TelecomBCN-UPC

Degree's Thesis

Automatic Drums Transcription for polyphonic music using Non-Negative Matrix Factor Deconvolution

Author:
Jordi Pons i Puig

Supervisors:
A. Bonafonte and A. Roebel

A thesis submitted in fulfilment of the requirements for the degree of Audiovisual Systems Engineering

developed at

IRCAM-Analysis/Synthesis Team

Paris and Barcelona, July 2014

Abstract

Automatic Drums Transcription for polyphonic music using Non-Negative Matrix Factor Deconvolution

by Jordi Pons i Puig

ENG: This thesis presents an automatic procedure for the detection and classification of percussive sounds in polyphonic audio mixes. The proposed method uses an extension of Non-negative Matrix Factorization (NMF) [1] which is capable to identify patterns with a temporal structure: Non-negative Matrix Factor Deconvolution (NMD) [2]. A complete drum transcription aims to be achieved with the time localization of the onsets and the identification of the percussive sounds. This work is focused on the percussion instruments found in the standard rock/pop drum kit: snare drum, bass drum, tom-toms, hi-hats and cymbals. This framework can be trained for identifying other percussive instruments or impulsive sounds.

CAT: Aquest treball presenta un procediment automàtic per a la detecció i la classificació de sons percussius en mescles d'àudio polifòniques. El mètode proposat utilitza una extensió de la tècnica Non-negative Matrix Factorization (NMF) [1] que és capaç d'identificar patrons amb una estructura temporal: Non-negative Matrix Factor Deconvolution (NMD) [2]. L'algorisme pretén dur a terme una transcripció completa identificant el moment en què toca un instrument determinat de la bateria. Aquest treball està centrat en el kit de bateria habitual en pop/rock: caixa, bombo, timbales, hi-hat i plats. Aquest entorn pot ser entrenat per a reconèixer altres instruments percussius o sons impulsionals.

CAST: Esta tesis presenta un procedimiento automático para la detección y la clasificación de sonidos percusivos en mezclas de audio polifónicas. El método propuesto utiliza una extensión de la técnica Non-negative Matrix Factorization (NMF) [1] que es capaz de identificar patrones con una estructura temporal: Non-negative Matrix Factor Deconvolution (NMD) [2]. El algoritmo pretende ser capaz de hacer una transcripción completa identificando qué instrumento de la batería toca en un determinado momento. Este trabajo está centrado en los instrumentos habituales en el kit de batería pop/rock: caja, bombo, toms, hi-hat y platos. Este entorno puede ser entrenado para reconocer otros instrumentos percusivos o sonidos impulsionales.

Acknowledgements

Thanks to Axel Roebel for accepting me at the Sound Analysis and Synthesis team and to Marco Liuni for the support along my internship at IRCAM.

També donar les gràcies a l'Antonio Bonafonte pel seguiment al llarg del meu TFG i a en Ferran Marquès per l'ajuda facilitada com a tutor al llarg del meu pas per l'escola.

Finalment, gràcies als amics i familiars pel suport al llarg d'aquests mesos a París.

Revision history and approval record

Revision	**Date**	**Purpose**
0	March 2014	Document Creation
1	June	Document Revision I
2	August	Document Revision II

Document distribution list:

Name	**E-mail**
Jordi Pons i Puig	idrojsnop@gmail.com
Antonio Bonafonte	antonio.bonafonte@upc.edu
Axel Roebel	axel.roebel@ircam.fr

Contents

List of Figures

List of Tables

Abbreviations

NMF	**N**on-Negative **M**atrix **F**actorization
NMD	**N**on-Negative **M**atrix Factor **D**econvolution
IRCAM	**I**nstitut de **R**echerche et **C**oordination **A**coustique/**M**usique
PFG	**P**rojecte **F**inal de **G**rau (Degree's Thesis)
MIREX	**M**usic **I**nformation **R**etrieval **E**valuation e**X**change
ISA	**I**ndependent **S**ubspace **A**nalysis
NNSC	**N**on **N**egative **S**parse **C**oding
HMM	**H**idden **M**arkov **M**odel
SVM	**S**uper **V**ector **M**achine
PSA	**P**rior **S**ubspace **A**nalysis
ICA	**I**ndependent **C**omponent **A**nalysis
W	Patterns/basis
H	Activations
Q	Channel tracking matrix
V	Time-frequency representation to analyze
$\hat{\mathbf{V}}$	Estimated time-frequency representation
K	Number of components
N	Number of time-frequency frames
M	Number of time-frequency bins
KL	**K**ullback-**L**eibler divergence
IS	**I**takura-**S**aito divergence
ML	**M**aximum **L**ikelyhood
STFT	**S**hort **T**ime **F**ourier **T**ransform

Chapter 1

Context

1.1 Introduction

The thesis is carried out at IRCAM (Institut de Recherche et Coordination Acoustique/-Musique) in Paris. IRCAM is a public institution for music and acoustic research linked with the Centre Pompidou. Their research groups go from people working on contemporary music to signal processing groups and acoustics. This degree's thesis is developed at the Sound Analysis and Synthesis Team under the supervision of Axel Roebel, the Head Researcher, and Marco Liuni, Researcher.

1.2 Objectives

The purpose of this project is to develop a framework capable to do automatic drums transcription in a polyphonic audio mix using a common source separation algorithm called Non-negative Matrix Deconvolution (NMD from now on) [2].

1.3 Requirements and specifications

Event detection in audio mixes is an extremely complex problem due to the fact that multiple overlapped layers can be present in scenes: background music, ambience, and so on. Taking in account this fact, the project requirement is to *detect and classify the drum onsets in a polyphonic mix scenario where multiple concurrent layers are present.*

In order to quantify the performance of the drum onsets detection and classification we are going to use F-measure in relation to the accuracy of the transcription of the drum events:

$$P = \frac{tp}{tp + fp} \quad R = \frac{tp}{tp + fn} \quad F = \frac{2 \cdot P \cdot F}{P + R}$$

Where tp (true positives), fp (false positives) and fn (false negatives) are computed considering all the targets together. $P \in [0, 1]$ is precision, $R \in [0, 1]$ is recall and $F \in [0, 1]$ is F-measure. Note that for $F = P = R = 1$ a perfect transcription is achieved.

IRCAM's aim is to develop a software capable to overcome the state of the art: 0.67 of F-measure [3]. For more information about the state of the art, check Chapter 2.

1.4 Previous work

Some research has been done previously at IRCAM in the field of automatic drums transcription with Non-negative Matrix Factorization (NMF from now on) [4]. The idea is to change the approach: using NMD instead of NMF. NMD is an extension of NMF which is capable to identify patterns with a temporal structure. Due to this improvement, the new approach fits better in our research problem because the elements of the drum set have a determined temporal structure. The goal is to check if using this technique the state of the art could be overcome. For more information about NMD and NMF, check Chapter 2 and Chapter 3.

IRCAM is developing a source detection framework in multi channel audio streams which is based on Non-Negative Tensor Deconvolution (NTD from now on) [5]. The evaluation is made on 3DTV 5.1 film soundtracks with impulsive target sounds like gunshots. The idea is to adapt the IRCAM's 3DTVS algorithm to detect drum onsets in order to do automatic drums transcription.

This thesis is organized as follows: Chapter 1 briefly introduces context and organization; Chapter 2 describes the state of the art; Chapter 3 presents Non-negative Matrix Deconvolution and its ameliorations; Chapter 4 shows the implemented framework; Chapter 5 illustrates the results and Chapter 6 summarizes the conclusions and future work.

Chapter 2

State of the art

This chapter presents an overview of the current signal processing techniques for automatic drums transcription, focusing on the source separation approaches.

2.1 Main methods for drums transcription

Many methods for drums transcription are proposed in the literature. They can be classified in three main groups:

1. **Segment and classify**: These approaches consist on cutting the signal in small segments. After that, the goal is to classify the segments as drum or no-drum. In case of being a drum segment, to point which kind of drum is. Mainly, is used for mixes that only contain percussive events.

    - *Main approaches for segmentation*: People use onsets detection for defining event zones to be cut [6].
    - *Main approaches for classification*: Support Vector Machine (SVM) [6] or Hidden Markov Models (HMM) [7] with different features (MFCC, temporal centroid, spectral centroid, energy, and so on).

2. **Match and adapt**: This procedure consists in searching for the occurrences of a pattern in the time-frequency representation of the music signal. Then the pattern is adapted to the current shape taking in account the masking effects due to other instruments. Used for drums transcription in polyphonic mixes context.

- *Main approach*: Yoshii, Goto and Okuno proposed a method, AdaMast [8], for template adapting that won the MIREX05 contest [3].

3. **Separate/decompose and detect**: The last family relies on the idea that an audio input can be separated in independent sources in order to detect the onsets for each source separately. In that way, drum transcription can benefit from source separation techniques that would cancel the contribution of non-percussive instruments from the audio mix.

2.2 Separate/decompose and detect

For drums transcription in polyphonic music, the last approach seems a reasonable strategy due to the fact that splitting the audio mix in separated sound sources will help to avoid the interference of the non-percussive instruments over drums. Even that, in future steps of the development we decided to combine "segment and classify" and "separate/decompose and detect" in order to only process the interest segments.

Along this degree's thesis we are going to refer to matrices in bold: *e.g.*, $\mathbf{H}$; to vectors with an arrow: *e.g.*, $\vec{H}$; and to scalar values with italics: *e.g.*, k.

This family assume that the mixture spectrogram $\mathbf{V} \in \Re^{M \times N}$ results from the superposition of K source spectrograms $\mathbf{Y}_k$ of the same size as $\mathbf{V}$. Where K is the number of sources (targets to detect and classify), M is the number of frequency bins and N is the number of time frames. Further, it is assumed that each of the spectrograms $\mathbf{Y}_k$ can be represented by the outer product of basis ($\vec{W}_k$ of length M) with a time-varying gain ($\vec{H}_k$ of length N).

$$\mathbf{V} = \sum_{k=1}^{K} \mathbf{Y}_k = \sum_{k=1}^{K} \vec{W}_k{}^T \vec{H}_k$$

The main methods for separate/decompose differ in how the decomposition of $\mathbf{V}$ is achieved. The different decomposition methods rely on 3 main ideas: statistical independence of the source, sparseness and non-negativity. Listed in historical order in which they appeared:

- Independent Subspace Analysis (ISA) [9]: This approach assume the statistical independence of the sources. ISA is, in fact, a more relaxed Independent Component Analysis (ICA) which separates the input signal into additive sub components that are statistical independents. The advantage of ISA respect ICA is that in ISA the number of sensors does not need to be larger than or equal to the number of sources. That means, in audio, to have as many microphones as sources when, usually, there is only one "mono" channel available. One of the problems of ISA is that decompositions can get negative values which have a difficult physical interpretation.

- Non Negative Sparse Coding (NNSC) [10]: Incorporates the idea of sparseness as a constraint for the activations, and non-negativity as a constraint for basis and activations. In that way, non-negativity allows us to give a physical interpretation at the obtained results and sparseness helps us to obtain more representative activations, which is perfect for automatic transcription.

- Non-Negative Matrix Factorization (NMF) [1]: An effective factorization method for decomposing multivariate data under constraints of non-negative components. Sparseness criteria can be introduced easily as in NNSC.

For improving the performance of the methods previously described, it was introduced the Prior Subspace Analysis (PSA) [11] idea that consists on learning information of the sources. In that way, we can train our system to recognise some specific drum-kit basis; what seems a helpful strategy instead of recognizing them in situ.

Also, in more recent days Smaragdis presented the NMD [2] which is an extension for the NMF algorithm which is capable to identify components with temporal structure. In fact, one of the main characteristics of the drum-set elements is that after the onset they last a determined time. Using NMD instead of NMF, makes us capable to exploit the temporal structure as a "signature" of each drum (pattern).

The approach that concerns this research is based on NMD considering PSA. In that way we should be able to deal with polyphonic mixes and take benefit from the advantages of NMD.

Only some basic experiments with NMD implementing drums-transcription where published before this work: [12]. Interesting results where achieved with simple loops that

contain only elements of the drum-kit (not a polyphonic scenario). We think that envolving this technique with the appropriate framework could overcome the nowadays state of the art.

2.3 Comparing performance respect to the MIREX05

To check the performance of our framework, we can consider the results of the algorithms presented in the MIREX05 [3] contest as the nowadays state of the art. As far as we know, even if further researches have been done, their results don't improve significantly the ones obtained in that contest.

Presented algorithms in MIREX05, in order of final classification:

- Yoshii, Goto and Okuno [8]: 0.67 of F-measure. Based on a "match and adapt" algorithm in the time-frequency domain.
- Tanghe, Degroeve and De Beats [6]: 0.611 of F-measure. Based on "segment and classify" detecting onsets and then classifying those with a SVM.
- Dittmar [13]: 0.588 of F-measure. Based on onset detection, ICA, and a posteriori classification to give interpretation to the ICA results.
- Paulus [7]: 0.499 of F-measure. Based on "segment and classify" uses Hidden Markov Models (HMM) along the STFT frames.
- Gillet and Richard [14]: 0.443 of F-measure. Based on "segment and classify". Separates on frequency-bands and extracts features to classify with SVM or HMM.

The F-measure results given in MIREX05 are based on a mean of the F-measure of each target, all evaluated along the same database.

Another important characteristic of the MIREX05 Drums Detection contest is that the detection is based only in three targets: hi-hat, kick and snare. At the end of the Chapter 4 we give more details about the targets that we work with.

Chapter 3

Non Negative Matrix Factor Deconvolution

In this chapter are presented the main theoretical concepts related to the different tools and approaches used in the developed framework described in Chapter 4: how we work with NMD considering a PSA, some NMF considerations and the introduction of a NMF modification that allows us to be robust against small noisy values.

3.1 Non-Negative Matrix Factorization

As introduced in Chapter 2, NMF approach restricts the frequency domain basis functions and their gains to non-negative values.

The core of the method is based in the formulation defined as follows:

$$\mathbf{V} \approx \hat{\mathbf{V}} = \mathbf{WH}$$

Where $\hat{\mathbf{V}}$ is an approximation of $\mathbf{V}$, both are non-negative $M \times N$ matrix: $\hat{\mathbf{V}} \in \Re^{\geq 0, M \times N}$ and $\mathbf{V} \in \Re^{\geq 0, M \times N}$. $\mathbf{W}$ are the non-negative bases that represent the independent sources where $\mathbf{W} \in \Re^{\geq 0, M \times K}$ and $\mathbf{H}$ is the activation matrix where $\mathbf{H} \in \Re^{\geq 0, K \times N}$. K is the number of basis used for the decomposition. In Figure 3.1 there is attached an example where we can observe the behaviour of the decompositions.

It doesn't exist a closed-form expression for NMF estimations; is why an iterative method based on minimizing a cost function is used (see section 3.4 for more information about the cost function). A first method was proposed by Paatero [15] and more recently other algorithms were proposed by Lee and Seung [1]. First NMF implementation in audio was done for Smaragdis and Brown in polyphonic music transcription [16].

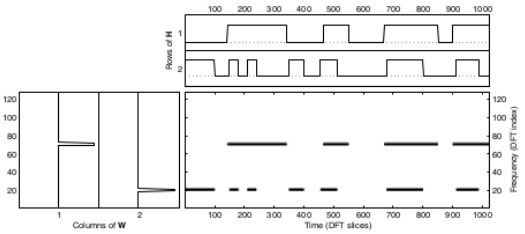

FIGURE 3.1: NMF example extracted from [2]: NMF on spectrograms. The lower right plot is the input magnitude spectrogram, it represents two sinusoids with randomly gated amplitudes. The two columns of W, interpreted as spectral bases, are shown in the leftmost plot. The rows of H, depicted at the top plot, are the time weights corresponding to the two spectral bases.

3.2 Non-Negative Matrix Factor Deconvolution

NMD approach is an extension of NMF that is really useful for automatic drums transcription. The fact that the basis describe the time-frequency evolution fits perfectly in our research problem.

From now on we are going to refer to the time-frequency signature used with the NMD (the equivalent of the basis in NMF) as "patterns".

The original formulation proposed by Smaragdis [2] is based on the formulation defined as follows:

$$\mathbf{V} \approx \hat{\mathbf{V}} = \sum_{t=0}^{lpattern-1} \mathbf{W}_t \cdot \overset{t\rightarrow}{\mathbf{H}}$$

Where $\hat{\mathbf{V}}$ is an approximation of $\mathbf{V}$, both are non-negative $M \times N$ matrix: $\hat{\mathbf{V}} \in \Re^{\geq 0, M \times N}$ and $\mathbf{V} \in \Re^{\geq 0, M \times N}$. $\mathbf{W}_t$ are the non-negative patterns that represent the independent sources where $\mathbf{W}_t \in \Re^{\geq 0, M \times K}$ and $\mathbf{H}$ is the activation matrix where $\mathbf{H} \in \Re^{\geq 0, K \times N}$. K is the number of patterns used for the decomposition and *lpattern* is the length of the pattern. Notice that if $lpattern = 1$, we go back to the particular case of NMF.

The operator $\overset{i\rightarrow}{(\cdot)}$ shifts the columns to the right. So that:

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix}; \overset{0\rightarrow}{A} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix};$$

$$\overset{1\rightarrow}{A} = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 0 & 5 & 6 & 7 \end{bmatrix}; \overset{2\rightarrow}{A} = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 0 & 5 & 6 \end{bmatrix};$$

In the same way, we can define the inverse operation $\overset{\leftarrow i}{(\cdot)}$.

In Figure 3.2 there is attached an example where we can observe the convolutive behaviour of the decompositions; see that the spectrogram is the result of the sum of "convolutions" between $\vec{H_k}$ and $\mathbf{W_k}$ where $k \in [1, 2]$.

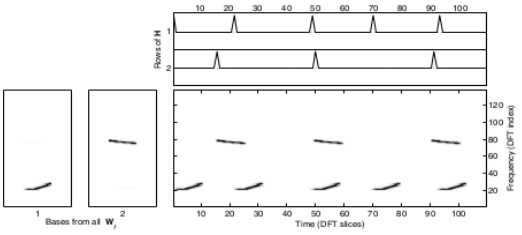

FIGURE 3.2: NMD example extracted from [2]: A spectrogram and the extracted NMD bases and weights. The lower right plot is the magnitude spectrogram that we used as an input to NMD. The two leftmost plots are derived from W, and are interpreted as temporal-spectral bases. The rows of H, depicted at the top plot, are the time weights corresponding to the two bases. Note that the leftmost plots have been zero-padded in these figures from left and right so as to appear in the same scale as the input plot.

3.3 Comparing NMF vs. NMD

In Figure 3.1 we can observe that **H** reflexes the amount of time that a pattern is present in the mix. But the elements of the drum-kit are impulsional sounds that lasts a determined time: our interest is in detecting where the drum sound starts. The kind of **H** resultant after the NMD decomposition is more interesting for our goal. Ideally (in controlled conditions like in Figure 3.2), we will get $\delta(t)$'s representing our target in the activation matrix; where $\delta(t)$ denotes the Kronecker delta function. This behaviour will be perfect for the detection stage, because there is no need to pre-process **H**; meanwhile, in a NMF context, an onset detection is needed before applying a peak-picking system to detect.

3.4 The cost function

Different cost functions for NMF are used in the literature. The first one was proposed by Paatero and Tapper [15] based on the euclidean distance:

$$d_{euc}(V,\hat{V}) = \sum_{m=1}^{M}\sum_{n=1}^{N}\frac{(V_{m,n}-\hat{V}_{m,n})^2}{2}$$

Lee and Seung [1] introduced other widely used cost function; based on the Kullback-Leibler divergence the following distance has been defined (KL from now on):

$$d_{divKL}(V,\hat{V}) = \sum_{m=1}^{M}\sum_{n=1}^{N}(V_{m,n}\log\frac{V_{m,n}}{\hat{V}_{m,n}} - V_{m,n} + \hat{V}_{m,n})$$

The presentation of these cost functions was together with an update rules that ensure convergence if $\mathbf{W}$ and $\mathbf{H}$ are non-negative.

Those two costs functions are part of the β-divergence [17] family parametrized by a single parameter β that describes the Euclidean distance ($\beta = 2$), the KL ($\beta = 1$) and the Itakura-Saito divergence ($\beta = 0$).

$$d_\beta(x|y) = \begin{cases} \frac{1}{\beta\,(\beta-1)}\left(x^\beta + (\beta-1)\,y^\beta - \beta\,x\,y^{\beta-1}\right) & \beta \in \mathbb{R}\backslash\{0,1\} \\ x\,(\log x - \log y) + (y-x) & \beta = 1 \\ \frac{x}{y} - \log\frac{x}{y} - 1 & \beta = 0 \end{cases}$$

This family has a single minimum in $\mathbf{V}$=$\hat{\mathbf{V}}$, which is essential to estimate properly the factorized spectrogram.

Notice that the β-divergence acts as a distance with $\beta = 2$, but for other β is not symmetric and we refer it as divergence.

A noteworthy property of the β-divergence is his behaviour with respect to scale:

$$d_\beta(\gamma x|\gamma y) = \gamma^\beta d_\beta(x|y)$$

So, for $\beta = 0$ (Itakura-Saito divergence) is scale-invariant. That means that the low energy components have the same relative importance as high energy components. This is important in music processing because a bad fit of the factorization for a low-power

coefficient will cost as much as a bad fit for a higher power coefficient. Audio spectra, especially drums, exhibit exponential power decrease along frequency and time (see Figure 3.3).

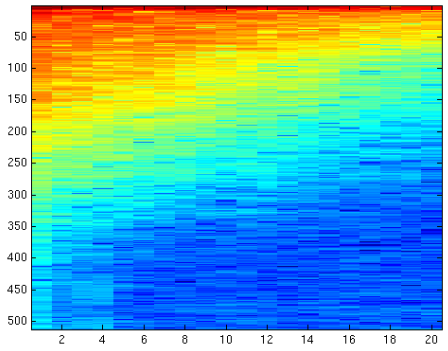

FIGURE 3.3: Spectrogram of a kick drum, where we can see the exponential decay and the low-power spectra parts.

Another important characteristic is that the β-divergence is convex on $\Re^+$ if and only if $1 \leq \beta \leq 2$ (see Figure 3.4). That means that if we use Itakura-Saito divergence ($\beta = 0$) as cost function, our decomposition is prone to a local minima.

3.4.1 Choosing Itakura-Saito divergence

The scale invariance property described above, seems a really interesting and strong argument to use IS as cost function.

In addition, Févotte in [18] demonstrates that using IS-NMD over the power spectrogram can be viewed as a maximum likelihood estimation of $\mathbf{W}$ and $\mathbf{H}$ if we consider the following statistical audio generative model:

$$\vec{x_n} = \sum_{k=1}^{K} c_{k,n} \qquad c_{k,n} \sim \mathcal{N}_c(0, h_{kn}\text{diag}(\vec{w_k}))$$

where $\vec{x_n}$ refers to the n STFT frame, $n \in [1, N]$, $\mathcal{N}_c(\mu, \sigma)$ denotes a complex Gaussian distribution of each $c_{k,n}$ component and K is, as usual, the number of components (number of patterns/basis).

Other work at IRCAM goes in the same direction: Grégoire Lafay et al. in [19] did an interesting study comparing the performance of the Euclidian distance, KL and IS for

event detection (unsupervised clustering) in audio scenes using spectral features where they conclude, too, that IS offers the best results.

Even if Itakura-Saito divergence (IS from now on) is more prone to fall in a local minima respect to the others, the results above shows that using IS seems reasonable in an audio context.

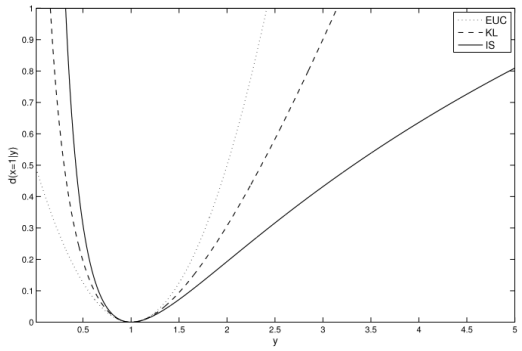

FIGURE 3.4: Extracted from [18]: Euclidean, KL and IS costs $d(x|y)$ as a function of y and for $x = 1$. The Euclidean and KL divergences are convex on $(0, \infty)$. The IS divergence is convex on $(0, 2x]$ and concave on $[2x, \infty)$

3.5 Update rules

The update rules used here are adapted from the NTD ones detailed in the 3DTVS framework [5] for the mono case. Considering the β-divergence as cost function, $\mathbf{V}$ should minimize the following equation:

$$J(\mathbf{V}||\hat{\mathbf{V}}) = \sum_{m=1}^{M}\sum_{n=1}^{N} d_\beta(\mathbf{V}_{m,n}|\hat{\mathbf{V}}_{m,n})$$

By setting the appropriate step in a gradient descend, the following multiplicative update rules can be deduced:

$$\mathbf{W}^t \leftarrow \mathbf{W}^t \circledast \frac{\left(\mathbf{V} \circledast \hat{\mathbf{V}}^{\circledast(\beta-2)}\right) \circ \mathbf{H}^{t\rightharpoonup}}{\hat{\mathbf{V}}^{\circledast(\beta-1)} \circ \mathbf{H}^{t\rightharpoonup}}$$

$$\mathbf{H} \leftarrow \mathbf{H} \circledast \frac{\sum_t \left(\mathbf{V} \circledast \hat{\mathbf{V}}^{\circledast(\beta-2)}\right)^T \circ \mathbf{W}^{t\rightharpoonup}}{\sum_t \left(\hat{\mathbf{V}}^{\circledast(\beta-1)}\right)^T \circ \mathbf{W}^{t\rightharpoonup}}$$

The $\circ$ symbol denotes the outer product, while $\circledast$ is the Hadamard product and powers of matrices indicated with $\circledast(\cdot)$ are element-wise.

Our research problem is not only about source separation, the transcription implies to extract information from the activation matrix. The energy of the patterns is normalized to one after each iteration ($l_1 - norm$ of the power spectrogram) so that we are able to derive the energy from H.

3.6 Time-frequency representation

V corresponds to the time-frequency representation of the audio to analyse, two kinds were considered: power spectrogram and MEL spectrogram.

3.6.1 Power spectrogram

As already mentioned in previous sections, Févotte [18] demonstrated that using the power spectrogram in a IS-NMF context, corresponds to a ML estimation. That's the reason why the choosed time-frequency representation is the power spectrogram.

3.6.1.1 STFT applied parameters

The following parameters are the ones used for implementing the algorithm:

- *Sampling rate*: 44100 Hz. (*fs* from now on)
- *Length of the window*: 1024. (*lw* from now on)
- *Type of window*: Hanning.
- *Overlapping*: 75%. (*%ov* from now on)
- *Number of points for STFT*: 1024. Which makes a frequency resolution of 512. In fact, this parameter defines *M*.

Taking in account the length of the window and the overlapping, we define N as:

$$\left\lfloor \frac{ls - lw}{[1 - \frac{\%ov}{100}] \cdot lw} \right\rfloor + 1$$

Where *ls* is the length of the signal and N is the number of frames.

3.6.2 MEL spectrogram

The MEL spectrogram is obtained combining the bins of the spectrogram considering the perceptual properties of the human auditory system.

In the study of Grégoire Lafay et al. [19] they also compare performance between different spectral representations using IS for unsupervised classification of audio events: spectrogram (with 1024 bins), MEL spectrogram (with 40 bands) and MFCC (with 13 and 40 coefficients). The best results were obtained with the spectrogram, but we can observe that the MEL spectrogram results are not significantly different.

Notice that the spectrogram uses 1024 bins meanwhile the MEL spectrogram uses 40 coefficients. As the framework based on the IS-NMD power spectrogram is computationally expensive, to use the MEL spectrogram as the time-frequency representation is as a good way to reduce the high computational cost.

3.6.2.1 MEL spectrum applied parameters

First, we consider the previous described applied parameters to the STFT power spectrogram. Then, we map the powers of the spectrum obtained above onto the MEL scale, using 40 triangular overlapping windows.

This mapping will lead us onto a 40 bands MEL representation ($M = 40$).

3.7 PSA: a priori learned patterns

The patterns are described along the matrix $\mathbf{W}$. This matrix is formed by two parts:

1. Fixed Trained Patterns ($\mathbf{W}_{tar}$): those trained patterns pretend to model the target, based on the idea of Prior Subspace Analysis. We can understand them as a priori trained dictionary of patterns. Section 4.1 explains how the patterns are learned.

2. Adaptive Patterns ($\mathbf{W}_{bgnd}$): to have a coherent description of the background scene: polyphonic instruments, voice, and so on.

The trained dictionary ($\mathbf{W}_{tar}$) is included with the initialisation of $\mathbf{W}$ for the NMD. This dictionary is formed by K_{tar} elements. The components in the target dictionary $\mathbf{W}_{tar}$ are not updated during the iterative NMD decompositions, and they should interpret all of the energy of the target events.

The background adaptive components are randomly initialised. The aim here is to obtain a decomposition where the activations of the target are separate from the ones of the rest of the scene. They are adaptive, because they should model the unknown background events in order to avoid interference of the background in the $\mathbf{H}_{tar}$.

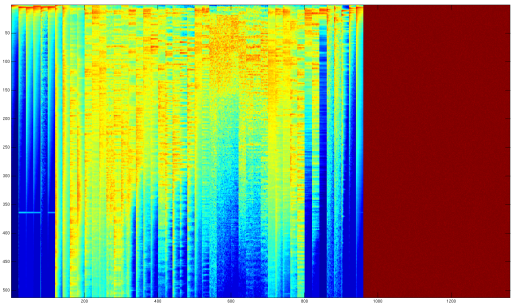

FIGURE 3.5: $\mathbf{W}$ matrix: background and target patterns.

In Figure 3.5 we can observe the two parts of the $\mathbf{W}$, the first ones correspond to the chain of the fixed trained patterns ($\mathbf{W}_{tar}$) and the second part corresponds to the random initialised background patterns ($\mathbf{W}_{bgnd}$).

K is the total number of basis, a sensitive parameter to set, where the remaining number of patterns are the dedicated to model the background: $K - Ktar = Kbgnd$. The total number of patterns has to be large enough to provide an exhaustive description of the audio scene to avoid the activation of the target dictionary for non-target events.

The length of pattern is another parameter to choose. This parameter is called *lpattern* and is set to 20 frames. That means:

$$\frac{lw + [lw - lw \cdot \frac{\%ov}{100}] \cdot (lpattern - 1)}{fs} =$$

$$= \frac{1024 + 256 \cdot 19}{44100} = 0.1335 \; seconds$$

Longer patterns could be tried for dealing with the larger decay time of the cymbals, but this will increment the processing time.

3.8 NMD modification: robustness against noise

The main issue along this thesis was related to a code line applied in each NMD iteration (after updating $\mathbf{W}$) that set to 10^{-10} all the values smaller to 10^{-10} for avoiding not a number (NaN) values.

The side effect of thresholding $\mathbf{W}$ is that affects dramatically the results of the factorization (see Appendix A.4 for more detailed information).

Another solution should be found to be robust against NaN's.

Checking the update rules:

$$\mathbf{W}^t \leftarrow \mathbf{W}^t \circledast \frac{\left(\mathbf{V} \circledast \hat{\mathbf{V}}^{\circledast(\beta-2)}\right) \circ \mathbf{H}^{t\rightharpoonup}}{\hat{\mathbf{V}}^{\circledast(\beta-1)} \circ \mathbf{H}^{t\rightharpoonup}}$$

$$\mathbf{H} \leftarrow \mathbf{H} \circledast \frac{\sum_t \left(\mathbf{V} \circledast \hat{\mathbf{V}}^{\circledast(\beta-2)}\right)^T \circ \mathbf{W}^{t\rightharpoonup}}{\sum_t \left(\hat{\mathbf{V}}^{\circledast(\beta-1)}\right)^T \circ \mathbf{W}^{t\rightharpoonup}}$$

The $\circ$ symbol denotes the outer product, while $\circledast$ is the Hadamard product and powers of matrices indicated with $\circledast(\cdot)$ are element-wise.

We can observe that NaN's can only be introduced by $\hat{\mathbf{V}}$. Adding an insignificant constant value to $\hat{\mathbf{V}}$ and $\mathbf{V}$ each time is computed, is enough to avoid NaN's.

Notice that we are adding this value in numerator and denominator to keep the ratio, which is in fact the principal of the IS divergence.

It is important to highlight that if we add an insignificant constant value to $\hat{\mathbf{V}}$ and $\mathbf{V}$ the global optimum remain the same.

This modification of the code is very significant, due to the fact that the decompositions that come out of the developed framework are now logical and understandable.

Meanwhile evaluating possible side effects as result of this new tract to avoid NaN's, we noticed that introducing a bigger value could help us to control robustness against noise, *e.g.*:

Considering V=[5,0.1] $\hat{V}$=[2.5,0.05] and β=0 (IS divergence):

$$D_{\beta=0}(V|(Np)\hat{V}) = \sum_{\in V} \frac{V}{\hat{V}} - \log\frac{V}{\hat{V}} - 1$$

The following cost value is obtained:

$$D_{\beta=0}(V|\hat{V}) = (\frac{5}{2.5} - \log\frac{5}{2.5} - 1) + (\frac{0.1}{0.05} - \log\frac{0.1}{0.05} - 1) = 0.3068 + 0.3068 = 0.6136$$

But if we add a bigger value (from now on we are going to refer to it as noise parameter):

$$D_{\beta=0}(V+0.2|\hat{V}+0.2) = \sum_{\in V} \frac{V+0.2}{\hat{V}+0.2} - \log\frac{V+0.2}{\hat{V}+0.2} - 1 =$$

$$= (\frac{5+0.2}{2.5+0.2} - \log\frac{5+0.2}{2.5+0.2} - 1) + (\frac{0.1+0.2}{0.05+0.2} - \log\frac{0.1+0.2}{0.05+0.2} - 1) =$$

$$= 0.2705 + 0.0176 = 0.2881$$

As you can observe in the previous example, this parameter acts decreasing the significant impact of the values below/near the *noise parameter*; meanwhile for the values far to the *noise parameter* still keeps the prominence of their cost. This behaviour is really interesting to avoid including in the cost the small random noisy parts of $\mathbf{V}$.

Here we have defined how this *noise parameter* (Np) works as a threshold that considers less cost for the values near the Np:

$$D_{\beta=0}(V+Np|\hat{V}+Np) = \sum_{\in V} \frac{V+Np}{\hat{V}+Np} - \log\frac{V+Np}{\hat{V}+Np} - 1$$

The underlying idea of this model modification relies on adding a constant pattern to the classic NMF model:

$$\mathbf{V} \approx \hat{\mathbf{V}} = \sum_{k=1}^{K} \vec{W}_k^T \vec{H}_k + \vec{W}_0^T \vec{H}_0 = \mathbf{WH}$$

Where K, $\mathbf{W}$ and $\mathbf{H}$ are as usual, $\vec{W}_0$ is the constant *noise parameter* base and $\vec{H}_0$ is the activation vector that allows the base to be activated along all the audio spectrogram.

Assuming the previous described NMF model means that:

1. Even the scale invariance property do not hold anymore, with an appropriate choice of the Np it is still approximately respected.
2. The ML estimation introduced by Févotte still applies.

3.8.1 Noise parameter: applied parameters

In order to use the noise parameter to avoid noisy small values, it will be set -60dB from the maximum of the considered time-frequency representation.

3.9 Summary: applied NMD conditions

An IS-NTD modification based on adding a *noise parameter* is used to decompose a time-frequency representation (MEL spectrogram) of the audio to analyse.

Two different kinds of patterns are used: learned-fixed (for drum-kit instruments target) and adaptive (to model the unknown present background audio).

Along this thesis we are going to compare the use of two time-frequency representations: power spectrogram or MEL spectrogram.

Chapter 4

Developed framework

The developed framework is conformed by two parts: training and detection. The training part is divided in two stages: learning the patterns and determining the thresholds.

In Appendix A is attached all the information related to the different steps of the research: different approaches, partial results and interesting (and didactic) observations. The information there could help the reader to understand better why those approaches are useful.

4.1 Training Patterns

4.1.1 Objective

The objective is to implement a training algorithm that leads to the minimum number of trained patterns that represents properly our training data-set. The resulting trained patterns should represent itself a complete element of the target class.

We don't want to allow the algorithm to split the target event in patterns that could represent other targets that no longer belong to the target-class. For example, we can consider a splash which is a drum-instrument that has a large band with spectra. If we don't ensure that we represent the whole event in one pattern, it could be splitted in two (low frequencies and high frequencies, for example). This means that the low frequencies resulting pattern could be used to represent a kick, what will lead us to false positives.

4.1.2 Algorithm

This procedure is an unsupervised β-divergence *k-means* clustering, where we split the training files in classes. For each class we find a centroid which will be the trained pattern. A common k-means clustering strategy that alternates two steps (class assignement and update centroids) is used to cluster our training space.

The training-sets are formed by J audio files with isolated target events of a certain drum class. That means that this algorithm should be run separately for each element of the drum-set: kick, snare, open hi-hat, closed hi-hat, and so on.

In the following lines the proposed algorithm is outlined and the details are provided afterwords.

For each different target:

1. **Load data**. Cut the J isolated drum events from the point that has maximum energy till we reach the length of the pattern (*lpattern*).
2. **Compute time-frequency representation**.
3. **Normalize** each training-clip spectrogram ($l_1 - norm$) to avoid scalar factors that could influence our similarity matrix.
4. **Compute Np**: set the *Np* in a common global reference point (the max of the training dataset -60db).
5. **For each *k* from 1 to *kmax***: testing with different number (k) of patterns-classes to get the best configuration. The obtained centroid for each class corresponds to the learned pattern.
   (a) Inicialize k centroids.
   (b) Given the initial set of k centroids the algorithm alternates between two steps till convergence:
      i. Find members of each *k-class*: using $d_\beta(\mathbf{X}_j|\mathbf{C}_k)$ where $\mathbf{X}_j$ is each training clip and $\mathbf{C}_k$ is each centroid.
      ii. Update centroid: for each *k-class* compute the NMD considering the members of the class as input (a chain of them) to factorize with only

one adaptive pattern. The resulting pattern **W** is the centroid of the *k-class*, which is in fact the pattern we are searching. In this step we enforce sparseness in the same way as introduced in [5]: constraining $\mathbf{H_{ini}}$. Imposing a time grid on $\mathbf{H_{ini}}$ where is set to 1 where each member of the class begin and to 0 all the others.

(c) Compute the NMD with the $\mathbf{C}_k$ learned patterns along the J chained files and save relevant performance data. In this step is considered as $\mathbf{H_{ini}}$ the energy contour (the sum over bins, which is in fact an approximation of the energy for each frame) of the input spectrogram and fixed **W**. A post processing of the **H** matrix is applied to consider the contribution of the secondary activations as part of the event.

6. **Choose minimum number of *k*'s** depending on the performance data computed in step 5.c.

The initialization is a sensitive step where a bad setting could influence importantly the final clusterings. Diferent scenarios are considered:

1. $K = 1$: A β-divergence mean is computed for all the training files.

2. $K \neq 1$: As we are testing different combinations of $k \in [1, kmax]$, we are considering the previous computed centroids as inicialization. To add a new class, the worst represented one is splitted in two. We have two criterias for consider the worst represented class:

   (a) $\max_k \|d_\beta(X_{j,k}|C_k)\|$. Where the $\|\cdot\|$ operator is the mean.

   (b) $\max_k \quad \max_j d_\beta(X_{j,k}|C_k)$

   Once the worst class is idenified, a *β-k-mean* clustering (with $k = 2$) is run within the class to split considering as initialization a *β-k-mean* setting as initial centroids the two most different training clips.

In step **5.c** we pretend to find the a centroid representation for all the members of each *k-class*. Which is equivalent to find **P** solving:

$$\hat{\mathbf{P}} = \arg\min_{\mathbf{P}} \sum_{m=1}^{M} d_\beta(\mathbf{P}, \mathbf{X}_m)$$

Where M is the number of members of the class, $\mathbf{P}$ represents the centroid that is going to be considered as the trained pattern and $\mathbf{X}_m$ is the spectrogram of each training clip class member. We expect that solving this optimitzation problem, considering as distance the β-divergence, gives similar results as those obtained by running the β-NMD with an input that contains all $\mathbf{X}_m$ and leading it to update with only one adaptive pattern ($\mathbf{P}$) to solve it.

To give all the samples to the β-NMD as input, the clips are concatenated in a single-channel audio file.

The post processing of $\mathbf{H}$ used in 5.c is to consider the contribution of the secondary activations as part of the event. Is implemented with a convolution of $\vec{H}_{target}$ with [1,1]. This approach follows the idea that the activation after an onset ($\vec{H}_{target}[n+1]$) contributes to explain the same onset ($\vec{H}_{target}[n]$).

To sum up, the previous procedure is an unsupervised β-divergence *k-means* clustering for a specific application: training patterns in a IS-NMF context.

4.1.2.1 Discussion: choosing *k*

Notice that:

- As result of the chaining, we know where the onsets are situated: along $\mathbf{H}$ it exists a controlled grid of onsets (we will refer to those positions where the onsets are situated as the *inGrid* samples and as *outGrid* to the others).
- As result of the normalization and the chaining we know that (if a perfect representation is achieved) the sum of activations would be one on the grid of onsets and zero to others.

As described in the previous algorithm, some performance data is extracted to select k:

- *Final cost value*: as result of the NMD in 5.d a cost value is obtained.
- *min(inGrid)*: as is known that the best scenario achieves activations to one on the grid of onsets, an interesting parameter could be the worst activation in the grid of onsets: *e.g.*, accepting a determined number of k if $min(inGrid) > 0.45$.
- *min(inGrid)/max(outGrid)*: to avoid false positives due to bad representations, an interesting parameter could be the relation with the *min(inGrid)* with the *max(outGrid)*. In fact, the *max(outGrid)* is the more prominent false activation due to bad representation of the patterns: *e.g.*, accepting a determined number of k if $min(inGrid)/max(outGrid) > 2$.

The two last options seems the more interesting ones because setting a criteria around the **H** matrix seems more reasonable due to the fact that we are going to use **H** to take decisions.

4.1.3 Observations

As result of the described training patterns step it could be that for the same target exists more than one pattern (k) to represent its target space. Due to this fact, along this project we use the sum of **H** over the K dimensions related to the interest target to take decisions.

Notice that no mathematical sparseness constraints are applied in the equations of the NMD model. In this framework we can consider that we are imposing a semantic sparseness instead of the common mathematical sparseness criteria along the cost function.

A big gain (10-15%) in F-measure came after removing the cross-stick class. For this class we didn't had many training files and the results after training were not the expected.

A gain of 5% came after a simplification of the database. Especially the splash class, where it where 'reversed' splashes, and the snare, where there were sounds of the snare similar to the toms.

4.2 Training Thresholds

4.2.1 Objective

The goal is to find the better thresholds (one for each drum-instrument target) along a representative training mix.

4.2.2 Adaptive thresholds

To make them adaptive, two multiplying factors are applied to a threshold that depends on the local energy of the audio file to analyse.

So: a multiplicative factor for each target (*e.g.*, closed hi-hat or snare) is learned.

4.2.3 Generating a training mix

It consists on concatenating several training audio files (and their annotations) into a training mix that represents different kinds of styles and techniques.

The database [Db3, described at Appendix B] used for generating the training mix contains polyphonic audio mixes (percussive targets and harmonic background instruments together). We don't use only drums audio files for learning the multiplicative factors because our threshold depends in the local energy of the signal to analyse. That means that for a scenario with only drums to detect (without background harmonic instruments) the local energy is less than the used to train and we will detect properly the events because the thresholds well be low.

4.2.4 Learning: optimization

An multi-step optimization is implemented for training each of the target thresholds. A first test is carried out with a grid of possible thresholds where the candidate that fits better is selected. Around that first candidate a second step optimization with more resolution is done (see Figure 4.1). We repeat this operation till we obtain F-measure 1 or a predetermined number of iterations is raised.

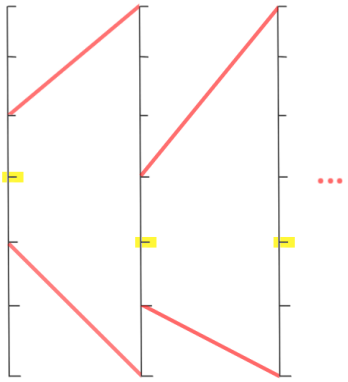

FIGURE 4.1: Graphical definition of the multi-step optimization. In yellow, the threshold that fits better on the proposed grid. After each iteration, around the best candidate, a grid with higher resolution is applied.

4.2.5 Taking decisions

The optimization it has to be carried out in a detection context, the detection conditions of our framework are detailed thereafter. The optimization of the multiplicative factors is carried out at in section 4.b.i considering a ground truth.

4.3 Detecting

4.3.1 Decomposing only interest zones: motivation

A large amount of processing time is required using NMD with a high number of patterns. For reducing the computational time, we decided only to analyse the interest zones: the onsets zones.

From the point of view of speed, we also observed that the algorithm process faster a signal if we compute it by segments instead of processing it at once. In that way, processing the onset zones separately, we expect a lower computational cost.

In addition, processing only the onsets zones we can ensure that outside of those zones we will never detect.

4.3.2 Estimating K_{bgnd}

K_{bgnd} is estimated before each NTD iteratively till a good decomposition is obtained: the K_{bgnd} patterns are random initialised and incremented till K_{bgnd} is sufficient to

obtain a proper decomposition. The quality of a decomposition is evaluated considering the mean of the cost of each of the bins at each time position. We consider that if this quality parameter (Q from now on) is below 0.01, K_{bgnd} is sufficient to obtain a good decomposition. To estimate K_{bgnd}, the learned patterns (W_{tar}) are not used; what means that K_{bgnd} is sufficient to represent everything including drums. This strategy is used because estimating K_{bgnd} using an NTD is computationally expensive; if the W_{tar} patterns are not included, the estimation is faster and sufficient.

4.3.3 Implemented Algorithm

1. **Load audio file**:
   (a) Down-mix from stereo to mono.
   (b) Normalize: $[\sum \vec{signal}^2]/length(\vec{signal}) = 1$
2. **Find onsets zones** using a onset detection by means of transient peak classification [20]. The interest zones are defined from the beginning of the transient till the end of the transient plus half of the analysis window (zone where the onset is supposed to be according to the algorithm defined at [20]). To fit this zone in a NMD context, we add *lpattern-1* frames at the end in order to model the tail of the onset (tail zone).
3. **For each onset zone**:
   (a) Compute the time-frequency representation: power spectrogram or MEL spectrogram.
   (b) Calculate the energy of the segment $Eseg = (\sum\sum \mathbf{V}^2)/length(\mathbf{V})$.
   (c) Set $\mathbf{H}_{ini}$ as the energy contour of the spectrogram (the sum over bins, which is in fact an approximation of the energy for each frame). It seems the better configuration that allows us to start with a slow cost function value.
   (d) Generate $\mathbf{H}_{mask}$. A mask for $\mathbf{H}$ is defined in order to discard the tail zone activations where is not expected to be the onset.
   (e) Compute the NMD and apply the $\mathbf{H}_{mask}$ to the computed $\mathbf{H}$ in order to take only into account the interest zones.

To be able to take decisions considering all the information of the interest zones, a chain is formed.

4. **Taking decisions** along the previous formed chain.

   (a) Defining threshold: $mean(\vec{H_{tar}})$. Where $\vec{H_{tar}}$ is the activations of the current target to analyze.

   (b) For each drum instrument:

      i. Obtain the threshold for each drum instrument: multiply the threshold with the corresponding multiplicative factor.

      ii. Obtain the corresponding $\vec{H}_{target}$: sum down the **H** components that correspond to the drum instrument to analyse.

      iii. Convolve $\vec{H}_{target}$ with [1,1] to consider the contribution of the secondary activations as part of the event: $\vec{H}_{filtered}$.

      iv. If the $\vec{H}_{filtered}$ of the instrument is over the threshold inside an interest zone, we detect our target in the frame where $\vec{H}_{filtered}$ is max along the interest zone. The interest zone is defined by $\mathbf{H}_{mask}$.

In Figure 4.2 we can observe clearly two onset zones in a zoomed example where where we can see the tail zones (in red) and the interest zones (in green).

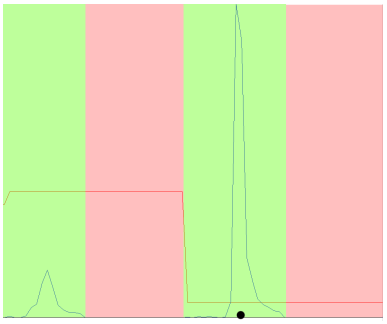

FIGURE 4.2: A zoomed example of the detection step: chained $\vec{H}_{target}$ for a specific target (blue line), its threshold (red line) and detections (black point). The interest zone is in green and the tail zone is in red.

In the Figure 4.3 we can see the chained $\vec{H}_{target}$ of a drum instrument and its associated threshold. The threshold (the line in red) corresponds to:

$$mean(\vec{H_{tar}})$$

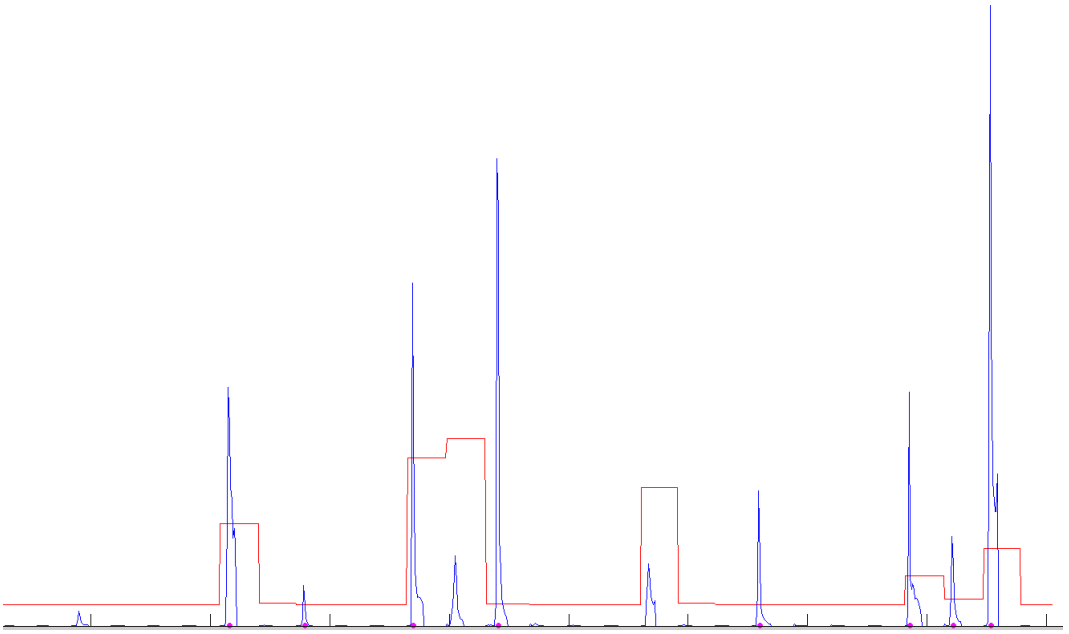

FIGURE 4.3: Chained $\vec{H}_{target}$ for a specific target (blue line), its threshold (red line) and detections (magenta points).

We can also observe the utility of adding this offset to the threshold: the lowest activations will lead us to false positives if no offset was included.

4.3.4 Evaluation

The drum sets are conformed of more than one target: hi-hat, low tom, snare drum, bass drum, splash, and so on. Our algorithm has to be able to detect all the drum instruments simultaneously and to give the results together: a global F-measure is implemented to evaluate performance.

Other approaches to check the performance are presented in the literature. For example in MIREX05, the F-measure that they refer is defined by the average of the separated F-measure for kick, snare drum and hi-hat. Considering the toms as a kick, the cymbals as hi-hats and the cowbell as snare drum (Figure 4.4 illustrates this hierarchical classification).

Our goal is to detect all the elements of the drum-set separately: working on Level 1.

To compare our results with the nowadays state of the art, hierarchical constraints are programmed with three levels: Level 1 is without constraints and Level 3 is grouping all the activations of the same family (see Figure 4.4). Of course, we expect to get better

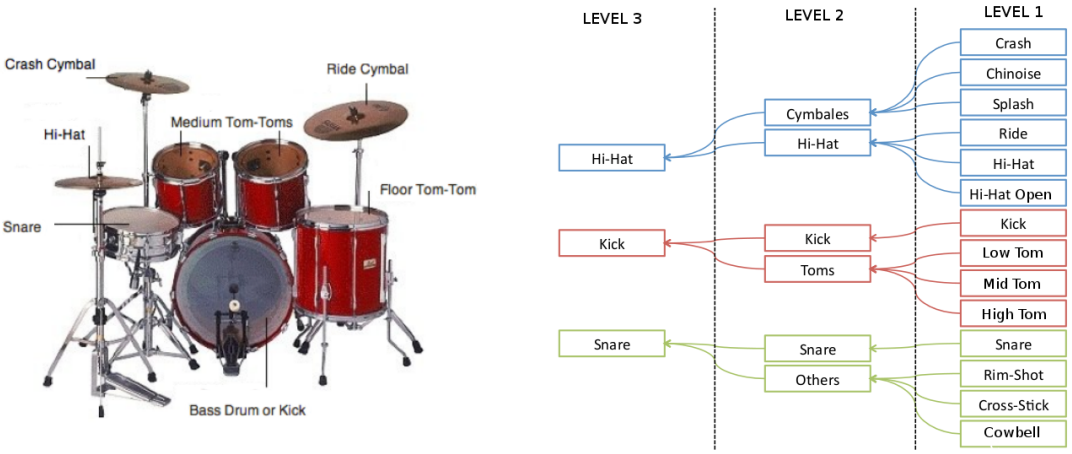

FIGURE 4.4: Graphical definition of the hierarchical classification.

results after grouping because with this strategy we avoid confusions due to cross-talk influence between similar elements of the drum set.

Chapter 5

Results

5.1 Training Patterns

Train database: Db1 **Drum-kit instrument**	**Number of patterns** **MEL spectrogram**
Kick	6
High Tom	5
Low Tom	10
Mid Tom	2
Closed Hi-hat	13
Open Hi-hat	15
Ride Cymbal	9
Chinese Cymbal	5
Crash Cymbal	25
Splash Cymbal	23
Cowbell	4
Snare Drum	10

TABLE 5.1: Results: training patterns

5.2 Training Thresholds

Trained considering the chain of the elements of the training thresholds database [Db3].

Train database: Db3 **Drum-kit instrument**	**Multiplicative factor MEL spectrogram**
Kick	4.8852
High Tom	15.3685
Low Tom	8.61856
Mid Tom	61.6336
Closed Hi-hat	4.25873
Open Hi-hat	21.3877
Ride Cymbal	22.64617
Chinese Cymbal	20.416
Crash Cymbal	7.8687
Splash Cymbal	9.8725
Cowbell	18.0204
Snare Drum	7.9618

TABLE 5.2: Results: training thresholds

5.3 Detection

5.3.1 MEL spectrogram

For polyphonic mixes, evaluated along the Db7:

Hierarchical level	Precision	Recall	F-measure
1	0.55073	0.57977	0.56095
2	0.63518	0.667	0.6462
3	0.68952	0.69808	0.68922

TABLE 5.3: Results with MEL spectrogram: polyphonic mixes

Chapter 6

Environment Impact

This projects inherits directly the code developed by Marco Liuni for the 3DTVS project. All the ameliorations introduced along the decomposition functions, new approaches in training patterns and new approaches for detection/thresholding will benefit directly the 3DTVS project.

This work should be interesting too for the Music Information Retrieval community as is the first work using NMD for Automatic Drums Transcription in a polyphonic scenario.

Smaragdis [2] in 2004 did a first experiment with synthetic drum mixes; more recently, in 2012 Linsay-Smith et al.[12] did the first steps towards obtaining a transcription system for drum mixes with encouraging results. As far as we know, those are the more recent publications that use NMD for automatic drums transcription.

Due to that, the results of this research project are going to be submitted as a paper for the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2015.

Chapter 7

Conclusions and future development

7.1 Conclusions

We can observe that the results for polyphonic mixes in H3 reach the MIREX05 Drums Detection [3] ones attached in Section 2, which is encouraging to keep working on get better results over H2. Notice, but, that the presented results don't consider the same test data set as MIREX05 Drums Detection.

This thesis describes the first steps for doing automatic drums transcription using IS-NMD in a realistic polyphonic scenario.

A couple of novelties in the field of NMD are introduced:

- Presents a modification of a common used cost function (the IS divergence) that, with an appropriate choice of the presented *noise parameter*, aims to increase noise robustness.
- A method to deal with polyphonic music in a PSA-NMD context: first, describing an efficient algorithm for learning the dictionary of patterns and then releasing an iterative method that finds the adequate number of background patterns (K_{bgnd}) to represent the whole audio scene.

Along the development of this project the framework improved significantly, specially:

- Computational cost reduction: using the MEL spectrum time-frequency representation, the framework goes 10-15 times faster.
- An important bug was fixed: NaN's tract. That was the main reason why the framework didn't performed as expected during the first months of the project.

The accumulated knowledge is important. The explored ideas (like cross-talk modelling for thresholding or the knowledge obtained around the behaviour of the patterns) could be interesting for the future research steps.

In addition, we noticed that in more relaxed environments (where the targets are not highly overlapped in frequency) works even better; as we can observe in the results of the 3DTVS project [5], the framework performs with a F-measure score up to 0'79.

Appendix A

Project Development

In order to follow coherently the development of the project, the tracking of the work packages is detailed in next sections. We could consider this Appendix as a daily tracking of the work done, where we can observe the changes of the framework and the different approaches attempted. This will help the reader to understand the difficulties encountered and the reason why the current approaches are proposed.

A.1 WP.3 and WP.4.T1

Those work packages correspond to the step for adapting IRCAM's 3DTVS code for Automatic Drums Transcription. Basically, is the first contact with the code. One of the main goals is to understand how it works and to be able to modify it without problems in future steps.

A.1.1 Initial approach

The initial approach implemented in the IRCAM's 3DTVS project is presented in the following paper [5]. The code was conformed by 3 steps (training patterns, training thresholds and detection):

A.1.1.1 Training patterns

The aim is to generate the smallest target dictionary that allow sparse representation.

Two steps are used to compute the training dictionary of patterns for each target:

1. A new spectrogram matrix (**Vv**) is constructed according to a given time-pattern grid, **Vv** is a chain of isolated drum target events. As a result of that we know exactly where the target sounds are presents in **Vv**, in that way we can deduce a priori where the activations are in **H**. Computing the NMD with the previously described **Vv** with a known **H** will allow us to get a first representation of our patterns.

2. The computed **H** and **W** are modified in order to impose sparseness and to find the optimal number of trained patterns: only highest activations and the associated patterns are saved for each considered time point (the more important ones). And other places are set to zero (here we reinforce sparseness). After this process a second NMD is computed. As a result of the last NMD, we obtain the trained patterns with sparsity constraints. The resulting target dictionary contains then K_{tar} patterns that allow a good and maximally sparse representation of the target event sound training database. $K_{tarInitial}$ has to be set in advance, but only the necessary number of patterns will finally be active such that the value K_{tar} is determined adaptive as a function of the target event training sound database.

This procedure is repeated for each target drum instrument.

Notice that no sparseness constraints are applied in the equations of the NMD model. In this framework we can consider that we are imposing a semantic sparseness instead of the common mathematic sparseness criteria.

A.1.1.2 Training Adaptive Thresholds

All the decisions are taken from the **H** matrix. As result of the training patterns step, it could be that for the same target exists more than one pattern to represent its space. Due to this fact, along this project we use the sum of **H** over the K dimension to take decisions.

1) Defining the thresholds:

A double threshold is implemented for taking decisions. As described in the Figure A.1, the first threshold is used for activation meanwhile the second threshold is used for

desactivation. We consider an activation when the signal rises the first threshold till the signal is below the second threshold. Inside the activation zone, no more detections are allowed.

This double threshold is really useful to avoid false positives due to fast oscillations in noisy signals.

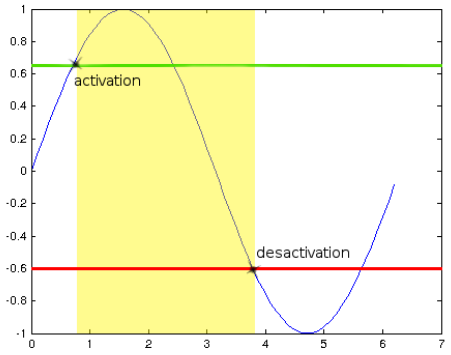

FIGURE A.1: In green we see the activation threshold, in red the desactivation threshold and in yellow the activation zone.

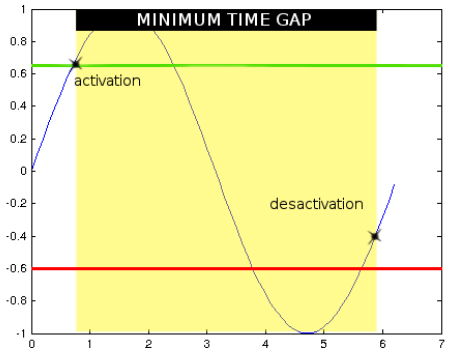

FIGURE A.2: In green we see the activation threshold, in red the desactivation threshold (that in this case is not useful), the minimum time gap allowed and in yellow the activation zone resulting to apply this time gap.

In addition to the desactivation threshold, a second constraint is introduced: a minimum time gap is imposed so that any detection can be triggered before the time gap is finished. If we compare Figure A.2 with Figure A.1 we can observe clearly the effect of this time gap.

Along the different experiments that use the double threshold system the time gap is set to 1 frame, which is 23'2 ms.

To make the thresholds adaptive, two multiplying factors are applied to the average of the activations: one for the activation threshold and another (different) for the desactivation threshold. In that way, our thresholds are adaptive depending on the global energy of the activations.

So, two multiplicative factors (one for activation and another for desactivation) related to the energy of the activations are learned for each drum-instrument.

2) How do we train the thresholds?

The initial approach implemented for training the adaptive thresholds in the IRCAM's 3DTVS original code was doing a brute force optimization along a training data base conformed by 1 film. That means designing a grid of possible thresholds, test all the combinations and keep the ones that gives better results.

Two weak points where detected:

1. If we learn only with one mix, we risk to get a not representative multiplicative factors; maybe they are only useful for this particular piece. To improve this, we decided to train along different mixes in order to get a more representative thresholds that model different styles or techniques.

2. The "force brute" optimization because a more accurate optimization could be done.

A.1.1.3 Test/Detection

The original IRCAM algorithm detects only one target for mix. The drum sets have more than one target: hi-hat, low tom, snare drum, bass drum, splash, and so on. Our algorithm has to be able to detect all the drum instruments simultaneously and to give all the results together: a global F-measure is going to be implemented.

Other approaches for computing F-measure (instead of computing a global F-measure) are presented in the literature. For example in MIREX05, the F-measure is defined as the average of the separated F-measure for kick, snare drum and hi-hat. They consider the toms as a kick, the cymbals as hi-hats and the cowbell as snare drum (Figure A.3 illustrates this hierarchical classification).

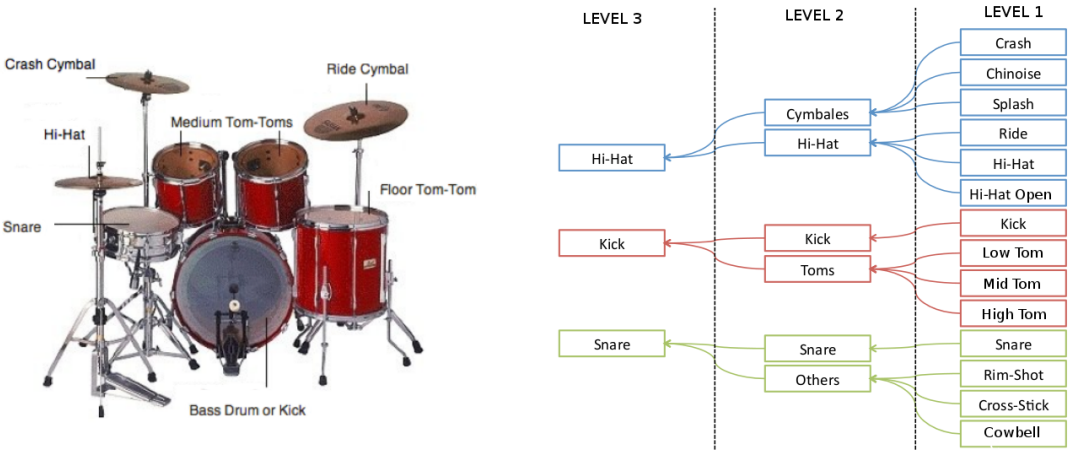

FIGURE A.3: Graphical definition of the hierarchical classification.

Another important difference between our framework and the MIREX05 one (and the state of the art in general), is that we try to detect all the elements of the drum-set separately: we work on Level 1 and they work on Level 3.

To compare our results with the state of the art, hierarchical constraints are programmed with three levels: Level 1 is without constraints and Level 3 is grouping all the activations of the same family (see Figure A.3). Of course, we expect to get better results after grouping because with this strategy we avoid confusions due to crass-talk influence between similar elements of the drum set.

A.1.2 Checking Initial approach

After the implementation of the previous described ideas, the first tests where carried out. Some important issues where found:

1. **Strange artefacts detected in activations**. The original IRCAM's 3DTVS includes an "online-approach" that allows sound tracking between audio channels. This online-approach is based on an analysis window without overlapping; the detected artefacts where due to the analysis window edges. To solve that issue, the analysis window was removed and we analysed all the mix at once. No channel tracking is needed because we work on monaural mixes.

2. **Confusions**: Is easy to get confusions between drum-elements because their basis are highly correlated. We assume this issue as normal. This will be one of the goals, one of the behaviours to model.

Next's sections describe the applied ameliorations in relation to the found deficiencies.

A.1.3 Improving the training thresholds step

Two approaches are proposed for training the thresholds:

Chaining Consist on concatenating several training audio files (and their annotations) into a training mix which represents different kids of styles, techniques and music. The algorithm finds the best thresholds in terms of F-measure for each drum-instrument along this representative training mix.

Averaging Consist on finding the optimal thresholds for each of the several training audio mixes that represents different kids of styles, techniques and music. The algorithm finds the best thresholds in terms of F-measure for each drum-instrument along each training clip. A mean is computed along all the obtained thresholds of the same instrument, which will become the trained threshold for that instrument.

With the framework described in previous chapters, we tested both approaches for training thresholds; similar results where obtained: around 50% in terms of F-measure. As the results where similar we choose the chaining approach because it seems more reasonable for obtaining more representative thresholds.

As we point previously, one of the weak points of the original IRCAM 3DTVS was the optimization for training the thresholds. In the following paragraph we introduce a more accurate optimization:

Multi-step optimization An optimization with more resolution is implemented for training the activation threshold (which is the more sensitive). For the desactivation threshold we keep using the force brute optimization, in that way the algorithm is less expensive in terms of computational cost.

The multi-step optimization for the activation threshold starts with a grid of possible thresholds and selects the one that fits better. Around that first candidate, a second optimization with more resolution is done (see Figure A.4). We repeat this operation till we obtain F-measure 1 or a predetermined number of iterations is raised.

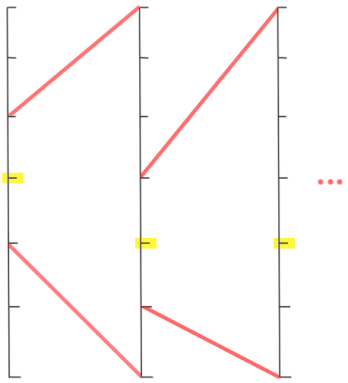

FIGURE A.4: Graphical definition of the multi-step optimization. In yellow, the threshold that fits better on the proposed grid. After each iteration, around the best candidate, a more resolution grid is applied.

A.1.4 Improving Decomposition

Our engineering problem is not only about source separation. The transcription implies to extract information from the activation matrix. For having reliable information on the activations, the patterns don't have to contain information about energy: they have to be transparent to energy. To achieve it, for each iteration of the NMD factorization each pattern is normalized to one ($l_1 - norm$) to obtain energy one.

$$\sum_{k=1}^{K} \sum_{l=1}^{lpattern} \mathbf{W}_k = 1$$

Notice that summing over all the samples of the power spectrogram (which is in fact the squared samples of the magnitude spectrogram) corresponds to the definition for computing the energy of a discrete signal.

In that way all the energy information is in **H**, which is the matrix we are going to use to find the events.

This modification of the NMD is particularly important in order to get meaningful activations.

A.1.5 Results

With the previous described framework:

	Kbgnd	Hierarchical level	F-measure
Drums mixes	0	1	0.4467
Polyphonic mixes	20	1	0.2434

The patterns are trained with Db1, the thresholds are trained with Db4, drums mixes evaluated along the Db6 and polyphonic mixes evaluated along the Db7, described at the Appendix B.

Those bad results are because the framework detects lots of false positives due to cross-talk influence. If hierarchical restrictions where applied, better results could be achieved. But our first goal is to try to arrive as far as we can without hierarchical restrictions.

Another weak point is the long computational time that is required. For example, for computing the Db7 lasts a week.

A.2 WP.4.T3

A.2.1 Improving thresholding system

One common problem is to have cross-talk confusions due to similar targets. Our goal in this work package is to try to avoid false activations as result of cross-talk influence.

The idea is to avoid the small activations of a target that occur at the same time of an strong activation of another similar target. We assume that strong activations are the ones that gives the real information of the event that is going on, meanwhile the smallest ones are modelling the residual.

In conclusion: if exist a harder activation of a pattern that enough explain the event, we should keep the thresholds of the other patterns high in order not to detect small activations because they would be an interference of the hard activation target.

A.2.1.1 Avoiding cross-talk influence: underlying idea for the algorithm

To model mathematically the previously described behaviour, a threshold that depends on the background is designed. This threshold considers the background activations

depending on its similarity to the target. As much similar it is, much important is for our thresholding system; as less similar is, less important.

So, first we need to define a distance to describe similarity. Different distances where considered: Euclidian, IS divergence, Cross-correlation and KL divergence; with no evidences that one would work better than others. After some experiments, the more used ones where: cross-correlation, Euclidian distance and IS divergence. Two first ones for easy comprehension, and the last one for coherence with the decomposition.

The similarity is computed between patterns and it could be described as a function (which is a result of shifting one of the patterns: $S(T, P_N)$) or as the maximum value of the function ($max[S(T, P_N)]$).

Algorithm for similarity functions

$$\sum_N \vec{H}_{bgndN} * S(T, P_N) \equiv \vec{thresholdFunction}$$

Algorithm for similarity values

$$\sum_N \vec{H}_{bgndN} \cdot max[S(T, P_N]) \equiv \vec{thresholdValue}$$

Where $\vec{H}_{bgnd1}$ corresponds to the activations corresponding of the first pattern of the background and $S(T, P_1)$ is similarity function between the target and the first pattern of the background. The resulting threshold (see Figure A.5) models the cross-talk influence of the background.

The value of N is an interesting point to discuss. Along the experiments that concern this chapter, N was set considering only the trained patterns (the ones that are learned that conform the target drum-set). But, in fact, that's not true because if we have a strong activation in our background due to a event that is similar to our trained patterns, we don't want to detect it.

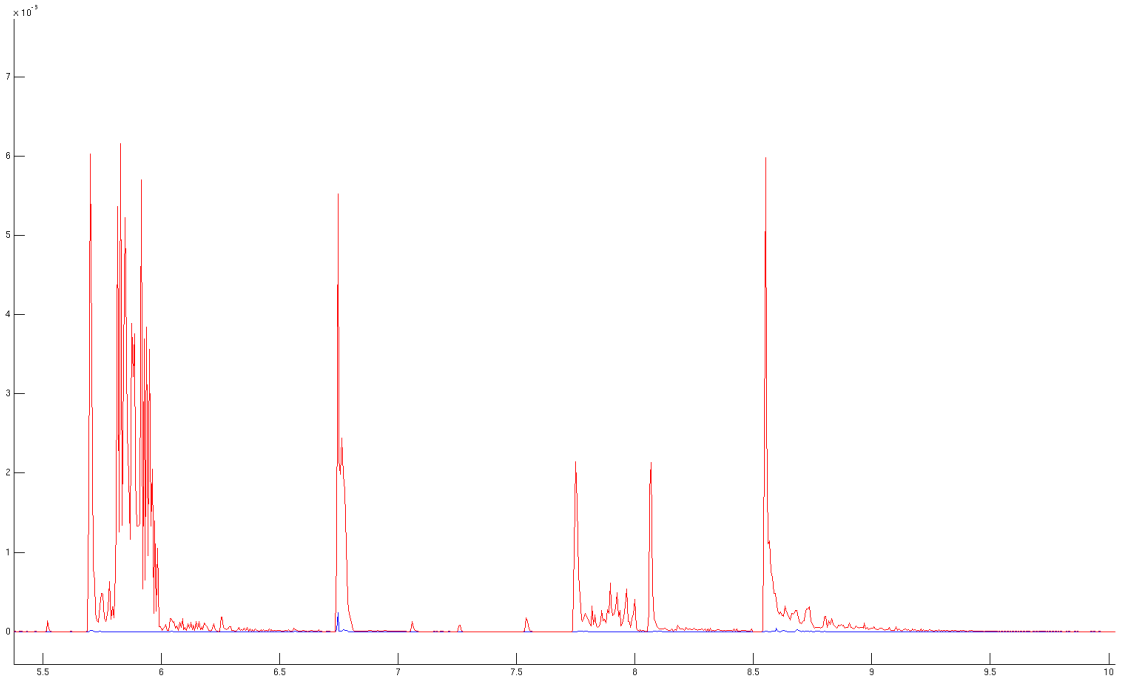

FIGURE A.5: Threshold modelling the cross talk influence. The threshold is in red and the activations are in blue. Notice that the threshold goes high to avoid secondary activations.

A.2.1.2 Resulting algorithms

The new approach relies on modifying the mean based threshold to:

$$\vec{trainingTh} \equiv max[\vec{thresholdValue}, mean(\vec{H}_{target})]$$

Notice that with the $\vec{thresholdValue}$ we are modelling the cross-talk influence of the background and with the $mean(\vec{H}_{target})$ we are considering the importance of the activations of the target. The max operation is for all the values of the $\vec{thresholdValue}$, a point by point operation.

For training the multiplicative factors the previous threshold is used. For detection stage, the threshold is based on the same principal but a pre-processing is introduced:

$$[\vec{trainingTh} \oplus ones(1, 100)] - \vec{H}_{target}$$

Where $\oplus$ is the dilation morphological operation defined as:

$$x[n] \oplus b[n-k] = \vee_{k=-\infty}^{\infty}(x[n] + b[n-k])$$

Where $\vee$ denotes the supremum.

The idea here is to keep the threshold high (with the dilation filter) to avoid false positives and then (with the applied subtraction) the threshold goes to catch the more important activations(see Figure A.6). In that way, the high activations would be raised by the threshold meanwhile the others not.

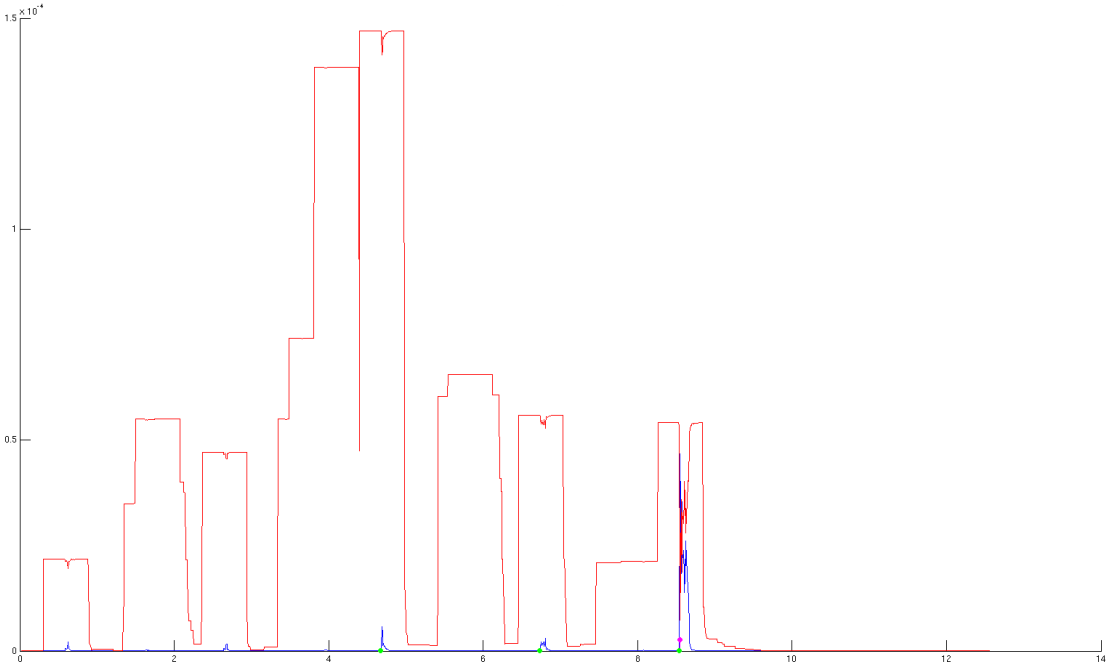

FIGURE A.6: Filtered filter example. The threshold is in red and the activations are in blue. Notice how the subtraction goes to catch efficiently the more prominent activations.

A.2.2 Results

Applying the previous procedure, the following results are achieved:

	$Kbgnd$	Hierarchical level	F-measure
Drums mixes	0	1	0.5815
Polyphonic mixes	5	1	0.4606
Polyphonic mixes	52	1	0.2494

The patterns are trained with Db1, the thresholds are trained with Db4, drums mixes evaluated along the Db6 and polyphonic mixes evaluated along the Db7, described at the Appendix B.

Where the trained patterns are normalized with energy one and the similarity measure is the Euclidian distance.

In the previous table we can observe that a properly determination of the $Kbgnd$ is critical for achieving good results.

The framework is working good in a only drums mixes scenario (but this is not our goal). Notice that if hierarchical restrictions apply, better results could be achieved.

A.3 WP.4.T4

Along this work package two approaches are presented: predicting cross-activations for thresholding and background energy contours for thresholding.

A.3.1 Predicting cross-activations for thresholding

This approach follows the idea that if we are able to predict the worst case of false positives, all the activations that are over this prediction are true positives. Somehow, this new approach pretends to find "sure" detections considering only parameters that depend on the mix to evaluate. The goal, so, is to avoid the training thresholds step where we learn global parameters that could not be true along the pieces to evaluate.

First, we are going to check a simple case in order to understand if this could be useful: 2 patterns (1 target + 1 background) and their corresponding activations.

A.3.1.1 Nomenclature

B_1 - Target pattern

A_1 - Target activations

B_2 - Background pattern

A_2 - Background activations

$S_1 = B_1 * A_1$ - Target signal

$S_2 = B_2 * A_2$ - Background signal

A.3.1.2 Question

- What is A_2 for target signal explained by A_1 with known B_1 and B_2?

We can estimate A_2 ($\hat{A}_2$) solving:

$$\sum_n (S_1(n) - S_2(n))^2 = \sum_n \left(A_1(n) * B_1(n) - A_2(n) * B_2(n)\right)^2 = Min$$

$$\frac{\partial}{\partial A_2(m)} \sum_n (A_1(n) * B_1(n) - A_2(n) * B_2(n))^2 = 0$$

Resolution:

$$\sum_n 2 \left(\sum_k A_1(k) B_1(n-k) - \sum_j \hat{A}_2(j) B_2(n-j) \right) B_2(n-m) = 0$$

$$\sum_k A_1(k) \sum_n B_1(n-k) B_2(n-m) = \sum_j \hat{A}_2(j) \sum_n B_2(n-j) B_2(n-m)$$

From one side:

$$\sum_k A_1(k) \sum_n B_1(n-k) B_2(n-m) = \sum_k A_1(k) \sum_{n'} B_1(n'+m-k) B_2(n') =$$

$$= \sum_k A_1(k) R_{B1,B2}(m-k) = A_1(m) * R_{B1,B2}(m)$$

From the other:

$$\sum_j \hat{A}_2(j) \sum_n B_2(n-j) B_2(n-m) = \sum_j \hat{A}_2(j) \sum_{n'} B_2(n') B_2(n'+j-m) =$$

$$= \sum_j \hat{A}_2(j) R_{B2}(j-m) = \hat{A}_2(m) * R_{B2}(m)$$

Then:

$$A_1(m) * R_{B1,B2}(m) = \hat{A}_2(m) * R_{B2}(m)$$

For $\hat{A}_2$ we need to apply the inverse filter of $R_{B2}(m)$ to $A_1(m) * R_{B1,B2}(m)$.

$\hat{A}_2$ are the activations that explain better our target without the target pattern. In that way, we get the worst case of "false activations" in the background. So, all target energy not covered by B_1 will lead to $\hat{A}_2 > A_2$.

With the previous formulation, we confirm the intuitive idea that confusions and false activations depend on the level of correlation between patterns assumed previously.

A.3.1.3 Thresholding

- When is useful $\hat{A}_2$ for thresholding?

When the local energy of S_2 is similar to the local energy of the $\hat{S}_2$ means that the activations in the background could be "false activations" that explains the target.

If the local energy of $\hat{S}_2$ [*E($\hat{S}_2$)*] is not similar to the local energy of S_2 [*E(S_2)*], two scenarios:

1. $E(S_2) \gg E(\hat{S}_2)$: we can ensure that exists a background event.
2. $E(S_2) < E(\hat{S}_2)$ then the activations could be: A) "False activations" that are modelling part of the target. B) True background activations. C) Both A and B at the same time. So, this approach does not help.

The previous idea can be moved to the relation in-between A_2 and $\hat{A}_2$:

1. $\hat{A}_2 \ll A_2$: we can ensure that exists a background event.
2. $\hat{A}_2 > A_2$ then the activations could be: A) "False activations" that are modelling part of the target. B) True background activations. C) Both A and B at the same time. So, this approach does not help.

A.3.2 General case: N patterns that models the target and M patterns that models the background

The previous study case is not the real case we are going to work with. Our problem is the one described in the following section.

A.3.2.1 Nomenclature

B_{T1} - Target pattern number 1

A_{T1} - Activations for pattern number 1 of the target

B_{B2} - Background pattern number 2

A_{B2} - Activations for pattern number 2 of the background

$S_T = \sum_{n=1}^{N} A_{Tn} * B_{Tn}$ - Target signal

$S_B = \sum_{m=1}^{M} A_{Bm} * B_{Bm}$ - Background signal

A.3.2.2 Question

- *What is* $A_{B1}, A_{B2}, \ ... \ , A_{BM}$ *for target signal explained by* $A_{T1}, A_{T2}, \ ... \ , A_{TN}$ *with known patterns?*

Then, for estimating $A_{B1}, A_{B2}, \ ... \ , A_{BM}$ we need to solve:

$$\sum_{n}(S_T(n) - S_B(n))^2 =$$

$$= \sum_{n}((A_{T1}(n) * B_{T1}(n) + A_{T2}(n) * B_{T2}(n) + ... + A_{TN}(n) * B_{TN}(n)) -$$

$$-(A_{B1}(n) * B_{B1}(n) + A_{B2}(n) * B_{B2}(n) + ... + A_{BM}(n) * B_{BM}(n)))^2 = Min$$

In this general case, the maths become not evident and the problem becomes more difficult to solve.

In addition to that, we should notice that the convolution inbetween matrices is not the operator $*$. We should substitute the $*$ for the operator described by Smaragdis at the NMD presentation paper [2].

For his complexity and because it doesn't exist any strong evidence that points to this approach as useful, we decided to move to other approaches.

A.3.3 Background energy contours

A.3.3.1 Motivation

If the energy of the target is over the total energy contribution of the background, we can ensure that is part of the target. This idea could be considered as a masking-model approach: all the activations that are not "masked" are true detections.

In order to detect the masked ones, we train the f_t multiplicative factors. The principal of using f_t is to "make appear" the ones that are masked.

A.3.3.2 Method for computing the Energy contours

1. Doing $\vec{H}_p(n) * \mathbf{B}_p(k,n)$ for each pattern: $\mathbf{S}_p(k,n)$. Where $\vec{H}_p(n)$ is the activation per pattern and $\mathbf{B}_p(k,n)$ is the corresponding pattern.
2. For each target, compute the energy contour of the background which includes fixed and adaptive patterns: $\vec{E}_b(n) = \sum_k \sum_{p \notin target} \mathbf{S}_p(k,n)$.
3. For each target, compute the energy contour of the target: $\vec{E}_t(n) = \sum_k \sum_{p \in target} \mathbf{S}_p(k,n)$.
4. We define the "candidate areas" where $\vec{E}_t(n) > f_t \cdot \vec{E}_b(n)$. Where f_t is a multiplicative factor associated to a target.
5. For each "candidate area" we take the local max as detected event. We are assuming that only one drum event can be detected for each "candidate area".

The f_t are going to be trained taking in consideration the criterias of the previous method. One for target.

Notice that the double threshold is not used from now on.

A.3.3.3 Results

	Kbgnd	Hierarchical level	F-measure	Recall
Drums mixes	5	1	0.4712	0.6341
Polyphonic mixes	5	1	0.3863	0.6474

The patterns are trained with Db1, the thresholds are trained with Db3, drums mixes evaluated along the Db6 and polyphonic mixes evaluated along the Db7, described at the Appendix B.

Where the trained basis are normalized with energy one.

A high recall is obtained, due to:

- The f_t in this scenario is really sensitive, and is not working as we expect. The trained f_t puts our threshold too low and lots of false positives are detected.
- There is a lot of cross-talk influence not modelled by the thresholding system.

In a a high recall scenario, a results refinement post-processing step could be useful.

Few experiments where carried out unusing f_t without success.

A.4 WP.4.T5

A.4.1 Detecting/decomposing only interest zones: onsets zones

A large amount of processing time is required using NMD with a high number of patterns. For going faster, we decided only to analyse the interest zones: the onsets zones. In that way we are more selective and we compute NMD only when is required.

From the point of view of velocity, we also observed that the algorithm process faster a signal if we compute it by segments instead of processing it at once. In that way, processing the onset zones separately, we expect to go faster.

In addition, processing only the onsets zones we can ensure that outside of those zones we will never detect.

A.4.2 Implemented Algorithm

1. **Load audio file**:
   (a) Down-mix from stereo to mono.
   (b) Normalize by the energy: $[\sum \vec{signal}^2]/length(\vec{signal}) = 1$

2. **Find onsets zones** using a onset detection by means of transient peak classification [20]. The interest zones are defined from the beginning of the transient till the end of the transient plus half of the analysis window (zone where the onset is supposed to be according to the algorithm defined at [20]). To fit this zone in a NMD context, we add *lpattern-1* frames at the end in order to model the tail of the onset (tail zone).

3. **For each onset zone compute**:

   (a) STFT ($\mathbf{V}$).

   (b) Energy of the segment $Eseg = (\sum\sum \mathbf{V}^2)/length(\mathbf{V})$.

   (c) $\mathbf{H}_{ini}$ set as the energy contour of the spectrogram ($sum(\mathbf{V}, 1)$). It seems the better configuration that allows us to start with a slow cost function value.

   (d) $\mathbf{H}_{mask}$. A mask for $\mathbf{H}$ is defined in order to discard the tail zone activations where is not expected to be the onset. This step is to avoid false positives due to overlapping between onsets.

   (e) Computing NMD and applying $\mathbf{H}_{mask}$ to the computed $\mathbf{H}$ to take only into account the interest zones.

   In order to be able to take decisions having all the information of the interest zones a chain is formed.

4. **Taking decisions** along the previous formed chain.

   (a) Defining threshold: $Eseg\vec{C}hain + mean(Eseg\vec{C}hain)$

   (b) For each drum instrument:

      i. Obtain threshold for each drum instrument: multiply the threshold by the previous learned multiplicative factor that correspond to the drum instrument to analyse.

      ii. Obtain $\vec{H}$ corresponding to each drum instrument: sum down the $\mathbf{H}$ components corresponding to the drum instrument to analyse.

      iii. If the $\vec{H}$ of the instrument is over its threshold inside an interest zone, we detect our target in the frame that $\vec{H}$ is max along the interest zone. The interest zone is defined by $\mathbf{H}_{mask}$.

For computing the "mean of the energy along all the onset zones" used for designing the threshold, our script first needs to know all the interest zones before. Is why a chain of **H** (with the mask applied) and *Eseg* is implemented, for testing this kind of configurations that are helpful to avoid false positives.

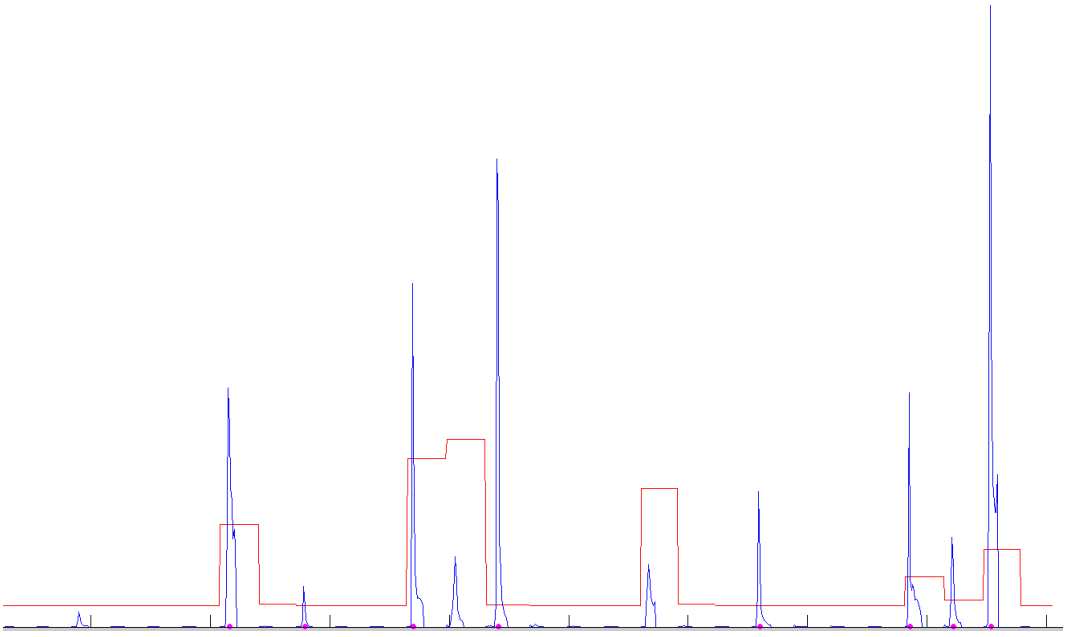

FIGURE A.7: Chained $\vec{H}$ for a specific target (blue line), its threshold (red line) and detections (magenta points).

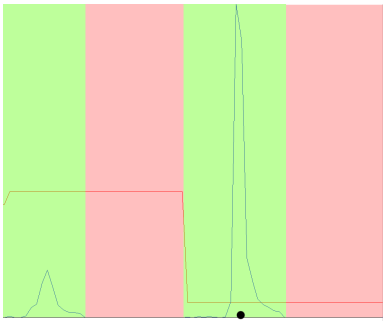

FIGURE A.8: A zoomed example of the detection step: chained $\vec{H}$ for a specific target (blue line), its threshold (red line) and detections (black point). The interest zone is in green and the tail zone is in red.

In the Figure A.7 we can see the chained $\vec{H}$ of a drum instrument along different onsets zones and its associated threshold. We can differentiate clearly each onset zone for the mask that is applied: observe that there is one part with values different to zero (interest zone where we expect the drum) and another part which is set to zero that correspond to the tail zone where we don't expect a drum. In Figure A.8 we can observe a zoomed

example where where we can see more easily the tail zone (in red) and the interest zone (in green).

Also we can observe the threshold (the line in red) that corresponds to:

$$Eseg\vec{C}hain + mean(Eseg\vec{C}hain)$$

In both Figures (A.7 and A.8) we can see the utility of adding this offset at the threshold: the lowest activations that are related to low energy onset zones will lead us to false positives if no offset was included.

A.4.3 Results I

A.4.3.1 Relevant conditions for this experiment

The trained patterns are normalized with energy one and the NMD algorithm is not allowed to do more than 4 iterations. Several experiments of our team conclude that after a few amount of iterations (*niter*=4-5), the activation matrix is enough meaningful to detect what is going on in the audio mix.

A.4.3.2 Testing

After training the framework, the following results are obtained:

	Precision	Recall	F-measure
Polyphonic mix	0.106	0.589	0.180
Drums mix	0.293	0.76	0.4234

The patterns are trained with Db1, the thresholds are trained with Db4, described at the Appendix B.

The previous results correspond to the mixes that gave us better results. In polyphonic: RMG005 and in drums: 67AD3. Evaluated without hierarchical constraints.

A.4.4 Conclusions and observations

1. **Goes much faster**: before introducing this new approach, the script lasts 2 weeks for decomposing the data-sets. With this new approach, with 2-3 days of processing is done. This big improvement is given because we process less information, with isolated small cuts and we allow the NMD algorithm to do only 4 iterations. Even the improvement, efforts should be done in order to reduce the processing time because maybe 4 iterations is not enough.

2. The unique problem with this approach could exist when there is **overlapping between onset zones** (the detected onsets are closed), this will lead us to process more data instead of less.

3. **Lots of false positives**: as we can observe in Figure A.9 the trained threshold don't work as we expected, we should consider a modification our thresholding system.

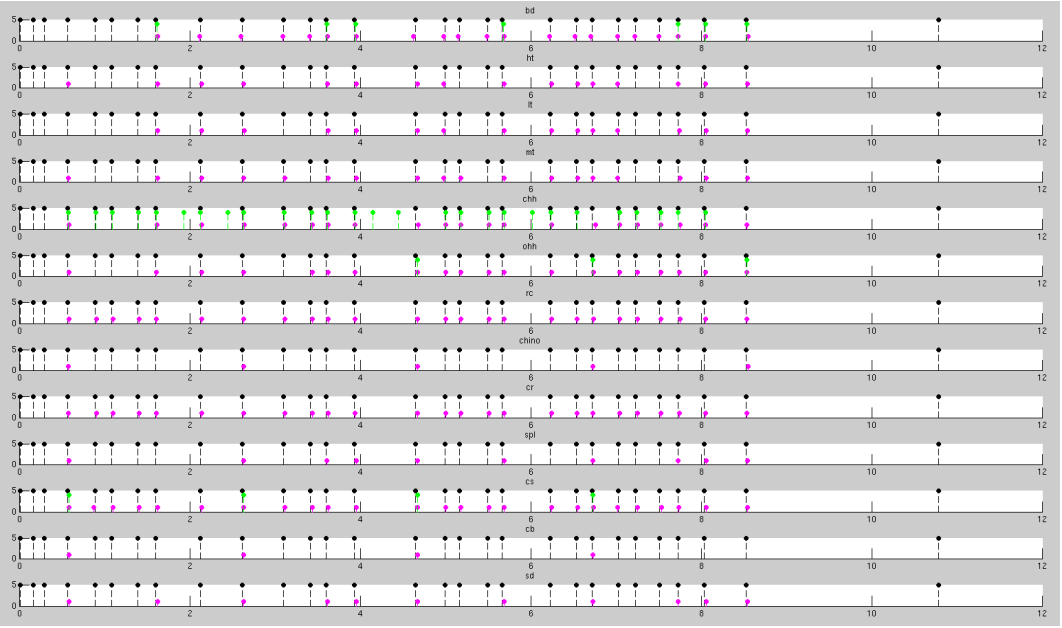

FIGURE A.9: In black we find the positions found for the onset detection algorithm, in green we see the ground truth and in magenta the detections of our framework. Each row represents an element of the drum-kit: bd (bass drum), ht (high tom) and so on. As we can see, our trained threshold system is not discriminative: lots of false positives are detected.

4. **The background patterns:** from one side, usually most of them are similar (check Figure A.10). That means that most of the background patterns we process are not useful to explain the background. From the other side, if the number of

background patterns (K_{bgnd}) is overestimated, the targets are modelled by the background patterns. Then, most of the energy of the activations that should go to the target patterns go to the background patterns (what will be a big issue because our thresholding system will consider the targets as background).

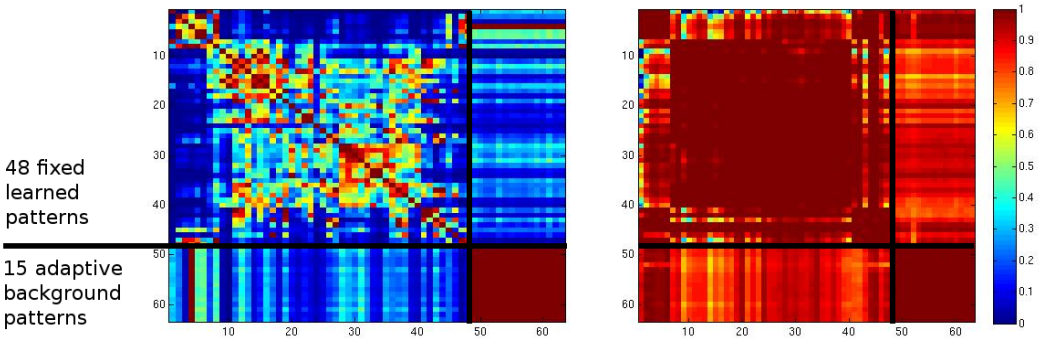

FIGURE A.10: Similarity matrices: cross-correlation measure (left) and IS divergence (right). They don't correspond to the same audio file. We can observe that the adaptive background patterns are similar. And, also, that the adaptive background patterns take information of the fixed trained ones.

5. **Cross-talk influence:** the event we are detecting is enough explained by another activation. The secondary activations should not be taken into account; when is this case, keep the threshold high. In fact, is to apply the cross-talk approach described previously.

6. **No energy conservation:** the energy of the approximated spectrogram is different than the activations used to estimate this spectrogram. That means that there is something wrong inside the NMD functions.

A.4.5 Improving previous considerations

1. **Ideas for avoiding false positives**:

   (a) Improve thresholding. Incorporating at the threshold a parameter that indicates if an event is better represented by another activation: cross-talk modelling approach.

   (b) Post processing step: SVM classification. In our approach we can observe a high recall due to false positives. Introducing this post-processing step, our goal would be to get this Recall as F-measure.

(c) Predicting the needed number of adaptive patterns for describing the background (K_{bgnd}).

(d) Check if the patterns represent properly the target, if not redesign the training patterns step.

2. **Increment niter:** increment the number of iterations allowed to the NMD, maybe 4 is not enough. After a few experiments, we observed that in our scenario (drums transcription processing only onsets in a polyphonic context) the algorithm usually converges around the 15th iteration. In terms of processing time, changing the number of iterations from 4 to 15 only increases 3 seconds per onset.

3. **Improve processing time:** Other open-source NMD algorithms where tested in the same conditions and we observed that our algorithm was much more slower. This is because our algorithm also factorizes a matrix that contains information about channels (because it's based on a 7.1 channels scenario). In our case, we work on monaural signals and is not necessary to compute this matrix. Without updating it, our algorithm becomes 20 times faster.

A.4.6 Results II

Considering the previous conclusions, a second round of experiments under the same context where processed.

A.4.6.1 Relevant conditions for this experiment

1. Number of NMD iterations allowed set to 15.

2. Number of adaptive background patterns set as 20.

3. Incorporating cross-talk influence of other activations with cross-correlation between patterns as similarity measure. That means changing the design of the threshold:

$$\vec{BgndInfluence} + mean(\vec{BgndInfluence})$$

4. Without updating the matrix that contains the tracking of the channels ($\mathbf{Q}$) to go faster.

A.4.6.2 Testing

Some preliminary experiments illustrates us that the problem was not that the number of iterations was small, neither that the thresholding model was not the adequate. Right now, the system is still not working: deeper inspections should be done.

A.4.7 Conclusions and observations

1. Even that right now the threshold is not working, the idea of using the sum of the cross-talk influence of the background as threshold seems a better approach instead of using the local energy: we avoid secondary activations that are better explained for stronger ones and, implicitly, is an adaptive approach that depends directly from the energy of the activations.
2. The threshold system is not working. The training thresholds step should be checked, understood and improved.
3. We have problems to control what the patterns model. From one side (as shown previously in Figure A.10) the background patterns model the target. From the other side (although there are background patterns to model non-target events), the target patterns suffer activations due to non-target events. The training patterns step should be checked too.

A.4.8 Next steps to improve

We can sum up that two main problems exist in our framework: our thresholding system is not enough discriminative and our patterns don't represent what we expected.

1. Estimate K.
2. We should ensure that the energy of $\hat{\mathbf{V}}$ is equal to energy of $\mathbf{H}$, in order to have meaningful activations.
3. Incorporating the cross-correlation in-between all the patterns increases our processing time. In case of deciding that this information is useful, efficient techniques for computing the similarity matrix should be implemented.
4. Checking the training (patterns and thresholds) steps.

A.5 WP.4.T6

A.5.1 Improving training patterns step I

As described in previous section, the patterns don't represent what we expected. In order to understand what is going on, further inspections where done in the learning step and along the learned patterns.

A.5.1.1 Motivation

We observed that our trained patterns model "parts of the event" instead of complete events. In the previous learning approach is possible that our target event could be splitted in different parts that summed represent a complete event. Our goal is to avoid this splitting because those "target parts" could be a good candidate to represent "parts of the background" and this will lead us to a non discriminative system.

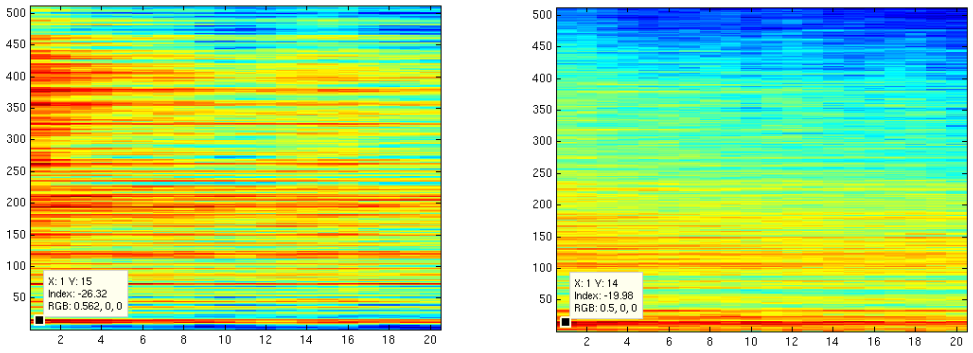

FIGURE A.11: Two selected patterns from the subgroup of K_{learn} patterns that model an open hi-hat. As we can see, the pattern situated at the left (pattern A) models the harmonic part of the cymbal, meanwhile the pattern situated at the right (pattern B) models the low frequencies of the cymbal.

This could be the reason, too, that our system has lot of cross-talk influence.

As we can see in Figure A.11 the trained patterns we are using model "parts of the event" that can easily be activated by a background event.

A.5.1.2 Objective

Implement a training algorithm that leads us to the best possible representation of our training-set with the minimum number of patterns. The resultant patterns should model entire target events and sparse representation.

A.5.1.3 Algorithm

The training-sets are conformed by J audio files with isolated target events. In a drums context, *e.g.*, that would mean J recordings of isolated snare hits. This algorithm should be run for each different target, that means that this algorithm should be run separately for each element of the drum-set: kick, snare, open hi-hat, closed hi-hat, and so on.

For each different target:

1. **Load data**. Cut the J isolated drum events from the point that has maximum energy till we reach the length of the pattern (*lpattern*). We get J training-clips of *1 x lpattern*.
2. **Compute time-frequency representation**, as result J training-clips.
3. **Normalize** each training-clip spectrogram ($l_1 - norm$) to avoid scalar factors that can influence our similarity matrix.
4. **Compute similarity matrix** for all J training-clips. We get a JxJ similarity matrix.
5. **For each *k* from 1 to *kmax***: testing with different number of patterns/classes to get the best configuration. k is the number of patterns/classes allowed in each iteration. The obtained centroid for each class, the *k-mean*, will be the learned pattern.
   (a) Select the k most different clips from the J training-clips (k initial-representants of *k-classes*).
   (b) Find the closest clips of the data-set for each k initial-representant (members of each *k-class*).

(c) For each *k-class* compute the NMD considering the members of the class as input (a chain of them) to factorize with only one adaptive pattern. The resulting pattern $\mathbf{W}$ is the centroid of the *k-class*, which is in fact the *k-mean* learned pattern we are searching. In this step we impose sparseness in the same way as we introduced in [5]: constraining $\mathbf{H_{ini}}$. Imposing a time grid on $\mathbf{H_{ini}}$ where is set to 1 where each member of the class begin along the chain, and to 0 all the others.

(d) Compute the NMD with the $\mathbf{C}_k$ learned patterns along the J chained files and save relevant performance data. In this step is considered as $\mathbf{H_{ini}}$ the energy contour (the sum over bins, which is in fact an approximation of the energy for each frame) of the input spectrogram and fixed $\mathbf{W}$. A post processing of the $\mathbf{H}$ matrix is applied to consider the contribution of the secondary activations as part of the event.

6. Choose minimum number of k's depending on the performance data computed in step 4.d.

Notice that 5.a and 5.b can be computed directly from the similarity matrix:

5.a: Two different scenarios.

(a) First iteration: find the more different. Sum over columns of the similarity matrix and the column that has the maximum value is the one that corresponds to the most different training-clip: the first "initial-representant".

(b) Other iterations: find the more different from the previous "initial-representants". Sum over the columns taking only the rows that correspond to the previous selected "initial-representants". The position of the maximum value over the previous described sum corresponds to the most different training-clip respect to the "initial-representants" already selected.

5.b: Check the smallest distance, using the similarity matrix, along all the training-clips respect to the k initial-representants.

Except for $\beta = 2$ the $\beta - divergence$ is not symmetric and don't act as a distance. In order to compute comfortably the similarity matrix, we will consider the $\beta - divergence$

as a distance:

$$\frac{d_\beta(X,Y) + d_\beta(Y,X)}{2}$$

In step **5.c** we pretend to find the *k-mean*, a centroid representation for all the members of each *k-class*. Which is equivalent to find **P** solving:

$$\sum_{m=1}^{M} d_\beta(\mathbf{P}, \mathbf{X}_m) = Min$$

Where M is the number of members of the class, **P** represents the centroid that is going to be considered as the trained pattern and $\mathbf{X}_m$ is the spectrogram of each training clip class member. Solving this problem, considering as distance the $\beta - divergence$, is the same as running the NMD with an input that contains all the mixes and leading it to update with only one adaptive pattern to solve it.

To give all the samples to the NMD as input, the sound-samples are concatenated in a single-channel audio file.

The post processing of **H** used in 5.d is to consider the contribution of the secondary activations as part of the event and is implemented using the convolution of $\vec{H}_{target}$ with [1,1]. This approach follows the idea that the activation after an onset contributes to explain the same onset.

To sum up, the previous procedure is an unsupervised β-divergence *k-means* clustering.

A.5.1.4 Discussion: choosing *k*

Notice that:

- As result of the chaining, we know where the onsets are situated along **H**: exists a controlled grid of onsets.
- As result of the normalization and the chaining we know that (if a perfect representation is achieved) the sum of activations would be one on the grid of onsets and zero to others.

As described in the previous algorithm, some performance data is used to select k:

- *Final cost value*: as result of the NMD in 5.d a cost value is obtained.
- *min(inGrid)*: as is known that the best scenario achieves activations to one on the grid of onsets, an interesting parameter could be the worst activation in the grid of onsets: *e.g.*, accepting a determined number of k if $min(inGrid) > 0.45$.
- *min(inGrid)/max(outGrid)*: to avoid false positives due to bad representations, an interesting parameter could be the relation with the *min(inGrid)* with the *max(outGrid)*. In fact, the *max(outGrid)* is the more prominent false activation due to bad representation of the patterns: *e.g.*, accepting a determined number of k if $min(inGrid)/max(outGrid) > 2$.

The two last options seems the more interesting ones because setting a criteria around the **H** matrix seems more reasonable due to the fact that we are going to use **H** to take decisions.

A.5.1.5 Observations

As result of the described training patterns step it could be that for the same target exists more than one pattern (k) to represent its target space. Due to this fact, along this project we use the sum of **H** over the K dimensions related to the interest target to take decisions.

Notice that no sparseness constraints are applied in the equations of the NMD model. In this framework we can consider that we are imposing a semantic sparseness instead of the common mathematic sparseness criteria along the cost function.

A.5.1.6 Results: improving decompositions

Once the previous algorithm was tested, the following results where obtained:

For $k = J$ (with as patterns as training-clips), we expect to have an **H** matrix with a 1 per row (where the training-clip associated to the row pattern is occurring) and all the other values of **H** to zero.

1's because **H**'s reflects the energy of the input (which was normalized to one), and all zeros because we imposed sparseness.

In addition to that, we expected to have the final cost function value to be zero, because we have as patterns as events.

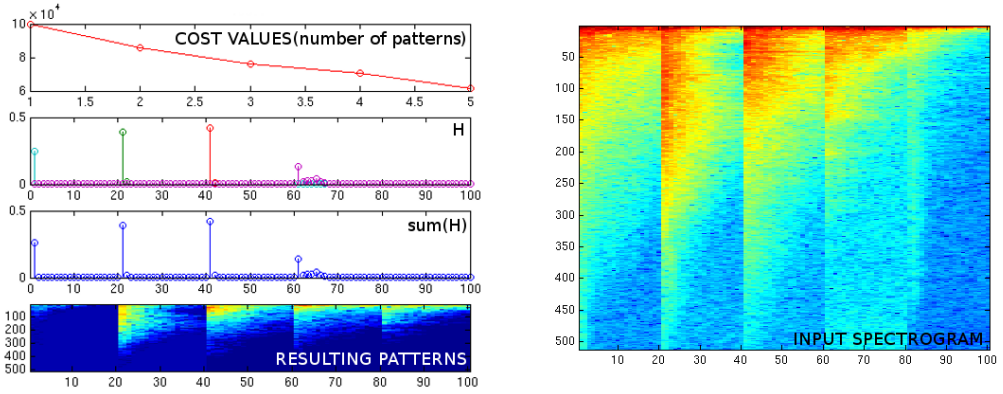

FIGURE A.12: At the right we can see the spectrogram to approximate (N=5). At the left we can see (in descending order): first, the cost values with different k's (from 1 to 5); second, the $\mathbf{H}$ matrix for k=N=5 (with a different colour per row); third, the sum along the rows (which is in fact the information we are going to use for taking decisions) for k=N=5; and forth, the learned basis (k=5).

As we can see in Figure A.12, is not what we expected: neither the final cost value, neither the activations. Something is wrong.

After searching, we found an error inside the NMD function: in each iteration (after updating $\mathbf{W}$), this code line was applied to avoid NaN's:

```
W = W.*(W >= 10^(-10)) + 10^(-10)*(W <= 10^(-10));
```

What means that the values lower than 10^{-10} are set to 10^{-10}, which implies a bad approximation of the low amplitude values. This is exactly what we can see in Figure A.12, where the bad approximation in $\mathbf{H}$ comes from the spectrogram part that corresponds to the onset that has low power in mid-high frequencies.

As we are considering β=0 (which is scale invariant) is a big issue. This is the reason why our algorithm doesn't work as we expected using IS-divergence.

In conclusion: that line of code should be removed as interferes gravely our results. Another solution should be found to be robust against NaN's.

Checking the update rules:

$$\mathbf{W}^t \leftarrow \mathbf{W}^t \circledast \frac{\left(\mathbf{V} \circledast \hat{\mathbf{V}}^{\circledast(\beta-2)}\right) \circ \mathbf{H}^{t\rightharpoonup}}{\hat{\mathbf{V}}^{\circledast(\beta-1)} \circ \mathbf{H}^{t\rightharpoonup}}$$

$$\mathbf{H} \leftarrow \mathbf{H} \circledast \frac{\sum_t \left(\mathbf{V} \circledast \hat{\mathbf{V}}^{\circledast(\beta-2)}\right)^T \circ \mathbf{W}^{t\rightharpoonup}}{\sum_t \left(\hat{\mathbf{V}}^{\circledast(\beta-1)}\right)^T \circ \mathbf{W}^{t\rightharpoonup}}$$

The $\circ$ symbol denotes the outer product, while $\circledast$ is the Hadamard product and powers of matrices indicated with $\circledast(\cdot)$ are element-wise.

We can observe that NaN's can only be introduced by the $\hat{\mathbf{V}}$. Adding *eps* (the smallest value can be used in Matlab) to $\hat{\mathbf{V}}$ each time is computed, should be enough to avoid NaN's. After this correction, as we can check in Figure A.13, the problem was solved and the algorithm is working as expected.

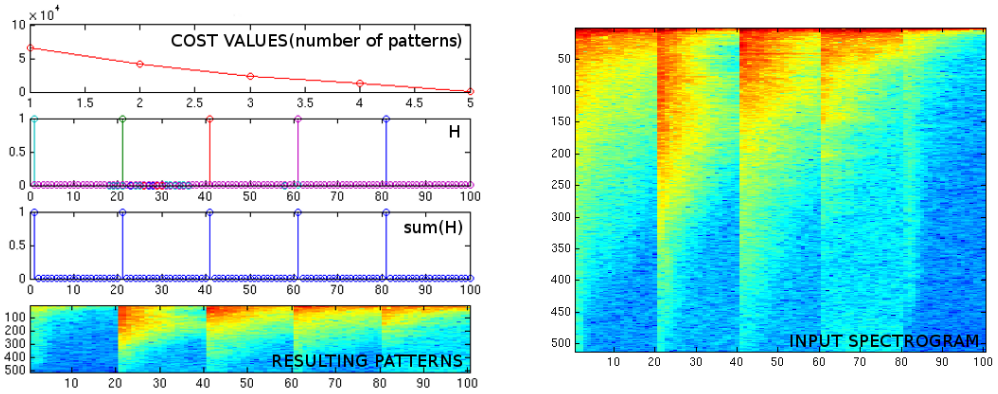

FIGURE A.13: At the right we can see the spectrogram to approximate (N=5). At the left we can see (in descending order): first, the cost values with different k's (from 1 to 5); second, the $\mathbf{H}$ matrix for k=N=5 (with a different colour per row); third, the sum along the rows (which is in fact the information we are going to use for taking decisions) for k=N=5; and forth, the learned basis (k=5).

Notice that:

1. The $\mathbf{H}$'s are exactly as expected.
2. Now exists energy coherence between: $\mathbf{V}$, $\hat{\mathbf{V}}$ and $\mathbf{H}$.
3. The cost function is near zero for k=N, as we expected.
4. Due to the way we train, the activation matrix naturally leads to a sparse representation without sparse constraints.

A.5.1.7 New opportunities: the noise parameter

Meanwhile evaluating possible side effects as result of this new tract to avoid NaN's, we noticed that introducing a bigger value instead of *eps* could help us to control robustness against noise, *e.g.*:

Considering V=[1,0.1] $\hat{V}$*=[0.5,0.05] and* β*=0 (IS divergence):*

$$D_{\beta=0}(V|\hat{V}) = \sum_{\in V} \frac{V}{\hat{V}} - \log \frac{V}{\hat{V}} - 1$$

The following cost value is obtained:

$$D_{\beta=0}(V|\hat{V}) = (\frac{1}{0.5} - \log \frac{1}{0.5} - 1) + (\frac{0.1}{0.05} - \log \frac{0.1}{0.05} - 1) = 0.3068 + 0.3068 = 0.6136$$

But if instead of adding an insignificant value like eps we add a bigger value (from now on we are going to refer to this parameter as noise parameter):

$$D_{\beta=0}(V + 0.2|\hat{V} + 0.2) = \sum_{\in V} \frac{V + 0.2}{\hat{V} + 0.2} - \log \frac{V + 0.2}{\hat{V} + 0.2} - 1 =$$

$$= (\frac{1 + 0.2}{0.5 + 0.2} - \log \frac{1 + 0.2}{0.5 + 0.2} - 1) + (\frac{0.1 + 0.2}{0.05 + 0.2} - \log \frac{0.1 + 0.2}{0.05 + 0.2} - 1) = 0.1752 + 0.0176 = 0.1928$$

Notice that we are adding the *noise parameter (Np)* in numerator and denominator to keep the ratio, which is in fact the principal of the IS divergence.

As you can observe this parameter acts decreasing the impact to the small values (associated to the random noisy parts of the spectrogram) respect to the big values. The cost associated to the big value is 10 times bigger than the cost associated to the small value.

Here we have defined how this *noise parameter* works as a threshold for the lower values that are less considered in terms of cost.

$$D_{\beta=0}(V + Np|\hat{V} + Np) = \sum_{\in V} \frac{V + Np}{\hat{V} + Np} - \log \frac{V + Np}{\hat{V} + Np} - 1$$

Notice that the underlying idea of this model modification relies on adding a constant pattern to the classic NMF model:

$$\mathbf{V} \approx \hat{\mathbf{V}} = \sum_{k=1}^{K} \vec{W}_k^T \vec{H}_k + \vec{W}_0^T \vec{H}_0 = \mathbf{WH}$$

Where K, $\mathbf{W}$ and $\mathbf{H}$ are as usual, $\vec{W}_0$ is the constant *noise parameter* base and $\vec{H}_0$ is the activation vector that allows the base to be activated along all the audio spectrogram.

Assuming the previous described NMF model means that conclusions of Févotte in [18] (described in Section 3.4.1) still applies.

A.5.1.8 Results

MEL spectrogram

For only drums mixes, evaluated along the Db6:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
58min 31seg	0	1	0.1076	0.96	0.1934
58min 25seg	0	3	0.2996	0.9379	0.4531
1h 32min 39seg	1	1	0.3388	0.4272	0.3650
1h 34min 31seg	1	3	0.5555	0.5814	0.5605
2h 45min 7seg	**5**	**1**	**0.3387**	**0.3537**	**0.3310**
1h 45min 26seg	**5**	**3**	**0.5793**	**0.5613**	**0.5636**

For polyphonic mixes, evaluated along the Db7:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
5h 18min 36seg	0	1	0.1318	0.7103	0.2203
5h 18min 6seg	0	3	0.2957	0.7310	0.4175
8h 54min 23seg	1	1	0.2058	0.7200	0.3151
8h 52min 28seg	1	3	0.3053	0.7196	0.4262
9h 28min 8seg	**5**	**1**	**0.2888**	**0.6306**	**0.3853**
9h 26min 59seg	**5**	**3**	**0.3342**	**0.6731**	**0.4431**
8h 32min	20	1	0.0696	0.8191	0.1280
8h 33min 9seg	20	3	0.2173	0.8591	0.3447
10h 5min 27seg	40	1	0.0696	0.8181	0.1280
10h 5min 34seg	40	3	0.2175	0.8596	0.3450

Power spectrogram

For only drums mixes, evaluated along the Db6:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
25h 37min	5	1	0.3662	0.2648	0.2720
25h 35min 51seg	5	3	0.5848	0.5430	0.5488

For polyphonic mixes, evaluated along the Db7:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
130h 32min 5seg	5	1	0.1732	0.6053	0.2672
129h 21min 46seg	5	3	0.2617	0.7871	0.3911

A.5.2 Improving training patterns step II

A.5.2.1 Motivation

Implement a training algorithm that leads us to the minimum number of trained patterns that represents properly our training data-set. The resulting trained patterns should represent itself a complete element of the target class. We don't want to allow the algorithm to split the patterns in elements that com- bined could represent another different class that no longer belongs to the trained class. The goal, so, is to obtain

patterns that if we combine them we still remain in the same class. We can ensure that we still remain in the same class if the resulting patterns constitutes itself a complete element of the target class.

A common k-means clustering strategy that alternates two steps (class assignement and update centroids) is used to cluster our training space.

A.5.2.2 Algorithm

The training-sets are conformed by J audio files with isolated target events. In a drums context, *e.g.*, that would mean J recordings of isolated snare hits. This algorithm should be run for each different target: kick, snare, open hi-hat, closed hi-hat, and so on.

In the following lines the proposed algorithm is outlined and the details are provided afterwords.

For each different target:

1. **Load data**. Cut the J isolated drum events from the point that has maximum energy till we reach the length of the pattern (*lpattern*).
2. **Compute time-frequency representation**.
3. **Normalize** each training-clip spectrogram ($l_1 - norm$) to avoid scalar factors that could influence our similarity matrix.
4. **Compute Np**: set the *Np* in a common global reference point (the max of the training dataset -60db).
5. **For each *k* from 1 to *kmax***: testing with different number (k) of patterns/-classes to get the best configuration. The obtained centroid for each class, the *β-k-mean*, corresponds to the learned pattern.
   (a) Inicialize k centroids.
   (b) Given the initial set of kResults centroids the algorithm alternates between two steps till convergence:
      i. Find membership: using $d_\beta(\mathbf{X}_j|\mathbf{C}_k)$ where $\mathbf{X}_j$ is each training clip and $\mathbf{C}_k$ is each centroid.

ii. Update centroid: for each *k-class* compute the NMD considering the members of the class as input (a chain of them) to factorize with only one adaptive pattern. The resulting pattern **W** is the centroid of the *k-class*, which is in fact the *β-k-mean* learned pattern we are searching. In this step we enforce sparseness in the same way as introduced in [5]: constraining $\mathbf{H_{ini}}$. Imposing a time grid on $\mathbf{H_{ini}}$ where is set to 1 where each member of the class begin and to 0 all the others.

(c) Compute the NMD with the $\mathbf{C}_k$ learned patterns along the J chained files and save relevant performance data. In this step is considered as $\mathbf{H_{ini}}$ the energy contour (the sum over bins, which is in fact an approximation of the energy for each frame) of the input spectrogram and fixed **W**. A post processing of the **H** matrix is applied to consider the contribution of the secondary activations as part of the event.

6. **Choose minimum number of *k*'s** depending on the performance data computed in step 5.c.

The initialization is a sensitive step where a bad setting could influence importantly the final clusterings. Diferent scenarios are considered:

1. $K = 1$: A β-divergence mean is computed for all the training files.

2. $K \neq 1$: As we are testing different combinations of $k \in [1, kmax]$, we are considering the previous computed centroids as inicialization. To add a new class, the worst represented one is splitted in two. We have two criterias for consider the worst represented class:

   (a) $\max_k \|d_\beta(X_{j,k}|C_k)\|$.

   (b) $\max_k \quad \max_j d_\beta(X_{j,k}|C_k)$

   Once the worst class is idenified, a *β-k-mean* clustering (with $k = 2$) is run within the class to split considering as initialization a *β-k-mean* setting as initial centroids the two most different training clips.

In step **5.c** we pretend to find the *β-k-mean* (a centroid representation for all the members of each *k-class*). Which is equivalent to find **P** solving:

$$\hat{\mathbf{P}} = \arg\min_{\mathbf{P}} \sum_{m=1}^{M} d_\beta(\mathbf{P}, \mathbf{X}_m)$$

Where M is the number of members of the class, **P** represents the centroid that is going to be considered as the trained pattern and $\mathbf{X}_m$ is the spectrogram of each training clip class member. Solving this optimitzation problem, is the same as running the β-NMD with an input that contains all $\mathbf{X}_m$ and leading it to update with only one adaptive pattern (**P**) to solve it.

To give all the samples to the β-NMD as input, the clips are concatenated in a single-channel audio file.

The post processing of **H** used in 5.c is to consider the contribution of the secondary activations as part of the event. Is implemented with a convolution of $\vec{H}_{target}$ with [1,1]. This approach follows the idea that the activation after an onset ($\vec{H}_{target}[n+1]$) contributes to explain the same onset ($\vec{H}_{target}[n]$).

To sum up, the previous procedure is an unsupervised β-divergence *k-means* clustering for a specific application: training patterns in a IS-NMF context.

A.5.2.3 Results

Under the following conditions:

- Threshold: $\vec{E}_{local} + mean(\vec{E}_{local})$.
- Each clip to analize is previously normalized by the energy:

$$\sum \vec{signal}^2]/length(\vec{signal}) = 1$$

- The time-frequency representation used is the MEL spectrogram with 40 frames (MEL mapping on the power spectrogram, without considering the phase).
- K_{bgnd}=5.
- N_p=max(**V**)-60dB.

Online approach

All the audio mix is analyzed with overlapped windows (the window size is of 200 frames and the overlapping is of 40 frames).

Under those new detection conditions:

- Using only the local maxima of the peaks for detection.
- A time gap constraint of 10 frames is used: inside those frames a secundary activation is not allowed. The max inside is the frame considered as detection.

Results for only drums mixes, evaluated along the Db6:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
1h 12min 12seg	5	1	0.30762	0.70786	0.42129
1h 12min 12seg	5	3	0.43359	0.86746	0.56622

With the following confusion matrix. In the columns there is the information of the detectections (number of events on each class and false detections) and in the rows there is the information of the ground truth information (itself and the non detected):

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*ND*
1	364	42	0	48	117	30	0	0	0	0	187	3	2	18
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	2	1	2	1	0	0	0	0	3	0	0	9
4	3	6	1	7	4	2	0	0	0	0	2	0	0	9
5	1	5	0	4	334	33	0	0	0	0	27	0	0	29
6	1	0	0	0	137	159	1	0	2	0	34	0	0	107
7	0	0	0	0	10	1	1	0	0	0	1	0	0	43
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	2	0	0	0	27	6	0	0	0	0	0	0	0	71
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	1	0	0	2	1	0	0	0	0	22	0	4	9
12	0	3	0	0	17	22	1	0	0	0	8	4	5	79
13	8	55	13	0	135	116	2	8	4	5	139	55	216	64
FD	18	6	0	0	405	374	0	0	6	0	437	13	2	0

For polyphonic mixes, evaluated along the Db7:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
3h 5min 37seg	5	1	0.19025	0.59057	0.28064
3h 5min 37seg	5	3	0.29096	0.80694	0.41707

With the following confusion matrix. In the columns there is the information of the detectections (number of events on each class and false detections) and in the rows there is the information of the ground truth information (itself and the non detected):

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*ND*
1	271	391	216	139	225	66	10	41	0	28	318	3	44	803
2	0	10	0	0	0	0	0	0	0	1	1	0	0	9
3	5	8	3	10	0	0	0	1	0	0	29	1	0	64
4	2	6	4	6	0	0	0	0	0	0	22	0	2	35
5	2	257	19	56	1940	93	1	56	0	30	427	58	2	87
6	4	50	10	34	201	420	2	2	1	1	134	1	3	223
7	0	2	0	0	9	3	0	0	0	1	47	0	0	173
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	11	0	2	1	1	0	10	25	0	13	0	4	111
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	18	382	8	124	160	69	6	430	35	15	787	408	457	632
FD	870	1280	102	278	544	1341	105	79	94	180	3000	73	165	0

Onsets approach

We only analyze the interest zones (onsets zones) with the detection conditions in section A.4.2.

Results for only drums mixes, evaluated along the Db6, for K_{bgnd}=5:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
1h 23min 8seg	5	1	0.26206	0.22745	0.22639
1h 23min 8seg	5	3	0.53794	0.40794	0.44218

With the following confusion matrix. In the columns there is the information of the detectections (number of events on each class and false detections) and in the rows there is the information of the ground truth information (itself and the non detected):

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*9*	*10*	*11*	*12*	*13*	*ND*
1	47	0	0	0	38	6	0	0	0	0	0	0	335
2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	0	0	0	0	0	0	0	11
4	0	0	0	0	0	0	0	0	0	0	0	0	16
5	6	0	0	0	207	19	12	1	0	10	2	2	155
6	18	0	0	0	165	22	70	19	3	9	0	0	244
7	0	0	0	0	0	0	0	0	0	0	0	0	44
9	0	0	0	0	16	2	3	0	0	0	0	0	71
10	0	0	0	0	0	0	0	0	0	0	0	0	0
11	1	0	0	0	0	0	0	0	0	7	0	2	24
12	0	0	0	0	41	11	10	2	2	4	0	1	83
13	18	1	3	0	90	13	15	1	2	27	0	69	211
FD	5	4	0	0	58	247	50	34	0	11	1	1	0

Results for only drums mixes, evaluated along the Db6, for K_{bgnd}=1:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
1h 13min 43seg	1	1	0.31133	0.3436	0.31258
1h 13min 43seg	1	3	0.54601	0.53209	0.52222

With the following confusion matrix. In the columns there is the information of the detectections (number of events on each class and false detections) and in the rows there is the information of the ground truth information (itself and the non detected):

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*9*	*10*	*11*	*12*	*13*	*ND*
1	190	1	0	0	51	3	2	1	0	0	0	0	192
2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	0	0	0	0	0	0	0	0	0	0	0	11
4	1	0	0	1	0	0	0	0	0	0	0	0	15
5	20	0	0	1	195	24	17	2	1	13	4	6	167
6	47	0	2	0	170	38	72	39	5	17	1	0	228
7	1	0	0	0	6	0	0	0	0	0	0	0	44
9	4	0	0	0	18	2	4	0	0	0	0	0	71
10	0	0	0	0	0	0	0	0	0	0	0	0	0
11	2	0	0	0	0	0	0	0	0	11	1	3	20
12	0	0	0	0	45	10	10	12	4	4	4	4	79
13	19	2	4	0	90	11	17	1	2	27	1	112	168
FD	12	11	2	3	59	226	61	40	6	15	3	2	0

For polyphonic mixes, evaluated along the Db7:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
4h 39min 57seg	5	1	0.22008	0.56972	0.30967
4h 39min 57seg	5	3	0.34075	0.76203	0.46037

With the following confusion matrix. In the columns there is the information of the detectections (number of events on each class and false detections) and in the rows there is the information of the ground truth information (itself and the non detected):

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*ND*
1	582	102	65	8	62	4	111	0	0	193	164	0	42	492
2	1	1	1	1	0	0	0	0	0	6	1	0	1	18
3	17	3	2	5	0	0	3	0	0	9	11	0	1	65
4	8	8	0	2	0	0	4	0	0	6	8	0	2	39
5	92	241	50	38	1773	32	114	0	13	426	166	4	169	253
6	57	71	9	9	142	14	84	0	14	109	94	4	48	629
7	6	1	0	0	8	0	59	0	0	19	15	0	2	114
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	1	0	2	0	1	12	11	0	1	134
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	154	311	21	116	167	52	527	0	23	221	668	556	646	443
FD	392	342	137	123	153	835	539	3	202	1831	443	29	377	0

Where the numbers correspond to the following table class:

Class	Class number
$Kick$	1
$H-tom$	2
$L-tom$	3
$M-tom$	4
$ClosedHH$	5
$OpenHH$	6
$Ride$	7
$Chinese$	8
$Crash$	9
$Splash$	10
$Cross-stick$	11
$Cowbell$	12
$Snare$	13
$NoDetection-FalseDetection$	ND - FD

A.6 WP.4.T7

Considering different thresholds for the Online Approach:

- **max(Htar(global energy),Sig(local energy))**

  Results for only drums mixes, evaluated along the Db6:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
1h 15min 42seg	5	1	0.28791	0.7164	0.40777
1h 15min 42seg	5	3	0.40361	0.86111	0.5447

  With the following confusion matrix. In the columns there is the information of the detectections (number of events on each class and false detections) and in the rows there is the information of the ground truth information (itself and the non detected):

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*ND*
1	366	112	39	48	101	21	1	1	0	1	162	7	8	14
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	1	7	1	2	1	0	0	0	0	3	0	3	4
4	3	9	4	7	2	1	0	0	0	0	1	1	0	9
5	1	10	3	4	301	32	3	0	0	1	34	0	4	61
6	0	0	0	0	132	163	8	0	6	1	37	1	0	103
7	0	1	0	0	10	1	2	0	0	0	1	0	0	42
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	3	3	2	0	26	14	1	0	2	0	0	0	0	69
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	3	1	0	2	2	0	0	0	0	21	1	9	10
12	2	3	0	0	15	29	1	0	0	1	8	3	5	80
13	12	83	25	0	114	106	23	37	6	73	134	95	240	40
FD	32	28	10	0	160	366	10	0	2	17	446	13	2	0

For polyphonic mixes, evaluated along the Db7:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
4h 39min 57seg	5	1	0.22132	0.58278	0.31406
4h 39min 57seg	5	3	0.32721	0.78359	0.45179

With the following confusion matrix. In the columns there is the information of the detectections (number of events on each class and false detections) and in the rows there is the information of the ground truth information (itself and the non detected):

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*ND*
1	268	295	156	53	200	50	19	36	0	19	310	5	31	806
2	0	8	0	0	0	0	0	0	0	1	1	0	0	11
3	5	5	2	3	0	0	0	1	0	0	26	1	0	65
4	2	5	1	3	0	0	0	0	0	0	20	0	2	38
5	1	88	13	11	1919	93	11	42	0	17	390	48	2	108
6	4	32	2	10	172	423	1	1	1	1	115	3	3	220
7	0	1	0	0	13	3	8	0	0	1	37	1	0	165
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	6	0	1	2	0	0	8	24	1	12	0	4	112
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	15	250	2	36	152	67	15	374	37	8	767	411	427	662
FD	838	629	42	54	382	1271	132	60	70	98	2736	82	115	0

- **Htar(global energy) + Sig(local energy)**

  Results for only drums mixes, evaluated along the Db6:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
1h 12min 12seg	5	1	0.28576	0.72282	0.40611
1h 12min 12seg	5	3	0.41186	0.86752	0.55243

  With the following confusion matrix. In the columns there is the information of the detectections (number of events on each class and false detections) and in the rows there is the information of the ground truth information (itself and the non detected):

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*ND*
1	366	93	23	49	101	22	11	2	0	1	155	8	6	16
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	1	6	1	2	1	0	0	0	0	3	0	2	5
4	3	9	4	7	2	1	0	0	0	0	1	1	1	9
5	1	10	2	4	309	35	12	0	0	1	32	0	4	53
6	0	0	0	0	133	161	44	0	11	0	31	1	0	105
7	0	0	0	0	10	1	11	0	0	0	1	0	0	33
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	3	2	1	0	27	14	7	0	3	0	0	0	0	68
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	3	1	0	2	3	0	0	0	0	21	1	9	10
12	2	3	0	0	17	29	4	0	0	1	8	1	5	82
13	11	73	22	0	116	105	92	46	11	60	130	91	239	41
FD	30	18	5	0	207	349	53	0	2	15	404	13	2	0

For polyphonic mixes, evaluated along the Db7:

Processing Time	K_{bgnd}	Hierarchical level	Precision	Recall	F-measure
4h 39min 57seg	5	1	0.2215	0.6026	0.31722
4h 39min 57seg	5	3	0.32358	0.79148	0.44983

With the following confusion matrix. In the columns there is the information of the detectections (number of events on each class and false detections) and in the rows there is the information of the ground truth information (itself and the non detected):

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*ND*
1	268	294	157	58	215	55	84	34	1	18	277	5	41	806
2	0	8	0	0	0	0	0	0	0	1	1	0	0	11
3	5	5	1	2	0	0	1	1	0	0	20	1	0	66
4	2	5	1	4	0	0	2	0	0	0	17	0	2	37
5	1	91	13	18	1942	95	53	51	1	17	312	57	6	85
6	4	31	2	15	187	423	19	1	1	1	102	2	3	220
7	0	1	0	0	14	3	28	1	0	1	34	1	1	145
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	6	0	0	2	0	2	8	25	0	12	0	5	111
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	15	249	2	46	157	72	83	419	52	8	743	447	489	600
FD	839	604	42	72	456	1257	606	53	85	96	2335	74	165	0

A.7 WP.4.T8

A.7.1 Estimating K

K_{kbgnd} is a sensitive parameter to set; due to that fact, we are going to try to estimate K. Two strategies are considered:

- Finding a general rule from the analysis of the white noise.
- Finding on the fly the optimal number of K for the specific mix to analyze.
- Dividing the spectrogram in K segments of *lpattern*.

We will refer as K_{opt} to the K "optimal" number of patterns that describe the whole audio scene to analyze.

A.7.1.1 Obtaining a rule analyzing white noise

The quality is evaluated by means of the cost for each bin at each time position. A good representation is considered if this quality parameter is below 0'01. Iterativelly, different number of K's are going to be tried till a good representation is achieved. K_{opt} is the number of K needed to achieve that the quality parameter is 0'01.

K_{opt} is given in function of the segment length:

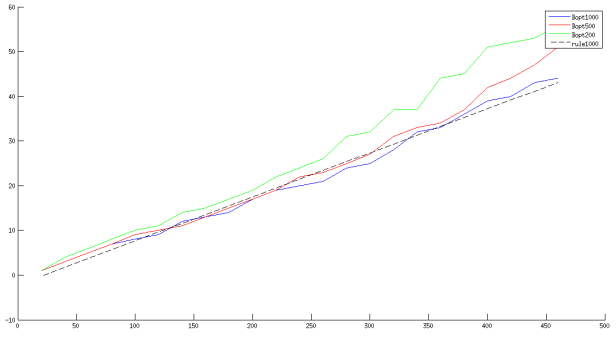

FIGURE A.14: That plot ilustrates the representation of K_{opt} in function of the length of the spectrogram. The three continous curves represent the K_{opt} parameter depending on the allowed number of iterations of the NMD (200 [green], 500 [red] and 1000 [blue]). The discontinous line corresponds to the obtained rule from the blue line (allowed number of iterations: 1000).

Where the discontinous black line is the linear function that approximates K_{opt} from the length of the signal to analyze in frames (L) considering as $lpattern$=20:

$$K_{opt} = 0.0985 \cdot L - 2.1976$$

Also is interesting to see the behaviour of the number of iterations. When there are not enough adaptive patterns, the algorithm can not be better and converges with a small ammount of iterations (less than 200). We can see this behaviour in the previous Figure A.14 where we can observe clearly that we need more patterns if we set a maximum number of iterations (because we don't leave the matrices to update till a good representation is achieved).

A.7.1.2 On the fly

The previous iterative method (to find K_{opt} in the general case of the white noise) could be computed before each NMD. So, for each specific piece of audio to analyze we are going to estimate K_{opt} trying different number of K's till the quality parameter is below 0'01.

Note that if using the rule derived from the general case of the noise, it can easily lead us to an over-dimensionated K_{opt}. But on the other side, trying to find the optimal number of patterns from each excerpt of audio iteratively, it would imply adding some computational cost.

The fact that in the "rule" case K_{opt} is overdimensionated can lead us to a slower system comparing with the "on the fly" case. If the segments to analyze (length of $\mathbf{V}$) are short enough, K_{opt} will be small and the computational cost of finding the optimal number of patterns will not be high.

The patterns are initialized random in each test, which is critical. A bad inicialization can lead us to a bigger K_{opt} estimation than the needed.

A.7.1.3 Segmenting with K patterns

This approach is based on the idea of dividing $\mathbf{V}$ in K segments of length *lpattern* to initialize the patterns. To obtain this representation, the $\mathbf{H}$ matrix is forced to be zero exept in the frame where the segment starts.

The last segment, which not necessarily is going to have the *lpattern* length, is going to be fulfilled with zeros.

A.7.2 Checking representation with random W_{bgnd}

In this section is going to be analyzed how the activations behave when K_{bgnd} is modified in different scenarios: analyzing white noise (where we expect that our background patterns model it) and analyzing drums (where we expect to have prominent activations at the target patterns).

The background patterns are initialized random. Here we want to study the behaviour of the system working with a $\mathbf{W}$ that has two parts (one random and the other with trained drums) focusing on the impact of using different K_{bgnd}.

A.7.2.1 White noise analysis

Considering as input a white noise signal (always the same) of 60 frames, the evolution of the mean of the sum over the interest patterns (background or target) of the activations in function of K_{bgnd} is the following:

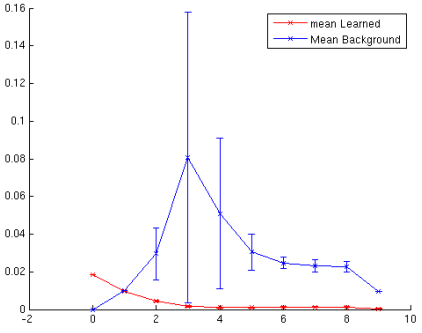

FIGURE A.15: Representation of the behaviour of the activations respect to K. In red we observe the mean of the sum of the activations over K_{tar} and in blue we observe the mean of the activations over K_{bgnd}.

In the previous example, K_{opt} is: 4 (considering the "rule" approach), 4 (considering the "on the fly" approach) or 3 (considering the "segmenting" approach).

Note that the activations of K_{tar} are nearly zero, but they are not zero. Depending on the initialization those are more or less used.

We can see in the next graphic how the trained patterns don't affect the decompositions as the signal to decompose is really different to the trained patterns (white noise):

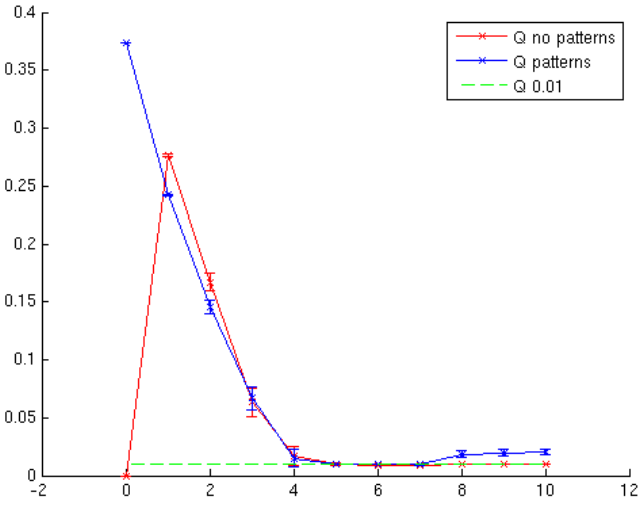

FIGURE A.16: Representation of the behaviour of the quality value respect to K. In red we observe its behaviour without including the trained patterns, in blue considering the trained patterns plus adaptive patterns and in green is the quality threshold.

In addition, is interesting to see how the trained patterns deal with the situation of approximating the white noise. Depending on the input white noise, the NMD uses the snare drum, the splash, the splash and/or the hi-hat to explain the noise.

A.7.2.2 Drums analysis

In that case we are going to generate the input signal with a chain of 5 drum events: Splash1, Splash1, Ride, High Tom and Splash1 (100 frames in total, 20 frames each). The used sounds to generate the input signal are from the trianing patterns dataset and Splash1 is always the same event. The evolution of the mean of the sum over the interest patterns (background and target) of the activations in function of K is the following:

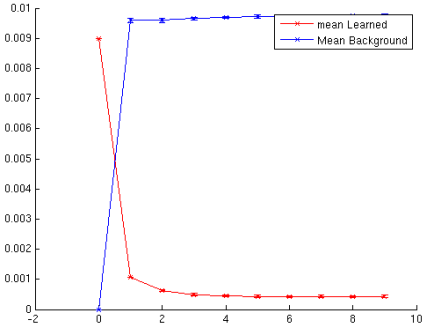

FIGURE A.17: Representation of the behaviour of the activations respect to K. In red we observe the mean of the sum of the activations over K_{tar} and in blue we observe the mean of the activations over K_{bgnd}.

In the previous example, K_{opt} is: 8 (considering the "rule" system), 6 (considering the "on the fly" system) or 5 (considering the "segmenting" approach). But we know a priori, that K_{opt} should be 3 (as there are 3 different sources).

The quality parameter, once the learned patterns are included, is never rised (even we include 100 K_{bgnd}). What means that the pre-trained patterns interfere to obtain perfect representation. The background is not able to adapt properly because the learned patterns explain enough good (with little error) the events.

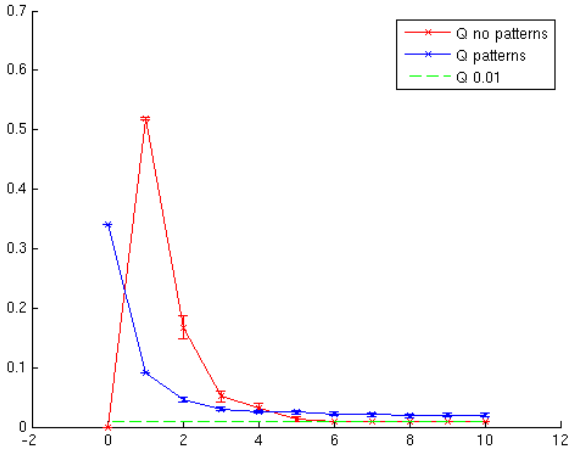

FIGURE A.18: Representation of the behaviour of the quality value respect to K. In red we observe its behaviour without including the trained patterns, in blue considering the trained patterns plus adaptive patterns and in green is the quality threshold.

In the following tables we can observe that the splash is the "worst" trained target (is the one that never rised the performance parameters of the training patterns step).

For $K_{bgnd} = 0$ it comes out the following scores:

Target	**Mean of the sum**	**Target**	**Mean of the sum**
Kick	0.000000	**Chinese**	0.000827
High Tom	0.000036	**Crash**	0.001801
Low Tom	0.000001	**Splash**	0.005789
Mid Tom	0.000000	**Cross-stick**	0.000000
Closed HH	0.000002	**Cowbell**	0.000065
Open HH	0.000000	**Snare**	0.000030
Ride	0.000455	-	-

For $K_{bgnd} = 1$ it comes out the following scores:

Target	**Mean of the sum**	**Target**	**Mean of the sum**
Kick	0.000000	**Chinese**	0.000011
High Tom	0.000036	**Crash**	0.000363
Low Tom	0.000001	**Splash**	0.000109
Mid Tom	0.000000	**Cross-stick**	0.000000
Closed HH	0.000002	**Cowbell**	0.000404
Open HH	0.000000	**Snare**	0.000041
Ride	0.000093	-	-

For $K_{bgnd} = 8$ it comes out the following scores:

Target	**Mean of the sum**	**Target**	**Mean of the sum**
Kick	0.000000	**Chinese**	0.000005
High Tom	0.000038	**Crash**	0.000119
Low Tom	0.000001	**Splash**	0.000017
Mid Tom	0.000000	**Cross-stick**	0.000000
Closed HH	0.000002	**Cowbell**	0.000184
Open HH	0.000001	**Snare**	0.000023
Ride	0.000038	-	-

We can observe that the High Tom and the Ride patterns can represent the event of their class, meanwhile the splash event needs the splash, the crash and the chinese cymbal.

A.7.3 Computational cost

Online approach: The following table reflects the required time to decompose the training thresholds dataset.

K	hh:mm:ss
Fixed to 5	04:00:30
K_{opt} rule	08:57:13
K_{opt} "on the fly"	17:50:42
K_{opt} "segmenting"	06:08:51

Note that, as we expected, the "on the fly method" is much more expensive in terms of computational cost. That's due to the fact that the analysis window of the online approach is not enough small (200 frames) and the K_{opt} approximation with that method is costly.

Onsets approach: The following table reflects the required time to decompose the training thresholds dataset and optimize the thresholds.

K	hh:mm:ss
Fixed to 5	06:58:23
K_{opt} rule	09:21:53
K_{opt} "on the fly"	08:32:49
K_{opt} "segmenting"	08:29:45

As we can see above, by means of estimating K_{opt} with the "on the fly" or the "segmenting" method the system goes faster to compute the same dataset.

A.7.4 Bug correction and results

An important bug was found in the onsets approach. A normalization of each onset zone was done leading the system to lots of false positives in zones where there was no events.

Onsets Approach: for polyphonic mixes considering the different K_{bgnd} estimations.

In those first results the used threshold is: mean(H).

K_{bgnd}	Level	Precision	Recall	F-measure
5	1	0.23766	0.60874	0.34
5	2	0.27666	0.64583	0.38448
5	3	0.35264	0.79173	0.48578
$Rule(2)$	1	0.31117	0.54172	0.39363
$Rule(2)$	2	0.36431	0.62881	0.4593
***Rule*(2)**	**3**	**0.44447**	**0.74294**	**0.55387**
Fly(2-3)	1	0.27981	0.60767	0.38083
Fly(2-3)	2	0.31066	0.64325	0.41679
Fly(2-3)	3	0.38794	0.77622	0.51464
$Segment(2)$	1	0.26878	0.6111	0.37104
$Segment(2)$	2	0.3186	0.68811	0.43312
$Segment(2)$	3	0.39271	0.82036	0.52831

The confusion matrix that comes out from the bold result of the previous table is the following:

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*ND*
1	678	28	81	0	165	1	2	10	1	77	578	5	80	396
2	0	0	0	0	0	0	0	0	0	2	6	0	1	19
3	11	0	0	0	0	0	0	0	0	1	33	0	0	67
4	11	0	1	0	0	0	0	0	0	2	23	0	5	41
5	14	1	18	0	1894	0	1	0	11	33	743	2	5	132
6	28	1	0	0	120	0	0	0	9	16	214	2	8	643
7	1	0	0	0	9	0	2	0	1	6	56	0	1	171
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	1	14	8	19	0	6	121
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	135	18	2	3	164	0	15	12	81	36	806	229	381	708
FD	292	8	7	4	169	26	13	9	122	127	1363	15	66	0

Where the numbers correspond to the following table class:

Class	Class number
$Kick$	1
$H-tom$	2
$L-tom$	3
$M-tom$	4
$ClosedHH$	5
$OpenHH$	6
$Ride$	7
$Chinese$	8
$Crash$	9
$Splash$	10
$Cross-stick$	11
$Cowbell$	12
$Snare$	13
$NoDetection-FalseDetection$	ND - FD

In those results the used threshold is: mean(H)+localEnergy.

K_{bgnd}	Level	Precision	Recall	F-measure
5	1	0.30021	0.68928	0.41259
5	2	0.33955	0.73071	0.45853
5	3	0.37721	0.83422	0.54691
Rule(2)	1	0.31026	0.68329	0.42011
Rule(2)	2	0.373	0.72852	0.48708
Rule(2)	3	0.44472	0.81645	0.56862
Fly(1-3)	1	0.31093	0.67072	0.41695
Fly(1-3)	2	0.37659	0.71391	0.48554
***Fly*(1-3)**	**3**	**0.4553**	**0.8086**	**0.57423**
Segment(2)	1	0.28591	0.67929	0.39594
Segment(2)	2	0.3424	0.72434	0.45859
Segment(2)	3	0.41316	0.82475	0.54304

The confusion matrix that comes out from the bold result of the previous table is the following:

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*ND*
1	645	108	121	7	194	41	176	1	1	104	136	9	53	429
2	0	1	0	0	0	0	0	0	0	3	1	0	1	18
3	14	3	1	1	1	0	3	0	0	3	13	0	1	66
4	7	4	2	0	0	1	2	0	0	2	5	0	3	41
5	19	57	45	3	1933	82	207	0	32	162	93	5	61	94
6	18	30	1	0	221	432	63	1	14	25	100	1	17	211
7	0	1	0	0	27	3	52	0	3	13	16	0	1	121
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	3	2	0	20	1	12	0	4	115
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	103	127	5	13	189	130	536	10	250	76	609	493	497	592
FD	314	152	38	23	293	528	481	3	157	348	425	19	132	0

Online Approach: for polyphonic mixes considering the different K_{bgnd} estimations. The used threshold is: mean(H)+localEnergy.

K_{bgnd}	Level	Precision	Recall	F-measure
5	3	0.24387	0.79169	0.0.36775
Rule	3	0.30643	0.78422	0.43178
Fly	**3**	**0.38644**	**0.73782**	**0.49704**
Segment	3	0.34513	0.79297	0.47057

The confusion matrix that comes out from the bold result of the previous table is the following:

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*ND*
1	256	246	167	68	182	5	96	76	0	9	49	1	79	818
2	0	6	0	0	0	0	0	1	0	1	0	0	0	13
3	6	3	2	6	1	0	1	3	0	0	2	1	0	65
4	3	6	2	2	0	0	1	3	0	0	3	0	2	39
5	11	179	32	43	1914	20	106	133	3	15	89	65	39	112
6	4	34	12	27	152	126	22	27	1	1	25	3	4	517
7	0	1	0	0	21	0	25	3	0	1	5	0	1	148
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	4	0	0	1	0	2	14	18	0	6	0	5	118
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	20	241	1	55	133	7	131	631	70	5	500	312	557	532
FD	932	590	89	191	428	139	740	212	76	70	545	23	222	0

A.8 WP.4.T9

A.8.1 Improving representation

We noticed that our detection/training problems could be caused because the representation of some of the learned target-class are not good enough (especially for the cross-stick, splash and snare).

A.8.1.1 Removing the cross-stick class

This class is not well represented because we don't have enough data to train it propperly. In order to avoid the distortion of the results that this class implies, we remove the cross-stick class and we consider all the events of the cross-stick as snare.

The results improve significantlly, as we can see in the following tables:

Onsets approach: evaluated along polyphonic mixes. In those results the used threshold is: mean(H_{tar}).

K_{bgnd}	Level	Precision	Recall	F-measure
5	3	0.4582	0.7285	0.5598
$Rule(2)$	**3**	**0.6244**	**0.6813**	**0.6518**
Fly(1-3)	3	0.5753	0.7417	0.6244
$Segment(2)$	3	0.5215	0.7571	0.6134

The confusion matrix that comes out from the bold result of the previous table is the following:

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*ND*
1	672	26	72	0	165	66	13	11	1	97	3	61	402
2	0	0	0	0	0	0	0	0	0	2	0	1	19
3	11	0	0	0	0	1	0	0	0	1	0	0	67
4	11	0	0	0	0	0	0	0	0	2	0	4	41
5	15	2	16	0	1894	66	1	0	9	45	1	5	132
6	27	1	0	0	120	334	0	0	8	24	1	5	309
7	1	0	0	0	9	0	4	0	1	6	0	1	169
8	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	2	0	1	15	8	0	5	120
10	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0
12	131	15	2	3	164	115	18	14	65	50	217	358	731
FD	295	8	5	4	169	363	18	10	115	165	12	55	0

In those results the used threshold is: mean(H_{tar})+localEnergy.

K_{bgnd}	Level	Precision	Recall	F-measure
5	3	0.43684	0.81141	0.56143
$Rule$(2)	3	0.46475	0.78597	0.57702
Fly(1-3)	**3**	**0.51394**	**0.77223**	**0.608**
$Segment$(2)	3	0.44447	0.79513	0.56319

The confusion matrix that comes out from the bold result of the previous table is the following:

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*ND*
1	639	42	116	7	188	42	156	1	1	105	9	53	435
2	0	0	0	0	0	0	0	0	0	3	0	1	19
3	12	2	2	1	1	0	2	0	0	3	0	1	65
4	7	2	2	0	0	1	2	0	0	2	0	3	41
5	20	19	38	1	1932	79	178	0	31	162	5	58	95
6	18	8	1	0	218	433	55	1	14	25	1	15	210
7	0	0	0	0	27	3	45	0	3	13	0	1	128
8	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	3	3	0	22	1	0	3	113
10	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0
12	105	38	4	12	188	134	526	10	255	81	489	496	593
FD	318	53	36	22	296	533	436	3	156	344	19	116	0

Online approach: evaluated along polyphonic mixes. In those results the used threshold is: mean(H_{tar})+localEnergy.

K_{bgnd}	Level	Precision	Recall	F-measure
5	3	0.2667	0.77286	0.39016
$Rule$(18)	3	0.34392	0.72369	0.45782
Fly(11-14)	**3**	**0.36478**	**0.76046**	**0.48156**
$Segment$(10)	3	0.35001	0.76909	0.47096

The confusion matrix that comes out from the bold result of the previous table is the following:

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*ND*
1	251	264	226	18	185	29	104	81	0	45	1	88	823
2	0	6	0	0	0	0	0	2	0	1	0	0	13
3	5	3	5	0	1	0	3	4	0	0	1	0	62
4	3	6	4	1	0	0	2	4	0	0	0	2	40
5	12	196	48	14	1915	92	102	136	8	59	65	53	111
6	5	39	29	11	153	386	26	31	1	3	1	3	257
7	0	1	0	0	21	3	28	4	0	1	0	1	145
8	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	4	0	0	0	1	3	20	20	0	0	7	116
10	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0
12	19	262	10	20	135	42	137	687	89	29	361	592	497
FD	925	606	173	40	422	1207	807	279	89	301	28	260	0

A.8.1.2 Improving representation of the splash and the snare

Onsets approach: evaluated along polyphonic mixes. In those results the used threshold is: mean(H_{tar}).

K_{bgnd}	Level	Precision	Recall	F-measure
5	3	0.59515	0.63498	0.61194
Rule(2)	3	0.6545	0.66959	0.65868
Fly(1-3)	3	**0.69454**	**0.69657**	**0.69106**
Fly with inicialization(1-3)	3	**0.68952**	**0.69808**	**0.68922**
Segment(2)	3	0.60642	0.71724	0.65332
Segment with inicialization(2)	3	**0.7688**	**0.5271**	**0.6240**

In those results the used threshold is: mean(H_{tar})+localEnergy.

K_{bgnd}	Level	Precision	Recall	F-measure
5	**3**	**0.56358**	**0.75485**	**0.63267**
Rule(2)	3	0.52175	0.77342	0.61244
Fly(1-3)	3	0.49342	0.71165	0.5717
Segment(2)	3	0.44754	0.60992	0.50565

The confusion matrix that comes out from the bold result of the previous table is the following:

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*ND*
1	639	7	169	1	167	66	246	4	1	15	21	29	435
2	0	0	0	0	0	0	0	0	0	2	0	0	19
3	11	0	4	0	0	0	7	0	0	0	0	1	63
4	10	0	5	0	0	0	4	0	0	0	0	1	41
5	20	0	34	1	1898	94	260	0	13	27	6	35	129
6	6	0	2	0	129	462	82	0	6	4	1	14	181
7	0	0	0	0	14	4	74	0	1	2	0	1	99
8	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	1	2	0	0	17	1	0	0	118
10	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0
12	68	1	8	11	175	152	560	1	110	4	510	501	588
FD	270	1	33	17	224	513	668	2	84	74	17	79	0

Online approach: evaluated along polyphonic mixes. In those results the used threshold is: mean(H_{tar})+localEnergy.

K_{bgnd}	Level	Precision	Recall	F-measure
5	3	0.42018	0.77155	0.5286
Rule(18)	3	0.38642	0.74141	0.49455
Fly(11-14)	3	0.41667	0.77152	0.52694
Segment(10)	**3**	**0.433**	**0.76338**	**0.53767**

The confusion matrix that comes out from the bold result of the previous table is the following:

-	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*ND*
1	293	213	263	0	189	40	59	13	3	14	18	38	781
2	0	6	0	0	0	0	0	0	0	0	0	0	13
3	5	3	6	0	1	2	0	1	0	0	1	0	61
4	2	7	3	1	0	0	2	0	0	0	2	0	40
5	13	103	58	0	1920	111	68	46	43	10	87	34	106
6	5	19	36	0	166	391	17	2	7	2	6	5	252
7	0	1	0	0	21	8	19	1	1	0	0	0	154
8	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	5	2	0	1	4	4	2	36	0	0	6	100
10	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0
12	13	216	29	3	125	39	80	168	204	4	421	636	453
FD	905	448	130	1	420	1135	467	15	150	84	55	147	0

Appendix B

Additional information about the Work plan

B.1 Extended methods and procedures

Some research has been done previously at IRCAM in the field of automatic drums transcription with Non-negative Matrix Factorization (NMF from now on) [4]. Axel Roebel thought in changing the approach for automatic drums transcription: using NMD instead of NMF. NMD is an extension of NMF which is capable to identify patterns with a temporal structure. Due to this improvement, the new approach fits better in our engineering problem because the elements of the drum set have a determined temporal structure. Our aim is to check if better results can be achieved. For more information about NMD and NMF, check Chapter 2 and Chapter 3.

Nowadays, IRCAM is developing a source detection framework in multi channel audio streams which is based on Non-Negative Tensor Factor Deconvolution (NTD from now on) [5]. The evaluation is made on 3DTV 5.1 film soundtracks with impulsive target sounds like gunshots. Axel Roebel's idea was to adapt the IRCAM's 3DTVs algorithm to detect drum onsets in order to do automatic drums transcription.

An adaptation of the IRCAM's 3DTVS algorithm should be done. All the work, can be summarised in 4 main blocks:

- *Improving detection step*: The 3DTVS algorithm detects only one target for mix. The drum sets have more than one target: hi-hat, low tom, snare drum, bass drum, splash, and so on. The algorithm has to be able to detect all the drum instruments at the same time and to give all the results together.

- *Improve training step*: IRCAM's 3DTVS framework needs to be trained: patterns (which describe each drum instrument, we can understand them as a time-frequency "signature") and thresholds (for taking good decisions depending on the usual dynamic range for each drum instrument activations). For training the thresholds, our goal is to get the most representative for each drum instrument. Those thresholds should be robust in different circumstances: different music styles, with/without background music, different drummers, and so on. By the other hand, confusions in-between drum instruments are normal due to the high correlation in-between them. In order to avoid confusions, new ways of training-test should be explored.

- *Modify the evaluation system*: As described before, we detect more than one target for mix at the same time. To quantify properly the performance of the framework a deep modification is needed. The system is going to evaluate the performance in terms of Precision, Recall and F-measure. In addition to a classical F-measure an evaluation with hierarchical constraints is going to be implemented for checking the performance in different levels. For more information about hierarchical constraints, see Chapter 4.1.1.3.

- *Adapt the NMD model for our exactly engineering problem: drums detection and transcription.* Even the problem is similar to the encountered in a 3DTVS context detecting impulsive sounds, incorporating more than one target and the own issues of drums transcription will lead us in thinking about methods that performs the transcription as expected.

B.2 Work plan: tables and figures.

B.2.1 Tasks

Project: Automatic Drums Transcription	**WP ref**: 1
Short description: Read and understand background literature about drums automatic transcription.	**Planned start date**: 16 Sept 2013 **Planned end date**: 27 Sept 2013
Internal task T1: Identify most relevant literature.	**Deliverables**: Database of most important publications (17 Sept 2013)
Internal task T2: Read and understand most relevant literature.	**Deliverables**: No deliverables (27 Sept 2013)

Table B.1: Work Package 1

Project: Automatic Drums Transcription	**WP ref**: 2
Short description: Read and understand the IRCAM's 3DTVs framework.	**Planned start date**: 30 Sept 2013 **Planned end date**: 11 Oct 2013
Internal task T1: Copy Matlab files to my workstation and make them work.	**Deliverables**: No deliverables (1 Oct 2013)
Internal task T2: Read and understand the IRCAM's 3DTVs Matlab code.	**Deliverables**: No deliverables (11 Oct 2013)

Table B.2: Work Package 2

Project: Automatic Drums Transcription	**WP ref**: 3
Short description: Adaptation of the IRCAM's 3DTVs framework.	**Planned start date**: 14 Oct 2013 **Planned end date**: 8 Nov 2013
Internal task T1: Modify detection step.	**Deliverables**: Team meeting approbation (22 Oct 2013)
Internal task T2: Modify the evaluation system.	**Deliverables**: Team meeting approbation (31 Oct 2013)
Internal task T3: Modify thresholds training step.	. **Deliverables**: Team meeting approbation (8 Nov 2013)

Table B.3: Work Package 3

Project: Automatic Drums Transcription	**WP ref**: 4
Short description: Fitting the NMD model to drums automatic transcription in order to get better results.	**Planned start date**: 11 Nov 2013 **Planned end date**: 20 Jun 2013
Internal task T1: Check, understand and improve the framework.	**Deliverables**: Team meeting approbation (20 Dec 2014)
Internal task T2: Christmas Holidays	**Return**: 3 Jan 2014
Internal task T3: Introduce New Approach 1. Check, understand and improve the framework.	**Deliverables**: Team meeting approbation (14 Feb 2014)
Internal task T4: Introduce New Approach 2. Check, understand and improve the framework.	**Deliverables**: Team meeting approbation (7 Mar 2014)
Internal task T5: Introduce New Approach 3. Check, understand and improve the framework.	**Deliverables**: Team meeting approbation (18 Apr 2014)
Internal task T6: Introduce New Approach 4. Check, understand and improve the framework.	**Deliverables**: Team meeting approbation (30 May 2014)
Internal task T7: Introduce New Approach 5. Check, understand and improve the framework.	**Deliverables**: Team meeting approbation (20 Jun 2014)

Table B.4: Work Package 4

Project: Automatic Drums Transcription	**WP ref**: 5
Short description: Redaction.	**Planned start date**: 23 Jun 2014 **Planned end date**: 27 Jun 2014
Internal task T1: Redaction of the UPC-TelecomBCN Report.	**Deliverables**: Report (27 Jun 2014)

Table B.5: Work Package 5

Project: Automatic Drums Transcription	**WP ref**: 6
Short description: Improving training thresholds approach.	**Planned start date**: 30 Jun 2014 **Planned end date**: 25 Jul 2014
Internal task T1: Improved training thresholds approach.	**Deliverables**: Report (25 Jul 2014)

Table B.6: Work Package 6

Project: Automatic Drums Transcription	**WP ref**: 7
Short description: Improving detection approach.	**Planned start date**: 28 Jul 2014 **Planned end date**: 22 Ago 2014
Internal task T1: Improved detection approach.	**Deliverables**: Report (22 Ago 2014)

Table B.7: Work Package 7

Project: Automatic Drums Transcription	**WP ref**: 8
Short description: Redaction: publications and IRCAM report.	**Planned start date**: 25 Ago 2014 **Planned end date**: 29 Ago 2014
Internal task T1: Redaction of the publications and finalizing the IRCAM report.	**Deliverables**: Publications and report.

Table B.8: Work Package 8

B.2.2 Milestones

#WP	#Task	Short Title	Milestone	Date
1	1	Identify most relevant literature.	Do a list of important publications.	17 Set 2013
1	2	Read and understated most relevant literature.	Understand most important approaches for automatic drums transcription.	27 Set 2013
2	1	Copy Matlab files to my workstation and make them work.	Matlab scripts running.	1 Oct 2013
2	2	Read and understated most the IRCAM's 3DTVs Matlab code.	Understand IRCAM's 3DTVs Matlab code.	11 Oct 2013
3	1	Modify detection step.	Improved detection step working.	22 Oct 2013
3	2	Modify the evaluation system.	Improved evaluation system working.	31 Oct 2013
3	3	Modify thresholds training step.	Improved thresholds training step working.	8 Nov 2013
4	1	Check, understand and improve the framework.	Improved framework working. Detection of weak points and think about new possible approaches to solve them.	20 Dec 2014
4	2	Christmas Holidays	—	3 Jan 2014

Table B.9: Milestones I

#WP	#Task	Short Title	Milestone	Date
4	3	Introduce New Approach 1. Check, understand and improve the framework.	Improved framework working. Detection of weak points and think about new possible approaches to solve them.	14 Feb 2014
4	4	Introduce New Approach 2. Check, understand and improve the framework.	Improved framework working. Detection of weak points and think about new possible approaches to solve them.	7 Mar 2014
4	5	Introduce New Approach 3. Check, understand and improve the framework.	Improved framework working. Detection of weak points and think about new possible approaches to solve them.	18 Apr 2014
4	6	Introduce New Approach 4. Check, understand and improve the framework.	Improved framework working. Detection of weak points and think about new possible approaches to solve them.	30 May 2014
4	7	Introduce New Approach 5. Check, understand and improve the framework.	Improved framework working.	20 Jun 2014

Table B.10: Milestones II

#WP	**#Task**	**Short Title**	**Milestone**	**Date**
5	1	Redaction of the UPC-TelecomBCN Report	Submission of the report.	27 Jun 2014
6	0	Improving training thresholds approach.	Framework working.	25 Jul 2014
7	0	Improving detection approach.	Framework working.	22 Ago 2014
8	0	Redaction: publications and IRCAM report.	Publications and report.	29 Ago 2014

TABLE B.11: Milestones III

B.2.3 Gantt Diagram

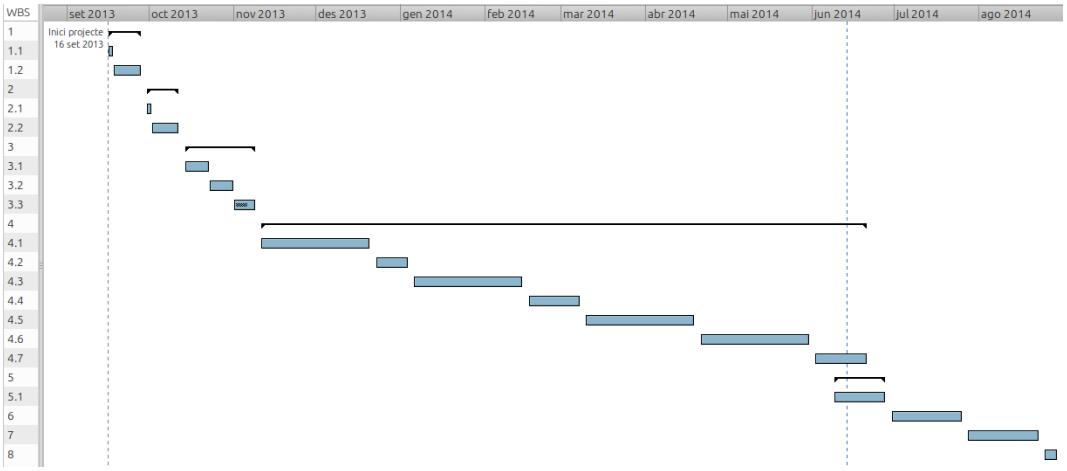

FIGURE B.1: Grantt Diagram

WBS	Nom	Feina
1	▼ **Read and understand background literature about drums automatic transcription.**	**10d**
1.1	Identify most relevant literature.	2d
1.2	Read and unerstand background literature about drums automatic transcription.	8d
2	▼ **Read and understand the IRCAM's 3DTVs framework.**	**10d**
2.1	Copy Matlab files to my workstation and make them work.	2d
2.2	Read and understand the IRCAM's 3DTVs Matlab code.	8d
3	▼ **Adaptation of the IRCAM's 3DTVs framework.**	**20d**
3.1	Improving detection step.	7d
3.2	Modify the evaluation system.	7d
3.3	Improving thresholds training step.	6d
4	▼ **Fitting the NMD model to drums automatic transcription.**	**160d**
4.1	Check, understand and improve the framework.	30d
4.2	Christmas Holidays	10d
4.3	Introduce New Approach 1. Check, understand and improve the framework.	30d
4.4	Introduce New Approach 2. Check, understand and improve the framework.	15d
4.5	Introduce New Approach 3. Check, understand and improve the framework.	30d
4.6	Introduce New Approach 4. Check, understand and improve the framework.	30d
4.7	Introduce New Approach 5. Check, understand and improve the framework.	15d
5	▼ **Redaction**	**15d**
5.1	Redaction of the UPC-TelecomBCN Report	15d
6	Improving training thresholds approach.	20d
7	Improving detection approach	20d
8	Redaction: publications and IRCAM report.	5d

FIGURE B.2: Grantt Diagram Titles

Appendix C

Used databases

Training databases (for training patterns and for thresholds) and testing databases were designed from the three main existent databases:

- *ENST Drums*[21]: database of recorded audio files. Free distribution for research purposes.
- *RWC Music Database*[22]: database of polyphonic synthetic (MIDI origin) audio files. Free distribution for research purposes
- *Vienna Symphonic Library*: recorded audio files. 160 EUR for a full percussion data-set.

The goal designing that data-set is to obtain an small database that is representative enough of our targets. We try to design an small database because we would like to avoid long processing times.

The databases could be divided in 3 blocs: training patterns data-set, training thresholds data-sets and test data-sets.

C.1 Training patterns data-set

This (**Db1**) training set is conformed by isolated mono sounds of each of the targets that we would like to identify in the mix:

- Kick: 64 isolated sounds (improved database is of 64 isolated sounds).
- Low tom: 100 isolated sounds (improved database is of 100 isolated sounds).
- Mid tom: 95 isolated sounds (improved database is of 95 isolated sounds).
- High tom: 95 isolated sounds (improved database is of 95 isolated sounds).
- Closed hi-hat: 66 isolated sounds (improved database is of 66 isolated sounds).
- Open hi-hat: 45 isolated sounds (improved database is of 45 isolated sounds).
- Ride cymbal: 60 isolated sounds (improved database is of 60 isolated sounds).
- Chinese cymbal: 10 isolated sounds (improved database is of 10 isolated sounds).
- Crash cymbal: 58 isolated sounds (improved database is of 82 isolated sounds).
- Splash cymbal: 277 isolated sounds (improved database is of 68 isolated sounds).
- Cross-stick: 2 isolated sounds (with the improved database this class is removed).
- Cowbell: 12 isolated sounds (improved database is of 12 isolated sounds).
- Snare drum: 160 isolated sounds (improved database is of 89 isolated sounds).

Origin: Vienna Symphonic Library and ENST drums.

C.2 Training thresholds data-sets

Each of those data-sets should contain all the targets trained at the training patterns step to be able to train each target threshold. Three data-sets are conformed:

Db2: Only drums mixes: 26 files, that represents 11min 37sec recorded minutes of audio. Origin: ENST Drums.

Db3: Polyphonic mixes: 4 files, that represents 16min 61sec of recorded audio. Origin: RWC Music Database.

Db4: Only drums mixes and polyphonic mixes: 25 files for only drums that represents 7min 10sec of recorded audio and 3 files for only drums that represents 13min 29sec of recorded audio. Origin: ENST Drums and RWC Music Database.

C.3 Test data-sets

Three data-sets are conformed. Two data-sets are different and the third is conformed by the two first ones:

Db5: Only drums mixes and polyphonic mixes: 25 files, that represents 19min 40sec of recorded audio. Origin: ENST Drums and RWC Music Database.

Db6: Only drums mixes: 22 files, that represents 4min 55sec of recorded audio. Origin: ENST Drums.

Db7: Polyphonic mixes: 3 files, that represents 14min 45sec of recorded audio. Origin: RWC Music Database.

DbX are the names we are going to use throughout this PFG to refer to those data-sets.

Bibliography

[1] D.D.Lee and H.S.Seung. Algorithms for non-negative matrix factorization. *Neural Information Processing Systems*, pages 556–562, 2001. URL http://hebb.mit.edu/people/seung/papers/nmfconverge.pdf.

[2] P. Smaragdis. Non-negative matrix factor deconvolution, extraction of multiple sound sources from monophonic inputs. *International Symposium on Independent Component Analysis and Blind Source Separation (ICA) 3195 (2004) 494*. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.60.4285&rep=rep1&type=pdf.

[3] ISMIR MIREX Contest Results of Audio Drum Detection, 2005. URL http://www.music-ir.org/evaluation/mirex-results/audio-drum/index.html.

[4] Antoine Bonnefoy. Transcription automatic de la partie percussive d'un morceau de musique. *Master Thesis, IRCAM-ATIAM Master*, 2012.

[5] A. Baker A. Roebel Y. Mitsufuji, M. Liuni. Online non-negative tensor deconvolution for source detection in 3dtv audio. In *2014 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

[6] Koen Tanghe, Sven Degroeve, and Bernard De Baets. An algorithm for detecting and labelling drum events in polyphonic music. *Proceedings of the 1st Annual Music Information Retrieval Evaluation Exchange*, pages 11–15, 2005. URL http://www.music-ir.org/mirex/abstracts/2005/tanghe.pdf.

[7] Jouni Paulus. Drum transcription from polyphonic music with instrument-wise hidden markov models. URL http://www.cs.tut.fi/sgn/arg/paulus/mirex05_paulus_ext.pdf.

[8] Kazuyoshi Yoshii, Masataka Goto, and Hiroshi G Okuno. Adamast: A drum sound recognizer based on adaptation and matching of spectrogram templates. *1st Annual Music Information Retrieval Evaluation eXchange (MIREX)*, 2005. URL https://staff.aist.go.jp/k.yoshii/papers/mirex-2005-yoshii.pdf.

[9] Michael A Casey and Alex Westner. Separation of mixed audio sources by independent subspace analysis. In *Proceedings of the International Computer Music Conference*, pages 154–161, 2000.

[10] Patrik O Hoyer. Non-negative sparse coding. In *Neural Networks for Signal Processing, 2002. Proceedings of the 2002 12th IEEE Workshop on*, pages 557–565. IEEE, 2002. URL http://arxiv.org/pdf/cs/0202009.pdf?origin=publication_detail.

[11] Derry FitzGerald, Robert Lawlor, and Eugene Coyle. Prior subspace analysis for drum transcription. In *Audio Engineering Society Convention 114*. Audio Engineering Society, 2003. URL http://arrow.dit.ie/cgi/viewcontent.cgi?article=1033&context=argcon.

[12] Skot McDonald Henry Lindsay-Smith and Mark Sandler. Drumkit transcription via convolutive nmf. URL http://dafx12.york.ac.uk/papers/dafx12_submission_39.pdf.

[13] Christian Dittmar. Drum detection from polyphonic audio via detailed analysis of the time frequency domain. *1st Music Information Retrieval Evaluation eXchange (MIREX)*, 2005. URL http://www.music-ir.org/evaluation/mirex-results/articles/all/dittmar.pdf.

[14] Olivier Gillet, Gaël Richard, and GET-TELECOM Paris. Drum event detection by noise subspace projection and classification. URL http://www.music-ir.org/mirex/abstracts/2005/gillet.pdf.

[15] P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environ-metrix, 5:111-126*, 1994.

[16] Paris Smaragdis and Judith C. Brown. Non–negative matrix factorization for polyphonic music transcription. *Applications of Signal Processing to Audio and Acoustics*, 2003. URL http://www.merl.com/publications/docs/TR2003-139.pdf.

[17] Cédric Févotte and Jérôme Idier. Algorithms for nonnegative matrix factorization with the β-divergence. *Neural Computation*, 23(9):2421–2456, 2011. URL http://arxiv.org/pdf/1010.1763.

[18] Cédric Févotte. Itakura-saito nonnegative factorizations of the power spectrogram for music signal decomposition. *Machine Audition: Principles, Algorithms and Systems*, pages 266–296, 2011. URL http://www.unice.fr/cfevotte/publications/chapters/isnmf.pdf.

[19] Mathias Rossignol Arshia Cont Gregoire Lafay, Mathieu Lagrange. Unsupervised event detection in acoustic scenes using bregman divergences. In *MLSP 2014 (Machine Learning for Signal Processing)*.

[20] Axel Roebel. A new approach to transient processing in the phase vocoder. In *Proc. of the 6th Int. Conf. on Digital Audio Effects (DAFx03)*, pages 344–349, 2003. URL http://recherche.ircam.fr/equipes/analyse-synthese/roebel/paper/dafx2003.pdf.gz.

[21] Olivier Gillet and Gaël Richard. Enst-drums: an extensive audio-visual database for drum signals processing. In *ISMIR*, pages 156–159, 2006. URL http://ismir2006.ismir.net/PAPERS/ISMIR0627_Paper.pdf.

[22] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. Rwc music database: Music genre database and musical instrument sound database. In *ISMIR*, volume 3, pages 229–230, 2003. URL https://staff.aist.go.jp/m.goto/PAPER/ISMIR2003rwcmdbPOSTERgoto.pdf.