Internship report

DJ mix reverse engineering using multi-pass non-negative matrix factorization

Author
Étienne ANDRÉ

Supervisors
Diemo SCHWARZ
Dominique FOURER

Abstract

Disc jockeys (DJs) create mixes by creatively combining existing music tracks, while applying various transformations such as time-warping and audio effects. DJ-mix reverse engineering involves computationally analyzing these mixes to extract the parameters used in their creation. Previous approaches have treated this process as two separate tasks: alignment and unmixing, which are difficult to generalize to the wide range of transformations that DJs employ. This report introduces an integrated approach for both tasks using a multi-pass Non-negative Matrix Factorization (NMF) algorithm that is able to extract arbitrary time-warping transformations and the mixing gains while being robust to noise. The method's effectiveness is evaluated both qualitatively, through representative examples, and quantitatively, using a publicly available dataset. Additionally, the report explores the challenges of developing suitable datasets for DJ mix reverse engineering, proposing potential approaches for dataset creation by ground truth measurement.

Résumé

Les disc-jockeys (DJ) créent des mixes en combinant des pistes musicales de manière créative, en appliquant diverses transformations telles que la manipulation temporelle ou des effets audio. La rétro-ingénierie de mix DJ consiste en l'analyse computationelle de ceux-ci afin d'en extraire les paramètres utilisés lors de leur création. Les approches antérieures ont traité ce processus comme deux tâches distinctes : l'alignement et le démixage, qui sont difficiles à généraliser à la large gamme de transformations employées par les DJ. Ce rapport introduit une approche intégrée pour les deux tâches en utilisant un algorithme de factorisation en matrices non-négatives (NMF) multi-passe capable d'extraire des manipulations temporelles arbitraires et le gain de mixage tout en étant robuste au bruit. L'efficacité de la méthode est évaluée à la fois qualitativement, à l'aide d'exemples représentatifs, et quantitativement, à l'aide d'un ensemble de données accessibles au public. En outre, le rapport explore les défis posés par le développement de jeux de données appropriés pour la rétro-ingénierie de mix DJ, en proposant de nouvelles méthodes pour la création de jeux de données par mesure de vérité terrain.

I would like to express my gratitude to:
Diemo and Dominique, for their continued mentorship and support;
Rémi, Roland and Geoffroy for their help and valuable insights;
Luis for their enthusiasm and keenness to share;
The band of merry IRCAM interns for making office days enjoyable.

Contents

Glossary and mathematical conventions

Track A recorded piece of music or song

Mix A creative combination of overlapping consecutive tracks, potentially transformed using various effects and techniques

Transition The overlapping segment where a track finishes and the next one starts playing

Fade-in, fade-out A way to introduce and de introduce tracks using a continuous gain increase or decrease

Cross-fade A transition made of the simultaneous use of a fade-out and a fade-in

DJ Disc-Jockey

DJ Deck A standalone unit that plays recorded media

DJ Mixer A specialized mixing console for DJs

Cue point the point of a track at which it is introduced in a mix

Time-warping, warping time transformation of an audio signal

- **Resampling** time-warping with transposition, akin to speeding a vinyl record up or down
- **Time-stretching** time-warping without transposition

DTW Dynamic Time Warping

NMF Non-negative Matrix Factorization

STFT Short-Time Fourier Transform

DJ-MIR DJ Music Information Retrieval

DAW Digital Audio Workstation

EQ Equalizer

SIFT Scale Invariant Feature Transform

- **Spectrogram** squared modulus of the STFT
- **Hop size** duration between two consecutive frames of the spectrogram
- **Window size** duration of the time segment used for a frame of the spectrogram
- **Overlap factor** ratio between the hop size and the window size

$t \in [1..T]$	Discrete time step in track context
$\tau \in [1..K]$	Discrete time step in mix context
$\odot$	Hadamard (term-wise) matrix product
$\delta_{a,b}$	Kronecker delta function: $\delta_{a,b} = \begin{cases} 1 \text{ if } a{=}b \\ 0 \text{ otherwise} \end{cases}$
$\mathbf{V}$	Mix spectrogram matrix
$\mathbf{W}$	Track spectrogram matrix
$\mathbf{H}$	Activation matrix
g	Mixing gain
f	Time-warping function

I. Introduction

I.1. The art of DJing

For decades, DJs (*Disc-Jockeys*) and the practice of DJing have played an integral role in shaping our musical landscape. DJing can be interpreted in various ways, this thesis defines it as the continuous playback of recorded media (*tracks*) to an audience—whether live, via radio or other broadcast media, or through recorded formats like *mixtapes* or streaming services.

A DJ's performance, often referred to as a *mix*, involves the careful selection and sequential playback of music tracks. However, DJing is not merely a passive process; it is a transformative art. DJs often overlap tracks, either to create entirely new musical compositions (*mashups*) or more commonly, to maintain a seamless flow between songs. The overlapping segments are known as *transitions*, with the simplest form being a cross-fade.

In the realm of dance music, these transitions are crafted to be as seamless as possible to maintain the energy and keep the audience dancing. DJs employ various techniques to achieve this, such as synchronizing the tempo and downbeats of overlapping tracks, manipulating EQ settings to highlight or diminish certain elements, and applying effects like reverberation and delay.

Traditional vinyl DJs are typically limited to selecting *cue points* — the specific time offset at which a track is introduced — and adjusting the playback speed, sometimes in extreme ways, as in the case of scratching. In contrast, modern digital DJ equipment offers a broader array of transformative tools, including transposition, time-warping, looping, and jumping between different sections of a track.

Moreover, DJ sets may include additional elements such as spoken word or sung vocals, rhythm machines, and sound effects like sirens or jingles, further enriching the auditory experience.

I.2. Prior art on DJ Music Information Retrieval

The pervasive influence of DJing in popular culture has made it a subject of interest in academic research. Indeed, a deeper understanding of DJ practices could significantly contribute to various fields, including musicology, cultural studies, music technology, and the automation of DJ techniques for both entertainment and commercial applications.

In particular, the field of DJ-MIR (DJ Music Information Retrieval) seeks to develop methods to extract metadata from recorded DJ mixes. This field has traditionally been divided into several key subtasks:

Identification Retrieving the playlist used to create the mix, as explored in [1]–[3];

Alignment Determining the start and end times of each track within the mix, along with the time-scaling factors, as in [4]–[10];

Unmixing Estimating and reversing the fade curves [11], [6], [12], [13], the EQ parameters [13], and any additional effects [14];

Content analysis Deriving metadata such as genre and indicators of various social and musical aspects of the mix, such as cue points [15], [16] or drops [17], [18].

Mix generation Automatically generating realistic mixes from a given playlist, as addressed in [19]–[21], [18], [22], [23]

I.3. Problem overview of DJ-mix reverse engineering

Within these subtasks, we focus our interest on the task of *DJ-mix reverse engineering*, as defined by D. Schwarz and D. Fourer [8]. This term encompasses the identification, alignment and unmixing subtasks described above. Given that identification is a well-explored problem space, we further narrow our focus on the alignment on unmixing tasks.

More formally, our goal is as follows:

Given:

- a recording of a DJ mix;
- the recordings of all the mix's constituting tracks;

Estimate:

- any temporal transformations applied to the tracks (cue points, play duration, time stretching, loops, jumps);
- the evolution of the mixing gain of each track
- any effects applied to the tracks/and or mix (distortion, compression, pitch-shifting, filtering, EQing…)

Previous research has typically treated alignment and unmixing as sequential, independent tasks. Alignment is generally accomplished through feature extraction or fingerprinting combined with dynamic time warping (DTW). Subsequently, time-scaling factors are estimated, transition regions are segmented, and the unmixing process follows.

This approach has shown to be effective for broadcast mixes and simpler DJ sets, where tracks are often assumed to played monotonously (i.e. without jumps and loops) and at a fixed speed to mitigate alignment issues in transition regions. However, it is precisely in these transition regions that DJs are likely to employ the most creative transformations and effects.

Therefore, we propose an integrated approach that treats alignment and unmixing as a single, conjoint tanscription task, enabling a more general understanding of the complexities inherent in DJ mixes.

II. Datasets for DJ-mix reverse engineering

In order to evaluate their methods, researchers are in the need of appropriate datasets. Despite the large amount of available mixes, thanks to decades of recorded mixes and specialized streaming services, they are rarely sufficiently and correctly annotated.

Appropriate datasets for DJ mix reverse engineering typically consist of mixes paired with their constituent tracks, accompanied by their respective time positions within the mix. Depending on the available annotations, these datasets may also include detailed information about the manipulations performed by the DJ during the creation of the mix, such as the volume levels for each track, time-stretching and transposition factors, the types and parameters of effects applied, and more. We operate under the assumption that all the transformations applied to the original tracks, if known, enable to reconstruct the resulting mix signal accurately.

II.1. Existing datasets

We identify two broad categories in which existing datasets can be classified:

Synthetic These consist of mixes generated according to a set of predefined rules.

Real These are derived from real-world mixes and are annotated either manually or automatically.

II.1.1. Synthetic datasets

Synthetic datasets offer the distinct advantage of precision and completeness, as the parameters used in the creation of the mixes are explicitly known and controlled. This allows for a high level of accuracy in analyses, as every aspect of the mix, from track selection to the application of effects, is documented by design. However, because these datasets are generated according to predefined rules, they may lack the diversity and complexity found in real-world DJ sets. This limitation can affect their ecological validity, meaning their ability to generalize to real-world scenarios may be compromised.

In the context of DJ-mix reverse engineering research, UnmixDB [24], which is based on the `mixotic` dataset from R. Sonnleitner, A. Arzt, and G. Widmer [2]), and the dataset from L. Werthen-Brabants, T. De Bie, and P. Simeone [6], are among the few accessible examples of synthetic datasets. These datasets are derived from copyleft tracks released on the Mixotic netlabel[1], which primarily features electronic music. As a result, the datasets may lack stylistic diversity, potentially limiting their applicability to broader DJ-MIR research that encompasses a wider range of musical genres and DJ practices. However, because UnmixDB is easily accessible and features a large amount of data, we used it for evaluation of our method.

[1] https://www.mixotic.net

II.1.2. Real datasets

Datasets derived from real DJ sets are valuable because they accurately reflect actual DJing practices, both past and present. Unlike synthetic datasets, they do not suffer from issues stemming from artificial generation, making them more representative of real-world scenarios. However, the use of real DJ sets presents several challenges. One of the primary obstacles is the need for time-intensive manual annotation or reliance on automatic annotation methods, which can be prone to errors.

The act of identifying a track from a DJ mix, known as colloquially as a *track ID*, is a recurring topic of interest within DJ culture, engaging both DJs and their audiences. This has led to the emergence of online platforms that host crowdsourced or automatically detected track IDs, such as 1001tracklists[2], CueNation[3], LiveTracklist[4], mixes.wiki (formerly known as MixesDB)[5] and TrackId.net[6].

These platforms collectively provide a vast amount of data, which can potentially be leveraged for data-mining activities and may be particularly useful for the content analysis aspect of DJ-MIR. However, the information available on these sites typically includes only track identification and approximate track positions within the mix. Furthermore, the audio content is often copyrighted, and the data itself may be subject to legal restrictions, which complicates its use in research. As a result, researchers must independently obtain the associated audio content, adding another layer of complexity to the process.

Nonetheless, this is how T. Kim, M. Choi, E. Sacks, Y.-H. Yang, and J. Nam [7] and T. Kim, Y.-H. Yang, and J. Nam [13] obtained large-scale datasets for evaluation of their methods, with the latter being available, albeit without audio data: the commercial nature of the mixes and the tracks within them raises legal concerns, particularly regarding their use in an academic setting. To circumvent this, audio data can be downloaded from streaming services, but then its random availability becomes a concern for study repetability.

Other noteworthy datasets are the one from T. Scarfe, W. M. Koolen, and Y. Kalnishkan [25], which has been obtained from radio show archives, and the `disco` dataset from R. Sonnleitner, A. Arzt, and G. Widmer [2] which features measured bespoke mixes. Both of these are not publicly available.

During the course of the internship, we obtained an extract of the TrackId.net database with the assistance of its owners. However, we found that this dataset was of limited interest for our purposes, but that it could be valuable for future research. A more detailed analysis of this dataset is provided in Appendix A.

II.2. Approaches for dataset creation

An ideal dataset for DJ-mix reverse engineering would consist of real-world DJ sets that are both diverse in the styles represented and annotated with precise and comprehensive

[2] https://www.1001tracklists.com
[3] https://cuenation.com/
[4] https://www.livetracklist.com/
[5] https://www.mixes.wiki/w/Main_Page
[6] https://trackid.net/

ground truth data. Achieving such a dataset is challenging, as current automatic annotation methods fall short in providing the necessary accuracy and completeness.

With this in mind, we suggest that ground truth may be measured during the DJ mixing process. We detail some approaches that could be leveraged to this extent.

II.2.1. Optical tracking of vinyl records

We implemented an optical tracking method to measure the angular velocity of a vinyl record during playback, enabling the extraction of the time-warping ground truth from a filmed DJ performance. The method can be especially useful given the vast amount of videos of DJ performances online. By tracking a reference picture of a vinyl record to each frame of a video of a DJ turntable, the record's rotation speed can be estimated.

Tracking: The tracking is based on the scale-invariant feature transform (SIFT) algorithm [26]. The algorithm extracts keypoints from a reference image of the vinyl's label (Figure 1) and from each frame of a video of the vinyl being played. By matching these keypoints, a homography matrix is computed for each frame. This matrix represents the geometric transformation between the reference and the current frame, capturing the rotation, translation, and scale changes (Figure 2). The sequence of homography matrices obtained across the frames is stored for further analysis.

Rotation computation: The stored homography matrices are then decomposed to extract the rotation angles of the vinyl record using the singular value decomposition method from O. Faugeras and F. Lustman [27]. The rotation on the z axis corresponds to the rotation of the vinyl record. The rotation angle sequence is unwrapped then derived to obtain the rotation speed. Then, with knowledge of the nominal speed of the record (e.g. 33 or 45 rpm), the time-warping function can be calculated. Example results are illustrated in Figure 3.

The accuracy of the extracted rotational speed is directly dependent on the temporal resolution of the video (usually 30 or 60 images per second), which limits the granularity of the

Figure 1: Reference image of the vinyl label.

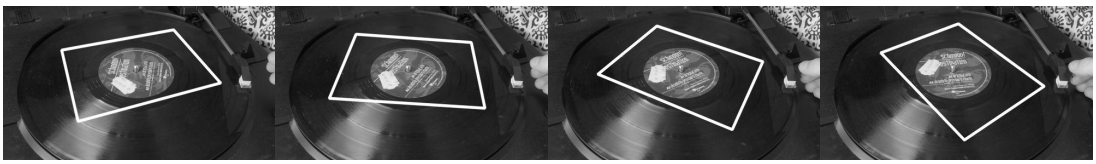

Figure 2: Frames from the vinyl video. Matches' bounding boxes are shown as white rectangles.

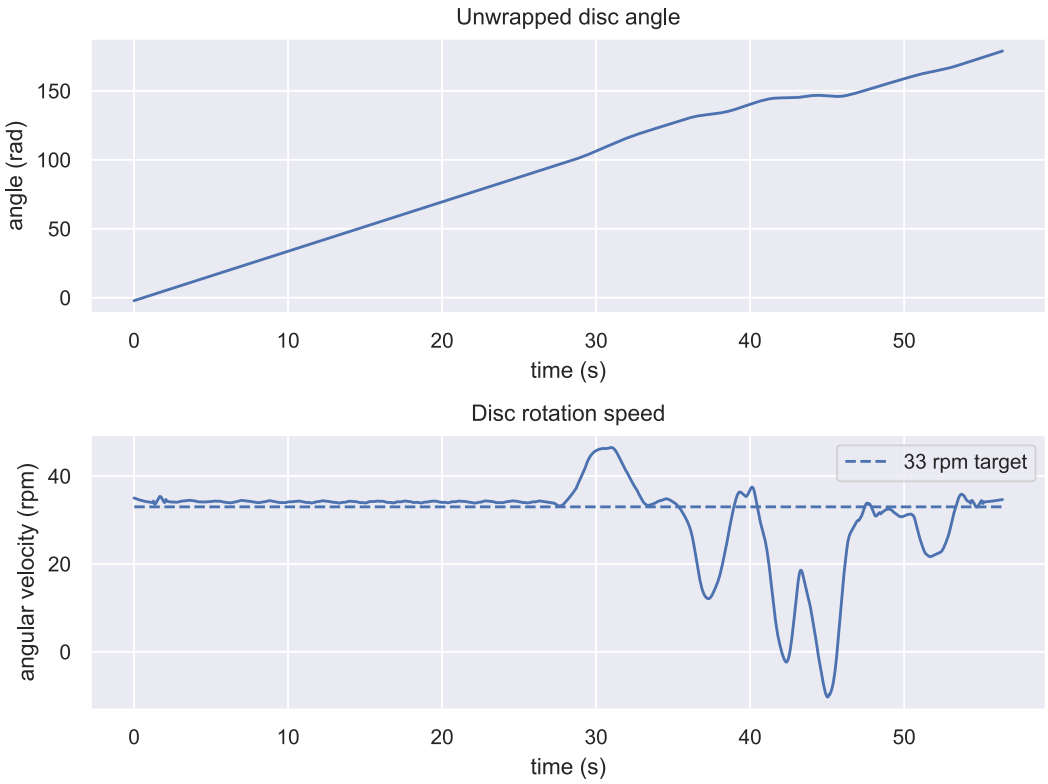

Figure 3: Computed angle and rotation speed of the vinyl. In the experiment, the record was played at 33 rpm for the first 30 seconds, then was "scratched" by hand for the remaining of the experiment.

measurements. Additionally, the method is susceptible to noise introduced by the tracking process, which can affect the precision of the rotation and speed calculations.

Similar object tracking techniques could be used to extract ground truth data from any visible physical control of the DJ equipment.

II.2.2. Metadata extraction from Digital Audio Workstation project files

DJ mixes are not only created in live settings but can also be constructed in studio environments using Digital Audio Workstations (DAWs). In such cases, an audio engineer can replicate a DJ performance offline by leveraging the DAW's audio editing and automation features. The resulting DAW project files then contain the reference tracks along with all the ground truth data required to generate the final mix audio file. This data can be programmatically extracted and processed to suit the needs of DJ-MIR research.

This approach has been notably previously explored by L. Werthen-Brabants, T. De Bie, and P. Simeone [6] for the creation of their dataset. Their tool, the *Ableton Live Mix Extractor*[7], is designed to extract ground truth data from *Ableton Live* project files, focusing on scenarios with fixed tempo, fixed stretch factors, and simple crossfades. But the *Ableton Live*

[7] https://github.com/werthen/ableton-extraction-tool

DAW supports many additional features that could be used to further imitate real DJ mixes, namely:

- Non-constant tempo curves;
- Non-constant time-warping;
- Complex effect chains with evolving parameters.

We have developed an enhanced version of this extraction tool. Our version supports the extraction of all aforementioned parameters and computes high-level ground truth data relevant to DJ-mix reverse engineering tasks. As a case study, we produced an example mix, illustrated in Figure 4. The mix features:

- Two tracks ("A" and "B") mixed together;
- Complex gain automation on both tracks, including:
  - Fade-in and fade-outs on both "A" and "B" (depicted with darker, curved backgrounds);
  - Multiple gain automations on "A" at both the mixer (first red line) and effect chain stages (second red line);
- Complex time manipulation:
  - Variable tempo curve (bottom red line);
  - Non-constant clip warping on "A" (yellow warp markers).

The extracted gain and time-warp data are illustrated in Figure 5, demonstrating that all the applied transformations have been captured[8].

Due to time constraints, our tool remains in a proof-of-concept stage, as its correctness has not been thouroughly evaluated.

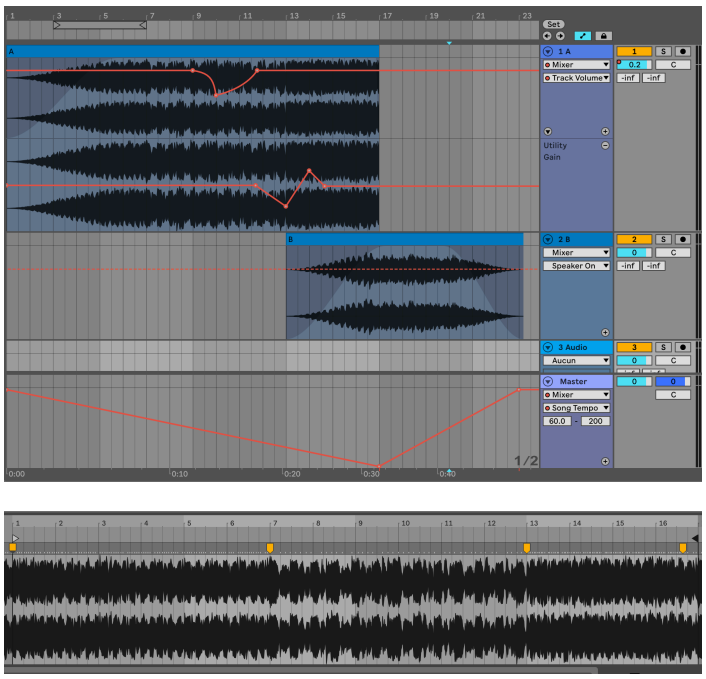

Figure 4: Example mix created in Ableton Live.
Top: session view with clips and automations.
Bottom: warp markers of the first clip.

[8]The events in Figure 4 do not visually align horizontally with Figure 5, as the time axis in the former is non-linear.

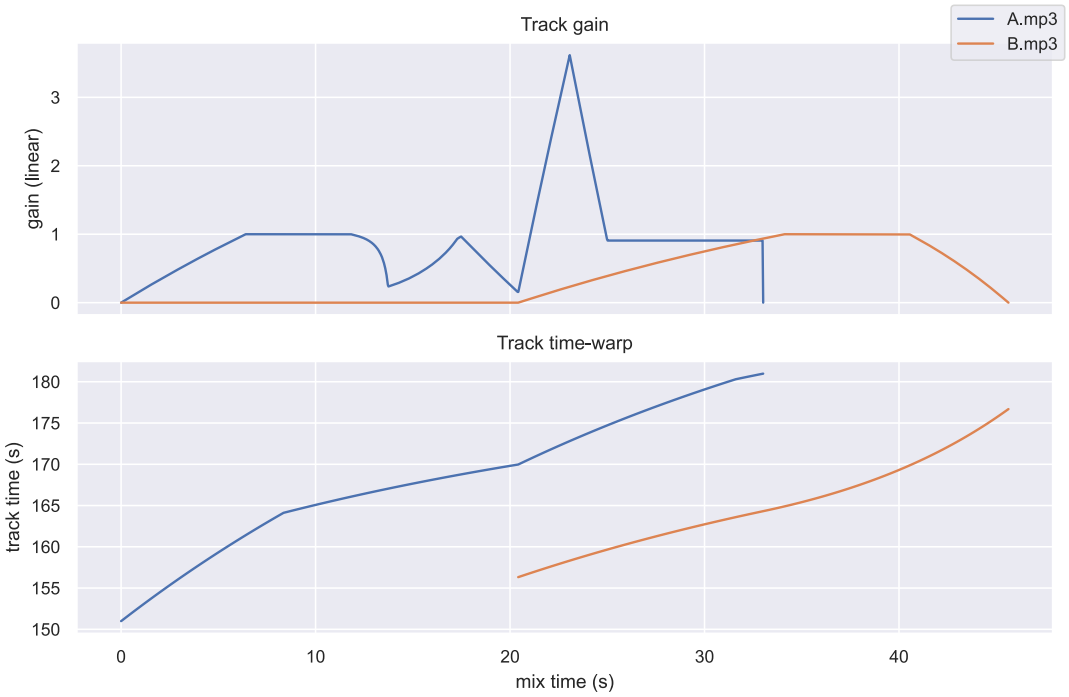

Figure 5: Extracted ground truth data from the example mix.

II.2.3. Instrumentation of DJ mixing software

Building on the approach outlined in the previous section, specialized DJ mixing software could similarly be exploited for dataset creation. DJ software is typically modeled after traditional DJ hardware, aiming to replicate the performance capabilities of physical equipment in a digital environment. Since all audio processing occurs within the software, it is theoretically possible to instrument the software to record ground truth data as a DJ mix is performed.

We identified *Mixxx*[9] as a suitable candidate for this approach due to its open-source nature, which allows for modification. By patching the software to include data-logging capabilities, it would be possible to simultaneously capture the mix, reference tracks, and associated ground truth data.

While we were unable to pursue this concept due to time and resource constraints, we believe that it holds significant potential for creating a real-world dataset with precise ground truth.

[9] https://mixxx.org

III. DJ mix transcription

In this section, we introduce a new application of NMF algorithm to perform DJ mix transcription. We first study DJ hardware and software to justify the transcription task as a matrix factorization problem, and introduce the base Beta-NMF algorithm. We then show that the matrix factorization can yield an intuitive representation of DJ mix parameters. We propose a multi-pass extension of the NMF algorithm that greatly improves its performance, and discuss additional modifications. We then present example results and evaluate our method on a publicly available dataset.

III.1. DJ mixing hardware

The DJs' field of expression is defined by its hardware and/or software: decks, mixing tables, and controllers. Despite the diversity in brands and models, which offer varying feature sets, the underlying workflow remains consistent across different setups. This consistency allows us to generalize the signal path from recorded tracks to the final mixed output, as depicted in Figure 6.

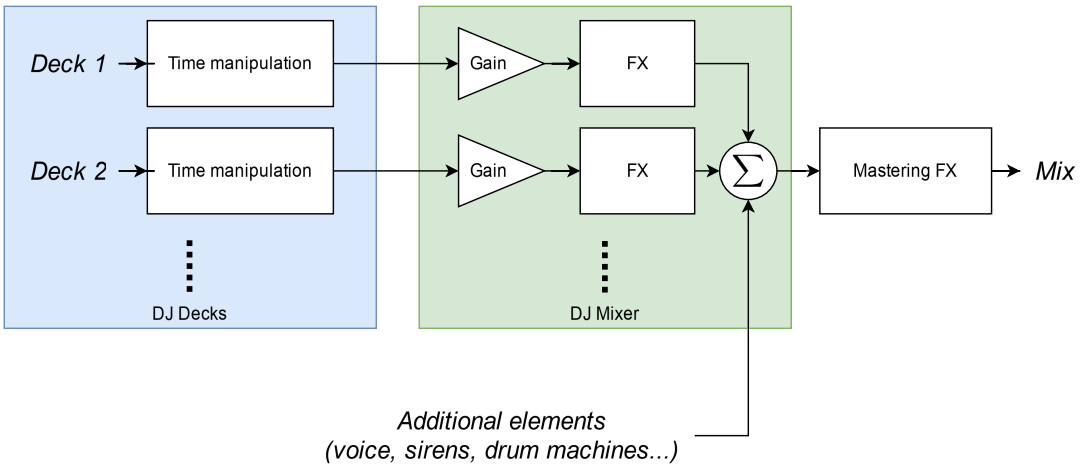

Figure 6: Schematic view of the DJ mixing process

The signal path illustrated is directly derived from standard DJ setups, encompassing both hardware and software environments.[10] The process can be described as follows:

- Two or more DJ decks (in blue) are used as signal sources and play pre-recorded tracks and apply time-warping.
- The signal from the decks is routed to the DJ mixer, which performs a weighted sum of the input signals. The mixer may also apply various effects, the most prevalent being a 3- or 4-band equalizer (EQ). Additional elements, such as external audio sources or audio effects, are also integrated at this stage.
- Post-mixing, additional processing might be applied to the mixed output to meet specific distribution or venue requirements. This processing typically involves light modifications such as compression and equalization. However, given the minimal nature of these mod-

[10]It is noteworthy that DJ software is typically designed to emulate the functionality of traditional DJ hardware, thereby preserving the validity of this signal path.

ifications, they will be considered negligible and thus omitted from further discussion in this report.

III.2. Matrix representation of the DJ mixing process

We now use this knowledge to introduce a matrix formulation of DJ mixing, by considering a spectrogram reprensentation of the signals. We achive this by grouping all non-time-based transformations, and modeling any additional elements and timbral effects as additive noise, as illustrated in Figure 7.

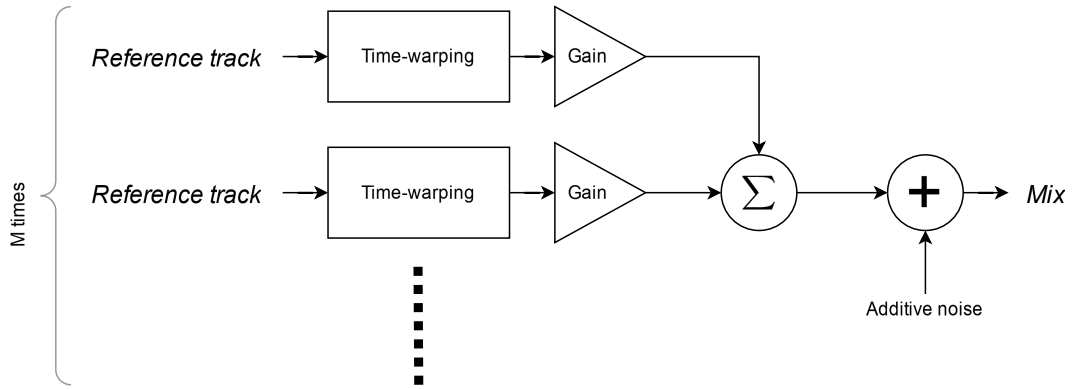

Figure 7: Separated signal path

Assuming the M constituting tracks of the mix are known, let $\forall i \in [1...M]$:

- $\mathbf{W}_{(i)}$ the spectrogram of track i;
- $\mathbf{H}_{(i)}$ the so-called *activation matrix* of track i, representing both the time-warping operations and the gain applied at the mixing stage;
- $\mathbf{V}_{(i)} = \mathbf{W}_{(i)}\mathbf{H}_{(i)}$ the time-remapped spectrogram of track i with gain appplied;
- $\mathbf{N}$ a noise matrix representing the timbral changes applied to the mix and/or tracks and any additional elements;
- $\mathbf{V}$ the spectrogram of the mix.

Using these notations, we can write:

$$\mathbf{V} = \mathbf{N} + \sum_{i=1}^{M} \mathbf{W}_{(i)}\mathbf{H}_{(i)} \tag{1}$$

An computation diagram of Equation 1 is given Figure 8.

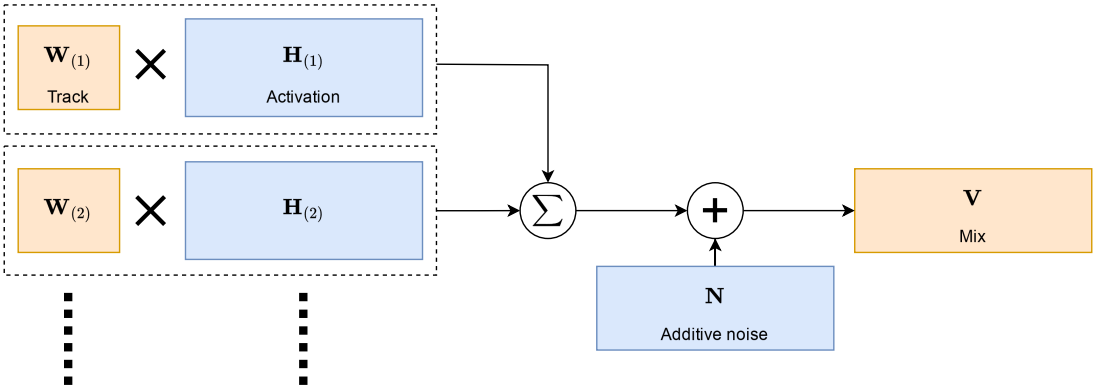

Figure 8: Matrix form of the DJ mixing process

Then, by defining two additional matrices $\mathbf{W}_{(a)}$ and $\mathbf{H}_{(a)}$ of compatible dimensions so that $\mathbf{N} = \mathbf{W}_{(a)}\mathbf{H}_{(a)}$, we can rewrite Equation 1 as a simple matrix multiplication of two large matrices by concatenation:

$$\begin{aligned}\mathbf{V} &= \mathbf{W}_{(a)}\mathbf{H}_{(a)} + \sum_{i=1}^{M} \mathbf{W}_{(i)}\mathbf{H}_{(i)} \\ &= \underbrace{\left(\mathbf{W}_{(1)}\mathbf{W}_{(2)}\cdots\mathbf{W}_{(M)}\mathbf{W}_{(a)}\right)}_{\mathbf{W}} \underbrace{\begin{pmatrix}\mathbf{H}_{(1)}\\ \mathbf{H}_{(2)}\\ \vdots\\ \mathbf{H}_{(M)}\\ \mathbf{H}_{(a)}\end{pmatrix}}_{\mathbf{H}}\end{aligned} \tag{2}$$

Because we assume that the constituent tracks of the mix are known, the ($\mathbf{W}_{(1)}$ to $\mathbf{W}_{(M)}$ submatrices are known. Transcribing the DJ mix then amounts to determining the other coefficients of Equation 2:

- If the noise matrix is assumed to be zero, estimating the gain and time-warping amounts to determining the coefficients of the $\mathbf{H}$ matrix while keeping the $\mathbf{W}$ matrix fixed.
- If not, only part of the $\mathbf{W}$ matrix ($\mathbf{W}_{(1)}$ to $\mathbf{W}_{(M)}$ submatrices) is kept fixed, while the $\mathbf{W}_{(a)}$, $\mathbf{H}_{(1)}$ to $\mathbf{H}_{(M)}$ and $\mathbf{H}_{(a)}$ submatrices are estimated.

This can be understood as a matrix factorization problem. It is well-suited to the NMF family of algorithms, which has proven especially effective in audio source separation tasks, which we present in the next section.

III.3. NMF Algorithm

III.3.1. Beta-NMF and Multiplicative Update rules

Let $\mathbf{W} \in \mathbb{R}_+^{F\times K}$, $\mathbf{H} \in \mathbb{R}_+^{K\times N}$ and $\mathbf{V} \in \mathbb{R}_+^{F\times N}$ non-negative matrices. The NMF algorithm in its most basic form aims to minimise a similarity measure $\mathcal{D}$ between the *target matrix* $\mathbf{V}$ and the *estimated matrix* $\mathbf{WH}$, and amounts to solving the following optimization problem:

$$\min_{\mathbf{W},\mathbf{H}} \mathcal{D}(\mathbf{V} \mid \mathbf{WH}) \text{ with } \mathbf{V} \geq 0, \mathbf{W} \geq 0, \mathbf{H} \geq 0 \tag{3}$$

The similarity measure we use is the beta-divergence, which is defined $\forall \beta \in \mathbb{R}$ as follows:

$$\mathcal{D}_\beta(\mathbf{V} \mid \mathbf{WH}) = \sum_{f=1}^{F} \sum_{n=1}^{N} d_\beta\Big(\mathbf{V}_{fn} \mid (\mathbf{WH})_{fn}\Big) \tag{4}$$

$$d_\beta(x \mid y) = \begin{cases} \frac{1}{\beta(\beta-1)}(x^\beta + (\beta-1)y^\beta - \beta x y^{\beta-1}) & \text{if } \beta \neq \{0,1\} \\ x \ln \frac{x}{y} - x + y & \text{if } \beta = 1 \\ \frac{x}{y} - \ln xy - 1 & \text{if } \beta = 0 \end{cases} \tag{5}$$

It can be noted that the beta-divergence is equivalent to:
- the Euclidian distance if $\beta = 2$;
- the Kullblack-Leibler divergence if $\beta = 1$;
- the Itakura-Saito divergence if $\beta = 0$.

As shown in [28] and later extended in [29], an efficient and simple gradient descent algorithm for $\mathbf{W}$ and $\mathbf{H}$ can be derived if the gradient of the divergence w.r.t. a parameter $\boldsymbol{\theta}$ is separable into its positive and negative parts:

$$\nabla_{\boldsymbol{\theta}} \mathcal{D}_\beta(\mathbf{V} \mid \mathbf{WH}) = \nabla^+_{\boldsymbol{\theta}} \mathcal{D}_\beta(\mathbf{V} \mid \mathbf{WH}) - \nabla^-_{\boldsymbol{\theta}} \mathcal{D}_\beta(\mathbf{V} \mid \mathbf{WH}) \tag{6}$$

Using the notation trick described in [29], the so-called *multiplicative update* rules can be obtained[11]:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} \odot \frac{\nabla^-_{\boldsymbol{\theta}} \mathcal{D}_\beta(\mathbf{V} \mid \mathbf{WH})}{\nabla^+_{\boldsymbol{\theta}} \mathcal{D}_\beta(\mathbf{V} \mid \mathbf{WH})} \tag{7}$$

With the beta-divergence, this yields the update rules of Algorithm 1 which can be efficiently implemented, with strong monotonicity guarantees when $\beta \in [0, 2]$.

An interesting property of this algorithm is that any zeroes in $\mathbf{H}$ or $\mathbf{W}$, by property of multiplication, remain zero troughout the optimization process. We will exploit this property in Section III.5.

More complex objective functions can be crafted by adding supplementary functions to the similarity measure, for penalization or regularization of the solutions. We detail the principle and some penalty functions considered for DJ mix transcription in Appendix B.

[11] $\odot$ and $\frac{\cdot}{\cdot}$ stand respectively for Hadamard's (element-wise) product and division.

Algorithm 1: NMF Algorithm with Multiplicative Updates

Initialize $\mathbf{W} \geq 0$ and $\mathbf{H} \geq 0$

Until convergence criterion is reached:

$$\mathbf{H} \leftarrow \mathbf{H} \odot \frac{\mathbf{W}^T\left((\mathbf{WH})^{\beta-2} \odot \mathbf{V}\right)}{\mathbf{W}^T(\mathbf{WH})^{\beta-1}}$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\left((\mathbf{WH})^{\beta-2} \odot \mathbf{V}\right)\mathbf{H}^T}{(\mathbf{WH})^{\beta-1}\mathbf{H}^T}$$

III.3.2. Choosing the divergence and the type of spectrograms

Previous research empirically shown that the Kullback-Leibler divergence yields favorable results in source separation tasks when applied to magnitude spectrograms [30]. Conversely, pairing the Itakura-Saito divergence with power spectrograms has been shown to perform well, supported by a robust statistical model [31].

Additionally, D. FitzGerald, M. Cranitch, and E. Coyle [32] explore the use of fractional values for the parameter beta in source separation and suggests that spectrograms could be raised to a fractional power. This approach is further refined by the introduction of the tempered beta-divergence, where beta acts as a temperature parameter that varies during the optimization process. However, despite these theoretical advancements, we found no conclusive evidence in the existing literature regarding the optimal approach for our specific task.

Ultimately, after conducting our own experiments, we selected the Itakura-Saito divergence applied to power spectrograms, as this combination consistently produced the best results in our trials.

III.4. Characterization of warp and gain transformations

In this section, we present a model for time-warping and applying gain to a signal within the spectral domain. We demonstrate that a specific solution for the activation matrix exists, which reveals intuitive structural properties. Following this, we define estimators for both gain and time-warping, and conduct an analysis of their robustness in the presence of noise and other sources of uncertainty.

III.4.1. The ideal kernel

Let $x[t]$ be a real-valued signal, and define the following:
- $f : \tau \mapsto t$ a time-warping injective function that maps a mix time step τ to a track time step t;
- $g[\tau]$ a gain factor sequence;
- $y[\tau]$ the time-warped (by f) and gain-modulated (by g) transformation of x;
- w an arbitrary window function of length M.

We define $\mathbf{X} = (\mathbf{X}_{mt})$ as the spectrogram matrix of x (M frequency bins $\times$ T time steps):

$$\mathbf{X}_{mt} = \left| \sum_{n=1}^{M} x[n+t]w[n]e^{-j2\pi n\frac{m}{M}} \right|^2 \tag{8}$$

Similarly, we define $\mathbf{Y} = (\mathbf{Y}_{m\tau})$ as the power spectrogram matrix of y (M frequency bins $\times$ K time steps). We show that $\mathbf{Y}$ can be expressed in terms of $\mathbf{X}$ as follows:

$$\begin{aligned} \mathbf{Y}_{m\tau} &= \left| g[\tau] \sum_{n=1}^{M} x[n+f[\tau]]w[n]e^{-j2\pi n\frac{m}{M}} \right|^2 \\ &= g[\tau]^2 \mathbf{X}_{m,f[\tau]} \end{aligned} \tag{9}$$

We can then find a matrix $\mathbf{H} = (\mathbf{H}_{t\tau})$ (of dimensions T time steps $\times$ K time steps) that satisfies:

$$\mathbf{Y} = \mathbf{X}\mathbf{H} \Leftrightarrow \mathbf{Y}_{m\tau} = \sum_{t=0}^{T-1} \mathbf{X}_{mt}\mathbf{H}_{t\tau} \tag{10}$$

The *ideal kernel* $\mathbf{H}^{\text{ideal}}$, a solution to Equation 10, is of particular interest. When viewed as an image, this matrix offers an intuitive understanding of the transformations applied to $x[t]$, as illustrated in Figure 9.

The ideal kernel is defined as:

$$\mathbf{H}^{\text{ideal}}_{t\tau} \stackrel{\text{def}}{=} g[\tau]^2 \delta_{t,f[\tau]} \tag{11}$$

where $\delta_{a,b}$ is the Kronecker delta function, defined by $\delta_{a,b} = \begin{cases} 1 \text{ if } a=b \\ 0 \text{ otherwise} \end{cases}$.

III.4.2. Estimation of the warp and gain values

We define the following estimators for the gain and time-warping functions:

$$\tilde{g}[\tau] = \sqrt{\sum_{t=1}^{T} \mathbf{H}_{t\tau}} \tag{12}$$

$$\tilde{f}[\tau] = \text{argmax}_{t \in [1 \dots T]} \mathbf{H}_{t\tau} \tag{13}$$

Intuitively, $\tilde{g}[\tau]$ represents the energy of a column of $\mathbf{H}$, while $\tilde{f}[\tau]$ corresponds to the position of its peak.

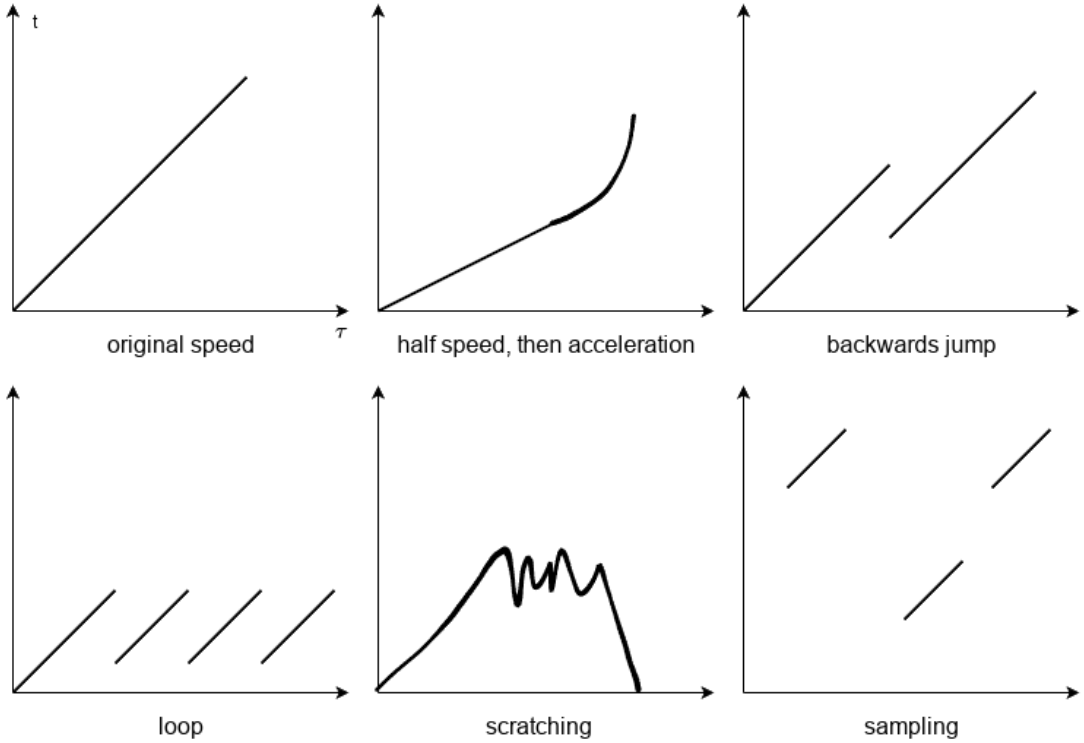

Figure 9: Some examples of the structures emerging in $\mathbf{H}^{\text{ideal}}$, with the associated DJ nomenclature.

In the case of the ideal kernel (Equation 11), it can be easily shown that $\tilde{f}$ and $\tilde{g}$ are exact estimators, meaning they perfectly recover the gain and time-warping functions. However, in practical scenarios, the optimization algorithm used to compute $\mathbf{H}$ does not inherently guarantee convergence to this ideal solution.

In practice, the NMF tends to converge towards the similarity matrix between y and x, rather than the idealized sparse solution with line features. This underscores the need to incorporate additional techniques that guide the algorithm towards convergence to the ideal solution, thereby ensuring that the estimators remain robust in the presence of noise and other uncertainties. We discuss briefly a few examples of such indeterminacies in the next section.

III.4.3. Sources of indeterminate forms

III.4.3.1. Impact of self-similar input signals

Given the nature of musical signals, two columns of $\mathbf{X}$ could be almost identical (Figure 11), for example in the presence of a loop in electronic music (Figure 10).

Let t_1 and t_2 be the time steps at which this is true, and $\tau_1 = f^{-1}[t_1]$ and $\tau_2 = f^{-1}[t_2]$ their antecedents. We then have $\forall m$:

$$\mathbf{Y}_{m\tau_1} = \mathbf{Y}_{m\tau_2} = g[\tau_1]^2 \mathbf{X}_{mt_1} = g[\tau_2]^2 \mathbf{X}_{mt_2} \tag{14}$$

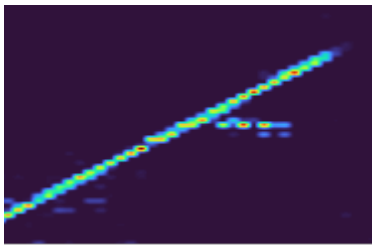

Figure 10: Artifacts in $\mathbf{H}$ caused by spectrally similar frames.

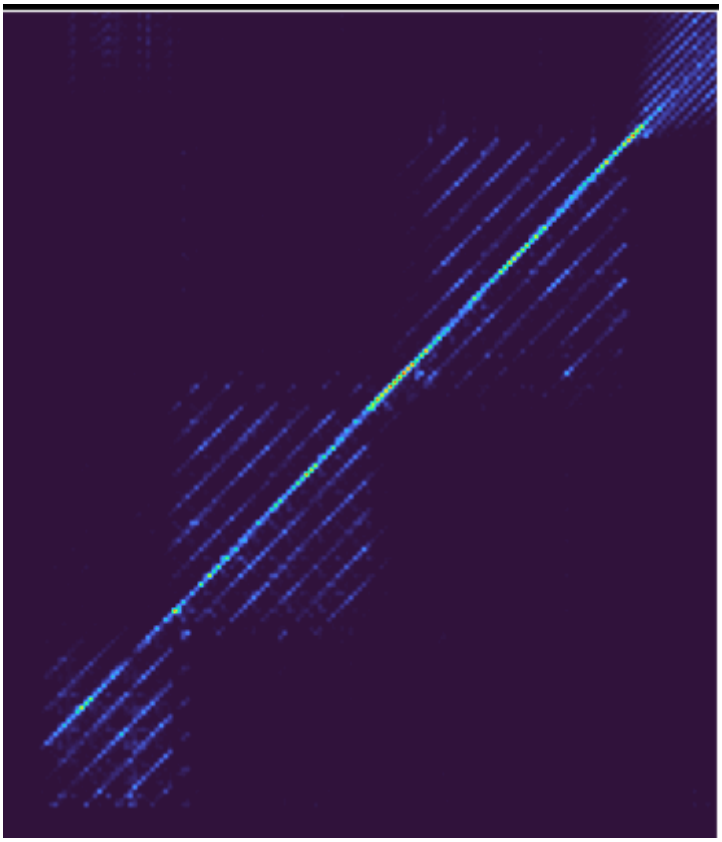

Figure 11: Parallel line structures in $\mathbf{H}$ caused by loops in the reference tracks.

Visually, this corresponds to multiple activations per column of $\mathbf{H}$, with the energy of the activations being distributed arbitrarily between four points: $\{(t_1, \tau_1), (t_1, \tau_2), (t_2, \tau_1), (t_2, \tau_2)\}$. Fortunately, such indeterminacies do not invalidate $\tilde{g}$, but the same can not be said of $\tilde{f}$.

III.4.3.2. Impact of hop size discretization

Usually, the spectrogram is not computed for every sample of a signal as in our earlier definition (Equation 8), but at uniformly sampled time instants spaced by a *hop size* h. This effectively downsamples the time steps to $\bar{t} = ht$ and $\overline{\tau} = h\tau$, leading to the following expressions for the spectrograms:

$$\mathbf{X}_{m\bar{t}} = \left| \sum_{n=0}^{M-1} x[n + ht]w[n]e^{-j2\pi n\frac{m}{M}} \right|^2 \tag{15}$$

$$\mathbf{Y}_{m\bar{\tau}} = \left| g[\tau] \sum_{n=0}^{M-1} x[n + f[h\tau]]w[n]e^{-j2\pi n\frac{m}{M}} \right|^2 \tag{16}$$

Due to this discretization, there may not be an exact alignment between $\bar{t}$ and $\bar{\tau}$. Consequently, the activations within $\mathbf{H}$ could be distributed across neighboring cells.

III.5. The Multi-pass NMF Algorithm

DJ mixes consist of multiple tracks, each typically appearing only within a specific segment of the mix. Consequently, the activation matrix $\mathbf{H}$ is expected to exhibit block-sparsity, as depicted in Figure 12.

While applying NMF directly to spectrograms computed with the desired hop size may yield the expected results, our experiments suggest that an increased number of tracks and smaller window sizes exacerbate cross-track indeterminacies. This issue arises especially because tracks in a mix are often stylistically or tonally similar.

To address this we propose a multi-pass NMF algorithm, described in Figure 13, paired with a filter-threshold-resize procedure inbetween each pass.

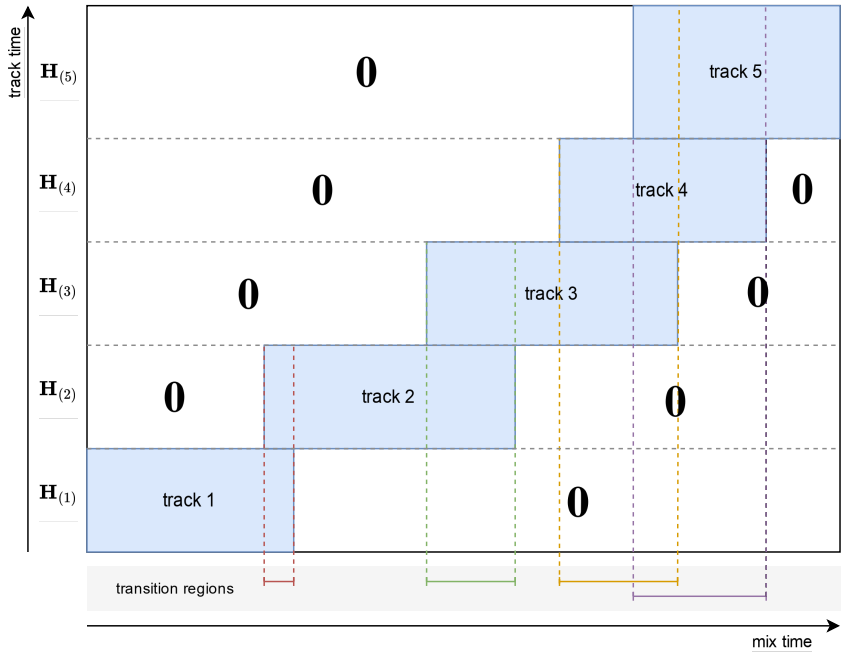

Figure 12: Expected block-sparse form of the activation matrix in a 5-track mix, with transition regions annotated.

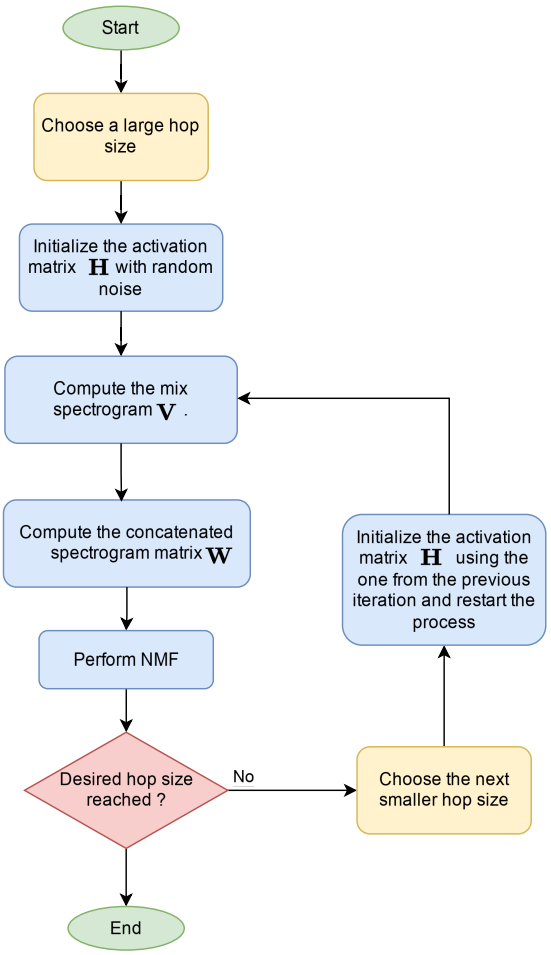

Figure 13: Flow chart of the multipass NMF algorithm

The methodology involves initially performing NMF on spectrograms computed with a significantly large hop size (on the order of minutes) to obtain the approximate position of each track in the mix. The resultant activation matrix is then processed through filtering to reduce noise, followed by blurring and thresholding. This matrix is resized to match the dimensions corresponding to the next smaller hop size, resulting in a larger activation matrix that serves as the initialization for the subsequent NMF pass. This iterative process continues until the desired hop size is reached.

By property of the NMF with multiplicative updates, regions of the matrix set to zero during thresholding will remain zero in subsequent iterations, thereby avoiding spurious activations. However, careful selection of filtering and thresholding techniques is crucial to avoid the inadvertent elimination of valid activations.

Moreover, when coupled with an appropriate block-sparse matrix representation, this approach significantly enhances processing efficiency and memory usage, which is particularly advantageous given the large size of spectrogram matrices at smaller hop sizes.

As an illustration, we ran the multipass NMF algorithm on a 3-track mix with gradually decreasing hop sizes. The Figure 14 depicts the estimated activation matrices at the end of each pass. The first hop size (15 seconds) gives a rough estimation of the positions of the

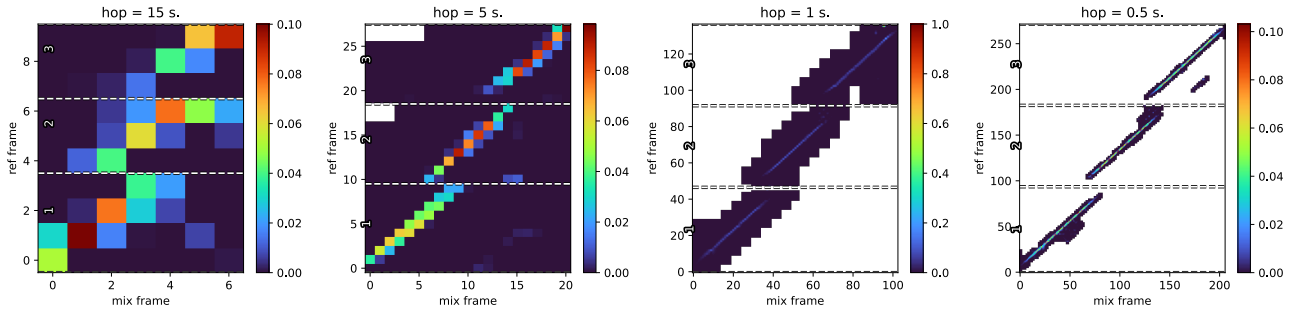

Figure 14: Vizualization of successive estimated activation matrices during execution of the multipass NMF algorithm. Zero-valued cells are depicted in white.

constituent tracks in the mix. and with each subsequent pass, the activation matrices are larger the activations more precise, and become less noisy and more sparse.

III.5.1. Filter-threshold-resize procedure

The filter-threshold-resize procedure is integral to the effectiveness of the multipass NMF algorithm. The steps of the procedure are described below, and illustrated Figure 15.

Morpohological filtering A line-enhancing filter is applied on each submatrix $\mathbf{H}_{(i)}$ of $\mathbf{H}$, inspired by M. Müller and F. Kurth [33]. This filter is designed using fixed-length one-pixel-wide straight line kernels with slopes distributed between a minimum and maximum value. A morphological opening operation is performed on $\mathbf{H}$ with these kernels, and the results are aggregated. This process eliminates activations shorter than the specified length and that do not meet the expected slope limits, effectively denoising the activation matrix.

Blurring A gaussian blur is applied on each submatrix with a small kernel. This has the effect of smearing the activations in time.

Thresholding Set activations below a specified threshold to zero.

Resizing The thresholded activation matrix is then resized to a larger size, i.e. corresponding to a smaller hop size, and returned.

III.6. Downsampling and use of the mel transform

DJ mixes are typically lengthy, ranging from 30 minutes to several hours, and as they consist of musical signals, their frequency bandwidth is notably extensive. When employing the Short-Time Fourier Transform (STFT) with standard hop durations and a typical number of frequency bins for musical signals, the resulting feature matrix can become exceedingly large. This leads to substantial memory usage and elevated resource consumption.

In order to mitigate these issues, we have opted to use relatively large hop durations. This approach not only reduces the computational load but also offers an additional advantage: longer hop durations are better adapted to the temporal structures inherent in music. The hop duration however is not fixed, as is explained in Section III.5.

Additionally, we compress the frequency information using the mel-scale transform [34]. This transform groups nearby frequencies into bins based on a perceptual model of human

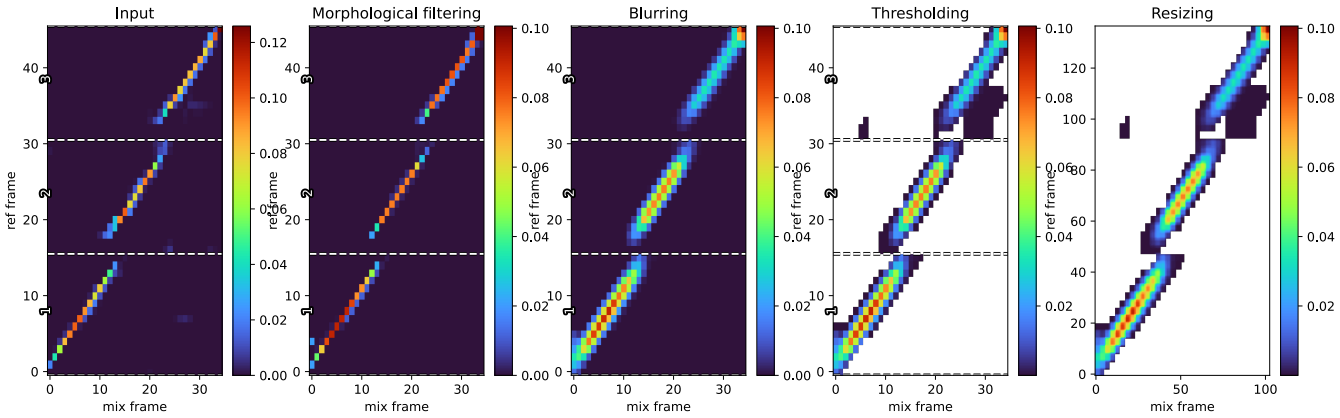

Figure 15: Vizualisation of the steps of the filter-threshold-resize procedure on a 3-track mix. The input activation matrix corresponds to a hop size of 3 seconds, and the output corresponds to a hop size of 1 second. Zero-valued cells are depicted in white.

hearing, which is particularly well-suited for processing musical signals. Importantly, this transform has no effect on the ideal kernel and our estimators.

Proof: Let $\boldsymbol{M}$ be a matrix of mel filterbank coefficients. The mel-spectrograms are calculated from the regular spectrograms: $\mathbf{X}^{\text{mel}} = \boldsymbol{M}\mathbf{X}$ and $\mathbf{Y}^{\text{mel}} = \boldsymbol{M}\mathbf{Y}$. Then we have:

$$\begin{aligned} \mathbf{Y}^{\text{mel}}_{m\tau} &= \sum_i \boldsymbol{M}_{mi}\mathbf{Y}_{i\tau} \\ &= g[\tau]^2 \sum_i \boldsymbol{M}_{mi}\mathbf{X}_{i,f[\tau]} \\ &= g[\tau]^2 \mathbf{X}^{\text{mel}}_{m,f[\tau]} \end{aligned} \tag{17}$$

So the ideal kernel $\mathbf{H}^{\text{ideal}}$ is still clearly a solution of Equation 10. □

III.7. Analysis window overlap

A key parameter when working with spectrograms is the overlap factor of the analysis windows. In order to emphasize the temporal continuity of the musical signals, we use high overlap factors: our experiments have shown that a window size of 6 to 8 times the hop size give the best results. It has experimentally shown to be highly effective in reducing indeterminacies, but tends to smooth out the results as a side-effect, which can potentially obscure finer details in the signal, as shown in Figure 16.

Typically, such large window sizes would result in a substantial increase in the number of frequency bins, leading to higher computational demands. However, by applying the mel-scale transform, we effectively mitigate this issue.

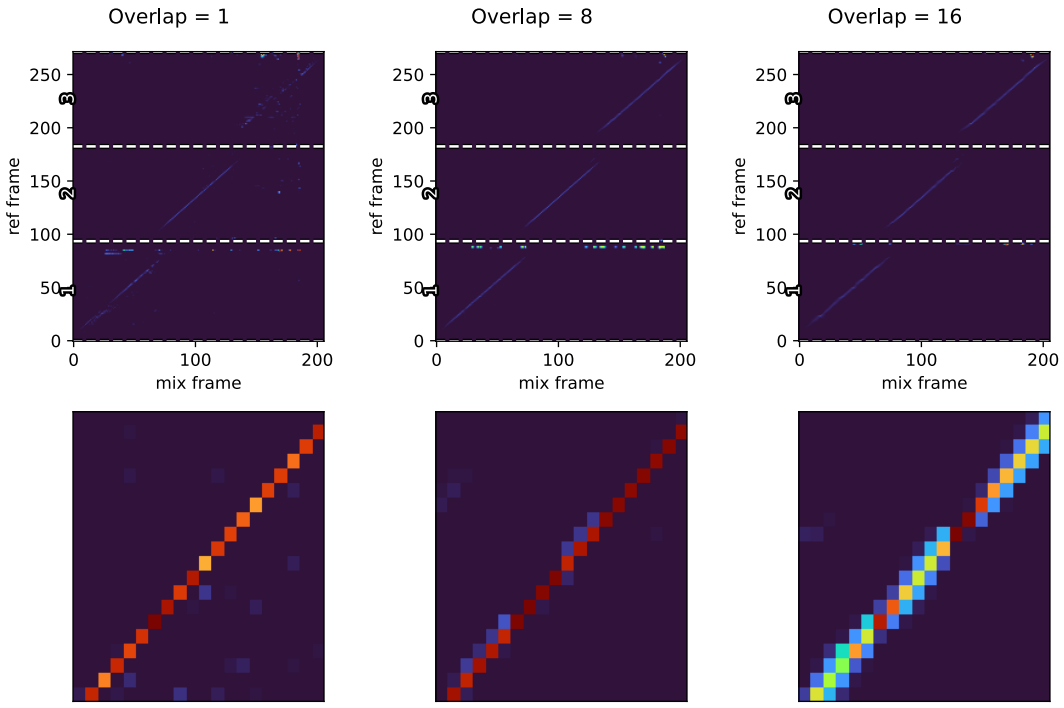

Figure 16: Comparison of the influence of different overlap factors. Top row: estimated activation matrices. Bottom row: zooms on the middle activation.

III.8. Spectrogram normalization

To improve the numeric stability of the NMF, the columns of $\mathbf{X}$ are typically normalized to sum to 1. We also normalize $\mathbf{Y}$ by a single factor[12]. We show that this results in a simple scaling factor for $\mathbf{H}$ and therefore for the estimators.

Proof: Let the scaling factors $\boldsymbol{k}_t \stackrel{\text{def}}{=} \sum_i \mathbf{X}_{it}$ and $\kappa \stackrel{\text{def}}{=} \sum_i \sum_t \mathbf{Y}_{it}$.

The normalized spectrograms are:

$$\mathbf{X}^{\text{norm}}_{mt} \stackrel{\text{def}}{=} \frac{\mathbf{X}_{mt}}{\boldsymbol{k}_t} \tag{18}$$

$$\mathbf{Y}^{\text{norm}} \stackrel{\text{def}}{=} \frac{\mathbf{Y}}{\kappa} \tag{19}$$

Using Equation 9:

[12]Normalizing by column as for $\mathbf{X}$ would cancel out the gain information in $\mathbf{H}$.

$$\mathbf{Y}^{\text{norm}}_{m\tau} = \frac{\boldsymbol{k}_t}{\kappa} g[\tau]^2 \mathbf{X}^{\text{norm}}_{m,f[\tau]} \tag{20}$$

We can then deduce the ideal normalized kernel $\mathbf{H}^{\text{norm}}$ as a solution to Equation 10:

$$\begin{aligned} \mathbf{H}^{\text{norm}}_{t\tau} &\overset{\text{def}}{=} \frac{\boldsymbol{k}_t}{\kappa} g[\tau]^2 \delta_{t,f[\tau]} \\ &= \frac{\boldsymbol{k}_t}{\kappa} \mathbf{H}^{\text{ideal}}_{t\tau} \end{aligned} \tag{21}$$

□

III.9. Thresholding of low-power frames

Recorded music tracks often contain moments of silence or faint noise at the beginning and end of the signal. It is also quite common to find fade-out endings at the end of tracks, or reverberation tails. These elements appear as low-power frames in the feature matrices. Reverberation tails, in particular, are problematic due to their spectral similarity to the rest of the track, which can introduce indeterminacies in the analysis.

Additionally, low-power frames are unlikely to be present in a DJ mix, as DJs typically focus on playing the most musically significant portions of tracks, omitting the very beginning and end.

To address this, we detect and mark low-power frames in the input track spectrograms as unused and set the corresponding cells in the activation matrix to zero, effectively preventing these frames from contributing to the optimization process.

III.10. Implementation

The algorithm has been implemented in Python. By leveraging the pytorch[13] framework, the optimization process can run on both CPU and GPU and benefit from parallel matrix multiplications on the latter.

We summarize the tunable parameters in Table 1 along with their typical values. These can be further adjusted with prior knowledge of the mixes' characteristics.

III.11. Example results

We run the algorithm on several constructed mixes, that showcase mix scenarios with the typical transformations that we aim to be able to reverse engineer. All figures depict:

- On the left, the final estimated activation matrix, with zero-valued cells in gray;
- In the middle, the estimated gain (dots) and ground truth (dashed)
- On the right, the estimated warp function (dots) and ground truth (dashed).

[13] https://pytorch.org

Name	Description	Unit	Typical value
`FS`	Sampling rate	Hz	22050
`HOP_SIZES`	Decreasing list of hop durations	s.	`[20, 10, 2, 0.5, 0.1]`
`OVERLAP`	STFT analysis window overlap factor	-	6 to 8
`NMELS`	Number of mel bands	-	64 to 256
`SPEC_POWER`	STFT power	-	2
`DIVERGENCE`	Divergence function	-	$\mathcal{D}_\beta$ with $\beta = 0$
`LOW_POWER_THRESHOLD`	Threshold under which frames from input tracks are discarded	dB	-40
`CARVE_THRESHOLD`	Threshold under which a cell of the activation matrix is deemed zero	dB	-120
`CARVE_BLUR_SIZE`	Size of the gaussian blur kernel	-	3
`CARVE_MIN_DURATION`	Minimum line duration for morphological filtering	s.	10
`CARVE_MAX_SLOPE`	Maximum allowed deviation from original playing speed	± %	10 to 50
`NOISE_DIM`	Number of columns for noise estimation	-	0 to 50

Table 1: Summary of tunable parameters

Firstly, we focus on time transformations using "mixes" composed of a single track. The Figure 17 presents a single track that has been sliced and looped, and the Figure 18 corresponds to a single track that has been time-stretched by the use of the Rubberband[14] library. Although presenting some artifacts, especially in the gain estimation for the timestretched track, we can see that the time-warping function is estimated accurately.

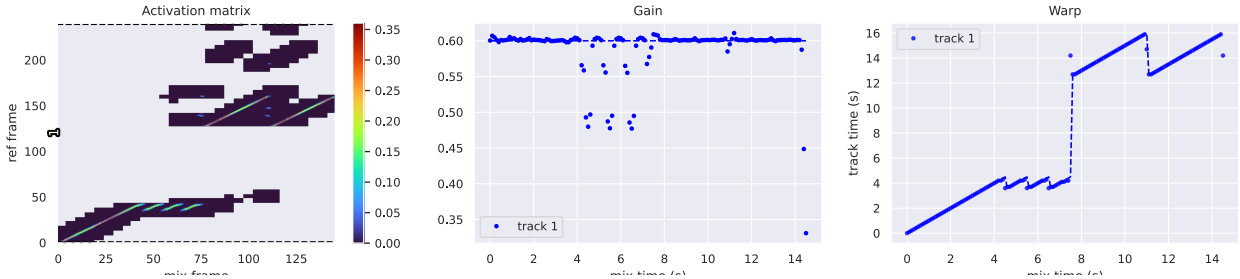

Figure 17: Results for a sliced track with loops and jumps.

[14] https://breakfastquay.com/rubberband/

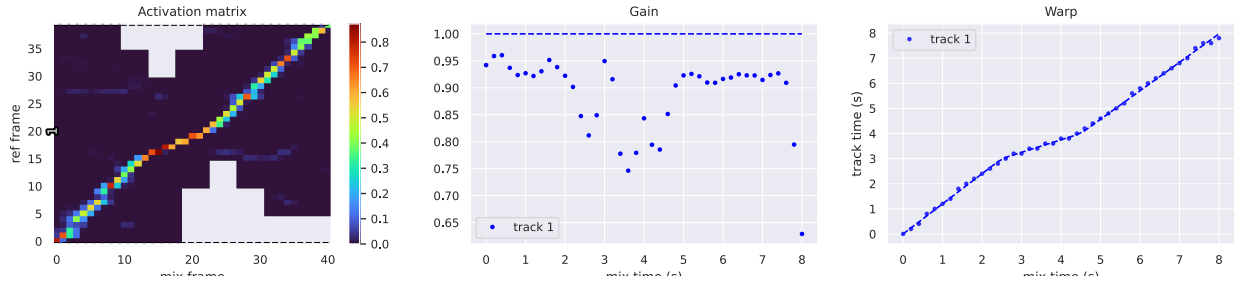

Figure 18: Results for a time-stretched track.

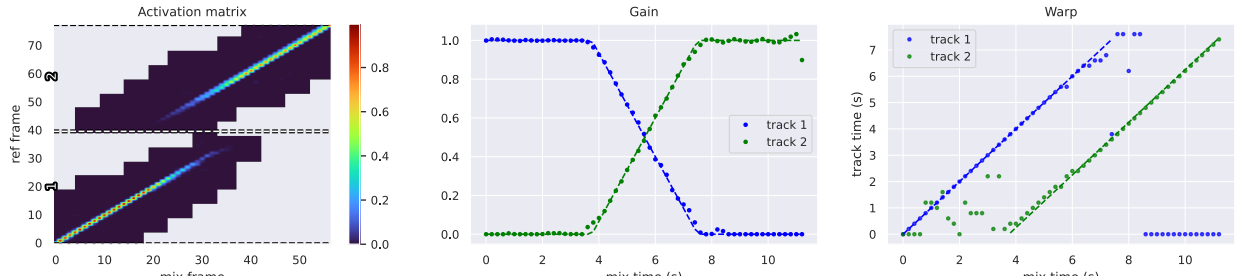

Figure 19: Beat-synchronous mix of two tracks with linear fades.

Secondly, we focus on overlapping tracks. The Figure 19 depicts a simple mix of two tracks with a large transition region composed of a crossfade. Despite having no kwowledge of the form of cross-fading used, the method correctly estimates the gain for both tracks, and mostly correctly estimates the warp function.

It should be noted that the figure depict the raw warp function estimations, which is computed for all tracks for the whole duration of the mix. Hence, the points outside of the tracks' playing range are effectively noise and could be filtered, for example, using a simple threshold on the gain estimation.

III.12. Evaluation on UnmixDB

We applied our algorithm to the UnmixDB dataset version 1.1[15], which includes excerpts from open-licensed dance tracks and their corresponding automatically generated mixes.

Each mix consists of three track excerpts, mixed in a beat-aligned manner with linear crossfades to simulate a realistic DJ context. Tracks are mixed with different combinations of audio effects (none, bass boost, compression, distortion) and time-scaling methods (none, resampling, time-stretching). The dataset provides complete ground truth for all mixes, and includes Python code for generating similar datasets. For further details, see D. Schwarz and D. Fourer [24].

The computation took about two hours on an Intel Xeon E5-2630 CPU with 12 threads. The tunable parameters used are summarized in Table 2.

We define the followig evaluation metrics:

Gain error mean absolute error between the estimated gain and ground truth gain: $\frac{1}{K}\sum_{\tau=1}^{K}|\tilde{g}[\tau]-g[\tau]|$

[15] https://github.com/Ircam-RnD/unmixdb-creation

Name	Value
FS	22050 Hz
HOP_SIZES	[4, 2, 1, 0.5] seconds
OVERLAP	8
NMELS	128
SPEC_POWER	2
DIVERGENCE	$\mathcal{D}_\beta$ with $\beta = 0$
LOW_POWER_THRESHOLD	−40 dB
CARVE_THRESHOLD	−120 dB
CARVE_BLUR_SIZE	3
CARVE_MIN_DURATION	10 seconds
CARVE_MAX_SLOPE	1.5
NOISE_DIM	15

Table 2: Tunable parameters for UnmixDB evaluation.

Warp error mean absolute error in seconds between the estimated and ground truth warp: $\frac{1}{K}\sum_{\tau=1}^{K}\left|\tilde{f}[\tau] - f[\tau]\right|$

The mixes in UnmixDB are generated with fixed time-scale factors. To obtain comparable metrics to D. Schwarz and D. Fourer [8], we estimate the speed factor and the cue point by linear regression over the warp sequence, and define the following metrics:

Speed ratio mean ratio between the estimated and the ground truth speed factors.

Cue point error mean absolute error in seconds between the estimated and the ground truth cue points.

Box plots of these metrics are represented respectively in Figure 20, Figure 21, Figure 22 and Figure 23.

These results demonstrate the validity of our method, particularly regarding the variants with time-stretching and without. The performance on resampled mixes, which feature transposition, is poorer. However, given that our mixing model doesn't include pitch-shifting in its assumptions, we find it is still acceptable performance.

The performance of cue point estimation is comparable to D. Schwarz and D. Fourer [8], but the same cannot be said of the speed ratio estimation. We explain this by the more lax assumptions of our approach regarding time-warping.

III.12.1. Impact of noise estimation

We conduct an additional experiment on UnmixDB to evaluate the effectiveness of the additive noise estimation. We compare in Figure 24 and Figure 25 two runs of the algorithm without noise (in blue) and with noise estimation (in orange). The results show that in the case of added audio effects, adding noise estimation to the optimization algorithm improves the estimation. The "dist" and especially the "bass" variants benefit the most.

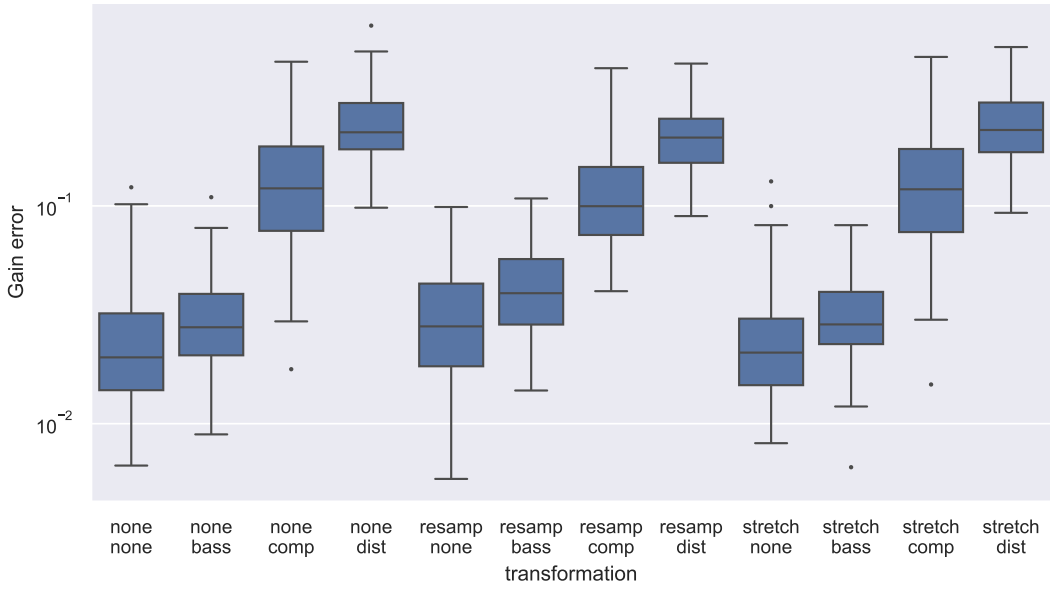

Figure 20: Box plot of the gain error per variant.

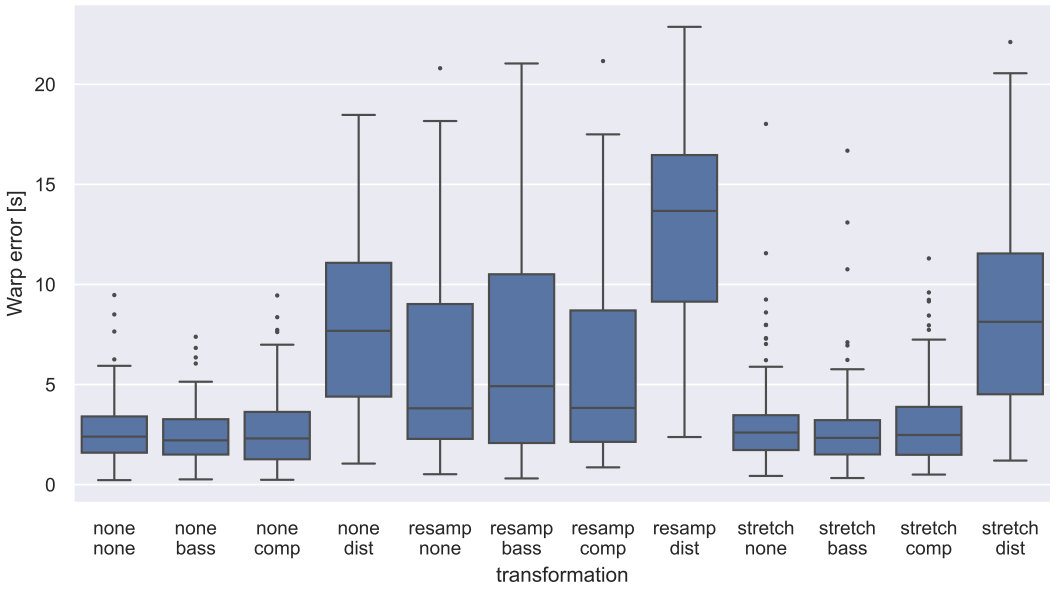

Figure 21: Box plot of the warp error per variant.

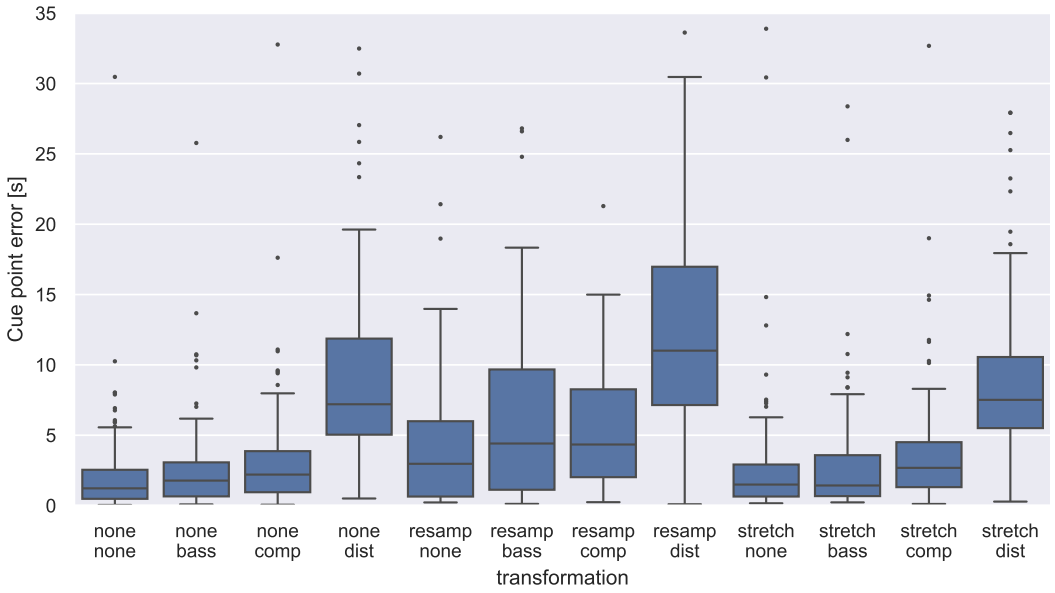

Figure 22: Box plot of the cue point error per variant.

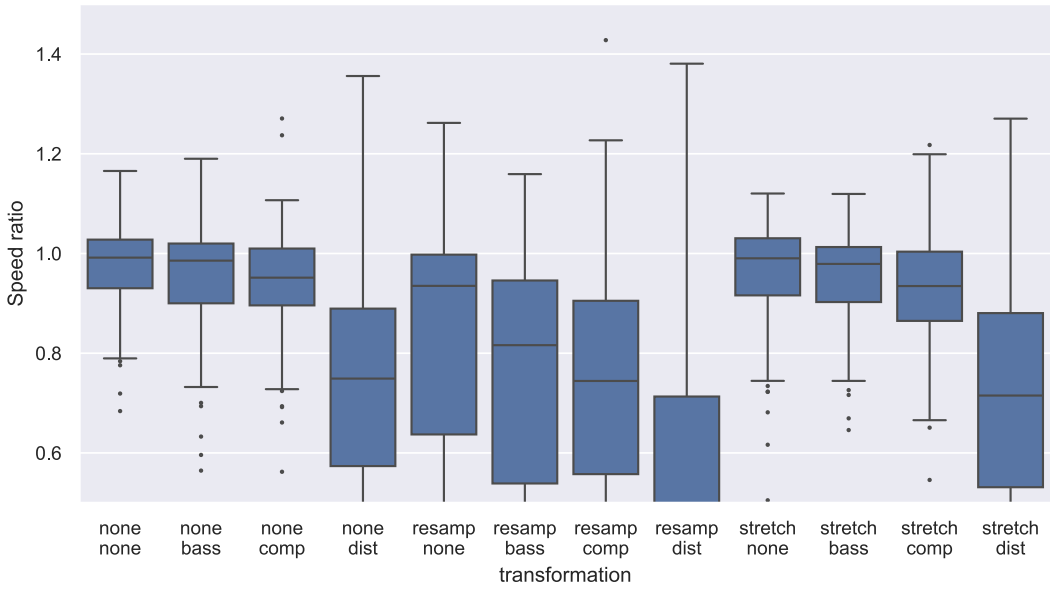

Figure 23: Box plot of the speed ratio per variant.

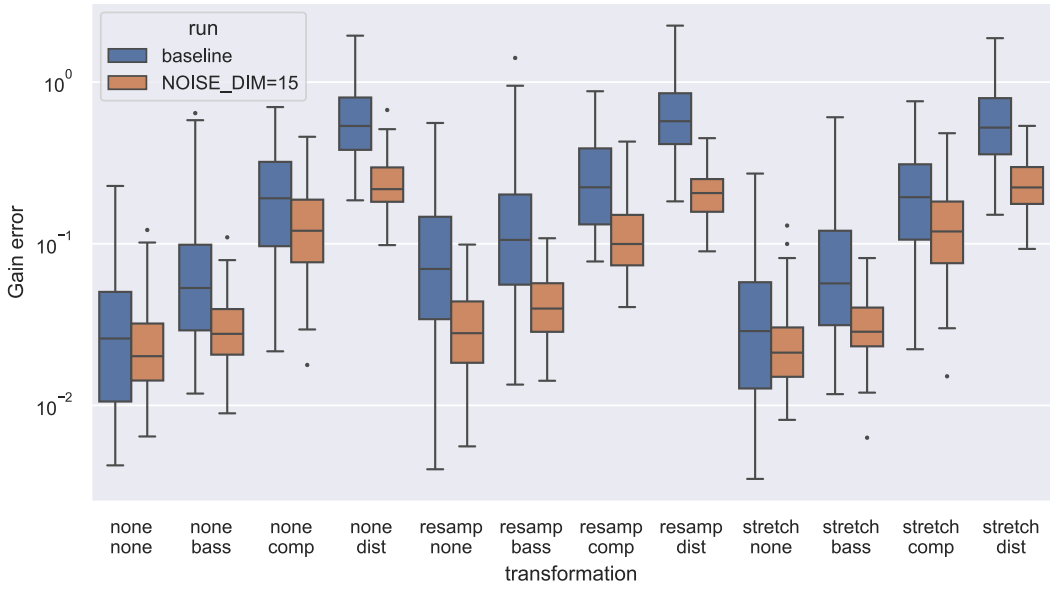

Figure 24: Box plot of the gain error per variant

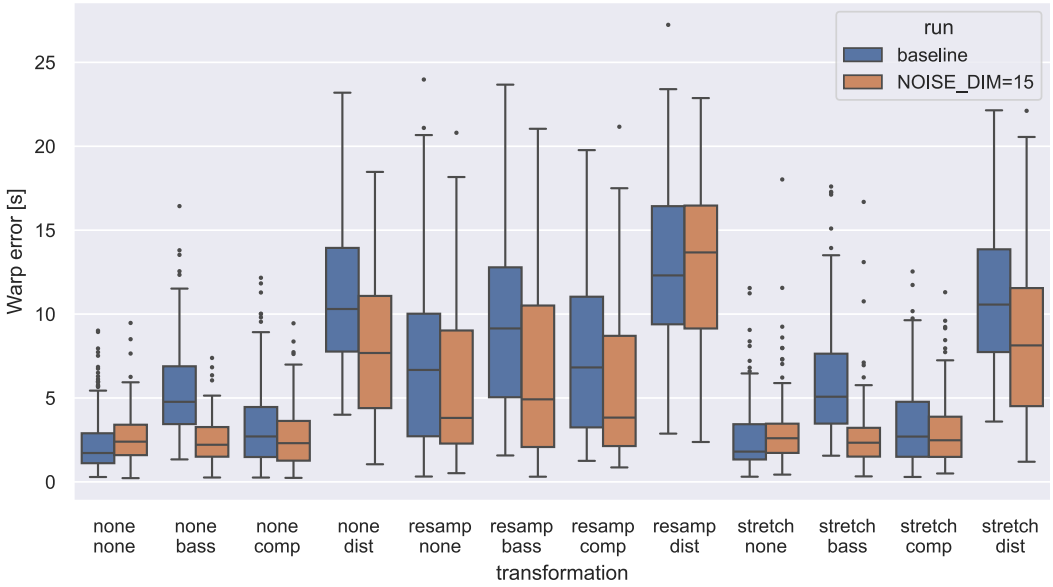

Figure 25: Box plot of the warp error per variant

IV. Conclusion

This internship report has discussed existing methods and datasets for DJ mix reverse engineering. The need for datasets with precise and complete ground truth annotations has been highlighted, emphasizing their importance for advancing research in this area, and explored potential methodologies for their creation.

In response to the challenges presented by DJ mixes with complex time-warping transformations, and to address limitations of previous work in this regard, a new integrated approach was proposed. This approach involves the use of Non-negative Matrix Factorization (NMF) with a multi-pass extension, supported by a mixing model grounded in the technical principles of DJ hardware. We demonstrated the effectiveness of the method on arbitrary time-warping transformations. While the results obtained in quantitative evaluation did not match the precision of previous methods, the proposed approach demonstrated potential in capturing a broader spectrum of DJ practices, offering a foundation for further refinement and exploration in future research.

Bibliography

[1] J. Six, "Olaf : A Lightweight, Portable Audio Search System," *JOURNAL OF OPEN SOURCE SOFTWARE*, vol. 8, no. 87, 2023, doi: 10.21105/joss.05459.

[2] R. Sonnleitner, A. Arzt, and G. Widmer, "Landmark-Based Audio Fingerprinting for DJ Mix Monitoring," *ISMIR*, 2016.

[3] A. L.-C. Wang, "An Industrial-Strength Audio Search Algorithm."

[4] S. Ewert, M. Muller, and P. Grosche, "High Resolution Audio Synchronization Using Chroma Onset Features," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, Taipei, Taiwan: IEEE, Apr. 2009, pp. 1869–1872. doi: 10.1109/ICASSP.2009.4959972.

[5] M. Ramona and G. Peeters, "Automatic Alignment of Audio Occurrences: Application to the Verification and Synchronization of Audio Fingerprinting Annotation," pp. 429–436, Jan. 2011.

[6] L. Werthen-Brabants, T. De Bie, and P. Simeone, "Ground Truth Extraction & Transition Analysis of DJ Mixes," 2018.

[7] T. Kim, M. Choi, E. Sacks, Y.-H. Yang, and J. Nam, "A Computational Analysis of Real-World DJ Mixes Using Mix-to-Track Subsequence Alignment," *ISMIR*, 2020.

[8] D. Schwarz and D. Fourer, "Methods and Datasets for DJ-Mix Reverse Engineering," *Perception, Representations, Image, Sound, Music*, vol. 12631. Springer International Publishing, Cham, pp. 31–47, 2021. doi: 10.1007/978-3-030-70210-6_2.

[9] D. Yang, K. Ji, and T. Tsai, "Aligning Unsynchronized Part Recordings to a Full Mix Using Iterative Subtractive Alignment," *ISMIR*, 2021.

[10] J. Six, "DiscStitch : Towards Audio-to-Audio Alignment with Robustness to Playback Speed Variabilities," in *Extended Abstracts for the Late-Breaking Demo Session of the 23rd International Society for Music Information Retrieval Conference*, ISMIR, 2022.

[11] M. Ramona and G. Richard, "A Simple and Efficient Fader Estimator for Broadcast Radio Unmixing," *Proc. Digital Audio Effects (DAFx)*, pp. 265–268, Sep. 2011.

[12] D. Schwarz and D. Fourer, "Towards Extraction of Ground Truth Data from DJ Mixes," *ISMIR*, p. 2017–2018.

[13] T. Kim, Y.-H. Yang, and J. Nam, "Joint Estimation of Fader and Equalizer Gains of DJ Mixers Using Convex Optimization," *DAFx*, 2022.

[14] D. Barchiesi and J. Reiss, "Reverse Engineering of a Mix," *J. Audio Eng. Soc.*, vol. 58, no. 7, 2010.

[15] M. Zehren, M. Alunno, and P. Bientinesi, "Automatic Detection of Cue Points for DJ Mixing," no. arXiv:2007.08411. arXiv, Jul. 2020.

[16] D. Schwarz, D. A. Schindler, and S. Spadavecchia, "A Heuristic Algorithm for DJ Cue Point Estimation," 2018.

[17] K. Yadati, M. Larson, C. C. S. Liem, and A. Hanjalic, "Detecting Drops in Electronic Dance Music: Content Based Approaches to a Socially Significant Music Event," *ISMIR*, 2014.

[18] A. Kim, S. Park, J. Park, J.-W. Ha, T. Kwon, and J. Nam, "Automatic Dj Mix Generation Using Highlight Detection," *Proc. ISMIR, late-breaking demo paper*, 2017.

[19] R. M. Bittner *et al.*, "Automatic Playlist Sequencing and Transitions," *ISMIR*, 2017.

[20] D. Cliff, "Hang the DJ: Automatic Sequencing and Seamless Mixing of Dance-Music Tracks," *Hp Laboratories Technical Report Hpl*, vol. 104, Aug. 2000.

[21] T. Fujio and H. Shiizuka, "A System of Mixing Songs for Automatic DJ Performance Using Genetic Programming," in *6th Asian Design International Conference*, 2003.

[22] B.-Y. Chen, W.-H. Hsu, W.-H. Liao, M. A. M. Ramírez, Y. Mitsufuji, and Y.-H. Yang, "Automatic DJ Transitions with Differentiable Audio Effects and Generative Adversarial Networks," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2022, pp. 466–470. doi: 10.1109/ICASSP43922.2022.9746663.

[23] X. Li, "LooPy: A Research-Friendly Mix Framework for Music Information Retrieval on Electronic Dance Music," no. arXiv:2305.01051. arXiv, May 2023.

[24] D. Schwarz and D. Fourer, "UnmixDB: A Dataset for DJ-Mix Information Retrieval," *ISMIR*, Sep. 2018.

[25] T. Scarfe, W. M. Koolen, and Y. Kalnishkan, "A Long-Range Self-similarity Approach to Segmenting DJ Mixed Music Streams," *Artificial Intelligence Applications and Innovations*, vol. 412. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 235–244, 2013. doi: 10.1007/978-3-642-41142-7_24.

[26] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004, doi: 10.1023/B:VISI.0000029664.99615.94.

[27] O. Faugeras and F. Lustman, "MOTION AND STRUCTURE FROM MOTION IN A PIECEWISE PLANAR ENVIRONMENT," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 2, no. 3, pp. 485–508, Sep. 1988, doi: 10.1142/S0218001488000285.

[28] C. Févotte, N. Bertin, and J.-L. Durrieu, "Nonnegative Matrix Factorization with the Itakura-Saito Divergence: With Application to Music Analysis," *Neural Computation*, vol. 21, no. 3, pp. 793–830, Mar. 2009, doi: 10.1162/neco.2008.04-08-771.

[29] C. Févotte and J. Idier, "Algorithms for Nonnegative Matrix Factorization with the Beta-Divergence," *Neural Computation*, vol. 23, no. 9, pp. 2421–2456, 2011.

[30] T. Virtanen, "Monaural Sound Source Separation by Nonnegative Matrix Factorization With Temporal Continuity and Sparseness Criteria," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 3, pp. 1066–1074, Mar. 2007, doi: 10.1109/TASL.2006.885253.

[31] C. Fevotte, "Majorization-Minimization Algorithm for Smooth Itakura-Saito Nonnegative Matrix Factorization," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic: IEEE, May 2011, pp. 1980–1983. doi: 10.1109/ICASSP.2011.5946898.

[32] D. FitzGerald, M. Cranitch, and E. Coyle, "On the Use of the Beta Divergence for Musical Source Separation," in *IET Irish Signals and Systems Conference (ISSC 2009)*, Dublin, Ireland: IET, 2009, p. 34–35. doi: 10.1049/cp.2009.1711.

[33] M. Müller and F. Kurth, "Enhancing Similarity Matrices for Music Audio Analysis," Jun. 2006, p. V–V. doi: 10.1109/ICASSP.2006.1661199.

[34] S. S. Stevens, J. Volkmann, and E. B. Newman, "A Scale for the Measurement of the Psychological Magnitude Pitch," *The Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 185–190, Jan. 1937, doi: 10.1121/1.1915893.

A. The TrackId.net dataset

Trackid.net[16] is an automated track identification service, presented as a freely accessible website, which features a collection of mixes along with their associated playlists. Users request track identification by submitting a link to a mix from streaming services, and the website uses a fingerprinting method to identify the tracks played. Registered users can also amend the tracklist to correct identification errors or manually add tracks.

During our internship, we contacted the website's owners, who kindly provided a snapshot of their database. As of April 10, 2024, this snapshot contained metadata for 136,231 mixes for a cumulative duration of 7,896 days of audio, and 666,625 unique tracks.

A.1. Dataset contents

For each mix, the available metadata includes:
- The title, duration, and source of the mix;
- The tracklist with titles, artists, and labels (when available);
- The start and end times for each track;
- The inferred style of the mix.

Notably, the database does not include any audio data. While the mixes' audio can be retrieved via the provided streaming service links (assuming they are still available), obtaining the associated tracks' audio is more challenging. Only fuzzy matching of the tracks' title, artist, and label is possible, making it difficult to ensure the retrieved tracks correspond exactly to those referenced in the database.

Additionally, the vast majority of the audio content is protected under copyright law, which presents challenges for its use in scientific research.

A.2. Analysis

A.2.1. General observations

Given the substantial amount of mixes included, the dataset offers insights about the current landscape of recorded DJ mixes.

As shown in Figure 26, most mixes range from 30 minutes to 2 hours in length, with an average duration of 83 minutes. Some mixes, however, are considerably longer. It is also noteworthy that the duration of many mixes is a multiple of 30 minutes.

The distribution of the number of tracks per mix follows a similar pattern (Figure 27), with an average of 12.5 tracks per mix.

The distribution of *play duration* of the tacks in mixes (Figure 28), which is defined as the cumulated time spans between the start and end of identified tracks, is also expected, with an average of 3 minutes and 30 seconds.

[16] https://trackid.net

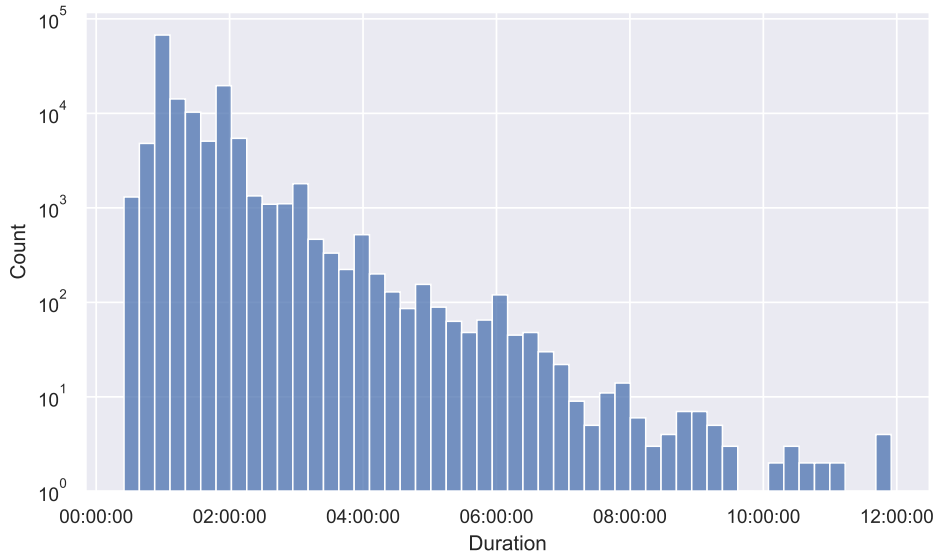

Figure 26: Distribution of the mixes' duration.

A.2.2. Styles

According to discussions with the owners of Trackid.net, the style(s) of a mix are inferred from the styles of its constituent tracks. The primary source for track styles is the Discogs API[17].

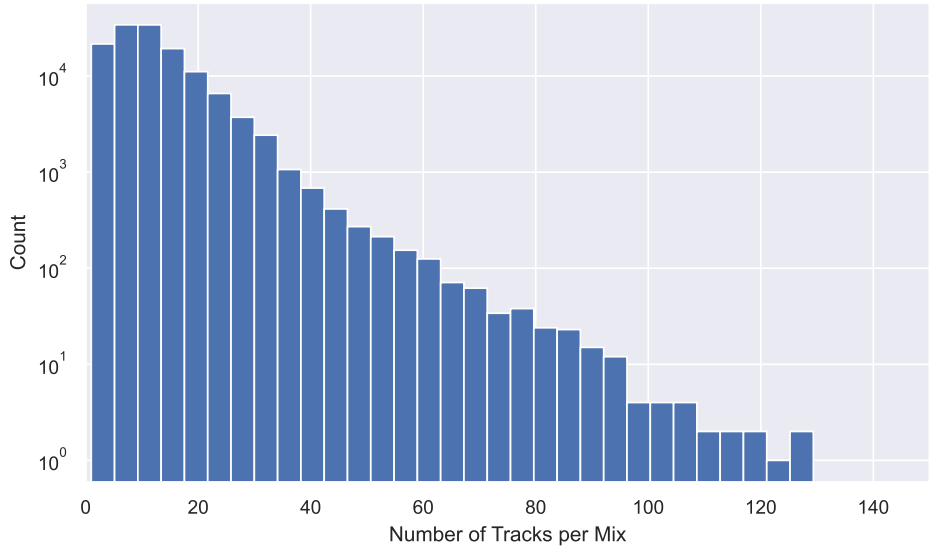

Figure 27: Distribution of the number of unique tracks per mix.

[17] https://www.discogs.com/developers

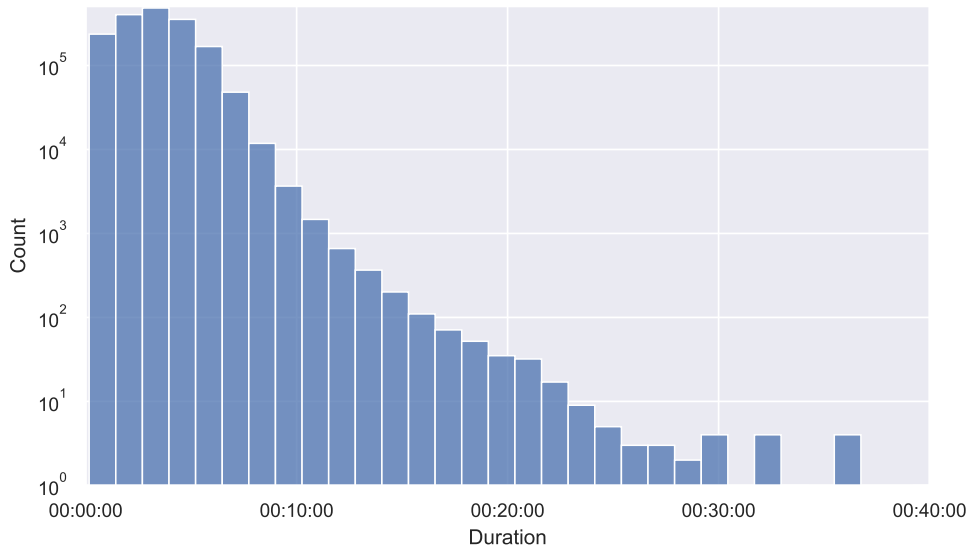

Figure 28: Distribution of the play duration of tracks in mixes.

The most frequent styles, as depicted in Figure 29, are predominantly electronic, with House and Techno leading. Interestingly, “Experimental” and “Ambient” are also popular, which may be due to the liberal application of these categories on Discogs.

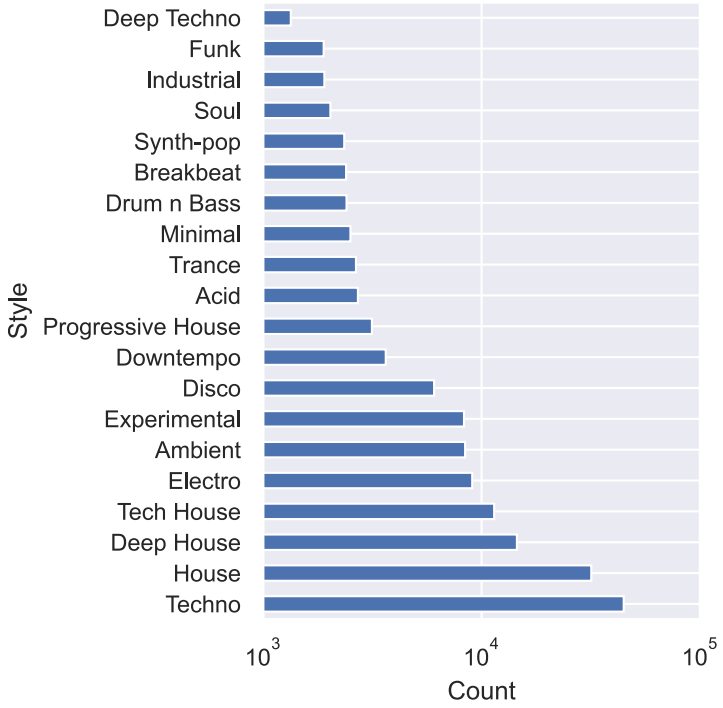

Figure 29: Most prevalent styles. Mixes can have more than one style.

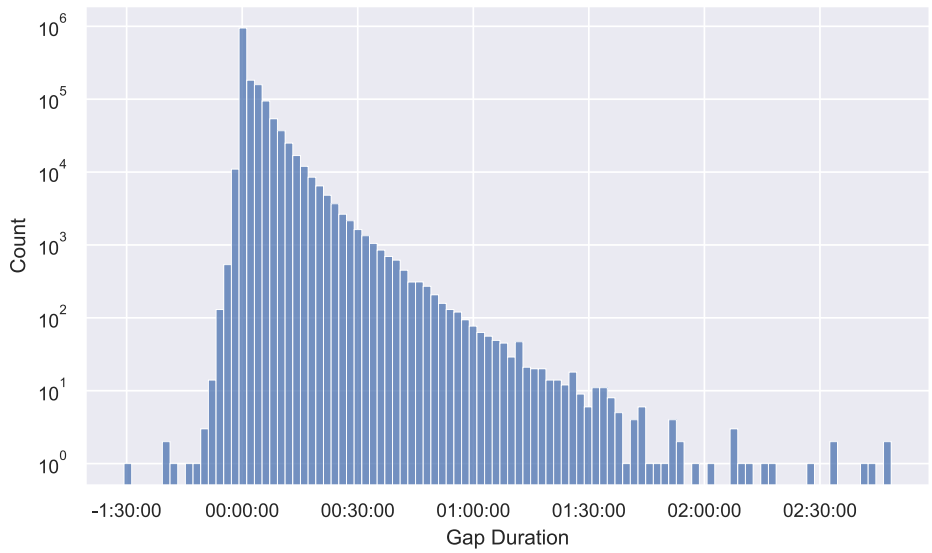

Figure 30: Distribution of the durations of gaps between tracks.

A.2.3. Quality of the annotations

To assess the quality of the metadata, we calculated the *gap duration* between tracks, defined as the difference between the detected start of track n and the detected end of track $n-1$. A positive difference indicates a gap between tracks, while a negative difference indicates overlapping tracks.

The distribution of gap durations (Figure 30) suggests a significant number of incompletely annotated mixes. However, it is important to note that gaps may not always signify poor annotations. For example, radio mixes may correctly present gaps for segments where people are talking between tracks.

This observation is further supported by examining the proportion of identified time per mix (Figure 31), defined as the ratio between the cumulative identified track time and the total duration of the mix. Very few mixes are completely or nearly completely annotated, with the average identification proportion being around 50%.

A.3. Conclusion

The trackid.net dataset stands out due to:
- A variety of styles represented;
- A large number of entries, making it suitable for big data and machine learning applications;
- High ecological validity.

However, there are some challenges that make it less suitable for research in its current form:
- Numerous false positives and negatives from the fingerprinting process;
- Imprecise timestamps and gaps in the data;

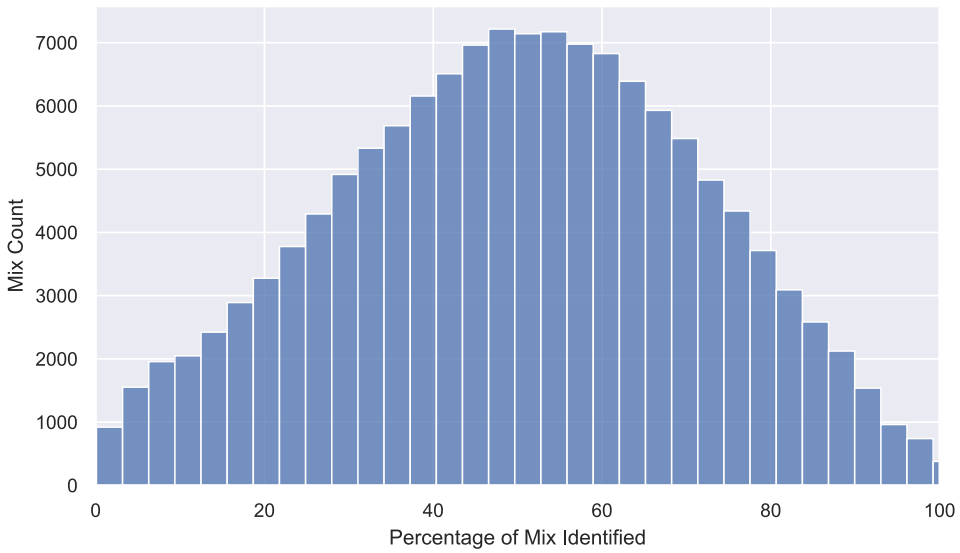

Figure 31: Distribution of the proportion of identified time per mix.

- Over- or under-representation of certain styles;
- Uncertain availability of audio.

We believe that all of these challenges can be curbed by curating a smaller, higher quality dataset from this one, by exploiting the statistical indicators used in the previous section. In particular, mixes should be filtered to favorize:

- A more uniform and diverse representation of musical styles;
- High identification proportion;
- Small gap durations.

B. Penalized NMF for DJ mix transcription

As stated in Section III.3.1, the objective function optimized by the NMF algorithm can be modified to include penalizations. To illustrate, we will consider a new objective function $\mathcal{C}$ involving the beta-divergence and an additional penalty function $\mathcal{P}$ on $\mathbf{H}$ weighted by $\lambda \in \mathbb{R}^+$:

$$\mathcal{C} = \mathcal{D}_\beta(\mathbf{V} \mid \mathbf{WH}) + \lambda \mathcal{P}(\mathbf{H}) \tag{22}$$

Assuming the gradients of $\mathcal{C}$ w.r.t. $\mathbf{H}$ and $\mathbf{W}$ are separable into their positive and negative parts, new multiplicative update rules can be derived by following the procedure from Section III.3.1. However, the monotonicity of the gradient descent is no longer guaranteed, and is dependent on the form of $\mathcal{P}$ and the choice of λ; which both can be validated through experimentation.

We present the following penalty functions to favorize certain characteristics of the activation matrix:

L1 regularization encourages sparsity of the activation matrix.
Gain smoothness discourages abrupt changes in gain.
Diagonal smoothness favorizes diagonal activations.
Lineness discourages multiple horizontal, vertical or diagonal activations.

However, during our experimentation, we observed that these modifications resulted in only marginal improvements in transcription quality. Nevertheless, for completeness and reference, the rationale behind each penalty function, along with their gradient derivations, is provided in the following sections.

B.1. L1 (Lasso) regularization

To promote sparsity in the activation matrix, the L1 regularization penalty can be added to the objective function. This penalty function favors solutions with many zero entries in the activation matrix, effectively encouraging a sparse representation of the data.

The penalty function for L1 regularization is defined as:

$$\mathcal{P}_{\text{lasso}}(\mathbf{H}) = \sum_{i=0}^{T-1} \sum_{j=0}^{K-1} |\mathbf{H}_{ij}| \tag{23}$$

Given that $\mathbf{H}$ is positive:

$$\mathcal{P}_{\text{lasso}}(\mathbf{H}) = \sum_{i=0}^{T-1} \sum_{j=0}^{K-1} \mathbf{H}_{ij} \tag{24}$$

Gradient calculation: The partial derivative of the penalty function with respect to $\mathbf{H}_{ij}$ is given by:

$$\frac{\partial \mathcal{P}_{\text{lasso}}}{\partial \mathbf{H}_{ij}} = 1 \tag{25}$$

Thus:

$$\begin{aligned} \nabla^{+}_{\mathbf{H}} \mathcal{P}_{\text{lasso}} &= 1 \\ \nabla^{-}_{\mathbf{H}} \mathcal{P}_{\text{lasso}} &= 0 \end{aligned} \tag{26}$$

B.2. Gain smoothness penalty

The DJ is expected to perform smooth transitions, so we assume that the gain applied to each track is varying relatively slowly. Consequently, we want to penalize abrupt changes in the gain. Using the gain estimator (Equation 12), we have:

$$\begin{aligned} g[\tau]^2 - g[\tau-1]^2 &= \sum_{t=0}^{T-1} \mathbf{H}_{t\tau} - \sum_{t=0}^{T-1} \mathbf{H}_{t,\tau-1} \\ &= \sum_{t=0}^{T-1} \big(\mathbf{H}_{t\tau} - \mathbf{H}_{t,\tau-1}\big) \end{aligned} \tag{27}$$

The gradient of this expression is not separable as-is, so we square it and define the following penalty function:

$$\mathcal{P}_g(\mathbf{H}) = \sum_{\tau=1}^{K-1} \sum_{t=0}^{T-1} \big(\mathbf{H}_{t\tau} - \mathbf{H}_{t,\tau-1}\big)^2 \tag{28}$$

Gradient calculation

$$\frac{\partial}{\partial \mathbf{H}_{ij}} \big(\mathbf{H}_{t\tau} - \mathbf{H}_{t,\tau-1}\big)^2 = \begin{cases} 2\big(\mathbf{H}_{ij} - \mathbf{H}_{i,j-1}\big) & \text{if } i = t \text{ and } j = \tau \\ -2\big(\mathbf{H}_{i,j+1} - \mathbf{H}_{ij}\big) & \text{if } i = t \text{ and } j + 1 = \tau \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

So:

$$\begin{aligned}\frac{\partial \mathcal{P}_g}{\partial \mathbf{H}_{ij}} &= 2\big(\mathbf{H}_{ij} - \mathbf{H}_{i,j-1}\big) - 2\big(\mathbf{H}_{i,j+1} - \mathbf{H}_{ij}\big) \\ &= 4\mathbf{H}_{ij} - 2\big(\mathbf{H}_{i,j-1} + \mathbf{H}_{i,j+1}\big)\end{aligned} \tag{30}$$

Thus:

$$\begin{aligned}\nabla^{+}_{\mathbf{H}} \mathcal{P}_g &= 4\mathbf{H} \\ \big(\nabla^{-}_{\mathbf{H}} \mathcal{P}_g\big)_{ij} &= 2\big(\mathbf{H}_{i,j-1} + \mathbf{H}_{i,j+1}\big)\end{aligned} \tag{31}$$

B.3. Diagonal smoothness penalty

We hypothesize the tracks to be played near their original speed, and that there will be significant time intervals without any loops or jumps. This results in diagonal line structures in $\mathbf{H}$. We define a **diagonal smoothness** penalty that minimises the difference between diagonal cells of $\mathbf{H}$:

$$\mathcal{P}_d(\mathbf{H}) = \sum_{t=1}^{T-1} \sum_{\tau=1}^{K-1} \big(\mathbf{H}_{t,\tau} - \mathbf{H}_{t-1,\tau-1}\big)^2 \tag{32}$$

Gradient calculation: Similarly to the the gain smoothness penalty:

$$\frac{\partial}{\partial \mathbf{H}_{ij}} \big(\mathbf{H}_{t\tau} - \mathbf{H}_{t-1,\tau-1}\big)^2 = \begin{cases} 2\big(\mathbf{H}_{ij} - \mathbf{H}_{i-1,j-1}\big) & \text{if } i = t \text{ and } j = \tau \\ -2\big(\mathbf{H}_{i+1,j+1} - \mathbf{H}_{ij}\big) & \text{if } i+1 = t \text{ and } j+1 = \tau \\ 0 & \text{otherwise} \end{cases} \tag{33}$$

So:

$$\begin{aligned}\frac{\partial \mathcal{P}_d}{\partial \mathbf{H}_{ij}} &= 2\big(\mathbf{H}_{ij} - \mathbf{H}_{i-1,j-1}\big) - 2\big(\mathbf{H}_{i+1,j+1} - \mathbf{H}_{ij}\big) \\ &= 4\mathbf{H}_{ij} - 2\big(\mathbf{H}_{i-1,j-1} + \mathbf{H}_{i+1,j+1}\big)\end{aligned} \tag{34}$$

Thus:

$$\begin{aligned}\nabla^{+}_{\mathbf{H}} \mathcal{P}_d &= 4\mathbf{H} \\ \big(\nabla^{-}_{\mathbf{H}} \mathcal{P}_d\big)_{ij} &= 2\big(\mathbf{H}_{i-1,j-1} + \mathbf{H}_{i+1,j+1}\big)\end{aligned} \tag{35}$$

B.4. Lineness penalty

The time-warping function $f[\tau]$ is expected to be injective and piecewise continuous. This means we can characterize the neighboring cells of a given activation in $\mathbf{H}$. Given an activated cell (i, j), only the up direction $(i+1, j)$, right direction $(i, j+1)$, or upper-right diagonal direction $(i+1, j+1)$ should be activated, but not any combination of the three.

Thus we define the following, that increases when more than one of these directions are activated near an activated cell:

$$\mathcal{P}_l(\mathbf{H}) = \sum_{t=0}^{T-2} \sum_{\tau=0}^{K-2} \mathbf{H}_{t,\tau} \big(\mathbf{H}_{t,\tau+1} \mathbf{H}_{t+1,\tau+1} + \mathbf{H}_{t+1,\tau} \mathbf{H}_{t+1,\tau+1} + \mathbf{H}_{t+1,\tau} \mathbf{H}_{t,\tau+1} \big) \tag{36}$$

Gradient calculation:

$$\begin{aligned} \frac{\partial \mathcal{P}_l}{\partial \mathbf{H}_{ij}} = \nabla^+_{\mathbf{H}} \mathcal{P}_l = \mathbf{H}_{i,j+1} \mathbf{H}_{i+1,j+1} + \mathbf{H}_{i+1,j} \mathbf{H}_{i+1,j+1} + \mathbf{H}_{i+1,j} \mathbf{H}_{i,j+1} \\ + \mathbf{H}_{i-1,j} \mathbf{H}_{i,j+1} + \mathbf{H}_{i-1,j} \mathbf{H}_{i-1,j+1} \\ + \mathbf{H}_{i,j-1} \mathbf{H}_{i+1,j} + \mathbf{H}_{i,j-1} \mathbf{H}_{i+1,j-1} \\ + \mathbf{H}_{i-1,j-1} \mathbf{H}_{i-1,j} + \mathbf{H}_{i-1,j-1} \mathbf{H}_{i,j-1} \\ \nabla^-_{\mathbf{H}} \mathcal{P}_l = 0 \end{aligned} \tag{37}$$