Méthodes d'apprentissage automatique pour la transcription automatique de la batterie

Céline Jacques

Thèse de doctorat de
Sorbonne Universités

Spécialité
Informatique

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Céline JACQUES

Pour obtenir le grade de
DOCTEUR de SORBONNE UNIVERSITÉ

Sujet de la thèse :

Méthodes d'apprentissage automatique pour la transcription automatique de la batterie

soutenue le 05 Avril 2019

devant le jury composé de :

M Sylvain Marchand	Rapporteur
M Gaël Richard	Rapporteur
M Carlos Agon	Examinateur
M Bertrand David	Examinateur
Mme Jimena Royo-Letelier	Examinatrice
M Emmanuel Vincent	Examinateur
M Axel Röbel	Directeur

Remerciements

Bien que ce paragraphe débute le manuscrit suivant, il est aussi une façon de conclure ces trois dernières années de recherche.

Je souhaiterais tout d'abord remercier mon directeur de thèse Axel Röbel pour m'avoir donné l'opportunité de réaliser cette thèse et de m'avoir encadrée pendant ces dernières années. Sa disponibilité, nos échanges et ses nombreux conseils et propositions m'ont permis d'aller au bout de ce défi.

J'ai passé ces trois dernières années au sein d'une équipe dynamique avec qui j'ai pu échanger dans des moments de doutes ou de perdition et qui m'a apporté énormément de connaissances et d'expertises durant mes recherches. Merci à Achille pour son travail durant son stage et notre collaboration pour MIREX. Je remercie particulièrement Guillaume pour nos discussions de "lève-tôt" sur des problèmes scientifiques et des sujets plus légers. Je souhaite aussi remercier les doctorants et ex-doctorants de l'équipe mais aussi de l'IRCAM pour leur bonne humeur et leur soutien. Un immense merci à l'équipe des "vrais", du verre du vendredi : Alice, Damien, Hugo et Tristan. Sans eux, cette aventure n'aurait pas été la même.

Mais une thèse ne s'arrête pas à la vie au labo et il me paraît essentiel de remercier les personnes qui m'ont toujours soutenue et qui ont réussi à me changer les idées durant ces années intenses. Tout d'abord merci à mes deux amies de toujours, Laure et Lorraine ! Merci aux B&B (Elsa, Laure, Léonie, Line et Sarah), toujours aussi drôles et raffraichissantes ! Merci au groupe de soutien des Doc de Choc (Jérôme et Line), aux "mecs" (Ftti, Viking, Tiguidou, Raph, Matthieu et Dadou) et à la team du Bouillon Belge. Je souhaite aussi remercier chaleuresement Raphaëlle, Jean-François et Béa. Et évidemment je remercie toutes les personnes que je n'ai pas pu citer ici.

Enfin, merci du fond du coeur aux deux personnes sans qui tout ça n'aurait pas été possible : mes parents. J'en profite pour remercier mon frère ainsi que toute ma famille pour n'avoir jamais douté de moi. Mes derniers remerciements vont à Bart de m'avoir toujours épaulée et soutenue tout ce temps.

Résumé

Résumé

La transcription automatique de la batterie s'inscrit dans le domaine plus large de la transcription de la musique. Les recherches exposées dans ce document se concentrent sur les méthodes d'apprentissage pour la transcription automatique de la batterie. Elles sont basées sur un algorithme de transcription utilisant une méthode de décomposition non-négative, la NMD, pour décomposer la partie de batterie grâce à des motifs préalablement appris autour des événements musicaux détectés par un autre algorithme.

Cette thèse soulève deux principales problématiques : l'adaptation des méthodes de décomposition non-négative pour prendre en compte des informations issues du signal analysé et l'utilisation de l'apprentissage profond pour la transcription de la batterie. Les réseaux de neurones peuvent permettre la spécialisation d'un algorithme de détection d'*onsets* basé sur un réseau de neurones convolutif aux *onsets* percussifs pour mieux informer l'algorithme de décomposition.

La prise en compte des informations du signal analysé dans le modèle peut être réalisée par leur introduction durant les étapes de décomposition. Une première approche est de reformuler l'étape de décomposition dans un contexte probabiliste avec des méthodes de décomposition telles que la SI-PLCA et la NMD statistique. Cette reformulation pourrait rendre l'adaptation plus simple. Une deuxième approche est d'implémenter directement dans la NMD une stratégie d'adaptation. Ainsi, l'application de filtres modelables aux motifs permet de modéliser la différence entre les conditions d'enregistrement. Une autre possibilité est d'adapter les motifs directement au signal mais en appliquant de fortes contraintes pour qu'ils gardent leur signification physique.

La deuxième approche porte sur la sélection des segments de signaux à analyser et donc des *onsets* à détecter. Il est préférable d'analyser les segments où au moins un événement percussif a lieu. L'émergence de l'apprentissage profond pour la tâche de détection d'*onsets* dans la littérature ces dernières années et les résultats prometteurs obtenus amènent à adapter un détecteur d'*onsets* basé sur un réseau de neurones convolutif (CNN) à détecter uniquement les *onsets* percussifs. Les résultats obtenus étant très intéressants, la spécialisation du détecteur d'*onsets* est approfondie. Le détecteur est entraîné à ne détecter qu'un seul instrument permettant la réalisation de la transcription des trois principaux instruments de batterie avec trois CNN. Finalement, l'utilisation d'un réseau de neurones convolutif multi-sorties est étudiée pour transcrire la partie de batterie avec un seul réseau.

Abstract

Automatic drum transcription is a part of a larger field, automatic music transcription. The researches in this document focus on learning methods applied to automatic drum transcription. They are based on an existing algorithm. This algorithm decomposes with NMD the signal of segments around reviously detected musical events.

This thesis raises two main issues: the adaptation of non-negative decomposition methods to take into account information from the analyzed signal and the use of deep learning for the transcription of the drums. Neural networks can allow the specialization of an onset detection algorithm based on a convolutional neuron network with percussive onsets to better inform the decomposition algorithm.

Taking into account the information of the analyzed signal in the model can be achieved by their introduction during the decomposition steps. A first approach is to reformulate the decomposition step in a probabilistic context with decomposition methods such as SI-PLCA and statistical NMD. This reformulation could make adaptation easier. A second approach is to implement an adaptation strategy directly in the NMD. Thus, the application of modelable filters to the patterns makes it possible to model the difference between the recording conditions. Another possibility is to adapt the patterns directly to the signal but by applying strong constraints so that they keep their physical meaning.

The second approach concerns the selection of the signal segments to be analyzed and therefore the onsets to be detected. It is best to analyze segments where at least one percussive event occurs. The emergence of deep learning for the task of onset detection in the literature in recent years and the promising results obtained lead to adapting an onset detector based on convolutional neural network (CNN) to detect only the percussive onsets. The results obtained are very interesting and lead to try to detect more specfic onsets. The detector is trained to detect only one instrument allowing the transcription of the three main drums with three CNNs. Finally, the use of a multi-output convolutional neural network is studied to transcribe the drum part with a single network.

Table des matières

Bibliographie **137**

Liste des figures

Liste des tableaux

Première partie

Introduction à la transcription automatique de la batterie

— Chapitre 1 —

Introduction

Contexte

La transcription de la musique s'inscrit dans le domaine plus vaste de la recherche d'informations musicales, *Music Information Retrieval* (MIR), et consiste à décrire symboliquement un morceau de musique à partir de ce que l'on entend.

À l'instar du traitement de la parole où l'on cherche à reconnaître les mots et les phrases prononcés, la transcription de la musique a pour but d'estimer à partir de la forme d'onde du morceau les notes jouées et leurs paramètres : hauteurs, durées, instants de l'attaque, à ces paramètres on peut ajouter des informations de plus haut niveau telles que les séquences rythmiques, la mesure, l'armure, l'instrumentation, etc.

Jusqu'au milieu du XX^{e} siècle, la tâche de transcription était réservée à l'expertise humaine. C'est une tâche difficile nécessitant un long apprentissage. Lors de la formation des musiciens, ceux-ci sont d'ailleurs très vite confrontés à l'exercice de la dictée musicale. Commençant par des motifs simples composés de quelques notes jouées séparément et sans rythme déterminé, les dictées se complexifient au fur et à mesure par l'introduction de motifs rythmiques, de polyphonie et/ou d'instruments différents.

Cependant, la musique a toujours intéressé les scientifiques. Déjà au IV^{e} av. J.C., Pythagore pense que la musique est liée aux mathématiques. Il découvre une relation entre la longueur de la corde tendue et la hauteur du son émis lorsqu'on l'a fait vibrer. Il crée d'ailleurs son propre instrument, le monocorde. Grâce à lui, il élabore la base de ce qu'on appellera la gamme pythagoricienne, gamme basée sur une suite de quintes.

Avec l'émergence de la pratique musicale et des techniques d'enregistrement, la volonté de transcrire la musique prend tout son sens. Ainsi la transcription de la musique devient un nouveau défi pour les scientifiques. Les recherches sur la représentation des sons et sur l'extraction d'informations utiles par des programmes informatiques permettent aujourd'hui de s'approcher d'un système de transcription automatique de la musique, c'est-à-dire sans intervention de l'oreille humaine.

En règle générale, il est plus facile de manipuler des symboles que des sons brutes. Cela rend plus aisées diverses applications dont la transcription automatique de la musique pourrait être une étape préalable comme l'aide à l'apprentissage de la musique, la

protection des droits d'auteur, la recommandation musicale ou encore l'indexation d'une bibliothèque musicale.

Approches et contributions

La transcription automatique de la musique vue comme l'écriture d'une partition à partir d'un morceau de musique est une tâche extrêmement difficile et nécessite l'application de plusieurs algorithmes de MIR. Dans cette thèse, nous appellerons transcription la tâche de description du morceau par ces différents paramètres : l'instant d'apparition des notes, leurs hauteurs et l'instrument qui les a jouées.

Toujours par souci de simplification, la transcription de la musique peut être subdivisée en différents objectifs suivant les événements à décrire (partie harmonique/ partie percussive) ou suivant les tâches à accomplir (détection du début de la note, estimation de la fréquence fondamentale, identification de l'instrument).

Dans cette thèse, nous nous concentrons sur la transcription de la partie percussive des musiques occidentales et plus particulièrement sur la transcription des trois principaux instruments de la batterie (la grosse-caisse, la caisse-claire et la Charleston) qui sont responsables d'une grande majorité des motifs rythmiques de la musique occidentale. Plusieurs méthodes basées sur une étape préalable d'apprentissage sont étudiées allant de la décomposition non-négative des spectrogrammes grâce à des motifs définis jusqu'à l'apprentissage profond grâce à des réseaux de neurones artificiels qui extraient des caractéristiques pour chaque instrument.

Les méthodes de décomposition sont basées sur l'apprentissage de motifs représentant un instrument. Le côté déterministe de ces motifs restreint leur possibilité de généralisation à des morceaux dont les instruments s'éloignent des motifs appris. Comment choisir et adapter les méthodes de décomposition non-négative pour les rendre plus robustes à la diversité des instruments dans la musique ?

Ces méthodes sont renseignées par les positions des débuts de note déterminées par un autre algorithme. Cet algorithme détecte tous les événements présents dans le morceau qu'ils soient déclenchés par un instrument dit mélodique ou par un instrument percussif. Renseigner l'algorithme de décomposition non-négative avec seulement les positions des événements percussifs permettrait-il d'améliorer la transcription de la batterie ? Comment adapter un détecteur d'*onsets* (début de note) pour qu'il ne détecte que les *onsets* percussifs voir les *onsets* d'un unique instrument ?

Dans cette thèse, nous proposons une étude de différentes méthodes de décomposition non-négative : une déterministe, la NMD, et deux probabilistes, la *Shift-Invariant Probabilistic Latent Componant Analysis* (SI-PLCA) et la NMD statistique avec la divergence d'Itakura-Saïto (IS) (IS-NMD/EM) que nous avons adaptée aux motifs temporels (à deux dimensions) à partir de la *Non negative Matrix Factorisation* (NMF) statistique avec la divergence d'IS (IS-NMF/EM). Théoriquement, les méthodes probabilistes per-

mettent l'introduction d'informations a posteriori rendant possible la prise en compte des caractéristiques du morceau analysé.

L'introduction d'informations issues du morceau à transcrire peut aussi être réalisée en adaptant les motifs au signal. Nous introduisons tout d'abord des filtres appliqués aux motifs permettant de représenter la réponse des microphones et de la salle d'enregistrement et ainsi d'arriver à prendre en compte des différences entre l'instrument du morceau et l'instrument utilisé pour apprendre les motifs, dues aux conditions d'enregistrement différentes. Ces filtres sont adaptés avec les informations issues du signal pour améliorer la généralisation du dictionnaire des motifs aux morceaux analysés. Nous montrons que cette méthode ne permet pas la convergence de la fonction de coût qui mesure la distance entre le spectrogramme cible et le spectrogramme estimé.

La deuxième approche que nous proposons est d'adapter directement les motifs au signal. Cependant, la modification des motifs eux-mêmes pendant la décomposition peut leur faire perdre leur sens physique. Un motif d'un instrument peut ne plus décrire cet instrument après modification. Nous proposons différentes manières de contraindre les mises à jour des motifs afin qu'ils conservent leur signification physique.

Pour adapter les motifs avec les informations les plus pertinentes, il est intéressant de n'utiliser que les informations relatives à l'instrument ciblé présentes dans le morceau analysé. Pour cela, nous nous intéressons à la spécialisation d'un détecteur d'*onsets* aux *onsets* percussifs que nous combinons avec la méthode de décomposition non-négative.

Motivé par des résultats encourageants obtenus avec le détecteur d'*onsets* spécifiques basés sur les réseaux de neurones artificiels, nous allons même plus loin en l'entraînant à détecter uniquement les *onsets* d'un seul instrument de la batterie. Le problème du manque de données d'apprentissage se pose alors et nous étudions l'influence de différentes méthodes d'augmentation des données sur la généralisation du détecteur d'*onsets*. Enfin, nous évaluons l'utilisation d'un seul réseau de neurones à plusieurs sorties pour effectuer la transcription de la batterie.

Publications

Articles publiés

Jacques, C. and Röbel, A. (2018). Automatic drum transcription with convolutional neural networks. *Proceedings of the International Conference on Digital Audio Effects (DAFx)*

Jacques, C., Aknin, A., and Röbel, A. (2018). Mirex 2018 : Automatic drum transcritpion with convolutional neural networks. *MIREX*

Articles soumis

À EUSIPCO 2019 :
Jacques, C. and Röbel, A. (2019). Data augmentation for onset detection and drum transcription (submitted) `https://arxiv.org/abs/1903.01416`

Organisation du document

Après avoir introduit les enjeux de la transcription de la batterie, ce document est organisé en deux parties. La première partie présente les différents outils utilisés pour nos recherches. La deuxième partie présente comment ces outils sont utilisés et comment nous les adaptons pour répondre aux différentes problématiques de cette thèse : l'adaptation des motifs utilisés par les méthodes de décomposition non-négative et la détection d'*onsets* spécifiques pour informer l'algorithme de transcription ou réaliser la transcription de la batterie.

***Première partie** : Des outils pour la transcription automatique de la batterie*

Dans le Chapitre 3, nous présentons les méthodes de décomposition non-négative suivantes : la NMF et sa variante avec motifs temporels la NMD, la *Probabilistic Latent Componant Analysis* (PLCA) et sa variante la SI-PLCA et enfin la IS-NMF/EM que nous adaptons aux motifs temporels avec la IS-NMD/EM. Dans le Chapitre 4, nous présentons le principe des réseaux de neurones ainsi que quelques architectures principales.

***Deuxième partie** : Applications à la transcription automatique de la batterie*

Le Chapitre 5 compare les différentes méthodes de décomposition non-négative présentées au Chapitre 3 afin de retenir la méthode permettant d'obtenir les meilleurs résultats.

Le Chapitre 6 présente différentes manières de prendre en compte les différences entre les instruments du morceau analysé et les motifs du dictionnaire : l'utilisation de filtres adaptables appliqués aux motifs ou l'adaptation contrainte des motifs afin qu'ils gardent leur signification physique après modifications.

Les Chapitres 7 et 8 sont basés sur l'utilisation des réseaux de neurones artificiels. Dans le Chapitre 7, un détecteur d'*onsets* est modifié pour ne détecter que les *onsets* percussifs. Ce détecteur est combiné à la méthode de décomposition non-négative retenue au Chapitre 5. Le Chapitre 8 utilise les réseaux de neurones convolutifs pour la transcription de la batterie. Trois réseaux indépendants sont d'abord entraînés à ne détecter les *onsets* que d'un seul instrument et pour améliorer leur généralisation, différentes méthodes d'augmentation de données sont comparées. Et enfin, un réseau de neurones multi-sorties permet de réaliser la transcription de la batterie.

— Chapitre 2 —

La transcription automatique de la batterie

2.1 Introduction à la transcription de la batterie

2.1.1 La batterie et ses défis

2.1.1.1 L'instrument

La batterie ou plus largement les percussions jouent un grand rôle dans les musiques occidentales, notamment pour certains genres tels que la pop, le jazz ou le rock. Ils sont les principaux acteurs de la rythmique d'un morceau de musique et ils établissent la structure du morceau en différentes parties.

Les sons percussifs sont très différents des sons dits mélodiques. Un son percussif est représenté par un impact, occupant une large bande de fréquence et dont l'énergie se dissipe rapidement. Au contraire, un instrument mélodique va produire une note dont le spectrogramme est plus parcimonieux et dont la durée dépend du musicien et de la partition. Quelques exceptions existent. Par exemple, le xylophone considéré comme instrument percussif produit lui aussi des notes.

La batterie est un ensemble d'instruments percussifs dont la plupart peut être classés en deux familles. Les membranophones sont constitués d'une membrane vibrante tendue sur un support cylindrique. On peut notamment citer les toms (aigü, medium et grave), la caisse-claire et la grosse-caisse. La deuxième famille est constituée des idiophones. Ces instruments au corps métallique vibrent comme un tout tel les cymbales (*ride* et *crash*) ou le Charleston (*hi-hat*). La figure 2.1 [1] présente les différents instruments d'une batterie classique.

Pour les tâches de transcription, il est commun de ne considérer que trois des instruments de la batterie. Ces trois instruments sont le socle de la plupart des rythmes de bases. La grosse-caisse, *Bass Drum* (BD) en anglais, est jouée au pied. Son énergie est localisée dans les basses fréquences. La caisse-claire, *Snare Drum* (SD) en anglais, est la contrepartie rythmique de la grosse-caisse. Elle est constituée de deux peaux tendues sur un cylindre. Sur la peau du dessous est fixé un timbre, caractéristique de la caisse-claire, qui gratte la peau vibrante obtenant ainsi un son craquant. Enfin, la Charleston, *Hi-Hat*

1. `http://batterie.poumtchac.com/presentation.html`

(HH) en anglais, est une double cymbale. Elle peut être jouée ouverte, produisant un son raisonnant, ou fermée, ce qui donne un son plus court. L'énergie qu'elle produit est localisée dans les hautes fréquences.

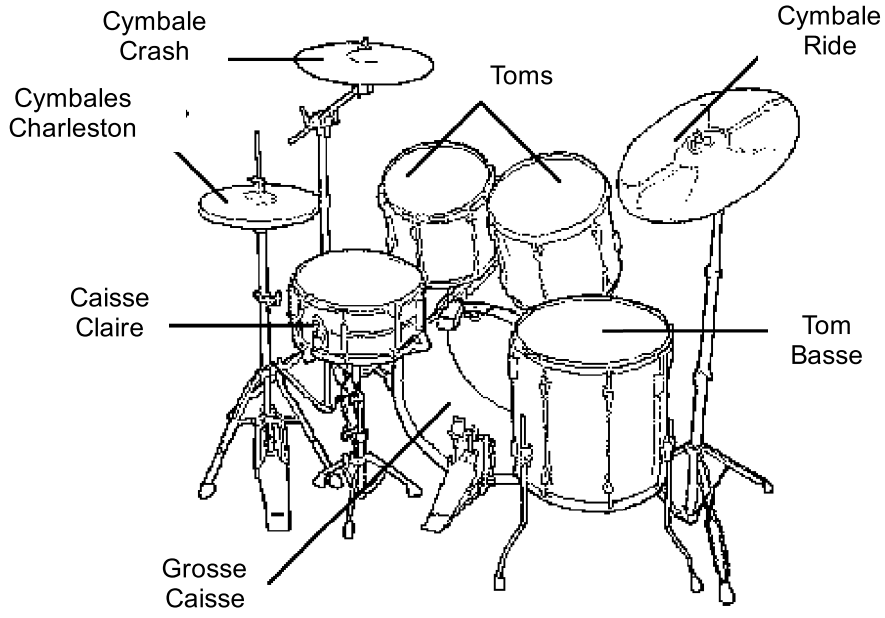

FIGURE 2.1 – *Les différents instruments de la batterie.*

Une batterie acoustique offre différents sons suivant l'emplacement de la frappe sur les différents éléments, la vélocité de la frappe ou même suivant le matériel. Ces variétés de sons ne sont pas possibles pour les échantillons numériques. Un élément de la batterie est souvent représenté par le même son. Pour pallier à cela, il est courant d'utiliser différentes banques de son pour générer des morceaux de batterie à partir de fichier MIDI.

2.1.1.2 Ses particularités et ses défis

La grande particularité des instruments de batterie réside dans le fait que les sons produits ne sont pas tonales. Ils couvrent une large bande de fréquence et ne présentent pas d'harmoniques. Il existe des instruments percussifs qui produisent des notes mais leurs résonnances ne suivent pas les mêmes règles qui régissent celles des instruments dits harmoniques ou mélodiques. La distribution de ces résonnances ne permet pas, par exemple, d'obtenir des résultats avec des algorithmes d'estimation de fréquence fondamentale.

Cependant, malgré la diversité des techniques de jeu et des batteries, on retrouve des caractéristiques communes. L'attaque, le relâchement et la distribution de l'énergie sont très semblables d'une grosse-caisse à une autre par exemple. Une approche par recherche ou identification de motifs (*pattern*) est donc possible. De plus, la batterie joue très souvent de manière régulière et répète des séquences semblables. La recherche

de séquences rythmiques peut s'entreprendre ainsi que des estimations des événements grâce par exemple à des modèles probabilistes.

Dans la musique occidentale, les différents instruments de batterie jouent très souvent ensembles créant alors des superpositions en temps et en fréquence. Ces superpositions rendent difficile l'identification des instruments impliqués dans un événements. Cela est encore plus difficile en cas de présence d'autres instruments non-percussifs.

2.1.2 Bases de données

Plusieurs bases de données sont utilisées pour évaluer les différents algorithmes qui sont présentés dans ce manuscrit.

2.1.2.1 RWC

La base de données *Real World Computing* (RWC) est une base de données musicale élaborée pour la recherche (Goto et al., 2002). Elle est composée de quatre sous-ensembles de morceaux originaux : *Popular Music Database*, *Royalty-Free Music Database*, *Classical Music Database* et *Jazz Music Database*. Deux nouvelles bases de données ont été ajoutées par la suite (Goto, 2003) : *Music Genre Database* et *Musical Instrument Sound Database*.

Les morceaux sont polyphoniques et sont fournis avec les annotations de tous les instruments présents sous format MIDI permettant de les resynthétiser.

- La base de données *Popular Music Database* contient 100 morceaux, 20 chansons en anglais du style des chansons pop du classement des titres américains des années 80 et 80 chansons japonaises dans un style de pop japonaise des années 90. Ces chansons ont été enregistrées spécialement pour cette base de données.
- Le sous-ensemble *Classical Music Database* est constitué de 4 symphonies, 2 concerto, 4 pièces orchestrales, 10 pièces de musique de chambre, 24 solo et 6 chants. Cela représente donc 50 pièces du domaine publique.
- La base de données *Royalty-Free Music Database* présente 15 chansons : 10 standards populaires écrites en anglais et 5 chansons pour enfant japonaises. Ce sous-ensemble a été généré pour fournir des chansons bien connues en plus des musiques spécialement composées et enregistrées pour la base de données *Popular Music Database*.
- La base de données *Jazz Music Database* contient 5 morceaux jouées avec 7 compositions instrumentales différentes, soit 35 pièces, 2 morceaux de jazz vocal, 2 jouées par un big-band, 2 de jazz modal, 2 de jazz funky et 1 de *Free jazz*. Enfin 6 pièces de jazz fusion sont ajoutées pour obtenir un corpus de 50 pièces. Tous les morceaux ont été composés et enregistrés pour la base de données.
- La base de données *Music Genre Database* est décomposée en 10 catégories de genre (populaire, rock, dance, jazz, latin, classique, marche, musique du monde, musique traditionnelle japonaise et voix). Les morceaux de musiques de *Music Genre Database* sont issus de compositions originales ainsi que de base de données existantes.

- L'ensemble *Musical Instrument Sound Database* couvre 50 instruments et fournit en principe trois variations pour chaque instrument.

2.1.2.2 ENST-Drums dataset

La base de données ENST-Drums a été élaborée pour la recherche sur la transcription automatique de la batterie. Elle contient environ 75 min de sons. Trois batteurs professionnels spécialisés sur des styles différents, ont joué sur leur propre batterie. Pour accroître la diversité des enregistrements différents types de baguettes (classiques, balais, mailloches et rodins) ont été utilisés et les batteries ont des compositions différentes allant d'un petite batterie avec deux tomes et deux cymbales à des batteries rock avec quatre toms et cinq cymbales.

Chaque batteur a enregistré cinq différents types de séquences. Pour chaque séquence, les batteurs ne suivaient aucune partition et il n'a pas été demandé de jouer une séquence rythmique particulière. Elles ont été par la suite annotées de manière semi-automatique.
Hits Des séquences de coups de baguette séparés par quelques secondes de silence ont été jouées sur chaque instrument pour chaque type de baguettes disponibles.
Phrases Environ soixante séquences courtes de différents styles ont été jouées par les batteurs sans accompagnement. Une liste de styles a été présentés aux batteurs et chaque batteur a sélectionné ses favoris. Pour chaque style, six phrases sont jouées à différents tempi (lent, médium, rapide) avec deux niveaux de complexité. Le premier est simple sans ornement et le deuxième plus complexe avec des ornements.
Soli Les trois batteurs ont joué au moins cinq soli dans des styles de leur choix. Les soli devaient durer au minimum 30s, contenir tous les instruments présents sur la batterie ainsi que des séquences très complexes.
Minus-one Une vingtaine de morceaux ont été joués par les batteurs sur des accompagnements pré-enregistrés sans séquence de batterie, les *minus-one* ou des musiques générées depuis des fichiers MIDI sans la partie de batterie. Les *minus-one* couvrent différents styles avec des morceaux d'environ 1 min. Les accompagnements sont aussi disponibles dans la base de données avec deux types de mixage différents. Le premier est un mixage brut, *dry* des différentes voix. Le deuxième, *wet*, ajoute de l'égalisation et de la compression pour se rapprocher de la musique commercialisée.

2.1.2.3 Base de données d'apprentissage du challenge MIREX pour la transcription automatique de la batterie

MIREX *Music Information Retrieval Evaluation eXchange* est un ensemble de challenges dans différents domaines du MIR. En 2018, deux challenges sur la transcription de la batterie ont été proposés (`https://www.music-ir.org/mirex/wiki/2018:Drum_Transcription`) : un avec trois classes et l'autre avec huit classes.

Les deux challenges sont évalués sur cinq bases de données différentes. Deux des trois sous-ensembles de la base proposée en 2005 pour ce même challenge (CD, KT) sont utilisées comme référence. On la notera 2005. Trois autres bases de données sont

ajoutées. MEDLEY [2] contient 23 morceaux enregistrés et annotés à la main avec une double vérification. GEN est composée de séquences de batterie générées à partir de fichier MIDI. RBMA [3] est un ensemble de musiques polyphoniques produites électroniquement et annotées à la main avec double vérification.

Seule la base d'entraînement est disponible. Elle est extraite de ces 5 bases de données.

2.1.3 La tâche de transcription

Un musicien effectuant la tâche de transcription de la musique écoute un morceau et l'écrit sous forme symbolique : partition ou tablature par exemple. Il reporte d'abord notes et valeurs temporelles des notes ainsi que les éléments de percussions puis complète avec des métadonnées telles que la mesure ou les nuances.

En transcription automatique, l'ajout de ces métadonnées requiert l'utilisation d'algorithme de *Music Information Retrieval* (MIR). Ici,i la transcription consiste à détecter et classifier les événements d'un morceau de musique pour les instruments percussifs, c'est-à-dire le moment où l'événement a eu lieu et quel instrument en est responsable. Pour les instruments mélodiques, il faudrait aussi estimer les fréquences fondamentales de la ou des notes qui ont été jouées.

2.1.4 Mesure d'évaluation des algorithmes

Bien que des méthodes plus perfectionnées et perceptivement motivées sont présentent dans la littérature (Emiya, 2008), la mesure de performance, ici, se limite aux mesures basées sur le dénombrement des événements correctement détectés, non détectés et des événements détectés en trop.

Pour calculer les différentes mesures de performance, il faut d'abord déterminer l'ensemble des événements correctement détectés, les *True Positive* (TP). Les événements qui ont été détectés mais qui sont absents de la référence sont les *False Positive* (FP). On désigne par *False Negative* (FN) les événements qui sont présents dans les données mais qui n'ont pas été détectés. Et enfin, les *True Negative* (TN) sont les événements qui n'ont pas été détectés et qui n'étaient pas supposés être détectés.

Pour la transcription de la batterie, l'événement est considéré comme correctement détecté s'il apparaît dans un intervalle de temps autour de la référence. Plusieurs intervalles sont envisagés dans la littérature : 50ms pour (Bello et al., 2006) et (Vincent et al., 2008), 70ms pour (Dixon, 2000) ou encore 30ms au challenge MIREX 2018.

Basées sur ces ensembles, plusieurs mesures de la performance peuvent être définies. La précision $\mathcal{P}$ mesure sur l'ensemble des événements détectés la part qui a été correctement détectée. Le rappel $\mathcal{R}$, *recall* en anglais, rend compte du nombre d'événements correctement détectés sur l'ensemble des événements présents dans la référence.

$$\mathcal{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \quad \mathcal{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{2.1}$$

2. `http://medleydb.weebly.com`
3. `https://rbma.bandcamp.com/album`

Ces deux critères peuvent être résumés par une seule grandeur, la F-mesure $\mathcal{F}$ qui est la moyenne harmonique entre les deux mesures et permet de faire un compromis entre ces dernières.

$$\mathcal{F} = \frac{2\mathcal{P}\mathcal{R}}{\mathcal{P} + \mathcal{R}} \tag{2.2}$$

2.1.5 Applications

La transcription automatique de la batterie peut s'utiliser pour diverses applications.

Education musicale

Il existe deux types de batterie : les batteries acoustiques et les batteries électroniques. Les premières sont composées des différents éléments décrits au paragraphe 2.1.1.1. Les secondes sont constituées de *pads* en caoutchouc ou en fausse peau. Les sons sont générés électroniquement à partir d'échantillons et peuvent être écoutés sur des enceintes. Certaines batteries électroniques permettent de mesurer les performances du batteur. Cela n'est pas possible sur les batteries acoustiques. La transcription appliquée aux batteries acoustiques pourrait permettre des mesures de performances similaires à celles des batteries électroniques.

Production musicale

Les enregistrements de batterie utilisent en général beaucoup de microphones. Des étapes de traitements sont donc souvent nécessaires pour obtenir le rendu sonore souhaité ainsi qu'un mixage entre les pistes et une étape d'égalisation. Il arrive parfois que le son de la batterie soit même remplacé. La transcription automatique de la batterie pourrait permettre de ne pas utiliser autant de micro et de remplacer plus facilement le son de la batterie par un autre son. De plus, les *samples*, c'est-à-dire des petites phrases musicales que l'on peut réutiliser, peuvent être extraites facilement.

Music Information Retrieval

La transcription automatique est un bon prétraitement pour obtenir des informations de plus haut niveau comme la structure d'un morceau, son tempo, les séquences rythmiques. Cela permet aussi de faciliter la recommandation musicale. Certains motifs rythmiques sont caractéristiques de genre musicaux. Si une personne écoute un morceau d'un certain style, il y a de fortes chances qu'il apprécie un morceau ayant la même structure et le même motif rythmique.

2.2 Un classement des méthodes de transcription

On trouve dans la littérature beaucoup d'algorithmes réalisant la transcription automatique de la batterie. Ces algorithmes sont basés sur différentes méthodes. D.FitzGerald et J.Paulus proposent dans (Springer, 2006) une répartition des différentes approches en quatre classes :

1. *Segmenter et classifier* - À l'aide de connaissances a priori le signal est découpé en segment autour des événements intéressants pour la transcription. Par exemple, une détection d'*onsets*, événements musicaux grossière peut avoir été appliquée.

Puis le signal est analysé sur ce segment et les différents intervenants sont classifiés et identifiés.

2. *Séparer et détecter* - Le signal peut être vu comme une superposition de plusieurs voix, chacune correspondant à un instrument. Chaque voix est isolée puis une détection d'événements est appliquée permettant la transcription de la voix cible.
3. *Correspondance et adaptation* - Comme décrit dans la section 2.1.1.2, les caractéristiques des sons de batterie permettent l'utilisation de motifs ou *patterns*. Dans les méthodes dites de "correspondance et d'adaptation", ces motifs sont recherchés dans le morceau et une fois qu'ils ont été trouvés, ils sont adaptés pour mieux rendre compte de l'instrument utilisé dans le morceau. En effet, les motifs sont appris sur des données d'entraînement. Ces données ne peuvent pas représenter tous les différents instruments. Il est donc peu probable que la batterie utilisée pour le morceau à transcrire soit une de celles utilisées pour l'apprentissage des motifs.
4. *Les modèles de Markov cachés, Hidden Markov Model* (HMM) - Les HMM sont des modèles probabilistes permettant de prendre en compte les relations temporelles qu'ont les événements les uns avec les autres.

Il est tout de même difficile de classifier toutes les méthodes de transcription automatique de la batterie dans ces catégories. Les frontières entre elles ne sont pas toujours très claires et certains algorithmes pourraient être classés dans plusieurs catégories. C. Wu propose dans (Wu et al., 2017) de ne plus décrire les méthodes par leurs approches mais de les diviser en briques qui sont interchangeables et combinables. Ces briques permettent de construire l'algorithme de transcription. Six briques génériques sont présentées : Représentation des caractéristiques des données, segmentations des événements, fonction d'activation, adaptation des représentations des caractéristiques, modèle de langage et classification d'événements. Avec ces six briques, plusieurs approches peuvent être construites. Elles sont présentées en détail dans (Wu et al., 2017) et sont discutées ici.

2.2.1 Quelques briques de base

2.2.1.1 Caractéristiques des données

La représentation basique d'un son est sa forme d'onde. Cette représentation ne donne pas accès à suffisamment d'informations pour les tâches de transcription. Mais il existe d'autres façons de représenter un signal sonore afin de faire ressortir certaines caractéristiques ou propriétés. Les représentations temps-fréquences (TF) sont très souvent utilisées. Elles donnent la distribution de l'énergie en fonction des bandes de fréquences et du temps. Les plus connues sont la Transformée de Fourier à court terme (TFCT) (*Short Time Fourier Transform* (STFT) en anglais), ou le *Constant Q Transform* (CQT) qui extrait des informations de bas niveau du signal. Des traitements supplémentaires peuvent être appliqués pour faire ressortir des caractéristiques encore plus particulières : filtrage, séparation des parties percussive et harmonique ou encore séparation de sources.

2.2.1.2 Segmentation à partir des événements

Le principal but de cette étape est de repérer les événements temporels avant d'analyser le signal. Cela consiste à construire une fonction qui permet de repérer les changements rapides. Par exemple, le flux spectral permet de mettre en valeur les changements d'énergie dans le signal. Les événements peuvent alors être détectés en cherchant les maxima locaux avec une méthode de *peak picking*, recherche des pics.

2.2.1.3 Fonction d'activation

Cette étape transforme les représentations caractéristiques d'un signal en une fonction dite fonction d'activation. Cette fonction indique le niveau d'activité des différents instruments. La PLCA ou la NMF sont des méthodes basées sur les fonctions d'activation ainsi que la *Independent Subspace Analysis* ou la *Prior Subspace Analysis*.

2.2.1.4 Transformation des caractéristiques

Il est parfois nécessaire de modifier la représentation des caractéristiques d'un signal pour obtenir une représentation plus compacte ou plus en adéquation avec les propriétés que l'on veut mettre en valeur. La *Principal Componant Analysis* (PCA) ou la *Linear Discriminant Analysis* (LDA) peuvent réaliser une transformation des caractéristiques ou simplement la sélection de certaines caractéristiques plutôt que d'autres.

2.2.1.5 Classification des événements

La classification des événements associe à un événement le ou les instruments responsables. Dans une grande part de la littérature, cette tâche est réalisée par des méthodes de *machine learning* comme les *Support Vector Machine* (SVM). Ces méthodes apprennent à discriminer les instruments cibles en s'appuyant sur des exemples d'apprentissage.

2.2.1.6 Modèle du langage

Les méthodes basées sur le modèle du langage prennent en compte les relations séquentielles entre les événements. Elles utilisent des modèles probabilistes qui sont capables d'apprendre la grammaire musicale puis d'inférer la structure des événements à venir, par exemple les HMM ou les *Recurrent Neural Network*, réseau de neurones récurrent (RNN).

2.2.2 Différentes combinaisons de briques

Les briques présentées dans la section précédente et résumées dans la figure 2.2 (Wu et al., 2017) peuvent être combinées, échangées, associées pour construire des algorithmes de transcription de la batterie. La liste n'étant pas exhaustive, il faut parfois ajouter des étapes de prétraitement, d'analyse ou de post-traitement non mentionnées ci-dessus.

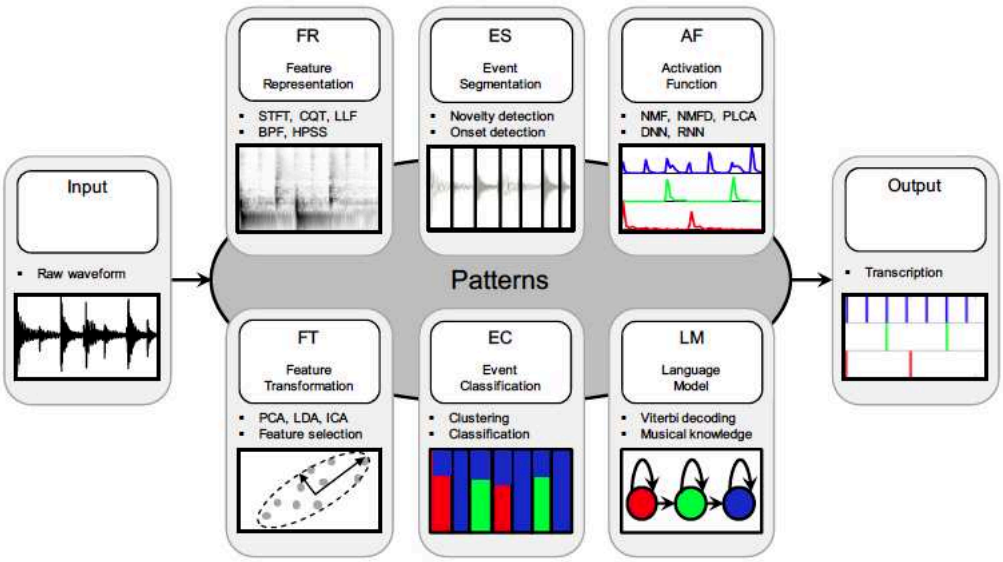

Figure 2.2 – *Les différentes briques utiles pour la transcription de la batterie.*

2.2.2.1 Méthodes basées sur la segmentation

La principale composante de ces méthodes est la segmentation des événements mais elles suivent souvent le même schéma. Tout d'abord, les propriétés des instruments sont mises en avant avec une représentation adaptée des caractéristiques. Par exemple, une séparation des parties harmoniques et percussives, HPSS *Harmonic Percussive Source Separation* (FitzGerald, 2010).

Le premier système a été élaboré par Schloss (Schloss, 1985). Il estime les enveloppes de la forme d'onde puis applique un seuil sur les pentes de l'enveloppe pour détecter les attaques.

Une autre méthode proposée dans (Tzanetakis et al., 2005) consiste à mettre en évidence les bandes de fréquences caractéristiques des différents instruments de batterie. Puis, la détection d'onsets pour chaque instrument est réalisée sur chaque enveloppe temporelle du signal des sous-bandes de fréquence. Cette méthode fonctionne si les instruments ne se superposent pas dans le domaine fréquentiel. Il est donc difficile d'augmenter le nombre d'instruments différents.

Les méthodes basées sur la segmentation présentent différents avantages : L'utilisation de la forme d'onde permet d'interpréter facilement les résultats et d'avoir un meilleur contrôle du système même pour des personnes non-expérimentées. De plus, l'application en temps réel est rendue possible avec les méthodes simples de représentation des caractéristiques ou de classification d'événements, comme la corrélation croisée ou l'utilisation de seuils. Cependant plusieurs inconvénients sont à relever : ces méthodes sont souvent peu robustes à l'ajout d'autres composantes sonores et l'utilisation de la représentation temporelle limite l'extraction de données plus détaillées.

2.2.2.2 Méthodes basées sur la classification d'événements

Les méthodes basées sur la classification présentent souvent le même schéma. Il s'agit d'abord d'extraire des représentations caractéristiques du signal audio puis de localiser les événements par une segmentation des événements pour ensuite déterminer quels instruments sont responsables de l'événement. Une étape d'adaptation et de transformation des caractéristiques peut éventuellement y être ajoutée.

Ces systèmes sont basés sur le paradigme du *pattern recognition*, reconnaissance de motif. Il existe beaucoup de systèmes différents avec un grand choix d'outils pour chaque étape. Les formes des données d'entrée les plus communes sont des combinaisons de caractéristiques spectrales (les centroïdes ou le flux spectral), de caractéristiques temporelles (*zero-crossing rate*, l'énergie locale moyenne ou les descriptions d'enveloppe) et des *Mel-Frequency Cepstral Coefficients* (MFCC). D'autres caractéristiques peuvent aussi être extraites par exemple en utilisant la NMF. Pour extraire les caractéristiques spectrales, il est très courant d'utiliser la TFCT, et ses dérivées comme la transformée à Q constant (Bello et al., 2006) et (Rossignol et al., 2015), *Line Spectral Frequencies* (Souza et al., 2015) et *Mel Scale Log Magnitude Spectrogram* (Gajhede et al., 2016).

Contrairement à l'étape de représentation des données et des caractéristiques, l'étape de transformation est optionnelle. Dans un morceau l'instrument de batterie peut jouer seul mais joue très souvent en même temps qu'un autre instrument. Il faut donc pouvoir extraire les caractéristiques propres à la batterie. Plusieurs techniques de transformation peuvent être utilisées comme la PCA (Gillet and Richard, 2005), *Information Gain Ration* (Roy et al., 2007), *Recursive Feature Elimination* (Gillet and Richard, 2008), *Correlation Based Feature Selection* (Herrera et al., 2002) et *Sparse coding matching pursuit* (Scholler and Perwins, 2010), (Scholler and Perwins, 2011).

Pour l'étape principale, celle de classification, beaucoup de méthodes ont été abordées dans la littérature. On trouve des méthodes classiques pour leur simplicité et leur interprétation facile comme *K-nearest Neighboors* (KNN) (Herrera et al., 2002, 2003, 2004; Tindale et al., 2004; Moreau and Flexer, 2007; Miron et al., 2013b). Des méthodes plus complexes qui prennent en compte les relations non-linéaires que les caractéristiques peuvent présenter peuvent être utilisées comme les SVM (Thompson et al., 2014; Gillet and Richard, 2008, 2004; Tindale et al., 2004; Gillet and Richard, 2005; Roy et al., 2007; Souza et al., 2015; Scholler and Perwins, 2011). Des méthodes d'ensemble comme Adaboost (Steelant et al., 2004) ou *Random Forest* (Scholler and Perwins, 2010) sont reconnues pour leur efficacité. Enfin des méthodes extraites du *deep learning* comme les *Convolutional Neural Network*, réseau de neurones convolutif (CNN) (Gajhede et al., 2016) sont aussi employées pour la transcription automatique de la batterie. Les dernières méthodes évoquées sont des méthodes dites supervisées. Des méthodes non supervisées sont aussi appliquées pour la classification comme l'algorithme de K-means (Gouyon et al., 2000; Bello et al., 2006; Miron et al., 2013a).

Certaines méthodes de classification d'événements sont basées sur des modèles probabilistes. Dans (Eronen, 2003), ils utilisent les HMM pour modéliser la progression temporelle des caractéristiques d'un extrait audio.

Un autre exemple d'algorithme de cette catégorie est l'algorithme de (Yoshii et al.,

2004, 2005, 2007). Ils commencent la décomposition avec les motifs initiaux des instruments de batteries. Ils sont ensuite mis à jour pour mieux correspondre à l'instrument étudié dans le morceau. L'étape d'adaptation alterne une tâche de détection d'événements avec le dernier motif estimé puis une mise à jour avec la moyenne des motifs extraits grâce à cette détection.

Pour résumer, les méthodes basées sur la classification, provenant du paradigme de *pattern recognition*, permettent une recherche efficace et automatique de paramètres. La possibilité d'ajouter différentes caractéristiques durant la phase de représentation fournit une flexibilité aux algorithmes quant à l'incorporation de connaissances expertes. Cependant, les performances de ces méthodes sont énormément liées aux performances de l'étape de segmentation. Les erreurs possibles à cette étape vont se propager dans tout le système. De plus, le système aura des difficultés à gérer l'apparition de nouveaux instruments et les superpositions qui nécessitent plus de classes d'instrument durant l'entraînement.

2.2.2.3 Méthodes basées sur le modèle du langage

Ces méthodes enchaînent une étape de représentation de caractéristique puis une de transformation auxquelles est ajoutée une étape basée sur le modèle du langage qui prend en compte l'enchaînement temporel des événements. Au lieu de prédire l'événement directement depuis le signal d'entrée, les modèles de langage infèrent l'événement en considérant les événements voisins et leur probabilité comme une séquence. Des modèles statistiques comme les HMM sont appris sur des données d'entraînement pour ensuite être appliqués à des morceaux inconnus.

Les premiers travaux à utiliser des modèles de langages se concentrent sur la transcription du *beat-boxing*, sons de percussion émis par la bouche (Nakano et al., 2005). Le système extrait les MFCC du signal audio puis traduit les caractéristiques acoustiques en séquences d'onomatopées avec l'algorithme de Viterbi. Enfin, les onomatopées sont appariées à une séquence de batterie en choisissant dans une base de données celle qui a le plus de similarité.

D'autres travaux utilisent les HMM pour modéliser les séquences de batterie (Paulus, 2009; Paulus and Klapuri, 2009). Le système représente le signal sous la forme *sinoids-plus-residual* pour supprimer les harmoniques. Les MFCC sont ensuite extraits et sont transformés dans une forme plus adaptée par LDA. Enfin l'algorithme de Viterbi et des HMM entraînés permettent de déterminer la séquence de batterie.

Les modèles de langage peuvent aussi servir comme post-traitement. Dans (Gillet and Richard, 2007), le modèle *Ngram* est appliqué sur des données symboliques pour affiner les *onsets* détectés.

Comme leurs noms l'indiquent, les modèles de langage sont appliqués à des langues qui ont une grammaire et qui sont basées sur des phonèmes, des mots, des séquences de mots. Ces modèles ne sont pas directement applicables à la musique. Par exemple, les pauses et les durées ont une importance moindre pour le langage que pour la musique où elles sont essentielles. De plus, les règles de la musique sont en général moins strictes que les règles de la parole. C'est pourquoi, pour pouvoir apprendre les différents modèles

probabilistes, les modèles de langage appliqués à la musique nécessitent énormément de données.

2.2.2.4 Méthodes basées sur les fonctions d'activation

La base de cette approche est la fonction d'activation. Elle permet de rendre compte de l'activité de l'instrument cible au cours du temps. En général, après avoir représenté les données sous une forme adéquate et déterminé la fonction d'activation, une étape de segmentation d'événements est appliquée. La plupart du temps, une recherche d'extrema locaux, *peak-picking*, est suffisante.

Une grande famille d'algorithmes utilisant les fonctions d'activation est basée sur un spectrogramme d'amplitude auquel on applique une factorisation matricielle qui le décompose en motifs de base accompagnés de leur fonction d'activation. Les premiers systèmes utilisés considèrent les signaux des sources comme statistiquement indépendants comme l'*Independant Component Analysis* (ICA), (FitzGerald et al., 2003b,a; FitzGerald, 2004) ou la *Non negative ICA* (NNICA), (Dittmar and Uhle, 2004). Le problème est que les signaux des différents instruments ne sont pas indépendants rythmiquement et ils peuvent aussi être liés à certains instruments harmoniques.

Aujourd'hui, de plus en plus de systèmes basés sur les fonctions d'activation utilisent la *Non negative Matrix Factorization*, NMF, et ses dérivées qui présentent moins de restriction sur l'indépendance statistique des sources et dont la seule contrainte, la non négativité des données, est naturellement remplie par l'emploi des spectrogrammes d'amplitude :

- *Non negative Matrix Factorisation*, (NMF), (Paulus and Virtanen, 2005; Alves et al., 2009)
- *Non negative Vector Decomposition* (NVD), (Battenberg et al., 2012)
- *Non negative Matrix Deconvolution*, (NMD), (Lindsay-Smith et al., 2012; Röbel et al., 2015)
- *Semi-adaptive* NMF, (Dittmar and Gärtner, 2014)
- *Partially fixed* NMF, (Wu and Lerch, 2015a,b, 2016)
- *Probabilistic Latent Componant Analysis*, (PLCA), (Benetos and Ewert, 2014)

Ces différentes méthodes nécessitent des motifs de base comme connaissance a priori. Elles présentent des résultats moins performants si les motifs présents dans le morceau ne correspondent pas exactement à ceux présents dans le dictionnaire. De plus, les différentes fonctions d'activation peuvent interférer entre elles.

La NMF et plusieurs méthodes dérivées de la NMF seront détaillées dans les chapitres suivants.

2.2.2.5 Méthodes basées sur les DNN

Les méthodes de *deep learning* permettent de créer des relations non-linéaires entre des entrées arbitraires et des sorties cibles grâce à des données d'entraînement. Les ré-

seaux de neurones, *Deep Neural Network* (DNN) sont généralement construits comme un empilement de couches apprenables de poids et de fonctions non-linéaires. L'apprentissage des poids du réseau se font par des méthodes de descente de gradient (Gloriot and Bengio, 2010). Un état de l'art plus détaillé est disponible au Chapitre 4.

Récemment une première version des DNN, les *Recurrent Neural Network*, réseau de neurones récurrent, RNN, a été appliquée à la transcription de la batterie ainsi que des déclinaisons des RNN : *Bidirectional* RNN (Southall et al., 2016), RNN *with label shift* (Vogl et al., 2016b) qui décale les annotations pendant l'apprentissage pour prendre en compte les données avant l'apparition de l'événement, et les RNN avec *Gated Recurrent Units*, GRU, ou *Long Short Term Memory*, LSTM, qui permettent d'introduire de la mémoire dans la récurrence pour prendre en compte les états précédents (Vogl et al., 2016a; Southall et al., 2017b). Récemment les CNN et leur dérivées, *Convolutional Recurrent Neural Network* (CRNN), ont obtenu des résultats prometteurs (Vogl et al., 2017; Southall et al., 2017b).

Ces méthodes produisent des représentations intermédiaires faciles à interpréter. De plus, elles permettent de prendre en compte les événements simultanés. Cependant, elles peuvent nécessiter beaucoup de données pour établir des relations non-linéaires robustes et pour pouvoir être ensuite généralisées sur des exemples qu'elles n'ont jamais rencontrés.

2.3 Les problèmes rencontrés lors de la transcription de la batterie

2.3.1 Les interférences entre instruments

Les interférences entre instruments est le plus gros défi rencontré par la transcription automatique de la batterie. Lorsque des instruments jouent ensemble, les superpositions temporelle et spectrale rendent la différenciation de ceux-ci difficile.

Un premier défi est la superposition d'instruments percussifs. Bien que les sons produits par les trois principaux instruments de la batterie soient relativement différents, il peut déjà être compliqué de les différencier lors d'événements simultanés. La tâche est encore plus ardue lorsqu'on prend en compte une batterie plus complète constituée des trois instruments principaux, la grosse-caisse, la caisse-claire et la Charleston auxquels on ajoute les toms basse, medium et aigu et les différentes cymbales. La différenciation se complexifie notamment lorsqu'ils jouent ensembles. Plus on ajoute de classes différentes d'instrument plus il est difficile de les détecter et de les différencier.

C'est ce que montrent des études qui ont tenté de différencier des éléments de percussion. (Herrera et al., 2003) procède à la classification de 33 instruments différents et (Souza et al., 2015) classe les différentes cymbales. Ces études ont été réalisées sur des sons de batterie isolés. Comme attendu, les performances décroîent lorsqu'on ajoute des instruments qui jouent en même temps.

La grande différence de sons entre la batterie et les instruments mélodiques pourraient permettre une différenciation plus facile. Pourtant certains sons peuvent se ressembler et se confondre. Par exemple, la grosse-caisse et la basse jouent dans les mêmes bandes de fréquence ou la caisse-claire et la guitare. Pour contourner ce problème, une solution

envisagée est de supprimer la partie harmonique comme (Yoshii et al., 2007; Gillet and Richard, 2008). Malheureusement, les améliorations apportées par ce prétraitement ne sont pas concluantes.

2.3.2 Les conditions d'enregistrement et de post-production

La superposition de deux événements est en principe une combinaison linéaire des deux sons. Cependant les conditions d'enregistrement ou les post-traitements appliqués peuvent faire que ces superpositions ne sont plus un mélange linéaire mais un mélange convolutif ou un mélange non-linéaire.

Les conditions acoustiques de la salle d'enregistrement ainsi que les microphones peuvent introduire des effets de réverbérations. De plus, certains filtrages sont appliqués et une égalisation est nécessaire après l'enregistrement des différents instruments de la batterie. Ces éléments peuvent être modélisés comme une convolution. De plus, des effets non-linéaires peuvent être ajoutés par des distorsions ou des compressions dynamiques.

Les méthodes qui partent du postulat d'une décomposition linéaire, comme la NMF, sont impactées par ces phénomènes. De plus, tous ces effets posent notamment problème aux méthodes utilisant le *machine learning*. S'ils ne sont pas pris en compte pendant l'apprentissage, les algorithmes pourront moins bien repérer ce qu'ils cherchent et les performances seront altérées. Une stratégie pour pallier ces effets négatifs pourrait être l'augmentation de données qui permettraient à ces méthodes de rencontrer plus de cas lors de l'apprentissage et ainsi de moins diminuer leurs performances.

2.3.3 Le manque de données annotées

Plusieurs méthodes nécessitent une étape préalable d'apprentissage. Pour cette étape, des données connues sont fournies à l'algorithme qui en extrait les caractéristiques dont il a besoin. Pour cela, les données, dite d'apprentissage, doivent être précisément annotées. Pour la transcription de la batterie, le moment de l'événement ainsi que l'instrument qui en est responsable sont nécessaires aux étapes d'apprentissage. Le *label* de l'instrument doit bien correspondre à l'exemple présenté, autrement l'algorithme apprendrait des caractéristiques d'un instrument pour en décrire un autre. La position de l'*onset* doit aussi être précise. Si l'*onset* est placé à plusieurs trames temporelles de différence, l'algorithme risque de ne pas considérer certaines caractéristiques importantes.

Générer ses annotations est difficile et très long notamment si elles sont réalisées "à la main". De plus, peu de bases de données sont disponibles, notamment parce que les lois de protection de la propriété intellectuelle ne permettent pas un partage facile des bases de données. D'autres limitations peuvent aussi être evoquées dont la complexité des bases de données ainsi que leur diversité ou encore le déséquilibre de la distribution des exemples.

- **Complexité** Certaines bases de données simplifient la tâche de transcription de la batterie. Elles ne contiennent que des événements isolés où seul un instrument intervient ou comme dans IDMT-SMT-Drums (Dittmar and Gärtner, 2014) présentent

des motifs rythmiques plutôt simples. Ainsi la tâche de transcription consiste simplement en une tâche de classification des événements.

- **Déséquilibre de répartition des exemples** Alors que certaines bases de données ne contiennent que des exemples isolées, d'autres sont composées uniquement de morceaux polyphoniques. En outre, les bases de données d'échantillons d'un seul coup de batterie ont beaucoup de fichiers (Tindale et al., 2004; Prockup et al., 2013) alors que les bases de données avec des séquences de batterie présentent mois d'exemples (Gillet and Richard, 2006; Dittmar and Gärtner, 2014; Marxer and Janer, 2013; Goto et al., 2002; Wu and Lerch, 2016).
- **Diversité** La musique utilisée dans les bases de données ne permet pas de représenter tous les styles ou les genres de musique. Par exemple, la section Pop de la base de données RWC de (Goto et al., 2002) ne contient que de la pop japonaise. Dans IDMT-SMT-Drum (Dittmar and Gärtner, 2014), les motifs rythmiques sont simples et issus du rock ou de la pop. Pour ENST-Drum (Gillet and Richard, 2006), seuls trois batteurs ont été enregistrés sur trois batteries différentes.

Des solutions au manque de données annotées ont été étudiées. Par exemple, Wu dans (Wu and Lerch, 2017) utilise des données non-annotées avec le paradigme de *student-teacher*. L'augmentation des données, *data augmentation* en anglais, est aussi souvent envisagée pour créer des données supplémentaires à partir de données correctement annotées.

Deuxième partie

Des outils pour la transcription automatique de la batterie

— Chapitre 3 —

Méthodes de décomposition non-négative

Les méthodes de décomposition non-négative sont des méthodes permettant d'extraire des informations caractéristiques d'un contenu, par exemple d'une image. Ces méthodes sont des méthodes de réduction de rang dans le sens où elles représentent l'information contenue dans l'objet à analyser sous forme parcimonieuse. Ces techniques sont utilisées dans de nombreux domaines :

- Traitement de l'image : représentation d'images et de visages (Lee and Seung, 1999; Li et al., 2001), classification (Guillamet et al., 2001)
- Traitement du texte : Surveillance des messages électroniques (Berry and Browne, 2005), classification de documents (Ding et al., 2008)
- Économie : diversification de porte feuille d'actions (Drakakis et al., 2008)
- Biologie : *clustering* de gênes impliqués dans le cancer (Liu and Yuan, 2008), détection de l'activité neuronale pour les interfaces cerveau-machine (Kim et al., 2005)

Les méthodes de décomposition non-négative regroupent différents algorithmes. Tous ces algorithmes décomposent l'information en plusieurs matrices. Tous les coefficients de la matrice à décomposer ainsi que des matrices résultantes sont positifs ou nuls. La non-négativité rend plus facile l'interprétation des matrices résultantes puisqu'aucune soustraction n'est possible. Cela empêche l'annulation de la contribution d'un des éléments.

Dans ce chapitre, on détaille trois méthodes de décomposition non-négative : une déterministe, la NMF et deux probabilistes la PLCA et la IS-NMF/EM. On détaille aussi leur variantes, la NMD et la SI-PLCA. La IS-NMF/EM n'existe que dans le cas de bases à une dimension. Dans ce chapitre on adaptera la IS-NMF/EM pour extraire des caractéristiques en deux dimensions. Ces méthodes seront par la suite comparées pour le problème de transcription automatique de la batterie.

3.1 NMF/NMD

3.1.1 NMF

La NMF est une méthode d'apprentissage introduite dans (Paatero and Tapper, 1994) et popularisée par Lee et Seung dans (Lee and Seung, 1999). Son but est d'approcher une matrice non-négative $\boldsymbol{V}$ de taille $F \times N$ par la résultante de la factorisation de deux matrices non-négatives :

$$\boldsymbol{V} \approx \tilde{\boldsymbol{V}} = \boldsymbol{W}\boldsymbol{H} \tag{3.1}$$

avec $\boldsymbol{W}$ et $\boldsymbol{H}$ de tailles respectives $F \times K$ et $K \times N$.

La NMF consiste donc à rechercher d'une matrice $\tilde{\boldsymbol{V}}$ de rang K, préalablement fixé, qui soit la plus proche de la matrice cible selon une fonction de coût choisie. La matrice dictionnaire $\boldsymbol{W}$ comporte les bases de la décomposition. La matrice d'activation $\boldsymbol{H}$ décrit à quelle intensité et quand ces bases sont activées. Le choix du rang K, petit devant F et N permet une réduction de rang de la matrice et donc une représentation plus parcimonieuse des données.

La principale propriété de la NMF est la non-négativité de ces paramètres. Chaque élément est positif ou nul, obligeant la décomposition des données à se faire par additivité. De ce fait, la soustraction des bases n'est pas possible et il est donc impossible d'annuler la contribution d'une base à la reconstruction.

Dans le cadre de l'audio, la matrice $\boldsymbol{V}$ est souvent une représentation temps-fréquence des données, par exemple un spectrogramme à partir de la TFCT. Le dictionnaire $\boldsymbol{W}$ est alors constitué de spectres de puissance des sources sonores et la matrice $\boldsymbol{H}$ est leurs activations temporelles. En général, les bases $\boldsymbol{W}$ sont normalisées et $\boldsymbol{H}$ représente l'évolution temporelle de l'énergie qui leur est allouée. Par définition, la matrice $\boldsymbol{V}$ ainsi que les matrices $\boldsymbol{W}$, $\boldsymbol{H}$ et donc $\tilde{\boldsymbol{V}}$ vérifient bien la propriété de non-négativité.

3.1.2 Fonctions de coût

L'estimation $\tilde{\boldsymbol{V}}$ des données $\boldsymbol{V}$ de l'équation (3.1) se fait par la minimisation, par rapport aux paramètres $\boldsymbol{W}$ et $\boldsymbol{H}$, d'une fonction de coût. Cette dernière mesure l'erreur entre la cible et la reconstruction et est de la forme :

$$D(\boldsymbol{V}|\boldsymbol{W}\boldsymbol{H}) = \sum_{f=1}^{F}\sum_{n=1}^{N} d(\boldsymbol{V}_{fn}|(\boldsymbol{W}\boldsymbol{H})_{fn}) \tag{3.2}$$

Les fonctions de coût usuelles pour l'audio sont la distance Euclidienne (Lee and Seung, 1999), les β-divergences (Kompass, 2005), notamment la divergence de Kullback-Leibler (KL) et d'Itakura-Saïto (IS) parmi d'autres (Cichocki et al., 2006). La distance Euclidienne et les divergence de KL et d'IS sont des cas particuliers et cas limites des β-divergences. Elles sont données dans le tableau 3.1.

La β-divergence présente des propriétés lui permettant d'être considérée comme une bonne mesure d'erreur de reconstruction (Bertin, 2009) :

β-divergence ($\beta \in \mathbb{R} \setminus \{0,1\}$)	$d_\beta(x\|y)$	$=$	$\frac{1}{\beta(\beta-1)}\left(x^\beta + (\beta-1)y^\beta - \beta x y^{\beta-1}\right)$
Itakura-Saito	$d_{IS}(x\|y)$	$=$	$\frac{x}{y} - \log\frac{x}{y} - 1$
		$=$	$\lim_{\beta\to 0} d_\beta(x\|y)$
Kullback-Leiber	$d_{KL}(x\|y)$	$=$	$x\log\frac{x}{y} + (y-x)$
		$=$	$\lim_{\beta\to 1} d_\beta(x\|y)$
Distance euclidienne	$d_{EUC}(x\|y)$	$=$	$\frac{1}{2}(x-y)^2$
		$=$	$d_{\beta=2}(x\|y)$ avec

TABLEAU 3.1 – *Formules de la β-divergence scalaire et de ses cas particuliers : IS, KL et EUC*

- $d_\beta(x|y)$ possède un unique minimum global de valeur nulle au point $x = y$.
- $d_\beta(x|y)$ est positif et croît avec la distance $|x - y|$.
- Pour tout $\lambda \in \mathbb{R}^*$, $d_\beta(\lambda x|\lambda y) = \lambda^\beta d_\beta(x|y)$.

Dans le cas de la divergence d'IS, la troisième propriété devient en particulier une propriété d'invariance d'échelle :

$$\forall \lambda \in \mathbb{R}^*, d_{IS}(\lambda x|\lambda y) = d_{IS}(x|y) \tag{3.3}$$

Cela permet de donner le même poids aux coefficients de faible et forte amplitudes. C'est la fonction de coût qui se rapproche le plus de la perception humaine. Cependant, il est possible que cette propriété perturbe la stabilité de l'algorithme si certains coefficients du spectrogramme cible $\boldsymbol{V}$ ont des valeurs très petites ou nulles. Un palier de bruit peut être ajouté aux spectrogrammes cible $\boldsymbol{V}$ et estimé $\tilde{\boldsymbol{V}}$ (Röbel et al., 2015). En plus d'assurer la stabilité, cela permet de définir la précision maximale de la reconstruction.

3.1.3 Mise à jour multiplicative

Un inventaire exhaustif d'algorithmes de minimisation de la fonction de coût ainsi que leur comparaison sont disponibles dans (Cichocki et al., 2006). On y retrouve notamment les algorithmes multiplicatifs proposés dans (Lee and Seung, 1999).

Les règles de mise à jour multiplicative sont particulièrement intéressantes par leur simplicité d'implémentation et par la conservation de la non-négativité des paramètres à chaque itération. Dans (Nakano et al., 2010), les règles de mise à jour multiplicative sont

données pour la β-divergence et ainsi que la preuve de la non-croissance de la fonction de coût. Elles sont rappelées ci-dessous :

$$\boldsymbol{W} \leftarrow \boldsymbol{W} \otimes \frac{(\boldsymbol{V} \otimes (\boldsymbol{W}\boldsymbol{H})^{\beta-2})\boldsymbol{H}^T}{(\boldsymbol{W}\boldsymbol{H})^{\beta-1}\boldsymbol{H}^T}, \quad \boldsymbol{H} \leftarrow \boldsymbol{H} \otimes \frac{\boldsymbol{W}^T(\boldsymbol{V} \otimes (\boldsymbol{W}\boldsymbol{H})^{\beta-2})}{\boldsymbol{W}^T(\boldsymbol{W}\boldsymbol{H})^{\beta-1}} \tag{3.4}$$

avec $\otimes$ le produit d'Hadamard et la division s'effectuant terme à terme.

Précisons que la preuve de non-croissance n'implique pas la convergence de l'algorithme vers un minimum local. En effet, il peut tendre vers un point d'équilibre qui n'est pas un minimum local, un point de selle par exemple qui serait un minimum par rapport à $\boldsymbol{W}$ mais un maximum par rapport à $\boldsymbol{H}$.

L'utilisation des règles de mise à jour multiplicative peut être vue comme un cas particulier de la descente de gradient avec un pas d'avancement ν déterminé automatiquement. Si on note C la fonction de coût, alors la descente de gradient appliquée à un paramètre, disons $\boldsymbol{W}$, s'écrit à l'itération $i+1$:

$$\boldsymbol{W}^{i+1} \leftarrow \boldsymbol{W}^i - \nu \nabla_{\boldsymbol{W}} C = \boldsymbol{W}^i - \nu(\nabla^+_{\boldsymbol{W}} C - \nabla^-_{\boldsymbol{W}} C) \tag{3.5}$$

avec $\nabla^+_{\boldsymbol{W}} C$ et $\nabla^-_{\boldsymbol{W}} C$ positifs. En choisissant ν tel que

$$\nu = \frac{\boldsymbol{W}^i}{\nabla^+_{\boldsymbol{W}}} \tag{3.6}$$

la mise à jour 3.5 devient :

$$\boldsymbol{W}^{i+1} \leftarrow \boldsymbol{W}^i \otimes \frac{\nabla^-_{\boldsymbol{W}} C}{\nabla^+_{\boldsymbol{W}} C} \tag{3.7}$$

Cette formulation du pas d'avancement et des règles de mise à jour multiplicative permet d'ajouter facilement des pénalités dans la fonction de coût et de changer facilement de divergence. Cependant, il n'existe aucune preuve de convergence de la descente de gradient.

3.1.4 NMD

La *Non negative Matrix Deconvolution* (NMD) ou *Convolutive Non negative Matrix Factorisation* (CNMF) est une version de la NMF (Smaragdis, 2004) qui permet d'identifier ou classifier des composantes présentant des structures temporelles spécifiques. Au lieu d'estimer le spectrogramme $\boldsymbol{V}$ en un produit de matrices, $\boldsymbol{V}$ est décomposé sous la forme d'une convolution. Les bases ne sont plus un vecteur représentant par exemple le spectre d'une note mais une matrice représentant un motif temporel. La matrice dictionnaire $\boldsymbol{W}$ est remplacée par un tenseur de taille $F \times K \times T$ avec T la longueur temporelle du motif. On note $\boldsymbol{W}^t$ la matrice extraite de $\boldsymbol{W}$, de taille $F \times K$, correspondant au temps t du motif. Le modèle s'écrit :

$$\boldsymbol{V} \approx \tilde{\boldsymbol{V}} = \sum_{t=1}^{T} \boldsymbol{W}^t \boldsymbol{H}^{t\rightarrow} \tag{3.8}$$

avec $.^{t\rightharpoonup}$ l'opérateur de translation suivant, explicité dans (Smaragdis, 2004) :

$$\boldsymbol{A} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{pmatrix} \quad \boldsymbol{A}^{2\rightharpoonup} = \begin{pmatrix} 0 & 0 & 1 & 2 \\ 0 & 0 & 5 & 6 \end{pmatrix} \tag{3.9}$$

Le principe de la NMD est le même que celui de la NMF : estimer les paramètres $\boldsymbol{W}$ et $\boldsymbol{H}$ par minimisation d'une fonction de coût (Bertin, 2009). Les règles de mise à jour multiplicative s'écrivent alors (Mitsufuji et al., 2007) :

$$\boldsymbol{W}^t \leftarrow \boldsymbol{W}^t \otimes \frac{\left(\boldsymbol{V} \otimes \tilde{\boldsymbol{V}}^{\otimes(\beta-2)}\right)\boldsymbol{H}^{t\rightharpoonup}}{\tilde{\boldsymbol{V}}^{\otimes(\beta-1)}\boldsymbol{H}^{t\rightharpoonup}}, \quad \boldsymbol{H} \leftarrow \boldsymbol{H} \otimes \frac{\sum_t \boldsymbol{W}^t\left(\boldsymbol{V} \otimes \tilde{\boldsymbol{V}}^{\otimes(\beta-2)}\right)^{t\leftharpoonup}}{\sum_t \boldsymbol{W}^t\left(\boldsymbol{V} \otimes \tilde{\boldsymbol{V}}^{\otimes(\beta-1)}\right)^{t\leftharpoonup}} \tag{3.10}$$

où $\otimes$ est le produit d'Hadamard, produit terme à terme. Les puissances notées $.^{\otimes(.)}$ sont appliquées aux composantes de la matrice. Les divisions se font terme à terme. Le principe est représenté sur la figure 3.1.

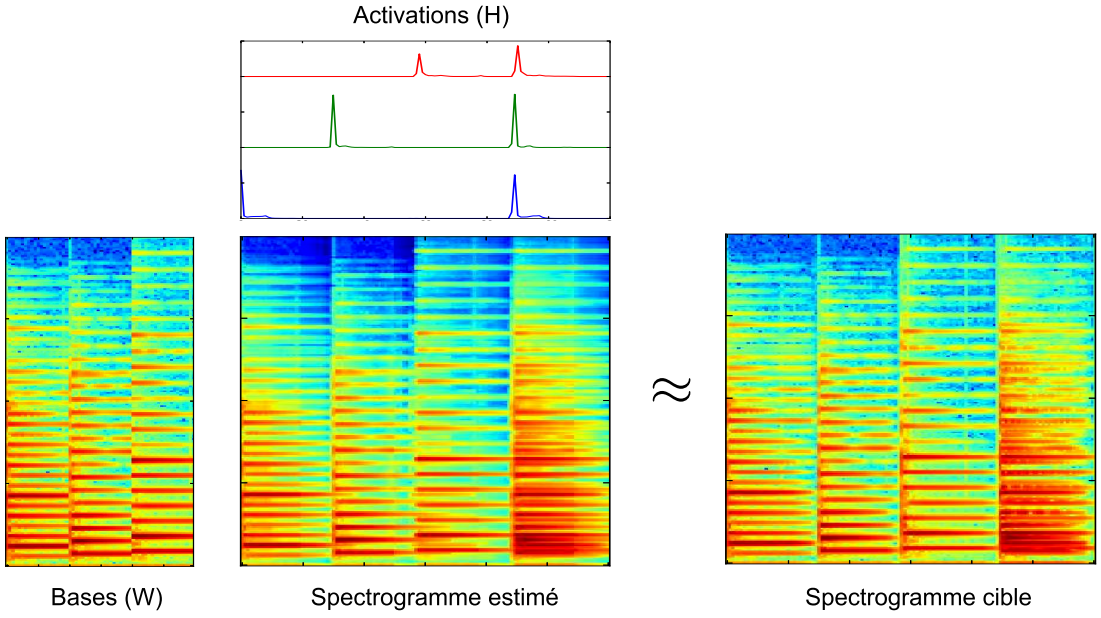

FIGURE 3.1 – *Principe de la NMD.*

Inclure les structures temporelles aux bases du dictionnaire permet à la la NMD de mieux représenter des données dont l'évolution temporelle en est une caractéristique importante. C'est notamment le cas pour la batterie. Les réponses temporelles des tambours et cymbales varient peu d'un coup de baguette à l'autre.

3.1.5 Variantes contraintes

La seule contrainte inhérente au problème de NMF est la non-négativité des paramètres. Toutes les autres propriétés qui pourraient apparaître sont des conséquences non prévues mais qui peuvent être intéressantes à maîtriser.

Plusieurs recherches ont porté sur l'introduction de certaines contraintes dans le modèle comme la parcimonie (Hoyer, 2004), la localisation spatiale (Li et al., 2001), la décorrélation entre les sources (Bouvier et al., 2016) ou la continuité temporelle (Virtanen, 2007; Chen et al., 2006).

Pour tous les modèles basés sur la NMF et ses variantes, l'introduction de contrainte se fait par l'ajout d'un terme de pénalité à la fonction de coût. La nouvelle fonction de coût est donc décomposée avec un coût de reconstruction qui mesure la distance entre la reconstruction et l'originale C_r et des coûts de pénalité P_c pour la contrainte c. Le problème revient à chercher les paramètres $\boldsymbol{W}$ et $\boldsymbol{H}$ pour minimiser la fonction

$$C(\boldsymbol{V}|\boldsymbol{WH}) = C_r(\boldsymbol{V}|\boldsymbol{WH}) + \sum_c \lambda_c P_c(\boldsymbol{W}, \boldsymbol{H}) \tag{3.11}$$

λ_c est un paramètre de pondération permettant de donner plus ou moins de poids à la contrainte. Le choix de ce paramètre doit être réalisé précautionneusement. En effet, plus la reconstruction est proche de l'originale, plus la contrainte va prendre de poids. S'il est trop grand, la pénalité prendra trop d'importance. Par contre s'il est trop petit, elle n'aura aucune incidence.

Dans la suite, différentes contraintes régulièrement utilisées sont présentées.

3.1.5.1 Parcimonie

Généralement, le dictionnaire $\boldsymbol{W}$ est appris préalablement ou bien fixé arbitrairement. Le choix ou l'apprentissage du dictionnaire est déjà un défi. Les bases doivent être choisies pour pouvoir décrire toutes ou du moins une majorité des données à décomposer. Le choix ou l'apprentissage des bases peut conduire à un recoupement des informations entre les bases. L'utilisation de plusieurs bases pour décrire le même objet rend l'analyse plus complexe.

La parcimonie, *sparsity* en anglais, peut être définie comme la propriété d'être "souvent nul". Plus un vecteur ou une matrice tend vers le fait d'avoir un seul coefficient non-nul, plus il est parcimonieux. En imposant la parcimonie à chaque trame temporelle sur les activations, on réduit les chances d'activer plusieurs bases qui décrivent les mêmes caractéristiques ou de distribuer l'information sur plusieurs bases dans le cas d'une décomposition non-informée.

En général, pour renforcer la parcimonie des activations, on cherche à résoudre

$$\arg\min_{\boldsymbol{W},\boldsymbol{H}} (D(\boldsymbol{V}|\boldsymbol{WH}) + ||\boldsymbol{H}||_0) \tag{3.12}$$

La norme $\ell 0$ consiste à dénombrer les coefficients non nuls de $\boldsymbol{H}$ mais elle n'est pas différenciable. Elle peut être remplacée par la norme $\ell 1$ comme dans (Hoyer, 2002).

D'autres formulations existent. Dans (Hoyer, 2004), la mesure de la parcimonie est définie comme le rapport entre les normes $\ell 1$ et $\ell 2$. (Eggert and Korner, 2004) propose aussi un terme de contrainte basé sur la norme $\ell 1$ qui renforce également la parcimonie.

3.1.5.2 Continuité temporelle

Pour la NMF, les lignes et les colonnes de la matrice à décomposer sont considérées indépendantes les unes des autres. Pour le signal audio, les lignes représentent généralement l'évolution temporelle de l'énergie dans une certaine bande de fréquences. Ces variations d'énergie se font relativement lentement. Or, il se peut que la décomposition fasse apparaître des changements brutaux entre deux trames successives.

Pour empêcher ces changements lors de la décomposition par NMF, on impose à la ligne de $\boldsymbol{H}$ correspondant à une des bases du dictionnaire d'être continue temporellement, ou régulière, c'est-à-dire que deux coefficients successifs ne différent pas énormément l'un de l'autre. Cette contrainte peut directement être exprimée par la différence entre deux coefficients successifs $(h_{k(n+1)} - h_{kn})$ (Virtanen, 2007) ou comme le rapport des variances à court et long terme (Chen et al., 2006).

Il est à noter que la régularité d'un paramètre peut renforcer la parcimonie d'un autre paramètre. Cependant l'utilisation des deux contraintes en même temps ne donnera pas forcément d'avantage comme dans l'exemple traité dans (Virtanen, 2007). De plus, la continuité temporelle des activations peut être contre-productive. En effet, si la base du dictionnaire est exactement l'événement analysé, l'activation devrait avoir la forme d'une impulsion de Dirac, qui n'est pas continue.

3.1.5.3 Décorrélation

Avec la contrainte de décorrélation, on souhaite limiter la séparation en plusieurs composantes d'une unité en essayant d'obtenir les lignes de $\boldsymbol{H}$ les plus décorrélées possible. Dans (Li et al., 2001), les auteurs expriment cette contrainte par une pénalisation de la fonction de coût dépendant du produit $\boldsymbol{H}\boldsymbol{H}^T$. Renforcer la diagonalité de cette matrice revient à minimiser la corrélation entre les lignes. La pénalisation est alors exprimée comme :

$$P_{corr}(\boldsymbol{H}) = \sum_{j \neq k} [\boldsymbol{H}\boldsymbol{H}^T]_{kj} \tag{3.13}$$

Dans (Bouvier et al., 2016), une version normalisée est proposée :

$$P_d(\boldsymbol{H}) = \sum_{k=1}^{K} \sum_{l=1, l \neq k}^{K} \frac{\sum_{n=1}^{N} \boldsymbol{H}_{kn}\boldsymbol{H}_{ln}}{\sqrt{\sum_{n=1}^{N} \boldsymbol{H}_{kn}^2}\sqrt{\sum_{n=1}^{N} \boldsymbol{H}_{kn}^2}} \tag{3.14}$$

3.2 PLCA/SI-PLCA

La *Probabilistic Latent Componant Analysis* est une technique de décomposition basée sur un modèle probabiliste introduite par (Smaragdis and Raj, 2007). À partir d'observations, on cherche à déduire une structure sous-jacente pour que la reconstruction de

l'objet soit la plus proche de l'objet réel. Pour cela, la décomposition s'appuie sur des propriétés statistiques de l'objet à décomposer.

Contrairement à la NMF qui essaie de caractériser les données directement, la PLCA essaie de caractériser la distribution sous-jacente des paramètres observés. Grâce à cette différence subtile, on conserve les avantages de la NMF et on surmonte certaines de ses limitations en fournissant un environnement facile à généraliser, à étendre et à interpréter.

La PLCA est efficace pour modéliser des sons non-stationnaires puisque le dictionnaire appris adapte les caractéristiques invariantes du signal d'entrée. Ainsi, une combinaison linéaire d'éléments du dictionnaire suffit à représenter le motif temporel d'un signal audio. De plus, la nature probabiliste de la PLCA permet l'introduction de connaissances a priori notamment pour suivre l'hypothèse de parcimonie.

3.2.1 PLCA

La PLCA est une extension de la *Probabilistic Latent Semantic Indexing* (PLSI) (Hofmann, 1999). Le modèle de base est défini comme :

$$P(\mathbf{x}) = \sum_z P(z) \prod_{j=1}^{N} P(x_j|z) \tag{3.15}$$

où $P(\mathbf{x})$ est la distribution de probabilité de dimension N de la variable aléatoire $\mathbf{x}$. z est une variable latente et $P(x_j|z)$ sont des distributions de probabilité.

Le modèle est représenté comme un mélange de distributions marginales. Dans le cadre d'un signal audio, $\mathbf{x}$ représente la distribution de deux variables aléatoires f et t, la fréquence et le temps. Ainsi, $\mathbf{x} = [f, t]$ et le modèle devient

$$P(\mathbf{x}) = \sum_z P(z)P(f|z)P(t|z) \tag{3.16}$$

Pour déterminer les distributions marginales $P(z)$, $P(f|z)$ et $P(t|z)$, on maximise la fonction de vraisemblance ou de log-vraisemblance avec une variante de l'algorithme *Expectation-Maximization* (EM). Tous les calculs sont détaillés en Annexe A.

$$Q(\Lambda, \Lambda^{(n)}) = E_{\mathbf{x}}[E_{z|\mathbf{x};\Lambda^{(n)}}[\log P(\mathbf{x}, z; \Lambda)]] \tag{3.17}$$

Pendant l'étape d'*Expectation*, la contribution de la variable latente z est estimée :

$$R(f, t, z) = \frac{P(z)P(f|z)P(t|z)}{\sum_{z'} P(z')P(f|z')P(t|z')} \tag{3.18}$$

et pendant l'étape de *Maximization*, les distributions marginales sont ré-estimées :

$$P^{(n+1)}(z) \leftarrow \sum_{f,t} P(f,t)R(f,t,z) \tag{3.19}$$

$$P^{(n+1)}(f|z) \leftarrow \frac{\sum_t R(f,t,z)P(f,t)}{P^{(n+1)}(z)} \tag{3.20}$$

$$P^{(n+1)}(t|z) \leftarrow \frac{\sum_f R(f,t,z)P(f,t)}{P^{(n+1)}(z)} \tag{3.21}$$

On peut écrire ces équations sous la forme de mises à jour multiplicatives.

$$R(f,t,z) \leftarrow \frac{P(z)P(f|z)P(t|z)}{\sum_{z'} P(z')P(f|z')P(t|z')} \tag{3.22}$$

$$P^{(n+1)}(z) \leftarrow P(z)\sum_{f,t} \frac{P(f,t)P(f|z)P(t|z)}{\sum_{z'} P(z')P(f|z')P(t|z')} \tag{3.23}$$

$$P^{(n+1)}(f|z) \leftarrow P(f|z)\frac{\sum_t P(f,t)P(z)P(t|z)}{\sum_{f,t} P(f,t)R(f,t,z)} \tag{3.24}$$

$$P^{(n+1)}(t|z) \leftarrow P(t|z)\frac{\sum_f P(f,t)P(z)P(f|z)}{\sum_{f,t} P(f,t)R(f,t,z)} \tag{3.25}$$

L'équivalence numérique entre la PLCA et la NMF avec la divergence de KL a été démontrée dans (Shashanka et al., 2008).

3.2.2 SI-PLCA

Certains modèles peuvent présenter des invariances par translation. Par exemple, avec une représentation temps-fréquence bien choisie, le spectre d'une note peut être la translation suivant les fréquences de celui d'une autre note. Mais l'invariance peut aussi être temporelle, lorsqu'un événement se répète toujours avec la même réponse temporelle.

La SI-PLCA (Smaragdis and Raj, 2007) modélise certaines distributions comme la convolution d'une distribution appelée noyau avec une distribution qui présente une invariance par translation appelée impulsion. On note $\mathbf{w}$ les noyaux et $\mathbf{y}$ les impulsions. Le modèle s'écrit alors :

$$P(\mathbf{x};\Lambda) = \sum_z P(z)\sum_\tau P(\mathbf{w},\tau|z)P(\mathbf{y}-\tau|z) \tag{3.26}$$

On applique une variante de l'algorithme EM pour obtenir les équations de mise à jour des paramètres. Tous les calculs sont présentés en annexe. On estime la contribution de z à l'étape E :

$$\begin{aligned} R(\mathbf{x},\tau,z) &= P(\tau,z|\mathbf{x}) \\ &= \frac{P(z)P(\mathbf{w},\tau|z)P(\mathbf{y}-\tau|z)}{\sum_z P(z)\sum_\tau P(\mathbf{w},\tau|z)P(\mathbf{y}-\tau|z)} \end{aligned} \tag{3.27}$$

puis on met à jour les distributions marginales :

$$P^{(n+1)}(z) = P(z)^{(n)} \sum_{\tau,f,t} \frac{P(f,\tau|z)P(t-\tau|z)}{\sum_{z'} P(z') \sum_{\tau'} P(f,\tau'|z')P(t-\tau'|z')} \tag{3.28}$$

$$P^{(n+1)}(t|z) = P^{(n)}(t|z) \frac{\sum_{\tau,f} P^{(n)}(z)P^{(n)}(f,\tau|z)P^{(n)}(f,t+\tau)}{P^{(n+1)}(z) \sum_{z'} P(z') \sum_{\tau'} P(f,\tau'|z')P(t-\tau'|z')} \tag{3.29}$$

$$P^{(n+1)}(f,\tau|z) = P^{(n)}(f,\tau|z) \frac{\sum_{t} P(f,t)P^{(n)}(z)P^{(n)}(t-\tau|z)}{P^{(n+1)}(z) \sum_{z'} P(z') \sum_{\tau'} P(f,\tau'|z')P(t-\tau'|z')} \tag{3.30}$$

Comme pour la NMF et la PLCA, on peut établir un lien entre la NMD avec la divergence KL et la SI-PLCA.

3.3 IS-NMD statistique

3.3.1 IS-NMF/EM

Comme pour la PLCA, un rapprochement peut être établi entre la NMF et d'autres modèles statistiques. On parle d'équivalence entre NMF et modèle statistique si une équivalence formelle entre la minimisation de la fonction de coût dans le cas de la NMF et l'estimation des paramètres du modèles statistiques posés est établie. Une liste de ces équivalences est donnée dans (Févotte and Cemgil, 2009).

Dans un contexte de séparation de source (Benaroya et al., 2003), on peut émettre l'hypothèse que la résultante du mélange des sources peut être considérée comme une somme de gaussiennes indépendantes. Dans le cadre de la transcription de la batterie, $\boldsymbol{V} = |\mathbf{X}|^2$ avec $\boldsymbol{x}_n = [x_{1n}, \ldots, x_{Fn}]^T$ est la représentation temps-fréquence d'un signal numérique x. f est alors l'indice du point fréquentiel et n l'indice de la trame temporelle.

On considère le modèle génératif suivant pour tout n de 0 à N :

$$\boldsymbol{x}_n = \sum_{k=1}^{K} \mathbf{c}_{kn} \tag{3.31}$$

tel que les $\mathbf{c}_{kn} \in \mathbb{C}^{F\times 1}$ suivent une loi gaussienne :

$$\mathbf{c}_{kn} \sim \mathcal{N}_C(0, h_{kn}\mathrm{diag}(\mathbf{w}_k)) \tag{3.32}$$

La représentation temps-fréquence est décomposée comme la somme d'éléments de base caractérisés par leur densité spectrale $\mathbf{w}_k$ modulée par leur activation temporelle h_{kn}.

Dans (Fevotte et al., 2009), le théorème de l'équivalence entre la NMF avec la divergence d'IS et l'estimation au sens du maximum de vraisemblance est posé. Sa preuve est donnée dans (Bertin, 2009). On rappelle le théorème ici :

Théorème 1 (IS-NMF comme estimateur du Maximum de Vraisemblance (MV) dans un modèle de somme de gaussiennes)**.** *L'estimation au sens du maximum de vraisemblance des matrices $\boldsymbol{W}$ et $\boldsymbol{H}$ à partir de l'observation $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]$ est équivalente à la résolution du problème NMF : $\boldsymbol{V} \approx \boldsymbol{W}\boldsymbol{H}$, par minimisation de la divergence d'Itakura-Saïto.*

La structure additive permet d'utiliser une extension des algorithmes EM, l'algorithme *Space Alternating Generalized EM* (SAGE) (Fessler and Hero, 1994). SAGE est conçu pour des données ayant des structures particulières. Dans ce modèle, la structure additive des données gaussiennes permet des mises à jour séparées des paramètres. On raisonne donc sur la participation de chaque base séparément. Dans (Bertin, 2009), les calculs sont détaillés dans le cadre de la NMF. Nous garderons la notation abrégé IS-NMF/EM. Nous détaillons par la suite l'adaptation de la IS-NMF/EM au modèle déconvolutif de la NMF.

3.3.2 IS-NMD/EM

N. Bertin a proposé la IS-NMF/EM avec un dictionnaire de bases à une dimension. Comme pour la NMF et PLCA, on souhaite avoir une version de la IS-NMF/EM pour des bases à deux dimensions. J'adapte donc ici les calculs pour un dictionnaire de motifs à deux dimensions.

Le modèle génératif est $\mathbf{x}_n = \sum_{k=1}^{K} \mathbf{c}_{kn}$ tels que $\mathbf{c}_{kn} \in \mathbb{C}^{F\times 1}$ suivent une loi gaussienne :

$$\mathbf{c}_{kn} \sim \mathcal{N}_C(0, \sum_{k,t} W_{fk}^t h_{kn}^{t\rightarrow}) \tag{3.33}$$

La fonction de coût s'écrit alors :

$$\begin{aligned} C_{MV}(\boldsymbol{W}, \boldsymbol{H}) &= -\sum_{n=1}^{N}\sum_{f=1}^{F} \log \mathcal{N}_C(x_{fn}|0, \sum_{k,t} W_{fk}^t h_{kn}^{t\rightarrow}) && (3.34)\\ &= NF\log\pi + \sum_{n=1}^{N}\sum_{f=1}^{F} \log(\sum_{k,t} W_{fk}^t h_{kn}^{t\rightarrow}) + \frac{|x_{fn}|^2}{\sum_{k,t} W_{fk}^t h_{kn}^{t\rightarrow}} && (3.35)\\ &= \sum_{n=1}^{N}\sum_{f=1}^{F} d_{IS}(|x_{fn}|^2 | \sum_{k,t} W_{fk}^t h_{kn}^{t\rightarrow}) + c && (3.36) \end{aligned}$$

où c est une constante.

Comme dans le cas NMF, on peut remarquer que déterminer les variables $\boldsymbol{W}$ et $\boldsymbol{H}$ par un estimateur du maximum de vraisemblance revient à résoudre le problème de la NMD utilisant la divergence d'IS.

Dans le modèle NMD, le spectrogramme est vu comme la convolution temporelle de plusieurs composantes gaussiennes. Comme pour la NMF/EM, l'algorithme SAGE permet d'estimer les paramètres du problème. La structure restant additive, elle permet des mises à jour séparées des paramètres. Les différentes étapes de l'algorithme sont expliquées ici.

Etape E

On note $\boldsymbol{\theta}_k = (W_k, h_k)$ une partition de l'ensemble des paramètres $\boldsymbol{\theta}$ avec W_k la matrice de la k-ième base et h_k la k-ième ligne de $\mathbf{H}$ représentant les activations de celle-ci.

L'algorithme consiste à maximiser la log-vraisemblance. On écrit alors la fonction de coût, égale à l'opposé de l'espérance conditionnelle de la log-vraisemblance des variables latentes $\mathbf{C}_k$ (les variables notées avec un $'$ sont les variables de l'itération en cours si elles ont déjà été mises à jour ou de l'itération précédente autrement) :

$$Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta}') = -\int_{\mathbf{C}_k} \log p(\mathbf{C}_k|\boldsymbol{\theta}_k) p(\mathbf{C}_k|\mathbf{X}, \boldsymbol{\theta}') d\mathbf{C}_k \tag{3.37}$$

$Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta}')$ fait intervenir deux quantités : la log-vraisemblance des données latentes $\log p(\mathbf{C}_k|\boldsymbol{\theta}_k)$ et la distribution a posteriori des données latentes $p(\mathbf{C}_k|\mathbf{X}, \boldsymbol{\theta}')$.
La première quantité se calcule avec (3.33) de la manière suivante :

$$-\log p(\mathbf{C}_k|\boldsymbol{\theta}_k) = -\sum_{n=1}^{N}\sum_{f=1}^{F} \log \mathcal{N}_c(c_{k,fn}|0, \sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}) \tag{3.38}$$

$$= \sum_{n=1}^{N}\sum_{f=1}^{F} \log(\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}) + \frac{|c_{k,fn}|^2}{\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}} \tag{3.39}$$

La deuxième quantité s'écrit :

$$p(\mathbf{C}_k|\mathbf{X}, \boldsymbol{\theta}') = \prod_{n=1}^{N}\prod_{f=1}^{F} \mathcal{N}_C(c_{k,fn}|\mu_{k,fn}^{post'}, \lambda_{k,fn}^{post'}) \tag{3.40}$$

où $\mu_{k,fn}^{post'}$ et $\lambda_{k,fn}^{post'}$ sont calculées grâce aux formules du filtrage de Wiener.

L'étape E consiste alors à calculer l'espérance de la log-vraisemblance des données latentes conditionnellement à la distribution a posteriori des données latentes :

$$Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta}') = \sum_{n=1}^{N}\sum_{f=1}^{F} \log(\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}) + \frac{|\mu_{k,fn}^{post'}|^2 + \lambda_{k,fn}^{post'}}{\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}} \tag{3.41}$$

et on pose $\hat{v}'_{k,fn} = |\mu_{k,fn}^{post'}|^2 + \lambda_{k,fn}^{post'}$.

Etape M

L'étape de maximisation revient à chercher les paramètres qui maximisent la vraisemblance donc qui minimisent la fonction $Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta}')$. On calcule les gradients par rapport à W_{fk}^t et par rapport à h_{kn}.

On obtient :

$$\nabla_{h_{kn}} Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta}') = \sum_f \sum_t W_{fk}^t \left(\frac{1}{\sum_{t'} W_{fk}^{t'} h_{k(n+t)}^{t'\rightharpoonup}} - \frac{\hat{v}'_{k,f(n+t)}}{(\sum_{t'} W_{fk}^{t'} h_{k(n+t)}^{t'\rightharpoonup})^2} \right) \tag{3.42}$$

$$\nabla_{W_{fk}^t} Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta}') = \sum_n h_{kn}^{t\rightharpoonup} \left(\frac{1}{\sum_{t'} W_{fk}^{t'} h_{kn}^{t'\rightharpoonup}} - \frac{\hat{v}'_{k,fn}}{(\sum_{t'} W_{fk}^{t'} h_{kn}^{t'\rightharpoonup})^2} \right) \tag{3.43}$$

En notant ∇^- la partie négative du gradient et ∇^+ la partie positive, on peut écrire les équations de mise à jour des paramètres : $\theta_k^{(it+1)} \leftarrow \theta_k^{(it)}.\frac{\nabla^-}{\nabla^+}$. On a alors :

$$h_{kn}^{(it+1)} \leftarrow h_{kn}^{(it)} \frac{\sum_f \sum_t \left(W_{fk}^t \frac{\hat{v}'_{k,f(n+t)}}{(\hat{v}_{k,f(n+t)}^{(it)})^2} \right)}{\sum_f \sum_t \left(\frac{W_{fk}^t}{\hat{v}_{k,f(n+t)}^{(it)}} \right)} \tag{3.44}$$

$$W_{fk}^{t(it+1)} \leftarrow W_{fk}^{t(it)} \frac{\sum_n \left(h_{kn}^{t\rightharpoonup} \frac{\hat{v}'_{k,fn}}{(\hat{v}_{k,fn}^{(it)})^2} \right)}{\sum_n \left(\frac{h_{kn}^{t\rightharpoonup}}{\hat{v}_{k,fn}^{(it)}} \right)} \tag{3.45}$$

Nous noterons ce modèle de décomposition non-négative, IS-NMD/EM.

3.4 Perspectives

Trois familles de méthodes de décomposition non-négative ont été présentées dans ce chapitre. Ces trois algorithmes et leurs versions avec des caractéristiques à deux dimensions peuvent permettent de classifier, d'isoler, de détecter des objets dans un texte ou une image par exemple.

L'étude de signaux musicaux nécessite souvent un prétraitement pour représenter les données de façon à pouvoir en extraire les informations pertinentes. Les signaux sonores sont donc transformés sous la forme d'une représentation temps-fréquence. Les méthodes de décomposition non-négative qui ont été détaillées peuvent parfaitement s'appliquer à ces représentations. Il reste à savoir quelle méthode est la plus adaptée au problème de transcription de la batterie.

La NMF et la NMD ont déjà fait leur preuve dans le traitement de la musique pour différentes tâches comme la séparation de sources. Pour la transcription de la batterie, ces méthodes doivent être utilisées sous leur forme informée. En effet, si tous les paramètres étaient libres, il serait difficile d'interpréter les bases et donc de transcrire les différents instruments. L'inconvénient de la NMF ou NMD avec dictionnaire fixe est l'impossibilité de s'adapter aux signaux étudiés. L'utilisation de méthodes probabilistes permettrait d'inclure facilement des informations a posteriori et donc provenant directement du signal étudié. On pourrait ainsi obtenir une flexibilité du dictionnaire pour mieux reconnaître les instruments de batterie.

— Chapitre 4 —

Les réseaux de neurones

4.1 Les réseaux de neurones

4.1.1 Un neurone

Le principe des réseaux de neurones provient au départ du fonctionnement des neurones biologiques et de l'entreprise de les modéliser. Un neurone artificiel est construit par analogie aux différentes parties d'un neurone biologique : les dentrites (entrées du neurone), les synapses (points de connexion), un axone (sortie) et le noyau (responsable de l'activation des sorties en fonction des entrées).

Comme pour le neurone biologique, les signaux d'entrée (dentrites), par exemple x_0 sont pondérés par des poids (synapses), W_0 pour x_0, qui permettent de moduler son influence. La combinaison de toutes les entrées modulées est finalement soumise à une étape de seuillage (noyau) réalisée grâce à une fonction d'activation pour générer la sortie (axone). Un neurone artificiel est représenté dans la figure 4.1.

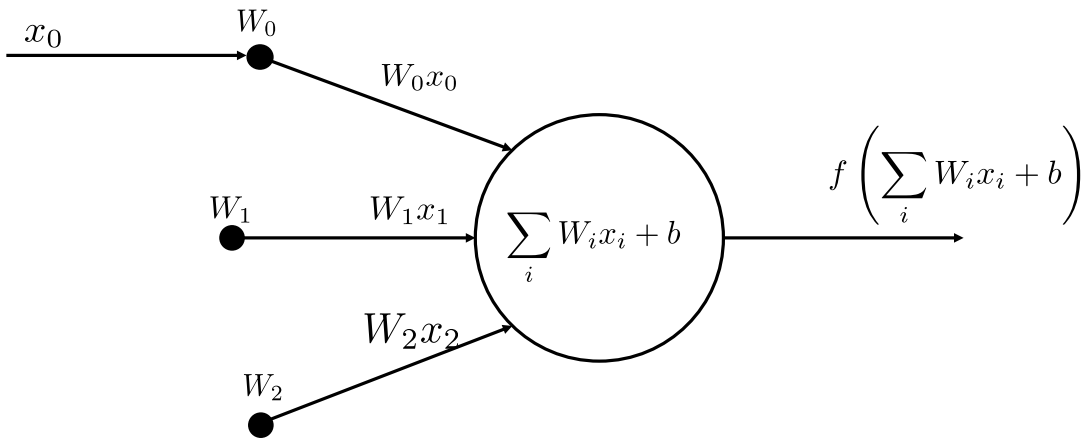

Figure 4.1 – Représentation d'un neurone artificiel.

Pour résumer, chaque neurone effectue un produit matriciel entre les poids et les en-

trées et applique parfois une fonction non-linéaire, la fonction d'activation pour produire sa sortie. Un biais b peut être ajouté après le produit matriciel. Cela permet de décaler de façon fictive le seuil de la fonction d'activation.

4.1.2 Les fonctions d'activation usuelles

Les fonctions d'activation permettent de définir un seuil qui, une fois atteint, entraîne l'activation du neurone. Historiquement, le choix de la fonction d'activation se porte sur la fonction sigmoïde. Cependant, d'autres fonctions d'activation semblent plus performantes selon les tâches. On présente ici une liste non exhaustive de fonctions d'activations les plus communes.

Fonction sigmoïde La fonction sigmoïde, de forme mathématique $\sigma(x) = 1/(1 + e^{-x})$, compresse les valeurs réelles à une plage de valeurs allant de 0 à 1. Ainsi, les valeurs fortement négatives sont associées à 0 alors que les valeurs très grandes et positives sont ramenées à 1. Pour ces valeurs très positives ou très négatives, le gradient devient nul, le réseau n'apprend donc pas avec ces données. La fonction sigmoïde est donc beaucoup moins utilisée aujourd'hui. Elle reste tout de même utilisée dans certains cas car la restriction des valeurs entre 0 et 1 permet un parallèle avec une mesure de probabilité.

Tanh La fonction tangente hyperbolique permet de donner à n'importe quelle valeur réelle une valeur entre -1 et 1. Contrairement à la fonction sigmoïde, la tangente hyperbolique est une fonction centrée en 0. On peut remarquer que tanh s'exprime en fonction de la fonction sigmoïde $\tanh(x) = 2\sigma(2x) - 1$.

ReLU La *Rectified Linear Unit* est donnée par la formule $f(x) = \max(0, x)$. Comparée aux fonctions d'activations sigmoïde ou tangente hyperbolique qui comportent des opérations coûteuse telles que e^{-x}, la fonction ReLU représente un simple seuillage à 0. Malheureusement les unités ReLU peuvent être fragiles : le gradient peut mettre à jour les poids tels que le neurone ne sera plus activé par aucune des données par la suite. Le gradient sera donc toujours nul pour ce neurone à partir de ce moment.

***Leaky* ReLU** La *leaky* ReLU essaie de résoudre le problème du "gradient mourant" énoncé ci-dessus. Elle s'énonce comme $f(x) = \mathbb{1}_{x<0}(ax) + \mathbb{1}_{x>=0}(x)$ avec a une constante de faible valeur. Elle autorise ainsi quelques valeurs négatives proches de 0.

4.1.3 Architecture d'un réseau de neurones

Les réseaux de neurones sont agencés sous forme de couches. La sortie d'un neurone peut devenir l'entrée d'un autre neurone et ainsi de suite pour former les couches. Les cycles ne sont cependant pas autorisés. Le type de couche le plus commun est la couche dense, *fully-connected layer*, pour laquelle tous les neurones de deux couches sont appariés entre eux mais aucune connexion sur la même couche n'est réalisée, voir figure 4.2 [1].

Par convention, la couche d'entrées n'est pas comptée lors de l'appellation d'un réseau. Par exemple, un réseau à N couches comportera N-1 couches dites cachées (*hidden layer*) et une couche de sortie (*output layer*) soit N couches et enfin une couche d'entrée (*input*

1. `http://cs231n.github.io/neural-networks-1/`

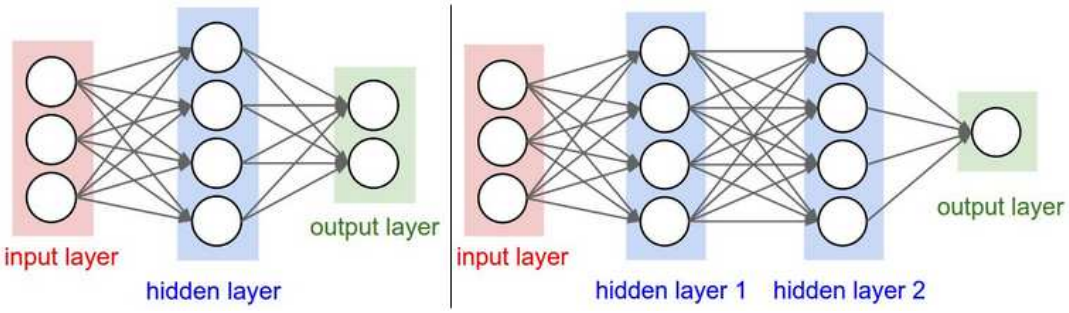

Figure 4.2 – *À gauche, un réseau à 2 couches denses.. À droite, un réseau à 3 couches denses.*

layer). Ainsi, un réseau à une couche ne contient aucune couche cachée et lie directement l'entrée à la sortie. Un réseau de neurones est caractérisé par son nombre de couches mais aussi par le nombre de paramètres (poids, biais) qu'il contient. Les valeurs de ces paramètres sont déterminées lors de la phase d'apprentissage. D'autres paramètres sont à fixer préalablement comme le nombre de couches ou le nombres de neurones par couche. On les appelle les hyper-paramètres. Ce sont tous les paramètres dont les valeurs doivent être fixées à la main.

4.2 Apprentissage des paramètres

Pour déterminer les paramètres optimaux du réseau pour la tâche qu'il doit remplir, une phase d'apprentissage est réalisée. Les mises à jour des paramètres se font à partir de la minimisation d'une fonction de coût entre la sortie générée et la sortie attendue. Pour optimiser les hyper-paramètres du réseaux et éviter des phénomènes non souhaités, plusieurs bases de données sont utilisées. Enfin, on peut ajouter des termes de régulation à la fonction de coût pour aider l'apprentissage.

4.2.1 Mise à jour des paramètres

Les règles de mise à jour des paramètres θ sont déterminées à partir du gradient $\mathcal{G}$ de la fonction de coût. En effet, l'architecture en couche permet l'utilisation d'un algorithme de rétropropagation, ou *backpropagation* en anglais, (Sutton, 1986) permettant de mettre à jour tous les paramètres du réseau en calculant leur gradient :

$$\Theta \leftarrow \Theta - \alpha.\mathcal{G} \tag{4.1}$$

avec α le pas d'avancement, *learning rate*.

L'hyper-paramètre *learning rate* doit être correctement choisi. En effet, s'il est choisi constant, la fonction de coût se rapproche d'un minimum mais ne converge pas totalement. Si sa valeur est grande, la fonction de coût va rapidement diminuer mais on ne

pourra pas approcher suffisamment du minimum et la fonction de coût risque même de diverger. Dans le cas contraire, plus il est choisi petit, plus on a de chance d'atteindre le minimum mais plus la vitesse de convergence est basse.

On peut changer la valeur du *learning rate* pendant l'entraînement. Plusieurs méthodes existent :

- **Momentum** - La descente de gradient a généralement du mal à naviguer dans des "ravins", c'est-à-dire dans une zone où la pente est beaucoup plus importante que dans d'autres zones. C'est souvent le cas autour d'un minimum local. Le gradient oscille entre les pentes et se dirige lentement vers l'optimum local. L'utilisation du momentum permet d'accélérer la descente de gradient dans la bonne direction.
- **Adagrad** - Adagrad adapte les mises à jour à chaque paramètre individuellement pour effectuer des mises à jour adaptées à l'importance du paramètres. Le plus gros avantage d'Adagrad est qu'il évite de définir manuellement le *learning rate*. Par contre, le *learning rate* devient de plus en plus petit à chaque itération et finit par être infiniment petit, à tel point que l'algorithme n'est plus capable d'acquérir de nouvelles connaissances.
- **Adadelta** - Adadelta une extension d'Adagrad. Adagrad prenait en compte tous les gradient qui précédaient. Adadelta se concentre sur les derniers gradients calculés.
- **Adam** (*Adaptive Moment*) - Adam est aujourd'hui la méthode la plus populaire. En plus, des carrés des gradients précédents (comme dans Adadelta), Adam utilise aussi la *exponential moving average* des gradients précédents.

4.2.2 Fonction de coût

Lors de l'apprentissage, des exemples dont on connaît la sortie que le réseau doit générer sont donnés en entrée. Le réseau en génère la sortie y qui est comparée à la sortie attendue $\tilde{y}$ avec un fonction de coût, appelée aussi *loss function*, $\mathcal{L}(y^{(i)}, \tilde{y}^{(i)})$, avec $.^{(i)}$ dénotant le i-ième exemple présenté au réseau.

Plusieurs fonctions de coût fréquemment utilisées sont présentées dans le tableau 4.1.

4.2.3 Les différentes bases de données

Plusieurs bases de données sont nécessaires pour optimiser et évaluer un réseau de neurones. Les paramètres sont optimisés sur une base d'apprentissage. Le réseau est alimenté par différents exemples d'entrée x dont la sortie y qu'il devrait générer est connue. Suivant la sortie que le réseau a générée, les paramètres sont modifiés grâce à une descente de gradient de la fonction de coût qui évalue la distance entre la sortie prédite par le réseau et la sortie attendue. Cette base de données est appelée base d'apprentissage.

À la fin de l'apprentissage, le réseau est évalué sur une base de données appelée base de test. Les exemples compris dans cette base n'ont jamais été présentés au réseau pendant l'apprentissage. Elle permet donc de tester la généralisation du réseau à des données inconnues.

Fonction de coût	$\mathcal{L}(y, \tilde{y})$
Mean Squared Error (MSE)	$\frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - \tilde{y}^{(i)})^2$
Mean Squared Logarithmic Error	$\frac{1}{n}\sum_{i=1}^{n}(\log(y^{(i)} + 1) - \log(\tilde{y}^{(i)} + 1))^2$
Mean Absolute Error	$\frac{1}{n}\sum_{i=1}^{n}\lvert y^{(i)} - \tilde{y}^{(i)}\rvert$
Mean Absolute Percentage Error	$\frac{1}{n}\sum_{i=1}^{n}\lvert\frac{y^{(i)} - \tilde{y}^{(i)}}{y^{(i)}}\rvert.100$
Norme $\ell 1$	$\sum_{i=1}^{n}\lvert y^{(i)} - \tilde{y}^{(i)}\rvert$
Divergence de KL	$\frac{1}{n}\sum_{i=1}^{n}(y^{(i)}(\log(y^{(i)}) - \log(\tilde{y}^{(i)})))$
Cross Entropy	$-\frac{1}{n}\sum_{i=1}^{n}(y^{(i)}\log(y^{(i)}) - (1 - y^{(i)})\log(1 - \tilde{y}^{(i)}))$
Negative Logarithmic Likelihood	$-\frac{1}{n}\sum_{i=1}^{n}\log(y^{(i)})$

Tableau 4.1 – *Fonctions de coût fréquemment utilisées pour les réseaux de neurones.*

En plus de l'optimisation des paramètres du réseau, les hyperparamètres doivent aussi être choisis avec soin. Le réglage de ces paramètres ne peut pas se faire sur la base de test puisque dans ce cas, les performances seraient biaisées. De même, les optimiser sur la base d'apprentissage n'aurait pas de sens puisque le réseau est appris pour faire correspondre parfaitement sa sortie à l'entrée. De plus, des problèmes de sur-apprentissage (*overfitting* en anglais) peuvent apparaître. En effet, si on oblige le réseau à répondre parfaitement à l'entrée donnée, il va être biaisé pour les autres données. Il sera trop spécialisé et ne pourra pas générer la bonne sortie par rapport à une entrée inconnue. L'optimisation des paramètres est alors réalisée par rapport à des propriétés trop caractéristiques aux données d'entraînement et ne permet pas la généralisation aux autres données. Pour éviter ce problème et déterminer les meilleurs hyperparamètres, une base est généralement extraite de la base d'apprentissage. C'est la base de validation. Les exemples ne sont donc pas utilisés pour l'entraînement. Les performances sont calculées sur cette base. Cela permet de vérifier qu'il n'y ait pas de sur-entraînement et de déterminer quels hyperparamètres donnent les meilleurs performances.

En général, la base d'apprentissage représente 80% des données et la base test, 20%. La base de validation est une petite partie de la base d'apprentissage soit 10 ou 20% de cette base.

Pour des bases de données petites, une autre stratégie peut être envisagée : la validation croisée (*cross-validation*). Au lieu d'extraire une base de validation de la base

d'apprentissage, cette dernière est divisée en N sous-bases. $N-1$ sous-bases sont utilisées pour l'entraînement et le réseau est validé sur la dernière sous-base. Ces entraînements sont répétés avec toutes les combinaisons possibles pour finalement faire une moyenne de toutes les performances.

4.2.4 Régularisation

Il peut être intéressant d'empêcher l'apparition du sur-apprentissage. On présente ici quelques stratégies possibles :
Régularisation par la norme $\ell 2$ - C'est sûrement la régularisation la plus commune. Un terme de pénalisation pour chaque poids est ajouté à la *loss function*. À cause des interactions multiplicatives entre les poids, cela force le réseau à n'utiliser qu'un peu toutes les entrées plutôt qu'un petit nombre.
Régularisation par la norme $\ell 1$ - Contrairement à la norme $\ell 2$, la régularisation par la norme $\ell 1$ pousse les neurones à n'utiliser qu'un petit nombre de leurs entrées.
Dropout - Le *dropout* est une technique apparue récemment, simple et qui s'avère très efficace. Elle consiste à annuler un certains nombre de sorties de neurone. Les neurones gardés actifs sont déterminés par une probabilité p qui est donc un hyperparamètre du réseau. Une couche dense lie la majorité des paramètres de deux couches entre eux et par conséquent, les neurones développent de la co-dépendance entre eux. Cela peut conduire à un sur-apprentissage des données d'entraînement. Ainsi, la régularisation basée sur le *dropout* permet au réseau d'apprendre des caractéristiques plus robustes. En effet, couper aléatoirement des unités du réseau rend plus difficile pour les unités de se compenser. Chaque unité du réseau produit une erreur et l'erreur d'une unité pourrait être compensée par l'erreur d'une autre. Cependant, il augmente le nombre d'itérations d'apprentissage nécessaire pour obtenir la convergence.

4.3 Différentes familles de réseaux de neurones

4.3.1 Principe d'un réseau de neurones récurrent

Dans certains cas, les données d'entrées ne sont pas indépendantes les unes des autres. Par exemple, pour prédire un mot dans une phrase, il faut connaître le ou les mot(s) qui le précédent. Un réseau de neurones classique ne peut pas prendre en compte les données qu'il a déjà traitées pour prendre une décision sur la donnée qui lui est présentée. Une extension des DNN permettent d'établir des liens avec les données précédemment traitées : les *Recurrent Neural Network*, réseau de neurones récurrent.

Les RNN sont définies par une unique couche qui est répétée le nombre de fois nécessaires. Cette couche reçoit en entrée la nouvelle donnée ainsi que la sortie de la couche à l'étape précédente. Les données de sortie dépendent donc de plusieurs entrées et de leurs traitements. En d'autres termes, un RNN présente une "mémoire" qui enregistre ce qui a été calculé jusqu'à l'étape où il se trouve.

La figure 4.3 [2] représente un RNN et sa représentation déroulée. Les notations sont

2. http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

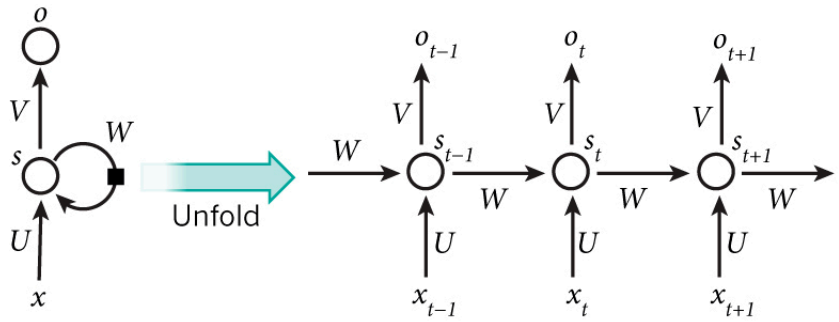

Figure 4.3 – *Recurrent Neural Network.*

les suivantes :

- x_t est l'entrée au temps t.
- s_t est l'état caché au temps t. C'est ce qui représente la "mémoire" du réseau. Il est donné en fonction de l'état caché précédent et l'entrée au temps t : $s_t = f(U.x_t + W.s_{t-1})$ où la fonction d'activation f est non-linéaire.
- o_t est la sortie au temps t.

Cette architecture permet de répondre au problème de disparition du gradient, *gradient vanishing* en anglais. Cela a pour conséquence de faire "oublier" au réseau les premières informations qu'il a rencontrées même si celles-ci pourraient être utiles au moment présent t. Finalement, le réseau ne prend pas en compte toutes les données précédentes mais seulement les dernières.

Certaines techniques ont été mises au point pour permettre de minimiser ce problème, par exemple les LSTM. L'idée est d'enregistrer les informations les plus importantes et d'oublier celles qui le sont moins. Le réseau est alors capable d'utiliser des informations plus anciennes.

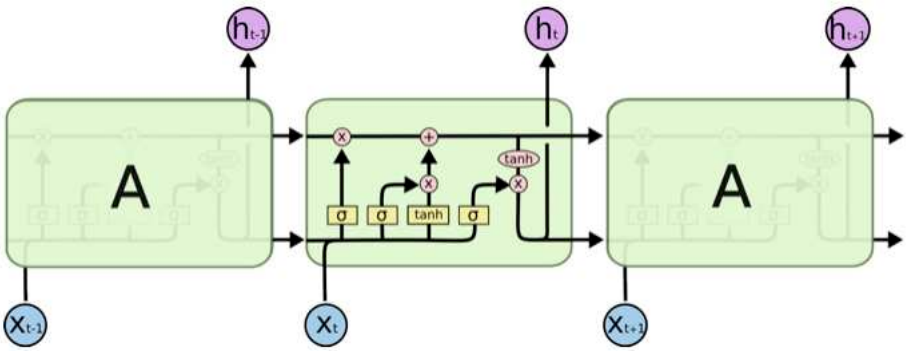

Figure 4.4 – *Cellule d'un LSTM*

La figure 4.4[1] décrit l'architecture de la cellule d'un LSTM. Les lignes noires repré-

sentent le chemin des vecteurs, les points rouges sont des opérations élément par élément et les boîtes jaunes sont les couches du réseau de neurones apprises lors de l'entraînement. Lorsque les lignes noires se rejoignent, les vecteurs sont concaténés et lorsqu'elles se séparent, ils sont copiés.

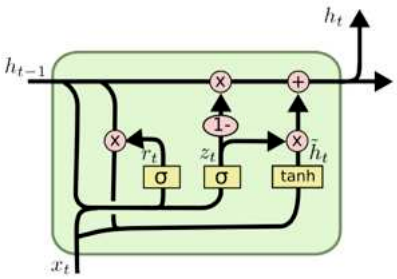

FIGURE 4.5 – *Cellule d'un GRU*

Les GRU, présentés en figure 4.5[1], sont une version des LSTM proposée par (Cho et al., 2014). Ils sont basés sur ce qu'on appelle les *update gates* et les *reset gates*. Les *update gates* aident le modèle à savoir quels informations garder pour les renseigner aux futures étapes. Les *reset gates* donnent moins d'importance aux données qui ne sont pas nécessaires pour la suite.

4.3.2 Principe d'un réseau de neurones convolutif

Les réseaux de neurones convolutifs sont très similaires aux réseaux décrits dans la section 4.1.3. Leur particularité est qu'ils ont été conçus pour traiter des images en entrée. Les entrées sont donc convoluées est plus multipliées.

Dans un réseau de neurones classique, chaque neurone d'une couche dense est connecté à tous les neurones de la couche précédente et ne présente aucune connexion avec les neurones de sa propre couche. Si l'entrée est une image codée sur trois canaux de couleur, la taille d'un neurone de la première couche serait *la largeur de l'image* × *la hauteur de l'image* × *les trois canaux de couleur*. Plus l'image est grande, plus le nombre de paramètres augmente. De plus, le réseau est constitué de plusieurs neurones.

Un neurone d'un CNN est arrangé en 3 dimensions : la largeur, la hauteur et la profondeur. Ce neurone permet de traiter une petite aire de l'image. Ainsi, il traite plus d'information de l'image avec moins de paramètres.

Un CNN est un réseau de neurones qui contient au moins une couche convolutive. Pour former un CNN quatre types de couches sont généralement utilisés :

- Couche de convolution : La sortie de ses neurones est le produit matriciel d'une petite partie de l'image (entrée du neurone) avec leurs poids. On appelle leurs neurones des filtres de convolution et leurs sorties, les *feature maps*.

1. source : `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`

- Couche d'activation : Elle applique à chaque élément une fonction d'activation, par exemple $\text{ReLU}(x) = max(0, x)$ soit un seuillage à 0.
- Couche de *pooling* (optionnel) : Elle réalise un sous-échantillonnage suivant une ou plusieurs dimensions. Elle permet de réduire le risque de sur-apprentissage et réduit le nombre de paramètres à apprendre. Aucun paramètre n'est à apprendre pour cette couche.
- Couche dense (optionnel) : Elle remplit les mêmes fonctions qui sont expliquées en section 4.1.3 sur les *feature maps* de la dernière couche concaténée en vecteur.

Troisième partie

Applications à la transcription automatique de la batterie

— Chapitre 5 —

Comparaison de modèles de décomposition non-négative pour la transcription automatique de la batterie

Comme il a été mentionné, la transcription de la musique est complexe et nos études se concentrent sur la transcription de la batterie. L'objectif est de répondre à deux questions : à quel moment un événement percussif à lieu et quel instrument en est responsable ?

Dans la musique occidentale, trois instruments de la batterie sont responsables de la majorité des rythmes : la grosse-caisse, la caisse-claire et la Charleston (*Bass Drum* (BD), *Snare Drum* (SD) et *Hi-Hat* (HH) en anglais). Ils sont en général essentiels pour la trame rythmique de base. De plus, le problème de transcription de la batterie est de plus en plus complexe à mesure où des classes d'instrument de batterie sont ajoutées. La transcription de ces trois instruments n'est pas encore totalement opérationnelle. Ainsi, lorsqu'on parlera de transcription de la batterie dans ce document, on fera référence à la transcription des trois principaux instruments de batterie cités ci-dessus.

Dans ce chapitre, on décrit un algorithme de transcription de batterie basé sur les décompositions en matrice non-négative. L'algorithme est basé originalement sur l'utilisation de la NMD. Dans cette méthode les bases sont fixes et ne s'adaptent pas au signal. Or, le nombre d'exemples dans le dictionnaire est limité et ils ne peuvent pas décrire tous les modèles ou les façons de jouer d'un instrument. On veut pouvoir ajouter de la liberté aux bases pour qu'elle puisse s'adapter au signal étudié. Pour cela, on compare la NMD à deux méthodes probabilistes dont une que l'on a adaptée aux motifs à deux dimensions dans le Chapitre 3 : la SI-PLCA et la IS-NMD/EM. Ces dernières peuvent facilement introduire des connaissances a priori et a posteriori qui pourraient permettre cette adaptation.

Dans un premier temps, on présente l'algorithme dans le lequel on introduira les différentes méthodes de décomposition non-négative. Cet algorithme nécessite une pré-étape d'apprentissage pour composer le dictionnaire des bases. Lors de la transcription, il traite le signal pour le mettre sous la forme adéquate, puis applique la méthode de décomposition non-négative afin d'apprendre les vecteurs d'activations. Enfin, après des étapes

de seuillage, l'algorithme fournit les positions des événements de chaque instrument. Les résultats de la comparaison permettront de choisir quelle méthode de décomposition non-négative choisir.

5.1 État de l'art : les modèles de décomposition non-négatifs pour la transcription de la batterie

(FitzGerald and Paulus, 2006) propose un panorama des premières approches pour la transcription de la batterie en les classant sous trois approches : une basée sur la séparation, une autre sur la segmentation et enfin une sur les modèles musicaux.

L'approche basée sur les modèles musicaux introduit un modèle statistique afin de prendre en compte les dépendances entre les séquences d'événement. Ces méthodes ont déjà fait leur preuve pour le traitement de la parole.

Celles basées sur la segmentation découpent le signal en segments puis les analysent indépendamment des autres comme dans (Miron et al., 2013b) ou (Gillet and Richard, 2008). Ces méthodes suivent à peu près toutes le même schéma :

- Segmenter le signal d'entrée soit en localisant les événements soit en appliquant une grille temporelle adéquate,
- Extraire un ensemble de caractéristiques de chaque segment,
- Classifier le contenu des segments grâce à ces caractéristiques,
- Combiner les informations obtenues avec l'échelle de temps pour obtenir la transcription.

Enfin, les méthodes basées sur la séparation cherchent quant à elle à séparer les différentes voix présentes dans un morceau de musique en s'appuyant sur des méthodes de séparation de sources. Différents algorithmes peuvent être utilisés : l'ICA, le plus commun, la NMF et ses variantes, ...

C'est dans (Smaragdis and Brown, 2003) que la NMF est utilisée pour la première fois pour la transcription de la musique. Ils montrent que la transcription automatique de la musique peut théoriquement atteindre de bons résultats en utilisant la NMF mais n'évaluent pas leur algorithme. Par la suite, plusieurs variations de la NMF ont été proposées pour la transcription de la batterie.

Comme détaillé en sec. 3.1.1, la NMF décompose une entrée en un dictionnaire et les activations correspondantes. L'initialisation de ce dictionnaire est déjà un problème en soi. Un premier questionnement est le nombre de bases par instrument. En général, pour la transcription de la musique, chaque élément du dictionnaire correspond à un instrument en particulier ou à une note dans le cas d'une transcription harmonique. On peut trouver d'autres formes de dictionnaire comme dans (Grindlay and Ellis, 2009) où le dictionnaire représente des *eigeninstruments*. L'instrument est représenté par une combinaison des différents *eigeninstruments*. L'important est de garder l'interprétation physique des différentes bases permettant la reconnaissance des sources pour les transcrire.

Certains auteurs comme (Battenberg, 2012) ou (Dittmar and Gärtner, 2014) initialisent puis fixent le dictionnaire pendant la décomposition. Le problème d'optimisation

devient alors convexe. Cependant, ce modèle pose problème si des composantes différentes des bases apprises sont présentent dans le morceau analysé.

Une variante de la NMF, appelée *Partially-Fixed* NMF (Wu and Lerch, 2015a) permet de prendre en compte la présence d'autres composantes que la batterie. En plus des bases fixées, l'ajout de bases supplémentaires initialisées aléatoirement et mises à jour pendant la décomposition peut permettre de décrire ce qui n'est pas de la batterie.

Pour la transcription de la musique, les bases représentent souvent le spectre d'une note. Mais il se peut que certains motifs présentent une certaine invariabilité temporelle. C'est le cas pour la batterie. En effet, les différents instruments de batterie présentent la même réponse temporelle d'un événement à l'autre. Il est possible d'inclure cette propriété directement dans le dictionnaire. Les bases représentants ces motifs prennent en compte la réponse temporelle des événements et présentent donc deux dimensions, la fréquence et le temps. L'étape de décomposition n'est plus une factorisation mais une déconvolution (Röbel et al., 2015). La méthode de décomposition est alors la NMD, détaillée en sec. 3.1.4.

Les méthodes citées précédemment sont des méthodes déterministes mais des méthodes de décomposition statistiques sont aussi utilisées dans l'analyse musicale. (Bello and Weiss, 2010) utilise la SI-PLCA, introduite par (Smaragdis and Raj, 2007) et détaillée à la sec. 3.2.2, pour repérer la structure du morceau et le segmenter. (Benetos and Dixon, 2012) ainsi que (Fuentes, 2013) l'appliquent à la transcription de la musique. (Bertin, 2009) propose un modèle statistique de la NMF, détaillée dans là a section 3.3.1.

5.2 Algorithme de transcription automatique de la batterie

L'algorithme de transcription de la batterie de (Röbel et al., 2015) décrit dans ce chapitre et schématisé sur la figure 5.2 utilise les modèles de décomposition non-négative des données. Il est décomposé en différentes parties. La première étape est une étape à réaliser préalablement. Elle consiste à élaborer un dictionnaire de motifs de référence pour les algorithmes de décomposition. Une fois le dictionnaire appris, l'algorithme peut être appliqué à n'importe quel morceau. Il consistera en un prétraitement du signal pour le représenter sous la forme adéquate puis d'une étape de décomposition pour ensuite pouvoir détecter les événements relatifs à la batterie et enfin la transcrire.

5.2.1 Apprentissage des bases des sources cibles

L'apprentissage du dictionnaire a pour but d'élaborer un ensemble de bases caratéristiques pour chaque instrument cible. La réponse temporelle des événements de batterie constitue une propriété extrêmement importante pour la détection et la reconnaissance de l'instrument. Les motifs prenant en compte la réponse temporelle semblent plus adaptés pour décrire ces événements. Les différents algorithmes qui sont étudiés dans cette section utilisent donc des dictionnaires constitués de bases à deux dimensions. Ces bases sont des représentations temps-fréquence des motifs.

Pour apprendre les bases, des exemples isolés de chaque instrument sont nécessaires. Pour chacun des instruments, un signal annoté comportant N_{ins} exemples de coups de

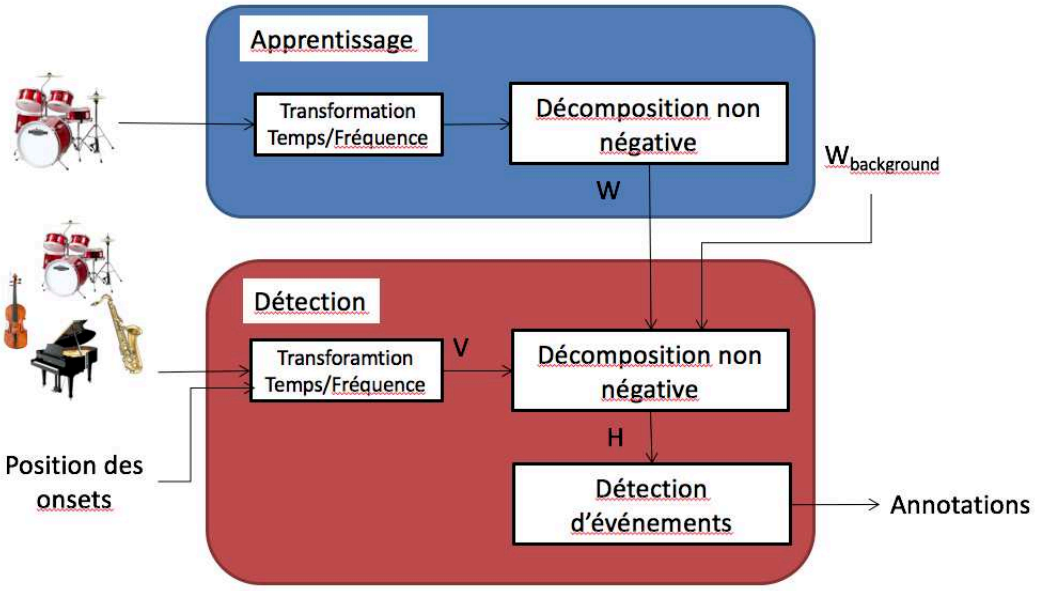

FIGURE 5.1 – *Schéma de l'algorithme de transcription de la batterie basé sur une méthode de décomposition en matrices non-négatives*

batterie provenant d'enregistrements différents est disponible. C'est dans ce signal que les bases seront sélectionnées. Le signal est d'abord converti sous la forme d'une représentation temps-fréquence, ici, sous la forme d'un spectrogramme mel. Les N_{ins} motifs sont alors des candidats pour intégrer le dictionnaire.

La première base ajoutée au dictionnaire est la moyenne de tous les événements présents dans le signal contenant les exemples. Le signal des événements devient le signal cible. Les positions des événements étant connues, le vecteur d'activations de la base sélectionnée est initialisé à 1 au début de chaque événement et à 0 ailleurs. Puis une décomposition en gardant la base fixe est réalisée pour apprendre le vecteur d'activation permettant d'approcher au mieux le signal cible. Finalement, la reconstruction est effectuée en calculant la convolution de ce vecteur avec le motif. Chaque événement reconstruit est comparé à l'événement correspondant du signal cible par le calcul d'une divergence (ici, la divergence d'IS, voir section 3.1.2). Le motif pour lequel la reconstruction est la plus éloignée est ajoutée au dictionnaire.

Il y a maintenant deux motifs au dictionnaire et donc deux vecteurs d'activation. L'initialisation des vecteurs d'activation est maintenant de $\frac{1}{2}$ à chaque début d'événement et 0 ailleurs. La décomposition est réalisée avec ses deux bases gardées fixes et la reconstruction est calculée. Chaque événement est comparé au signal d'origine. Le plus éloigné est ajouté au dictionnaire.

Ces étapes sont répétées autant de fois que nécessaire. Pour l'apprentissage de la $(k+1)^e$ bases, les vecteurs d'activation sont initialisés à $\frac{1}{k}$ au début de chaque événement et 0 ailleurs. Une fois que l'on atteint le nombre de bases voulu ou que l'on est satisfait de

la concordance des observations et des estimations, on stoppe les itérations. On obtient alors l'ensemble des bases pour l'instrument. Une fois tous les instruments traités, le dictionnaire est prêt.

Dans ce chapitre, on sélectionne 15 bases par instruments sauf pour la Charleston où on en sélectionne 15 pour la version ouverte et 15 pour la version fermée. Le dictionnaire contient donc 60 bases pour les instruments de batterie.

5.2.2 Étape de décomposition

L'algorithme est résumé dans la figure 5.2.

5.2.2.1 Prétraitement du signal et ajout d'éléments au dictionnaire

Les algorithmes de décomposition non-négative peuvent être très coûteux appliqués à de longs signaux. Appliquer à des segments plus courts, l'algorithme peut être beaucoup moins coûteux. Le signal peut être découpé en segments. Pour éviter qu'un événement apparaisse sur la bordure entre deux segments, il faudrait utiliser des segments qui se chevauchent. Le temps de calcul risque d'alors d'augmenter par rapport au temps d'application de l'algorithme sur le signal complet. Pour réduire les coûts de calcul, les segments sont choisis. L'algorithme de détection d'*onsets* de (Elowsson and Friberg, 2013), basé sur le flux spectral, est appliqué au signal étudié et seuls les segments autour de ces événements sont analysés.

Le signal des parties sélectionnées est mis sous la forme d'une représentation temps-fréquence, ici d'un spectrogramme mel. Le dictionnaire appris préalablement permet de réaliser la décomposition du segment. Cependant, le signal musical à décomposer peut contenir d'autres instruments. Or le dictionnaire possède seulement des bases permettant de décrire les trois instruments principaux de la batterie. Les événements relatifs aux autres instruments n'ont pas d'autres possibilités que d'être décrits par les bases de batterie et être considérés comme de la batterie par la suite. Cela engendre des faux positifs. Pour éviter ce problème des bases supplémentaires sont ajoutées au dictionnaire.

Ces bases, appelées bases de *background*, sont ajoutées pour décrire tout ce qui n'est pas de la batterie au sein du segment analysé. Les segments étant choisis de longueur égale à trois fois la taille d'une base, trois bases de *background* sont nécessaires. Chaque base est dédiée à une partie du segment. Leurs activations sont initialisées à 1 à l'endroit où le motif débute et à 0 ailleurs. Ces trois bases sont initialisées en réalisant une décomposition du segment avec uniquement ces trois bases. Le problème est que les bases doivent décrire tout ce qui n'est pas de la batterie dans le segment. Cependant, elles ne doivent pas pouvoir représenter les instruments de la batterie. Il faudra donc pénaliser leurs activations afin que les événements de batterie soient bien décrits par les bases qui leur sont dédiées.

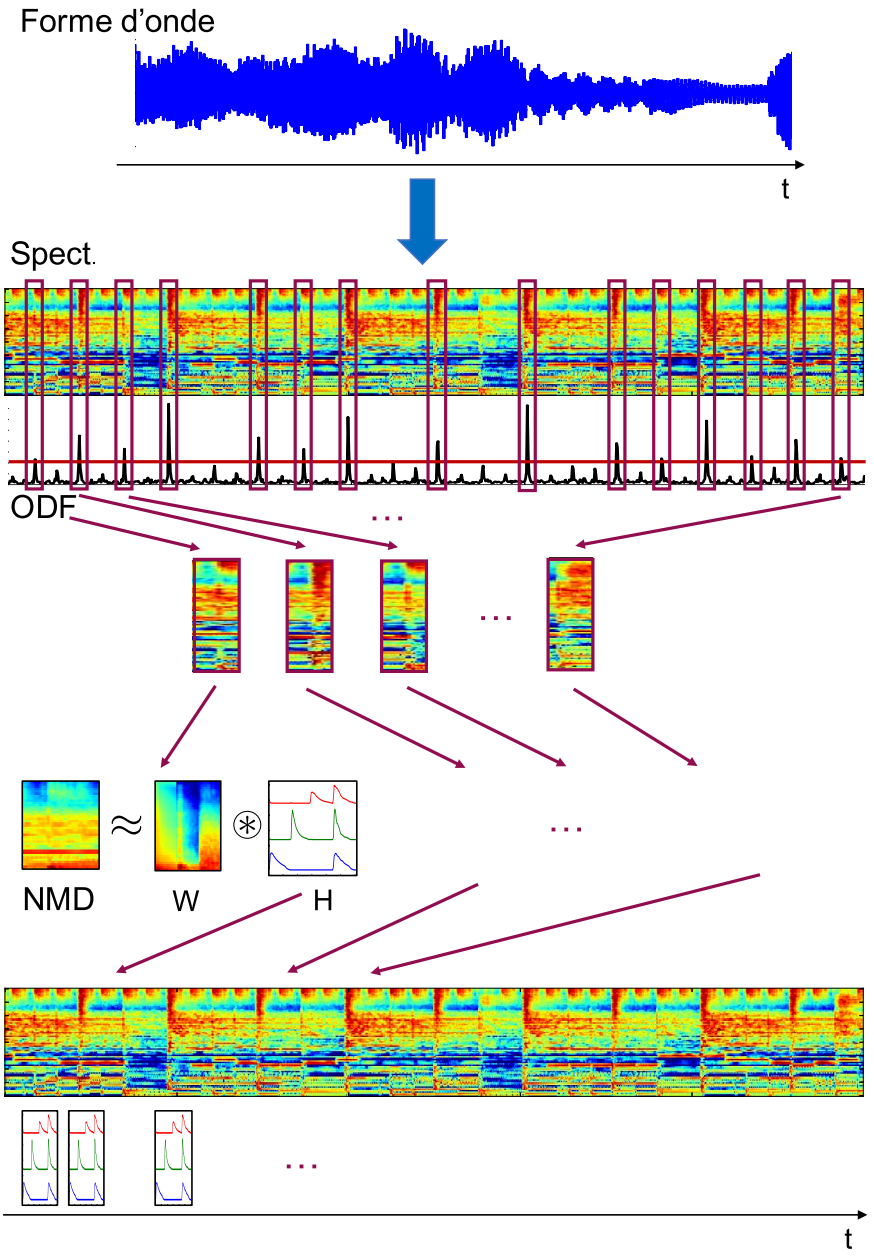

FIGURE 5.2 – *Résumé de l'algorithme de transcription de la batterie avec décomposition non-négative*

5.2.2.2 Décomposition

Le dictionnaire est finalement composé des bases de batterie issues de l'apprentissage et des bases de *background*. Les bases relatives à la batterie sont fixes tout au long

de la décomposition. Les bases de *background* sont mises à jour à chaque itération pour s'adapter au signal. Les bases de batterie sont communes à tous les segments alors que les bases de *background* sont initialisées pour chaque segment. La matrice d'activation $\boldsymbol{H}$ est initialisée en utilisant l'énergie à chaque trame du segment. Le segment est décomposé en matrices non-négatives avec un des algorithmes suivants : NMD (section 3.1.4), SI-PLCA (section 3.2.2) ou IS-NMD/EM (section 3.3.2).

La décomposition du segment est réalisée par la détermination des fonctions d'activations correspondantes aux différentes bases du dictionnaire. Comme les bases sont normalisées, les activations nous renseignent sur quelles bases sont les plus activées à l'instant t et nous permettent de réaliser la transcription de chaque instrument.

5.2.2.3 Etape de seuillage

Une fois les activations de chaque base déterminées, on somme à chaque trame temporelle les activations des bases relatives à un même instrument. On obtient ensuite un vecteur d'activation par instrument.

On applique à chaque fonction d'activation une convolution avec le noyau $[0.9, 1, 0.9]$ pour pouvoir prendre en compte les *onsets* qui pourraient apparaître entre les trames temporelles. On note $\boldsymbol{H}_c(n)$ la nouvelle fonction d'activation avec n la trame temporelle et c la classe de l'instrument.

Plusieurs seuils sont considérés pour détecter les événements de la batterie (Röbel et al., 2015). Premièrement, seules les trames où $\boldsymbol{H}_c(n)$ présentent un maximum local sont retenues. Les activations doivent alors vérifier $\boldsymbol{H}_c(n) \geq \mu_P \hat{\boldsymbol{H}}_c$ avec $\hat{\boldsymbol{H}}_c$ la médiane et μ_P un facteur positif. Cela permet de repérer les endroits où les maxima locaux apparaissent.

Deuxièmement, la somme de toutes les activations est calculée puis filtrée par une fenêtre rectangle pour être lissée. On obtient approximativement l'énergie du signal $\boldsymbol{H}_F(n)$. Un deuxième seuil μ_F est alors appliqué et les trames telles que $\boldsymbol{H}_c(n) \geq \mu_F \boldsymbol{H}_F(n)$ sont retenues. Ce critère établit un rapport signal sur bruit minimum pour les événements détectés.

Enfin, un critère de perception est ajouté pour vérifier que la partie de l'instrument occupe une place assez importante par rapport à l'énergie présente dans le spectrogramme. Le spectre de puissance est resynthétisé pour la classe d'instrument cible tel que

$$P(k,n) = \frac{F(n) \times v_c(k,n)^2 / \hat{v}(k,n)}{F(n) \times v_c(k,n)} \tag{5.1}$$

avec $P(k,n)$ l'énergie relative moyenne au point k et temps n, $F(n)$ une fenêtre de lissage rectangulaire, $v_c(k,n)$ les puissances au point fréquentiel k et trames temporels n du spectrogrammes observé et $\hat{v}_c(k,n)$ le spectrogramme estimé. La moyenne des trois plus grandes valeurs de $P(k,n)$ sur les bandes de fréquences est comparée au seuil μ_{PE}.

Si ces trois conditions sont remplies, la trame étudiée contient un *onset* de l'instrument cible. Sinon ce n'est pas le cas.

5.2.3 Contrainte d'activation du *background*

Les bases de *background* sont disponibles dans le dictionnaire pour décrire tout ce qui n'est pas relatif à la batterie. Elles sont adaptées au signal au fur et à mesure de la décomposition. Du fait de leur initialisation, ces bases peuvent décrire entièrement le segment. L'algorithme de décomposition en matrices non-négatives aura tendance à n'utiliser que ces bases puisqu'alors la reconstruction sera parfaite et la fonction de coût nulle. Les bases relatives aux instruments de batterie ne seront alors pas activées et la transcription sera impossible.

Les bases de *background* sont ajoutées pour décrire ce qui n'est pas relatif à la batterie donc pour compléter la décomposition produite avec le dictionnaire de la batterie. Pour inciter l'algorithme à utiliser prioritairement les bases du dictionnaire appris, l'activation des bases de *background* est pénalisée. Pour cela un terme de pénalité est ajouté à la fonction de coût modulé par un facteur de pondération λ_{seg}. Dans l'algorithme d'origine de (Röbel et al., 2015), le terme de pénalité est basé sur la norme $\ell 1$. On adapte cette pénalité aux deux méthodes probabilistes de décomposition non-négative, la SI-PLCA et la IS-NMD/EM pour pouvoir comparer leurs performances avec et sans pénalité à l'algorithme utilisant la NMD. Les trois pénalisations sont détaillées ci-dessous.

5.2.3.1 NMD

Pour la NMD, un terme de pénalité P est ajouté à la fonction de coût. Il est basé sur la norme $\ell 1$.

$$\begin{aligned} C(\boldsymbol{W},\boldsymbol{H}) &= d_{IS}(\boldsymbol{V}|\tilde{\boldsymbol{V}}) + \lambda_{seg} P(\boldsymbol{H}) \\ &= d_{IS}(\boldsymbol{V}|\tilde{\boldsymbol{V}}) + \lambda_{seg} \sum_{bg,n} \boldsymbol{H}_{bg,n} \end{aligned} \tag{5.2}$$

avec n la trame temporelle et bg les indices des bases de *background*.

5.2.3.2 SI-PLCA

Dans le cas de la SI-PLCA, la pénalité est exprimée sous la forme d'une distribution de probabilité a priori :

$$Pr(\Lambda) = Pr(P(z,t)) = Pr(P(z)P(t|z)) = \prod_{z_{bg},t|z_{bg}} e^{\frac{-|P(z)P(t|z)|}{1/\lambda_{seg}}} \tag{5.3}$$

avec bg les indices des bases de *background*.

5.2.3.3 IS-NMD/EM

La contrainte s'exprime sous la forme d'une distribution de Laplace :

$$p(\boldsymbol{H}) = \prod_{k \in bg} e^{-\frac{|\boldsymbol{H}_k|}{1/\lambda_{hseg}}} \tag{5.4}$$

où $\boldsymbol{H}_k$ représente la k^e colonne de $\boldsymbol{H}$ et bg l'ensemble des indices des bases contraintes.

Morceaux	Genres	Durée	Nombre d'*onsets*		
			BD	SD	HH
RM001	Rock	05 :49	443	343	662
RM007	Rock	04 :12	372	248	889
RM008	Rock	03 :28	308	189	345
RM012	Jazz	03 :22	534	284	465

Tableau 5.1 – *Détails des morceaux utilisés de Music Genre Database.*

5.3 Résultats de la comparaison des algorithmes de décomposition non-négative

L'algorithme de transcription de la batterie de (Röbel et al., 2015) est utilisé seulement avec la NMD. L'utilisation de la NMD présente déjà de bons résultats, (Röbel et al., 2015), mais est théoriquement moins flexible que les méthodes probabilistes. Nous remplaçons la NMD par la SI-PLCA et la IS-NMD/EM, deux méthodes de décomposition probabilistes présentées au chapitre 3. Nous comparons leur performances sur un extrait de la base de données *Music Genre Database* et les résultats sont donnés ci-dessous ainsi que leurs performances avec pénalisation de l'utilisation du *background*.

5.3.1 Base de données

Pour évaluer et comparer les performances des algorithmes basés sur les méthodes de décomposition non-négative, un échantillon de la base de données RWC *Music Genre Database* 2.1.2.1 est extrait. Quatre morceaux sont sélectionnés : trois morceaux rock et un morceau jazz. Les quatre morceaux sont synthétisés à partir du format MIDI.

Malgré la segmentation, l'analyse d'un morceau complet prend beaucoup de temps. On les compare tout d'abord sur des extraits de 30 secondes des morceaux cités dans le tableau 5.1. Ces extraits sont choisis de façon à ce que les trois instruments principaux de la batterie y soient présents.

5.3.2 Résultats

Trois méthodes de décomposition non-négative et des variantes de deux d'entre elles sont intégrées à l'algorithme décrit à la section 5.2.2 : la NMD avec pénalisation du *background* qui est la méthode utilisée dans l'algorithme d'origine (Röbel et al., 2015), la SI-PLCA avec ou sans pénalisation du *background* et la IS-NMD/EM avec ou sans pénalisation. Ces pénalisations sont pondérées par un facteur λ_{seg}, voir section 5.2.3. Les meilleurs résultats pour la NMD et la SI-PLCA ont été obtenus avec λ_{hseg} égal à 10. Pour la IS-NMD/EM, il est tout d'abord fixé également à 10.

Les *onsets* détectés par les algorithmes sont considérés comme correctement détectés si la différence entre l'*onset* détecté et l'annotation correspondante n'excède pas 30 ms, c'est-à-dire qu'il se trouve dans un intervalle de [−30 ms ; 30 ms] de l'annotation.

Algorithmes	BD			SD			HH		
	P	R	F	P	R	F	P	R	F
NMD+penal. (original)	84.7	76.7	80.5	89.1	88.4	88.7	70.5	85.1	77.1
SI-PLCA	84.4	77.7	80.9	84.6	83.3	83.9	82.2	78.7	80.4
SI-PLCA+penal.	84.4	77.7	80.9	82.5	85.5	84.0	79.8	78.4	79.1
IS-NMD/EM	54.0	79.2	64.2	64.7	91.1	75.7	84.6	75.4	79.7
IS-NMD/EM+penal.	58.7	80.2	67.8	64.9	88.4	74.8	53.9	79.7	64.3

TABLEAU 5.2 – *Résultats de transcription de la batterie avec les algorithmes basés sur les décompositions non-négative sur la base extraite de Music Genre Database.*

La SI-PLCA, obtient des résultats meilleurs pour la grosse-caisse (BD) et la Charleston (HH) que la NMD. Cependant, la F-mesure diminue fortement pour la caisse-claire (SD). Malgré une recherche de la meilleure pondération pour la pénalisation, les résultats pour la caisse-claire restent très bas par rapport à ceux obtenus avec la NMD.

Les résultats de la IS-NMD/EM sont très éloignés des résultats obtenus avec la NMD et la SI-PLCA. Les calculs se faisant composante après composante, les temps de calcul de la décomposition avec la IS-NMD/EM sont très longs. L'analyse de l'ensemble des segments des quatre extraits durent environ 13 h pour la IS-NMD/EM contre moins de 2 h pour les deux autres algorithmes de décomposition. Il est donc compliqué d'optimiser cet algorithme notamment avec un choix de pondération des contraintes adéquat. Nous avons décidé de ne pas chercher à l'optimiser.

Bien que l'algorithme avec la SI-PLCA présente de meilleures performances pour la grosse-caisse et la Charleston, les points perdus pour la caisse-claire sont trop importants. C'est l'algorithme NMD qui est donc retenu pour l'algorithme de transcription de la batterie.

5.4 Conclusion

Ce chapitre présente l'algorithme de transcription de la batterie original basé sur la NMD, extension de la NMF aux motifs présentant des réponses temporelles caractéristiques. Cet algorithme consiste en deux étapes distinctes : l'apprentissage du dictionnaire du motif et ensuite les analyses des différents morceaux. Lors de cette analyse, le dictionnaire de motifs ne présente aucune flexibilité. L'intégration d'informations a posteriori est difficilement possible.

L'utilisation de méthodes de décomposition probabilistes pourraient permettre l'intégration de connaissances acquises pendant l'analyse. Deux méthodes de décomposition non-négative probabilistes, la SI-PLCA et la IS-NMD/EM ont été intégrées à l'algorithme de transcription de la batterie afin d'être comparées à l'algorithme original basé sur la NMD. La représentation des éléments qui ne sont pas de la batterie se fait toujours par l'ajout de bases libres dont l'utilisation est pénalisée. L'ajout de contrainte est nécessaire aux méthodes probabilistes.

La SI-PLCA a obtenu des résultats proches voir meilleurs dans certains cas que la NMD. Cependant, pour la caisse-claire, les résultats sont nettement en deça de la NMD. La IS-NMD/EM, adaptée de la IS-NMF/EM a une vitesse de convergence très lente. L'analyse d'un morceau prend environ 6 fois plus de temps avec la IS-NMD/EM qu'avec la NMD. L'optimisation de cet algorithme n'a pas été envisagé. La méthode de décomposition retenue est finalement la NMD. Le cadre probabiliste n'est pas indispensable pour introduire des informations supplémentaires durant la décomposition et l'analyse peut être adapté aux signaux étudiés.

— Chapitre 6 —

Adaptation du dictionnaire de la NMD

L'algorithme NMD va chercher à décomposer chaque événement afin de minimiser la divergence d'IS entre le spectrogramme cible et le spectrogramme estimé. Pour cela, il va choisir les bases à activer pour composer la meilleure combinaison permettant d'estimer le spectrogramme cible. De par sa construction, le dictionnaire des bases permet de décrire une multitude d'événements d'un instrument de la batterie, soit par l'utilisation d'une seule base soit par la combinaison de plusieurs d'entre elles. Ces bases restant fixes durant la décomposition du spectrogramme cible, elles gardent leur interprétabilité physique permettant un post-traitement pour en déduire la transcription de la partie correspondante à la batterie.

Cependant, le dictionnaire comporte un nombre fini de bases par instrument et bien que les combinaisons entre elles soient possibles, le caractère additif de la NMD ainsi que l'inadaptabilité des bases pendant la décomposition diminuent le nombre d'événements qu'il est possible de recréer. Ainsi, outre le problème d'interférences entre les instruments jouant simultanément, la taille réduite du dictionnaire peut ne pas permettre la description et donc la détection d'un événement de batterie ni par l'activation d'une base ni par la combinaison de certaines d'entre elles.

Prenons un cas simple où un événement d'un des instruments de la batterie a lieu seul. Idéalement, l'algorithme devrait activer une ou plusieurs bases du dictionnaire correspondantes à l'instrument responsable. Comme l'instrument du morceau étudié peut différer des exemples disponibles dans le dictionnaire, l'algorithme va chercher à compenser ce qui ne peut pas être décrit par les bases de l'instrument avec les bases de *background*. Si l'événement est très éloigné des bases disponibles, l'algorithme de décomposition va utiliser exclusivement les bases de *background*. L'événement de batterie ne sera alors pas détecté.

L'absence de flexibilité du dictionnaire réduit le nombre de possibilités pour décrire les événements qui apparaissent dans la musique mais autoriser la mise à jour des bases pendant la décomposition accroît le risque que les bases perdent leur interprétation physique. Comment adapter les bases au signal étudié sans qu'elles perdent leur signification ? Est-il possible de modéliser les différences de modèle, de conditions d'enregistrement, de façons de jouer sous forme d'un filtre appliqué aux bases d'un instrument ? La mise à

jour des bases peut-elle être suffisamment contrainte pour que ces dernières correspondent toujours aux instruments qu'elles décrivaient ?

Dans ce chapitre, nous construirons un modèle source/filtre pour atténuer les différences entre le signal étudié et le dictionnaire. Puis, nous étudierons plusieurs manières de contraindre la mise à jour du dictionnaire.

6.1 État de l'art : l'adaptation des bases

Le problème de l'adaptation des bases a déjà été abordé pour différentes tâches. K. Kashino a utilisé l'adaptation de motifs pour l'identification de sources sonores dans un morceau de musique dans (Kashino and Murase, 1997).

Pour la transcription de la musique, K. Yoshii introduit dans (Yoshii et al., 2004) une adaptation des motifs en sélectionnant les segments du morceau où il a détecté la présence du motif et met à jour le motif avec la médiane de tous ces segments après traitements. Il affine cette méthode dans (Yoshii et al., 2007).

On retrouve la notion d'adaptation appliqué à la NMF dans (Vincent et al., 2008). Il décrit le spectre d'amplitude à court terme du signal à décomposer en une somme de spectres de base représentant chacun une hauteur multipliés par une amplitude variant au cours du temps. Chaque spectre de base est décomposé en une somme pondérée de spectres à bande étroite. L'idée est de contraindre fortement la structure des spectres de base mais de laisser des degrés de liberté à l'enveloppe spectrale.

X. Jaureguiberry propose un modèle source/filtre dans (Jaureguiberry et al., 2011) pour la NMF. Il applique un filtre aux bases du dictionnaire qui restent fixes. Le filtre permet d'adapter les bases par rapport au signal étudié.

Enfin, E. Benetos aborde l'adaptation des motifs pour la transcription automatique de la musique en utilisant la PLCA dans (Benetos et al., 2014). L'idée est de procéder à une première décomposition par NMF dite conservative. On cherche à avoir un *recall* très bas mais une précision très haute. Pour cela, on ne retient que les *pitchs* les plus fiables. On laisse donc de côté un certain nombre de *pitchs*. Avec ces événements détectés, une collection de spectres correspondant au *pitch* détecté dans le signal à trancrire est construite. Grâce à cette collection, de nouveaux motifs sont créés pour chaque *pitch* détecté dans le prétraitement. Enfin, comme un certain nombre de *pitchs* a été laissé de côté, ils adaptent les motifs restant par translation des nouveaux motifs.

Dans l'algorithme de transcription automatique de la musique que nous développons, le principal problème est d'adapter les bases sans que ces dernières ne perdent les caractéristiques liées à l'instrument qu'elles décrivent puisque nous voulons pouvoir reconnaître les instruments a posteriori. Les méthodes utilisées dans la littérature ne prennent pas en compte la contrainte de garder l'interprétation des bases après adaptation de celles-ci. Les adaptations ne sont pas contraintes. Notre principal objectif est donc d'adapter les bases au morceau tout en les gardant interprétables pour permettre le post-traitement.

Dans ce chapitre, nous développons pour répondre à notre objectif différentes approches : une approche par filtrage, une approche par mise à jour des bases du diction-

naire avec un dictionnaire contenant quinze bases par instrument puis un dictionnaire ne contenant qu'une base par instrument.

L'approche par filtrage se base sur un modèle source/filtre de (Hahn, 2015). Les bases du dictionnaire sont modulées par un filtre. Ce dernier pourra donner la signature de la salle ou du microphone et permettre une adaptation lissée qui empêchera les bases adaptées de trop s'éloigner des bases du dictionnaire.

Dans (Yoshii et al., 2004), les auteurs travaillent directement sur les motifs des instruments en modifiant le motif suivant le morceau. La deuxième approche suit cette idée en adaptant les bases par mise à jour grâce aux informations contenues dans les segments étudiés. Ici encore, la base doit correspondre à l'instrument cible tout au long du morceau. L'analyse des segments indépendamment les uns des autres n'est plus possible. De plus, le problème de signification physique des bases est ici particulièrement pointu et nous expliciterons différentes contraintes des mises à jour pour que les bases continuent de décrire le même instrument. Nous établissons dans ce chapitre les équations de mises à jour. Nous contraindrons fortement les mises à jour pour conserver les significations physiques des bases.

6.2 Adaptation du dictionnaire par filtrage

Les conditions d'enregistrement d'un morceau peuvent être grossièrement représentées par la réponse de la salle ainsi que la réponse du microphone. Par une approximation, chaque événement d'un instrument obtiendra la même réponse de la salle et du microphone utilisé. L'idée serait d'utiliser un filtre pour représenter ces réponses et modéliser les différences entre le dictionnaire et le morceau étudié. De plus, en évitant de prendre un filtre à résolution fine, les changements sur les bases seront lissés. En effet, le filtre ne pourra pas modifier une bande de fréquences en particulier mais devra adapter plusieurs bandes fréquentielles. Comme dans le modèle source/filtre de (Hahn, 2015), les filtres seront basés sur des courbes *B-splines*.

6.2.1 Construction des *B-splines*

Les *B-splines* sont des fonctions polynômiales par morceaux utilisées pour les problèmes d'interpolation ou d'approximation. Elles sont en générale contrôlées par deux ensembles de points : un vecteur de points t_i appelés nœuds (*knots*) et d'un vecteur de points de contrôle P_i, de même taille que le vecteur des nœuds, appelés points de contrôle. Deux nœuds, t_i et t_{i+1}, peuvent être confondus. Alors le sommet P_i est ignoré.

On définit les k fonctions *B-splines*, $B_{i,k}$, par récurrence, avec $i = 0, \ldots, m-k-1$,

$$B_{i,0}(t) = \begin{cases} 1 & \text{pour } t \in [t_i, t_{i+1}[\\ 0 & \text{sinon} \end{cases}$$

et pour $k \geq 1$,

$$B_{i,k}(t) = \omega_{i,k}(t) B_{i,k-1}(t) + (1 - \omega_{i+1,k}(t)) B_{i+1,k-1}(t) \tag{6.1}$$

avec, pour $j = 1, \ldots, m+1-i$, $\omega_{i,k} = \dfrac{t - t_i}{t_{i+k} - t_i}$ si $t_i < t_{i+1}$ et $\omega_{i,k} = 0$ si $t_i - t_{i+1}$.

Dans cette section nous utiliserons des *B-splines* d'ordre 2 comme dans la figure 6.2.1.

.

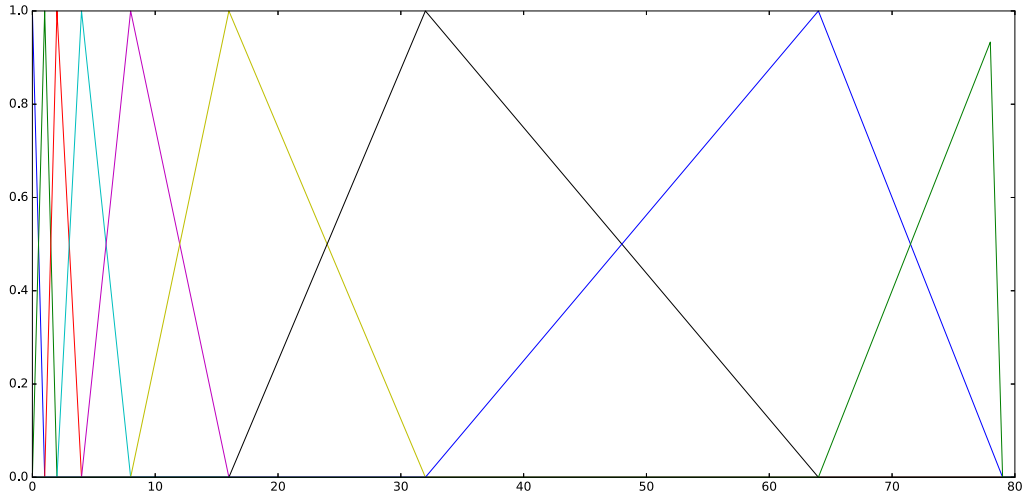

FIGURE 6.1 – *B-splines d'ordre 2.*

6.2.2 Modèle exponentiel et gaussien

L'utilisation d'un modèle source/filtre a été adaptée à la NMF dans (Durrieu et al., 2009) pour la séparation d'instruments dans un morceau polyphonique et dans (Bouvier et al., 2016) pour la séparation de la parole. Le nouveau modèle pour la NMD s'écrit comme suit :

$$\begin{aligned} \boldsymbol{V}_{fn} \approx \tilde{\boldsymbol{V}}_{fn} &= \sum_{t=1}^{T} [\boldsymbol{W}^{t\phi} \boldsymbol{H}^{t\rightarrow}]_{fn} = \sum_{t=1}^{T} \sum_{k=1}^{K} \boldsymbol{W}^{t\phi}_{fk} \boldsymbol{H}_{k(n-t)} \\ &= \sum_{t=1}^{T} \sum_{k=1}^{K} \phi_{fk} \boldsymbol{W}^{t}_{fk} \boldsymbol{H}_{k(n-t)} \end{aligned} \tag{6.2}$$

où les indices f, k et n représentent respectivement le *bin* fréquentiel, le numéro de la base et enfin la trame temporelle. $\boldsymbol{W}^{t\phi}$ représente la matrice des motifs filtrés à l'instant t alors que $\boldsymbol{W}^{t}$ représente la matrice des motifs non filtrés à l'instant t, ϕ est le filtre appliqué aux bases et enfin $\boldsymbol{H}$ est la matrice d'activation.

Les filtres ϕ doivent présenter certaines propriétés. Tout d'abord, bien qu'il y ait plusieurs bases correspondantes à un instrument, elles décrivent le même instrument. Il n'y a pas plusieurs instruments de type grosse-caisse, caisse-claire ou Charleston dans un

morceau. Ainsi, il y a un filtre par instrument, c'est-à-dire qu'il sera appliqué à chaque base de l'instrument correspondant. De plus, l'instrument utilisé ne change pas durant le morceau ni les conditions d'enregistrement. Le filtre doit donc être valable tout au long du morceau. Les modifications que la salle d'enregistrement ou les microphones seront les mêmes au début et à la fin du morceau. Enfin, comme les filtres doivent permettre d'adapter les bases au morceau, ils doivent être mis à jour comme la matrice d'activation.

Les filtres s'écrivent, avec B^{sp} les filtres *B-splines* et P les poids alloués à chaque *B-splines*, comme :

$$\phi_{l_{f,I}} = \sum_{knots} B^{sp}_{f,knots} P_{knots,I} \tag{6.3}$$

Le vecteur de poids P permet de moduler les valeurs des maxima que les *B-splines* peuvent atteindre. On peut ainsi modifier le filtre pour obtenir la forme voulue. Dans le domaine linéaire, deux représentations ont été considérées : une expression exponentielle et une expression gaussienne. Dans le premier cas, le filtre devient donc, avec I désignant l'instrument à adapter :

$$\phi_{f,I} = 10^{-\frac{1}{20}\sum_{knots} B^{sp}_{f,knots} P_{knots,I}} \tag{6.4}$$

Dans le second cas,

$$\phi_{f,I} = e^{-\frac{1}{2\sigma}\left(\sum_{knots} B^{sp}_{f,knots} P_{knots,I}\right)^2} \tag{6.5}$$

Pour trouver le filtre qui permet de mieux adapter les bases au morceau étudié, il faudra donc trouver les poids convenant le mieux.

Pour rappel, dans l'algorithme de détection de la batterie de (Röbel et al., 2015), un algorithme extérieur donnait les positions possibles des *onsets*. Ensuite, les décompositions en matrice non-négatives étaient réalisées sur les spectrogrammes autour de ces *onsets* comme expliqué à la section 5.2.2. Enfin, une dernière étape permettait de statuer sur la présence d'un *onset* de l'instrument étudié et dans le cas positif, de donner sa position.

Puisque les filtres doivent être valables pour tout le morceau, cela ne nous permet plus de considérer les segments étudiés indépendamment les uns des autres . En effet, les filtres sont utilisés pour atténuer les différences entre les bases de l'instrument cible et l'instrument réellement présent dans le morceau. Le filtre a donc besoin des informations à tous les endroits où l'instrument est présent. S'il n'était valable que sur un segment, le filtre pourrait prendre en compte des informations du *background* et donc d'autres instruments. Pour permettre de prendre en compte tous les instants où les filtres sont présents, il faudra étudier tous les segments ensembles. Pour cela, à chaque itération de l'algorithme de déconvolution en matrices non-négatives (NMD) les bases de *background* de la matrice $\boldsymbol{W}$ et les activations $\boldsymbol{H}$ sont mises à jours sur chaque segment puis les poids des filtres sont mis à jour, par mise à jour multiplicative, en prenant en compte

toutes les informations des segments en suivant la règle de mise à jour suivante :

$$P_{knots,ins} \leftarrow P_{knots,ins} \frac{\frac{\ln 10}{20} \sum_{fn} \frac{1}{\tilde{\boldsymbol{V}}_{fn}} B_{f,knots} \tilde{\boldsymbol{V}}_{fn}^{ins} + \frac{\partial^- P(\phi|\phi^{B-spline})}{\partial P_{knots,ins}}}{\frac{\ln 10}{20} \sum_{fn} \frac{\boldsymbol{V}_{fn}}{[\tilde{\boldsymbol{V}}_{fn}]^2} B_{f,knots} \tilde{\boldsymbol{V}}_{fn}^{ins} + \frac{\partial^+ P(\phi|\phi^{B-spline})}{\partial P_{knots,ins}}} \tag{6.6}$$

pour la configuration exponentielle et pour la configuration gaussienne

$$P_{knots,ins} \leftarrow P_{knots,ins} \frac{\frac{2}{2\sigma} \sum_{fn} B_{f,kn} \phi_{f,ins} \frac{\tilde{\boldsymbol{V}}_{fn}^{ins}}{\tilde{\boldsymbol{V}}_{fn}} + \frac{\partial^- P(\phi|\phi^{B-spline})}{\partial P_{knots,ins}}}{\frac{2}{2\sigma} \sum_{fn} B_{f,kn} \phi_{f,ins} \tilde{\boldsymbol{V}}_{fn}^{ins} \frac{V_{f,n}}{\tilde{\boldsymbol{V}}_{fn}^2} + \frac{\partial^+ P(\phi|\phi^{B-spline})}{\partial P_{knots,ins}}} \tag{6.7}$$

avec $P(\phi|\phi^{B-spline})$ une pénalisation possible de la mise à jour des poids. La règle de mise à jour de $\boldsymbol{H}$ est, elle aussi, modifiée :

$$\boldsymbol{H}_{pq} \leftarrow \boldsymbol{H}_{pq} \frac{\sum_f \sum_t \frac{\boldsymbol{V}_{f(q+t)} \phi_{fp} \boldsymbol{W}_{fp}^t}{[\tilde{\boldsymbol{V}}_{f(q+t)}]^2}}{\sum_f \sum_t \frac{\phi_{fp} \boldsymbol{W}_{fp}^t}{\tilde{\boldsymbol{V}}_{f(q+t)}}} \tag{6.8}$$

Le détail des calculs des équations de mise à jour est donné en Annexe C.

Lors des expériences, des oscillations de la fonction de coût ont été observées. Le gradient étant vérifié, le problème est simplifié pour comprendre d'où elles peuvent provenir.

6.2.3 Provenance de la non-convergence

Réalisons une expérience simple. Le spectrogramme cible est constant. On initialise $\boldsymbol{W}$ et $\boldsymbol{H}$ de façon à pouvoir reconstruire parfaitement le spectrogramme. Enfin, on initialise le filtre telle que la reconstruction avant les mises à jour diffère du spectrogramme cible. Seuls les poids du filtre seront mis à jour, $\boldsymbol{W}$ et $\boldsymbol{H}$ restent fixes. Ici on présente les résultats obtenus pour le modèle gaussien. Les résultats sont similaires dans le cas du modèle exponentiel.

Au bout de quelques itérations, le spectrogramme estimé est tel que toutes les composantes sont égales entre elles. Lorsque l'on parlera de relation d'ordre entre les spectrogrammes, on fera donc ici référence à la valeur prise par toutes leurs composantes.

Dans cette configuration, on attend de la décomposition qu'elle agisse sur les poids du filtres pour que le spectrogramme reconstruit soit égal ou du moins qu'il soit de plus en plus proche du spectrogramme cible. La fonction de coût devrait donc diminuer de plus en plus. Cependant, ce n'est pas ce qu'il se passe comme le montre la représentation de la fonction de coût, figure 6.2, au fur et à mesure des itérations.

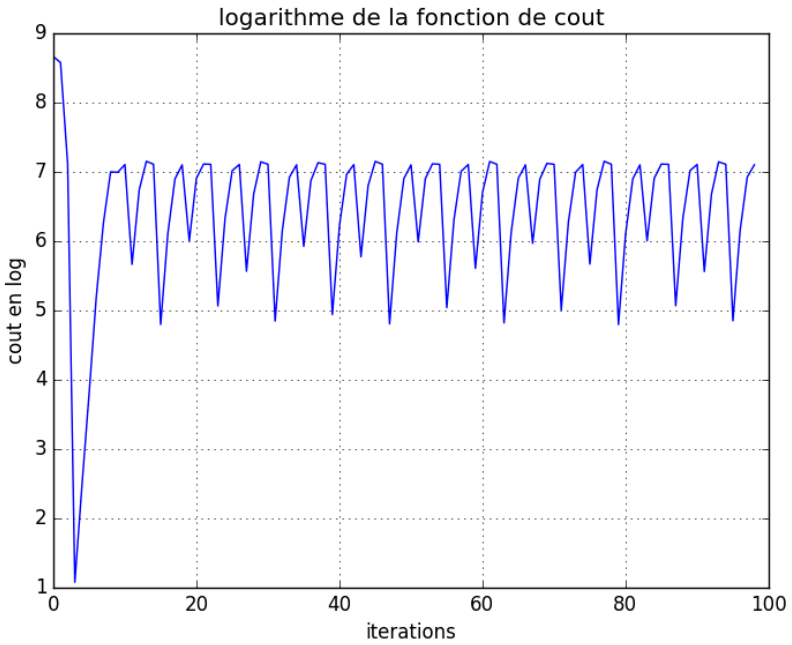

Figure 6.2 – *Fonction de coût (en log) pour l'adaptation par filtrage*

On remarque que la fonction de coût oscille sans jamais approcher d'une valeur limite. Dans le même temps, l'expérience montre que tant que le spectrogramme estimé est au-dessus du spectrogramme cible alors la fonction de coût diminue et dès que le spectrogramme estimé passe en dessous, l'algorithme cherche à remonter la valeur du spectrogramme estimé mais s'éloigne du spectrogramme cible. La fonction de coût augmente alors.

On a pu vérifier pendant cette expérience que le gradient de la fonction de coût donne bien l'erreur attendue de la fonction de coût. Cependant, contrairement à (Jaureguiberry et al., 2011) dans le cadre de la NMF, l'expérience a montré que les mises à jour multiplicatives pour l'adaptation des filtres dans le cas de la NMD ne permettaient pas la convergence de la fonction de coût.

Comme le fait remarquer N. Bertin (Bertin, 2009), il y a peu de preuves de la convergence avec l'utilisation des mises à jour multiplicatives.

Nous avons donc décidé de ne pas retenir cette approche.

6.3 Mise à jour contrainte du dictionnaire à 15 bases par instrument

Une autre manière d'adapter le dictionnaire au morceau est d'intervenir directement sur les bases en les mettant à jour avec les informations tirées du morceau étudié. Com-

ment assurer qu'après adaptation par mise à jour des bases, ces dernières conservent leur interprétation physique ? La mise à jour devra être fortement contrainte pour que les bases adaptées décrivent toujours l'instrument de départ. De plus, comme pour l'adaptation par filtre, il est important de prendre en considération toutes les informations disponibles dans le morceau et pas seulement les informations des segments étudiés les uns indépendamment des autres. Le schéma de la nouvelle approche est donné dans la figure 6.3.

6.3.1 Contrainte basée sur la divergence d'IS

Le modèle reste le même que pour la NMD présentée dans (Röbel et al., 2015) mais on autorise ici les mises à jour du dictionnaire W. Nous voulons pénaliser la mise à jour des bases pour que les bases adaptées s'éloignent peu des bases du dictionnaire et ainsi conservent leur signification physique, c'est-à-dire qu'une base de grosse-caisse doit rester une base de grosse-caisse. Pour cela, il peut être intéressant de minimiser une distance ou une divergence entre les bases adaptées et le dictionnaire. On choisit la divergence d'IS. Cela permet de limiter la modification des bases. Si la base adaptée s'éloigne trop de la base initiale, la distance entre les deux bases augmentera et donc la fonction de coût. La descente de gradient permettra de modifier les coefficients de la base pour qu'elle soit plus proche de la base existante. La partie de la fonction de coût dépendant de $\boldsymbol{W}$ devient alors :

$$\begin{aligned} C(\boldsymbol{W}) &= \sum_{f=1}^{F}\sum_{n=1}^{N} d_{IS}(\boldsymbol{V}_{fn}, \tilde{\boldsymbol{V}}_{fn}) + \lambda_W \sum_{k=1}^{K}\sum_{f=1}^{F}\sum_{t=1}^{T} d_{IS}(\overline{\boldsymbol{W}_{ft}^k}, \boldsymbol{W}_{ft}^k) \\ &= \sum_{f=1}^{F}\sum_{n=1}^{N}\left(\frac{\boldsymbol{V}_{fn}}{\tilde{\boldsymbol{V}}_{fn}} - \log\frac{\boldsymbol{V}_{fn}}{\tilde{\boldsymbol{V}}_{fn}} - 1\right) + \lambda_W \sum_{k=1}^{K}\sum_{f=1}^{F}\sum_{t=1}^{T}\left(\frac{\overline{\boldsymbol{W}_{ft}^k}}{\boldsymbol{W}_{ft}^k} - \log\frac{\overline{\boldsymbol{W}_{ft}^k}}{\boldsymbol{W}_{ft}^k} - 1\right) \end{aligned} \quad (6.9)$$

où $\overline{\boldsymbol{W}}$ désigne les bases du dictionnaire non modifiées et λ_W est une pondération permettant de modifier l'influence de la pénalisation. Les autres notations sont restées inchangées par rapport à la partie précédente. On en déduit l'équation de mise à jour de $\boldsymbol{W}$:

$$\boldsymbol{W}_{lb}^p \leftarrow \boldsymbol{W}_{lb}^p \frac{\sum_n \frac{\boldsymbol{V}_{ln}\boldsymbol{H}_{p,n-b}}{\tilde{\boldsymbol{V}}_{ln}^2} + \lambda_W \frac{\overline{\boldsymbol{W}_{lb}^p}}{(\boldsymbol{W}_{lb}^p)^2}}{\sum_n \frac{\boldsymbol{H}_{p,n-b}}{\tilde{\boldsymbol{V}}_{ln}} + \lambda_W \frac{1}{\overline{\boldsymbol{W}_{lb}^p}}} \quad (6.10)$$

Les détails de calculs des équations de mise à jour sont donnés en Annexe C.

Lors d'une première expérience, on a pu remarquer que les bases présentaient des éléments de forte amplitude à des endroits où elles n'auraient pas dû. Par exemple pour la grosse-caisse, l'information se trouve dans les basses fréquences mais après adaptation, on pouvait remarquer de fortes amplitudes dans des bandes de fréquences plus élevées. Nous avons pu observer aussi des changements abruptes d'énergie au cours du temps pour certains motifs. En autorisant les bases de batterie à être modifiées, le système peut

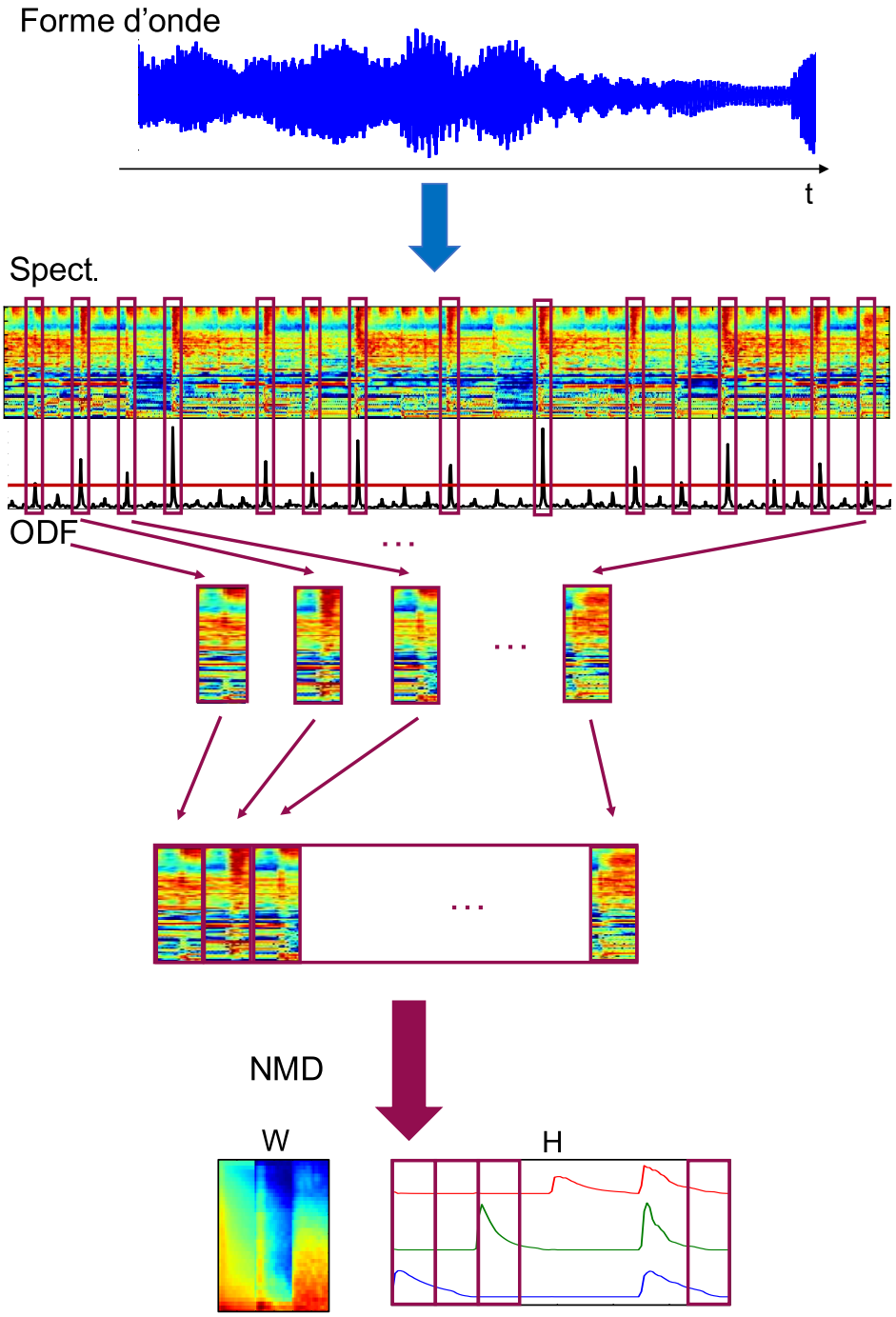

Figure 6.3 – *Algorithme de décomposition NMD appliqué autour de tous les onsets en même temps.*

potentiellement mélanger les instruments. Imaginons qu'un instrument (mélodique) joue souvent voir tout le temps au même moment qu'un instrument de batterie. L'algorithme risque alors d'introduire de l'information issue de cet instrument dans les motifs de batterie, pensant que les instruments ne sont en fait qu'un seul instrument. En étudiant les matrices de confusion des instruments de batterie étudiés et les instruments pouvant être présents dans le morceau suivant la valeur de λ_W, on observe que la confusion entre les instruments est influencée par la liberté laissée aux bases pour la mise à jour. La confusion augmente entre les instruments si λ_W a une petite valeur. La décorrélation entre les bases pourrait déjà atténuer ce problème.

6.3.2 Ajout de contraintes de décorrélation entre les instruments

L'usage de la décorrélation permet aux bases modifiées de ne pas inclure de l'information issue d'autres instruments. L'algorithme mélangerait moins les informations issues de différents instruments.

L'idée est donc de minimiser la corrélation croisée entre les instruments, c'est-à-dire entre les bases d'instruments différents. En revanche on a aucun prérequis sur la corrélation entre les bases d'un même instrument.

Dans (Li et al., 2001), les auteurs ont proposé de limiter la séparation en plusieurs composantes d'une unité en essayant d'obtenir les lignes de $\boldsymbol{H}$ les plus décorrélées possible. Ils ont exprimé cette contrainte par une pénalisation de la fonction de coût dépendant du produit $\boldsymbol{H}\boldsymbol{H}^T$. Renforcer la diagonalité de cette matrice revient à minimiser la corrélation entre les lignes. La pénalisation est alors exprimée comme :

$$P_{corr}(H) = \sum_{j \neq k} [\boldsymbol{H}\boldsymbol{H}^T]_{kj} \tag{6.11}$$

Une version normalisée est proposée dans (Bouvier et al., 2016). Elle est adaptée ici à notre problème en l'appliquant aux bases et non aux activations :

$$\mathcal{P}_d(W) = \sum_{N_b=0}^{N_{ins}} \sum_{l=B[N_b]}^{B[N_b+1]} \sum_{c=0}^{B[N_b]} \frac{(\boldsymbol{W}\boldsymbol{W}^T)_{lc}}{\sigma_c \sigma_l} \tag{6.12}$$

avec $\sigma_k = \sqrt{\sum_{p=1}^{P} \boldsymbol{W}_{kp}^2}$, $k \in \{l, c\}$ et P le nombre de coefficients de la base k. N_{ins} est le nombre d'instrument, B est le nombre de bases cumulées pour chaque instrument. L'obtention de la nouvelle règle de mise à jour est détaillée en Annexe C.

$$\boldsymbol{W}_{lb}^p \leftarrow \boldsymbol{W}_{lb}^p \frac{\sum_n \frac{V_{ln} \boldsymbol{H}_{p,n-b}}{V_{est_{ln}}^2} + \lambda_W \frac{\overline{\boldsymbol{W}_{lb}^p}}{(\boldsymbol{W}_{lb}^p)^2} + \sum_{N_b=0}^{N_{ins}} \left(\sum_{l'=0}^{B[N_b]+B[N_b+1]} \frac{(\boldsymbol{W}\boldsymbol{W}^T)_{l'l} \sigma_l^{-1}}{\sigma_{l'} \sigma_l^2} \right)}{\sum_n \frac{\boldsymbol{H}_{p,n-b}}{V_{est_{ln}}} + \lambda_W \frac{1}{\boldsymbol{W}_{lb}^p} + \sum_{N_b=0}^{N_{ins}} \left(\sum_{l'=0}^{B[N_b]+B[N_b+1]} \frac{\boldsymbol{W}_{l'b}}{\sigma_{l'} \sigma_l} \right)} \tag{6.13}$$

L'ajout de la contrainte de décorrélation introduit un nouveau paramètre de pondération dans la fonction de coût. Ce paramètre de pondération doit être optimisé. Or, le paramètre de pondération de la pénalisation de l'utilisation des bases de *background* et celui de la mise à jour des bases $\boldsymbol{W}$ vont avoir un impact sur celui de la décorrélation. Il devient coûteux d'optimiser ces trois paramètres à la fois.

6.3.3 Contraintes de parcimonie des activations

On a donc vu que le choix de l'information utilisée pour l'adaptation des bases est extrêmement important. L'utilisation d'informations qui ne sont pas relatives à l'instrument peut dévier la base de sa signification physique. Aider l'algorithme à trouver dans le segment autour de l'*onset* où se trouve l'information la plus pertinente pour la base considérée peut aider l'algorithme à mieux adapter la base. Nous avons voulu pour cela renforcer la parcimonie des activations. Nous cherchons à avoir des activations étroites, l'évolution temporelle des événements percussifs étant représentée dans les motifs.

Pour réaliser cela, les mises à jour des différents paramètres sont réalisés en alternance. Les bases sont d'abord adaptées. Pour pointer plus précisément l'information, la matrice $\boldsymbol{H}$ est modifiée. Pour chaque base, seul le maximum de l'activation correspondante est conservé et les autres coefficients sont égaux à 0. Les bases $\boldsymbol{W}$ sont mises à jour puis $\boldsymbol{H}$ retrouve ses valeurs d'origine et est mis à jour juste après les bases de *background*. Ces itérations sont répétées jusqu'à la convergence de la fonction de coût.

6.3.4 Contrainte d'activation d'une seule base par instrument

Un problème qui peut se présenter avec l'utilisation de plusieurs bases pour un même instrument est que la combinaison de ces bases pourrait ne plus représenter l'instrument. En reprenant la stratégie de la section précédente 6.3.3, on peut forcer qu'une seule base soit privilégiée par segment. Cette base n'est donc pas nécessairement la même d'un segment à l'autre. Au lieu de considérer le maximum de l'activation par base, tous les coefficients des activations concernant un instrument sont mis à zéro excepté le maximum de toutes ces activations.

6.3.5 Résultats

Les résultats de l'adaptation des bases d'un dictionnaire comportant 15 bases par instrument sont donnés dans le tableau 6.1 et ont été obtenus sur la base de données utilisées à la section 5.3.1. Ils sont comparés aux résultats obtenus sans adaptation à la section 5.3.2. Les adaptations sont bénéfiques en terme de F-mesure pour la grosse-caisse avec toutes les contraintes appliquées, sauf avec la décorrélation, ainsi que pour la Charleston sauf avec l'alternance de $\boldsymbol{W}$ et $\boldsymbol{H}$. Cependant, aucune contrainte ne permet d'améliorer la détection de la caisse-claire. Ainsi aucune adaptation contrainte ne permet d'améliorer la transcription des trois principaux instruments de batterie.

La contrainte de décorrélation ne peut être imposée qu'entre les différents instruments de batterie. N'ayant aucune connaissance a priori des instruments qui seront présents

		R	P	F
Sans adaptation	BD	76.7	84.7	80.5
	SD	88.4	89.1	88.7
	HH	85.1	70.5	77.1
Avec divergence IS seule	BD	86.2	75.7	**80.6**
	SD	82.5	90.4	86.3
	HH	79.5	79.8	**79.7**
Avec décorrelation	BD	85.6	74.9	79.9
	SD	80.3	94.0	86.6
	HH	82.5	77.7	**80.0**
Avec parcimonie par base	BD	80.9	83.5	**82.2**
	SD	83.9	77.2	80.4
	HH	94.2	64.9	76.8
Avec parcimonie par instrument	BD	77.7	88.5	**82.7**
	SD	86.9	85.6	86.2
	HH	79.9	74.8	**77.2**

TABLEAU 6.1 – *Résultats de la transcription de la batterie avec la NMD avec adaptation des bases différemment contraintes.*

dans le morceau, on ne peut pas imposer la décorrélation avec ces derniers. Pour pouvoir pleinement tirer profit de cette contrainte, il serait intéressant d'avoir un modèle complet avec les instruments percussifs ainsi que les instruments harmoniques.

Imposer la parcimonie à $\boldsymbol{H}$ lors de l'adaptation des bases permet d'obtenir de bons résultats pour la grosse-caisse ainsi que la Charleston et c'est la contrainte qui permet de garder des résultats pour la caisse-claire les plus proches des résultats sans adaptation.

6.4 Mise à jour contrainte du dictionnaire à 1 base par instrument

Les résultats précédents ont montré que contraindre l'algorithme à ne pas faire de combinaison de bases d'un même instrument et donc renforcer l'utilisation d'une seule base par instrument permet d'améliorer la détection de la grosse-caisse et de la Charleston et de limiter la perte de performance pour la caisse-claire par rapport à d'autres méthodes. Même si les bases peuvent varier dans un certain domaine pour garder leur signification, il n'y a aucune certitude que la combinaison de ces bases reste dans ce domaine.

Dans cette section, nous réduisons le nombre de bases du dictionnaire à une base par instrument comme dans (Yoshii et al., 2007). L'utilisation d'une seule base par instrument ne permet pas la combinaison entre elles et diminue donc le risque de sortir du domaine. La sélection des bases, cf. section 5.2.1, était réalisée pour que les bases choisies présentent une grande variabilité et décrivent au plus proche les exemples disponibles.

La base unique doit décrire le maximum d'événements dans le morceau analysé. Si la base est apprise au préalable sur des exemples autres que le morceau étudié, elle peut

ne pas correspondre du tout à l'instrument joué et beaucoup d'événements peuvent ne pas être détectés. Nous changeons la façon de choisir la base pour qu'elle corresponde au morceau étudié.

De plus, lors de l'expérience précédente en contraignant la parcimonie par instrument, nous avons pu remarquer que les bases sélectionnées par l'algorithme n'était pas les mêmes d'un segment à l'autre. En réduisant le nombre de bases allouées à chaque instrument, la variabilité du dictionnaire est supprimée malgré la possibilité d'adaptation des bases. Si la base unique est adaptée avec les informations de tous les segments, elle pourrait ne pas correspondre à certains événements de l'instrument cible qui ne serait alors pas détecté. Nous réaliserons l'adaptation de la base pour chaque segment. Les segments sont donc de nouveau considérés indépendamment les uns des autres comme dans la figure 5.2.

6.4.1 Choix de la base

Le choix des quatre bases (une par instrument) qui constituera le dictionnaire est une étape importante. La construction du dictionnaire comme décrit à la section 5.2.1 était réalisée préalablement puis le dictionnaire était utilisé pour n'importe quel morceau. Les bases étaient choisies pour décrire le maximum d'événements différents. Le dictionnaire présentait donc une certaine variabilité en contenant 15 bases par instrument. En ne sélectionnant qu'une seule base par instrument, le dictionnaire perd de la flexibilité. C'est pourquoi, dans le cas du dictionnaire à une base par instrument, nous choisissons les bases en fonction du morceau et non par rapport aux différents enregistrements d'exemples disponibles. La construction du dictionnaire doit ainsi permettre de se rapprocher le plus possible des instruments de batterie présents dans le morceau.

On considère maintenant les différents enregistrements d'exemples de l'instrument utilisé pour construire le dictionnaire comme l'ensemble des bases candidates pour le morceau étudié. Nous cherchons l'exemple qui coïncide le mieux avec les événements de l'instrument présents dans le morceau. Les corrélations entre chaque exemple et tous les segments sont calculées. Les exemples sont classés en fonction de la moyenne des corrélations avec les segments. L'exemple qui obtient la corrélation moyenne maximale est sélectionnée pour le dictionnaire. Les 60 exemples suivant sont sélectionnés pour permettre de contraindre l'adaptation de cette base.

6.4.2 Les différents modèles étudiés

Les modèles de la section 6.3.1 d'adaptation contrainte par la divergence d'IS et celle avec l'alternance entre la mise à jour de $\boldsymbol{H}$ et celle de $\boldsymbol{W}$ avec le maximum des activations de la section 6.3.3 sont adaptés au dictionnaire avec 1 base par instrument. Dans les équations de mise à jour de $\boldsymbol{W}$, $\overline{\boldsymbol{W}}$ désigne les exemples sélectionnés par rapport à leur score de corrélation.

6.4.3 Résultats

Les résultats sont consignés dans le tableau 6.2 et comparés aux résultats obtenus sans adaptation des bases avec 15 bases par instrument ainsi qu'avec les résultats obtenus sans adaptation avec 1 seule base par instrument. Les résultats sans adaptation avec 15 bases par instruments ont été obtenus à la section 5.3.2.

		R	P	F
Sans adaptation (15 bases)	BD	76.7	84.7	80.5
	SD	88.4	89.1	88.7
	HH	85.1	70.5	77.1
Sans adaptation (1 base)	BD	87.8	81.7	**84.6**
	SD	82.5	89.0	85.6
	HH	82.8	85.3	**84.0**
Avec divergence IS seule	BD	80.3	83.9	**82.1**
	SD	79.6	87.2	83.2
	HH	81.2	78.9	**80.0**
Avec parcimonie par instrument	BD	89.9	84.1	**86.9**
	SD	79.6	84.5	82.0
	HH	82.5	82.5	**82.5**

TABLEAU 6.2 – *Résultats de la transcription de la batterie avec la NMD avec adaptation des bases différemment contrainte.*

Les résultats sans adaptation avec un dictionnaire contenant une base par instrument surpassent ceux pour le dictionnaire contenant 15 bases par instrument pour la grosse-caisse et la Charleston. Cependant, la F-mesure perd 3 points pour la caisse-claire. Il semble qu'aucune adaptation permet d'obtenir de meilleurs résultats pour la caisse-claire que pour le modèle de dictionnaire à 15 bases. Cependant, la détection de la grosse-caisse est nettement améliorée par adaptation avec parcimonie par instrument avec un dictionnaire à une base par instrument avec plus de 6 points de plus que le modèle original. Cependant aucun modèle ne permet d'améliorer les résultats de tous les instruments de la batterie.

6.5 Mise à jour du dictionnaire avec définition du domaine de variation

Le défi le plus important pour l'adaptation des bases est que les bases doivent garder leur signification physique. En pénalisant les bases adaptées à rester proches des bases originales, il n'y a pas de certitude qu'elles restent dans le domaine qui leur est alloué.

Au cours des expérimentations précédentes, il semble que, malgré les différentes contraintes appliquées lors de la mise à jour des bases du dictionnaire, celles-ci s'éloignent assez de la base d'origine pour perdre leur signification physique. Elles sortent de leur domaine de définition, domaine dans lequel elles sont supposées varier.

L'idée dans ce paragraphe est de définir les frontières de ce domaine et d'imposer aux bases adaptées de garder tous leurs éléments dans ce domaine en appliquant différentes contraintes.

6.5.1 Définition du domaine de variation et contraintes appliquées

On définit le domaine au sein duquel les bases peuvent varier à partir des événements disponibles pour l'apprentissage du dictionnaire, voir section 5.2.1. Les bases sont sélectionnées comme expliqué dans cette section. Les événements écartés sont utilisés pour définir le contour du domaine. Pour cela on cherche pour chaque coefficient de la matrice représentant le motif le maximum de la distance entre le coefficient de la base et ceux des autres exemples $\overline{\boldsymbol{W}_{ft}^k}$.

$$S_{tf}^k = \max_{k'} |\overline{\boldsymbol{W}_{ft}^{k'}} - \boldsymbol{W}_{ft}^k| \tag{6.14}$$

À chaque itération pendant la décomposition, les bases sont adaptées. Les nouvelles bases sont comparées aux bases délimitant le contour du domaine. Si certains de leurs éléments (*bins*) sortent du domaine, les *bins* seront pénalisés pour permettre de les garder proches de la base de départ.

$$C(\boldsymbol{W}) \quad = \quad \sum_{fn} d_{IS}(\boldsymbol{V}_{fn}, \tilde{\boldsymbol{V}}_{fn}) + \lambda_W \sum_k \sum_{ft} P(\boldsymbol{W}_{ft}^k, \overline{\boldsymbol{W}_{ft}^k}) \mathbb{1}_{|\boldsymbol{W}_{ft}^k - \overline{\boldsymbol{W}_{ft}^k}| < S_{ftk}} \tag{6.15}$$

Comme dans la section 6.4, les bases qui n'ont pas été sélectionnées sont utilisées pour calculer le terme de pénalisation ajouté à la fonction de coût. Le terme de pénalisation est basé sur la divergence d'IS ainsi $\sum_k \sum_{ft} P(\boldsymbol{W}_{ft}^k, \overline{\boldsymbol{W}_{ft}^k}) = \sum_k \sum_{ft} d_{IS}(\boldsymbol{W}_{ft}^k, \overline{\boldsymbol{W}_{ft}^k})$. Les bases $\boldsymbol{W}$ sont mises à jour en suivant la règle :

$$\boldsymbol{W}_{lb}^p \leftarrow \boldsymbol{W}_{lb}^p \frac{\sum_n \frac{V_{ln} \boldsymbol{H}_{p,n-b}}{V_{est_{ln}}^2} + \lambda_W \frac{\overline{\boldsymbol{W}_{lb}^p}}{(\boldsymbol{W}_{lb}^p)^2} \mathbb{1}_{|\boldsymbol{W}_{ft}^k - \overline{\boldsymbol{W}_{ft}^k}| < S_{ftk}}}{\sum_n \frac{\boldsymbol{H}_{p,n-b}}{\tilde{\boldsymbol{V}}_{ln}} + \lambda_W \frac{1}{\boldsymbol{W}_{lb}^p} \mathbb{1}_{|\boldsymbol{W}_{ft}^k - \overline{\boldsymbol{W}_{ft}^k}| < S_{ftk}}} \tag{6.16}$$

6.5.2 Les différents modèles étudiés

Les différents modèles de contraintes présentés à la section 6.3 sont étudiés avec la contrainte supplémentaire de ne pas sortir du domaine de variation. Pour le dictionnaire à 15 bases par instrument, les résultats sont présentés avec la contrainte de divergence IS, celle de parcimonie par base et et celle de parcimonie par instrument. Pour le dictionnaire avec 1 base par instrument, la parcimonie par activation revient à la parcimonie par instrument. Les résultats sont donc présentés qu'avec la contrainte de divergence et celle de parcimonie par instrument.

6.5.3 Resultats

Les F-mesures des différents modèles sont données dans le tableau 6.3.

	15 bases			1 base		
Modèles	BD	SD	HH	BD	SD	HH
Sans adaptation	80.5	88.7	77.1	84.6	85.6	84.0
Divergence IS	**84.3**	82.5	74.8	**85.0**	**88.5**	79.0
Parcimonie par base	78.4	80.6	77.1	-	-	-
Parcimonie par instrument	**85.0**	88.2	**78.1**	**86.4**	80.6	80.6

TABLEAU 6.3 – *Résultats de la transcription de la batterie avec la NMD avec adaptation en imposant le domaine de variation.*

Seule la grosse-caisse profite des bénéfices des mises à jour en imposant le domaine de variation. En utilisant la parcimonie par instrument, les résultats de la grosse-caisse et la Charleston sont meilleurs que sans adaptation et comme pour la mise à jour contrainte (cf. section 6.3.1) permet d'obtenir les résultats les plus proches pour la caisse-claire. Cependant, la mise à jour contrainte seulement par la divergence d'IS avec un dictionnaire comportant qu'une seule base par instrument est la seule méthode qui permet d'obtenir de meilleurs résultats pour la caisse-claire par rapport aux résultats obtenus sans adaptation et un dictionnaire à une base. Cela n'améliore pas les résultats du modèle original avec 15 bases par instrument.

6.6 Conclusion

Plusieurs approches d'adaptation du dictionnaire au signal analysé ont été étudiées dans ce chapitre. Une première approche basé sur des modèles source/filtre ne permettait pas à la fonction de coût de converger même pour un exemple simple. Cette méthode aurait permis l'adaptation du dictionnaire aux conditions d'enregistrement notamment et aux différents modèles d'instrument. Les filtres, construits à partir de *B-splines*, modifient plusieurs bandes de fréquences. Cela assure à la base de garder leur interprétation physique car il n'est pas possible de modifier drastiquement une seule bande de fréquence. Cependant, cette méthode n'étant pas stable, elle ne peut pas être envisagée.

La deuxième approche consiste à modifier directement le dictionnaire constitué d'une ou de quinze bases par instrument. Le principal défi est toujours que les bases adaptées gardent leurs interprétations physiques. Si ce n'était pas le cas, la transcription ne serait plus possible. Plusieurs contraintes (décorrélation entre les différents instruments, divergence entre la base adaptée et la base originale ou parcimonie de activations) ont été envisagées. Suivant les modèles, certains instruments ont été mieux détectés que d'autres (la grosse-caisse et la Charleston) mais aucune approche n'a permis d'améliorer la détection de tous les instruments.

Il est possible que les instruments soient encore confondus avec d'autres malgré les contraintes. Ainsi si un instrument joue souvent voir toujours avec un autre, l'adaptation

des bases risquent d'incorporer des informations de l'autre instrument dans la base. Les décompositions sont réalisées autour des *onsets* qu'un algorithme extérieur fournit à l'algorithme de transcription de la batterie. Cet algorithme de détection d'*onsets* détectent tous les *onsets*, quelque soit l'instrument qui en est responsable. De plus, il est préférable de fournir plus d'*onsets* car un *onset* qui n'est pas détecté à ce moment là ne sera jamais détecté. Le problème est que l'algorithme de transcription de la batterie et notamment la décomposition en matrice non-négative va chercher à activer des bases relatives à la batterie alors qu'aucun instrument de batterie n'a joué à ce moment là. Cela crée des faux positifs et détériore les performances de la transcription.

Une solution envisageable est de fournir à l'algorithme seulement la position des *onsets* percussifs et donc de créer un détecteur d'*onsets* spécifiques qui détecte uniquement les *onsets* dont un des trois principaux instruments de la batterie soit responsable.

— Chapitre 7 —

Détection d'onsets pour la transcription de batterie

L'algorithme de détection d'*onsets* utilisé dans les chapitres précédents pour informer l'algorithme de transcription de la batterie (cf. section 5.2) est basé sur le flux spectral. Il met en valeur les changements d'énergie abruptes pour construire une *Onset Detection Function* (ODF). La recherche de maxima locaux de cette fonction permet ensuite de localiser les événements musicaux dans le morceau étudié. L'algorithme de transcription analyse ensuite un segment de signal autour des *onsets* détectés. Cette étape permet de réduire considérablement les temps de calcul. Cependant, si un *onset* n'est pas détecté à cette étape, le segment ne sera pas analysé. Les performances de la transcription dépendent donc aussi des performances de la détection d'*onsets*. Améliorer la détection d'*onsets* permet-il d'améliorer la transcription des instruments de batterie ?

Pour chaque segment, plusieurs cas de figure peuvent se présenter.
Un *onset* percussif est détecté et est joué seul. La NMD doit alors activer une des bases de l'instrument responsable.
L'instrument de batterie est accompagné par un instrument harmonique. L'algorithme doit activer au moins une des bases de l'instrument et compléter avec les bases de *background*.
Enfin, si seul un instrument harmonique est joué, l'algorithme doit uniquement décrire l'événement avec les bases de *background*.
Or comme l'utilisation des bases de *background* est pénalisée, il se peut que l'algorithme utilise tout de même les bases des instruments de batterie pour décrire l'événement. Et on obtient un faux positif. Informer l'algorithme avec les *onsets* de batterie pourrait-il permettre d'éviter ces faux positifs ? Si oui, cela pourrait aussi permettre de mieux localiser l'information pertinente pour adapter le dictionnaire au morceau.

Depuis quelques années, l'utilisation des réseaux de neurones pour la détection d'*onsets* a permis d'atteindre de très bons résultats. Dans ce chapitre, après avoir établi une revue des méthodes de l'état de l'art, nous comparons plusieurs configurations d'un *Convolutional Neural Network*, réseau de neurones convolutif utilisé pour la détection d'*onsets* généraux pour qu'il ne détecte que les *onsets* percussifs (n'importe quel *onset* joué par

une grosse-caisse, une caisse-claire ou une Charleston).

7.1 État de l'art : la détection d'onsets

L'apparition d'un événement musical est décrit par plusieurs paramètres comme l'*onset*, le début de l'événement ou l'*offset*, la fin de l'événement. Les différentes parties d'un d'*onset* sont présentées sur la figure 7.1 [1].

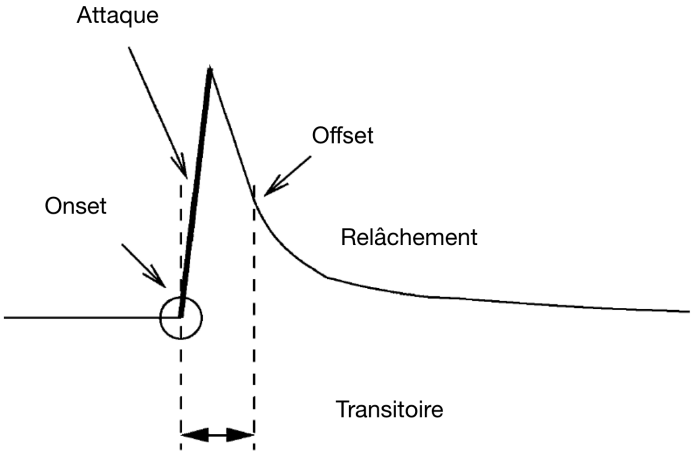

Figure 7.1 – *Onset, transitoire, attaque et relâchement d'un événement musical.*

La détection d'*onsets*, c'est-à-dire la localisation des événements dans une pièce de musique, est une importante étape pour la transcription de la musique. Cela constistue un excellent prétraitement pour la transcription comme pour le piano dans (Wang et al., 2017) et pour la transcription de la batterie dans (Röbel et al., 2015).

Il existe une multitude d'approches pour la détection d'*onsets*. Un panorama de ces méthodes est proposé dans (Bello et al., 2005). Ces méthodes sont souvent basées sur le même schéma. Une première étape de prétraitement permet de mettre en relief certaines propriétés du signal permettant ainsi de faciliter la détection. La fonction de détection, ODF est ensuite calculée. Les *onsets* sont ensuite localisés en repérant les maximaux locaux de la fonction de détection.

Récemment, des méthodes basées sur le *machine learning* ont obtenu de bons résultats. Alors que certains travaux essaient d'améliorer l'étape de recherche des maximaux locaux (Marolt et al., 2002), appelée *peak-picking*, d'autres utilisent les RNN pour créer la fonction de détection (Böck et al., 2012). Les méthodes utilisant les CNN obtiennent, d'après MIREX 2017 [2], les meilleurs résultats de l'état de l'art.

Dans (Schlüter and Böck, 2014), ils observent que les poids des filtres des couches de convolution suivent des distributions différentes suivant le type d'*onsets* qu'ils permettent

1. adapté de : (Bello et al., 2005)
2. `http://nema.lis.illinois.edu/nema_out/mirex2017/results/aod/summary.html`

de détecter, c'est-à-dire s'ils proviennent d'un instrument harmonique ou d'un instrument percussif.

7.2 Architecture pour la détection d'onsets percussifs

Les *onsets* percussifs sont les événements musicaux dont les instruments de la batterie sont responsables, ici la grosse-caisse, la caisse-claire et la Charleston. On trouve dans la littérature différents détecteurs d'*onsets* mais aucun n'est spécialisé dans la détection d'*onsets* spécifiques. Cependant, dans (Schlüter and Böck, 2014), ils développent un détecteur d'*onsets* basé sur un réseau de neurones convolutif et remarquent que la distribution des poids est différente pour un instrument percussif que pour un instrument harmonique. Il paraît donc possible de construire un détecteur d'*onsets* percussifs.

L'idée de ce chapitre est donc d'élaborer un détecteur d'*onsets* qui détecterait exclusivement les événements percussifs. Le détecteur d'*onsets* percussifs est basé sur le modèle de (Schlüter and Böck, 2014). Le réseau de neurones alterne des couches de convolution avec des couches d'activation (*Rectified Linear Units* (ReLU)) et des étapes de max-pooling. La dernière couche est une couche dense de ReLU et la couche de sortie contient une seule unité. La figure 7.2 représente un des modèles du réseau. Les données d'entrées sont des représentations temps-fréquence. La sortie du réseau est une fonction de détection d'*onset*, ODF.

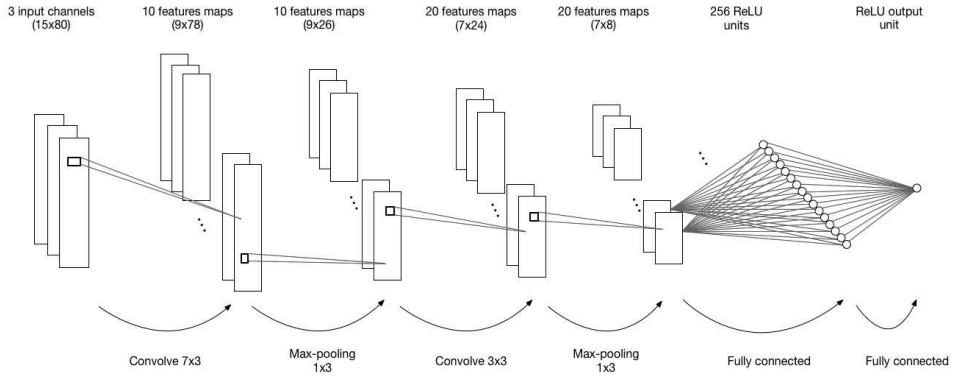

Figure 7.2 – *Une des configurations du réseau de neurones convolutif.*

Le post-traitement suit un schéma classique de détection d'*onset*. Les maximaux locaux de la fonction de détection sont identifiés puis filtrés par un seuil de 0.5. Chaque *feature map* peut être considérée comme la convolution de l'entrée avec un filtre. Nous entraînons ce réseau à ne détecter que les *onsets* percussifs.

Comme dans (Schlüter and Böck, 2014), on applique un dropout de 50% après la première couche dense afin de réduire le sur-apprentissage du réseau.

Fonction de coût		Activation correspondante	
CE	$C = -\frac{1}{n}\sum_x (y \ln a + (1-y)\ln(1-a))$	Sigmoïde	$f_{sig}(z) = \frac{1}{1+\exp(-z)}$
MSE	$C = \frac{1}{2n}\sum_x \|\|y-a\|\|^2$	ReLU	$f_{ReLU}(z) = \max(0, z)$

Tableau 7.1 – *Fonctions de coût comparées et leurs activations correspondantes.*

7.3 Comparaison de différentes configurations

Les performances des réseaux de neurones sont influencées notamment par le choix des hyper-paramètres, c'est-à-dire les paramètres qui sont fixés avant l'étape d'apprentissage. Dans un premier temps, on compare deux fonctions de coût différentes. Dans un second temps, les structures d'entrée sont comparées. En effet, dans (Kelz et al., 2016), plusieurs hyper-paramètres pour la transcription du piano sont comparés afin de les classer par ordre d'importance. La structure des données d'entrée s'élève à la deuxième position du classement après le *learning-rate*. Ici, l'optimiseur Adam est utilisé pour optimiser le *learning-rate*. Enfin, la taille des filtres utilisés pourrait aussi avoir de l'influence sur la performance du détecteur d'*onsets*. Nous comparons, dans cette section, plusieurs combinaisons de taille de filtres différentes pour les deux couches de convolution.

7.3.1 Fonction de coût

La fonction de coût est utilisée pour diriger l'optimisation du réseau de neurones. Elle mesure la divergence entre la valeur prédite (la sortie du réseau) et la valeur ciblée (les annotations). Pour la tâche de détection d'*onsets*, la *cross-entropy* est usuellement utilisée. La détection d'*onsets* dans une trame s'approche d'une classification binaire : si l'*onset* est présent (sortie du réseau au-dessus d'un certain seuil), la trame est marquée par un 1 et s'il n'est pas présent par un 0.

Cependant, la génération de l'ODF par le réseau peut être réalisée en minimisant la fonction de coût MSE. En appliquant ensuite le post-traitement des algorithmes de détection d'*onsets* (seuillage, localisation des maxima locaux), les trames contenant un *onset* peuvent être marquée d'un 1 et les autres d'un 0 comme pour l'entraînement avec l'entropie croisée binaire (*binary cross-entropy*).

On compare donc l'utilisation de ces deux fonctions de coût pendant la phase d'apprentissage. D'un côté, on associe la *binary cross-entropy* à une activation sigmoïde. Le seuil est alors de 0,5 car la sortie de la fonction sigmoïde peut être considérée comme une probabilité donc comprise entre 0 et 1. De l'autre côté, la sortie linéaire ReLU est utilisée avec la fonction de coût MSE. Les fonctions de coût et les activations correspondantes sont détaillées dans le tableau 7.1 avec n le nombre total de données d'entrée, x les données d'entrée, y la prédiction du réseau leur correspondant, a la sortie attendue et z l'entrée de la couche.

Comme le CNN est continu par rapport à tous les paramètres, l'ODF qu'il produit

est elle aussi continue. Il est donc plus difficile d'obtenir des pics (idéalement, placés aux trames contenant un *onset*) se rapprochant d'une impulsion de Dirac. Par conséquent la fonction cible est construite en étendant l'annotation à trois trames centrées sur l'*onset*. Cette extension permet de forcer un peu plus le réseau à représenter correctement les annotations cibles et, en plus, de réduire les problèmes d'alignement des annotations. D'après nos expériences, l'extension des annotations permet aussi de réduire le temps d'apprentissage et d'améliorer légèrement les résultats.

7.3.2 Structure des données d'entrée

Les entrées des réseaux que l'on veut comparer peuvent avoir différentes structures. Est-il avantageux d'avoir plusieurs spectrogrammes avec différentes résolutions comme entrée ou est-ce qu'une seule résolution suffit ? Dans le deuxième cas, quelle résolution donne les meilleurs résultats ?

Dans (Schlüter and Böck, 2014), ils utilisent trois spectrogrammes Mel logarithmiques obtenus avec trois résolutions temps-fréquence différentes. Ils appliquent la TFCT avec un pas d'avancement de 10 ms et des tailles de fenêtres d'analyse de 23 ms, 46 ms et 93 ms. Comme les spectrogrammes doivent avoir la même taille, ils sont filtrés par un banc de filtre Mel couvrant les *bins* fréquentiels de 27,5 Hz à 16 kHz. On appellera cette représentation *Multi Channel Mel Spectrogram* (MCMS). À l'inverse, dans (Wang et al., 2017), ils utilisent comme entrée une seule transformation à Q constant.

Dans ce chapitre, on compare plusieurs structures d'entrée : on teste deux valeurs de taille de fenêtre 0.064 s et 0.125 s et quatre nombres de bandes Mel : 116, 174, 231 et 289 bandes. Huit structures différentes sont donc utilisées en entrée des réseaux auxquelles on ajoute la structure MCMS.

7.3.3 Taille des filtres de convolution

En traitement de l'image, les filtres des réseaux de neurones convolutifs sont habituellement carrés. La question de la forme des filtres pour le traitement du son peut se poser. Bien que le spectrogramme puisse être considéré comme une "image du son", les deux dimensions de cette image ne représentent pas les mêmes grandeurs. On peut penser que suivant la tâche que le réseau est amené à effectuer la forme de filtres peut avoir de l'importance.

Par exemple pour la détection d'*onset*, le réseau est supposé détecter des changements d'énergie suivant le temps. Les filtres rectangulaires étroits dans la dimension fréquentielle semblent plus adaptés à cette tâche. On compare plusieurs formes et tailles de filtres données dans le tableau 7.2.

7.4 Expériences et résultats

7.4.1 Bases de données

Deux bases de données sont utilisées pour comparer les différentes configurations : la base de données RWC et ENST-Drums. Elles sont détaillées dans les sections 2.1.2.1 et 2.1.2.2.

7.4.1.1 RWC

RWC contient des morceaux de différents styles (section 2.1.2.1). Deux genres ont été choisis pour construire la base de données : Pop et Jazz. Seules les pistes contenant de la batterie ont été retenues.

La base de données Jazz contient 34 morceaux et la base Pop 100 morceaux enregistrés au format MIDI. Chaque morceau a été généré avec trois *soundfonts* différents : FluidR3_GM, GuGS_1.47 et HQOrchestralSFCollv2.1.1.

La base finale sur laquelle on réalise l'apprentissage comporte finalement 102 morceaux de Jazz et 300 de Pop. On l'agrémente en plus d'enregistrements des trois éléments de batterie joués seuls de IDMT-SMT-Drums (Dittmar and Gärtner, 2014). On retire quatre pistes pour l'évaluation que le réseau ne rencontrera jamais pendant l'apprentissage.

7.4.1.2 ENST-Drums

Avec la base de données ENST-Drums, une validation croisée est réalisée. Les réseaux sont appris sur tous les morceaux disponibles de deux des trois batteurs et sont évalués sur les morceaux avec accompagnement du dernier. Les trois combinaisons possibles de batteur sont exploitées.

Les *minus-one* (voir section 2.1.2.2) sont présentés au réseau avec accompagnement. Comme dans (Paulus and Klapuri, 2009), un facteur de mixage de 2/3 est appliqué à la batterie et de 1/3 à l'accompagnement. Certains morceaux des bases d'apprentissage sont mis de côté afin d'éviter le sur-apprentissage et déterminer le réseau optimal avant l'évaluation sur la base test.

7.4.2 Expériences et résultats

7.4.2.1 Fonction de coût

Pour comparer les deux fonctions de coût, plusieurs réseaux sont entraînés avec différentes configuration de données : quatre nombres de bandes Mel (116, 174, 231 et 289) et deux tailles de fenêtre pour la TFCT (0.064 s et 0.125 s) ainsi que la configuration qu'on a appelée MCMS (7.3.2).

Les réseaux sont entraînés à détecter les *onsets* des trois éléments principaux des parties percussives du Jazz et de la Pop (Charleston, grosse caisse et caisse claire) dans

la base de données RWC et ils sont testés sur la partie que l'on a extraite et mise de côté pour l'évaluation de cette même base de données.

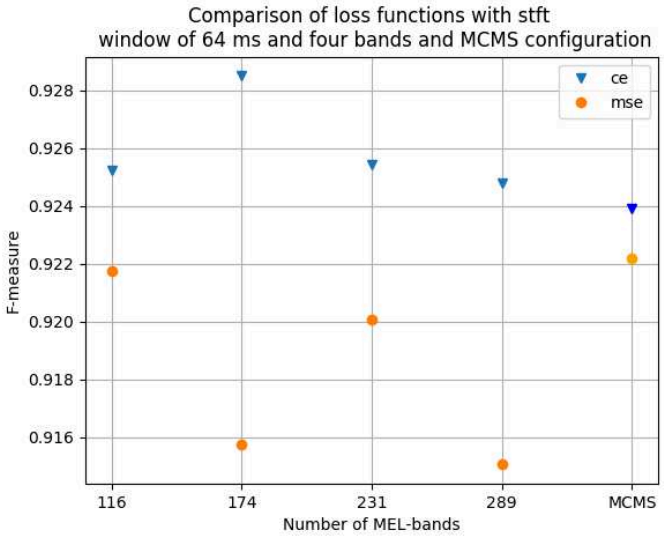

Figure 7.3 – *Comparaison des fonctions de coût sur la base RWC : binary cross entropy and mean square error, pour une fenêtre de 0.064 s pour la TFCT.*

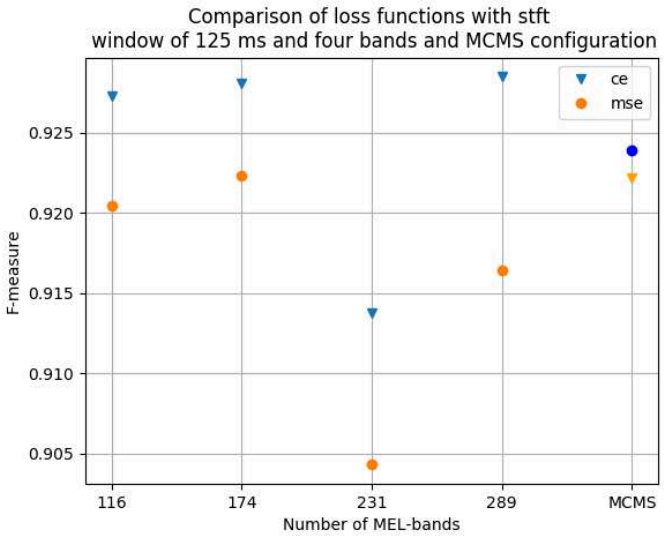

Figure 7.4 – *Comparaison des fonctions de coût sur la base RWC : binary cross entropy and mean square error, pour une fenêtre de 0.125 s pour la TFCT.*

Dans les figures 7.3 et 7.4, les résultats obtenus avec la cross-entropy surpassent ceux

obtenus avec le MSE. Pour la suite, nous utiliserons donc essentiellement la *binary cross entropy*.

Il est intéressant de noter que les résultats sont encourageants avec toutes les F-measure supérieures à 90%. On peut aussi voir qu'il n'y a pas de différences significatives entre les différentes structures de données.

7.4.2.2 Structure des données d'entrée

Pour appuyer la pertinence des résultats de comparaison des structures de données, des expériences de cross-validation sont réalisées pour détecter les *onsets* percussifs de la base ENST-Drums. L'apprentissage se fait sur deux batteurs et l'évaluation sur le troisième puis un roulement est appliqué. Toutes les pistes disponibles sont utilisées pour la base de données d'apprentissage des poids et on évalue seulement les mix des *minus-one* des mêmes batteurs pour déterminer les poids optimaux. On teste enfin le réseau obtenu sur les *minus-one* du dernier batteur.

On compare différentes structures de données d'entrées : deux tailles de fenêtre de TFCT (64 ms et 125 ms) et quatre nombres de bandes Mel (116, 174, 231 et 289) ainsi que la représentation MCMS. La figure 7.5 présente la moyenne des résultats des trois expériences de validation croisée pour la détection d'*onsets* percussifs.

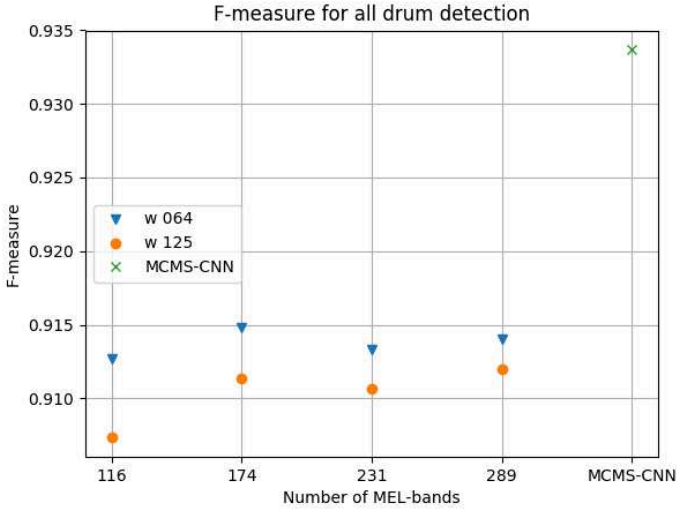

FIGURE 7.5 – *Comparaison des différentes structures des données d'entrée pour la détection des onsets percussifs dans la base de données ENST-Drums.*

Contrairement à RWC, l'utilisation de la représentation MCMS (trois spectrogrammes de différentes résolutions en entrée) semble donner de meilleurs résultats pour la base ENST-Drums avec 93,5 % contre 91,5 % pour les meilleurs résultats avec un seul spectrogramme Mel en entrée. Le fait que la représentation MCMS donne des meilleurs résultats qu'avec un seul spectrogramme en entrée pour ENST-Drums alors qu'ils étaient

équivalents pour RWC suggère que l'utilisation de plusieurs résolutions permet d'obtenir une meilleure robustesse du réseau pour la détection d'*onsets*.

Il est intéressant de regarder à quel point le MCMS entraîné pour détecter seulement les trois éléments cibles change sa performance. Pour cela, on utilise la configuration MCMS fournie dans l'algorithme madmom (Böck et al., 2016). Les deux détecteurs sont évalués sur une partie de la base de données RWC. On constate que le détecteur d'*onsets* spécifiques (*onsets* percussifs) améliore considérablement les performances de détection en terme de F-measure. Le détecteur d'*onsets* généraux obtient un score de 86,8% pour la détection de tous les *onsets* alors que le détecteur d'*onsets* spécifiques obtient un score de 93,2% pour la détection d'*onsets* percussifs.

7.4.2.3 Taille des filtres de convolutions

Pour évaluer l'influence de la taille des filtres sur la performance du réseau MCMS sur la détection d'*onsets* spécifiques, on procède à différents apprentissages des paramètres avec différentes combinaisons de filtres. On teste pour cela trois tailles pour les filtres des première et deuxième couches avec une validation croisée sur la base de données ENST-Drums. On donne dans le tableau 7.2 les résultats de chaque évaluation ainsi que la moyenne des trois expériences croisées pour chaque configuration de filtres.

Couche 1		3×7			3×3			7×3		
Couche 2		3×7	3×3	7×3	3×7	3×3	7×3	3×7	3×3	7×3
App.	Eval.	F-mesure			F-mesure			F-mesure		
1 & 2	3	93.7	93.6	93.6	93.8	93.8	93.1	93.9	93.5	93.4
2 & 3	1	93.3	93.4	93.3	93.5	92.9	92.6	93.3	93.3	93.2
3 & 1	2	93.9	93.8	93.9	94.1	94.1	93.9	94.0	94.0	94.0
Moyenne		93.6	93.6	93.6	**93.8**	93.6	**93.2**	93.7	93.6	93.5

Tableau 7.2 – *Influence de la taille des filtres de convolution dans l'expérience de validation croisée de ENST-Drums.*

La F-mesure moyenne la plus basse est obtenue avec la configuration 3×3 pour la couche 1 et 7×3 pour la deuxième couche. On obtient la meilleure F-mesure avec la configuration 3×3 pour la première couche et 3×7 pour la deuxième. Cependant l'écart entre les résultats est de seulement 0.6 point. La taille des filtres ne semble donc pas énormément influencer les performances du réseau pour la détection d'*onsets* spécifiques.

7.5 Conclusion

La position des *onsets* est une information essentielle pour l'algorithme de transcription de la batterie utilisé dans ce document. En plus de réduire le temps de calcul, cela permet de ne pas risquer de détecter des *onsets* aux endroits où il n'y en a pas. Le détecteur d'*onsets* précédemment utilisé détectait tous les *onsets* qu'ils soient percussifs ou harmoniques. Certains segments étaient donc analysés alors qu'aucun instrument per-

cussif n'y est présent. Cependant, ne pas en détecter assez augmente le nombre de faux négatifs c'est-à-dire le nombre d'*onsets* qui auraient dû être détectés mais qui ne l'ont pas été. Le signal autour de ces *onsets* ne peut donc pas être analysé. Ces *onsets* sont définitivement mis de côté.

Le but de ce chapitre était d'élaborer un détecteur d'*onsets* donnant de meilleurs résultats que le précédent et surtout qui détecte des *onsets* spécifiques. Ce détecteur d'*onsets* ne détecte que les *onsets* des trois instruments principaux de la batterie (la grosse-caisse, la caisse-claire et la Charleston) mais sans donner quel instrument en est responsable. Les segments étudiés contiennent donc en théorie au moins un *onset* de batterie.

Plusieurs approches de configuration de réseau de neurones convolutif pour la détection d'*onsets* afin d'améliorer la transcription automatique de la batterie ont été abordées. Nous avons comparé deux *loss function* différentes : la *cross-entropy* et la *mean square error*. Plusieurs configurations d'entrée ont été testées : plusieurs résolutions de spectrogramme pour une entrée avec un unique spectrogramme ainsi qu'une entrée avec trois spectrogrammes de différentes résolutions. Enfin, différentes combinaisons de tailles de filtres de convolution ont permis de montrer que les résultats ne différaient pas énormément suivant la taille des filtres. Le réseau convolutif avec comme entrée la représentation MCMS et la *cross-entropy* comme fonction de coût permet d'obtenir les meilleurs résultats avec plus 90% des *onsets* percussifs détectés.

Cet algorithme de détection d'*onsets* spécifiques peut remplacer l'algorithme basé sur le flux spectral qui détectait tous les *onsets* dans l'étape préalable à l'algorithme de transcription automatique de la batterie. De plus, nous avons montré qu'un réseau peut reconnaître des *onsets* particuliers. Il pourrait être possible de détecter des *onsets* encore plus spécifiques, par exemple tous les *onsets* dont un seul instrument est responsable et ainsi transcrire seulement sa partie.

— Chapitre 8 —

CNN pour la transcription automatique de la batterie

Les étapes de prétraitement sont importantes pour mettre en forme le signal afin de faire ressortir les caractéristiques pertinentes ou pour réduire les coups de calcul. Cependant, l'utilisation de ces prétraitements ou d'algorithmes complémentaires peuvent faire apparaître des erreurs. Dans notre cas, quand l'algorithme de détection d'*onsets* fournit à l'algorithme de transcription de la batterie la position d'un *onset* mais qu'aucun instrument de batterie n'est responsable de cet *onset*, un faux positif peut être détecter. Au contraire, si le détecteur d'*onsets* ne détecte pas un des *onset*, l'algorithme de transcription n'analysera pas le segment et ne pourra pas détecter la présence d'un des instruments de la batterie.

Au chapitre précédent, un détecteur d'*onsets* basé sur les CNN a été élaboré pour détecter uniquement les *onsets* percussifs. Ces *onsets* sont fournis à l'algorithme de transcription de la batterie. De plus, les résultats encourageants pour la détection d'*onsets* spécifiques avec les réseaux de neurones du chapitre précédent ainsi que les nouveaux résultats obtenus dans la littérature motivent un approfondissement de l'utilisation des réseaux de neurones pour la transcription de la batterie.

Dans un premier temps, les résultats du détecteur d'*onsets* sont fournis à l'algorithme de transcription de la batterie basé sur la NMD. Par ailleurs, le détecteurs d'*onsets* spécifique basé sur les CNN sont entraînés à détecter des *onsets* encore plus spécifiques : les *onsets* d'un unique instrument. Pour entraîner ces réseaux, de nombreuses données sont nécessaires mais peu sont disponibles et correctement annotées. L'utilisation de l'augmentation de données pourrait permettre de mieux généraliser les réseaux. Enfin, l'utilisation d'un réseau multi-sorties permettrait la transcription automatique de la batterie avec un unique réseau.

8.1 État de l'art : Les réseaux de neurones pour la transcription automatique de la batterie

Les RNN, utilisés pour la transcription de la parole (Sak et al., 2014), sont appliqués à la transcription de la musique pour le piano dans (Böck et al., 2012). En 2016, les RNN sont utilisés pour la première fois dans (Vogl et al., 2016b) et (Southall et al., 2016) pour la transcription de la batterie. Dans (Vogl et al., 2016b), la batterie est transcrite à partir de fichier où elle joue toute seule alors que dans (Southall et al., 2016) la tâche est complexifiée par l'ajout d'instruments harmoniques.

Dans (Southall et al., 2016), ils utilisent un réseau de neurones récurrent bidirectionnel, *Bidirectional Recurrent Neural Network* (BDRNN), pour la transcription de la batterie dans des morceaux avec d'autres instruments. Les BDRNN sont des RNN à deux sous-couches : une couche avec des connexions de récurrence dans la direction *forward* et l'autre dans la direction *backward*.

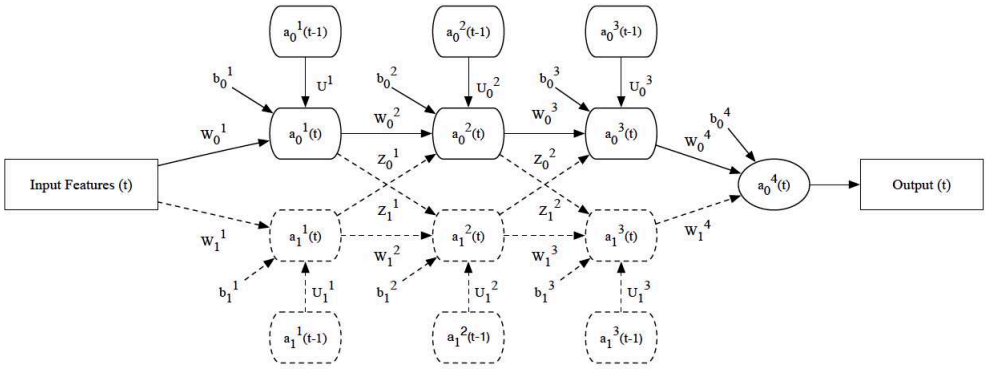

FIGURE 8.1 – *BRNN proposé par (Southall et al., 2016). Les lignes pleines sont les connexions correpondantes au RNN et les lignes en pointillés au BDRNN.*

Ce type de réseau permet de prendre en compte les informations passées ainsi que les informations futures par rapport à un temps t. De ce fait, l'utilisation de ces réseaux en temps réel n'est pas possible bien que l'utilisation de courtes séquences peut réduire le temps de traitement.

Dans (Vogl et al., 2016b), ils comparent les résultats obtenus avec des BDRNN avec des RNN traditionnels mais en utilisant un décalage temporel des annotations sur différentes bases de données (IDMT-SMT-Drums et différentes combinaisons de ENST-Drums). Toutes les annotations sont décalées de +25 ms. Le réseau a alors accès aux informations qui précédent l'événement mais aussi celles qui suivent. Il est donc plus facile au RNN avec décalage des annotations d'exécuter la tâche demandée en temps réel (avec un léger délai) que les BDRNN. Il apparaît que le RNN avec décalage des annotations (tsRNN) est plus performant que le modèle BDRNN.

Des cellules de mémoire interne, les LSTM, peuvent être ajoutées aux connexions de récurrence des RNN. Cela permet au réseau d'apprendre des dépendances avec des

données plus éloignées. Dans (Southall et al., 2017b), des BDRNN sont couplés avec des LSTM. Dans (Vogl et al., 2016a) des GRU sont combinés au décalage temporel des annotations. Les LSTM et les GRU sont présentés dans la section 4.3.1.

Enfin l'utilisation des CNN a donné des résultats prometteurs dans (Southall et al., 2017b).

8.2 NMD combinée avec l'algorithme de détection d'onsets percussifs

L'algorithme de transcription de la batterie, détaillé en section 3.1.4, basé sur la NMD est informé préalablement avec la position des *onsets*. Un algorithme de détection d'*onsets* détecte les événements qui ont lieu dans le morceau étudié. Puis l'algorithme de transcription étudie le signal autour des positions détectées.

À ces positions plusieurs configurations peuvent apparaître. Un instrument harmonique joue seul. L'algorithme NMD ne devrait donc pas trouver d'*onset* pour un des éléments de batterie et seules les bases allouées au *background* (cf. section 5.2) sont activées. Un instrument percussif est joué seul et fait partie des trois principaux éléments de batterie, l'algorithme doit activer au moins un motif correspondant à cet instrument. Enfin, plusieurs instruments jouent ensemble, l'algorithme doit activer des motifs de *background* ainsi que des motifs relatifs aux éléments de batterie qui jouent.

Le terme de régularisation (cf. section 5.2.3) pénalise l'activation des motifs de *background*. Si l'algorithme détecte des *onsets* où seuls des instruments harmoniques jouent, l'algorithme de transcription NMD risque de détecter un élément de batterie. Notamment si l'instrument harmonique qui joue se rapproche d'un des éléments de la batterie, comme la basse de la grosse-caisse.

L'idée de ce chapitre est de combiner un détecteur d'*onsets* spécifiques, ici un détecteur des trois principaux éléments de la batterie à transcrire (Charleston, caisse-claire et grosse-caisse) afin que l'algorithme de transcription NMD puisse les différencier. Ainsi, si le détecteur est parfait, on est sûr qu'il y a au moins un *onset* percussif dans le segment. On ne présente plus de segment où aucun instrument à transcrire n'est présent. Cela permet d'éviter certains faux positifs.

Le détecteur d'*onsets* spécifiques (des trois principaux instruments de la batterie, cf. 7) est détaillé dans la figure 8.2.

L'évaluation des performances de l'algorithme est réalisée sur la base de données ENST-Drums avec une validation croisée (cf. section 7.4.1.2). Les résultats sont données dans le tableau 8.1 et sont comparés aux résultats obtenus dans (Paulus and Klapuri, 2009) et à ceux obtenus avec le modèle de (Southall et al., 2017a) dont le code est disponible [1]. On ajoute les résultats qu'on obtient par le même procédé de validation croisée avec l'ancien détecteur d'*onsets* basé sur le flux spectral, (Elowsson and Friberg, 2013), et l'algorithme basé sur la NMD avec un dictionnaire comportant 15 bases par instrument. Les premiers tests se font sans l'adaptation des bases du dictionnaire. La

1. `https://github.com/CarlSouthall/ADTLib`

FIGURE 8.2 – *Configuration du CNN pour la détection d'onsets spécifiques.*

première expérience permet de voir si alimenter l'algorithme qu'avec les *onsets* percussifs permet d'obtenir des meilleurs résultats qu'avec les *onsets* généraux.

Méthodes	Mesure	BD	SD	HH
HMM+	P(%)	80.2	66.3	84.7
MLLR (Paulus and Klapuri, 2009)	R(%)	81.5	45.3	82.8
	F(%)	80.8	53.9	83.6
Soft Attention	P(%)	98.5	88.2	67.8
mechanisms (Southall et al., 2017a)	R(%)	62.2	40.1	87.9
	F(%)	72.0	53.7	76.4
NMD alimenté	P(%)	79.9	48.4	76.7
par le détecteur	R(%)	81.3	67.0	70.0
d'*onsets* (flux spectral)	F(%)	80.5	55.8	72.8
NMD alimenté	P(%)	79.6	68.8	72.6
par le détecteur	R(%)	64.7	43.9	67.1
d'*onsets* CNN	F(%)	68.9	52.6	68.3

TABLEAU 8.1 – *Résultat de la transcription dans la validation croisée de ENST-Drums.*

Alimenter l'algorithme de transcription NMD avec les *onsets* fournis par le détecteurs d'*onsets* spécifiques ne permet d'atteindre ni les résultats de Paulus ni ceux de Southall. De plus l'algorithme NMD avec le détecteur d'*onsets* généraux donne de meilleurs résultats qu'avec le détecteur d'*onsets* spécifiques. En analysant le rappel et la précision, on peut voir que l'algorithme avec le détecteur d'*onsets* spécifiques détecte moins d'*onsets* (rappel bas) mais il fait moins d'erreur lorsqu'il les détecte (précision plus haute). Alors qu'en général c'est l'inverse pour l'algorithme avec le détecteur d'*onsets* généraux.

Bien que les résultats de l'algorithme de transcription basé sur la NMD combiné au détecteur d'*onsets* spécifiques n'atteignent pas les résultats escomptés, les résultats du détecteur d'*onsets* spécifiques basé sur un réseau de neurones convolutif sont très

encourageants et ce dernier pourrait être entraîné à ne détecter qu'un seul élément de la batterie pour permettre sa transcription.

8.3 CNN POUR CHAQUE INSTRUMENT

En gardant la même représentation MCMS, trois réseaux ont été entraînés à ne détecter qu'un seul des trois éléments principaux de la batterie : Charleston, grosse-caisse et caisse-claire. Chacun des trois réseaux est évalué par une validation croisée sur les trois batteurs de ENST-Drums (section 2.1.2.2). On donne dans le tableau 8.2 les résultats moyens de cette évaluation et les résultats de (Paulus and Klapuri, 2009) qui utilise les modèles de Markov cachés (HMM, *hidden markov model*) et la *Maximum Likelihood Linear Regression* (MLLR) et ceux de (Southall et al., 2017b) obtenus avec un réseau de neurones avec *Soft Attention Mechanism*. On rappelle les résultats obtenus avec l'algorithme basé sur la NMD informé par la position des *onsets* généraux (Elow.) et spécifiques (CNN).

FIGURE 8.3 – *Configuration des CNN pour la transcription de la batterie.*

Méthodes	BD			SD			HH			Batterie		
	P	R	F	P	R	F	P	R	F	P	R	F
Elow. + NMD	79,9	81,3	80,5	48,4	67,0	65,8	76,7	70,0	72,8	-	-	-
CNN + NMD	79,6	64,7	68,9	68,8	43,9	52,6	72,6	67,1	68,3	-	-	-
HMM + MLLR	80.2	81.5	**80.8**	66.3	45.3	53.9	84.7	82.8	**83.6**	79.0	70.9	74.7
SA *mechanisms*	98.5	62.2	72.0	88.2	40.1	53.7	67.8	87.9	76.4	92.4	82.1	86.9
MCMS+CNN	77.5	75.0	76.2	57.9	67.0	**62.1**	71.0	89.7	79.3	93.7	93.0	**93.4**

TABLEAU 8.2 – *Résultat de la transcription de chacun des trois principaux éléments de batterie dans la validation croisée sur ENST-Drum avec trois réseaux individuels.*

Les réseaux donnent de bons résultats comparables à l'état de l'art. Ils gagnent 8 points pour la caisse-claire mais perdent environ 4 points pour les deux autres éléments par rapport aux résultats de (Paulus and Klapuri, 2009). Cependant pour les trois ins-

truments les résultats sont plus élevés que ceux obtenus avec l'algorithme de (Southall et al., 2017b) (SA *mechanisms*).

Comparons les résultats dans le cas de la grosse-caisse entre les différentes configurations de la validation croisée. Les résultats sont présentés dans le tableau 8.3.

Apprentissage	Evaluation	P	R	F
1 et 2	3	82.5	96.7	89.1
2 et 3	1	75.1	36.7	45.0
3 et 1	2	74.6	98.1	84.8

TABLEAU 8.3 – *Résultats de la transcription de la grosse-caisse dans chaque configuration de la validation croisée sur ENST-Drums.*

On remarque que la détection de la grosse-caisse des batteurs 2 et 3 est performante alors que la détection de la grosse-caisse du batteur 1 est très basse. L'écoute des parties de grosse-caisse de chaque batteur révèle que le signal du batteur 1 est très bas par rapport aux deux autres.

Nous avons essayé de combler la différence en ajoutant des mixages avec différents taux pendant l'apprentissage mais cela n'a donné aucune amélioration. En observant le spectrogramme, il est possible de remarquer que le signal du batteur 1 contient des *onsets* beaucoup moins prononcés que les autres batteurs. Ce qui peut aussi expliquer cette différence de résultats. La haute spécificité du CNN amène à un sur-apprentissage qui réduit le *recall* pour le batteur 1. Le modèle HMM ne rencontre apparemment pas ce problème. Cela peut indiquer que le CNN qui pourtant fonctionne bien pour la détection d'*onsets* percussifs, est trop complexe pour détecter un instrument en particulier.

Une idée pour améliorer la détection de la grosse-caisse du batteur 1 est de normaliser le signal sur le temps pour chaque bande de fréquence. Cette normalisation permet de mettre en valeur les changements brutaux d'énergie, caractéristiques d'un *onset*. Cependant, ce type de normalisation modifie les relations d'énergie entre les différentes bandes.

Méthodes	BD			SD			HH			Batterie		
	P	R	F	P	R	F	P	R	F	P	R	F
Elow. + NMD	79,9	81,3	80,5	48,4	67,0	65,8	76,7	70,0	72,8	-	-	-
CNN + NMD	79,6	64,7	68,9	68,8	43,9	52,6	72,6	67,1	68,3	-	-	-
HMM + MLLR	80.2	81.5	**80.8**	66.3	45.3	53.9	84.7	82.8	**83.6**	79.0	70.9	74.7
SA *mechanisms*	98.5	62.2	72.0	88.2	40.1	53.7	67.8	87.9	76.4	92.4	82.1	86.9
MCMS+CNN	77.5	75.0	76.2	57.9	67.0	**62.1**	71.0	89.7	79.3	93.7	93.0	**93.4**
MCMS+CNN+N	84.0	80.7	**81.5**	54.2	68.1	59.4	71.9	86.6	77.8	93.8	91.7	92.7

TABLEAU 8.4 – *Résultat de la transcription de chacun des trois principaux éléments de batterie dans la validation croisée sur ENST-Drum avec trois réseaux individuels.*

L'énergie de la grosse-caisse étant localisée dans les basses fréquences, le réseau est capable de détecter correctement les *onsets*. En effet, la F-mesure du batteur 1 atteint

67.7% au lieu de 45%. Les résultats moyens de la validation croisées sont donnés dans le tableau 8.4 (MCMS+CNN+N). Bien que les performances pour la caisse-claire et la Charleston sont dégradées par rapport aux résultats sans normalisation, ils sont bien meilleurs pour la grosse-caisse.

Nous avons mis en évidence une transformation des données qui permet d'améliorer la détection de la grosse caisse. Cependant cette transformation dégrade légèrement la détection des autres instruments et reste particulière à cette base de données. La mauvaise généralisation du réseau appris avec les batteurs 2 et 3 peut venir du peu d'exemples différents présents dans la base de données. Ainsi, le manque d'exemples présents dans la base de données ENST-Drums et la particularité du batteur 1 diminue l'intérêt de cette base de données pour l'évaluation et les apprentissages des modèles. Cependant, le problème du manque de données annotées demeure un problème important pour l'utilisation des réseaux de neurones pour la transcription automatique de la batterie. Sans base de données supplémentaires, un moyen d'obtenir plus d'exemples est la *data augmentation*, l'augmentation des données.

8.4 Augmenter les données d'apprentissage

Un problème récurrent pour l'apprentissage des réseaux de neurones est le manque de données. La *data augmentation*, c'est-à-dire l'augmentation du volume de données, peut permettre de combler ce manque. En traitement d'image par exemple, si un objet est toujours présenté avec la même orientation au réseau pendant l'apprentissage, ce dernier risque de rencontrer des difficultés à le détecter s'il est présenté tourné. Pour la musique, c'est le même problème. Les données d'apprentissage disponibles ne peuvent pas couvrir toutes les possibilités. La façon dont un instrument sonne varie énormément d'un morceau à l'autre. De plus, comme il est difficile d'avoir des bases de données précisément annotées et que les annoter à la main prendrait énormément de temps, les bases de données ne contiennent pas autant de variété que l'on peut trouver dans le monde. Pourtant plus les réseaux sont entraînés sur des données présentant une grande variété, mieux ils se généralisent et plus ils sont performants.

L'augmentation des données consiste à appliquer différentes transformations aux signaux annotés pour créer de nouvelles données avec des annotations aussi précises qu'avant. Il est nécessaire que les données augmentées gardent la même signification physique que la donnée d'origine. Par exemple, une grosse-caisse doit sonner encore comme une grosse-caisse même après transformation. Toutes les informations étant contenues dans le signal, la qualité des transformations est extrêmement importante. Autrement, le réseau pourrait apprendre ses poids sur des données contenant des artéfacts et les considérer comme des caractéristiques de l'instrument. On a donc écouté les données transformées afin de les valider.

Certaines transformations qui préservent les annotations ont été présentées dans (McFee et al., 2015; Schlüter and Grill, 2015) comme le *time stretching* (étirement du temps) ou le *pitch shifting* (translation de la note). (Vogl et al., 2018) veut transcrire 18 classes d'instrument de batterie différentes. La distribution des instruments dans la base de don-

nées qu'il utilise (RBMA) n'est pas équilibrée. Certains instruments sont plus représentés que d'autres. L'idée était donc de resynthétiser la base de données avec une nouvelle distribution des instruments de batterie permettant d'obtenir un meilleur équilibre. Pour augmenter le nombre de données et d'exemples de batterie utilisés, nous avons synthétisé la base RWC avec trois bases de sons (*soundfonts*) différentes (Jacques and Röbel, 2018). La boîte à outils MUDA de (McFee et al., 2015) est utlisée par (Salamon and Bello, 2017) pour transformer des sons d'environnement pour leur classification.

8.4.1 Les transformations

Quatre transformations sont étudiées dans cette section : remixage des parties bruit et sinuoïdales, remixage des attaques, transposition avec ou sans compensation temporelle incluant différents degrés de transposition de l'enveloppe spectrale. Toutes ses transformations sont effectuées sur le signal audio avant tout autre pré-traitement et avant la conversion en représentation MCMS. Toutes les transformations sont réalisées avec la version 3 du programme AudioSculpt (Bogaards et al., 2004).

- **Remixage des parties bruit et sinusoïdales** : Après avoir classifié les pics sinusoïdaux et les pics de bruit du signal audio (Zivanovic et al., 2004), ils sont séparés par l'application d'un masque spectral puis remixer avec un facteur choisi que l'on notera pour la suite r_n. Ce traitement conduit à une modification de la balance d'énergie entre les composantes sinusoïdales et les composantes de bruit. Les attaques contiennent différents niveau de bruit et de transitoires. Les remixer change leur distribution d'énergie et cela peut être considéré comme un changement de propriétés de l'*onset*.
- **Remixage des attaques** : Après avoir détecté la localisation en temps et fréquences des transitoires (Röbel, 2003), ces derniers sont échelonnés en appliquant un facteur linéaire r_a pour accentuer ou adoucir l'attaque. Cette transformation permet de moduler les propriétés de l'attaque.
- **Transposition avec ou sans compensation temporelle** : La transposition est réalisée en ré-échantillonnant le signal sans changer l'échantillonnage auquel on joue le morceau. Le ré-échantillonnage implique un changement de tempo et une évolution plus rapide des attaques. Ces changements temporels peuvent être compensés par l'application d'un vocoder de phase (Laroche and Dolson, 1999) et avec la préservation des transitoires (Röbel, 2003). On note t la transposition en cents.
- **Transposition des enveloppes spectrales** : On ajoute à la transposition avec ou sans compensation temporelle une transposition de l'enveloppe spectrale. Tout d'abord, l'enveloppe spectrale est estimée (Röbel and Rodet, 2005) puis transposée alors que le *pitch* n'est pas modifié. Cet effet implique un changement de la couleur du son. On note t_e le paramètre de transposition en cents.

8.4.2 Evaluation des différentes transformations

8.4.2.1 Stratégie d'évaluation

Chaque transformation est évaluée individuellement. Pour réaliser l'évaluation de la transformation, on utilise la validation croisée. La base de données est divisée en plusieurs sous-ensembles. Le réseau est entraîné sur les différents sous-ensembles sauf un et la performance de généralisation est évaluée sur le dernier sous-ensemble. Afin d'éviter le sur-apprentissage et de déterminer le réseau optimal plusieurs morceaux des sous-ensembles d'apprentissage sont mis de côté et forment la base test.

Les réseaux sont donc appris sur les sous-ensembles d'apprentissage et leurs versions augmentées, testés sur les morceaux originaux mis de côté et évalués sur le dernier sous-ensemble original.

8.4.2.2 Bases de données

Deux expériences de validation croisée sont réalisées : la première sur la base de données ENST-Drums (cf. section 2.1.2.2) et la deuxième sur la base de données fournie par MIREX (cf. section 2.1.2.3) pour le challenge de transcription automatique de la batterie. La première est composée de trois sous-ensembles : batteur 1, batteur 2 et batteur 3. La deuxième se compose de quatre sous-ensembles : 2005, GEN, MEDLEY et RBMA. Les différentes transformations sont appliquées à ces sous-ensembles avec les paramètres détaillés dans le tableau 8.5.

Facteur	Valeurs
r_a	0.1, 0.3, 0.6, 1.5, 2, 3
r_n	0.6, 1.5, 2, 3
t	-300,-200,-100,100,200,300
t_e	-300,-200,-100,100,200,300

Tableau 8.5 – *Paramètres de transformation pour les bases de données ENST-Drums et MIREX 2018.*

L'ensemble ENST-Drums représente environ 2,5 heures d'enregistrement. On obtient environ 14 heures avec le remixage des attaques, 9,5 heures pour le remixage de la partie bruit, 85,5 heures pour la transposition avec compensation du temps et enfin 85,5 heures sans la compensation du temps.

Les bases de données de MIREX 2018 représentent environ 3 heures d'enregistrement. Avec les transformations, on obtient environ 19 heures pour le remixage de la partie bruit, 12,5 heures pour le remixage des attaques, 113 heures pour la transposition avec compensation de temps et transposition de l'enveloppe spectrale et 113 heures sans la compensation de temps.

8.4.2.3 Résultats

Les apprentissages avec les bases de données augmentées sont comparés aux apprentissages sur les bases de données originales. Les résultats détaillés sont donnés dans les tableaux 8.6 et 8.8.

Trans.	**BD**			**SD**			**HH**		
	R	**P**	**F**	**R**	**P**	**F**	**R**	**P**	**F**
Originale	77.5	75.0	76.2	57.9	67.0	62.1	71.0	89.7	79.3
Remixage bruit	79.5	77.4	76.1	69.0	58.7	62.5	**90.1**	**73.0**	**79.9**
Remixage attaques	**80.8**	**76.2**	**76.8**	65.4	61.0	62.2	88.3	70.2	79.5
Transposition	80.4	76.3	76.3	**69.2**	**58.3**	**62.6**	89.6	71.7	79.0
Transp. sans comp.	79.1	76.4	75.7	65.6	61.1	61.9	90.2	71.1	78.8

TABLEAU 8.6 – *Résultats avec apprentissage sur les bases de données ENST-Drums augmentées.*

Pour la base de données ENST-Drums, aucune stratégie n'augmente radicalement les résultats mais la transposition sans compensation temporelle donne toujours des résultats moins élevés qu'avec l'apprentissage uniquement sur la base originale. De plus, les meilleures performances sont obtenues pour chaque instrument avec des stratégies différentes.

App.	Eval.	Orig.	Remix. bruit	Remix. attaques	Transp.	Trans. sans comp.
1 et 2	3	89.1	88.8	86.9	90.1	89.5
2 et 3	1	45.0	56.7	58.7	56.6	53.2
3 et 1	2	84.8	82.7	84.8	82.2	83.7

TABLEAU 8.7 – *F-mesure de la transcription de la grosse-caisse dans chaque configuration de la validation croisée sur ENST-Drums augmenté.*

Cependant si on compare les résultats détaillés pour la grosse-caisse, on peut voir que l'augmentation de données permet une meilleure généralisation de l'apprentissage sur les batteurs 2 et 3. Pour les autres apprentissages, les résultats sont un peu détériorés mais restent proches des résultats obtenus pour les apprentissages sans bases augmentées. Les différentes F-mesures obtenues pour chaque apprentissage pour la détection de la grosse-caisse sont données dans le tableau 8.7.

Pour la base de données MIREX 2018, la première chose à noter est que l'augmentation des bases de données permet d'améliorer plus franchement les résultats de la F-mesure excepté pour la Charleston avec les transpositions. En outre, avec le remixage des parties bruit et sinusoïdales, la F-mesure gagne 3 points pour la grosse-caisse et 2 points pour la caisse-claire en comparaison avec les modèles entraînés sans les données augmentées.

De plus, pour certaines transformations, non seulement le rappel est augmenté mais aussi la précision. Ainsi, non seulement les réseaux détectent plus d'*onsets* mais ils dé-

Trans.	BD			SD			HH		
	R	**P**	**F**	**R**	**P**	**F**	**R**	**P**	**F**
Originale	86.9	73.4	79.3	65.8	51.4	57.1	77.8	64.3	69.5
Remixage bruit	**89.1**	**73.0**	**82.3**	**66.3**	**55.3**	**59.6**	78.0	64.4	69.7
Remixage attaques	89.6	73.0	80.3	65.7	53.4	58.4	**79.4**	**64.0**	**70.0**
Transposition	90.6	67.0	82.0	69.0	52.4	58.6	80.0	61.8	68.7
Transp. sans comp.	89.9	74.2	81.2	67.9	53.3	59.1	79.5	62.0	68.6

Tableau 8.8 – *Résultats avec apprentissage sur les bases de données MIREX 2018 augmentées.*

tectent aussi moins de faux positifs. Cela semble indiquer que les réseaux apprennent plus de caractéritiques et les choisissent de façon plus précises. Ils deviennent invariants aux transformations employées.

8.5 CNN multi-sorties pour la transcription de la batterie

Les modèles utilisés à la section 8.3 sont constitués de trois réseaux indépendants, chacun dédié à un instrument. Ces réseaux détectent les *onsets* de l'instrument auquel ils sont dédiés. À chaque trame, la sortie est 1 si un *onset* est détecté, 0 autrement. La transcription de l'instrument est alors considérée comme une tâche de détection des événements de l'instrument.

Cependant, la tâche de transcription peut être aussi vue comme une classification multi-classes en utilisant un réseau multi-sorties. À une trame d'entrée, un score pour chaque classe est calculé par le réseau permettant de déterminer à quelle classe l'événement appartient.

8.5.1 Architecture générale

L'architecture du réseau multi-sorties est très similaire à celle détaillée à la section 7.2. La couche de sortie ne consiste plus en une seule sortie mais est constituée de trois unités. Chaque unité correspond à un instrument. Cette architecture permet de réaliser la transcription avec un seul réseau. Elle est représentée par la figure 8.4.

L'avantage de ce réseau est que les poids des premières couches sont communs aux trois instruments. Ainsi, moins de poids sont à apprendre et le réseau se focalise sur les caractéristiques communes aux trois instruments pour les deux premières couches. La troisième couche est segmentée en trois sous-couches, chacune dédiée à un instrument en particulier.

8.5.2 Comparaison des différents modèles

On compare 4 modèles de transcription de la batterie basés sur des CNN. Les modèles ont été appris sur la base d'entraînement fournie par MIREX 2018, détaillée à la section

7.4.1.2, pour le challenge de transcription de la batterie. La base de données est détaillée à la section 2.1.2

Le premier modèle est le modèle constitué de trois CNN indépendants détaillés dans la section 8.3 dont on a changé le nombre de filtres par couche. La première couche contient 20 filtres de taille 7×3, la deuxième 40 filtres de taille 3×3 et la couche dense contient 512 unités. On note JAR5 ce modèle en référence aux notations utilisées pour MIREX. Sa configuration est donnée figure 8.5

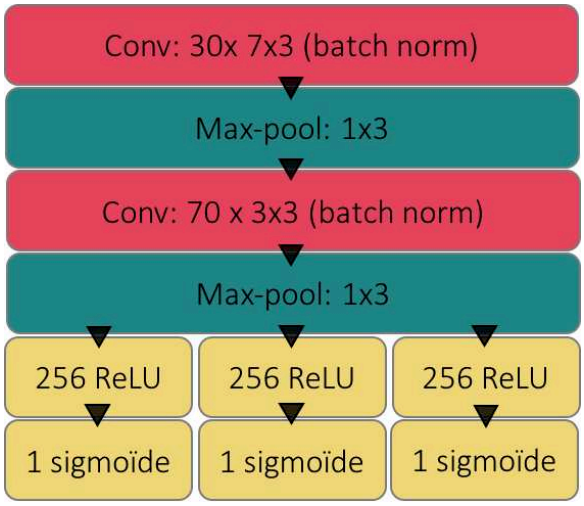

FIGURE 8.4 – Réseau multi-sorties pour la transcription de la batterie (JAR 1 et 2).

Les deuxième et troisième modèles détaillés en section 8.5.1 contiennent 30 filtres de taille 7×3 pour la première couche, 70 de taille 3×3 pour la deuxième et la troisième couche est divisée en trois sous-couches de taille 256 unités avec leur 4e couche correspondante. Leur configuration est donnée dans la figure 8.4. Ils sont différenciés par deux méthodes d'entraînement. La première, JAR1, minimise la fonction de coût entre le triplet d'entrée et le triplet estimé. La deuxième, JAR2, décompose la fonction de coût E comme la somme de trois composantes, chaque composante correspondant à un instrument :

$$\begin{aligned} E(x,y) &= E_{bd} + E_{sd} + E_{hh} \\ &= E(x_{bd}, y_{bd}) + E(x_{sd}, y_{sd}) + E(x_{hh}, y_{hh}) \end{aligned}$$

avec x l'annotation et y la prédiction. À chaque itération une des trois composantes E_{bd}, E_{sd}, E_{hh} est minimisée.

Durant les phases d'entraînement et de validation, JAR5 et JAR2 ont obtenu des résultats très différents pour le rappel et la précision. En effet, JAR5 a plutôt tendance a avoir un bon rappel alors que JAR2 a une meilleure précision. De plus JAR2 a obtenu de meilleurs résultats que JAR1. Une combinaison des deux modèles JAR5 et JAR2 est réalisée en calculant la moyenne des sorties correspondantes à chaque instrument.

Figure 8.5 – Paramètres des trois réseaux indépendants (JAR5).

	JAR5	JAR1	JAR2	JAR3
IDMT	0.62	0.63	0.63	0.64
KT	0.62	0.63	0.63	0.64
RBMA	0.67	0.68	0.70	0.69
MDB	0.66	0.68	0.65	0.67
GEN	0.76	0.81	0.79	0.78
Overall	0.67	**0.69**	0.68	**0.69**

Tableau 8.9 – *F-mesure moyenne sur la base de données d'évaluation MIREX 2018.*

Les résultats (F-mesure) obtenus au challenge MIREX 2018 sont détaillés sur chaque base de données dans le tableau 8.9 (Jacques et al., 2018). Le détail est donné sur le site internet de MIREX : `https://www.music-ir.org/mirex/wiki/2018:Drum_Transcription_Results`. Une marge d'erreur de 30 ms est tolérée, c'est-à-dire qu'un *onset* détecté est considéré comme correctement détecté s'il est à moins de 30 ms de l'annotation.

Le réseau multi-sorties JAR1 obtient les meilleurs résultats ainsi que la combinaison du modèle multi-sorties avec les trois réseaux indépendants. La figure 8.6 détaille les résultats de tous les algorithmes présentés au challenge. En plus de nos quatre modèles, huit autres modèles ont été présentés : quatre sont détaillés dans (Schroeter, 2018) (JS), deux dans (Vogl and Knees, 2018) (RV) et enfin deux dans (Southall, 2018) (CS). Les résultats sont une moyenne sur toutes les bases de données. Les barres verticales sont les moyennes de la F-mesure pour la détection des trois instruments. Les résultats obtenus pour chaque instrument sont détaillés par les barres horizontales avec de gauche à droite la grosse-caisse, la caisse-claire et la Charleston pour chaque algorithme. Les résultats sont rappelés dans le tableau 8.10.

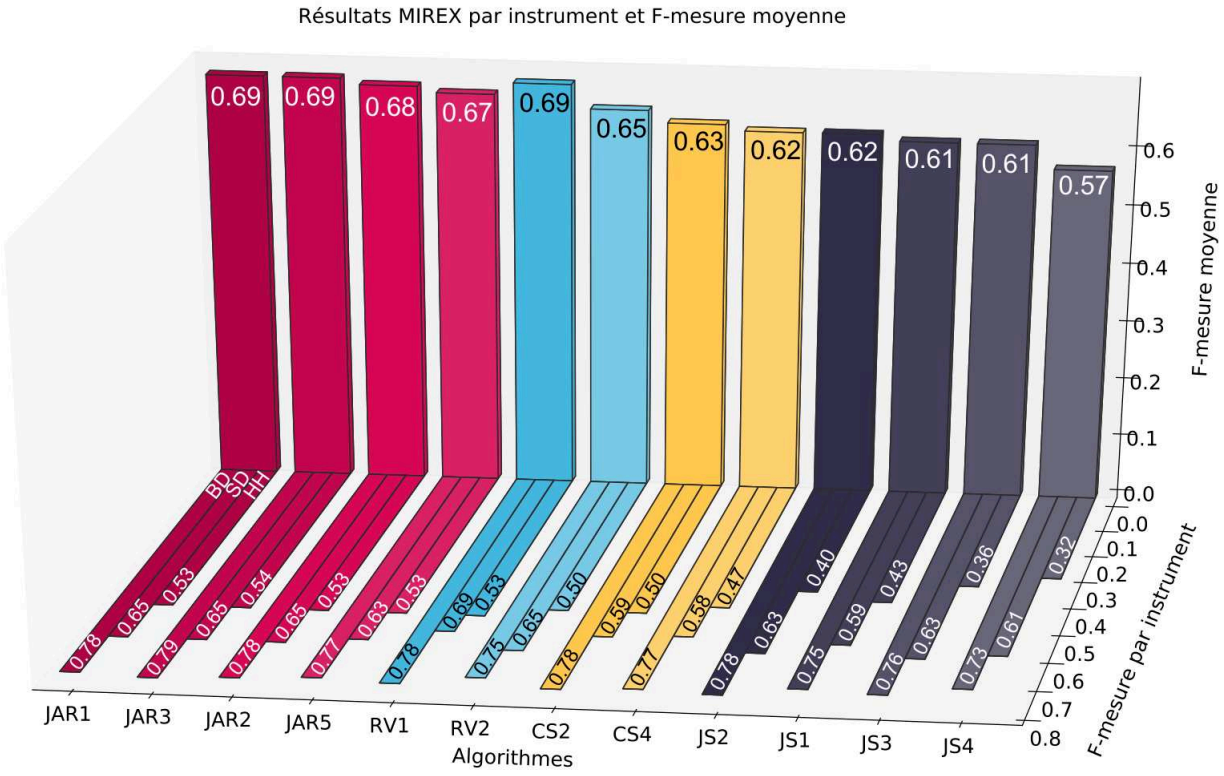

FIGURE 8.6 – *Résultats (F-mesure moyenne) des algorithmes présentés au challenge de transcription de la batterie à MIREX 2018.*

Deux de nos algorithmes, JAR1 et JAR3 ont obtenu les meilleurs résultats du challenge de cette année ainsi que le modèle basé sur des CNN de R. Vogl avec une F-mesure moyenne de 0.69. Nos deux autres algorithmes obtiennent les résultats suivants avec des F-mesure atteignant 0.68 et 0.67.

8.6 Conclusion

Dans ce chapitre, on a essayé d'associer un détecteur d'*onsets* qui ne détecte que les *onsets* causés par les trois principaux instruments de batterie à l'algorithme de transcription de batterie. Bien que cela permet à l'algorithme de transcription de mieux analyser les segments qui lui sont fournis et donc d'améliorer la précision de sa transcription, certains événements dont ces instruments sont responsables sont manqués d'où un score de rappel bas. Les résultats n'atteignent donc pas des résultats équivalents aux résultats précédents avec le détecteur d'*onsets* généraux.

Les résultats de détection d'*onsets* spécifiques obtenus au chapitre précédent ont tout

Algorithmes	Transcription	BD	SD	HH
JAR1	**0.69**	0.78	0.65	0.53
JAR3	**0.69**	0.79	0.65	0.54
JAR2	0.68	0.78	0.65	0.53
JAR5	0.67	0.77	0.63	0.53
RV1	**0.69**	0.78	0.69	0.53
RV2	0.65	0.75	0.65	0.50
CS2	0.63	0.78	0.59	0.50
CS4	0.62	0.77	0.58	0.47
JS2	0.62	0.78	0.63	0.40
JS1	0.61	0.75	0.59	0.43
JS3	0.61	0.76	0.63	0.36
JS4	0.57	0.73	0.61	0.32

Tableau 8.10 – Résultats au challenge MIREX 2018 de transcription automatique de la batterie.

de même motivé un approfondissement de l'utilisation d'un détecteur d'*onsets* spécifiques. Des réseaux de même architecture ont été entraînés à détecter des *onsets* encore plus spécifiques. Trois réseaux indépendant détectent chacun un instrument. Pour la caisse-claire et la Charleston, les résultats sont meilleurs que pour la combinaison du détecteur d'*onsets* spécifiques mais ces résultats restent derrière l'état de l'art pour la Charleston.

La généralisation semble ne pas s'effectuer correctement. L'ajout de données par augmentation des données par remixage des parties bruit et sinusoïdales ainsi que des attaques donne un petit avantage à la généralisation.

Enfin, plusieurs modèles ont été présentés au challenge MIREX 2018 de transcription automatique de la batterie. Deux modèles multi-sorties avec des hyper-paramètres différents et une combinaison des sorties d'un des modèles multi-sorties avec les sorties des réseaux indépendants. Deux des quatre modèles ont atteint les meilleurs résultats avec un autre modèle basé aussi sur les CNN. Les deux autres modèles obtiennent les résultats suivant.

Quatrième partie

Conclusions

— Chapitre 9 —

Conclusions et perspectives

Bilan de la thèse

Cette thèse s'est intéressé à deux problématiques principales : l'adaptation des méthodes de décomposition non-négative afin de prendre en compte les informations issues des morceaux analysés ainsi que la spécification des *onsets* à détecter pour améliorer la transcription de la batterie voir de directement la réaliser.

Plusieurs méthodes de décomposition non-négative présentées au Chapitre 3 sont comparées au Chapitre 5 afin de sélectionner la méthode qui est utilisée dans la suite de nos recherches. Nous avons proposé dans le Chapitre 6 différentes méthodes pour intégrer les informations issues du morceau analysé et ainsi assurer une meilleure robustesse de l'algorithme : une basée sur l'application de filtres aux motifs du dictionnaire permettant de prendre en compte les différentes conditions d'enregistrement et une qui adapte directement les motifs avec les informations contenues dans le morceau. Pour cette deuxième méthode, nous avons proposé et comparé différentes contraintes permettant aux motifs de garder leur signification physique nécessaire au post-traitement pour la transcription des différents instruments.

Les algorithmes de décomposition non-négative sont informés par les positions des *onsets* présents dans le morceau analysé. Au Chapitre 7, nous avons proposé un détecteur d'*onsets* basé sur un réseau de neurones convolutifs qui ne détecte que les *onsets* percussifs et nous avons fourni ces résultats à l'algorithme de décomposition non-négative NMD. Enfin, au Chapitre 8, nous avons spécialisé le détecteur pour qu'il détecte uniquement les *onsets* d'un seul instrument permettant de réaliser la transcription de chaque instrument de la batterie et nous avons développé un réseau de neurones convolutifs multi-sorties pour transcrire la batterie avec un seul réseau.

Comparaison des méthodes de décomposition non-négative. L'algorithme de transcription de la batterie sur lequel nos études se basent est présenté au Chapitre 5 et utilise originellement la *Non negative Matrix Deconvolution*, NMD. Théoriquement, le contexte probabiliste sur lequel les méthodes de décomposition non-négative (la *Shift-Invariant Probabilistic Latent Componant Analysis*, SI-PLCA, et NMD statistique avec

la divergence d'IS, IS-NMD/EM) s'appuient, permet d'introduire des informations a posteriori issues du signal analysé. Cependant, les performances de la SI-PLCA n'ont pas atteint les résultats obtenus avec la NMD et les temps de calcul très longs de la IS-NMD/EM n'ont pas permis d'optimiser cette méthode de décomposition. Nous avons donc retenu la NMD pour la suite de nos recherches.

Adaptation des méthodes de décomposition non-négative. La première stratégie pour prendre en compte les différences pouvant apparaître entre les motifs préalablement appris et les instruments présents dans les morceaux étudiés est d'appliquer un filtre modifiable sur les motifs d'un instrument au cours de la décomposition. Ce filtre permet de modéliser les différences de conditions d'enregistrement telles que le modèle de l'instrument, la réponse de la salle ou encore la réponse des microphones. Le but de la NMD dans cet algorithme est d'apprendre les filtres et activations permettant de minimiser une fonction de coût entre le spectrogramme cible et le spectrogramme reconstruit à partir des paramètres appris. Les expériences réalisées sur des exemples simples avec différentes modélisations des filtres n'ont pas permis d'obtenir la convergence de la fonction de coût.

La seconde stratégie est de modifier directement les motifs du dictionnaire avec les informations du morceau. La problématique est alors de contraindre ces modifications pour que le motif garde sa signification physique au fur et à mesure des itérations. Malheureusement aucune des mises à jour contraintes n'a permis d'améliorer les résultats pour tous les instruments étudiés de la batterie. Il semble que certaines informations non relatives à l'instrument cible soient introduites dans les motifs malgré les contraintes appliquées.

Spécialisation d'un détecteur d'*onsets*. Une des problématiques de la thèse était de savoir si un détecteur d'*onsets* pouvait être spécialisé pour détecter des *onsets* spécifiques, ici des *onsets* percussifs. L'utilisation de réseaux de neurones convolutifs pour la détection d'*onsets* a déjà fait ses preuves. Plusieurs configurations ont été étudiées dans cette thèse pour obtenir un réseau de neurones convolutif qui, une fois entraîné, ne détecte que les *onsets* percussifs (*onsets* réalisés par un des trois principaux instruments de la batterie : grosse-caisse, caisse-claire ou Charleston). Les F-mesures obtenues dépassent les 90%. Bien que la combinaison du détecteur avec l'algorithme de décomposition non-négative n'a pas amélioré les performances, les excellents résultats de détection d'*onsets* percussifs nous ont invités à considérer une spécialisation encore plus précise du détecteur d'*onsets*.

Détecteur d'*onsets* spécifiques pour la transcription automatique de la batterie. Trois réseaux de neurones convolutifs ont été entraînés indépendamment. Chaque réseau avait pour objectif de détecter les *onsets* d'un unique instrument. En combinant les résultats de ces trois réseaux, nous obtenons une transcription obtenant un des meilleurs résultats au challenge MIREX 2018. L'indépendance des réseaux leur permet de se spécialiser complètement à un instrument. Cependant, il y a alors un risque de sur-apprentissage et donc d'une moins bonne généralisation des réseaux. Une grande

quantité d'exemples d'apprentissage est nécessaire. Plusieurs transformations des données (remixage des attaques, remixage du bruit, transposition des enveloppes spectrales avec et sans compensation temporelle) ont été étudiées pour permettre de générer de nouveaux exemples (*data augmentation*). L'augmentation de données permet d'obtenir de l'invariance par rapport à certaines transformations permettant au réseau une meilleure généralisation.

Réseau de neurones multi-sorties. En plus des trois réseaux indépendants, nous avons développé un réseau de neurones convolutif avec trois sorties, une pour chaque instrument. Nous avons étudié deux manières de l'entraîner. Ces deux réseaux multi-sorties, l'ensemble des trois réseaux indépendants et la combinaison d'un réseau multi-sorties avec les trois réseaux indépendants ont été proposés au challenge MIREX 2018. Un des réseaux multi-sorties et la combinaison de sa sortie avec les sorties des trois réseaux indépendants ont obtenus les deux meilleurs résultats ex aequo avec un réseau convolutif proposé par R. Vogl.

Perspectives

Augmenter le nombre de classes d'instruments percussifs à transcrire. Dans cette thèse, nous nous sommes concentrés sur les trois principaux instruments de la batterie : la grosse-caisse, la caisse-claire et la Charleston. Ces trois instruments sont responsables de la majorité des rythmes dans la musique occidentale et les instruments supplémentaires peuvent être remplacés par un de ces trois instruments. Cependant, ces trois instruments ne peuvent pas recréer la diversité des sons d'une batterie et des motifs rythmiques. Bien que l'idée du motif rythmique puisse être donné par ces trois instruments, la richesse des sons ne pourra pas être rejouée.

L'ajout de différentes classes d'instruments telles que les toms ou les cymbales (*crash*, *ride*) est nécessaire au développement d'un algorithme de transcription automatique de la batterie afin d'obtenir une transcription plus complète. Cependant, le manque de données annotées reste un problème important pour l'introduction de classes supplémentaires.

Augmenter les données pour l'apprentissage. Le manque de données précisément annotées reste un obstacle. Il est important de présenter lors de l'étape d'apprentissage des exemples comportant une grande diversité permettant une meilleure généralisation du réseau de neurones aux données à analyser par la suite. La production de données annotées est un travail long, fastidieux et coûteux. La *data augmentation* permet par des transformations de créer des exemples supplémentaires à partir de données annotées. Le choix des transformations est primordial. Dans le domaine musical, il est important que les données après transformations soient toujours reconnaissables c'est-à-dire que grosse-caisse doit sonner comme une grosse-caisse même après transformation. Améliorer les transformations et étudier leurs conséquences sur la généralisation semblent pouvoir donner des avantages aux réseaux de neurones pour la transcription de la batterie. D'autres stratégies peuvent aussi être envisagées comme le paradigme *student-teacher*.

Transcription des instruments harmoniques. La transcription automatique de la batterie s'inscrit dans un objectif plus ambitieux : la transcription automatique de la musique. Pour réaliser un système complet de transcription de la musique, il faudrait combiner les résultats de la transcription de la batterie avec un système de transcription de la partie harmonique qui présente de nouveaux enjeux comme l'estimation des fréquences fondamentales des notes jouées et les durées de ces dernières. Enfin, pour obtenir un système complet, l'emploi d'autres algorithmes MIR est nécessaire pour déterminer la mesure et le tempo.

Annexes

— Annexe A —

Calculs PLCA et SIPLCA

La *Probabilistic Latent Componant Analysis* est une technique de décomposition en matrices non-négatives. À partir d'observations, on cherche à déduire une structure sous-jacente pour que la reconstruction de l'objet soit la plus proche de l'objet réel. Pour cela, la décomposition s'appuie sur des propriétés statistiques de l'objet à décomposer. Les calculs de (Smaragdis and Raj, 2007) sont détaillés ici et appliqués au signal audio.

A.1 PLCA

A.1.1 Le modèle

Les observations sont modélisées par la distribution $P(\mathbf{x})$ d'un vecteur $\mathbf{x}$ de N variables aléatoires ($\mathbf{x} = x_1, x_2, ..., x_N$). On cherche à déterminer les paramètres qui permettront d'approcher cette distribution. La distribution $P(\mathbf{x})$ peut s'écrire en introduisant une variable latente z avec la formule de Bayes :

$$\begin{aligned} P(\mathbf{x}) &= \sum_z P(z)P(\mathbf{x}|z) \\ &= \sum_z P(z)\prod_{j=1}^{N} P(x_j|z) \end{aligned} \tag{A.1}$$

L'ensemble des paramètres à déterminer est donc $\Lambda = \{P(z), P(x_j|z); \forall(z,j)\}$ afin que la divergence KL entre $P(\mathbf{x})$ et $P(\mathbf{x};\Lambda)$ soit minimale avec $P(\mathbf{x};\Lambda)$ la distribution du modèle estimée à partir des paramètres Λ.

Dans la suite, l'exposant (n) signifie que l'on considère l'objet à l'itération n.

A.1.2 Dans le cas général

Maintenant que le modèle est décrit et afin de déterminer $Q(\Lambda, \Lambda^{(n)})$, calculons $E_{z|\mathbf{x};\Lambda^{(n)}}[\log P(\mathbf{x}, z; \Lambda)]$:

$$\begin{aligned} E_{z|\mathbf{x};\Lambda^{(n)}}[\log P(\mathbf{x},z;\Lambda)] &= \sum_z P^{(n)}(z|\mathbf{x}) \log(P_\Lambda^{(n)}(\mathbf{x},z)) \\ &= \sum_z P^{(n)}(z|\mathbf{x}) \log(P(z)P(\mathbf{x}|z)) \\ &= \sum_z P^{(n)}(z|\mathbf{x}) \log P(z) \prod_j P(x_j|z) \\ &= \sum_z P^{(n)}(z|\mathbf{x})(\log P(z) + \sum_j P(x_j|z)) \end{aligned} \tag{A.2}$$

Posons

$$\begin{aligned} R(\mathbf{x},z) &= P^{(n)}(z|\mathbf{x}) \\ &= \frac{P^{(n)}(z)P^{(n)}(\mathbf{x}|z)}{P^{(n)}(\mathbf{x})} \\ &= \frac{P^{(n)}(z)\prod_j P^{(n)}(x_j|z)}{P^{(n)}(\mathbf{x})} \\ &= \frac{P^{(n)}(z)\prod_j P^{(n)}(x_j|z)}{\sum_{z'} P^{(n)}(z')P^{(n)}(\mathbf{x}|z')} \\ &= \frac{P^{(n)}(z)\prod_j P^{(n)}(x_j|z)}{\sum_{z'} P^{(n)}(z')\prod_j P(x_j|z)} \end{aligned} \tag{A.3}$$

Alors l'équation A.2 devient :

$$E_{z|\mathbf{x};\Lambda^{(n)}}[\log P(\mathbf{x},z;\Lambda)] = \sum_z R(\mathbf{x},z)(\log P(z) + \sum_j \log P(x_j|z)) \tag{A.4}$$

On ajoute les contraintes de sommation à 1 des densités de probabilité et on doit

donc maximiser :

$$\begin{aligned}
Q(\Lambda,\Lambda^{(n)}) &= E_{\mathbf{x}}[E_{z|\mathbf{x};\Lambda^{(n)}}[\log P(\mathbf{x},z;\Lambda)]] - \lambda\sum_z P(z) \\
&\quad - \sum_j\sum_z \lambda_{z,j}\sum_{x_j} P(x_j|z) \\
&= E_{\mathbf{x}}[\sum_z R(\mathbf{x},z)(\log P(z) + \sum_j \log P(x_j|z))] - \lambda\sum_z P(z) \\
&\quad - \sum_{j,z} \lambda_{z,j}\sum_{x_j} P(x_j|z) \\
&= E_{\mathbf{x}}[\sum_z R(\mathbf{x},z)\log P(z)] + E_{\mathbf{x}}[\sum_z R(\mathbf{x},z)\sum_j \log P(x_j|z)] \\
&\quad -\lambda\sum_z P(z) - \sum_{j,z}\lambda_{z,j}\sum_{x_j} P(x_j|z) \\
&= \sum_z (E_{\mathbf{x}}[R(\mathbf{x},z)]\log P(z) - \lambda P(z)) \\
&\quad + \sum_{z,j}(E_{\mathbf{x}}[R(\mathbf{x},z)\log P(x_j|z)] - \lambda_{z,j}\sum_{x_j} P(x_j|z)) \qquad \text{(A.5)}
\end{aligned}$$

Le problème s'écrit donc :

$$\begin{aligned}
\Lambda^{(n+1)} &= \arg\max_{\Lambda} Q(\Lambda,\Lambda^{(n)}) \qquad \text{(A.6)} \\
Q(\Lambda,\Lambda^{(n)}) &= \sum_z (E_{\mathbf{x}}[R(\mathbf{x},z)]\log P(z) - \lambda P(z)) \\
&\quad + \sum_{z,j}(E_{\mathbf{x}}[R(\mathbf{x},z)\log P(x_j|z)] - \lambda_{z,j}\sum_{x_j} P(x_j|z)) \qquad \text{(A.7)}
\end{aligned}$$

Sous forme continue :

$$\begin{aligned}
\Lambda^{(n+1)} &= \arg\max_{\Lambda} Q(\Lambda,\Lambda^{(n)}) \qquad \text{(A.8)} \\
Q(\Lambda,\Lambda^{(n)}) &= \int_z (E_{\mathbf{x}}[R(\mathbf{x},z)]\log P(z) - \lambda P(z))dz \\
&\quad + \int_z\int_j (E_{\mathbf{x}}[R(\mathbf{x},z)\log P(x_j|z)] - \lambda_{z,j}\int_{x_j} P(x_j|z)dx_j)djdz \qquad \text{(A.9)}
\end{aligned}$$

La prochaine étape est de déterminer les paramètres $\Lambda = \{P(z), P(x_j|z); \forall(z,j)\}$ à l'itération $n+1$ qui maximisent $Q(\Lambda,\Lambda^{(n)})$.

Optimiser $Q(\Lambda,\Lambda^{(n)})$ par rapport à $P(z)$ revient à déterminer $P(z)$ tel que

$$\frac{d}{dP(z)}(E_{\mathbf{x}}[R(\mathbf{x},z)]\log P(z) - \lambda P(z)) = 0$$

puisque

$$\int_z \int_j (E_{\mathbf{x}}[R(\mathbf{x},z)\log P(x_j|z)] - \lambda_{z,j}\int_{x_j} P(x_j|z)dx_j)djdz$$

ne dépend pas de $P(z)$. Le calcul donne :

$$\frac{d}{dP^{(n+1)}(z)}\Big(E_{\mathbf{x}}[R(\mathbf{x},z)]\log P^{(n+1)}(z) - \lambda P^{(n+1)}(z)\Big) = 0 \tag{A.10}$$

$$\frac{E_{\mathbf{x}}[R(\mathbf{x},z)]}{P^{(n+1)}(z)} - \lambda = 0 \tag{A.11}$$

$$\lambda P^{(n+1)}(z) = E_{\mathbf{x}}[R(\mathbf{x},z)] \tag{A.12}$$

Or $\sum_z P^{(n)}(z) = 1$ et $\sum_z E_{\mathbf{x}}[R(\mathbf{x},z)] = \sum_{\mathbf{x},z} P^{(n)}(z|\mathbf{x})P(\mathbf{x}) = 1$ donc $\lambda = 1$ et

$$P^{(n+1)}(z) = E_{\mathbf{x}}[R(\mathbf{x},z)]P^{(n+1)}(z) = \sum_{\mathbf{x}} P(\mathbf{x})R(\mathbf{x},z)$$

Faisons de même, pour $P(x_j|z)$. On peut écrire :

$$\begin{aligned} E_{\mathbf{x}}[R(\mathbf{x},z)\log P(x_j|z)] &= E_{x_j}[[E_{\mathbf{x}|x_j}[R(\mathbf{x},z)\log P(x_j|z)]] \\ &= E_{x_j}[[E_{\mathbf{x}|x_j}[R(\mathbf{x},z)]\log P(x_j|z)] \\ &= \sum_{x_j} P(x_j)E_{\mathbf{x}|x_j}[R(\mathbf{x},z)]\log P(x_j|z) \end{aligned} \tag{A.13}$$

Comme précédemment, l'optimisation de $Q(\Lambda,\Lambda^{(n)})$ par rapport à $P(x_j|z)$ donne :

$$\frac{d}{P(x_j|z)}\left(\sum_j (E_{\mathbf{x}}[R(\mathbf{x},z)\log P(x_j|z)] - \lambda_{z,j}\sum_{x_j} P(x_j|z))\right) = 0$$

$$\frac{d}{P(x_j|z)}\left(\sum_j \left[\sum_{x_j} P(x_j)E_{\mathbf{x}|x_j}[R(\mathbf{x},z)]\log P(x_j|z) - \lambda_{z,j}\sum_{x_j} P(x_j|z))\right]\right) = 0$$

$$\frac{d}{P(x_j|z)}\left(\sum_j\sum_{x_j}\Big[P(x_j)E_{\mathbf{x}|x_j}[R(\mathbf{x},z)]\log P(x_j|z) - \lambda_{z,j}P(x_j|z)\Big]\right) = 0$$

$$\frac{P^{(n+1)}(x_j)E_{\mathbf{x}|x_j}[R(\mathbf{x},z)]}{P^{(n+1)}(x_j|z)} - \lambda_{z,j} = 0 \tag{A.14}$$

Or $\sum_{x_j} P^{(n+1)}(x_j|z) = 1$ et

$$\sum_{x_j} P^{(n+1)}(x_j)E_{\mathbf{x}|x_j}[R(\mathbf{x},z)] = E_{\mathbf{x}}[R(\mathbf{x},z)] \tag{A.15}$$

$$= E_{\mathbf{x}}[P^{(n+1)}(z|\mathbf{x})] \tag{A.16}$$

$$= \sum_{\mathbf{x}} P^{(n+1)}(z\mathbf{x})P(\mathbf{x}) = P^{(n+1)}(z) \tag{A.17}$$

alors $\lambda_{z,j} = P^{(n+1)}(z)$. L'équation de mise à jour s'écrit alors

$$P^{(n+1)}(x_j|z) = \frac{P(x_j)E_{\mathbf{x}|x_j}[R(\mathbf{x},z)]}{P^{(n+1)}(z)} \tag{A.18}$$

En résumé, les équations de mise à jour sont :

$$\begin{aligned} R(\mathbf{x},z) &= \frac{P^{(n)}(z)\prod_j P^{(n)}(x_j|z)}{\sum_{z'} P^{(n)}(z')\prod_j P(x_j|z)} && \text{(A.19)}\\ P^{(n+1)}(z) &= E_{\mathbf{x}}[R(\mathbf{x},z)] && \text{(A.20)}\\ P^{(n+1)}(x_j|z) &= \frac{P(x_j)E_{\mathbf{x}|x_j}[R(\mathbf{x},z)]}{P^{(n+1)}(z)} && \text{(A.21)} \end{aligned}$$

A.1.3 Dans le cas d'un signal audio

Dans cette partie, on décrit un signal audio par son spectrogramme. Le signal est donc vu comme la distribution de deux variables aléatoires f et t. Ainsi $\mathbf{x} = (f,t)$. La fonction de vraisemblance s'écrit :

$$\begin{aligned} Q(\lambda,\lambda^{(n)}) = &\sum_z \sum_{f,t} P(f,t)R(f,t,z)\log P(z) - \lambda\sum_z P(z)\\ &+ \sum_z \sum_{f,t} P(f,t)R(f,t,z)\log P(f|z) - \sum_z \lambda_{z,f}\sum_f P(f|z)\\ &+ \sum_z \sum_{f,t} P(f,t)R(f,t,z)\log P(t|z) - \sum_z \lambda_{z,t}\sum_t P(t|z) \end{aligned} \tag{A.22}$$

Les équations de mise à jour sont alors données par :

$$\begin{aligned} R(f,t,z) &= \frac{P^{(n)}(z)P^{(n)}(f|z)P(t|z)}{\sum_{z'} P^{(n)}(z')P(f|z')P(t|z')} && \text{(A.23)}\\ P^{(n+1)}(z) &= \sum_{f,t} P(f,t)R(f,t,z) && \text{(A.24)}\\ P^{(n+1)}(f|z) &= \frac{P(f)E_{f,t|f}[R(f,t,z)]}{P^{(n+1)}(z)} && \text{(A.25)}\\ P^{(n+1)}(t|z) &= \frac{P(t)E_{f,t|t}[R(f,t,z)]}{P^{(n+1)}(z)} && \text{(A.26)} \end{aligned}$$

Or

$$\begin{aligned} P(f)E_{f,t|f}[R(f,t,z)] &= P(f)E_{t|f}[R(f,t,z)] && \text{(A.27)}\\ &= P(f)\sum_{t|f}[R(f,t,z)]P(t|f) && \text{(A.28)}\\ &= \sum_{t|f}[R(f,t,z)]P(f,t) && \text{(A.29)} \end{aligned}$$

et

$$\begin{aligned} P^{(n+1)}(z) &= \sum_{f,t} P^{(n+1)}(z|f,t)P(f,t) \\ &= \sum_{f,t} R(f,t,z)P(f,t) \end{aligned} \tag{A.30}$$

alors

$$R(f,t,z) \leftarrow \frac{P^{(n)}(z)P^{(n)}(f|z)P(t|z)}{\sum_{z'} P^{(n)}(z')P(f|z')P(t|z')} \tag{A.31}$$

$$P^{(n+1)}(z) \leftarrow \sum_{f,t} P(f,t)R(f,t,z) \tag{A.32}$$

$$P^{(n+1)}(f|z) \leftarrow \frac{\sum_t R(f,t,z)P(f,t)}{P^{(n+1)}(z)} \tag{A.33}$$

$$P^{(n+1)}(t|z) \leftarrow \frac{\sum_f R(f,t,z)P(f,t)}{P^{(n+1)}(z)} \tag{A.34}$$

On peut écrire ces équations sous la forme de mises à jour multiplicatives.

$$R(f,t,z) \leftarrow \frac{P^{(n)}(z)P^{(n)}(f|z)P(t|z)}{\sum_{z'} P^{(n)}(z')P(f|z')P(t|z')} \tag{A.35}$$

$$P^{(n+1)}(z) \leftarrow P^{(n)}(z) \sum_{f,t} \frac{P(f,t)P^{(n)}(f|z)P^{(n)}(t|z)}{\sum_{z'} P^{(n)}(z')P^{(n)}(f|z')P^{(n)}(t|z')} \tag{A.36}$$

$$P^{(n+1)}(f|z) \leftarrow P^{(n)}(f|z) \frac{\sum_t P(f,t)P^{(n)}(z)P^{(n)}(t|z)}{\sum_{f,t} P(f,t)R(f,t,z)} \tag{A.37}$$

$$P^{(n+1)}(t|z) \leftarrow P^{(n)}(t|z) \frac{\sum_f P(f,t)P^{(n)}(z)P^{(n)}(f|z)}{\sum_{f,t} P(f,t)R(f,t,z)} \tag{A.38}$$

Dimension des matrices utilisées pour modéliser le problème :

$R^{(n)}$	matrice de taille $F * N * K$
$W^{(n)} = [P^{(n)}(f\|z)]$	matrice de taille $F * K$ à l'itération n
$H^{(n)} = [P^{(n)}(t\|z)]$	matrice de taille $K * N$ à l'itération n
$Q^{(n)} = [P^{(n)}(z)]$	matrice de taille $K * K$ à l'itération n

A.2 SI-PLCA

A.2.1 Le modèle

La SI-PLCA modélise certaines distributions comme la convolution d'une distribution noyau avec une distribution qui présente une shift-invariance appelée impulsion. Comme

dans le cas précédent, nous appellerons $\mathbf{x}$ le vecteur de variables aléatoires. On notera $\mathbf{w}$ les noyaux et $\mathbf{y}$ les impulsions. Le modèle s'écrit alors :

$$P(\mathbf{x};\Lambda) = \sum_z P(z) \sum_\tau P(\mathbf{w},\tau|z)P(\mathbf{y}-\tau|z) \tag{A.39}$$

Posons :

$$\begin{aligned} R(\mathbf{x},\tau,z) &= P(\tau,z|\mathbf{x}) \\ &= \frac{P(\mathbf{x},\tau|z)P(z)}{P(\mathbf{x})} \\ &= \frac{P(\mathbf{w},\tau|z)P(\mathbf{y}-\tau|z)P(z)}{\sum_z P(\mathbf{x}|z)P(z)} \\ &= \frac{P(z)P(\mathbf{w},\tau|z)P(\mathbf{y}-\tau|z)}{\sum_z P(z)\sum_\tau P(\mathbf{w},\tau|z)P(\mathbf{y}-\tau|z)} \end{aligned} \tag{A.40}$$

A.2.2 Dans le cas général

Pour déterminer les équations de mise à jour, il faut tout d'abord exprimer $Q(\Lambda,\Lambda^{(n)})$:

$$\begin{aligned} Q(\Lambda,\Lambda^{(n)}) &= E_{\mathbf{x}}[E_{z,\tau|\mathbf{x};\Lambda}[\log P(\mathbf{x},z,\tau;\Lambda)]] \\ &= \sum_{\mathbf{x}} P(\mathbf{x}) \sum_{z,\tau} P(z,\tau|\mathbf{x}) \log P(\mathbf{x},z,\tau;\Lambda) \\ &= \sum_{z,\tau}\sum_{\mathbf{x}} P(\mathbf{x})P(z,\tau|\mathbf{x}) \log P(\mathbf{x},\tau|z)P(z) \\ &= \sum_{z,\tau}\sum_{\mathbf{x}} P(\mathbf{x})P(z,\tau|\mathbf{x}) \log P(z)P(\mathbf{w},\tau|z)P(\mathbf{y}-\tau|z) \\ &= \sum_{z,\tau}\sum_{\mathbf{w},\mathbf{y}} P(\mathbf{w},\mathbf{y})R(\mathbf{w},\mathbf{y},z,\tau)[\log P(z) + \log P(\mathbf{w},\tau|z) + \log P(\mathbf{y}-\tau|z)] \end{aligned} \tag{A.41}$$

On ajoute maintenant les termes de contrainte de sommation à 1 des probabilités marginales et on obtient la fonction Q à optimiser :

$$\begin{aligned} Q(\Lambda,\Lambda^{(n)}) = &\sum_{z,\tau}\sum_{\mathbf{w},\mathbf{y}} P(\mathbf{w},\mathbf{y})R(\mathbf{w},\mathbf{y},z,\tau)[\log P(z) + \log P(\mathbf{w},\tau|z) + \log P(\mathbf{y}-\tau|z)] \\ &- \lambda\sum_z P(z) - \sum_z \lambda_{\tau,z}\sum P(\mathbf{w},\tau|z) - \sum_z \lambda_{y,z}\sum_{\mathbf{y}} P(\mathbf{y}|z) \end{aligned} \tag{A.42}$$

Dans un premier temps, on cherche $P^{(n+1)}(z)$ qui optimise $Q(\Lambda,\Lambda^{(n)})$ à l'itération

n+1. On a :

$$\begin{aligned} Q(\Lambda, \Lambda^{(n)}) &= \sum_{z,\tau} \sum_{\mathbf{w},\mathbf{y}} P(\mathbf{w},\mathbf{y}) R(\mathbf{w},\mathbf{y},z,\tau)[\log P(z)] - \lambda \sum_{z} P(z) + C \\ &= \sum_{z,\tau} E_{\mathbf{w},\mathbf{y}}[R(\mathbf{w},\mathbf{y},z,\tau)] \log P(z) - \lambda \sum_{z} P(z) + C \\ &= \sum_{z} [\sum_{\tau} E_{\mathbf{w},\mathbf{y}}[R(\mathbf{w},\mathbf{y},z,\tau)] - P(z)] + C \end{aligned} \tag{A.43}$$

où C est la partie de $Q(\Lambda, \Lambda^{(n)})$ qui ne dépend pas de $P(z)$.

On cherche donc $P^{(n+1)}(z)$ qui optimise $Q(\Lambda, \Lambda^{(n)})$ à l'itération n+1 :

$$\frac{\sum_{\tau} E_{\mathbf{w},\mathbf{y}}[R(\mathbf{w},\mathbf{y},z,\tau)]}{P(z)} - \lambda = 0 \tag{A.44}$$

$$\sum_{\tau} E_{\mathbf{w},\mathbf{y}}[R(\mathbf{w},\mathbf{y},z,\tau)] = \lambda P(z) \tag{A.45}$$

En sommant sur z, l'équation A.45 devient :

$$\begin{aligned} \lambda \sum_{z} P(z) &= \sum_{z} \sum_{\tau} E_{\mathbf{w},\mathbf{y}}[R(\mathbf{w},\mathbf{y},z,\tau)] \\ \lambda &= \sum_{z,\tau,\mathbf{w},\mathbf{y}} R(\mathbf{w},\mathbf{y},z,\tau) P(\mathbf{w},\mathbf{y}) \\ \lambda &= \sum_{z,\tau,\mathbf{w},\mathbf{y}} P(z,\tau|\mathbf{w},\mathbf{y}) P(\mathbf{w},\mathbf{y}) \\ \lambda &= \sum_{z,\tau,\mathbf{w},\mathbf{y}} P(z,\tau,\mathbf{w},\mathbf{y}) \\ \lambda &= 1 \end{aligned} \tag{A.46}$$

Enfin, on obient la première équation de mise à jour

$$P^{(n+1)}(z) = \sum_{\tau} E_{\mathbf{w},\mathbf{y}}[R^{(n)}(\mathbf{w},\mathbf{y},z,\tau)] \tag{A.47}$$

Remarque :

$$\begin{aligned} E_{w,y}[h(P(x)] &= \sum_{w,y} P(w,y) h(P(x)) = \sum_{w,y} P(w|y) P(y) h(P(x)) \\ &= \sum_{y} P(y) \sum_{w|y} P(w|y) h(P(x)) = \sum_{y} P(y) E_{w|y}[h(P(x))] \\ &= E_{y}[E_{w|y}[h(P(x))]] \end{aligned} \tag{A.48}$$

On veut maintenant optimiser $Q(\Lambda, \Lambda^{(n)})$ par rapport à $P(\mathbf{y}|z)$. Pour cela, on peut écrire $Q(\Lambda, \Lambda^{(n)})$ sous la forme :

$$\begin{aligned}
Q(\Lambda, \Lambda^{(n)}) &= \sum_{z,\tau} E_{\mathbf{w},\mathbf{y}}[R(\mathbf{w},\mathbf{y},z,\tau)\log P(\mathbf{y}-\tau|z)] - \sum_z \lambda_{y,z}P(\mathbf{y}|z) + C \\
&= \sum_{z,\tau} E_{\mathbf{y}}[E_{\mathbf{w}|\mathbf{y}}[R(\mathbf{w},\mathbf{y},z,\tau)\log P(\mathbf{y}-\tau|z)]] - \sum_z \lambda_{y,z}P(\mathbf{y}|z) + C \\
&= \sum_{z,\tau} E_{\mathbf{y}}[\log P(\mathbf{y}-\tau|z)E_{\mathbf{w}|\mathbf{y}}[R(\mathbf{w},\mathbf{y},z,\tau)]] - \sum_z \lambda_{y,z}P(\mathbf{y}|z) + C \\
&= \sum_{z,\tau,\mathbf{y}} P(\mathbf{y})\log P(\mathbf{y}-\tau|z)E_{\mathbf{w}|\mathbf{y}}[R(\mathbf{w},\mathbf{y},z,\tau)] - \sum_z \lambda_{y,z}P(\mathbf{y}|z) + C \\
&= \sum_{z,\tau,\mathbf{y}} P(\mathbf{y}+\tau)\log P(\mathbf{y}|z)E_{\mathbf{w}|\mathbf{y}}[R(\mathbf{w},\mathbf{y}+\tau,z,\tau)] - \sum_z \lambda_{y,z}P(\mathbf{y}|z) + C \\
&= \sum_z \sum_{\mathbf{y}} [\log P(\mathbf{y}|z) \sum_\tau P(\mathbf{y}+\tau)E_{\mathbf{w}|\mathbf{y}}[R(\mathbf{w},\mathbf{y}+\tau,z,\tau)] - \lambda_{y,z}P(\mathbf{y}|z)] + C
\end{aligned} \tag{A.49}$$

On cherche $P^{(n+1)}(\mathbf{y}|z)$ tel que $Q(\Lambda, \Lambda^{(n)})$ soit optimisé, c'est-à-dire qui vérifie :

$$\begin{aligned}
\frac{\sum_\tau P(\mathbf{y}+\tau)E_{\mathbf{w}|\mathbf{y}}[R(\mathbf{w},\mathbf{y}+\tau,z,\tau)]}{P^{(n+1)}(\mathbf{y}|z)} - \lambda_{y,z} &= 0 \\
\lambda_{y,z}P^{(n+1)}(\mathbf{y}|z) &= P(\mathbf{y}+\tau)E_{\mathbf{w}|\mathbf{y}}[R(\mathbf{w},\mathbf{y}+\tau,z,\tau)]
\end{aligned} \tag{A.50}$$

Si on somme sur $\mathbf{y}$ des deux côtés, on obtient :

$$\begin{aligned}
\lambda_{y,z} &= \sum_{\mathbf{y}} \sum_\tau P^{(n)}(\mathbf{y}+\tau) \sum_{\mathbf{w}} P^{(n)}(\mathbf{w}|\mathbf{y})R^{(n)}(\mathbf{w},\mathbf{y}+\tau,\tau,z) \\
&= \sum_{\mathbf{y},\tau,\mathbf{w}} P^{(n)}(\mathbf{w},\mathbf{y}+\tau)R^{(n)}(\mathbf{w},\mathbf{y}+\tau,\tau,z)
\end{aligned} \tag{A.51}$$

Alors la deuxième équation de mise à jour est :

$$\begin{aligned}
P^{(n+1)}(\mathbf{y}|z) &= \frac{\sum_{\tau,\mathbf{w}} P^{(n)}(\mathbf{y}+\tau)P(\mathbf{w}|\mathbf{y})R^{(n)}(\mathbf{w},\mathbf{y}+\tau,\tau,z)}{\sum_{\mathbf{y},\tau,\mathbf{w}} P^{(n)}(\mathbf{w},\mathbf{y}+\tau)R^{(n)}(\mathbf{w},\mathbf{y}+\tau,\tau,z)} \\
&= \frac{\sum_{\tau,\mathbf{w}} P^{(n)}(\mathbf{w},\mathbf{y}+\tau)R^{(n)}(\mathbf{w},\mathbf{y}+\tau,\tau,z)}{\sum_{\mathbf{y},\tau,\mathbf{w}} P^{(n)}(\mathbf{w},\mathbf{y}+\tau)R^{(n)}(\mathbf{w},\mathbf{y}+\tau,\tau,z)}
\end{aligned} \tag{A.52}$$

De la même manière, on obtient la troisième équation de mise à jour :

$$P^{(n+1)}(\mathbf{w},\tau|z) = \frac{\sum_{\mathbf{y}} P(\mathbf{w},\mathbf{y})R(\mathbf{w},\mathbf{y},\tau,z)}{P^{(n+1)}(z)} \tag{A.53}$$

Pour résumer, on obtient les quatre équations de mise à jour suivantes :

$$R(\mathbf{x},\tau,z) = \frac{P(z)P(\mathbf{w},\tau|z)P(\mathbf{y}-\tau|z)}{\sum_z P(z)\sum_\tau P(\mathbf{w},\tau|z)P(\mathbf{y}-\tau|z)} \tag{A.54}$$

$$P^{(n+1)}(z) = \sum_\tau E_{\mathbf{w},\mathbf{y}}[R^{(n)}(\mathbf{w},\mathbf{y},z,\tau)] \tag{A.55}$$

$$P^{(n+1)}(\mathbf{y}|z) = \frac{\sum_{\tau,\mathbf{w}} P^{(n)}(\mathbf{w},\mathbf{y}+\tau)R^{(n)}(\mathbf{w},\mathbf{y}+\tau,\tau,z)}{\sum_{\mathbf{y},\tau,\mathbf{w}} P^{(n)}(\mathbf{w},\mathbf{y}+\tau)R^{(n)}(\mathbf{w},\mathbf{y}+\tau,\tau,z)} \tag{A.56}$$

$$P^{(n+1)}(\mathbf{w},\tau|z) = \frac{\sum_{\mathbf{y}} P(\mathbf{w},\mathbf{y})R^{(n)}(\mathbf{w},\mathbf{y},\tau,z)}{P^{(n+1)}(z)} \tag{A.57}$$

A.2.3 Dans le cas d'un signal audio

Dans le cas d'un signal audio $\mathbf{x} = [w, y] = [f, t]$, les équations de mise à jour deviennent alors :

$$R(f,t,\tau,z) = \frac{P(z)P(f,\tau|z)P(t-\tau|z)}{\sum_{z'} P(z)\sum_{\tau'} P(f,\tau|z')P(t-\tau'|z')} \tag{A.58}$$

$$P^{(n+1)}(z) = \sum_\tau E_{f,t}[R^{(n)}(f,t,z,\tau)] \tag{A.59}$$

$$P^{(n+1)}(t|z) = \frac{\sum_{\tau,f} P^{(n)}(f,t+\tau)R^{(n)}(f,t+\tau,\tau,z)}{\sum_{t,\tau,f} P^{(n)}(f,t+\tau)R^{(n)}(f,t+\tau,\tau,z)} \tag{A.60}$$

$$P^{(n+1)}(f,\tau|z) = \frac{\sum_t P(f,t)R(f,t,\tau,z)}{P^{(n+1)}(z)} \tag{A.61}$$

ou bien :

$$R(f,t,\tau,z) = \frac{P(z)P(f,\tau|z)P(t-\tau|z)}{\sum_{z'} P(z)\sum_{\tau'} P(f,\tau|z')P(t-\tau'|z')} \tag{A.62}$$

$$P^{(n+1)}(z) = P(z)^{(n)} \sum_{\tau,f,t} \frac{P(f,\tau|z)P(t-\tau|z)}{\sum_{z'} P(z')\sum_{\tau'} P(f,\tau'|z')P(t-\tau'|z')} \tag{A.63}$$

$$P^{(n+1)}(t|z) = P^{(n)}(t|z)\frac{\sum_{\tau,f} P^{(n)}(z)P^{(n)}(f,\tau|z)P^{(n)}(f,t+\tau)}{P^{(n+1)}(z)\sum_{z'} P(z')\sum_{\tau'} P(f,\tau'|z')P(t-\tau'|z')} \tag{A.64}$$

$$P^{(n+1)}(f,\tau|z) = P^{(n)}(f,\tau|z)\frac{\sum_t P(f,t)P^{(n)}(z)P^{(n)}(t-\tau|z)}{P^{(n+1)}(z)\sum_{z'} P(z')\sum_{\tau'} P(f,\tau'|z')P(t-\tau'|z')} \tag{A.65}$$

A.3 Introduction d'une contrainte de parcimonie

Ce chapitre présente comment la contrainte de parcimonie peut être intégrée à la SI-PLCA d'après l'article de M. Hoffman (Hoffman, 2010).

On veut estimer l'ensemble des paramètres $\boldsymbol{\theta}$ comme une distribution multinomiale qui génère N observations $x_i \in \{1,\ldots,K\}$. La fonction de log-vraisemblance des données

est alors :

$$\log p(\mathbf{x}) = \sum_i \log \theta_i \tag{A.66}$$

Pour un problème sans contrainte, on chercherait les paramètres θ_i qui maximisent cette fonction. Ici nous voulons inclure la connaissance a priori sur $\boldsymbol{\theta}$ suivante : $\boldsymbol{\theta}$ est parcimonieux.
Pour inclure cette connaissance, on doit ajouter un terme de pénalité à la fonction de log-vraisemblance qui permet de traduire la parcimonie de $\boldsymbol{\theta}$.

Dans les problèmes classiques d'optimisation, la parcimonie est introduite par un terme de pénalité basé sur la norme L1. Mais dans notre problème c'est difficilement envisageable puisque la norme L1 des paramètres $\boldsymbol{\theta}$ est égale à 1 par définition. Une alternative est d'inclure un terme négatif d'entropie dans la fonction de log-vraisemblance.

$$\log p(\mathbf{x}, \boldsymbol{\theta}) = constant + a \sum_k \theta_k \log \theta_k + \sum_i log\theta_{x_i} \tag{A.67}$$

La constante a contrôle la puissance de la distribution a priori $p(\boldsymbol{\theta}) \propto \exp a \sum_k \theta_k \log \theta_k$.

L'estimateur au sens du Maximum a posteriori (MAP) des paramètres $\boldsymbol{\theta}$ n'a pas de solution analytique simple pour ce modèle. Matthew Hoffman a donc proposé un algorithme itératif pour calculer l'estimateur au sens du MAP de $\boldsymbol{\theta}$ quand a est positif.

A.3.1 Algorithme

On considère une fonction auxiliaire permettant d'approximer le terme négatif d'entropie de l'équation A.3 :

$$\ell(a, \nu, \boldsymbol{\theta}, \boldsymbol{\alpha}) \triangleq a \sum_k \alpha_k (\nu \log \theta_k - (\nu - 1) \log \alpha_k) \tag{A.68}$$

ou $\boldsymbol{\alpha}$ est un paramètre libre tel que $\sum_k \alpha_k = 1$ et $\alpha_k \geq 0$ et ν est une valeur contrainte, supérieure à 1. Regardons le Lagrangien par rapport à α_k :

$$\frac{\partial \ell}{\partial \alpha_k} = a\nu \log \theta_k - a(\nu - 1)(1 + \log \alpha_k) + \lambda \tag{A.69}$$

Pour trouver les paramètres qui maximisent le Lagrangien, on égalise la partie de droite à zéro et on obtient :

$$\alpha_k \propto \exp\left(\frac{\nu}{\nu - 1} \log \theta_k\right) = \theta_k^{\frac{\nu}{\nu - 1}} \tag{A.70}$$

Quand ν est grand, ℓ est optimal seulement si $\alpha_k \approx \theta_k$. Ainsi on retrouve la forme originale de la distribution a priori de l'entropie :

$$\ell(a, \nu, \boldsymbol{\theta}, \boldsymbol{\theta}) = a \sum_k \theta_k (\nu \log \theta_k - (\nu - 1) \log \theta_k) = a \sum_k \theta_k \log \theta_k \tag{A.71}$$

Pour une valeur de ν assez grande et si $\boldsymbol{\alpha}$ est correctement choisi, ℓ est une approximation de l'*entropic prior*. On peut alors remplacer par ℓ la distribution a priori de l'équation A.3 et enfin trouver les paramètres $\boldsymbol{\theta}$ et $\boldsymbol{\alpha}$ qui maximisent la log-vraisemblance :

$$\mathcal{L} \triangleq a \sum_k \alpha_k (\nu \log \theta_k - (\nu - 1) \log \alpha_k) + \sum_i \log \theta_{x_i} \tag{A.72}$$

En optimisant par rapport à $\boldsymbol{\alpha}$ on obtient $\alpha_k \approx \theta_k$ et $\mathcal{L} \approx p(\mathbf{x}, \boldsymbol{\theta})$. On peut utiliser un simple point fixe pour optimiser $\mathcal{L}$ selon $\boldsymbol{\theta}$ et $\boldsymbol{\alpha}$. Le gradient du Lagrangien par rapport à $\boldsymbol{\theta}$ égalisé à 0 s'écrit :

$$\frac{\partial \mathcal{L}}{\partial \theta_k} = \frac{1}{\theta_k} \left(a \alpha_k \nu + \sum_i \mathbb{I}[x_i = k] \right) + \lambda \tag{A.73}$$

avec $\mathbb{I}$ une fonction indicatrice égale à 1 si l'argument est vrai, 0 sinon. Alors $\mathcal{L}$ est maximal par rapport à $\boldsymbol{\theta}$ when

$$\theta_k \propto a \alpha_k \nu + \sum_i \mathbb{I}[x_i = k] \tag{A.74}$$

Et $\mathcal{L}$ est maximal par rapport à $\boldsymbol{\alpha}$ quand

$$\alpha_k \propto \theta_k^{\frac{\nu}{\nu - 1}} \tag{A.75}$$

En procédant par itération, on atteint un point stationnaire de $\mathcal{L}$. A ce point, $\mathcal{L} \approx \log p(\mathbf{x}, \boldsymbol{\theta})$ donc on peut conclure que la valeur de $\boldsymbol{\theta}$ permet à $\mathcal{L}$ d'atteindre un optimum local de $\log p(\mathbf{x}, \boldsymbol{\theta})$.

— Annexe B —

Adaptation de la IS-NMF/EM à la déconvolution

B.1 Modélisation gaussienne dans le cas de la NMD

Dans le cas de la NMD, le modèle génératif est le suivant :

$$\mathbf{x}_n = \sum_{k=1}^{K} \mathbf{c}_{kn} \tag{B.1}$$

tels que $\mathbf{c}_{kn} \in \mathbb{C}^{Fx1}$ suivent une loi gaussienne :

$$\mathbf{c}_{kn} \sim \mathcal{N}_C(0, \sum_{k,t} W_{fk}^t h_{kn}^{t\rightarrow}) \tag{B.2}$$

La fonction de coût s'écrit alors :

$$\begin{aligned} C_{ML}(\mathbf{W}, \mathbf{H}) &= -\sum_{n=1}^{N}\sum_{f=1}^{F} \log \mathcal{N}_C(x_{fn}|0, \sum_{k,t} W_{fk}^t h_{kn}^{t\rightarrow}) && \text{(B.3)} \\ &= NF \log \pi + \sum_{n=1}^{N}\sum_{f=1}^{F} \log(\sum_{k,t} W_{fk}^t h_{kn}^{t\rightarrow}) + \frac{|x_{fn}|^2}{\sum_{k,t} W_{fk}^t h_{kn}^{t\rightarrow}} && \text{(B.4)} \\ &= \sum_{n=1}^{N}\sum_{f=1}^{F} d_{IS}(|x_{fn}|^2 | \sum_{k,t} W_{fk}^t h_{kn}^{t\rightarrow}) + const && \text{(B.5)} \end{aligned}$$

où *const* est une constante.

Comme Nancy Bertin le remarque à propos de l'équivalence MV/NMF (Bertin, 2009), ici on peut remarquer que déterminer les variables $\mathbf{W}$ et $\mathbf{H}$ par un estimateur du maximum de vraisemblance revient à résoudre le problème de la NMD utilisant la divergence d'Itakura-Saito.

B.2 Equations de mise à jour

B.2.1 L'algorithme SAGE

Dans notre modèle, le spectrogramme est vu comme la somme de plusieurs composantes gaussiennes.
Pour résoudre les problèmes de maximisation de la log-vraisemblance, on utilise usuellement des algorithmes Expectation-Maximization (EM). Le principe de ces derniers est l'alternance d'une étape d'Estimation pendant laquelle on calcule la fonction de log-vraisemblance et d'une étape de Maximisation qui consiste à déterminer les paramètres recherchés pour maximiser la log-vraisemblance.
Ici, la structure additive permet d'utiliser une extension des algorithmes EM, l'algorithme SAGE (Space Alternating Generalized EM) (Fessler and Hero, 1994). SAGE est conçu pour des données ayant des structures particulières. Dans ce modèle, la structure additive des données gaussiennes permet des mises à jour séparées des paramètres.

B.2.2 Etape E

On note $\boldsymbol{\theta}_k = (W_k, h_k)$ une partition de l'ensemble des paramètres $\boldsymbol{\theta}$ avec W_k la matrice du k-ième pattern et h_k la k-ième ligne de $\mathbf{H}$ représentant les activations de la k-ième base.

L'algorithme consiste à maximiser la log-vraisemblance. On écrit alors la fonction de coût, égale à l'opposé de l'espérance conditionnelle de la log-vraisemblance des variables latentes $\mathbf{C}_k$:

$$Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta}') = -\int_{\mathbf{C}_k} \log p(\mathbf{C}_k|\boldsymbol{\theta}_k) p(\mathbf{C}_k|\mathbf{X}, \boldsymbol{\theta}') d\mathbf{C}_k \tag{B.6}$$

$Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta}')$ fait intervenir deux quantités : la log-vraisemblance des données latentes $\log p(\mathbf{C}_k|\boldsymbol{\theta}_k)$ et la distribution a posteriori des données latentes $p(\mathbf{C}_k|\mathbf{X}, \boldsymbol{\theta}')$.

La première quantité se calcule avec B.1 de la manière suivante :

$$-\log p(\mathbf{C}_k|\boldsymbol{\theta}_k) = -\sum_{n=1}^{N}\sum_{f=1}^{F} \log \mathcal{N}_c(c_{k,fn}|0, \sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}) \tag{B.7}$$

$$= \sum_{n=1}^{N}\sum_{f=1}^{F} \log(\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}) + \frac{|c_{k,fn}|^2}{\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}} \tag{B.8}$$

La deuxième quantité peut être calculée par une méthodologie basée sur le filtre de Wiener et proposée par L. Benaroya en 2003 (Benaroya et al., 2003). Quelques hypothèses sont tout de même à relever : on écrit $x_n = c_{kn} + \sum_{i\neq k} c_{jn}$ avec les c_{kn} de distribution gaussienne indépendante et identiquement distribuée. Ainsi, le problème revient à un problème de séparation de sources gaussiennes et on utilise le filtrage de Wiener pour déterminer l'estimateur optimal.

Toujours en utilisant la modélisation des données B.1, on écrit la distribution a posteriori des données latentes :

$$p(\mathbf{C}_k|\mathbf{X},\boldsymbol{\theta}') = \prod_{n=1}^{N}\prod_{f=1}^{F}\mathcal{N}_C(c_{k,fn}|\mu_{k,fn}^{post'},\lambda_{k,fn}^{post'}) \tag{B.9}$$

où $\mu_{k,fn}^{post'}$ et $\lambda_{k,fn}^{post'}$ sont calculer grâce aux formules du filtrage de Wiener :

$$\begin{aligned}
\mu_{k,fn}^{post'} &= \mathbb{E}[c_{kn}(f)|x_n] && \text{(B.10)}\\
&= \left(\frac{var(c_{kn}(f))}{var(c_{kn}(f)+\sum_{j\neq k}c_{jn}(f))}\right)x_n(f) && \text{(B.11)}\\
&= \frac{\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}}{\sum_l\sum_t W_{fl}^t h_{ln}^{t\rightharpoonup}}x_{fn} && \text{(B.12)}\\
&= \frac{\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}}{\sum_l\sum_t W_{fl}^t h_{ln}^{t\rightharpoonup}}\sqrt{v_{fn}} && \text{(B.13)}
\end{aligned}$$

$$\begin{aligned}
\lambda_{k,fn}^{post'} &= var(c_{kn}(f)|x_n) && \text{(B.14)}\\
&= \left(\frac{var(c_{kn}(f))}{var(c_{kn}(f)+\sum_{j\neq k}c_{jn}(f))}\right)\sum_{j\neq k}var(c_{jn}(f)) && \text{(B.15)}\\
&= \frac{\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}}{\sum_l\sum_t W_{fl}^t h_{ln}^{t\rightharpoonup}}\sum_{l\neq k}\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup} && \text{(B.16)}
\end{aligned}$$

Les paramètres utilisés dans ces formules sont les paramètres les plus à jour. L'étape E consiste alors à calculer l'espérance de la log-vraisemblance des données latentes conditionnellement à la distribution a posteriori des données latentes :

$$Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta}') = \sum_{n=1}^{N}\sum_{f=1}^{F}\log(\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}) + \frac{|\mu_{k,fn}^{post'}|^2+\lambda_{k,fn}^{post'}}{\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}} \tag{B.17}$$

On note par la suite G_k le gain du filtre de Wiener :

$$G_k = \frac{\sum_t W_{fk}^t h_{kn}^{t\rightharpoonup}}{\sum_l\sum_t W_{fl}^t h_{ln}^{t\rightharpoonup}} \tag{B.18}$$

et

$$\begin{aligned}
V_{k,fn} &= |\mu_{k,fn}^{post'}|^2 + \lambda_{k,fn}^{post'} \\
&= |\mu_{k,fn}^{post'}|^2 + \frac{\sum_t W_{fk}^t h_{kn}^{t\rightarrow}}{\sum_l \sum_t W_{fl}^t h_{ln}^{t\rightarrow}} (\sum_l \sum_t W_{fl}^t h_{ln}^{t\rightarrow} - \sum_t W_{fk}^t h_{kn}^{t\rightarrow}) \\
&= |\mu_{k,fn}^{post'}|^2 + \sum_t W_{fk}^t h_{kn}^{t\rightarrow} \left(\frac{\sum_l \sum_t W_{fl}^t h_{ln}^{t\rightarrow}}{\sum_l \sum_t W_{fl}^t h_{ln}^{t\rightarrow}} - \frac{\sum_t W_{fk}^t h_{kn}^{t\rightarrow}}{\sum_l \sum_t W_{fl}^t h_{ln}^{t\rightarrow}} \right) \\
&= \left(\frac{\sum_t W_{fk}^t h_{kn}^{t\rightarrow}}{\sum_l \sum_t W_{fl}^t h_{ln}^{t\rightarrow}} \right)^2 v_{fn} + \sum_t W_{fk}^t h_{kn}^{t\rightarrow} \left(1 - \frac{\sum_t W_{fk}^t h_{kn}^{t\rightarrow}}{\sum_l \sum_t W_{fl}^t h_{ln}^{t\rightarrow}} \right)
\end{aligned}$$

D'où

$$V_k = G_k^2 V + (1 - G_k) \sum_t W_{fk}^t h_{kn}^{t\rightarrow} \tag{B.19}$$

B.2.3 Etape M

L'étape de maximisation revient à chercher les paramètres qui maximisent la vraisemblance donc qui minimise la fonction $Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta'})$. Pour cela, on calcule les gradients par rapport à W_{fk}^t et par rapport à h_{kn}.

On obtient :

$$\nabla_{h_{kn}} Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta'}) = \sum_f \sum_t W_{fk}^t \left(\frac{1}{\sum_{t'} W_{fk}^{t'} h_{k(n+t)}^{t'\rightarrow}} - \frac{\hat{v}'_{k,f(n+t)}}{(\sum_{t'} W_{fk}^{t'} h_{k(n+t)}^{t'\rightarrow})^2} \right) \tag{B.20}$$

$$\nabla_{W_{fk}^t} Q_k^{MV}(\boldsymbol{\theta}_k|\boldsymbol{\theta'}) = \sum_n h_{kn}^{t\rightarrow} \left(\frac{1}{\sum_{t'} W_{fk}^{t'} h_{kn}^{t'\rightarrow}} - \frac{\hat{v}'_{k,fn}}{(\sum_{t'} W_{fk}^{t'} h_{kn}^{t'\rightarrow})^2} \right) \tag{B.21}$$

En notant ∇^- la partie négative du gradient et ∇^+ la partie positive, on peut écrire les équations de mise à jour des paramètres $\theta_k^{(it+1)} \leftarrow \theta_k^{(it)} . \frac{\nabla^-}{\nabla^+}$. On a alors :

$$h_{kn}^{(it+1)} \leftarrow h_{kn}^{(it)} \frac{\sum_f \sum_t \left(W_{fk}^t \frac{\hat{v}'_{k,f(n+t)}}{(\hat{v}_{k,f(n+t)}^{(it)})^2} \right)}{\sum_f \sum_t \left(\frac{W_{fk}^t}{\hat{v}_{k,f(n+t)}^{(it)}} \right)} \tag{B.22}$$

$$W_{fk}^{t(it+1)} \leftarrow W_{fk}^{t(it)} \frac{\sum_n \left(h_{kn}^{t\rightarrow} \frac{\hat{v}'_{k,fn}}{(\hat{v}_{k,fn}^{(it)})^2} \right)}{\sum_n \left(\frac{h_{kn}^{t\rightarrow}}{\hat{v}_{k,fn}^{(it)}} \right)} \tag{B.23}$$

— Annexe C —

Détails des calculs pour l'adaptation des bases par filtrage ou par mise à jour

C.1 Détails de l'adaptation par filtrage

C.1.1 Fonction de coût

La nouvelle fonction de coût s'écrit :

$$\begin{aligned} C &= \sum d_{IS}(\boldsymbol{V}|\tilde{\boldsymbol{V}}) + P(\boldsymbol{H}|\boldsymbol{H}^M) + P(\phi|\phi^{B-spline}) \\ &= \sum_{fn} \frac{\boldsymbol{V}_{fn}}{\tilde{\boldsymbol{V}}_{fn}} - \log \frac{\boldsymbol{V}_{fn}}{\tilde{\boldsymbol{V}}_{fn}} + P(\boldsymbol{H}|\boldsymbol{H}^M) + P(\phi|\phi^{B-spline}) \end{aligned} \tag{C.1}$$

où $\boldsymbol{V}$ et $\tilde{\boldsymbol{V}}$ représentent respectivement les spectrogrammes des données et le spectrogramme estimé, $P(\boldsymbol{H}|\boldsymbol{H}^M)$ est la pénalisation que l'on ajoute à la matrice d'activation et $P(\phi|\phi^{B-spline})$ la pénalisation qu'on ajoute aux filtres.

Si l'on note θ un paramètre dans l'ensemble $\{\boldsymbol{W}^t_{fk}, \boldsymbol{H}_{k(n-t)}, P_{knots,ins}\}$ de la matrice dictionnaire, la matrice d'activation et les poids appliqués au B-splines des filtres, la dérivée par rapport à θ est donnée par :

$$\frac{\partial C}{\partial \theta} = \sum_{fn} \left(\frac{-\boldsymbol{V}_{fn}}{[\tilde{\boldsymbol{V}}_{fn}]^2} + \frac{1}{\tilde{\boldsymbol{V}}_{fn}} \right) \frac{\partial \tilde{\boldsymbol{V}}_{fn}}{\partial \theta} + \frac{\partial P(\boldsymbol{H}|\boldsymbol{H}^M)}{\partial \theta} + \frac{\partial P(\phi|\phi^{B-spline})}{\partial \theta} \tag{C.2}$$

C.1.2 Equations de mise à jour

Pour calculer les nouvelles équations de mise à jour, il est utile de donner les dérivées de $\tilde{\boldsymbol{V}}$ par rapport à $\boldsymbol{H}$, $\boldsymbol{W}$ et P :

$$\frac{\partial \tilde{\boldsymbol{V}}_{fn}}{\partial \boldsymbol{H}_{pq}} = \sum_t \sum_k (\phi_{fk} \boldsymbol{W}^t_{fk}) \delta_{pk} \delta_{q(n-t)} \tag{C.3}$$

$$\frac{\partial \tilde{\boldsymbol{V}}_{fn}}{\partial \boldsymbol{W}^t_{lp}} = \sum_k (\phi_{fk} \boldsymbol{H}_{k(n-t)}) \delta_{lf} \delta_{pk} \tag{C.4}$$

$$\begin{aligned} \frac{\partial \tilde{\boldsymbol{V}}_{fn}}{\partial P_{knots,ins}} &= \frac{-\ln 10}{20} \sum_t \sum_k B_{f,knots} \boldsymbol{W}_{f,t,k} \boldsymbol{H}_{k,n-t} 10^{\frac{-1}{20}\phi_{f,ins}} \delta_{k \in ins} \\ &= \frac{-\ln 10}{20} \sum_t \sum_k B_{f,knots} \boldsymbol{W}_{f,t,k} \boldsymbol{H}_{k,n-t} \phi_{l_{f,ins}} \\ &= \frac{-\ln 10}{20} B_{f,knots} \tilde{\boldsymbol{V}}^{ins}_{fn} \end{aligned} \tag{C.5}$$

où $p \in ins$ désigne le numéro des bases relatives au même instrument et $\tilde{\boldsymbol{V}}^{ins}_{fn}$ est la contribution de l'instrument ins.

Nous pouvons maintenant détailler le calcul des dérivées de la fonction de coût par rapport aux paramètres $\boldsymbol{W}$, $\boldsymbol{H}$ et ϕ et donner les équations de mise à jour multiplicative :

Mise à jour de $\boldsymbol{W}^t$

$$\begin{aligned} \frac{\partial C}{\partial \boldsymbol{W}^t_{mp}} &= \sum_{fn} \left(\frac{-\boldsymbol{V}_{fn}}{[\tilde{\boldsymbol{V}}_{fn}]^2} + \frac{1}{\tilde{\boldsymbol{V}}_{fn}} \right) \sum_k (\phi_{l_{fk}} \boldsymbol{H}_{k(n-t)}) \delta_{mf} \delta_{pk} \\ &= \sum_n \left(\frac{-\boldsymbol{V}_{mn} \phi_{l_{mp}} \boldsymbol{H}_{p(n-t)}}{[\tilde{\boldsymbol{V}}_{mn}]^2} + \frac{\phi_{l_{mp}} \boldsymbol{H}_{p(n-t)}}{\tilde{\boldsymbol{V}}_{mn}} \right) \end{aligned} \tag{C.6}$$

Et :

$$\boldsymbol{W}^t_{mp} \leftarrow \boldsymbol{W}^t_{mp} \frac{\sum_n \frac{\boldsymbol{V}_{mn} \phi_{l_{mp}} \boldsymbol{H}_{p(n-t)}}{[\tilde{\boldsymbol{V}}_{mn}]^2}}{\sum_n \frac{\phi_{l_{mp}} \boldsymbol{H}_{p(n-t)}}{\tilde{\boldsymbol{V}}_{mn}}} = \boldsymbol{W}^t_{mp} \frac{\sum_n \frac{\boldsymbol{V}_{mn} \boldsymbol{H}_{p(n-t)}}{[\tilde{\boldsymbol{V}}_{mn}]^2}}{\sum_n \frac{\boldsymbol{H}_{p(n-t)}}{\tilde{\boldsymbol{V}}_{mn}}} \tag{C.7}$$

Mise à jour de $\boldsymbol{H}$

$$\begin{aligned} \frac{\partial C}{\partial \boldsymbol{H}_{pq}} &= \sum_{fn} \left(\frac{-\boldsymbol{V}_{fn}}{[\tilde{\boldsymbol{V}}_{fn}]^2} + \frac{1}{\tilde{\boldsymbol{V}}_{fn}} \right) \sum_t \sum_k (\phi_{l_{fk}} \boldsymbol{W}^t_{fk}) \delta_{pk} \delta_{q(n-t)} + \boldsymbol{H}^M_{pq} \\ &= \sum_f \left(\frac{-\boldsymbol{V}_{f(q+t)}}{[\tilde{\boldsymbol{V}}_{f(q+t)}]^2} \sum_t \phi_{l_{fp}} \boldsymbol{W}^t_{[f,p]fp} + \frac{\sum_t \phi_{l_{fp}} \boldsymbol{W}^t_{fp}}{\tilde{\boldsymbol{V}}_{f(q+t)}} \right) + \boldsymbol{H}^M_{pq} \end{aligned} \tag{C.8}$$

Et :

$$\boldsymbol{H}_{pq} \leftarrow \boldsymbol{H}_{pq} \frac{\sum_f \sum_t \frac{\boldsymbol{V}_{f(q+t)} \phi_{l_{fp}} \boldsymbol{W}^t_{fp}}{[\tilde{\boldsymbol{V}}_{f(q+t)}]^2}}{\sum_f \sum_t \frac{\phi_{l_{fp}} \boldsymbol{W}^t_{fp}}{\tilde{\boldsymbol{V}}_{f(q+t)}}} \tag{C.9}$$

Mise à jour de P dans la configuration exponentielle
La dérivée de la fonction de coût est donnée par :

$$\frac{\partial C}{\partial P_{knots,ins}} = \sum_{fn} \left(\frac{-\boldsymbol{V}_{fn}}{[\tilde{\boldsymbol{V}}_{fn}]^2} + \frac{1}{\tilde{\boldsymbol{V}}_{fn}} \right) \left(\frac{-\ln 10}{20} B_{f,knots} \tilde{\boldsymbol{V}}^{ins}_{fn} \right) + \frac{\partial P(\phi|\phi^{B-spline})}{\partial P_{knots,ins}}$$

et on déduit l'équation de mise à jour

$$P_{knots,ins} \leftarrow P_{knots,ins} \frac{\frac{\ln 10}{20} \sum_{fn} \frac{1}{\tilde{\boldsymbol{V}}_{fn}} B_{f,knots} \tilde{\boldsymbol{V}}^{ins}_{fn} + \frac{\partial^- P(\phi|\phi^{B-spline})}{\partial P_{knots,ins}}}{\frac{\ln 10}{20} \sum_{fn} \frac{\boldsymbol{V}_{fn}}{[\tilde{\boldsymbol{V}}_{fn}]^2} B_{f,knots} \tilde{\boldsymbol{V}}^{ins}_{fn} + \frac{\partial^+ P(\phi|\phi^{B-spline})}{\partial P_{knots,ins}}} \tag{C.10}$$

Mise à jour de P dans la configuration gaussienne
La dérivée de V_{est} est donnée par :

$$\begin{aligned} \frac{\partial \tilde{\boldsymbol{V}}_{fn}}{\partial P_{kn,ins}} &= -\frac{2}{2\sigma} B_{f,kn} \sum_{kn} B_{f,kn} P_{kn,ins} \sum_{k,t} e^{-\frac{1}{2\sigma} \left(\sum_{kn} B_{f,kn} P_{kn,ins} \right)^2} \boldsymbol{W}_{f,t,k} \boldsymbol{H}_{k,n-t} \delta_{k \in ins} \\ &= -\frac{2}{2\sigma} B_{f,kn} \phi_{f,ins} \tilde{\boldsymbol{V}}^{ins}_{f,n} \end{aligned} \tag{C.11}$$

et de la fonction de coût :

$$\frac{\partial C}{\partial P_{knots,ins}} = \sum_{fn} \left(\frac{-\boldsymbol{V}_{fn}}{[\tilde{\boldsymbol{V}}_{fn}]^2} + \frac{1}{\tilde{\boldsymbol{V}}_{fn}} \right) \left(-\frac{2}{2\sigma} B_{f,kn} \phi_{f,ins} \tilde{\boldsymbol{V}}^{ins}_{fn} \right) + \frac{\partial P(\phi|\phi^{B-spline})}{\partial P_{knots,ins}}$$

On en déduit l'équation de mise à jour

$$P_{knots,ins} \leftarrow P_{knots,ins} \frac{\frac{2}{2\sigma} \sum_{fn} B_{f,kn} \phi_{f,ins} \frac{\tilde{\boldsymbol{V}}^{ins}_{fn}}{\tilde{\boldsymbol{V}}_{fn}} + \frac{\partial^- P(\phi|\phi^{B-spline})}{\partial P_{knots,ins}}}{\frac{2}{2\sigma} \sum_{fn} B_{f,kn} \phi_{f,ins} \tilde{\boldsymbol{V}}^{ins}_{fn} \frac{V_{f,n}}{\tilde{\boldsymbol{V}}^2_{fn}} + \frac{\partial^+ P(\phi|\phi^{B-spline})}{\partial P_{knots,ins}}} \tag{C.12}$$

C.2 Détails de l'adaptation par mise à jour

On donne ici les notations valables pour cette partie.

f	bin fréquentiel
n	trame temporelle
t	trame temporelle des bases
k	numéro de la base
$\boldsymbol{V}$	spectrogramme cible
$\tilde{\boldsymbol{V}}$	spectrogramme estimé
$\boldsymbol{W}$	bases adaptées
$\boldsymbol{W}^k_{ft}$	valeur de la k-e base au bin fréquentiel f et à la trame t
$\overline{\boldsymbol{W}}$	bases du dictionnaire
$\boldsymbol{H}$	matrice d'activation des bases

C.2.1 Fonction de coût

On rappelle la fonction de coût qu'on cherche à minimiser.

$$C(\boldsymbol{W}) = \sum_{fn} d_{IS}(\boldsymbol{V}_{fn}, \tilde{\boldsymbol{V}}_{fn}) + \lambda_W \sum_k \sum_{ft} d_{IS}(\overline{\boldsymbol{W}^k_{ft}}, \boldsymbol{W}^k_{ft}) \tag{C.13}$$

$$= \sum_{fn} \frac{\boldsymbol{V}_{fn}}{\tilde{\boldsymbol{V}}_{fn}} - \log \frac{\boldsymbol{V}_{fn}}{\tilde{\boldsymbol{V}}_{fn}} - 1 + \lambda_W \sum_k \sum_{ft} \left(\frac{\overline{\boldsymbol{W}^k_{ft}}}{\boldsymbol{W}^k_{ft}} - \log \frac{\overline{\boldsymbol{W}^k_{ft}}}{\boldsymbol{W}^k_{ft}} - 1 \right) \tag{C.14}$$

C.2.2 Equations de mise à jour

L'équation de mise à jour de $\boldsymbol{H}$ reste la même. Seul la mise à jour de $\boldsymbol{W}$ est modifée. On donne ici la dérivée de $C(\boldsymbol{W})$ par rapport à $\boldsymbol{W}^p_{lb}$:

$$\frac{\partial C(\boldsymbol{W})}{\partial \boldsymbol{W}^p_{lb}} = \sum_{fn} \left(\frac{-\boldsymbol{V}_{fn}}{V^2_{est_{fn}}} + \frac{1}{\tilde{\boldsymbol{V}}_{fn}} \right) \boldsymbol{H}_{k,n-t} \delta_{lf} \delta_{pk} \delta_{bt} \tag{C.15}$$

$$+ \lambda_W \sum_k \sum_{ft} \left(\frac{-\overline{\boldsymbol{W}^k_{ft}}}{(\boldsymbol{W}^k_{ft})^2} + \frac{1}{\boldsymbol{W}^k_{ft}} \right) \delta_{lf} \delta_{pk} \delta_{bt} \tag{C.16}$$

$$= \sum_n \left(\frac{-V_{ln}}{V^2_{est_{ln}}} + \frac{1}{\tilde{\boldsymbol{V}}_{ln}} \right) \boldsymbol{H}_{p,n-b} + \lambda_W \left(\frac{-\overline{\boldsymbol{W}^p_{lb}}}{(\boldsymbol{W}^p_{lb})^2} + \frac{1}{\boldsymbol{W}^p_{lb}} \right) \tag{C.17}$$

On en déduit l'équation de mise à jour :

$$\boldsymbol{W}_{lb}^{p} \leftarrow \boldsymbol{W}_{lb}^{p} \frac{\sum_{n} \frac{V_{ln}\boldsymbol{H}_{p,n-b}}{V_{est_{ln}}^{2}} + \lambda_{W} \frac{\overline{\boldsymbol{W}_{lb}^{p}}}{(\boldsymbol{W}_{lb}^{p})^{2}}}{\sum_{n} \frac{\boldsymbol{H}_{p,n-b}}{\tilde{\boldsymbol{V}}_{ln}} + \lambda_{W} \frac{1}{\boldsymbol{W}_{lb}^{p}}} \tag{C.18}$$

C.3 Détails de l'adaptation par mise à jour avec contrainte de décorrélation

Dans un soucis de clareté, nous noterons $B[N_b]$ comme b_1 et $B[N_b+1]$, b_2 dans la suite des calculs. Le gradient par rapport à $\boldsymbol{W}_{l'c'}$ s'écrit :

$$\begin{aligned}(\nabla_W)_{l'c'}\mathcal{P}_d(\boldsymbol{W}) &= \sum_{N_b=0}^{N_{ins}} \sum_{l=b_1}^{b_2} \sum_{c=0}^{b_1} \frac{\partial}{\partial \boldsymbol{W}_{l'c'}} \left(\frac{(\boldsymbol{W}\boldsymbol{W}^T)_{lc}}{\sigma_c \sigma_l} \right) \\ &= \sum_{N_b=0}^{N_{ins}} \sum_{l=b_1}^{b_2} \sum_{c=0}^{b_1} \frac{1}{\sigma_c \sigma_l} \frac{\partial (\boldsymbol{W}\boldsymbol{W}^T)_{lc}}{\partial \boldsymbol{W}_{l'c'}} - \frac{(\boldsymbol{W}\boldsymbol{W}^T)_{lc}}{(\sigma_c \sigma_l)^2} \left(\sigma_c \frac{\partial \sigma_l}{\partial \boldsymbol{W}_{l'c'}} + \sigma_l \frac{\partial \sigma_c}{\partial \boldsymbol{W}_{l'c'}} \right)\end{aligned}$$

On détaille chaque composante :

$$\frac{\partial (\boldsymbol{W}\boldsymbol{W}^T)_{lc}}{\partial \boldsymbol{W}_{l'c'}} = \boldsymbol{W}_{lc'}\delta_{l'c} + \boldsymbol{W}_{cc'}\delta_{ll'} \tag{C.19}$$

$$\frac{\partial \sigma_l}{\partial \boldsymbol{W}_{l'c'}} = \frac{\boldsymbol{W}_{lc'}}{\sigma_l}\delta_{ll'} \tag{C.20}$$

$$\frac{\partial \sigma_c}{\partial \boldsymbol{W}_{l'c'}} = \frac{\boldsymbol{W}_{cc'}}{\sigma_c}\delta_{cl'} \tag{C.21}$$

En injectant ces résultats dans $(\nabla_W)_{l'c'}\mathcal{P}_d(\boldsymbol{W})$, on obtient :

$$\begin{aligned}(\nabla_W)_{l'c'}\mathcal{P}_d(\boldsymbol{W}) &= \sum_{N_b=0}^{N_{ins}} \sum_{l=b_1}^{b_2} \sum_{c=0}^{b_1} \frac{\boldsymbol{W}_{lc'}\delta_{l'c} + \boldsymbol{W}_{cc'}\delta_{ll'}}{\sigma_c\sigma_l} - \frac{(\boldsymbol{W}\boldsymbol{W}^T)_{lc}(\sigma_c\sigma_l^{-1}\boldsymbol{W}_{lc'}\delta_{ll'} + \sigma_l\sigma_c^{-1}\boldsymbol{W}_{cc'}\delta_{cl'})}{(\sigma_c\sigma_l)^2} \\ &= \sum_{N_b=0}^{N_{ins}} \sum_{l=b_1}^{b_2} \sum_{c=0}^{b_1} \frac{\boldsymbol{W}_{lc'}\sigma_c\sigma_l - (\boldsymbol{W}\boldsymbol{W}^T)_{lc}\sigma_l\sigma_c^{-1}\boldsymbol{W}_{cc'}}{(\sigma_c\sigma_l)^2}\delta_{cl'} + \frac{\boldsymbol{W}_{cc'}\sigma_c\sigma_l - (\boldsymbol{W}\boldsymbol{W}^T)_{lc}\sigma_c\sigma_l^{-1}\boldsymbol{W}_{lc'}}{(\sigma_c\sigma_l)^2}\delta_{ll'} \\ &= \sum_{N_b=0}^{N_{ins}} \left(\sum_{l=b_1}^{b_2} \frac{\boldsymbol{W}_{lc'}\sigma_{l'} - (\boldsymbol{W}\boldsymbol{W}^T)_{ll'}\sigma_{l'}^{-1}\boldsymbol{W}_{l'c'}}{\sigma_l\sigma_{l'}^2} + \sum_{c=0}^{b_1} \frac{\boldsymbol{W}_{cc'}\sigma_{l'} - (\boldsymbol{W}\boldsymbol{W}^T)_{l'c}\sigma_{l'}^{-1}\boldsymbol{W}_{l'c'}}{\sigma_c\sigma_{l'}^2} \right)\end{aligned} \tag{C.22}$$

Enfin, si on isole la partie positive et la partie négative,

$$(\nabla_W)_{l'c'}\mathcal{P}_d(\boldsymbol{W}) = \sum_{N_b=0}^{N_{ins}} \left(\sum_{l=B[N_b]}^{B[N_b]+B[N_b+1]} \frac{\boldsymbol{W}_{lc'}}{\sigma_l \sigma_{l'}} + \sum_{c=0}^{B[N_b]} \frac{\boldsymbol{W}_{cc'}}{\sigma_c \sigma_{l'}} \right) - \sum_{N_b=0}^{N_{ins}} \left(\sum_{l=B[N_b]}^{B[N_b]+B[N_b+1]} \frac{(\boldsymbol{W}\boldsymbol{W}^T)_{ll'}\sigma_{l'}^{-1}}{\sigma_l \sigma_{l'}^2} + \sum_{c=0}^{B[N_b]} \frac{(\boldsymbol{W}\boldsymbol{W}^T)_{cl'}\sigma_{l'}^{-1}}{\sigma_c \sigma_{l'}^2} \right) \boldsymbol{W}_{l'c'} \tag{C.23}$$

$$= \sum_{N_b=0}^{N_{ins}} \left(\sum_{l=0}^{B[N_b]+B[N_b+1]} \frac{\boldsymbol{W}_{lc'}}{\sigma_l \sigma_{l'}} \right) - \sum_{N_b=0}^{N_{ins}} \left(\sum_{l=0}^{B[N_b]+B[N_b+1]} \frac{(\boldsymbol{W}\boldsymbol{W}^T)_{ll'}\sigma_{l'}^{-1}}{\sigma_l \sigma_{l'}^2} \right) \tag{C.24}$$

La nouvelle équation de mise à jour s'écrit donc :

$$\boldsymbol{W}_{lb}^p \leftarrow \boldsymbol{W}_{lb}^p \frac{\sum_n \frac{V_{ln}\boldsymbol{H}_{p,n-b}}{V_{est_{ln}}^2} + \lambda_W \frac{\overline{\boldsymbol{W}_{lb}^p}}{(\boldsymbol{W}_{lb}^p)^2} + \sum_{N_b=0}^{N_{ins}} \left(\sum_{l'=0}^{B[N_b]+B[N_b+1]} \frac{(\boldsymbol{W}\boldsymbol{W}^T)_{l'l}\sigma_l^{-1}}{\sigma_{l'}\sigma_l^2} \right)}{\sum_n \frac{\boldsymbol{H}_{p,n-b}}{V_{est_{ln}}} + \lambda_W \frac{1}{\boldsymbol{W}_{lb}^p} + \sum_{N_b=0}^{N_{ins}} \left(\sum_{l'=0}^{B[N_b]+B[N_b+1]} \frac{\boldsymbol{W}_{l'b}}{\sigma_{l'}\sigma_l} \right)} \tag{C.25}$$

Bibliographie

Bibliographie

Alves, D. S., Paulus, J., and Fonseca, J. (2009). Drum transcription from multichannel recordings with non-negative matrix factorization. *Proceedings European Signal Processing Conference (EUSIPCO)*, pages 894–898.

Battenberg, E. (2012). *Techniques for machine understanding of live drum performances.* PhD thesis, University of California at Berkeley.

Battenberg, E., Huang, V., and Wessel, D. (2012). Live drum separation using probabilistic spectral clustering based on itakura-saïto divergence. *Proceedings of Audio Engineering Society Convention on Time-Frequency Processing in Audio (AES).*

Bello, J. P., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., and Sandler, M. B. (2005). A tutorial on onset detection in musical signals. *IEEE Transactions on Speech and Audio Processing*, 13(5) :1035–1047.

Bello, J. P., Ravelli, E., and Sandler, M. B. (2006). Drum sound analysis for the manipulation of rythm in drum loops. *Proceedings of first MIREX.*

Bello, J. P. and Weiss, R. J. (2010). Music structure segmentation using shift-invariant probabilistic latent component analysis. *MIREX.*

Benaroya, L., Macdonagh, L., Bimbot, F., and Gribonval, R. (2003). Non negative sparse representation for wiener based source separation with a single sensor. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).*

Benetos, E., Badeau, R., Weyde, T., and Richard, G. (2014). Template adaptation for improving automatic music transcription. *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR).*

Benetos, E. and Dixon, S. (2012). A shift-invariant latent variable model for automatic music transcription. *Computer Music Journal*, 36(4) :81–94.

Benetos, E. and Ewert, S. NAD Weyde, T. (2014). Automatic transcription of pitched and unpitched sounds from polyphonic music. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3107–3111.

Berry, M. W. and Browne, M. (2005). Email surveillance using non-negative matrix factorization. *Mathematical Organization Theory*.

Bertin, N. (2009). *Les factorisations en matrices non-négatives. Approches contraintes et probabilistes, application à la transcription automatique de la musque polyphonique.* PhD thesis, Telecom ParisTech.

Böck, S., Artz, A., Krebs, F., and Shedl, M. (2012). Online real-time onset detection with recurrent neural networks. *Proceedings of the 15th International Conference on Digital Audio Effects (DAFx)*.

Böck, S., Korzeniowski, F., Schlüter, J., Krebs, F., and Widmer, G. (2016). Madmom : a new python audio and music signal processing library. *Late-Breaking Demo Session of the 17th International Society for Music Information Retrieval Conference (ISMIR)*.

Bogaards, N., Röbel, A., and Rodet, X. (2004). Sound analysis and processing with AudioSculpt 2. In *Proc. Int. Computer Music Conference (ICMC)*.

Bouvier, D., Obin, N., Liuni, M., and Roebel, A. (2016). A source/filter model with adaptive constraints for nmf-based speech separation. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 131–135.

Chen, Z., Cichocki, A., and Rutkowski, T. M. (2006). Constrained non-negative matrix factorization method for eeg analysis in early detection of alzheimer's disease. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5 :893–896.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, Dzmitry NAD Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Cichocki, A., Zdunek, R., and Amari, S.-I. (2006). Csiszár's divergences for non-negative matrix factorization : Family of new algorithms. In *International Conference on Independent Component Analysis and Blind Signal Separation (ICA)*, pages 32–39, Charleston, USA.

Ding, C., Li, T., and Peng, W. (2008). On the equivalence between non-negative matrix factorization and probabilistic latent semantic indexing. *Computational Statistics and Data Analysis*.

Dittmar, C. and Gärtner, D. (2014). Real-time transcription and separation of drum recordings based on nmf decomposition. *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, pages 187–194.

Dittmar, C. and Uhle, C. (2004). Further steps towards drum transcription of polyphonic music. *Proceedings of Audio Engineering Society Convention (AES).*

Dixon, S. (2000). On the computer recognition of solo piano music. *iAustralasian Computer Music Conference*, pages 31–37.

Drakakis, K., Rickard, S., de Fréin R., and Cichocki, A. (2008). Analysis of financial data using non-negative matrix factorization. *International Mathematical Forum.*

Durrieu, J.-L., Ozerov, A., Fevotte, C., Gaël, R., and David, B. (2009). Main instrument separation from stereophonic audio signals using a source/filter model. *EUSIPCO*, 1(EPFL-CONF-163310) :15–19.

Eggert, J. and Korner, E. (2004). Sparse coding and nmf. *IEEE International Joint Conference on Neural Networks*, 4 :2529–2533 vol.4.

Elowsson, A. and Friberg, A. (2013). Modelling perception of speed in music audio. *Proceedings of the Sound and Music Computing Conference.*

Emiya, V. (2008). *Transcription Automatique de la Musique de Piano.* PhD thesis, Telecom Paristech.

Eronen, A. (2003). Musical instrument recognition using ica-based transform of features and discriminatively trained hmms. *Proceedings of Intl. Symposium on Signal Processing and its Application (ISSPA)*, 2 :133–136.

Fessler, J. A. and Hero, A. O. (1994). Space-alternating generalized expectation-maximization algorithm. *IEEE Transactions on signal processing.*

Fevotte, C., Bertin, N., and Durrieu, J.-L. (2009). Nonnegative matrix factorization with the itakura-saito divergence : With application to music analysis. *Neural computation*, 21(3) :793–830.

Févotte, C. and Cemgil, A. T. (2009). Nonnegative matrix factorizations as probabilistic inference in composite models. *17th European Signal Processing Conference*, pages 1913–1917.

FitzGerald, D. (2004). *Automatic drum transcription and source separation.* PhD thesis, Dublin Institute of Technology, Ireland.

FitzGerald, D. (2010). Harmonic/percussive separation using median filtering. *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, pages 246–253.

FitzGerald, D., Lawlor, R., and Coyle, E. (2003a). Drum transcription in the presence of pitched instruments using prior subspace analysis. *Proceedings of Irish Signals and Systems Conference (ISSC).*

FitzGerald, D., Lawlor, R., and Coyle, E. (2003b). Prior subspace analysis for drum transcription. *Proceedings of Audio Engineering Society Convention (AES)*.

FitzGerald, D. and Paulus, J. (2006). *Unpitched percussion transcription*. FitzGerald, D. AND Paulus, J.

Fuentes, B. (2013). *L'analyse probabiliste en composantes latentes et ses adaptations aux signaux musicaux. Application à la trancription automatique de la musique et à la séparation de sources*. PhD thesis, Telecom ParisTech.

Gajhede, N., Beck, O., and Purwins, H. (2016). Convolutional neural networks with batch normalization for classifying hi-hat, snare, and bass percussion sound samples. *Proceedings of Audio Mostly : A Conference on Interaction with Sound*, pages 111–115.

Gillet, O. and Richard, G. (2004). Automatic transcription of drum loops. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4 :269–272.

Gillet, O. and Richard, G. (2005). Automatic transcription of drum sequences using audiovisual features. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 205–208.

Gillet, O. and Richard, G. (2006). Enst-drums : an etensive audio-visual database for drum signals processing. *Proceedings of the 7th International Society for Music Information Retrieval Conference (ISMIR)*.

Gillet, O. and Richard, G. (2007). Supervised and unsupervised sequence modelling for drum transcription. *Proceedings of the 8th International Society for Music Information Retrieval Conference (ISMIR)*, pages 219–224.

Gillet, O. and Richard, G. (2008). Transcription and separation of drum signals from polyphonic music. *IEEE Transactions on Audio, Speech and Language Processing*, 16(3) :529–540.

Gloriot, X. and Bengio, Y. (2010). Understanding the difficulty of feedforward in neural networks. *Aistats*, 9 :249–256.

Goto, M. (2003). Rwc music database : Music genre database and musical instrument sound database. *Proceedings of the 4th International Society for Music Information Retrieval Conference (ISMIR)*, pages 229–230.

Goto, M., Hashiguchi, H., Nishimura, T., and Oka, R. (2002). RWC music database : Popular, classical and jazz music databases. *Proceedings of the 3rd International Society on Music Information Retrieval Conference (ISMIR)*, 2 :287–288.

Gouyon, F., Pachet, F., and Delerue, O. (2000). On the use of zero-crossing rate for an application of classification of percussive osunds. *Proceedings of the International Conference on Digital Audio Effects (DAFx)*.

Grindlay, G. and Ellis, D. P. (2009). Multi-voice polyphonic music transcription using eigeninstruments. *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*.

Guillamet, D., Schile, B., and Vitrià, J. (2001). Color histogram classification using nmf. Technical report, Centre de Visiò Per Computador, Université autonome de Barcelone.

Hahn, H. (2015). *Expressive sampling synthesis - Learning extended source-filter model from instrument sound database for expressive sample manipulations*. PhD thesis, UPMC Université Paris VI.

Herrera, P., Dehamel, A., and Gouyon, F. (2003). Automatic labeling of unpitched percussion sounds. *Proceedings of Audio Engineering Society Convention (AES)*.

Herrera, P., Sandvold, V., and Gouyon, F. (2004). Percussion-related semantic descriptors of music audio files. *Audio Engineering Society Conv. : Metadata for Audio (AES)*.

Herrera, P., Yeterian, A., and Gouyon, F. (2002). Automatic classification of drum sounds : A comparison of feature selection methods and classification techniques. *Proceedings of Intl. Conf. on Music and Artificial Intelligence (ICMAI)*.

Hoffman, M. D. (2010). Approximate maximum a posteriori interference with entropic priors. *CoRR*, abs/1009.5761.

Hofmann, T. (1999). Probabilistic latent semantic analysis. *Uncertainty in Artificail Intelligence*.

Hoyer, P. O. (2002). Non-negative sparse coding. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing, NNSP 2002, Martigny, Valais, Switzerland, September 4-6, 2002.*, pages 557–565.

Hoyer, P. O. (2004). Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*.

Jacques, C., Aknin, A., and Röbel, A. (2018). Mirex 2018 : Automatic drum transcritpion with convolutional neural networks. *MIREX*.

Jacques, C. and Röbel, A. (2018). Automatic drum transcription with convolutional neural networks. *Proceedings of the International Conference on Digital Audio Effects (DAFx)*.

Jacques, C. and Röbel, A. (2019). Data augmentation for onset detection and drum transcription (submitted).

Jaureguiberry, X., Leveau, P., Maller, D., and Burred, J. J. (2011). Adaptation of source-specific dictionaries in non-negative matrix factorization for source separation. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Kashino, K. and Murase, H. (1997). A sound source identification system for ensemble music based on template adaptation and music stream extraction. *Speech Communication*.

Kelz, R., Dorfer, M., Korzeniowski, F., Böck, S., Artz, A., and Widmer, G. (2016). On the potential of simple framewise approaches to piano transcription. *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*.

Kim, S.-P., Rao, Y. N., Ergodmus, D., C., S. J., Nicolelis, M. A. L., and Principe, J. C. (2005). Determining patterns in neural activity for reaching movements using non-negative matric factorization. *EURASIP Journal on Applied Sig. Proc., Special Issue of Trnds in Brain Computer Interfaces*.

Kompass, R. (2005). A generalized divergence measure for non-negative matrix factorization. In *Neuroinformatics workshop*, Torun, Poland.

Laroche, J. and Dolson, M. (1999). Improved phase vocoder time-scale modification of audio. *IEEE Transactions on Speech and Audio Processing*, 7(3) :323–332.

Lee, D. and Seung, S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755) :788–791.

Li, S., Hou, X., Zhang, H., and Cheng, Q. (2001). Learning spatially localized, parts-based representation. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1 :207–212.

Lindsay-Smith, ., McDonald, S., and Sandler, M. (2012). Drumkit transcription via convolutive nmf. *Proceedings of the International Conference on Digital Audio Effects (DAFx)*.

Liu, W. and Yuan, K. (2008). Sparse p-norm nonnegative matrix factorization for clustering gene expression data. *International Journal of Data Mining and Bioinformatics*.

Marolt, M., Kavcic, A., and Provosnik, M. (2002). Neural network for note onset detection in piano music. *Proceedings of the International Computer Music Conference (ICMC)*.

Marxer, R. and Janer, J. (2013). Study of regularizations and constraints in nmf-based drums monaural separation. *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, pages 1–6.

McFee, B., Humphrey, E., and Bello, J. (2015). A software framework for musical data augmentation. *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*.

Miron, M., Davies, M. E. P., and Gouyon, F. (2013a). Improving the real-time performance of a causal audio drum transcription system. *Proceedings of Sound and Music Computing Conference (SMC)*, pages 402–407.

Miron, M., Davies, M. E. P., and Gouyon, F. (2013b). An open-source drum transcription system for pure data and max msp. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 221–225.

Mitsufuji, Y., Liuni, M., Baker, A., and Roebel, A. (2007). Online separation tensor deconvolution for source detection in 3dtv audio. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Moreau, A. and Flexer, A. (2007). Drum transcription in polyphonic music using non-negative matrix factorisation. *Proceedings of the 8th International Society for Music Information Retrieval Conference (ISMIR)*, pages 353–354.

Nakano, M., Kameoka, H., Le Roux, J., Kitano, Y., Ono, N., and Sagayama, S. (2010). Convergence-guaranteed multiplicative algorithms for nonnegative matrix factorization with β-divergence. *International Workshop on Machine Learning for Signal Processing, In Proc. IEEE*, 10 :283–288.

Nakano, T., Goto, M. NAD Ogata, J., and Hiraga, Y. (2005). Voice drummer : a music notation interface of drum sounds using voice percussion inpu. *Proceedings of Annual ACM Symposium on User Interface Software and Technology (UIST)*, pages 49–50.

Paatero, P. and Tapper, U. (1994). Positive matrix factorization : A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2) :111–126.

Paulus, J. (2009). *Signal processing methods for drum transcription and music structure analysis*. PhD thesis, Tampere University of Technology, Finland.

Paulus, J. and Klapuri, A. (2009). Drum sound detection in polyphonic music with hidden markov models. *EURASIP Journal on Audio, Speech, and Music Processing*.

Paulus, J. and Virtanen, T. (2005). Drum transcription with non-negative spectrogram factorisation. *Proceedings European Signal Processing Conference (EUSIPCO)*.

Prockup, M., Schmidt, E., Scott, J., and E., K. Y. (2013). Toward understanding expressive percussion through content based analysis. *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*.

Röbel, A. (2003). A new approach to transient processing in the phase vocoder. *Proceedings of the 6th International Conference on Digital Audio Effects (DAFx)*, pages 344–349.

Röbel, A., Pons, J., Liuni, M., and Lagrange, M. (2015). On automatic drum transcription using non-negative matrix deconvolution and itakura-saito divergence. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 414–418.

Röbel, A. and Rodet, X. (2005). Efficient Spectral Envelope Estimation and its application to pitch shifting and envelope preservation. *International Conference on Digital Audio Effects*, pages 30–35. cote interne IRCAM : Roebel05b.

Rossignol, M., Lagrange, M. NAD Lafay, G., and Benetos, E. (2015). Alternate level clustering for drum transcription. *Proceedings of European Signal Processing Conf. (EUSIPCO)*, pages 2023–2027.

Roy, P., Pachet, F., and Krakowski, S. (2007). Improving the classification of percussive sounds with analytical features : A case of study. *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, pages 229–232.

Sak, H., Senior, A. W., and Beaufays, F. (2014). Long short-term memory recurrent neural network architecture for large scale acoustic modelling. *Proceedings of 15th Annual Conference of the International Speech Communication Association*, pages 338–342.

Salamon, J. and Bello, J. P. (2017). Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, abs/1608.04363.

Schloss, W. A. (1985). *On the automatic transcription of percussive music - From acoutic signal to high-level analysis*. PhD thesis, Standford University.

Schlüter, J. and Böck, S. (2014). Improved musical onset detection with convolutional neural networks. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Schlüter, J. and Grill, T. (2015). Exploring data augmentation for improved singing voice detection with neural networks. *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*.

Scholler, S. and Perwins, H. (2010). Sparse coding for drum classification and its use as a similarity measure. *Proceedings of International Workshop on Machine Learning and Music (MML)*, pages 9–12.

Scholler, S. and Perwins, H. (2011). Sparse approximations for drum sound classification. *Journal of Selected Topics Signal Processing*, 5(5) :933–940.

Schroeter, J. (2018). Mirex 2018 : Drum transcription. *MIREX*.

Shashanka, M., Raj, B., and Smaragdis, P. (2008). Probabilistic latent variable model as nonnegative factorizations. *Computational Intelligence and Neuroscience*.

Smaragdis, P. (2004). Nonnegative matrix factor deconvolution ; extraction of multiple sound source from monophonic inputs. *Independent Component Analysis and Blind Signal Separation*, 3195.

Smaragdis, P. and Brown, J. (2003). Non-negative matrix factorization for polyphonic music transcription. *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 177–180.

Smaragdis, P. and Raj, B. (2007). Shift-invariant probabilistic latent component analysis. *Journal of Machine Learning Research*.

Southall, C. (2018). Mirex 2017 drum transcription submissions. *MIREX*.

Southall, C., Jillings, N., Stables, R., and Hockman, J. (2017a). Adtweb : An open source browser based automatic drum transcription system. *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*.

Southall, C., Stables, R., and Hockman, J. (2016). Automatic drum transcription using bi-directional recurrent neural networks. *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, pages 591–597.

Southall, C., Stables, R., and Hockman, J. (2017b). Automatic drum transcription for polyphonic recordings using soft attention mechanisms and convolutional neural networks. *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, pages 606–612.

Souza, V. M. A., Batista, G. E. A. P. A., and Souza-Filho, N. E. (2015). Automatic classification of drum sounds with indefinite pitch. *Proceedings of Intl. Joint Conf. on Neural Networks (IJCNN)*.

Springer, E., editor (2006). *Unpitched percussion transcription*. FitzGerald, D. AND Paulus, J. AND Klapuri, A. AND Davy M.

Steelant, D. V., Tanthe, K., Degroeve, S., Baets, B., Leman, M., and Martens, J.-P. (2004). Classification of percussive sounds using support vector machine. *Proceedings of Annual Machine Learning Conference of Belgium and The Netherlands (BENELEARN)*, pages 146–152.

Sutton, R. S. (1986). Two problems with backpropagation and other steepest descent learning procedures for network. *Annual Conference of Cognitive Science Society*, pages 823–831.

Thompson, L., Dixon, S., and Mauch, M. (2014). Drum transcription via classification of bar-level rhythmic patterns. *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, pages 187–192.

Tindale, A., Kapur, A., Tzanetakis, G., and Fujinaga, I. (2004). Retrieval of percussion gestures using timbre classification technique. *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR).*

Tzanetakis, G., Kapur, A., and McWalter, R. I. (2005). Subband-based drum transcription for audio signals. *Proceedings of Workshop on multimedia Signal Processing.*

Vincent, E., Bertin, N., and Badeau, R. (2008). Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech, and Language Processing.*

Virtanen, T. (2007). Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *IEEE Transactions on Audio, Speech and Language Processing*, 15(3) :1066–1074.

Vogl, R., Dorfer, M., and Knees, P. (2016a). Drum transcription from polyphonic music with recurrent neural networks. *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).*

Vogl, R., Dorfer, M., and Knees, P. (2016b). Recurrent neural networks for drum transcription. *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, pages 730–736.

Vogl, R., Dorfer, M., Widmer, G., and Knees, P. (2017). Drum transcription via joint beat and drum modelling using convolutional recurent neural networks. *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR).*

Vogl, R. and Knees, P. (2018). Mirex submission for drum transcription 2018. *MIREX.*

Vogl, R., Widmer, G., and Knee, P. (2018). Towards multi-instrument drum transcription. *Proceedings of the International Conference on Digital Audio Effects (DAFx).*

Wang, Q., Zhou, R., and Yan, Y. (2017). A two stage approach to note-level transcription of a specific piano. *Applied Science.*

Wu, C.-W., Dittmar, C., Southall, C., Vogl, R., Widmer, G., Hockman, J., Müller, M., and Lerch, A. (2017). A review of automatic drum transcription. *IEEE Transactions on Audio, Speech, and Language Processing.*

Wu, C.-W. and Lerch, A. (2015a). Drum transcription using partially fixed non-negative matrix factorization. *Proceedings of European Signal Processing Conference (EUSIPCO).*

Wu, C.-W. and Lerch, A. (2015b). Drum transcription using partially fixed non-negative matrix factorization with template adaptation. *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*.

Wu, C.-W. and Lerch, A. (2016). On drum playing technique detection in polyphonic mixtures. *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, pages 218–224.

Wu, C.-W. and Lerch, A. (2017). Automatic drum transcription using the student-teacher learniing paradigm with unlabeled music data. *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*.

Yoshii, K., Goto, M., and Okuno, H. G. (2004). Automatic drum sound description for real-world music using template adaptation and matching methods. *Proceedings of the 5th International Society for Music Information Retrieval Conference (ISMIR)*.

Yoshii, K., Goto, M., and Okuno, H. G. (2005). Adamast : A drum sound recognizer on adaptation and matching of spectrogram templates. *Annual Music Information Retrieval Evaluation eXchange (MIREX)*.

Yoshii, K., Goto, M., and Okuno, H. G. (2007). Drum sound recognition for polyphonic audio signals by adaptation and matching of spectrogram template with harmonic structure suppression. *IEEE Transactions on Audio, Speech and Language processing*, 15(1) :333–345.

Zivanovic, M., Röbel, A., and Rodet, X. (2004). A new approach to spectral peak classification. In *Proceedings of the 12th European Signal Processing Conference (EUSIPCO)*, pages 1277–1280, Vienna, Austria. cote interne IRCAM : Zivanovic04a.