Master thesis on Sound and Music Computing

Universitat Pompeu Fabra

Towards a new compatibility measure for harmonic EDM mixing

Gabriel Bibbó Frau

Supervisor: Ángel Faraldo

August 2021

Master thesis on Sound and Music Computing

Universitat Pompeu Fabra

Towards a new compatibility measure for harmonic EDM mixing

Gabriel Bibbó Frau

Supervisor: Ángel Faraldo

August 2021

Contents

Acknowledgement

I would like to begin by thanking Martín Rocamora, a professor from my undergraduate career who showed me the existence of this academic field, where engineering and music converge.

To all the SMC students, for exploring new forms of solidarity and companionship in this particular year defined by Covid.

I also want to thank my tutor for having trusted in the project from the beginning and for providing me with advice in difficult times. To all the researchers who have so kindly made themselves available: Miguel Pérez Fernández, Frederic Font, Gilberto Bernardes, António Ramires, Alia Ahmed Morsi, Diego Frau, Luciano Garrido, Bruno Chiappetta and Pedro Ramoneda.

Abstract

We explore the applicability of a new harmonic compatibility (HC) measure in the field of DJ's harmonic mixing. We present a system that complements harmonic mixing by calculating the HC between the tracks of a user-defined music collection. The user must define a target track for which the calculation is to be made, and obtains the HC values with respect to each track in the collection, expressed as a percentage. The system also calculates a pitch transposition interval for each candidate track that, if applied, maximizes the HC with respect to the target track.

To analyse the tracks, they are first source-separated to remove the percussive elements, with no melodic information. A single chroma vector is computed as the mean of the NNLSChroma from each frame, that then is converted into a Tonal Interval Vector (TIV). The HC is calculated as the *small-scale* measure score between the pairs TIVs of candidate and target track.

The system is implemented with a graphical user interface (GUI) that allows running on a parallel window next to the DJ software of choice, allowing the program to be incorporated into the DJ's live performances. The implementation allows real-time calculation of the HC, each time a new target track is selected.

The suggestions of the system are evaluated in an experiment where musically trained participants are asked about their preference between pairwise comparisons of mixes.

Keywords: DJ; Harmonic compatibility; Harmonic mixing; Tonal Interval Vector (TIV); Pitch Transposition; Interface

Chapter 1

Introduction

1.1 Context

Discos, clubs and raves have been focal points for the development of today's Electronic Dance Music (EDM). The 1970s witnessed essential elements of musical practice such as beat-matching (using variable-speed turntables to bring the tempi, beats, measures and phrases of records into perfect alignment) and overlapping records to create a constantly sounding, never interrupted mix.

The increasingly electronic modes of production gave rise to the first dance-music styles centered exclusively around electronic instrumentation, ushering in a trend that has subsequently been regarded as a defining feature. Faraldo [1] eloquently expresses this paradigm shift, emphasising that the electronic aspect of this type of music establishes a clear boundary with other popular music styles, mostly vocal and guitar-centred, "bringing into play a new type of musician (the DJ), a new instrument (the turntable), and a new notion of musical skill, consisting in playing records instead of notes, in combining existing musics to arrive at a new sound, rather that composing with a previously defined palette of notes and chords."

Musical instruments gradually abandoned the intervention of loutiers in favour of professionals trained in artificial intelligence, computer engineering, music technology, and information retrieval. In four decades of progress, today's digital era of

DJ-mixing opened DJing to a massive range of users and enabled new technical possibilities in music creation and remixing. The industry-leading DJ-software tools (e.g. Native Instruments Traktor Pro 3.2 [2], Mixed in Key [3] and Pioneer KEY SYNC [4]) now offers users of all technical abilities the opportunity to rapidly and easily create DJ mixes out of their personal music collections, or those stored online.

1.2 Common DJ practices for harmonic mixing

"A DJ is a person who mixes existing music records in a seamless way to create a continuous stream of music. With a *seamless* mix, we understand a mix that blends songs such that the resulting music is uninterrupted (no silences in between songs), and such that the music is structurally coherent on *beat*, *downbeat*, and *phrase* level. Additionally, successive songs should be "compatible" to some extent with respect to their harmonic, rhythmic, and/or timbrical properties. In essence, a seamless mix flows from song to song such that the transition between those songs appears to be a part of the music itself, and where it consequently is often hard to tell where one song ends and the other begins." Vande [5]

The primary challenge in music mixing is aligning two or more music tracks both in their temporal and spectral dimension. Speaking in more musical terminology, the pieces have to fit both in tempo and harmony [6].

Knowing the structural and rhythmical properties of the music is extremely important for a DJ. The beat and tempo positions, when combined with high-quality audio time-stretching, are used to align the tracks in tempo, a technique called *beatmatch* in music.

While beat-matching has been explored quite extensively, the automatic harmonic alignment turns out to be a more challenging computational task [6]. Professional DJs use a technique for *harmonic mixing* based on key notation, where successive songs of the DJ set have the same or a related key. That ensures that they fit together harmonically and reduces dissonance when playing two songs at the same time. Harmonic mixing based on the musical key is the most common practice in

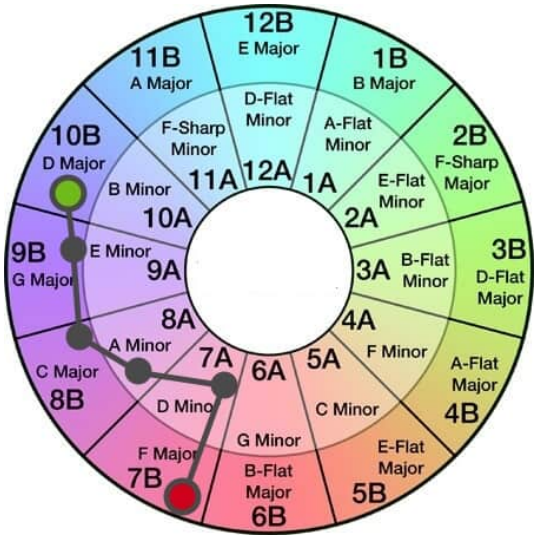


Figure 1: *Camelot Wheel, from Mixed in Key [3]. The grey path shows an example of harmonic transitions between tracks in a DJ set.*

EDM, although recent work [6] questions the use of this technique in specific cases (e.g. for music outside the major/minor scale framework).

There is one major and one relative minor key in Western music for each of the 12 equal-tempered enharmonic notes, totaling 24 keys. Usually, DJs use software to automatically estimate the key of each song in a pre-processing step. They are stored on disk, allowing fast retrieval. Employing the displayed key value, the user can choose tracks to mix that fit according to the theory of the circle of fifths. Adhering to well-known relationships within the circle of fifths, the allowed key changes for the next song are changing to the *relative* key of the current song's key or going a perfect fifth down or up (i.e., changing to the *subdominant* or *dominant* key) [5]. The Camelot Wheel (see figure 1), patented by Mixed in Key [3], is a simple representation of the circle of fifths commonly used by non-musicians to create harmonic mixes. The combinations between different keys that result in harmonic mixes are limited. But by changing the key of the tracks, one could "force" the harmonic alignment on two pieces of music that would otherwise be incompatible. The ability to change the key and transpose audio by some semitones independent of its temporal structure is usually referred as audio pitch-shifting functionality.

Unlike existing commercial DJ-mixing software [3], which determines compatible

matches between songs via key estimation, as reviewed in Section 2.1.1, we can identify two more automatic harmonic mixing techniques: chroma vectors similarity and (minimal) psychoacoustic dissonance. Chroma vector similarity measures the compatibility of two given audio samples as the cosine distance of their chroma vectors representations [7], [8], [9]. Psychoacoustic dissonance methods are built around the measurement of musical consonance according to psychoacoustic models of roughness and pitch commonality, attempting to minimize the combined level of sensory roughness of pitch-shifted versions of overlapping musical audio [10].

1.3 Motivation

This project arises from my interest in studying the tools for harmonic mixing available in the industry. Usually these ideas are approached by developing a key-detection algorithm. However, after further research, we have discovered that it makes sense to create a new metric to reflect harmonic compatibility between tracks.

Even though the key-based matching approach has shown to correlate well with user enjoyment, some authors argue that it has several important limitations and erroneously represents certain aspects of harmonic compatibility [11]. First, the primary issue is that the key estimation itself might be error-prone. Then, it is not clear how listeners might perceive the mix of two tracks since a global property like the key does not provide lower-level information about other elements of the musical composition [12]. Furthermore, this method makes it impossible to determine harmonic compatibility between different tracks sharing the same key, thus motivating a metric below the key level. In addition, key detection algorithms usually return the most prevalent key throughout the track's duration without giving the user the chance to identify the best points in time for mixing. Finally, if the pitch-shifting functionality is used to transpose the musical key, it is essential to consider the quantization effect of only comparing whole semitone shifts. Not considering fine-scale tuning between songs that still share the same key could lead to highly dissonant mistuned mixes [12].

The compatibility based on the distance of the chroma vectors has the quality of being very effective at finding matches between two audio tracks, but if the tracks have different pitch configurations, this method is unable to represent the perceptual results of the mix [13]. Regarding psychoacoustic models, while they show improved performance over the previous approaches, some studies reported that they violate the harmonic principles of Western Music and decrease the performance if the spectral content of the samples do not overlap [14].

To address this lack of perceptual analysis of chroma in HC models, new methods based on perceptual relatedness and consonance have shown to give promising results [13]. The perceptually-motivated Tonal Interval Vector (TIV) feature computes the small- and large-scale HC between a collection of audio tracks. Pérez evaluated the advantages and drawbacks of the existing algorithms for HC using a database of EDM loops of 32 beats. He found that the TIVs implementations are usually preferred by most listeners [15].

1.4 Objectives

1. Implement an open-source version of State-of-the-Art algorithms for HC measure between two EDM tracks (one target, one candidate).

2. Present the harmonic mixing suggestions in a format that is compatible with commercial software, that include the HC estimated value, the suggested pitch transposition interval, and the resulting HC after pitch transposing.

3. Implement a simple, easy-to-use GUI to run the algorithm, visualize the suggestions and allow to select a music collection folder and a target track. Achieve a real-time execution.

4. Assess user's preferences regarding HC. Compare with the algorithm's performance and re-adjust it to match user preferences. Evaluate the quality of the suggestions provided by the algorithm.

1.5 Structure of the Report

Besides this Introduction, this Master thesis dissertation contains 5 Chapters. Chapter 2 reviews the existing tools that assist in harmonic mixing, presents the state-of-the-art technique used to calculate HC, and discusses some methods used to evaluate its performance. Chapter 3 explains the steps used in the iterative development process, from the starting point, the dataset and metrics created to evaluate the modifications incorporated, the different strategies considered for the system architecture, and the performance results obtained for each of the implementations. Chapter 4 describes in detail the final architecture of the system, and present the graphical user interface. Chapter 5 exposes the evaluation process involving the generation of audio samples, the survey, the statistical process used to analyze the results, and a comparative table with the final results of the different implementations. Finally, in Chapter 6, we outline our conclusions, considering the utility of this tool for DJs, and point out new directions for research in HC.

Chapter 2

State of the art in Harmonic Mixing

Traditionally, tools that assist DJs in harmonic mixing have focused on key notation [3], [4], [8], [16], [17], [18], [19], [20], [21]. However, there have been some other approaches that seek to represent HC in terms of a single linear, bounded variable [22], [23], [24], [25]. In recent years, the emergence of a new feature (the TIV [13], [26]) that allows to describe the harmonic characteristics of music taking into account perceptual aspects, has pushed the limits of existing HC algorithms [15]. The effectiveness of using these descriptors in electronic dance music (EDM) has also been tested with promising results [23]. However, these tools belong to the academic field, and are not necessarily adapted to a professional DJ's day-to-day work.

In this section, we review the tools that DJs have at their disposal to be assisted in harmonic mixing. Then we understand the characteristics of the TIV and highlight exciting results that validate the use of this descriptor to calculate the HC. Finally, as the quality of a song-to-song transition are highly subjective, we rely on user studies that assess the perceptual validity to evaluate the performance of our systems.

2.1 Tools available for harmonic mixing

Some implementations [5][27] can generate seamless music mixes using songs from a given library much like a human DJ does, being totally automated DJ systems. In these cases, interaction with the machine is limited to specifying the path of the desired music collection.

However, many of the implemented systems we review below, seek to generate a space that allows for interactions between the user and machine. In the case of computer-assisted harmonic mixing, this interaction aims to give the user the freedom to make the ultimate decisions regarding the mix, whilst the machine simultaneously feeds back information on the harmonic relations across the tracks.

We can classify these implementations under two criteria: commercial vs. open-source and key notation-based vs. HC-based. Our implementation aims to be an open-source alternative based on HC, adapted to be easily integrated with commercial systems based on key notation, such as Traktor Pro 3[2], which adopts a one-to-many mapping strategy between a user-defined track and a ranked list of compatible tracks to show the results to the user. For this reason, the objective is to replicate the usability of existing commercial interfaces.

2.1.1 Commercial products

It is worth considering some essential milestones in the history of DJ products concerning harmonic mixing. In 2006 the German company Zplane [16] launched an interface for Windows that implements its own automatic key detection algorithm. A year later, the company Mixed In Key [3] patented its algorithm based on the previous one. They continued its development until the latest version in 2017. In the meantime, other programs for key tagging appeared. However, Mixed In Key is considered the best performing software for key notation [28] and has helped define the state-of-the-art.

In 2000, the German company Native Instruments [17] launched Traktor, a DJ soft-

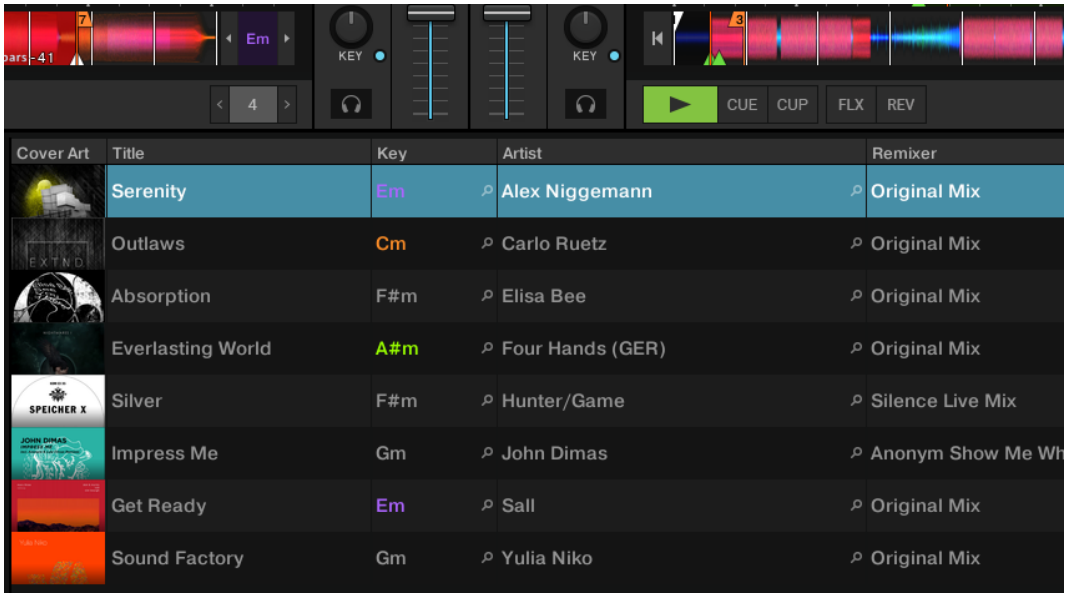


Figure 2: *Traktor[2] interface screenshot. The software suggests potential tracks to mix by highlighting them with colours. In this example, the track being played is in E minor. It also suggests with a different colour the tracks that are potentially harmonically compatible if a pitch transposition of 1 semitone is applied. One candidate is A-sharp minor, which, when lowered by 1 semitone, results in A minor, compatible with E minor. Similarly, C minor results in B minor when lowered by one semitone. The harmonic relationships can be deducted from figure 1.*

ware for live mixing. Since then, the software has evolved, integrating methods to alter both dimensions independently using time stretching and pitch shifting techniques. In Traktor, the built-in synchronization option offers automatic temporal fitting, but there is no functionality to automate the harmonic alignment [6]. However, changing the pitch can be manually done. The latest versions of Traktor (v3.2 - launched in 2019, and onwards) [2], incorporate excellent features for visualizing the tracks' keys while playing live, as can be seen in figure 2. The interface allows the user to highlight all the key-compatible tracks within a folder. Additionally, it can also highlight the potentially compatible tracks if the user applies a transposition of one semitone. In the case of pitch transposing, the program displays the resulting key. That version of Traktor also implemented Elastique Pitch v1.3.3 [29], an algorithm for pitch shifting developed by Zplane, that reduces distortion.

In 2015 Serato [30], a music software company from New Zealand, launched KEY SYNC as a new feature for Serato DJ 1.8 [18]. KEY SYNC allows the synchroniza-

tion of cued tracks to the live track. When pressing the KEY SYNC button on the deck that needs synchronizing, the software matches the key to the track that is playing live. The display shows the new key and how many semitones it is adjusted by. That was a significant step towards the automation of harmonic mixing.

The most popular DJ hardware brand used to equip nightclub booths around the world, Pioneer Corporation [31], launched in 2019 the DDJ-1000 [19] a four-deck controller for Serato. They introduced KEY SYNC [4] on their controller, but as the DDJ-1000 is a personal controller, it didn't have a significant impact on mixing methods. However, in 2020 they released the CDJ-3000 [20] digital console to replace the CDJ-2000 Nexus, which incorporates the KEY SYNC button. This way, they brought harmonic mixing into the DJ booth.

While most companies were developing products to assist in harmonic mixing based on key notation, in 2014, a product with a different approach appeared on the market. That year Mixed In Key launched Mashup2 [22], a software of little transcendence next to the products mentioned above, which changed the shape of music. Mashup2 was designed to find tracks that sound good together, *beatmatch* them instantly, and produce a fast *mashup* (equivalent to a remix). In addition, this program implemented an exciting functionality that computed the HC between two tracks expressed according to how well they will sound when mixed, such that: *A value of 100 means the mashup will be harmonic. A value of 0 means it will be dissonant.*

2.1.2 Open-source implementations

Almost simultaneously, in 2014, the IEEE published the work of Davies, M. E. et al. [8], AutoMashUpper. They present a system for making multi-song music mashups. Central to the system is a measure of *mashability* calculated between phrase sections of an input song and songs in a music collection. They define mashability in terms of harmonic and rhythmic similarity and a measure of spectral balance. Their approach to harmonic similarity is based on key notation. Interestingly, they introduce, for the first time, the notion of how well elements of songs can be made to fit together

using key transposition and tempo modification, rather than just based on their unaltered properties. In this way, they created a user interface that allows altering the songs' properties to maximize their perceptual compatibility. Additionally, the user can define the maximum allowed range for key transposition.

As mentioned latter in Section 2.2, in 2015, Bernardes, G. et al. improved state-of-the-art measurement of HC by introducing the TIV. In the 2017 publication [23], the authors proposed an interface design that adopts a many-to-many mapping strategy. Each audio track in a collection is represented as a graphical element in a navigable 2-dimensional interface that offers a global view of the compatibilities, as shown in figure 3. Distances among these elements indicate HC, and the additional graphic variables of these elements, such as color and shape, indicate rhythmic and spectral information relevant to mashup creation. This interface design aims to promote an overview of the harmonic relations between all the tracks within a collection. Their main contributions are metrics for computing the HC between musical audio tracks at small- and large-scale structural levels. Small-scale HC results from the combination of dissonance and perceptual relatedness indicators from the Tonal Interval Space to assist users in finding good local alignments between mixed tracks. Large-scale harmonic compatibility relies on key estimates and aims to assist users in planning the global harmonic structure of a mix.

In 2018 Maçãs, C. et al. presented an assistive tool for music mashup creation from large music collections: MixMash [24],[25], intended to extend the aforementioned methods. They introduce novel degrees of harmonic, rhythmic, spectral, and timbral similarity metrics. Furthermore, they create a graphic model that revises and improves some interface design limitations identified in the former model software implementation.

Finally, I would like to mention KeyFinder [21], an open-source implementation for key estimation originally developed by Ibrahim Sha'ath in 2011 as part of his MSc in Computer Science. For years it was the free alternative tool for key estimation that many DJs (like me) used for key-tagging music collections. With support for Windows and OSX users, his approach based on classification against tone profiles

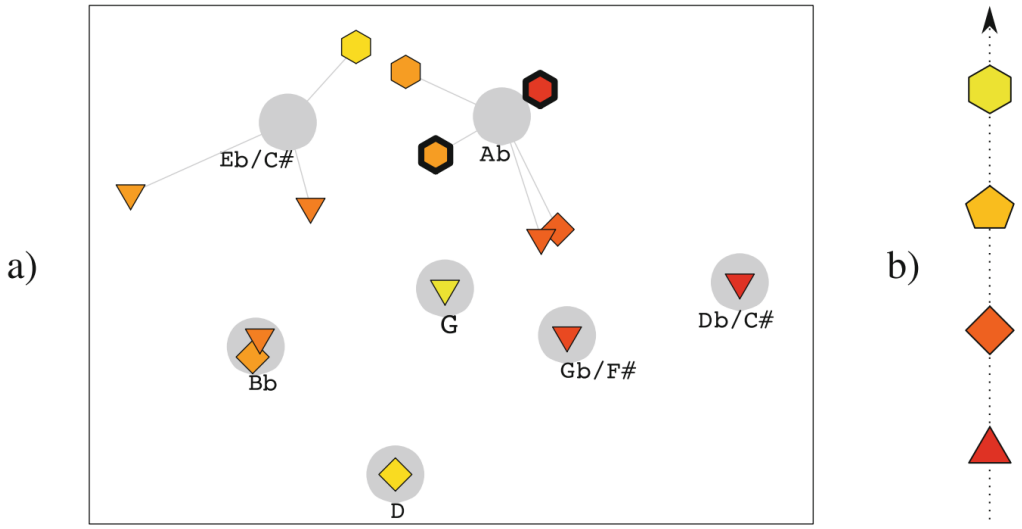


Figure 3: *Interface of "A Hierarchical Harmonic Mixing Method", by Bernardes, G. et al. [23]. (a) Tracks in a collection are visualized distanced according to their HC. Audio tracks are represented with polygons. Circles represent key centers. The distances between polygons indicate small-scale HC and the links from circles the large-scale HC. The selected files currently playing are indicated with polygons with thick outlines. (b) The hierarchical order reference.*

achieves an accuracy performance[1] that is slightly under Mixed In Key.

2.2 Harmonic Compatibility (HC) measure

We can understand the harmonic compatibility (HC) as the degree of consonance generated when two pieces of music are blended. Underlying much Western music, a high HC score is usually perceived as pleasant, stable, and positively valenced.

The computational models that predict the HC can be separated into two main groups: Spectral similarity and Dissonance/consonance based measures [15].

Spectral similarity measures try to find a candidate track with similar characteristics to the target track, making it more suitable to find mixes where the components have similar features. Chromagrams, pitch-spectrograms, TIVs, and other similar features are used as high-level representations of music to generalize the harmonic content of the sounds. That means that if we want the most compatible audio excerpt for

[1] KeyFinder accuracy comparison 2016: `http://ibrahimshaath.co.uk/keyfinder/comparison.pdf`

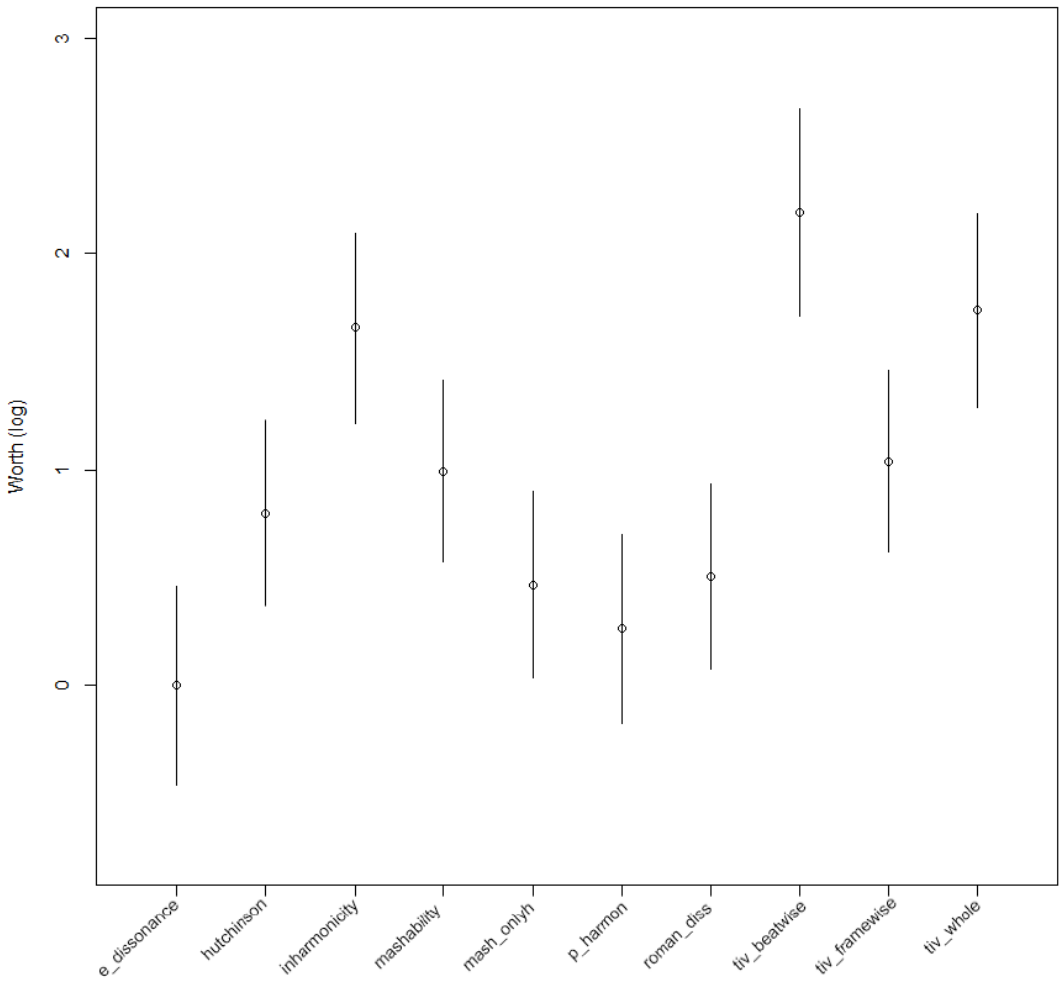


Figure 4: *Different HC models compared under the same framework by Pérez [15]. Algorithms reference from left to right: Plompt Levelt, Hutchinson Knopoff, Gebhardt et al., Inharmonicity, Harrison & Pearce Harmonicity, AutoMashupper, AutoMashupper no spectral balance, TIV Framewise, TIV Beatwise, TIV Whole.*

an Augmented triad, the result will be close or precisely another Augmented triad.

On the other hand, Dissonance/consonance measures try to find a candidate track such that when mixed with the target track, it produces as much consonance as possible. As they consider the pleasantness, beauty, or attractiveness of the resulting mix of the two tracks, this can lead to different results. In this case, for an Augmented triad, the most consistent result could be just a dyad containing the root and the octave.

In that regard, Pérez [15] did an interesting compendium of the most referenced computational models for HC. He compared ten different implementations under the

same framework, as seen on the graph 4. For the specific case of TIV (a type of Spectral Similarity measure that uses high-level representations of music, as explained below in more detail), he compared three different implementations by varying the audio section used to calculate the feature:

1. "Framewise: A TIV is calculated for every frame of the target and the candidate song. We calculate the small scale compatibility score between the simultaneous frames of candidate and target song. The final score is the mean small scale compatibility across all frames." TIV-framewise shape: [6 x number of frames] complex numbers.

2. "Beatwise: Similar to framewise but now the time resolution is the beat. For every beat a chroma vector is extracted and transformed to TIV. The mean is taken across the small scale compatibility score given by beatwise-TIVs of target and candidate song." TIV-beatwise shape: [6 x number of beats] complex numbers.

3. "Whole: A single TIV for an audio excerpt is taken and the small-scale measure score is calculated between the pairs TIVs of candidate and target song." TIV-whole shape: [6 x 1] complex numbers.

Pérez concluded that the *TIV-beatwise* feature gives the best results in performance in the desired task, which makes sense as the harmonic changes of this type of music typically occur on beats [32].

2.2.1 Tonal Interval Vector (TIV)

The Tonal Interval Space represents many multi-level pitch configurations (i.e., human perceptions of pitches, chords, keys, and music theory principles) on a topological space. Those configurations are mapped in space as 12-dimensional TIVs. The Euclidean distance between the TIVs represents the perceptual proximity among pitch configurations and their level of consonance.

Proposed originally in 2015 by Gilberto Bernardes et al. in [26] and extensively described in [13], the Tonal Interval Space is a new tonal pitch space inspired by the Tonnetz[2], Chew's [33] Spiral Array, and Harte et al.'s [34] 6-dimensional (6-D) Tonal Centroid Space. It constitutes a series of controlled distortions of the chroma space calculated as the weighted Discrete Fourier Transform (DFT) of normalized 12-element chroma vectors. The proximity of multi-level pitch configurations and their level of consonance can be measured. It has been applied to various music theory and practice problems, such as key finding from symbolic music notation and musical audio [35], generation of harmonic progressions [36], harmonisation of user-given melodies [35], and as a control surface for a consonance-based MIDI keyboard pedal.

One novelty introduced by this space with remaining Fourier spaces was the combined use of the six coefficients in a TIV, *T(k)*. Moreover, the perceptual basis of the space is guaranteed by weighting each coefficient by empirical ratings of dyad consonance, $w_a(k)$. In this sense, *T(k)* allows the representation of hierarchical or multi-level pitch due to the imposed L_1 norm, as in equation 2.1.

$$T(k) = w_a(k) \sum_{n=0}^{N-1} \bar{c}(n) e^{\frac{-j2\pi kn}{N}}, k \in \mathbb{Z} \texttt{ with } \bar{c} = \frac{c(n)}{\sum_{n=0}^{N-1} c(n)} \tag{2.1}$$

where $N = 12$ is the dimension of the chroma vector $c(n)$ and, since the remaining coefficients are symmetric, k is set to $1 \leq k \leq 6$. The weights, $w_a(k) = \{3, 8, 11.5, 15, 14.5, 7.5\}$, adjust the contribution of each dimension k of the space to comply with empirical ratings of dyad consonance as summarised in [37]. $w_a(k)$ accounts for the harmonic structure of musical audio driven from an average spectrum of orchestral instruments [23].

Equation 2.2 computes two indicators of musical audio dissonance, D_{tc}, and the perceptual relatedness, R_{tc}, from the space as distance metrics. $T_t(k)$ and $T_c(k)$, represent the two overlapping audio tracks, the target t and the candidate c. Vari-

[2]Tonnetz: `https://en.wikipedia.org/wiki/Tonnetz`

ables a_t and a_c are the amplitudes of $T_t(k)$ and $T_c(k)$.

$$D_{tc} = 1 - \left(\frac{\|a_t T_t(k) + a_c T_c(k)\|}{a_t + a_c \|w_a(k)\|} \right), \quad R_{tc} = \sqrt{\sum_{k=1}^{M} |T_t(k) - T_c(k)|^2} \tag{2.2}$$

The small-scale harmonic compatibility metric, HC, is then computed in equation 2.3 as the product of the two indicators $\bar{R}$ and $\bar{D}$, that are R and D scaled to the range $\{0, 1\} \in \mathbb{R}$.

$$HC_{t,c} = \bar{R}_{t,c}.\bar{D}_{t,c} \tag{2.3}$$

In the Tonal Interval Space, transpositions by p semitones result in rotations of *T(k)* by $\varphi(p) = \frac{-j2\pi kn}{N}$ radians. Transposition of *c(n)*, which by definition are circular in the chroma domain, are represented as *c(n-p)*. So, transposing D by $p = 3$ results in F and by $p = 12$ results in D. Using the properties of the Fourier transform, the pair $c(n) \overset{\mathscr{F}}{\Longrightarrow} T(k)e^{\frac{-j2\pi k}{N}p}$, where $\mathscr{F}$ represents the DFT. Denoting $T_{\mathrm{p}}(k)$ as the TIV of $c(n-p)$ results in equation 2.4

$$T_{\mathrm{p}}(k) = |T(k)|e^{\frac{-j2\pi(k+p)}{N}} \tag{2.4}$$

Hence, any transposition $c(n-p)$ resulting in $T_{\mathrm{p}}(k)$ has the same magnitude $|T(k)|$ as the original sequence $c(n)$ and a linear phase component $e^{\frac{-j2\pi k}{N}p}$. That means that to transpose the key of TIVs, we only need to shift the phase, a process that is computationally light and can be executed in real-time.

In 2020 Ramíres et al. [38] presented `TIV.lib`[3], an open-source library implemented in Python for the content-based tonal description of musical audio signals, using only `Numpy` and `Scipy` as dependencies. The `TIV.lib` takes as input the pitch class profile (PCP) [4] of the audio signal from which multiple instantaneous and global

[3] TIV.lib: `https://github.com/aframires/TIVlib/`

[4] *Pitch Class Profiles (PCP) are vectors of low-level instantaneous features, representing the intensity of each of the twelve semitones of the tonal scale. They are largely used in all applications*

representations, descriptors and metrics for characterizing the tonal content can be computed - e.g., harmonic change, dissonance, diatonicity, and musical key.

2.3 Evaluation in audio comparison

Measuring harmonic compatibility is a highly subjective task, which makes it challenging to evaluate. Not much work has been done in this area, lacking a "Ground Truth" against which to test the results. However, references to different tests based on audio comparison can be found.

Munson has described [40] ABX tests, designed to identify detectable differences between them by comparing two audio choices. Users are presented with two benchmarks (A and B) and a third example (X) which they must identify as either A or B.

Another example can be found in Automashupper[8], where the authors presented the target song to the participants and then three different mixes for that song. The song used for each of those mixes was the one with the highest mashability score (equivalent to an HC measure), the one with the score closest to the mean mashability score for that target song, and the lowest mashability score. Participants were asked to vote on how much they enjoyed the mix on a scale of 1 to 10. Then this enjoyment mark was used to find the correlation between the Automashupper mashability score and the rating given by the user, using the three mixes mentioned before. The mixes used were 32 beats long, that is a trade-off between short 8 beats with a high likelihood of harmonic similarity, and a long piece of 64 beats that can make users feel tired during the experiments.

Gebhardt & Davies did an interesting work [12] centered on the use of psychoacoustic models of roughness and pitch commonality to identify an optimal harmonic alignment between different pieces of music across a wide range of possible pitch

involving harmonic content, especially chord recognizers, tonality estimators and key detectors. The main advantages of the PCPs are the simplicity of calculation, the concision of the harmonic information and the power to unify various dispositions of a single chord class. Although the main steps of the PCP calculation are very similar over all the scientific literature, some implementation details may vary. Cabral, G. et al., 2005 [39]

shifts. In their experiment, they compared the output of different configurations of their consonance-based mixing approach with the suggestions derived from the DJ-mixing software Traktor[2]. Therefore, the final comparison is not between different rankings of the same algorithms but the best results from different algorithms, allowing to find which one performs better through different examples. Furthermore, they asked two different rankings to musically trained participants: the consonance and the pleasantness of the mixes. They concluded that their models could improve pitch transposition suggestions compared to commercial systems.

In his work [15], Pérez compared the output of different algorithms to compute the HC to determine which produces the best results. His evaluation is a modified version of the ABX tests. First, he presented one reference (the target audio) and then two possible mixes, chosen using different algorithms. Finally, the user had to decide which mix sounded the best.

2.4 Limitations of existing work

From 2006 [16] to date [20], the electronic music industry has been developing computational methods that enhance the DJ experience in harmonic mixing, indicating a genuine interest in pushing the current boundaries. However, the focus has always been on key notation to create a harmonic mix based on the rules of the circle of fifths (see section 2.1.1). There are sufficient reasons [11], [12], [13] to consider that key notation alone is not the best approach. Evidence [12] shows other models achieve better results in harmonic mixing based on HC. However, these techniques are not yet used by the industry.

Among the techniques for calculating HC, the one that has shown the best results are the ones based on TIVs [15]. After being first described in 2015 [12], few implementations of TIV have appeared for HC calculation. Fortunately, a Python library [38] published in 2020 standardizes the calculation of TIV and HC from the chroma vectors (or PCP) of audio tracks. However, there is still insufficient evidence on the optimal calculation of the chroma vectors for this application. From Perez's results

[15], we can infer that the *TIV beatwise* is the best implementation in terms of temporal resolution. However, the signal pre-processing steps before the calculation of the chroma vectors are not specified.

Maçãs, C. et al. created a complete system [23], [24], [25] that analyses the tracks within a folder and represent their harmonic relationships in a graphical interface. However, the system displays all this information on a static 2D map (see figure 3) with a many-to-many technique (the harmonic relationships of all the tracks are plotted against each other). That makes the system hard to integrate into traditional DJ software (see figure 2), which works with a one-to-many strategy (a target track is defined, and the harmonic relationships between this track and the other tracks within the selected music folder are displayed).

The proposal by Gebhardt Davies to express HC as a function of two variables is interesting. However, given the application that our system will have, it is advisable that HC be expressed with as few variables as possible. Considering that our system is intended to be included in DJ software, which currently already represents a lot of metadata about the tracks, the amount of information displayed should be simplified.

Traditional methods for harmonic mixing, based on key notation, are easier to evaluate, as it is possible to have an objective database against which to test the system's predictions. However, HC is an entirely subjective parameter, and its evaluation is not without complications. Furthermore, there is not much work evaluating the performance of a system that calculates HC. And of the few papers that do exist, we have some concerns that prevent us from replicating their procedure:

The most direct reference to the evaluation of our work is found in the Automashapper experiment. Participants were presented with a target track and then three different mixes for that track. We consider that participants may have preferred one of the mixes based on aspects other than necessarily HC. It is important that in this type of experiment, the options presented to the participants are equivalent from all points of view except HC. In other words, the different variables that can influence the participant's decision making should be neutralised in all three clips.

In addition to this, participants were asked to rate the mixes on a scale of zero to ten. We believe that requiring participants to be so precise with their ratings does not add relevant information, as they are unlikely capable of noticing a one-point difference. Moreover, such a broad scale may stress the participant after several examples.

Another reference for our work is the experiment conducted by Miguel Pérez, where he compared the performance of different HC estimation algorithms. In this case, the metric he used for his experiment was intended to allow comparison between different implementations within the same research work. However, his results do not allow to compare the performance of his system with the implementations of other researchers. If we want to compare two experiments carried out on different population samples, both must have the theoretical support that allows us to generalise the results obtained.

For this reason, we believe that there is a gap in HC evaluations, and more work needs to be done on this topic in order to create a standard evaluation strategy that is reliable and allows the comparison of systems from different researchers.

Chapter 3

Optimal TIV retrieval method design

3.1 Introduction

As described in more detail in Chapter 4, the Harmonic Mix System (HMS) we propose in this master thesis is composed of several modules. The Data Management Module (DMM) and the Graphical User Interface (GUI) are a practical case of a complete system implementation. However, the core module of this system is the one in charge of extracting the features of the audio tracks selected by the user, named Feature Extractor Module (FEM). The relevance of the FEM in the context of our research is driven by the fact that we are experimenting with the latest developments for the calculation of harmonic compatibility (HC), which gives HMS the potential to complement and improve the current tools used by DJs for harmonic mixing.

In this chapter we first review the starting point based on Pérez's research [15] as an antecedent, for which we try to understand and adapt his conclusions to the context of our work. We then describe the database we developed to evaluate the module. We present our first implementation based on Pérez's one. We continue by defining the equations used to evaluate the successive implementations. Referring to previous papers, we present different strategies considered to improve the computation of the HC. Finally, we mention limitations as well as possible improvements of the methodology, and review our main contributions.

3.2 Initial requirements

The experiments of Pérez were conducted on a dataset of audio loops 32-beats long (extracted from Looperman [1]) that were labelled with their BPM (beats per minute). However, unlike Pérez, we are interested in that our system performs the feature extraction over the original audio tracks. In this sense, to extract a representative fragment of exactly 32 beats from each track, the BPM and the section of the track from which the fragment is to be extracted must be known. As mentioned in section 2.2, these tasks are commonly referred to in the literature as *beat and downbeat tracking* and *structural segmentation*.

We must therefore know what criteria should be used to choose the loops of the track or, in other words, what is the most representative temporal position of each track to calculate the HC. To answer this question, we must first understand the criteria used by the DJ to overlap the tracks on time.

Formally, we identify three sections on an EDM track: the intro (the initial section), the core (the central section), and the outro (the last section). We refer by "switch-in" to the point in time where the intro ends and the core begins. Similarly, by "switch-out" we understand the point in time where the core ends and the outro begins.

Look at figure 5 and imagine a live performance scenario where target track (A) is being played, and candidate track (B) is selected to be played next. Depending on the mixing style, the intro and/or outro may not exist, leading to the sudden introduction of track B and/or the sudden removal of track A, respectively. Some other mixing styles tend to overlap the two tracks during the transition, using a cross-fade that weakens the volume of A while increasing the volume of B. That means that during the transition, the outro of the track that is being played overlaps with the intro of the track coming in. In this work we will assume that this last one is the mixing style used on a DJ mix.

[1] Looperman: `https://www.looperman.com/`

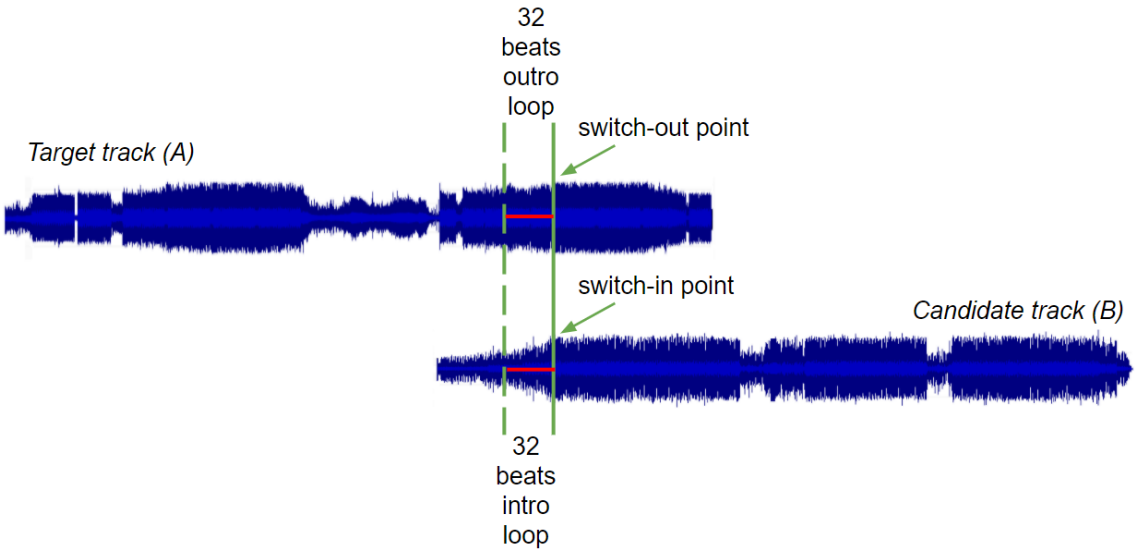


Figure 5: *Schematic representation of a simple transition from target track (A) and candidate track (B). Green lines represent time indexes of intro and outro loops.*

Based on the above, the module should extract the loops of each track from the sections that the DJ would overlap while mixing. Consequently, the module should extract two loops from each track: one from the switch-in and the other from the switch-out positions. In addition, for comparing loops with the same length, we must know the BPM, the position of the beat and the downbeat of each track.

At the moment, the state-of-the-art for the beat estimation task is a publication of 2020 [41], where Böck, S. et al. presented results for the estimation of tempo, beat, and downbeat by increasing the performance of up to 6% points over existing systems. For the structural segmentation task, applied to EDM, a 2020 publication by Zehren et al. [42], developed a system that achieves a precision of 86%, indicating that most of the candidates are of good quality and a recall of 49%, which means their algorithm successfully identifies half of the annotations.

However, these models are not perfect, and possible errors introduced by the previous pre-processing stages can affect the FEM and decrease the performance of the HC calculation. For this reason, we prefer to first develop a strategy that allows us to calculate the HC directly, to then reconsider the incorporation of a beat-tracker and a structural segmentator for estimating the BPM, and switch-in and switch-out positions.

3.3 Dataset

In order to first compute the *TIV-beatwise* to evaluate its performance in HC calculation, we opted to generate a dataset of EDM music tracks with manual annotations of BPM and switch points. The dataset containing the audio files in *.mp3* format and their respective annotations can be found within the Github repository of this project[2].

The EDM sub-genres we chose to target were Progressive House, which is characterised by a lot of melodic and harmonic content, and Techno, which is mainly percussive with little melodic content. Having two genres with such opposite harmonic characteristics allows us to evaluate the performance of the module under different conditions.

For each sub-genre we created a collection of 24 of the most listened tracks on Beatport[3], with a distribution of 2 tracks for each of the 12 minor scales (we chose to work only with minor scales since in EDM there are many more tracks in minor keys than in major). In addition, all tracks in each sub-genre share the same BPM: 124 bpm for Progressive House, and 128 bpm for Techno.

For the search of tracks within Beatport, we relied on the site's own annotations (which are performed by its own algorithms). After selecting the tracks, we verified the information and, in some cases, discarded tracks in order to select new ones that met the requirements. We then generated manual annotations for each of the tracks, containing the following information:

- Time position of first candidate to switch-in
- Time position of first candidate to switch-out
- Time position of second candidate to switch-in
- Time position of second candidate to switch-out

[2]Dataset repository `https://github.com/gbibbo/harmonic_mix`
[3]Beatport `https://www.beatport.com/`

- Key
- Mode
- BPM

The difference between the first and second switch-in candidates is that the first is closer to the start of the track. In the case of the first and second switch-out candidates, the first is closer to the end of the track. In other words, the second candidates are closer to the middle of the track, being more likely to capture richer harmonic information.

3.4 A prototype of Feature Extraction Module (FEM)

To help in the understanding of the following sections, we briefly describe the implementation of our first prototype of the FEM, based on Pérez's system [15]. The annotations of the dataset enabled us to bypass the *beat tracking* and *structural segmentation's* tasks. From the BPM, switch-in and switch-out annotations of first candidates to switch points, we extract a representative loop from the intro section (from *startTime=switch-in - 60*32/BPM* to *endTime=switch-in*) and another from the outro (from *startTime=switch - out-60*32/BPM* to *endTime=switch-out*). The sample rate of the audio is 44.100 Hz. Then, for the audio of each beat measure we compute the harmonic pitch class profile (HPCP). Then, with the *TIVCollection()* class from `TIV.lib` we construct the *TIV-beatwise* as a matrix of 6x32 imaginary numbers, where each column was the *TIV-whole* of one bar measure. Finally we computed the HC between the *TIV-beatwise* of the outro loop of the target track and the *TIV-beatwise* of the intro loop of the candidate track.

Figure 6 illustrates the working principle of Pérez implementation. Figure 9 shows its performance under the Progressive House music collection. In table 5, we refer to this implementation as "FEM#1", so we can compare its performance with upcoming implementations.

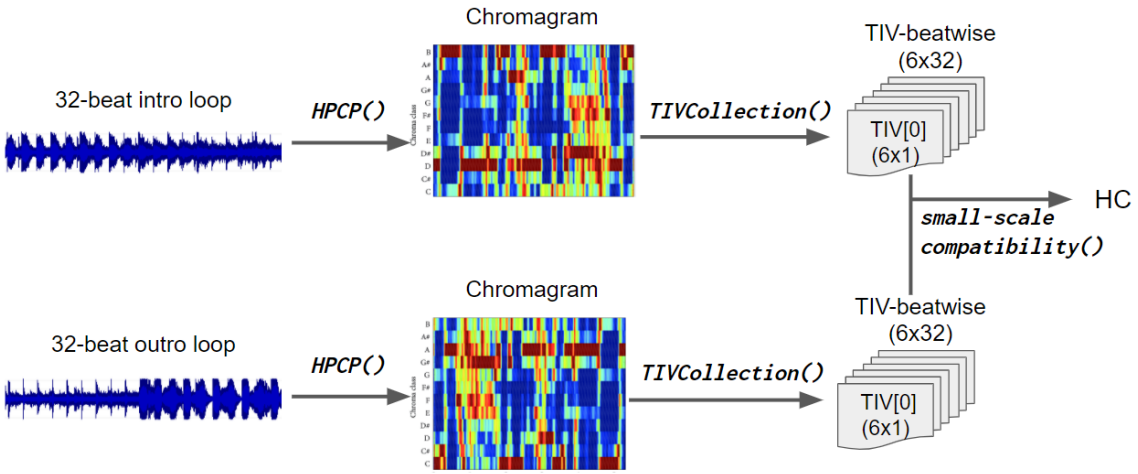


Figure 6: *Diagram of Pérez implementation based on TIVs for HC estimation between loops [15].*

3.5 Performance metric (Perf)

In early stages of development it is good to have a quick and automatic testing and evaluation method that helps to make design decisions by comparing different implementations. Below we present an initial evaluation method based on the above mentioned dataset. To assess whether implementations were useful for generating harmonic mixes, we compared the results obtained with the traditional key-based harmonic mixing method.

We rely on the fact that two tracks with the same key should have a high HC. Two tracks with keys 7 semitones apart should have a lower but quite high HC, as well as tracks 5 semitones apart since, in both cases, they have only an alteration in the key signature. Two tracks with keys 6 semitones apart should have a very low HC. These symmetries, well represented in the circle of fifths, are shown in the figure 7.

In equation 3.1 we define the harmonic compatibility percentage between the target and the candidate tracks. t and c are their respective indexes within the music folder. F represents the set of all the indexes of the music tracks within the folder.

$$HC_{t,c} \in (0, 100) \qquad \forall t, c \in F \tag{3.1}$$

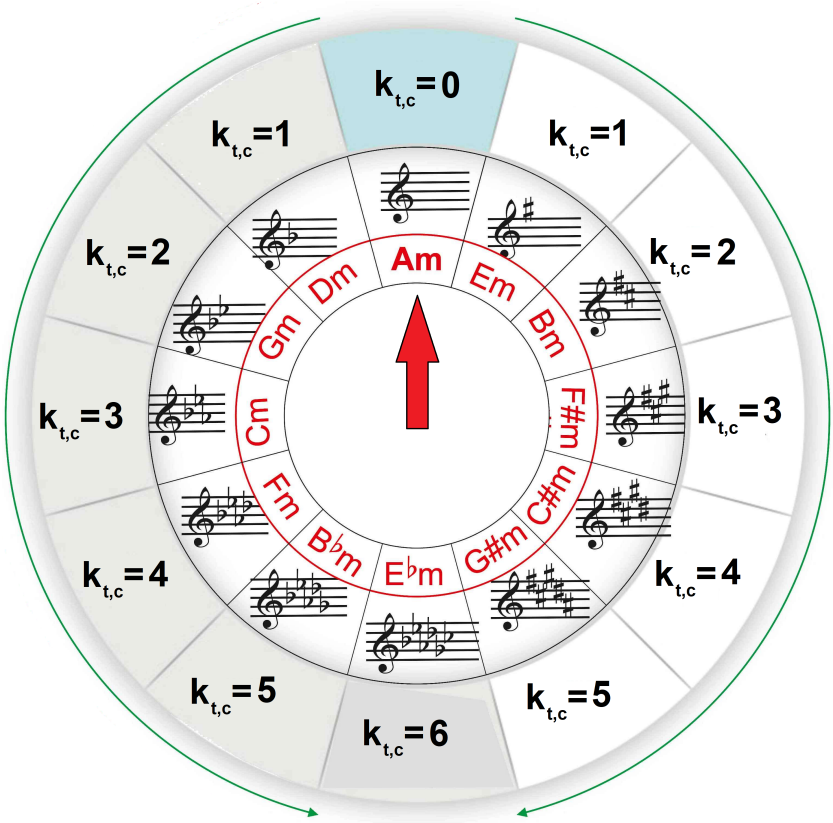


Figure 7: *Symmetry in HC as a function of the number of key signature modifications between t and c, denoted with the variable $k_{t,c}$, where t and c are the indices of the target and candidate tracks respectively. In the example of the figure, the tonality of the target track is Am. With respect to the traditional key-based harmonic mixing method, either a candidate track in Em or in Dm should have the same HC with respect to Am due to its position on the circle of fifths. Same with Bm and Gm, etc.*

c	$HC_{t,c}(\%)$	$k_{t,c}$	c	$HC_{t,c}(\%)$	$k_{t,c}$	c	$HC_{t,c}(\%)$	$k_{t,c}$	c	$HC_{t,c}(\%)$	$k_{t,c}$
1	84.56	5	7	93.05	1	13	88.31	2	19	88.23	2
2	87.99	3	8	87.59	2	14	89.0	6	20	87.01	5
3	88.46	3	9	89.88	4	15	84.56	4	21	93.82	0
4	91.48	3	10	89.53	1	16	85.52	5	22	92.05	1
5	87.79	4	11	90.58	6	17	85.66	4	23	91.54	2
6	91.33	1	12	89.43	3	18	84.85	5			

Table 1: *Results of the first implementation of the algorithm. $HC_{t,c}$ is computed between the target track "Blanka Barbara - Lost in Digital Fog (Original)" and all the other tracks in the Progressive House folder. $k_{t,c}$ represents the distance in semitones between the key of the target track and the candidate.*

We refer in equation 3.2 to the number of key signature differences between a pair of target and candidate track. Those differences are represented in figure 7.

$$k_{t,c} \in [0, ..., 6] \qquad \forall t, c \in F \tag{3.2}$$

In table 1 we can see the results of the first implementation. It is important to highlight that the HC given by the `TIV.lib` [38] library is represented between 0 and 1, being 0 the maximum HC. These values must be converted to a scale from 0 to 100, where 100 represents the maximum HC. Then, looking at how high the HC values in the table are, it seems that all candidate tracks have very good HC with the target, although this is not true. Furthermore, if for example we look at the HC obtained when $k = 3$, these values are similar to those obtained for another value of k. All this indicates that the module is still not good enough for making harmonic suggestions. In continuation, we will try to establish a metric that allows us to compare the different implementations under a common framework, taking into account the mean HC value and the variation of HC for a fixed k.

Given a target track t, we define in 3.3 the mean HC beween t and all other tracks

within the folder so that $k_{t,c} = p$.

$$\mu HC_t[p] = \frac{\sum_c HC_{t,c}.[k_{t,c} = p]}{\sum_c [k_{t,c} = p]} \qquad \forall t \in F \wedge p \in [0, ..., 6] \tag{3.3}$$

In equation 3.4 we show an example of that idea. If we have a target track with key Am from our Progressive House music collection (as in figure 7), $\mu HC_t[3]$ is the average of HC with respect to the two tracks in Cm and the two in $F\#m$.

$$\mu HC_t[3] = \frac{HC_{t,c1} + HC_{t,c2} + HC_{t,c3} + HC_{t,c4}}{4} \qquad : k_{t,c1} = k_{t,c2} = k_{t,c3} = k_{t,c4} = 3 \tag{3.4}$$

We also define, in equation 3.5, the standard deviation of HC beween t and all the others track within the same folder so that $k_{t,c} = p$.

$$\delta HC_t[p] = \sqrt{\frac{1}{\sum_c [k_{t,c} = p]} \sum_c (HC_{t,c}.[k_{t,c} = p] - \mu HC_t[p])^2} \qquad \forall t \in F \wedge p \in [0, ..., 6] \tag{3.5}$$

To clarify ideas, in figure 8 we see an example of $\mu HC_t[p]$ and $\delta HC_t[p]$ plotted when the Progressive House's target track is "Blanka Barbara - Lost in Digital Fog (Original)". This target track is in Am and its BPM is 124. p is plotted on the horizontal axis and HC on the vertical one. Dots represent $\mu HC_t[p]$ and vertical black lines represent $\delta HC_t[p]$. Red line represents the mean of $\mu HC_t[p]$. We can observe that $\mu HC_t[p]$ reaches a maximum when $p = 0$, and $\mu HC_t[p]$ tends to decrease as p increases. As mentioned before, these results are reasonable due to the harmonic relationships between the keys in the circle of fifths. For the case of $p = 0$, that the target and candidate track share the same key, we can observe that $\delta HC_t[0] = 0$. That is because there is only one candidate track in Am in the folder. For $p = 1, ..., 5$, $\delta HC_t[p]$ is calculated with four tracks. In the same way, for the case of $p = 6$ there are only two tracks in $E^{\flat}m$ in the folder, so $\delta HC_t[6]$ is relatively small.

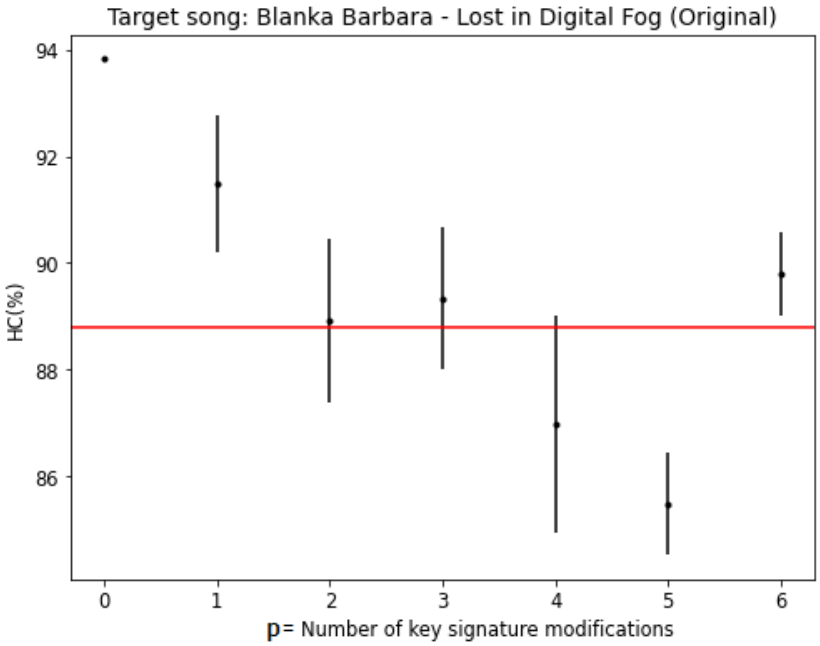


Figure 8: *Results of our first FEM prototype when the Progressive House's target track is "Blanka Barbara - Lost in Digital Fog (Original)". $\mu HC_t[p]$ is represented with dots and $\delta HC_t[p]$ with vertical black lines.*

The example of figure 8 only helps us to understand the concepts, but is not a good measure the "performance" of the algorithm as it is considering only one target track. We should compute new averages to have a global understanding of the performance of the module with all the 24 tracks of Progressive House considered as target. Therefore, for every p, consider the average of $\mu HC_t[p]$ over all the tracks in the folder, as in equation 3.6. Equivalently, consider the average of $\delta HC_t[p]$ over all the tracks, as in equation 3.7. N is the number of tracks in the folder.

$$\mu\mu HC[p] = \frac{1}{N}\sum_{t=1}^{N} \mu HC_t[p] \quad : N = n(F) \tag{3.6}$$

$$\delta\delta HC[p] = \frac{1}{N}\sum_{t=1}^{N} \delta HC_t[p] \quad : N = n(F) \tag{3.7}$$

Figure 9 is a graphical representation of the new variables $\mu\mu HC[p]$ (plotted with dots) and $\delta\delta HC[p]$ (plotted with vertical black lines). It is worth noting how $\mu\mu HC[p]$ drops more clearly in this case than in figure 8, where we analysed the

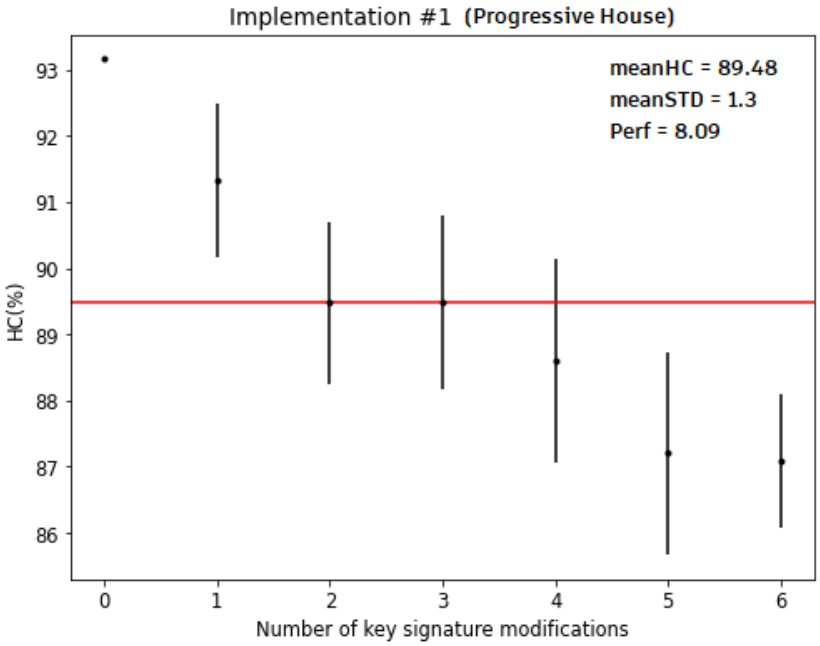


Figure 9: *Results of our first FEM prototype. $\mu\mu HC[p]$ and $\delta\delta HC[p]$ are plotted, averaging $\mu HC_t[p]$ and $\delta HC_t[p]$ over all the tracks of Progressive House.*

behaviour with only one track. This is because, with the new variables and the new plot, we are averaging the 24 tracks, so we can observe the overall behaviour.

As a final measure of the "performance" of the algorithm we have to average $\mu\mu HC[p]$ over the number of differences in the key signature (p), as in equation 3.8. Equivalently, the average of $\delta\delta HC[p]$ over p is expresed in equation 3.9.

$$meanHC = \frac{1}{7}\sum_{p=0}^{6}\mu\mu HC[p] = \frac{1}{7}\sum_{p=0}^{6}\frac{1}{N}\sum_{t=1}^{N}\mu HC_t[p] \quad : N = n(F) \tag{3.8}$$

$$meanSTD = \frac{1}{7}\sum_{p=0}^{6}\delta\delta HC[p] = \frac{1}{7}\sum_{p=0}^{6}\frac{1}{N}\sum_{t=1}^{N}\delta HC_t[p] \quad : N = n(F) \tag{3.9}$$

In the case of this first implementation, these values are $meanHC = 89.48$ and $meanSTD = 1.30$.

It can be seen that all HC values are in the range of approximately 85% to 100%. It would be expected that these values would use the full range from 0% to 100%. As a criterion to improve results in successive implementations, we seek to achieve

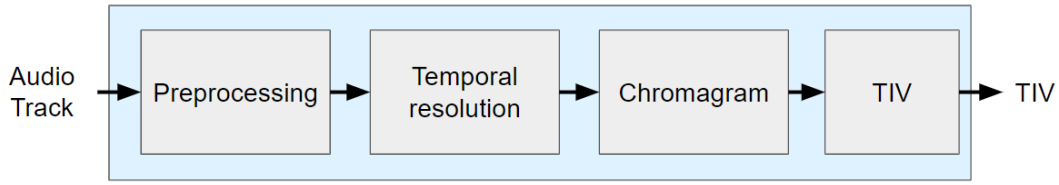


Figure 10: *Architecture of a general FEM for TIV calculation.*

a $meanHC$ as low as possible, allowing for greater excursion of the results, and a $meanSTD$ as low as possible, providing greater accuracy of the measurements. We can compare different implementations with a single number that combines the above variables, as defined in equation 3.10. This new variable represents the proportion of standard deviation in the excursion range of the HC values. We seek for an implementation with the highest $Perf$.

$$Perf = \frac{100 - meanHC}{meanSTD} \tag{3.10}$$

3.6 Strategies for TIV calculation

In figure 10 we show the architecture of a general module that calculates a TIV from a given audio track. The first module that composes the FEM, is responsible for determining the fragment of the audio signal to be analysed. In the case of *TIV-beatwise*, the audio fragments are each of the 32 beat measures of both the intro and the outro sections of the track. Later, we will explore implementations with *TIV-whole*, where the TIV is calculated over a specific section of the audio. The second module is responsible for altering the frequency domain representation of the audio waveform. In this sense, we will compare the results obtained: without preprocessing, by modifying the sampling rate and by applying different source separation libraries to remove the percussive part. The third module calculates the chromagrams. We examine two different strategies for audio chroma representation: HPCP and NNLS. The final stage of the module computes the TIV from the chroma vector, using the library `TIV.lib`.

3.6.1 Temporal resolution

Table 2 summarises the different strategies considered for this module. According to Pérez's experiments, the *TIV-beatwise* strategy maximises its results when using a loop length of 32 beats. This adds the extra complexity of determining the specific sections of the track from which the loops are to be extracted. To evaluate the variation of switch-point positions in HC calculation, we consider the first and second switch-point candidates from the annotations (being the latter closer to the central part of the track rather than the former).

On the other hand, if we consider the uncertainty intervals of Pérez's measurements (see figure 4), we can observe that both *TIV-whole* and *TIV-beatwise* are, at some point, equivalent. This, added to the fact that the *TIV-whole* strategy is simpler to implement (since it is not necessary to know the BPM or the switch-points of the track), motivates us to analyse this other strategy.

In fact, *TIV-whole* strategy may make sense in this context, since extracting two such small audio fragments from an entire music track may not make musical sense, given the nature of EDM tracks. As described by Faraldo [1], EDM typically contains highly compressed tonal information that is presented to the listener in a very sparse manner throughout the entire track. Therefore, this type of music is highly ambiguous from a tonal point of view. To capture the full sense of a tonality or scale on the basis of this hypothesis, we consider a strategy based on *TIV-whole*, that represents a specific section of the audio with a single TIV. This allows to analyse fragments that include almost the entire harmonic content of the track so that in the aggregation of all tonal material, a scale or mode can be more clearly observed. We explored a strategy that analyses the entire track (100%), one that discards the beginning and ending (central 50%), and finally one that analyses only the main central part of the track (central 30%). Analysing only the central part of the track may make sense since most of the musical information is concentrated in the central part.

Time resolution	{beatwise, whole}
Switch-point position (only for TIV-beatwise)	{first candidate, second candidate}
Section of track to analyze (only for TIV-whole)	{ 100 %, central 50%, central 30%}

Table 2: *Different strategies and parameters considered for the Temporary Resolution module.*

3.6.2 Preprocessing

This module is intended to prepare the signal for the chroma vectors calculation. Table 3 summarises the strategies for this module and the parameters considered. We seek to clean the frequency spectrum to keep only the harmonic content of the signal, and consequently improve the HC estimations.

Although there are not many publications that represent audio with TIVs, a recent paper by Ramoneda [43] implements a system, similar to ours, for harmonic detection function (onset of chords). He explores different strategies, among which downsampling and source separation gave the best results. Downsampling may attenuate fast fluctuations in the musical audio, similarly to the effect of a low pass filtering. For his application in harmonic transient detection, the sampling rate that gave the best results was 8000Hz. He additionally decomposed the musical audio into harmonic and percussive sources, by adopting a harmonic percussive source separation (HPSS) algorithm with median filtering, using HPSS from librosa. By discarding the percussive part, he aimed to exclude transient frequencies that have a negative impact on providing an optimal transcription of the harmonic content of the audio signal.

Apart from librosa, we also evaluated newer technologies for source separation. We tested Open-Unmix [44], a deep neural network reference implementation for music source separation. Awarded 2nd place in the PyTorch Global Summer Hackathon 2020, Open-Unmix[4] provides ready-to-use models that allow to separate the audio into four stems: vocals, drums, bass and the remaining other instruments. In our

[4]OpenUnmix `https://github.com/sigsep/open-unmix-pytorch`

HPSS	{true, false}
Sampling rate	{8.000, 44.100}
Library	{librosa, OpenUnmix}

Table 3: *Strategies and parameters considered for the Prepossessing module.*

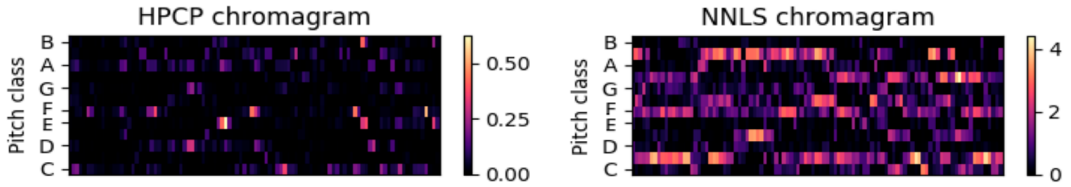


Figure 11: *Chromagrams for an A major chord. Extracted from [43].*

module, we tested the version of Open-Unmix that extracts drums.

3.6.3 Chromagram

This module receives an audio frame as input and calculates the chromatic vector. The chroma vector (or chromagram) is typically a 12-element feature vector indicating how much energy of each pitch class, (C, C#, D, D#, E, ..., B), is present in the signal. They are calculated by first applying the Short Fourier Transform to small sections of the audio track. These sections are known as "frames" and their length in number of samples is known as frameSize. Varying this parameter results in different resolutions in the frequency domain. In applications such as ours, where frequency identification is desired and temporal resolution is not so important, relatively large frameSizes are often used. In this thesis we explore the impact that different frame sizes have on the final result.

Many algorithms for computing chromagrams have been proposed in the literature, such as pitch class profiles (PCP), harmonic pitch class profiles (HPCP), the CRP chroma or the NNLSChroma. They differ mainly because they change the degrees of invariance of a particular musical attribute (e.g., timbre) or enhance the level of symbolic transcription (e.g. by reducing the impact of harmonics or transient noise) [43]. Figure 11 shows the different results between a HPCP and a NLSS chroma when representing a A major chord.

Chroma vector	{HPCP, NNLS}
frameSize	{4096, 8192, 16384}

Table 4: *Different strategies and parameters considered for the Chromagram module.*

The (HPCP), described by Gómez [45], is based on the (PCP). It is tuning independent, considers the presence of harmonic frequencies and is robust to noise (e.g. noise from percussion). The size of the vector can be either 12, 24 or 36 bins.

The non-negative least-squares chromagram (NNLS) was proposed by Mauch [46]. It computes an approximate note transcription from musical audio before the chroma computation by adopting a non-negative least-squares optimization algorithm. Typically it provides a sparse representation of the input signal in the chroma domain.

HPCP chromagrams are probably the natural implementations for the calculation of TIVs, as is reflected in TIV tutorial examples [5] and Pérez implementation [6]. However, in two previous works from Bernardes et al. [35][47] based on TIVs, the NNLSChroma has shown to be best at key detection, as it provides much less residual noise and harmonic content. These results motivate us to test both strategies, as reflected in table 4.

In our module architecture, the Chromagram module precedes the TIV calculation one, forcing compatibility between the chroma vectors and the allowed inputs for `TIV.lib`. This library perfectly supports PCP and HPCP either for *TIV-framewise*/ *TIV-beatwise* $(12 \times n)$ and *TIV-whole* (12×1) calculation. However, for the case of NNLSChroma, `TIV.lib` only supports single chroma vectors, therefore this type of chroma can only be implemented for *TIV-whole* calculation.

Another aspect of `TIV.lib` and `NNLSChroma()` from Essentia, is that are implemented only for 12-bin chroma vectors. Some electronic music tracks are detuned, meaning that their notes are not located at their tonal centre. In these cases, the

[5] TIV.lib Example Notebook `https://github.com/aframires/TIVlib/blob/master/TIVlib_example.ipynb`

[6] Pérez implementation `https://github.com/migperfer/harmonic_compatibility`

chromagrams obtained are closer to theoretical tonal hierarchies. To identify such cases, Faraldo [1] proposes to use a resolution of 3 bins per semitone, which is equivalent to a 36-bin chromagram. We did not explore the impact of this variant on the calculation of HC, but it would be interesting to do so in future works.

3.7 FEM prototypes and their results

To evaluate the performance of a system with a wide range of parameters, a grid search method is often employed. In our case, given all the strategies considered, a minimum of 88 modules should be implemented and tested. Instead of performing that, we selected the implementations with the most promising results based on, the literature presented in the previous sections, and the elections that showed to improve performance. Table 5 shows some of the implementations considered and their performance for the calculation of the HC under the Progressive House folder.

The final configuration for our FEM is selected based on two criteria: maximising the value of *Perf* and minimising the execution time. It is worth remembering that *Perf* is an estimate of the accuracy with which our module replicates the harmonic relationships, derived from the circle of fifths, between our songs. Although we do not calculate the standard deviation of *Perf* to establish the accuracy of this measure, it is clear that the value of *Perf* is closely related to our music folder, so it is not representative of the performance of the different FEM. For this reason, we have considered all FEMs with *Perf* greater than 10 (i.e. FEM #6,7,8,9 and 10) as possible candidates for our final module.

To break the tie, we measured the execution time to analyse each song for FEM #6, 7, 8, 9 and 10. These times were measured running the modules on Google Colab, considering a song duration of 7:03 minutes on average. We assume that the execution times are proportional if a personal computer is used. The final decision of our FEM is a compromise between good analysis and reasonable execution time.

Figure 12 represents the processing of the signal before TIV calculation. In figure 13 we can appreciate the results obtained by running FEM#10 on our Progressive

	FEM#1	**FEM#2**	**FEM#3**	**FEM#4**	**FEM#5**
Time resolution	beatwise	beatwise	beatwise	whole	whole
Switch-point position	first candidate	second candidate	second candidate	-	-
Section of track to analyze	-	-	-	100%	100%
Sampling rate	44.100	44.100	44.100	44.100	8.000
HPSS	false	false	true	false	false
Library	-	-	librosa	-	-
Chroma vector	HPCP	HPCP	HPCP	HPCP	HPCP
frameSize	16384	16384	16384	16384	16384
Perf	8.09	8.59	8,29	8.7	4.95

	FEM#6	**FEM#7**	**FEM#8**	**FEM#9**	**FEM#10**
Time resolution	whole	whole	whole	whole	whole
Switch-point position	-	-	-	-	-
Section of track to analyze	100%	100%	30%	50%	100%
Sampling rate	44.100	44.100	44.100	44.100	44.100
HPSS	true	true	true	true	true
Library	OpenUnmix	librosa	librosa	librosa	librosa
Chroma vector	NNLS	NNLS	NNLS	NNLS	NNLS
frameSize	8192	16384	8192	8192	8192
Perf	10.77	10.68	10.44	10.68	11.26
Execution time	332s	61.5	16.7s	24.6s	51.4s

Table 5: *Some of the most relevant FEM prototypes implemented for testing. The parameters and algorithms selected, as well as the performance for the calculation of HC (Perf), are listed.*

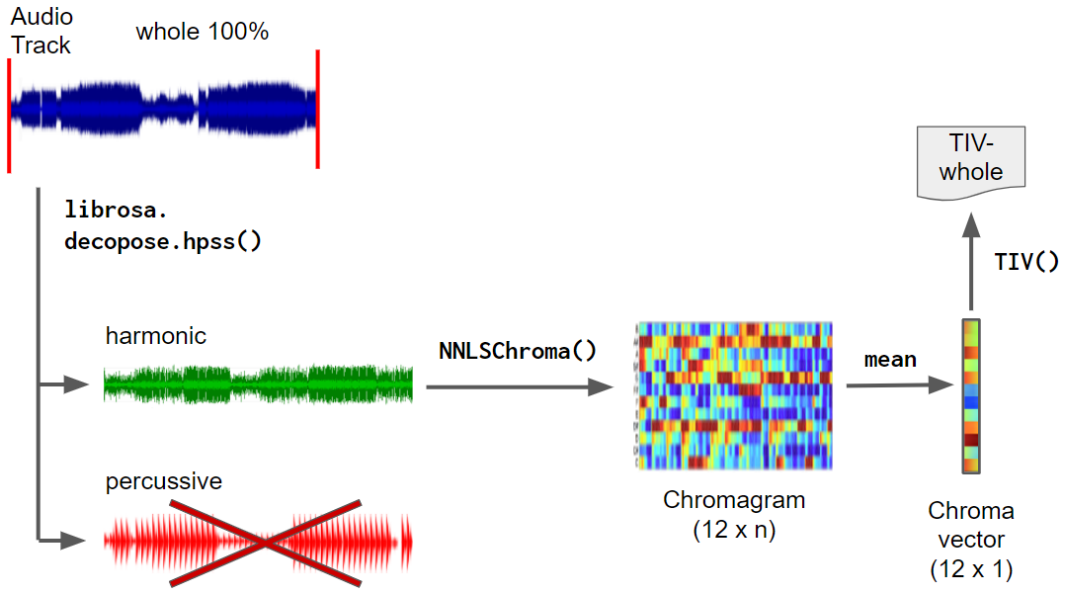


Figure 12: *Illustration of how FEM#10 extracts a TIV from an audio track.*

House music collection.

3.8 Discussion

The most obvious limitation of this chapter is the small size of the dataset (24 Progressive House and 24 Techno tracks). While many datasets exist with key and BPM annotations, the same is not true for switch-in and switch-out points. This forced us to generate manual annotations which, due to the constraints of the human resources available for this project, are scarce.

If our FEM was ideal and the dataset infinite, we would expect to see that graph 13 (which shows the results obtained by running our final FEM on the Progressive House folder) is clearly descending. This is because the expected consonance between two tracks decreases as the number of alterations in the key signature increases. However, graph 13 presents a plateau between p = 2-3 and p = 5-6. This can be explained either by a low performance of our FEM or, more probably, by the non-homogeneous distribution of HC among the tracks of our Progressive House folder. This problem is usually solved by increasing the dataset size.

We did not enlarge the dataset size because the implementations of FEM #1,2 and 3

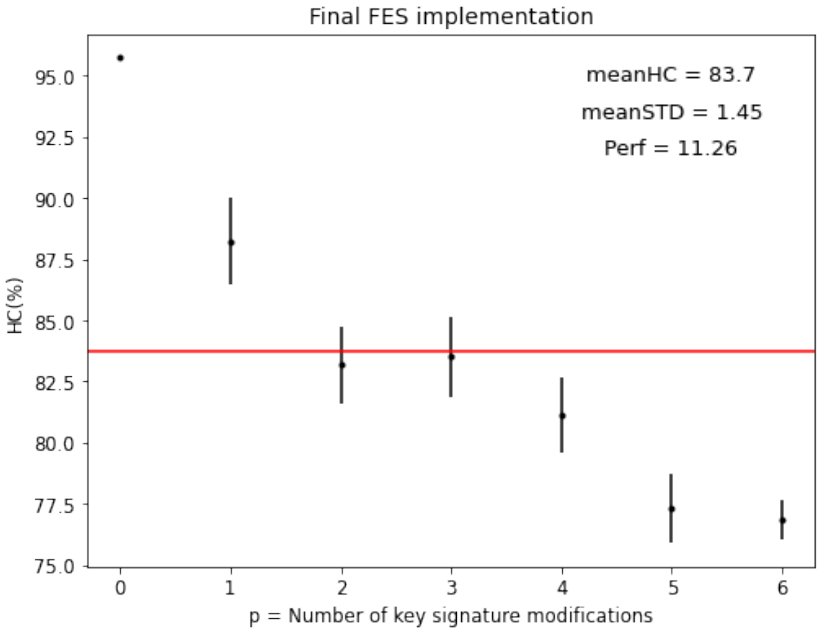


Figure 13: *Results of definitive module of figure 12. For each p, black dots represent the mean HC over all the tracks from the Progressive House music folder. Black vertical lines represent the standard deviation of HC over all the tracks for each p. The final rating values stand this implementation out from the others.*

do not allow it, as they require manual annotations. We also decided not to enlarge the dataset size for the other implementations so all FEMs could be compared under the same conditions.

In addition, all implementations were tested with the Progressive House folder. This decision was taken due to the poor performance of the algorithm in the Techno folder. The results showed that the *meanHC* increases a little bit, meaning that Techno tracks often sound better together despite the key. In addition, $Perf$ decreased slightly, which means that the HC values are tighter and it is more difficult to distinguish differences between harmonic compatibilities. For the design stages of the module, this genre of music did not allow us to appreciate the different harmonic relationships between the tracks.

On the other hand, some of all potential combinations between different strategies, could not be implemented. In particular, if the `TIV.lib` library is included, the NNLSChroma cannot be implemented in a FEM with a temporal resolution of beatwise or framewise. This is because, according to the authors, this technique for

calculating chroma is not supported by `TIV.lib`. However, it is possible to employ the NNLSChroma for representing the entire audio signal with a single vector. Taking advantage of this particularity, we decided to successfully test *TIV-whole* based modules.

Chapter 4

Harmonic Mix System (HMS)

In this chapter we describe the Harmonic Mix System (HMS) we provide to overcome the constraints identified in Section 2.4. The entire software is open source and can be downloaded from the project's Github repository [1]. It was conceived to be used by DJs in live performances, where they have a main track playing and want to find another track that allows them to create a harmonic transition.

For this purpose, we provide information regarding the HC, with a one-to-many mapping, similar to other existing mixing software. This implementation saves the information on disk to optimise the computational cost of analysing the audio tracks, providing a smooth real-time operation. The HMS is implemented with a simple user interface from which tracks can be analysed and HC information visualised.

The subsequent acronyms will be used in the future sections:

- **HC (%)** = **H**armonic **C**ompatibility between the main track and the others in the folder
- **T(st)** = pitch **T**ransposition interval that maximizes the HC (in **s**emi**t**ones)
- **THC (%)** = resulting **H**armonic **C**ompatibility if the pitch **T**ransposition is applied

[1] Project repository `https://github.com/gbibbo/harmonic_mix`

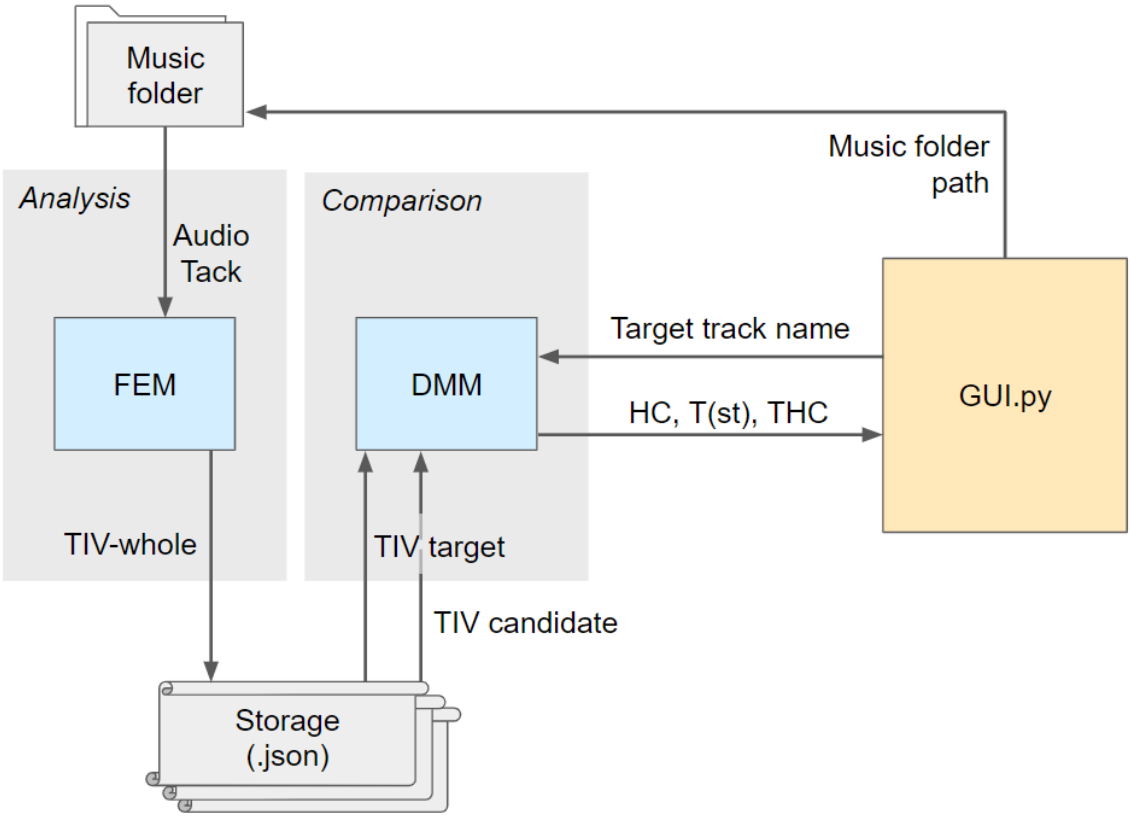


Figure 14: *Block diagram of complete Harmonic Mix System.*

4.1 Complete architecture overview

Figure 14 shows a block diagram representing the overall architecture of the HMS. The first step for using the HMS is to define the path to the music folder with audio files in .mp3 format through the graphical user interface. Then, the system has two distinct instances of execution:

4.1.1 Analysis

In this stage, the features of each of the audio tracks are calculated. This is done using the Feature Extractor Module (FEM) described in more detail in Section 4.2, where each audio track is described by a *TIV-whole*. The operation takes approximately one minute to analyse 7min of audio. Once the features have been calculated, the information is saved in a `.json` file, inside a new folder called "annotations" which is created inside the music folder. For each audio track, a `.json` file is created.

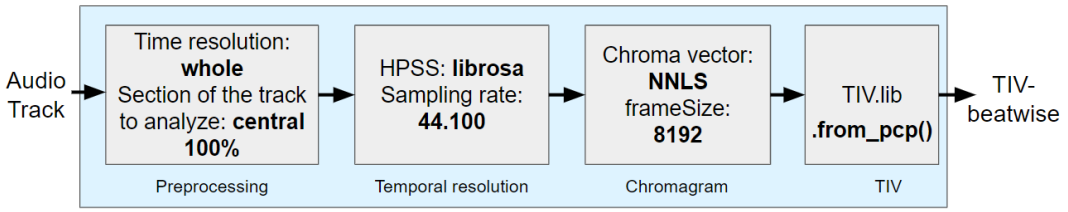


Figure 15: *Block diagram of the architecture of the FEM that extracts a TIV for the future calculation of HC between two audio tracks. The algorithms and selected parameters are shown.*

4.1.2 Comparison

After the Analysis step is successfully completed, a target track can be selected from the user interface, which triggers the Comparison step. In this instance, the HC between the target track and each of the tracks inside the folder is calculated, making use of the Data Management Module (DMM), described in more detail in Section 4.3. For this, the `.json` file corresponding to each track is read to extract the TIV. Then, using `TIV.lib`, the HC, T(st) and THC are calculated. This information is then displayed in the user interface.

4.2 Feature Extraction Module (FEM)

The choices for the architecture of this module are based on the results obtained in Chapter 3. The block diagram of this module is represented in figure 15. Our FEM gets the audio from a music track and returns a *TIV-whole* that better represents the signal for computing the HC, as illustrated in figure 12.

To do so, it applies source separation over the entire audio track, to extract the harmonic content of the signal using "Harmonic Percussive Source Separation" from librosa library. By using the Essentia library, it calculates the NNLSChroma for each frame, with a `frameSize=8193` and a 'hann' window. Then, it averages all the chroma vectors to get a single vector that represents the signal. Finally, it stores the result of applying the `.from_pcp()` function to the chroma vector, on a TIV instance of the `TIV()` class from `TIV.lib`.

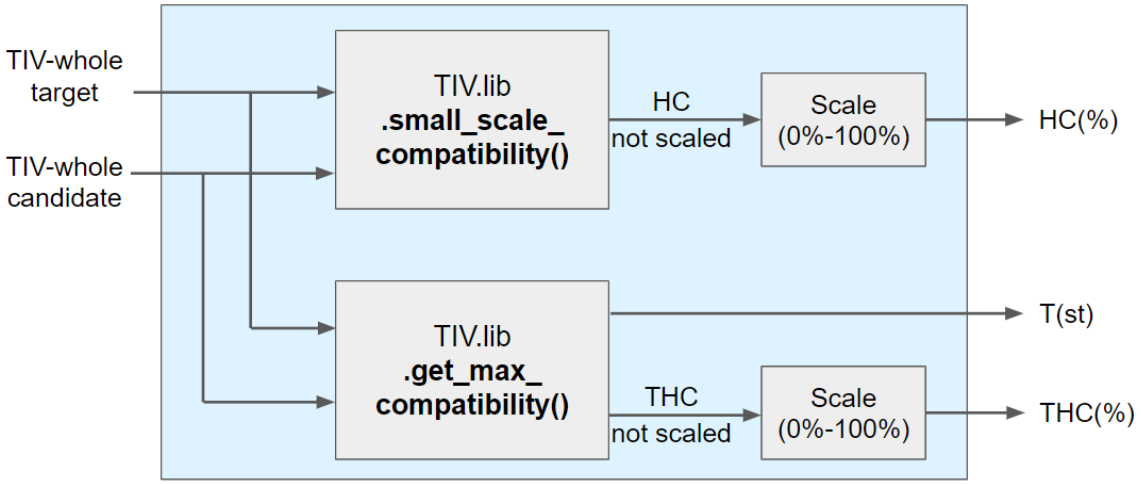


Figure 16: *Block diagram of Data Management Module.*

4.3 Data Management Module (DMM)

This module is in charge of calculating the HC and the other values that will be displayed in the user interface, based on the previously calculated TIVs. For this, it is assumed that the analysis stage has been previously executed, and that there is a folder called "annotations" with a . json file for each of the audio tracks. Every time the user defines a new main track, this module compares it with each of the tracks inside the folder. This way, in at least one of its executions, it compares the main track with itself, achieving a 100% HC.

Figure 16 shows a block diagram of how it works. Its architecture is fundamentally based on the use of functions from the `TIV.lib` library. The HC is computed with the function `small_scale_compatibility()`. The function `get_max_compatibility()` performs several key transpositions of the candidate track (as mentioned in Section 2.2.1) and calculates the HC between each of them and the main track, selecting the interval that maximises the HC. The transposition intervals of the function, range from -6 to +6 semitones, spanning a full octave.

The returned HC value from the library ranges from O (very compatible) to 1 (very dissonant). First it has to be converted into a percentage range from 0% (very dissonant) to 100% (very compatible) and then, as the actual values are close in the 70-100% range, a scaling factor has to be applied to use the 0-100% range, as shown

in equation 4.2.

$$HC_{70,100} = 100(1 - HC_{1,0}) \tag{4.1}$$

$$HC_{0,100} = \frac{100(HC_{70,100} - 70)}{100 - 70} \tag{4.2}$$

4.4 Graphical User Interface (GUI)

We aim to ensure that the user experience does not change significantly from what the DJ is accustomed to. The design is intended to be incorporated into current software. Such software usually analyses tracks, as soon as they are added to the library, to estimate the key and bpm, among others. They also expect the user to be able to define a track as the main track, as shown in the figure 2. For this reason, we suggest an interface design like the one in the figure 17.

As proposed by Bernardes, G. et al [23], the HC value and the key are complementary, since both are important to understand the harmonic relationships between the tracks. While key signature, alone, does not provide enough information regarding harmony (as discussed in Section 1.3), the HC lacks information about melody (i.e., a particular HC value can be due to two different keys). The HC is helpful in finding the best harmonic matches between the tracks in the collection, and the key in guaranteeing control over the overall harmonic structure of the mix.

We created a software prototype in Tkinter that implements the proposed design, for which a screenshot is shown in figure 18. It can be downloaded from the Github repository of this project [2], to be used and tested by now. The window is resizable to fit your monitor, while using other software of choice in parallel.

[2]Project repository `https://github.com/gbibbo/harmonic_mix`

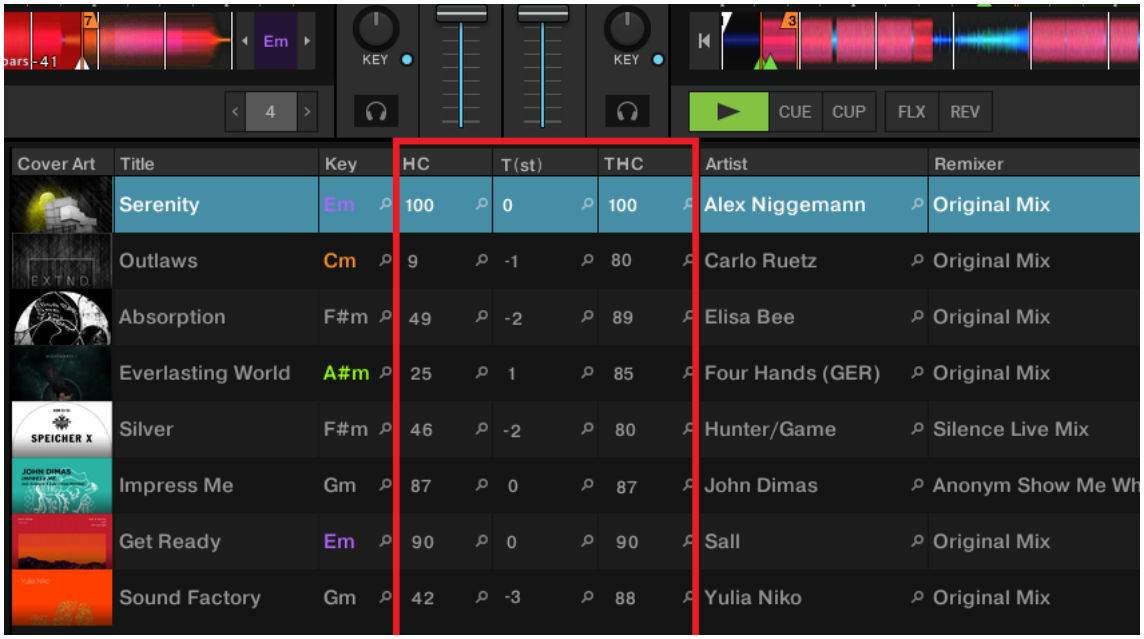
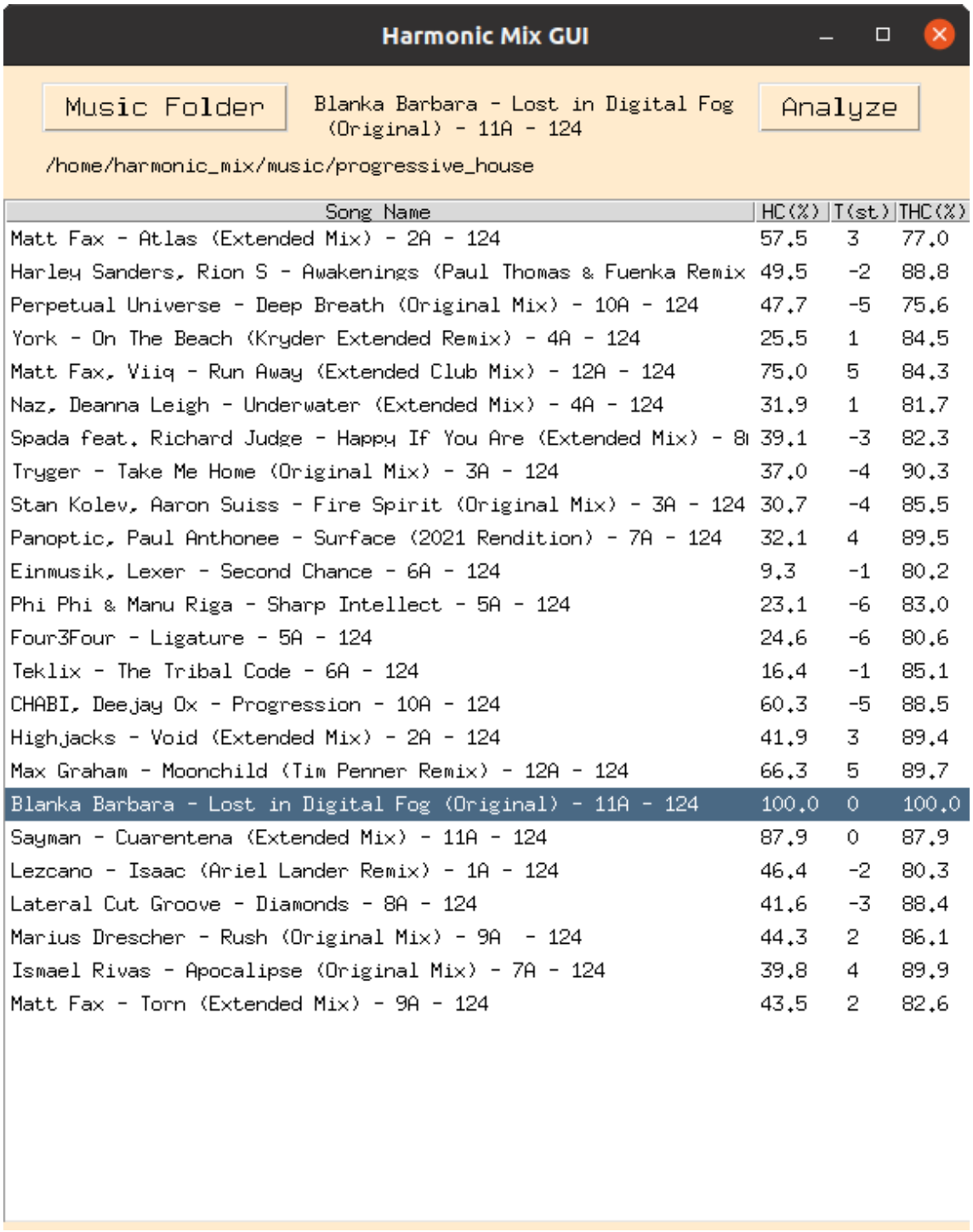


Figure 17: *Mock-up of the incorporation of HC information in current DJ software that would enhance the harmonic mixing experience. First column displays the harmonic compatibility (HC) between the main track and the others in the folder, second one the suggested pitch transposition interval in semitones that maximizes the HC (T(st)), and third one the resulting harmonic compatibility after pitch transposing (THC).*



Song Name	HC(%)	T(st)	THC(%)
Matt Fax - Atlas (Extended Mix) - 2A - 124	57.5	3	77.0
Harley Sanders, Rion S - Awakenings (Paul Thomas & Fuenka Remix	49.5	-2	88.8
Perpetual Universe - Deep Breath (Original Mix) - 10A - 124	47.7	-5	75.6
York - On The Beach (Kryder Extended Remix) - 4A - 124	25.5	1	84.5
Matt Fax, Viiq - Run Away (Extended Club Mix) - 12A - 124	75.0	5	84.3
Naz, Deanna Leigh - Underwater (Extended Mix) - 4A - 124	31.9	1	81.7
Spada feat. Richard Judge - Happy If You Are (Extended Mix) - 8A	39.1	-3	82.3
Tryger - Take Me Home (Original Mix) - 3A - 124	37.0	-4	90.3
Stan Kolev, Aaron Suiss - Fire Spirit (Original Mix) - 3A - 124	30.7	-4	85.5
Panoptic, Paul Anthonee - Surface (2021 Rendition) - 7A - 124	32.1	4	89.5
Einmusik, Lexer - Second Chance - 6A - 124	9.3	-1	80.2
Phi Phi & Manu Riga - Sharp Intellect - 5A - 124	23.1	-6	83.0
Four3Four - Ligature - 5A - 124	24.6	-6	80.6
Teklix - The Tribal Code - 6A - 124	16.4	-1	85.1
CHABI, Deejay Ox - Progression - 10A - 124	60.3	-5	88.5
Highjacks - Void (Extended Mix) - 2A - 124	41.9	3	89.4
Max Graham - Moonchild (Tim Penner Remix) - 12A - 124	66.3	5	89.7
Blanka Barbara - Lost in Digital Fog (Original) - 11A - 124	100.0	0	100.0
Sayman - Cuarentena (Extended Mix) - 11A - 124	87.9	0	87.9
Lezcano - Isaac (Ariel Lander Remix) - 1A - 124	46.4	-2	80.3
Lateral Cut Groove - Diamonds - 8A - 124	41.6	-3	88.4
Marius Drescher - Rush (Original Mix) - 9A - 124	44.3	2	86.1
Ismael Rivas - Apocalipse (Original Mix) - 7A - 124	39.8	4	89.9
Matt Fax - Torn (Extended Mix) - 9A - 124	43.5	2	82.6

Figure 18: *Graphical User Interface of the Harmonic Mix System, programmed with Tkinter. "Music Folder" button allows to select a music folder and display the name of the files in .mp3 format. For each music folder, the "Analyze" button should be pressed only once at the beginning to compute the TIV of the audio tracks. That information is stored on disk. Double-clicking on the target track displays the harmonic compatibility (HC), suggested transposition interval in semitones (T(st)), and resulting harmonic compatibility after pitch transposing (THC).*

Chapter 5

Experiment and evaluation methods

5.1 Introduction

The main goal of this project is to test the validity of a new measure of HC in the context of DJing, and not the development of an interface itself. For this reason, the usability of the interface is not going to be evaluated as it is a simple mock-up. We consider that its basic design is already validated as it replicates commercial softwares. The actual GUI can be run on a separate window in paralel to the DJ software of choice, which makes it not very practical on real scenarios.

Then, to evaluate whether our system is useful for making harmonic mixes of EDM music, it is necessary to test the system using this type of music. The resulting mixes of two EDM tracks can be pleasant or not depending on several factors, besides their HC. For this reason, the experiment design we propose, based on ABX tests, focuses on comparing two different mixes where the only modification introduced alters just the HC, by means of a pitch transposition. All other variables are the same in both mixes, thus neutralising them.

Also, in the case of rankings based on score scales the ratings for the same mixes can be different between different users, even when preserving the same order of preference. Therefore, we will express the users' preference with a new variable which is the difference in score between the clips. This new variable can be positive

or negative, which allows us to know the order of preference according to the sign. Similarly, the suggestions of our system will also be expressed in a new variable representing the difference in HC between one mix and the other.

We do this because the absolute value we assign to the HC between two tracks has no meaning per se. It becomes meaningful when we have several HC values (from several candidate tracks) against which they can be compared, in order to choose the track that would sound optimal in the mix. Thus, we are interested in assessing the system's ability to measure HC in relative terms, rather than in absolute terms.

In this chapter we will start by describing the experiment, then we will explain the strategy used to represent the results, and finally we will show the scores obtained and compare these results with those of Section 3.7.

5.2 Experiment description

In this experiment, we asked musically trained participants to score the consonance of two audio fragments, and we compared their answers with those proposed by our HMS. For this, we created an online survey[1]. Before starting the experiment, they were informed about the average time required to complete the experiment (ten minutes aprox.) and what is expected of them. Then users were presented with 10 examples, where each example involves one target track and one candidate track.

As shown in Figure 19, each example is composed of two 32 beats (16 seconds approx.) audio fragments, which we call "Clips". According to the authors of Automashupper [8], 'the likelihood of very high harmonic similarity is much greater for a short section of 8 beats than one which is 64 beats containing multiple chord changes, hence it is not trivial to meaningfully relate mashability (HC) between phrase sections of different lengths'. Based on this, we present to the participants, Clips of a fixed length of 32 beats, which are longer than 8 beats (avoiding high likelihood of harmonic similarity), and shorter than 64 beats (that can make users feel tired during the experiments).

[1]Online survey `https://forms.gle/1VaerdHwhvxehuz77`

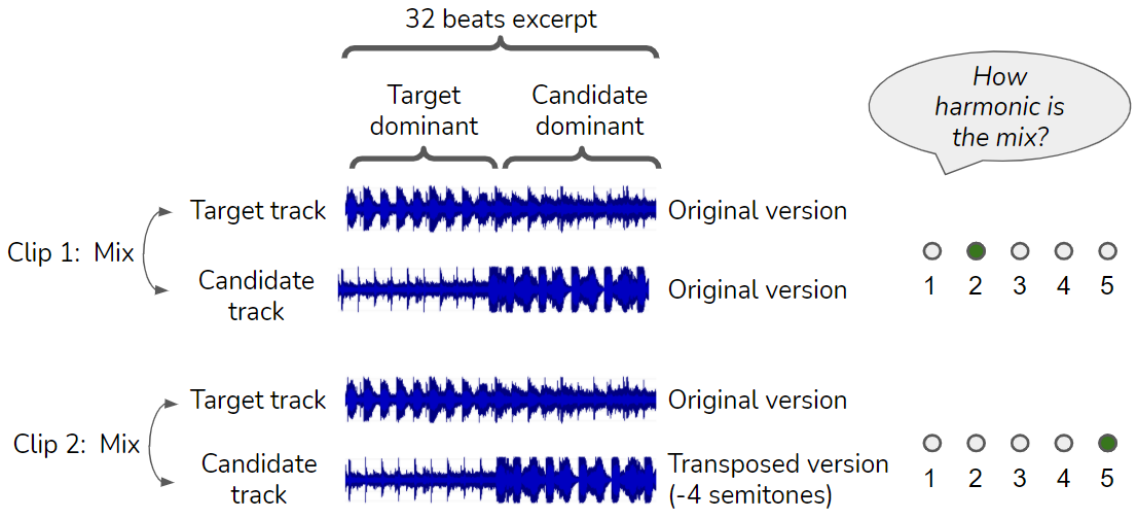


Figure 19: *Illustration of one Example of the experiment. The user is presented with two Clips of two audio fragments (32 beats long) mixed with the same method. The only difference between the clips, is a pitch transposition applied to the target track of one of the clips. In the example of the figure, a pitch transposition of -4 semitones was applied to improve the HC. The user is asked to rate how harmonic they found each clip on a scale from 1 to 5.*

Each Clip is the resulting mix of a 32-beat excerpt from the switch-out of the target track (the point at which the core finishes and the outro starts), and another excerpt from the switch-in of the candidate track (finishes the intro and starts the core). We chose to work with these sections as they are musically intended to be mixed and overlapped with other tracks. This allows us to apply a standard mixing technique to all the examples, avoiding the introduction of artistic artifacts that may alter the participant's choice. We did not use the central part of the tracks as they usually contain a lot of musical information and the mixing has to be done carefully with artistic criteria, making it difficult to standardise.

Different professional party DJs were interviewed about the standard mixing technique for mixing transitions in a 32 beat interval. Generally speaking, the mixes, recorded on Traktor, follow this criteria: The target track is mixed from the 16 beats before a switch-out (where usually the first 16 beats have kick and the next 16 beats do not). The candidate track is mixed 16 beats before a switch-in (where usually the first 16 beats do not have a kick and the next 16 beats do have). Mids and treble are at 90% on both tracks. The lows of the target track are at 100% for

the first 12 beats. From beat 12 to 16 there are no lows in the mix. From the last 16 beats onwards the target track's lows are lowered to 0% and the candidate track's lows are raised to 100%.

One of the two Clips contains the mix of the two tracks in their original version. The other contains the same mix with the original target track, but the candidate track has a pitch transposition applied to make it higher or lower. In some cases, pitch transposition is applied to maximise HC. But in other cases, where the HC of the original versions was already maximised, pitch transposition was applied to reduce the HC. After each user listens to the clip, they are asked to score the harmonic result of each mix on a range of 1 to 5.

We sought to work with different musical genres that would allow us to cover different aspects of musical composition. According to Beatport's genre labels, the music genres used in the 10 examples are: 2 Progressive House, 2 Afro House, 2 Indie Dance, 4 Techno. Progressive House was used because it is one of the genres with the highest high-frequency content. Afro house because of the importance of its rhythmic pattern (which is eliminated by applying source separation) and its melodic content (similar to house). Indie dance for having a lot of melodic content and usually vocals. And 4 examples of Techno were used because it is the most listened to genre within the EDM sub-genres [48]. Also, as Techno is such an all-encompassing sub-genre, 2 of these tracks belong to a more melodic Techno and the other 2 to a more commercial one.

Also, the transposition intervals of the mixes are equi-spaced. That is: there is an example with a clip transposed -5 semitones, another with +5, another with -4 and +4st, with -3 and +3st, -2 and +2st, and -1 and +1st. In this way, the pitch differences between clip 1 and 2 are evenly distributed in the experiment. On the other hand, the HC values given by the system for each of the mixtures were also uniformly distributed among the Clips of the experiment. In this way we can analyse the behaviour of our system under different conditions.

In synthesis, participants are presented with two practically identical Clips, the

only difference being that one of them has a pitch transposition and, consequently, a different HC value. In this way, we aim to eliminate the variables that can generate pleasure when listening to a mix, so that the user focuses his or her attention on the modification introduced by the pitch alteration. We seek to verify whether users' preferences, in favour of one of the two Clips, coincide with the suggestions of our HMS.

5.3 True Positive Rate (TPRc)

As mentioned above, we will not consider the HC measures in absolute terms, but will consider them in relative terms, as the difference between Clip 1 and Clip 2 values. This consideration will be valid for both the user scores and the HC values suggested by our system. One of the advantages of representing the results in this way is that it allows us to analyse on a case-by-case basis whether the HMS suggestions align with user preferences (what we call "true positive"). In other words, we can understand whether users agree with the HMS on which Clips sound best.

Let us define some new variables. Given HC_e^1, the harmonic compatibility returned by the system for Clip 1 of the example e, and equivalently HC_e^2 for Clip 2, we can define a new variable d_e as the difference between the HC values of Clip 1 and 2, as shown in equation 5.1. For each different system architecture, we will obtain a different D vector.

$$D = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_{10} \end{pmatrix} \quad : \text{where} \quad d_e = HC_e^1 - HC_e^2 \tag{5.1}$$

Once the experiment is finished, we obtain the answer matrix A, expressed as in the

equation 5.2.

$$A = \begin{pmatrix} a_1^1[1] & a_1^1[2] & \cdots & a_1^1[P] \\ a_1^2[1] & a_1^2[2] & \cdots & a_1^2[P] \\ a_2^1[1] & a_2^1[2] & \cdots & a_2^1[P] \\ a_2^2[1] & a_2^2[2] & \cdots & a_2^2[P] \\ \vdots & \vdots & \ddots & \vdots \\ a_{10}^1[1] & a_{10}^1[2] & \cdots & a_{10}^1[P] \\ a_{10}^2[1] & a_{10}^2[2] & \cdots & a_{10}^2[P] \end{pmatrix} \quad : A \in \mathbb{M}^{20\times P} \tag{5.2}$$

Where $a_e^1[u]$ is the answer score given by user u to the Clip 1 of the example e. Similarly $a_e^2[u]$ for the Clip 2. P is the total number of participants in the experiment.

If we calculate the difference in scores that users gave to Clip 1 and 2, we obtain the matrix C, as in equation 5.3.

$$C = \begin{pmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,P} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,P} \\ \vdots & \vdots & \ddots & \vdots \\ c_{10,1} & c_{10,2} & \cdots & c_{10,P} \end{pmatrix} \quad : C \in \mathbb{M}^{10\times P} \tag{5.3}$$

Where $c_{e,u} = a_e^1[u] - a_e^2[u]$ is the difference of scores given by user u between Clip 1 and 2 for example e.

At this point, we have a vector D (of length 10) representing the suggestions calculated by the system for each of the 10 examples. And we have a matrix C (of dimensions $10 \times P$) representing the user preferences obtained in the experiment. For our HMS incorporating FEM #10, we have plotted system suggestions on the horizontal axis, and all users preferences on the vertical axis, as shown in Figure 20.

We will say that a user agrees with the system's suggestions if, and only if, one of the two Clips was rated best by the user while also having the best HC value given by our HMS. This can be seen in the graph by studying the sign of the variables,

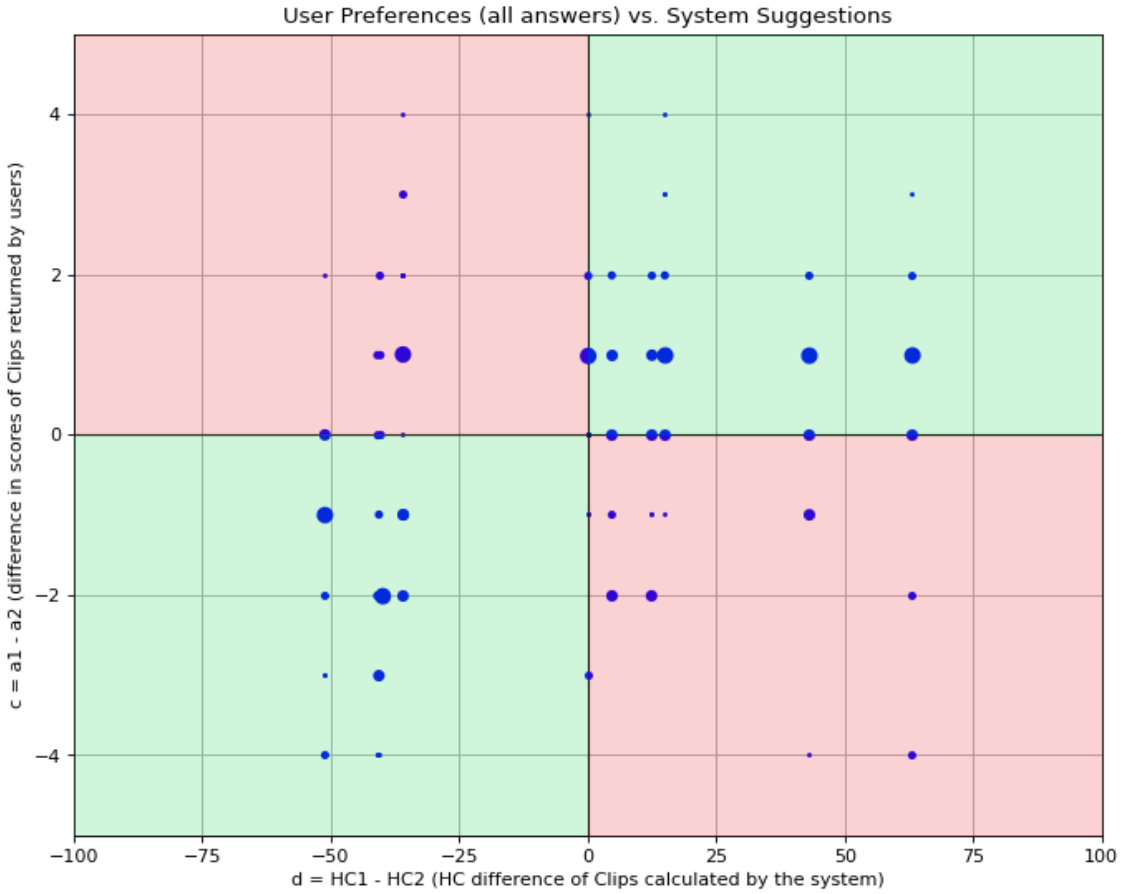


Figure 20: *All user responses in the experiment (calculated as the difference in scores between Clip 1 and Clip 2), plotted against the system suggestions (calculated as the difference in the HC value of Clip 1 and Clip 2). Bigger dots mean accumulation of equal responses in the same place. The system used for this graph implements FEM #10, described in detail in table 5. The dots in the green area indicate that the system's suggestions are aligned with users' preferences. Dots in the red zone indicate the opposite.*

what defines us two regions. The top right quadrant (where d ranges from 0 to 100% and c from 0 to 5) and the bottom left quadrant (where d ranges from -100 to 0% and c from -5 to 0), represents the area where the suggestions of our system match well the user preferences. We give to that region a green colour on the graph. But the top left quadrant (with d from -100% to 0% and c from 0 to 5) and the bottom right (d form 0% to 100 and c from -5 to 0), are the areas of the graph where the suggestions do not match the user preferences, represented with red on the graph.

Considering the number of scores in the green area, over the total of answers N, we

can define the True Positive Rate (TPR), as in equation 5.4.

$$TPR = \frac{\sum_u \sum_e [c_{e,u} > 0][d_e > 0] + \sum_u \sum_e [c_{e,u} < 0][d_e < 0]}{N} \tag{5.4}$$

However, while the experiment was designed to induce participants to rate one Clip above the other, we observed that in some of the responses the scores are equal. In these cases, it is impossible to assess whether our system provided correct suggestions, as the HC values returned by the system are always different for Clip 1 and 2 (and the difference is never equal to zero). For this reason, we will correct the TPR by eliminating the responses where users scored Clip 1 and 2 the same, resulting in the corrected $TPRc$ of equation 5.5.

$$TPRc = \frac{\sum_u \sum_e [c_{e,u} > 0][d_e > 0] + \sum_u \sum_e [c_{e,u} < 0][d_e < 0]}{N - \sum_u \sum_e [c_{e,u} = 0]} \tag{5.5}$$

5.4 Chi-squared test

In statistics and applied statistics, the standard way of assessing whether two variables are related and dependent is by means of the χ^2 test. In that sense, any test in which the statistic used follows a χ^2 (chi-squared) distribution if the null hypothesis is true, is called a χ^2 test.

Pearson's χ^2 test is considered a non-parametric test that measures the discrepancy between an observed and a theoretical distribution (goodness of fit), indicating the extent to which differences between the two, if any, are due to chance in the hypothesis test. It is also used to test the independence of two variables from each other by presenting the data in contingency tables.

The formula that gives the statistic is as follows:

$$\chi^2 = \sum_{d,c} \frac{(fo_{d,c} - fe_{d,c})^2}{fe_{d,c}} \tag{5.6}$$

$d \times c$	c_n	c_p
d_n	37	15
d_p	16	50

Table 6: *Contingency table with the results of the experiment (observed frequencies) using the suggestions of the system that incorporates the FEM #10. The values of $c = 0$ are discarded.*

Where fo is the observed frequency given by the answers of our experiment, and fe the expected frequency computed as in equation 5.7.

$$fe_{d,c} = \frac{\text{(Total d-th row)(Total c-th column)}}{\text{Overall total}} \tag{5.7}$$

If the test statistic is improbably large according to that chi-squared distribution, then one rejects the null hypothesis of independence.

The higher the χ^2 value, the more likely the variables will be related. Similarly, the closer the χ^2 to zero, the more different both distributions are. To study the independence of our variables, we perform a hypothesis test, where the null hypothesis (H0) implies that the variables are independent, and the alternative hypothesis (H1) that the variables are related.

If H0 is true, χ^2 follows a Chi-squared distribution with $(r-1)(k-1)$ degrees of freedom, where r is the number of rows and k the number of columns in our contingency table. We reject H0 when:

$$\chi^2_{experimental} > \chi^2_{critic} \tag{5.8}$$

Where $\chi^2_{experimental}$ is computed from equation 5.6, and χ^2_{critic} is derived from the tabular χ^2 theoretical model, for certain degrees of freedom and a level of risk α that we are willing to assume.

Applying these ideas to our particular case, let us define the variables and parameters of our experiment. Performing the same study of regions as in the previous section,

we will say that our variable d (which measures the score difference calculated by our system) can take positive (d_p) or negative (d_n) values. Similarly, the variable c (which measures the difference in participants' scores) can take positive (c_p) or negative (c_n) values. As in the previous case, we discard the values of $c = 0$, as we cannot compare them directly with the suggestions of our system.

Based on the contingency table 6 of our experiment, which has 2 columns and 2 rows, our statistic has one degree of freedom. We then assume an $\alpha = 0.05$, which represents a 5% risk in our estimation. In this case $\chi^2_{critic,1,0.05} = 3.8415$.

After calculating $\chi^2_{experimental}$ and comparing it with χ^2_{critic}, we can check whether our two variables are related.

5.5 Results

In section 3.5 we have presented *Perf*, a measure of how effectively our system replicates the suggestions of the traditional key notation method. In section 5.3, we have presented *TPRc*, that is the proportion of users' preferences that are aligned with the system's suggestions, considering a binary scenario. And finally, in section 5.4 we have presented χ^2, a statistical model that allows us to know whether the two variables are related.

In table 7 we present the performance results of the systems incorporating the FEMs #6, #7, #8, #9 and #10 (presented earlier in table 5) evaluated with the three metrics just mentioned.

5.6 Discussion

The system development stage (described in more detail in Chapter 3) preceded the system evaluation in this project. For this reason, we optimised the selection of parameters using *Perf*. We then performed the experiments with users in order to verify our previous choice. However, if we had reversed the order, doing the user experiments first, it would have been possible to optimise the system to the users'

	FEM#6	**FEM#7**	**FEM#8**	**FEM#9**	**FEM#10**
Perf	10.77	10.68	10.44	10.68	11.26
TPRc	63.3%	58.5%	66.1%	71.2%	73.7%
$\chi^2_{experimental}$	13.8290	16.2662	12.5345	19.9428	36.6717
H0	Rejected	Rejected	Rejected	Rejected	Rejected

Table 7: *Comparative performance results of the different FEMs. Perf represents how well the system replicates the suggestions of traditional key notation method; TPRc indicate the amount of suggestions that improved the mix.* $\chi^2_{experimental}$ *is the value of the contrast statistic. H0 is rejected when* $\chi^2_{experimental} > 3.8415$*, indicating that the two variables are related. The references of the architecture of each FEM can be found in table 5.*

preferences directly. Fortunately, either *Perf*, *TPRc* and χ^2 agree on the optimal system architecture.

This experiment involved the participation of 15 musically trained people. To correctly generalise the results obtained in our experiment, we should have involved many more users. This would allow us to compare the results of our implementation with those of other researchers.

Although the design of the experiment is intended to induce participants to rate one Clip above the other, in 21% of the cases participants rated both Clips with the same score. This implies that they were not able to tell the difference between the two. However, our system (which represents HC with a real number) always assigns different values to the two Clips in the example. For this reason we analysed the results using two different metrics to account for this phenomenon. Considering that there should not be equal answers, *TPRc* measures the amount of suggestions that are validated by users, and χ^2 allows us to know the degree of adjustment between both variables, and provides us with a decision criterion to determine whether they are related.

Finally, from the individual tracks analysis, we observe that for all implementations, there is always one particular example that the suggestions of our system clearly contradict the preferences of the users. Listening carefully to this example, the two mixes sound good from a harmonic point of view. However, we notice that the

discrepancy in this example is not explained by the HC, but by a phenomenon that we did not consider when making the mixes, which is tonal tension. In music there are a series of harmonic progressions that create a tonal tension curve. At the Clip that users perceive as more consonant, there is less tonal change, which creates a relaxation that we perceive as familiar. However, in the other Clip better rated by our system, the tonal shift is greater and goes towards a note that goes from relaxation to tension, standing in suspension. The Clip in this example ends before it goes to a rest chord.

The duration of the examples had to be limited so as not to fatigue the test participants. In most cases, examples of 32 beats are sufficient for them to appreciate harmonic compatibility. However, in cases such as the example above, it would have been desirable to allow the user to listen to the mix for a longer time, so that they could notice the stabilisation of the tonal tension.

Chapter 6

Conclusions and future work

This report describes the design and implementation of the Harmonic Mix System, a software package that can estimate the harmonic compatibility between digital music recordings, with a particular focus on modern dance music and the workflow of the DJ. This final section brings together the conclusions of the project, and considers opportunities for further work on the problem.

6.1 Contributions

In this project, we have explored the characteristics of using a new metric for harmonic mixing adapted to the needs of the DJ. Currently, the most widely used technique for harmonic mixing is solely based on key notation. However, there is sufficient evidence that this technique has limitations and it is convenient to complement it with small-scale compatibility information.

Recently, several researches have been published using new techniques based on TIVs, but so far this knowledge has not been brought closer to the final user. With our proposal, we aim to provide the DJ with more information for decision making. Our HC value is not intended to replace key notation but to complement it since both are important to understand the harmonic relationships between the tracks. While key signature, alone, does not provide enough information regarding harmony, the

HC lacks information about melody (i.e., a particular HC value can be due to two different keys). Therefore, the HC helps find the best harmonic matches between the tracks in the collection, and the key in guaranteeing control over the overall harmonic structure of the mix.

We have described the basic building blocks of a system that extracts harmonic information based on TIVs and have exposed the most promising current strategies based on the literature. Regarding the state-of-the-art HC calculation based on TIV, to the best of our knowledge, this is the first implementation using source-separation and NNLSchroma.

We have tested different system architectures and evaluated them using a metric that allows us to replicate the performance of the traditional key notation method, taking as a reference the harmonic ratios derived from the circle of fifths. To test some specific architectures suggested by literature, we constructed a dataset of 48 EDM songs with key annotations and switch points. This allowed us to discard strategies for the calculation of HC based on the extraction of 32-beat loops from representative sections of the audio tracks.

Measuring the performance of a system that calculates HC is challenging due to the subjective nature of this measurement, and the limited precedents. Thus, we propose an innovative evaluation method that forces participants to judge HC by neutralising other factors that may affect users' perceptions. This method infers the users' preferences from the difference in the score they have given to two identical mixes, apart from a pitch transposition.

Our results indicate that the system architecture most preferred by users is also the one that best fits the traditional key notation method. The HMS generates pitch transposition suggestions that improve mixes in 73.7% of the cases.

The HMS has been implemented under a graphical user interface that any DJ can download to start incorporating HC into their live performances. Its design recreates the representation of the music collections of commercial DJ software.

6.2 Limitations

The initial idea of this project required us to create a dataset to start testing different implementations. As a result, a 24-track dataset of either Progressive House or Techno was obtained. Unfortunately, it may be too small a sample size to be confident in HMS's results, as it may have led to a bias in the development and selection of strategies that may limit its usefulness. Unfortunately, the time scale of this project did not allow for manual annotation of a larger selection of tracks, and there are apparently no annotated datasets of dance music with switch points, BPM, and key. Similarly, another limitation is the small number of participants in the experiment, which makes it difficult to generalise the conclusions of our work.

After observing the individual responses of the participants, we noticed that their musical background greatly influences their perception of HC (e.g. a trained ear for jazz music is not the same as a trained ear for Andean music). Thus, another limitation lies in the notion of the HC itself, where subjectivities are intended to be modelled. There are elements of our experiment that undoubtedly are missed, due to self-limiting constraints imposed by the subjectivity measurement. Each user's response is a combination of cultural, educational, and even physiological biases. It is not possible to appeal to an objectivity because, the beauty here is that, it does not exist.

We considered the data presentation design of our system as validated since it replicates the one-to-many mapping of commercial software. However, it would have been interesting if several DJs had used the system by means of the GUI we made available to gather their opinions on usability and the quality of the suggestions.

Some restrictions of the recently published `TIV.lib` library for HC calculation influenced the final architecture of the system. On the one hand, the incompatibility of `TIV.lib` with the chromatic vector NNLSChroma did not allow us to implement some architectures based on *TIV-framewise* or *TIV-beatwise* over the whole audio track. For this reason, we decided to work with a *TIV-whole* representation, where all the tonal information of the track is concentrated in a single vector. On the

other hand, a bug in the library causes the imaginary part of two of the seven numbers that constitute the TIV never update, so they are always equal to zero. This can limit the overall performance of the system, as the HC value is calculated from vectors that are not entirely right.

Our experiments revealed the difficulty our system has in generating suggestions considering local elements that may influence how we perceive the HC.

The key-based mixing method relies on the idea that a very high percentage of EDM tracks can be represented from a single key expressing the notes in the audio track. Similarly, our system (which implements FEM #10), calculates a single TIV from the total aggregate of all the track elements.

In dance floors where this type of music is usually played, tracks are overlapped to generate new music by superimposing and aggregating the different elements that compose them. The concept of scale, then, is defined texturally by the layering of different elements such as bass notes, piano or synthesiser chords, etc. The overlapping of notes is more important than the temporal order. Generally, there are no significant harmonic changes, and each track usually lasts between 5 and 10 minutes.

This characteristic of creating by aggregation is different from other genres of music. In Pop music, classical music, and in some particular genres of EDM (such as House), tonal appreciation is less related to the sum of the elements that make up the tonality, and more to the succession of these elements. In this case, tonality is perceived by the temporal combination of the different elements, where the horizontal order of the notes determines the identity of the track.

6.3 Future work

The project has highlighted some areas where further work would be beneficial. First, to avoid dissonant mixing of detuned tracks, the quantization effect of comparing whole semitone shifts should be considered. Our current implementation uses a chromatic vector of length 12, which is equivalent to tonal distances of one semitone. It would be desirable to use a length of 36 or more, so that the transposition

intervals are smaller than the semitone.

Perhaps it would make sense to explore new implementations that perform the TIV calculation on a smaller scale, using only the last bars of the audio output. In this way, the HC values would be permanently updated. The HC would be calculated between the current mix playing through the speakers, and each of the tracks within the music folder. The current mix can be a single track, an overlay of tracks, effects, etc. Our current implementation assumes that the mix is solely the result of the overlay of two tracks; but this is not true in all cases (e.g. when using 4 decks). On the other hand, by using a time window of only a few beats backwards, we can better generalise the system for those musical genres that contain tonal changes throughout the track like the one in the experiment. This real-time computation of the TIV presents the added difficulty of optimising the execution and computational cost of the algorithm.

It would be important to implement and test a FEM that incorporates a beattracker for the calculation of the *TIV-beatwise* over the entire duration of the track. According to the literature, this strategy has shown encouraging results, but due to compatibility problems, we have not been able to implement it. Its performance results should be compared to those of the other system architectures presented in this work.

Then, a sufficiently large database of user preferences regarding HC should be available to properly evaluate the systems presented in this paper and future systems that may emerge from other researchers. In addition, the statistical model for evaluating all these systems should be standardised. The experiment and the metrics proposed in this research can be considered as a precedent for more ingenious methods. Once the statistical framework on which to optimise the algorithms has been developed, and considering all the different strategies presented in this work for calculating HC from TIVs, deep learning models could be used to achieve system architectures unimagined by us.

As mentioned in Section 2.1.1, with the incorporation of KEY SYNC, the industry

is showing a clear intention to move towards the automation of harmonic mixing, in the same way as it did previously with the automation of time alignment or beatmatching. In this sense, it would be desirable that in the future our system would be able to automate pitch transposition, so that the resulting mix would be harmonic. The DJ would only have to select the track he/she wants to play and the computer would automatically align the tracks in time and frequency. The user would also have the possibility to determine the maximum allowed transposition interval for his mixes. For an optimal implementation of this functionality, highly percussive music could benefit from improved pitch transposition algorithms.

It would be interesting to see this improvement incorporated into commercial products or currently available systems, which would really open up possibilities to mass test the suitability of this harmonic compatibility measure in real case scenarios.

List of Figures

List of Tables

Bibliography

[1] Faraldo Pérez, Á. *et al. Tonality estimation in electronic dance music: a computational and musically informed examination.* Ph.D. thesis, Universitat Pompeu Fabra (2018).

[2] Native Instruments. Traktor pro 3: User manual (2019). URL `https://www.native-instruments.com/fileadmin/ni_media/downloads/manuals/traktor/TRAKTOR_PRO_3.2_Manual_English_0719.pdf`. Accessed on 23.02.2021.

[3] Mixed In Key. Camelot wheel: Harmonic mixing guide (2007). URL `https://mixedinkey.com/harmonic-mixing-guide/`. Accessed on 23.02.2021.

[4] Pioneer. Learn more about Key Shift and Key Sync (2019). URL `https://www.pioneerdj.com/es-419/product/features/controller/key-shift-and-key-sync/`. Accessed on 23.02.2021.

[5] Veire, L. V. & De Bie, T. From raw audio to a seamless mix: creating an automated dj system for drum and bass. *EURASIP Journal on Audio, Speech, and Music Processing* **2018**, 1–21 (2018).

[6] Gebhardt, R. B. & Margraf, J. Applying psychoacoustics to key detection and root note extraction in edm. In *Proc. of the 13th International Symp. on CMMR*, 482–492 (2017).

[7] Davies, M., Stark, A. M., Gouyon, F. & Goto, M. Improvasher: a real-time mashup system for live musical input (2014).

[8] Davies, M. E., Hamel, P., Yoshii, K. & Goto, M. Automashupper: Automatic creation of multi-song music mashups (2014).

[9] Lee, C.-L., Lin, Y.-T., Yao, Z.-R., Lee, F.-Y. & Wu, J.-L. Automatic mashup creation by considering both vertical and horizontal mashabilities. In *ISMIR*, 399–405 (2015).

[10] Gebhardt, R. B., Davies, M. E. & Seeber, B. U. Psychoacoustic approaches for harmonic music mixing. *Applied Sciences* **6**, 123 (2016).

[11] Davies, G. B. M. E. & Guedes, C. A perceptually-motivated harmonic compatibility method for music mixing. In *Proceedings of the 13th International Symposium on CMMR, Matosinhos, Portugal* (2017).

[12] Gebhardt, R., Davies, M. & Seeber, B. Harmonic mixing based on roughness and pitch commonality (2015).

[13] Bernardes, G., Cocharro, D., Caetano, M., Guedes, C. & Davies, M. E. A multi-level tonal interval space for modelling pitch relatedness and musical consonance. *Journal of New Music Research* **45**, 281–294 (2016).

[14] Johnson-Laird, P. N., Kang, O. E. & Leong, Y. C. On musical dissonance. *Music Perception: An Interdisciplinary Journal* **30**, 19–35 (2012).

[15] Pérez, M. Harmonic compatibility for loops in electronic music (2020, August). URL `https://doi.org/10.5281/zenodo.4091438`. Accessed on 23.02.2021.

[16] Zplane. Zplane company profile (2000). URL `https://products.zplane.de/company/profile`. Accessed on 23.02.2021.

[17] Native Instruments. N I company profile (2021). URL `https://www.native-instruments.com/es/company/company-profile/`. Accessed on 23.02.2021.

[18] Serato. Serato DJ 1.8 (2015). URL `https://serato.com/dj/pro/downloads/1.8.0/releasenotes`. Accessed on 23.02.2021.

[19] Pioneer. DDJ-1000SRT Dj Controller (2019). URL `https://www.pioneerdj.com/es-419/product/controller/ddj-1000srt/black/overview/`. Accessed on 23.02.2021.

[20] Pioneer. CDJ-3000 Professional Dj Multiplayer (2020). URL `https://www.pioneerdj.com/en-us/product/player/cdj-3000/black/overview/`. Accessed on 23.02.2021.

[21] Sha'ath, I. Estimation of key in digital music recordings. Tech. Rep., Tech Report, Birkbeck College, University of London (2011).

[22] Mixed In Key. Mashup 2 (2014). URL `https://mashup.mixedinkey.com/`. Accessed on 23.02.2021.

[23] Bernardes, G., Davies, M. E. & Guedes, C. A hierarchical harmonic mixing method. In *International Symposium on Computer Music Multidisciplinary Research*, 151–170 (Springer, 2017).

[24] Maçãs, C., Rodrigues, A., Bernardes, G. & Machado, P. Mixmash: A visualisation system for musical mashup creation. In *2018 22nd International Conference Information Visualisation (IV)*, 471–477 (IEEE, 2018).

[25] Maçãs, C., Rodrigues, A., Bernardes, G. & Machado, P. Mixmash: An assistive tool for music mashup creation from large music collections. *International Journal of Art, Culture and Design Technologies (IJACDT)* **8**, 20–40 (2019).

[26] Bernardes, G., Cocharro, D., Guedes, C. & Davies, M. Conchord: an application for generating musical harmony by navigating in a perceptually motivated tonal interval space. In *Proceedings of the 11th International Symposium on Computer Music Modeling and Retrieval (CMMR)*, 71–86 (2015).

[27] Kim, A., Park, S., Park, J., Ha, J. W., Kwon, T., Nam, J. Automatic dj mix generation using highlight detection. *Proc. ISMIR, late-breaking demo paper* (2017).

[28] Reddit user: u/bascurtiz. Key analysis comparison 800+ tracks 2019 (rekordbox, traktor, mik, keyfinder...) (2019). URL `https://www.reddit.com/r/DJs/comments/cbyxlb/key_analysis_comparison_800_tracks_2019_rekordbox/`. Accessed on 23.02.2021.

[29] Zplane. Elastique Pitch v1.3.3 (2014). URL `https://products.zplane.de/company/news/elastique-pitch?start=5`. Accessed on 23.02.2021.

[30] Serato. The Serato Story (2021). URL `https://serato.com/about`. Accessed on 23.02.2021.

[31] Pioneer. Company Vision (2021). URL `https://www.pioneerdj.com/company/company-info/#vision`. Accessed on 23.02.2021.

[32] Goto, M. An audio-based real-time beat tracking system for music with or without drum-sounds. *Journal of New Music Research* **30**, 159–171 (2001).

[33] Chew, E. *Towards a mathematical model of tonality.* Ph.D. thesis, Massachusetts Institute of Technology (2000).

[34] Harte, C., Sandler, M. & Gasser, M. Detecting harmonic change in musical audio. In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, 21–26 (2006).

[35] Bernardes, G., Cocharro, D., Guedes, C. & Davies, M. E. Harmony generation driven by a perceptually motivated tonal interval space. *Computers in Entertainment (CIE)* **14**, 1–21 (2016).

[36] Navarro, M., Caetano, M., Bernardes, G., de Castro, L. N. & Corchado, J. M. Automatic generation of chord progressions with an artificial immune system. In *International Conference on Evolutionary and Biologically Inspired Music and Art*, 175–186 (Springer, 2015).

[37] Huron, D. Interval-class content in equally tempered pitch-class sets: Common scales exhibit optimum tonal consonance. *Music Perception* **11**, 289–305 (1994).

[38] Ramires, A., Bernardes, G., Davies, M. E. & Serra, X. Tiv. lib: an open-source library for the tonal description of musical audio. *arXiv preprint arXiv:2008.11529* (2020).

[39] Cabral, G., Briot, J.-P. & Pachet, F. Impact of distance in pitch class profile computation. In *Proceedings of the Brazilian Symposium on Computer Music*, 319–324 (Citeseer, 2005).

[40] Munson, W. & Gardner, M. B. Standardizing auditory tests. *The Journal of the Acoustical Society of America* **22**, 675–675 (1950).

[41] Böck, S. & Davies, M. E. Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation. *Proc. of ISMIR (International Society for Music Information Retrieval). Montreal, Canada* 574–582 (2020).

[42] Zehren, M., Alunno, M. & Bientinesi, P. Automatic detection of cue points for dj mixing. *arXiv preprint arXiv:2007.08411* (2020).

[43] Ramoneda, P. & Bernardes, G. Revisiting harmonic change detection. In *Audio Engineering Society Convention 149* (Audio Engineering Society, 2020).

[44] Stöter, F.-R., Uhlich, S., Liutkus, A. & Mitsufuji, Y. Open-unmix-a reference implementation for music source separation. *Journal of Open Source Software* **4**, 1667 (2019).

[45] Gutierrez, E. G. *et al. Tonal description of music audio signals* (Citeseer, 2006).

[46] Mauch, M. & Dixon, S. Approximate note transcription for the improved identification of difficult chords. In *ISMIR*, 135–140 (2010).

[47] Bernardes, G., Davies, M. E. & Guedes, C. Automatic musical key estimation with adaptive mode bias. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 316–320 (IEEE, 2017).

[48] Beatport. Beatportal, 10 best-selling genres (2020). URL `https://www.beatportal.com/news/`

beatport-insider-2020-best-selling-tracks-artists-labels-and-genres/. Accessed on 15.08.2021.