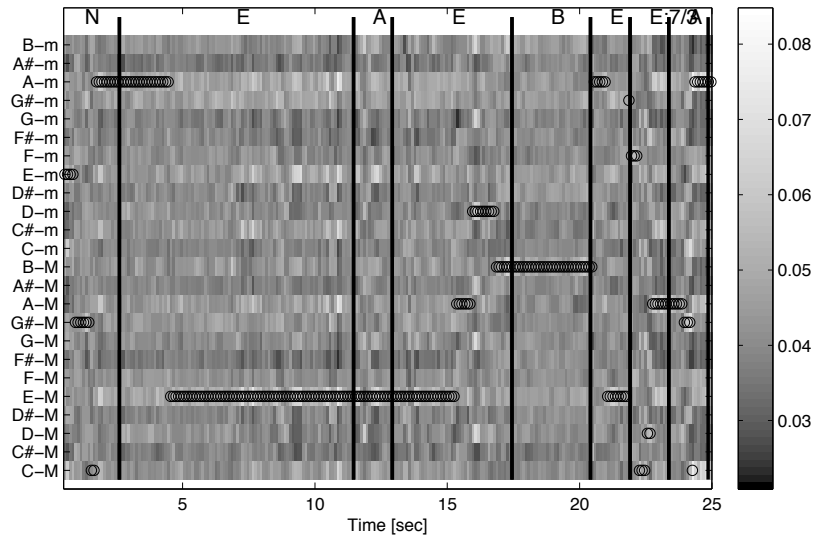


Figure 19.8: Example of HMM-based chord recognition on the track “Saw Her Standing There” by The Beatles.

have been taken as $p_{trans}(c_j|c_i) = 1 - d(c_i, c_j)/6$ where $d(c_i, c_j)$ is the distance between two chords c_i and c_j in the doubly-nested circle of fifths as explained before. It is then normalized such that $\sum_j p_{trans}(c_j|c_i) = 1 \forall i$. We represent by circles the most likely path over time as decoded by the Viterbi algorithm given the hidden Markov model parameters and the observations over time $t_{chroma}[\lambda, p]$. Compared to the results illustrated in Figure 19.5, the path obtained by the HMM-based system is much less fragmented over time.

19.3.2 Data-driven transition probabilities

Instead of explicitly relying on musicological theory to derive the transition probabilities, those can also be determined by the statistical analysis of corpus of symbolic music. In its most basic form, this amounts to counting the proportion of occurrence of each chord pair. If the set comes with synchronised audio, the HMM can also be trained such that the optimal combination of observation and transition probabilities can be found that maximises recognition performance on that set. These approaches are not always exclusive either. Since data-driven approach can lead to a local optima which is musically meaningless, one can use theoretical model to initialize the training.

19.4 Joint chord and key recognition

In order to define change probabilities that go beyond the most elementary chord transitions, we need to express chords in terms of their key. In music theory, harmony is mostly analysed as the movement of chord degrees in a key. Therefore we need the key to be available at the point where we try to recognize the chords. There are two kinds of approaches to accomplish this: - either we build a sequential system where the key is recognised first, and is then followed by a chord recognition step; - or the key and chord are recognised simultaneously.

When discussing key recognition, there is one additional distinction that needs to be made which does not exist in chord recognition. We need to decide if we want to find the *global* key or the *local* key. The former assigns a single label that applies to the whole music piece whereas the latter segments the song into segments of a constant key and labels them. Global estimation has the advantage that it is easier, as there are more observations to base the decision on. Its disadvantage is that in pieces with key changes, it will inevitably lead to a loss of information, namely the secondary keys and the location of the changes. In our use-case, this will cause the interpretation of chords as degrees relative to the key to be wrong.






19.4.1 Key-only recognition

In general, key recognition is conceptually similar to chord recognition. There is of course the difference in time-scale, especially for global key recognition, which results in differences in analysis window sizes and/or duration modelling, but the same techniques can easily be reused. Here too, the most popular signal representation is the chroma/PCP vector. They are commonly matched with key templates derived from the perceptual experiments of Krumhansl [11], or the variations of Temperley [31]. When estimating local keys, an HMM can be used to provide the required temporal stability, as in [25].

19.4.2 Joint chord and key recognition

If keys and chords are recognised simultaneously, the states of the HMM represent key-chord combinations. This leads to a strong increase in the number of states and an exponential growth of the transition matrix. To keep the number of variables tractable, and also because the current size of data sets annotated with both local keys and chords are too small to train them straightforwardly, the observation and transition probabilities can be decomposed into a combination of smaller parts as in [22]. The optimal key and chord sequences are then jointly decoded. Another option is to consider only the global key, which causes the majority of the values in the transition matrix to be set to zero. This can equivalently be seen as constructing a separate key-dependent HMM for each key, as in [12], instead of one large HMM. Finally, when the key is determined beforehand, the state space stays the same, but the optimal key path becomes deterministic instead of probabilistic, which brings the number of transitions that should be investigated at each step back down to the same number as for a chord-only HMM.


In combined  and chord recognition systems, it is common to express chords as relative degrees to a key. For example in the key of C major, the chord CM is the first degree (denoted by IM), Dm is the second (IIm), G major the fifth (VM) and so on. It is then common to tie transition probabilities together such that they are invariant under transposition. This means that a IIm to VM change is as likely in C major as in G major, but not in C minor however. The main point of expressing chords relatively to a key, is that we can then reason about chord transitions in the same way as in the study of music theory, but there is also evidence that this representation of harmony reduces confusion about the expected chords when compared to an absolute sequence of chords without the context of a key [28]. On the other hand, according to [22], the improvement in recognition performance due to the better modelling of chord changes is only modest compared to the improvements brought by a better modelling of chord duration.

A drawback of the usage of a standard HMM, is that it can only take into account pairwise transitions, whereas we know from music theory that looking at a wider context can be even more enlightening. For instance, a Dm-G7-CM  sequence is more representative of a C major key than either Dm-G7 or G7-CM  sequences. Therefore multiple attempts have been made to include higher-order models of musical context, ranging from frame-based approaches [4], over lattice rescoring of HMM output [10], to a full search over all key and chord trigram sequences [23].

19.5 Evaluating the performances of chord and key estimation

In order to compare the performance of the different methods for chord and/or key recognition we discussed in the previous section, we need a way to quantify the correctness of their estimated outputs. We therefore need an evaluation procedure that takes a sequence of timed key or chord labels and numerically expresses the extent to which it resembles a certain reference sequence, preferably obtained through manual annotation. This resemblance can be measured according to a number of different aspects.

19.5.1 Evaluating segmentation quality

A first way to compare sequences is their degree of segmentation. We only need to take into account the positions of the key or chord changes for this, not their exact labels. Consequently, the same procedures as used to evaluate other segmentation tasks can be used here (see Chapter 13, pg. 281). A commonly used pair of measures is based on the directional Hamming divergence [13]. It is calculated by matching each segment in the tested sequence to the segment in the reference  it overlaps it most, and then adding the durations of the non-covered parts in the tested sequence. Normalised by the sequence length and subtracted from one to make an increase in value correspond to an increase in performance, this gives a measure for over-segmentation. A corresponding measure for under-segmentation can be achieved by swapping the reference sequence and the sequence under test.

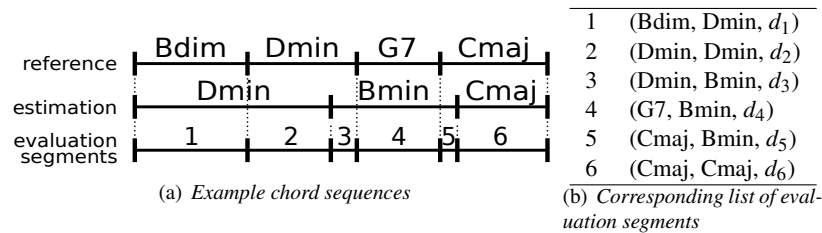


Figure 19.9: Creating a list of evaluation segments from two sequences

19.5.2 Evaluating labeling quality

A second aspect of evaluating key or chord sequences is the correspondence in harmonic content. This is complementary to the segmentation evaluation, because here the positions of the changes are not important, only the proportion of time both sequences match. What exactly constitutes a match between the two sequences will be made clear by the following steps. First of all, we line up the two sequences we want to compare and take the union of all key or chord change positions. This brings the two sequences onto a common time scale, which we call evaluation segments. We end up with a list of label pairs and associated durations. An example of this procedure for two chord sequences can be seen in Figure 19.9

Then we decide which label pairs we want to include in the evaluation. For a first overall score, all segments should be included of course, but in some cases it makes sense to evaluate only on a subset of them. If the dictionary of the algorithm is such that no output can be generated that suitably resembles the reference, that segment can be dropped such that the score can span its full range. For example, it makes sense to drop segments annotated with diminished chords from the evaluation when the algorithm can only output major and minor chords, such as segment 1 in Figure 19.9. Another possible use-case is to limit the evaluation only to certain categories of labels, for instance to compare the performance on triads with the one on tetrads (segment 4 versus segments 2,3,4,6). In all cases, this decision of inclusion should be based on the reference label only.

Finally, the retained label pairs are compared to each other and a score is assigned to the evaluation segment, which then gets weighted by the segment's duration. All segment scores of the whole data set are summed together and divided by the total duration of retained segments to arrive at the final result. The pairwise score itself can be calculated according to a number of methods. Obviously, when both labels are the same (taking into account enharmonic variants and the transformation to the previously defined chord and key vocabularies), the score is 1. The remaining question is if, and how, a difference is made between erroneous estimations that are close and those that are completely off.

The case of related keys is reasonably well defined. It is common to assign a part of the score to keys that are a perfect fifth away from, relative or parallel to the reference key. For C major, these are F and G major, A minor, and C minor according to

the rules of the Music Information Retrieval Evaluation Exchange (MIREX)³ audio key detection contest, they get 0.5, 0.3 and 0.2 points respectively. The best scores obtained in MIREX-2014 was 0.8683.

For chords, there is less consensus about how to account for related chords, because the chord dictionary is typically more complex, so many times almost correct estimations do not contribute at all. One option is to consider chords as sets of chromas and to take the precision and the recall of these sets. This is useful to detect over- or under-estimation of the chord cardinality when mixing triads and tetrads, but cannot measure if the root has been estimated correctly. Therefore it can be complemented by a score that only looks at a match between the roots.

Just like for designing a recognition algorithm, the evaluation procedure requires a dictionary on which the music will be projected, and the transformation rules to achieve this. This is used to bring the reference and tested sequence into the same space. So far we have assumed that this evaluation dictionary is the same as the algorithmic dictionary, which is the easiest and most recommended option, but this is not always possible. A notable example is when we want to compare multiple algorithms with different vocabularies to the same ground truth, as is done for the MIREX audio chord estimation task.

To handle these differences, extra rules need to be formulated about which segments should be included in the evaluation and which evaluation dictionary should be used. To this end, a framework for the rigorous definition of evaluation measures is explained in [24]. The accompanying software, as used in MIREX too, is freely available on-line⁴. Naturally, it can also be used for the evaluation of a single algorithm on its own.

When multiple algorithms are compared to each other, it is also important to know to what degree their differences are statistically significant. The method used for MIREX is described in [2], and is also freely available as an R package⁵.

MIREX-CHORD-RESULTS ???

19.6 Further reading

In this chapter, we have described the basic techniques for building the blocks of a chord recognition system. We finish this chapter by providing short descriptions and further readings on possible variations around the techniques used for these blocks: the audio signal representations, the representation of the chord labels and the joint estimation of several musically related parameters.

19.6.1 Alternative audio signal representations

Beat-synchronous chroma/PCP. One common variation is the computation of the chroma/PCP vector in a beat-synchronous way [19]. In this, a beat-tracking algorithm is first used (see Chapter 19.5.1) to estimate the beat positions b_i (where

³http://www.music-ir.org/mirex/wiki/MIREX_HOME

⁴<https://github.com/jpauwels/Mus00Evaluator>

⁵<https://bitbucket.org/jaburgoyne/mirexace>

$i \in \mathbb{N}$ is the beat number). Chroma/PCP are then extracted to represent every beat of a track. This is done by centering the frame positions on the beats. This method allows to make the chroma/PCP tempo-invariant. Since chords often last an integer number of beats (usually 2 or 4), having chroma/PCP attached to beat duration allows to estimate more easily the duration of the chords.

Multi-bands chroma/PCP. Another common variation is to compute separate chroma/PCP vectors for the lower and higher part of the spectrum [13]. Each part is then considered to bring different observations related to chords.

Tonal centroid. Despite their obvious appeal due to the proximity to the theoretical definition of chords, chroma/PCP vectors are not the only type of signal representations that have been tried. Harte et al. [7] proposed a six dimensional vector named *tonal centroid*, which emphasizes the intervals of a third and a fifth because these are the most discriminative for chord recognition. This representation is used in the key-dependent HMMs of [12].

Full spectra. Another option is to keep the full spectrum, instead of reducing it to a single octave chroma/PCP vector. The redundancy present in the different octaves then needs to be dealt with by the mapping to the internal chord representation itself. This is more complicated, but potentially more powerful. Such an approach has been tried in pioneering work [15], before the establishment of the chroma/PCP vector as de facto standard, and has recently seen renewed interest with the advent of deep-learning techniques for neural networks [8].

19.6.2 Alternative representations of the chord labels

Other than changing the signal representation, there are also alternatives for the chord representations. In addition to generative models such as Gaussian distributions and mixtures thereof, discriminative methods can be used as well. A simple frame-based classifier can be implemented as a support vector machine, as in [33]. A discriminative counterpart for a system that can take into account the broader musical context, can be achieved by using a linear-chain conditional random field [3].

19.6.3 Taking into account other musical concepts

Besides the joint estimation with keys, other musical concepts can be included in the modelling of musicological context. Just like with the inclusion of the key, the idea is that there is a dependency between chords and the other notion that can help to narrow down chords to more specific positions and combinations.

Joint chord/meter recognition. A first example is the co-recognition with metric position. Here the premise is that a chord change is more likely to happen at some positions in the measure than at others. An intuitive example would be that it is more likely to change chords on the first beat of a measure than on the second. This dependency can be taken into account both with chords directly [20], as well as in a larger context of chords in a key [13].

Joint chord/bass line recognition. Another musical concept that is intertwined with chords is the bass line. The bass note that is played together with a chord often

gives an indication of the chord itself. Because there are less interfering harmonics of other notes in the lower part of the spectrum, the bass note can also be estimated comparatively easily. The combination of these two qualities makes the bass line a valuable addition to a chord context model, as demonstrated in [30].

Joint chord/key/structure recognition. A final case of including other concepts, is the co-recognition of chords and keys with musical structure. It is based on the idea that certain chord combinations, especially when expressed in a key, are indicative of structural endings. Examples are cadences in classical music or typical turnarounds in jazz and blues. The effect on the chord and key recognition output seems negligible in this case, but it provides an alternative method of structure estimation [21].

Bibliography

- [1] Juan Pablo Bello and Jeremy Pickens. A robust mid-level representation for harmonic content in music signals. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 304–311, 2005.
- [2] John Ashley Burgoyne, W. Bas de Haas, and Johan Pauwels. On comparative statistics for labelling tasks: what can we learn from MIREX ACE 2013? In *Proceedings of the 15th Conference of the International Society for Music Information Retrieval (ISMIR)*, Taipei, Taiwan, October 27–31 2014.
- [3] John Ashley Burgoyne, Laurent Pugin, Corey Kereliuk, and Ichiro Fujinaga. A cross-validated study of modelling strategies for automatic chord recognition in audio. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, pages 251–254, 2007.
- [4] Heng-Tze Cheng, Yi-Hsuan Yang, Yu-Ching Lin, I-Bin Liao, and Homer H. Chen. Automatic chord recognition for music classification and retrieval. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 1505–1508, Hannover, 2008.
- [5] Takuya Fujishima. Realtime chord recognition of musical sound: a system using common lisp music. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 464–467, Beijing, China, 1999.
- [6] Emilia Gómez. Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing, Special Cluster on Computation in Music*, 18(3), 2006.
- [7] Christopher Harte, Mark Sandler, and Martin Gasser. Detecting harmonic change in musical audio. In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, pages 21–26, New York, NY, USA, 2006. ACM.
- [8] Eric J. Humphrey and Juan P. Bello. Rethinking automatic chord recognition with Convolutional Neural Networks. In *Proceedings of the IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 357–362, Boca Raton, FL, 12–15 December 2012.
- [9] Özgür Izmirli. Template based key finding from audio. In *Proceedings of the*

- International Computer Music Conference (ICMC)*, pages 211–214, Barcelona, Spain, 2005.
- [10] Maksim Khadkevich and Maurizio Omologo. Use of hidden Markov models and factored language models for automatic chord recognition. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR)*, pages 561–566, 2009.
 - [11] Carol L. Krumhansl and Edward J. Kessler. Tracing the dynamic changes in perceived tonal organization in a spatial representation of musical keys. *Psychological Review*, 89(4):334–368, July 1982.
 - [12] Kyogu Lee and Malcolm Slaney. Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2):291–301, February 2008.
 - [13] Matthias Mauch and Simon Dixon. Simultaneous estimation of chords and musical context from audio. *IEEE Transactions on Audio, Speech and Language Processing*, 18(6):1280–1289, August 2010.
 - [14] Joshua Morman and Lawrence Rabiner. A system for the automatic segmentation and classification of chord sequences. In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, pages 1–10. ACM, 27 October 2006.
 - [15] S. Hamid Nawab, Salma Abu Ayyash, and Robert Wotiz. Identification of musical chords using constant-Q spectra. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 5, pages 3373–3376, Salt Lake City, UT, USA, 7–11 May 2001. IEEE.
 - [16] Laurent Oudre, Yves Grenier, and Cédric Févotte. Chord recognition using measures of fit, chord templates and filtering methods. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, USA, 2009.
 - [17] Hélène Papadopoulos. *Joint Estimation of Musical Content Information*. PhD thesis, University Paris VI, 2010.
 - [18] Hélène Papadopoulos and Geoffroy Peeters. Large-scale study of chord estimation algorithms based on chroma representation. Bordeaux, France, 2007.
 - [19] Hélène Papadopoulos and Geoffroy Peeters. Joint estimation of chords and downbeats from an audio signal. 19(1):138 – 152, January 2010.
 - [20] Hélène Papadopoulos and Geoffroy Peeters. Joint estimation of chords and downbeats from an audio signal. *IEEE Transactions on Audio, Speech and Language Processing*, 19(1):138–152, January 2011.
 - [21] Johan Pauwels, Florian Kaiser, and Geoffroy Peeters. Combining harmony-based and novelty-based approaches for structural segmentation. In *Proceedings of the 14th Conference of the International Society for Music Information Retrieval (ISMIR)*, pages 138–143, Curitiba, Brazil, 2013.
 - [22] Johan Pauwels and Jean-Pierre Martens. Combining musicological knowledge

- about chords and keys in a simultaneous chord and local key estimation system. *Journal of New Music Research*, 43(3):318–330, 2014.
- [23] Johan Pauwels, Jean-Pierre Martens, and Marc Leman. Modeling musicological information as trigrams in a system for simultaneous chord and local key extraction. In *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, Beijing, China, September 2011.
 - [24] Johan Pauwels and Geoffroy Peeters. Evaluating automatically estimated chord sequences. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013.
 - [25] Geoffroy Peeters. Musical key estimation of audio signal based on hidden Markov modelling of chroma vectors. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, pages 127–131, Montreal, Quebec, Canada, September 18–20 2006.
 - [26] Thomas Rocher, Matthias Robine, Pierre Hanna, and Myriam Desainte-Catherine. A survey of chord distances with comparison for chord analysis. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 187–190, Ann Arbor, MI, USA, 2010. MPublishing, University of Michigan Library.
 - [27] Matti P. Ryyänänen and Anssi P. Klapuri. Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, 32(3):72–86, Fall 2008.
 - [28] Ricardo Scholz, Emmanuel Vincent, and Frédéric Bimbot. Robust modeling of musical chord sequences using probabilistic n-grams. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 53–56, 2009.
 - [29] Roger N. Shepard. Circularity in judgements of relative pitch. *Journal of the Acoustical Society of America*, 36:2346–2353, 1964.
 - [30] Kouhei Sumi, Katsutoshi Itoyama, Kazuyoshi Yoshii, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Automatic chord recognition based on probabilistic integration of chord transition and bass pitch estimation. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, pages 39–44, 2008.
 - [31] David Temperley. What’s key for key? The Krumhansl-Schmuckler key-finding algorithm reconsidered. *Music Perception*, 17(1):65–100, 1999.
 - [32] Gregory H. Wakefield. Mathematical representation of joint time-chroma distributions. In *Proc. of SPIE conference on Advanced Signal Processing Algorithms, Architecture and Implementations*, pages 637–645, Denver, Colorado, USA, 1999.
 - [33] Adrian Weller, Daniel Ellis, and Tony Jebara. Structured prediction models for chord transcription of music audio. In *Proceedings of the IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 590 – 595, Miami Beach, FL, December 13–15 2009.

- [34] Yongwei Zhu and Mohan S. Kankanhalli. Precise pitch profile feature extraction from musical audio for key detection. *IEEE Transactions on Multimedia*, 8(3):575 – 584, June 2006.