ANR-14-CE24-0002-01

Projet DYCI2,
WP1 Écoute informée créative,
WP1.1 Séparation informée multi-canal

Rapport de livrable :

L1.1 Séparation informée multi-canal.

Livrable	Date	Contributeurs	Rédacteurs	Contenu
L1.1 Version 01	Octobre 2016	E. Vincent, A. Liutkus (Inria)	E. Vincent	Maquette Logicielle KAMIR, Rapport scientifique

Résumé

Ce document présente nos travaux la séparation informée multi-canal d'enregistrements musicaux, et plus particulièrement une méthode basée sur le kernel additive modeling *matérialisée par le logiciel KAMIR et une autre méthode basée sur* le deep learning.

Adresse du livrable logiciel

DYCI2_WP1_L1.1.zip
sur
https://forge.ircam.fr/p/Dyci2/

Projet DYCI2 - Livrable 1.1
Séparation informée multicanal

Emmanuel Vincent et Antoine Liutkus (Inria)
28 octobre 2016

Table des matières

1 Introduction

Dans le projet DYCI2, nous souhaitons que des agents musicaux virtuels (ordinateurs) soient capables d'écouter la scène sonore, afin d'analyser la structure musicale à la manière d'un musicien. Le système auditif humain retrouve à partir du signal sonore des paramètres musicaux et les organise, par exemple verticalement ou horizontalement dans un plan temps-fréquence. La tâche 1.1 porte sur la structuration verticale du son musical, aussi dénomée "séparation de sources". Il s'agit d'extraire à chaque instant le signal correspondant à chaque source sonore (piano, basse, voix...) ou à un ensemble de sources (percussions, accompagnement...). Cette tâche est nécessaire car, en conditions *live*, chaque microphone capte les signaux de toutes les sources à la fois. Ce phénomène est connu sous le nom de "repisse".

Jusqu'à récemment, il était impossible de séparer les différentes sources d'un enregistrement musical avec une qualité suffisante sans information *a priori* [6, 13]. Une information *a priori* exploitable dans le cadre de DYCI2 concerne la position approximative des microphones utilisés pour enregistrer la scène. En effet, les microphones ne sont pas placés au hasard : chaque microphone est plus proche d'une source ou d'un ensemble de sources que des

autres de sorte que, pour chaque microphone, une source ou un ensemble de sources sont prépondérants. Nous avons effectué des travaux, en collaboration avec International Audio Laboratories Erlangen (Allemagne) et New York University (États-Unis), sur la suppression de la repisse dans les enregistrements musicaux multicanaux informée par la position approximative des microphones. Ces travaux ont donné lieu à deux publications [11, 10] et sont décrits dans la partie 2 de ce rapport.

En parallèle, nous avons effectué des travaux sur la séparation d'enregistrements multicanaux par apprentissage profond (*deep learning*). Cette méthode d'apprentissage automatique a révolutionné l'état de l'art du traitement de la parole et du texte. Depuis sa première application à la séparation d'enregistrements musicaux en 2014, elle est devenue le nouvel état de l'art. Nous sommes parmi les premiers au niveau mondial à avoir adapté le *deep learning* à la séparation d'enregistrements musicaux et nous sommes les premiers à notre connaissance à l'avoir fait sur des enregistrements multicanaux. Ces travaux ont donné lieu à deux publications [8, 9] et sont décrits dans la partie 3 de ce rapport. Notre système de séparation d'enregistrements musicaux multicanaux basé sur le *deep learning* a été classé premier *ex aequo* lors des campagnes d'évaluation internationales SiSEC 2015 [1] [2] et 2016 [3].

Enfin, nous avons écrit un article sur la séparation d'enregistrements musicaux dans une revue de culture scientifique en ligne destinée au grand public [7] et nous avons commencé un travail en collaboration avec Cork Institute of Technology (Irelande) sur un nouveau formalisme de modélisation des signaux multicanaux, également évalué dans un cadre informé par la position des sources et des microphones [2]. Ce dernier travail n'est pas détaillé dans ce document.

2 Suppression informée de la repisse

Lorsqu'il enregistre des musiciens en *live*, un ingénieur du son utilise typiquement un micro pour chaque *voix*, qui peut être un seul instrument ou un ensemble d'instruments du même type, par exemple une section de violons ou de cuivres. Comme illustré dans la Figure 1, chaque microphone n'enregistre alors pas uniquement la voix désirée mais plutôt un *mélange* où la voix désirée est certes prépondérante mais les autres voix sont audibles. L'annulation de la repisse par filtrage adaptatif a une performance limitée en

1. `https://sisec.inria.fr/sisec-2015/2015-professionally-produced-music-recordings/`
2. `http://www.onn.nii.ac.jp/sisec15/evaluation_result/MUS/MUS2015.html`
3. `https://sisec.inria.fr/home/2016-professionally-produced-music-recordings/`

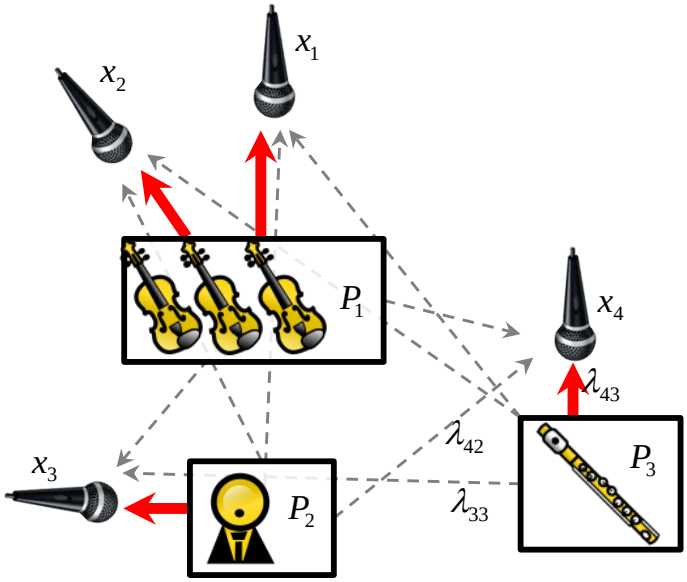

FIGURE 1 – Illustration du phénomène de repisse. Chaque voix est associée à un microphone au moins. Les flèches rouges indiquent la voix prépondérante et les flèches en pointillés représentent la repisse.

pratique, en raison des mouvements des sources dans l'espace. Pour résoudre ce problème, nous proposons un algorithme basé sur un filtrage purement énergétique, dans la lignée de [4]. Compte tenu de l'importante masse de données à traiter, cet algorithme repose sur l'approche de *Kernel Additive Modeling* (KAM) [5], particulièrement efficace du point de vue calculatoire. Nous appelons notre algorithme KAMIR : *Kernel Additive Modeling for Interference Reduction*.

2.1 Modèle

Soit J le nombre de voix et I le nombre de microphones. Le signal $x_i(t)$ enregistré par le microphone $i \in \{1, \ldots, I\}$ est appelé *canal* et il est égal à la somme des contributions $c_{ij}(t)$ de toutes les voix, appelées *images*. Nous calculons la transformée de Fourier à court terme (STFT) $x_i(f, n)$ de chaque canal $x_i(t)$. Il en résulte I matrices de dimension $F \times N$, où F est la nombre de bandes de fréquence et N le nombre de trames temporelles. En chaque

point temps-fréquence (f, n), nous avons

$$x_i(f,n) = \sum_{j=1}^{J} c_{ij}(f,n), \tag{1}$$

où $c_{ij}(f,n)$ est la STFT de $c_{ij}(t)$. Nous faisons l'hypothèse que les $c_{ij}(f,n)$ sont indépendants et distribués de façon gaussienne

$$c_{ij}(f,n) \sim \mathcal{N}_c(0, \sigma_{ij}^2(f,n)), \tag{2}$$

où le paramètre $\sigma_{ij}^2(f,n)$ représente le spectre de puissance à court terme de $c_{ij}(t)$, qui peut s'exprimer comme le produit d'un spectre $v_j(f,n)$ commun à tous les microphones et d'un gain $\lambda_{ij}(f)$ dépendant du microphone :

$$\sigma_{ij}^2(f,n) = \lambda_{ij}(f) v_j(f,n) \approx |c_{ij}(f,n)|^2, \tag{3}$$

Le gain $\lambda_{ij}(f)$ quantifie le degré de repisse de la voix j dans le canal i à la fréquence f.

2.2 Algorithme

Les gains $\lambda_{ij}(f)$ sont initialisés par

$$\forall\,(i,j,f),\ \lambda_{ij}(f) = \begin{cases} 1 & \text{si } i \in \varphi(j) \\ \rho & \text{sinon,} \end{cases} \tag{4}$$

où $\varphi(j)$ dénote l'ensemble (connu *a priori*) des microphones où la voix j est prépondérante et $\rho \in [0,\ 1]$ le degré minimal de repisse attendu sur les autres microphones. L'image estimée $\widehat{c}_{ij}$ de chaque voix j est initialisée comme x_i pour tout $i \in \varphi(j)$.

Les paramètres $\lambda_{ij}(f)$ et $v_j(f,n)$ et les images $\widehat{c}_{ij}(f,n)$ sont ensuite réestimés de façon itérative en alternant deux étapes. Dans l'étape de *séparation*, les images $\widehat{c}_{ij}(f,n)$ sont réestimées par filtrage de Wiener en fonction des paramètres estimés à l'itération précédente :

$$\widehat{c}_{ij}(f,n) = \frac{\lambda_{ij}(f) v_j(f,n)}{\sum_{j'=1}^{J} \lambda_{ij'}(f) v_{j'}(f,n)} x_i(f,n). \tag{5}$$

Le choix de ce filtre, optimal au sens du minimum de l'erreur quadratique moyenne (MMSE), est une conséquence du modèle local gaussien (2). Dans

l'étape de *mise à jour*, les paramètres $\lambda_{ij}(f)$ et $v_j(f,n)$ sont réestimés en fonction des images estimées à l'étape de séparation :

$$v_j(f,n) \leftarrow \frac{1}{|\varphi(j)|} \sum_{i \in \varphi(j)} \frac{1}{\lambda_{ij}(f)} |\widehat{c}_{ij}(f,n)|^2 \tag{6}$$

$$\lambda_{ij}(f) \leftarrow \lambda_{ij}(f) \frac{\sum_t \widehat{z}_i(f,n)^{-2} z_i(f,n) v_j(f,n)}{\sum_t \widehat{z}_i(f,n)^{-1} v_j(f,n)}, \tag{7}$$

où $|\varphi(j)|$ dénote le nombre de canaux non-nuls de la fonction φ, $z_i(f,n) = |x_i(f,n)|^2$ et $\widehat{z}_i(f,n) = \sum_j \lambda_{ij}(f) v_j(f,n)$. La mise à jour de $\lambda_{ij}(f)$ est une mise à jour multiplicative, motivée par le fait que l'estimation de $\lambda_{ij}(f)$ au sens du maximum de vraisemblance équivaut à minimiser la divergence d'Itakura-Saito (IS) entre z_i et $\widehat{z}_i$, ce qui est un problème de factorisation matricielle positive (NMF).

En pratique, l'expérience montre qu'il est bénéfique de tempérer ces mises à jour de sorte que $\lambda_{ij}(f)$ ne soit pas trop modifié. Pour cela, nous forçons le facteur multiplicatif dans (7) à être compris entre 1/10 et 10, puis nous renormalisons $v_j(f,n)$ et $\lambda_{ij}(f)$ par

$$v_j(f,n) \leftarrow v_j(f,n) \sum_i \lambda_{ij}(f) \tag{8}$$

$$\lambda_{ij}(f) \leftarrow \max[\rho, \frac{\lambda_{ij}(f)}{\sum_{i'} \lambda_{i'j}(f)}] \tag{9}$$

de sorte que $\lambda_{ij}(f)$ reste dans l'intervalle $[\rho,\ 1]$ après chaque mise à jour. La Figure 2 montre un example de $\lambda_{ij}(f)$ appris. Après convergence, les images $\widehat{c}_{ij}(t)$ dans le domaine temporel sont obtenues par STFT inverse.

2.3 Évaluation

Nous avons comparé 5 réglages différents de KAMIR (notés K1 à K5) à l'algorithme de Kokkinis [4] pour un enregistrement avec 21 microphones et jusqu'à 11 voix simultanées. Le réglage K5 correspond à une version simplifiée de KAMIR de complexité calculatoire réduite. La performance a été évaluée par un test d'écoute impliquant 21 sujets et portant sur deux questions : le niveau de réduction de la repisse et la qualité globale perçue.

La Figure 3 montre que l'algorithme de Kokkinis a tendance à fournir une qualité globale légèrement meilleure, en particulier pour les voix #1 and #4. Mais les différentes versions de KAMIR fonctionnent bien aussi, notamment lorsque plusieurs microphones sont disponibles pour une voix donnée. C'est

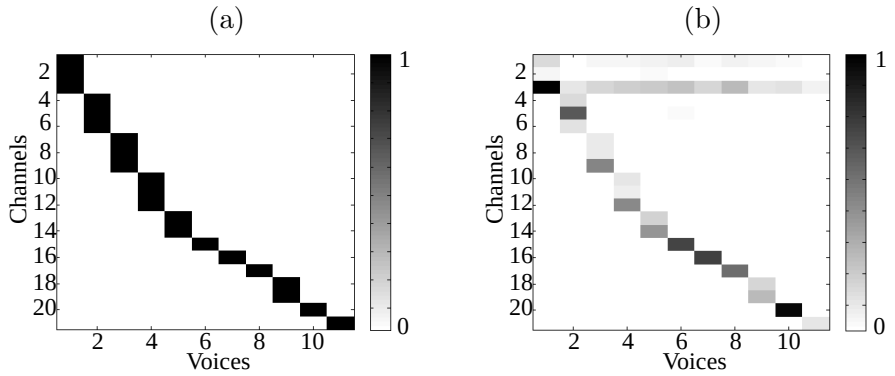

FIGURE 2 – Gains $\lambda_{ij}(f, n)$ (a) initiaux et (b) appris pour un exemple à $I = 21$ canaux et $J = 11$ voix.

le cas pour la voix #5 où KAMIR fournit une bien meilleure réduction de la repisse sans réduction de la qualité globale. D'autre part, l'algorithme K5 fonctionne en temps réel, contrairement à celui de Kokkinis.

3 Séparation multicanal par *deep learning*

Le *deep learning* repose sur l'apprentissage de la tâche à effectuer (classification, régression) par un réseau de neurones profond (DNN) modélisant une fonction non-linéaire complexe. Les DNNs sont aujourd'hui couramment utilisés pour le rehaussement de signaux de parole bruitée monocanal. Ils remplacent les modèles spectraux classiques de la parole et du bruit afin de prédire le spectre de puissance de la parole propre ou bien le filtre de Wiener à partir du spectre de puissance de la parole bruitée. Ils peuvent être vus comme une méthode de séparation de sources informée, dans la mesure où ils reposent sur un apprentissage supervisé qui nécessite de connaître *a priori* les types de sources présentes dans le mélange. Leur usage pour la séparation d'enregistrements musicaux reste toutefois peu répandu [3, 12], tout comme leur usage dans un contexte multicanal. Nous décrivons ci-dessous notre algorithme dnnsep, qui est à notre connaissance le premier algorithme de séparation d'enregistrements multicanaux basé sur les DNNs.

3.1 Modèle

Nous dénotons par $\mathbf{c}_j(t)$ le vecteur de taille $I \times 1$ des images de la source j sur tous les canaux et $\mathbf{x}(t)$ le mélangede taille $I \times 1$ observé. En chaque

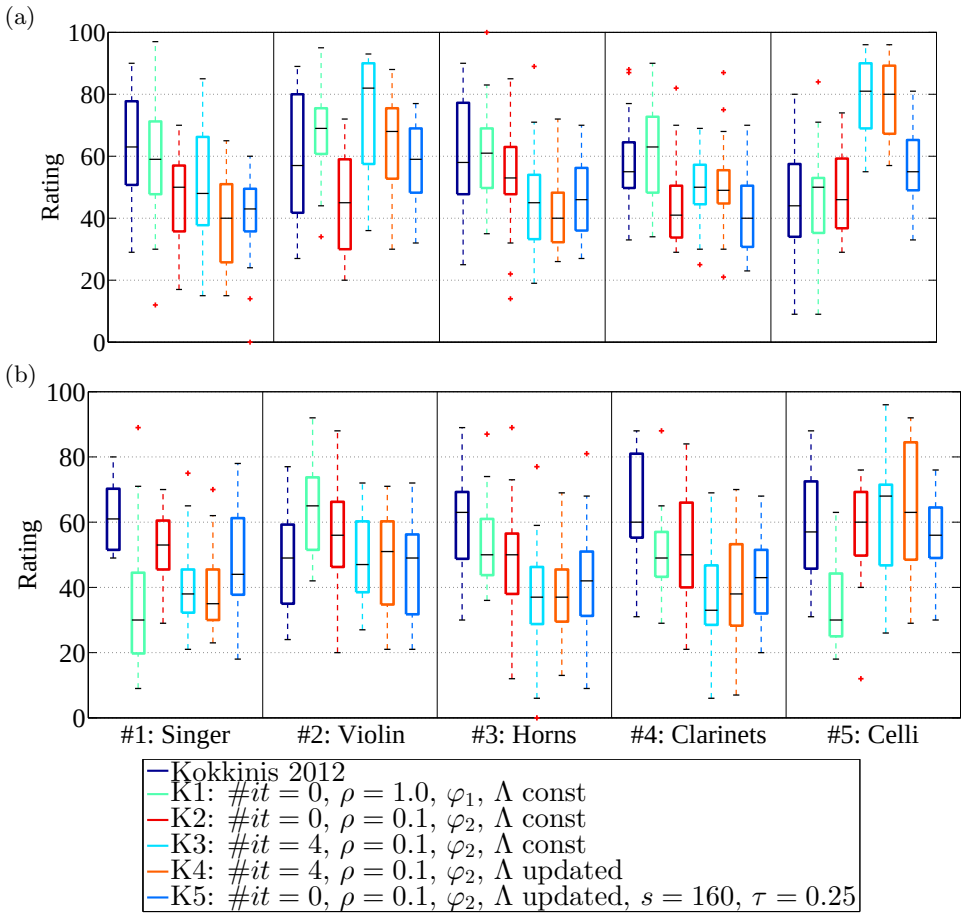

FIGURE 3 – (a) Niveau de réduction de la repisse et (b) la qualité globale perçus. Les boîtes, les traits noirs et les croix rouges représentent respectivement l'intervalle inter-quartiles, la médiane et les valeurs extrêmes.

point temps-fréquence (f, n), nous avons donc :

$$\mathbf{x}(f,n) = \sum_{j=1}^{J} \mathbf{c}_j(f,n). \tag{10}$$

Nous faisons l'hypothèse que les $\mathbf{c}_j(f,n)$ sont indépendants et distribués de façon gaussienne multivariée à valeurs complexes

$$\mathbf{c}_j(f,n) \sim \mathcal{N}_c(\mathbf{0}, v_j(f,n)\mathbf{R}_j(f)), \tag{11}$$

où $v_j(f,n)$ est le spectre de puissance à court terme de la source j et $\mathbf{R}_j(f)$ sa *matrice de covariance spatiale* de taille $I \times I$, qui représente à la fois sa

position et sa largeur spatiale [1].

3.2 Algorithme

Les spectres de puissance $v_j(f,n)$ de toutes les sources sont initialisés par un DNN noté DNN_0 qui prédit leur valeur à partir du spectre de puissance du mélange $\|\mathbf{x}(f,n)\|^2/I$. Les matrices de covariance spatiale $\mathbf{R}_j(f)$ sont initialisées comme des matrices identité de taille $I \times I$.

Les paramètres $v_j(f,n)$ et $\mathbf{R}_j(f)$ et les images $\widehat{\mathbf{c}}_j(f,n)$ sont ensuite réestimés de façon itérative par un algorithme espérance-maximisation (EM). Dans l'étape E, les images $\widehat{\mathbf{c}}_j(f,n)$ sont réestimées par

$$\widehat{\mathbf{c}}_j(f,n) = \mathbf{W}_j(f,n)\mathbf{x}(f,n), \tag{12}$$

où

$$\mathbf{W}_j(f,n) = v_j(f,n)\mathbf{R}_j(f)\left(\sum_{j'=1}^{J} v_{j'}(f,n)\mathbf{R}_{j'}(f)\right)^{-1} \tag{13}$$

est le filtre de Wiener multicanal, et leurs moments d'ordre 2 non-centrés sont calculés par

$$\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) = \widehat{\mathbf{c}}_j(f,n)\widehat{\mathbf{c}}_j^H(f,n) + \left(\mathbf{I} - \mathbf{W}_j(f,n)\right) v_j(f,n)\mathbf{R}_j(f), \tag{14}$$

avec $\mathbf{I}$ la matrice identité. Dans l'étape M, les matrices de covariance spatiale $\mathbf{R}_j(f)$ sont mises à jour comme

$$\mathbf{R}_j(f) = \frac{1}{N}\sum_{n=1}^{N} \frac{1}{v_j(f,n)}\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n). \tag{15}$$

Les spectres de puissance $v_j(f,n)$ sont d'abord estimés de façon non-contrainte en chaque point temps-fréquence comme

$$z_j(f,n) = \frac{1}{I}\,\text{tr}(\mathbf{R}_{jf}^{-1}\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)), \tag{16}$$

où $\text{tr}(\cdot)$ dénote la trace d'une matrice. Ils sont ensuite mis à jour à partir d'un DNN qui corrige l'estimation du spectre $v_j(f,n)$ à partir de l'estimée initiale $z_j(f,n)$. Le DNN utilisé à l'itération l est noté DNN_l.

Nous avons constaté que la mise à jour (15), optimale au sens du maximum de vraimsemblance, ne donnait pas toujours de bons résultats lorsque $v_j(f,n)$ est mal estimé. Nous avons proposé la mise à jour pondérée suivante :

$$\mathbf{R}_j(f) = \left(\sum_{n=1}^{N}\omega_j(f,n)\right)^{-1}\sum_{n=1}^{N}\frac{\omega_j(f,n)}{v_j(f,n)}\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n), \tag{17}$$

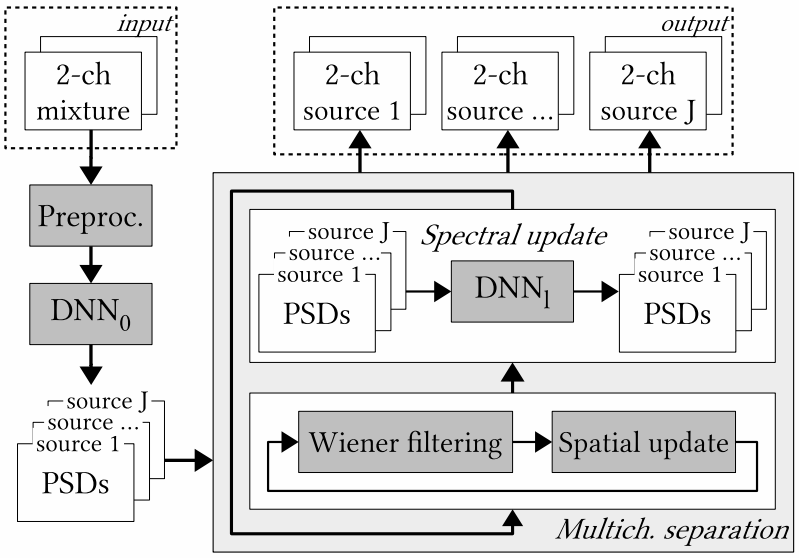

FIGURE 4 – Illustration de l'algorithme dnnsep.

où $\omega_j(f,n)$ est le poids associé à la source j au point temps-fréquence (f,n). En pratique, nous utilisons $\omega_j(f,n) = v_j(f,n)$. Cette pondération accroît l'importance des points temps-fréquence de puissance élevée dans le calcul, et ainsi la précision de $\mathbf{R}_j(f)$ obtenue expérimentalement. Nous considérons aussi une version simplifiée de cette mise à jour pondérée, où $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)$ est simplement estimé comme $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) = \widehat{\mathbf{c}}_j(f,n)\widehat{\mathbf{c}}_j^H(f,n)$.

Dans nos expériences, les DNNs sont des perceptrons multicouches (MLP) à 3 couches cachées, appris au sens du minimum de l'erreur quadratique moyenne sur un ensemble de développement pour lequel les signaux sources sont connus. Ils prennent en entrée un supervecteur constitué du spectre de la trame courante et des spectres de quelques trames passées et futures (contexte), dont la dimension a été réduite par analyse en composantes principales (PCA). Les spectres d'amplitude sont utilisés plutôt que les spectres de puissance comme entrées et sorties des DNNs. Pour chaque initialisation ou mise à jour "spectrale" de $v_j(f,n)$ par DNN, nous effectuons plusieurs mises à jour "spatiales" de $\mathbf{R}_j(f)$ en appliquant (13), (12), (14) puis (15) ou (17). De plus, afin d'éviter les erreurs numériques dues à l'usage de nombres flottants simple précision, nous seuillons $v_j(f,n)$ et régularisons $\mathbf{R}_j(f)$ après chaque mise à jour.

L'algorithme dnnsep est illustré dans la Figure 4. Après convergence, les images $\widehat{\mathbf{c}}_j(t)$ dans le domaine temporel sont obtenues par STFT inverse.

3.3 Évaluation

Nous avons testé dnnsep sur le corpus Demixing Secret Dataset (DSD100) utilisé pour la campagne SiSEC 2016, qui comprend 100 chansons complètes de genres musicaux et artistes divers. Le corpus est divisé en un ensemble de développement (utilisé pour l'apprentissage des DNNs) et un ensemble d'évaluation (utilisé pour le test).

La Figure 5 montre l'impact positif de la pondération (17) sur la performance, pour 1 seule itération de l'algorithme EM. Sans cette pondération, la qualité de séparation décroirait au fil des mises à jour . La mise à jour (14) (et non sa version simplifiée) reste toutefois nécessaire.

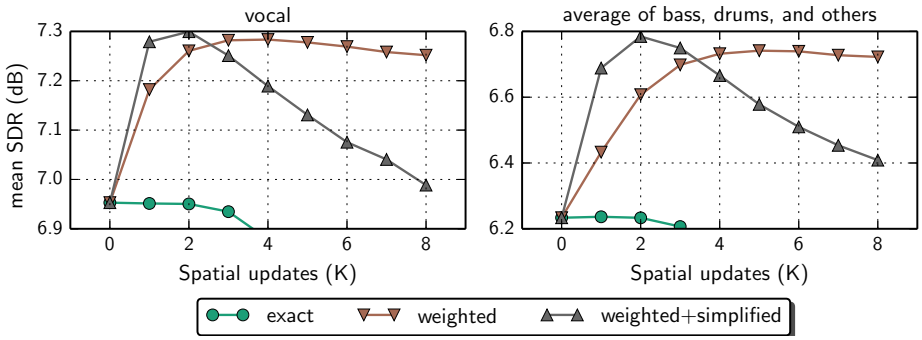

FIGURE 5 – Comparaison des mises à jour de $\mathbf{R}_j(f)$ sur l'ensemble de développement en fonction du nombre de mises à jour effectuées.

La Figure 6 compare diverses variantes de dnnsep (`NUG1` pondéré simplifié 1 itération, `NUG2` pondéré 1 itération, `NUG3` pondéré simplifié 2 itérations, `NUG4` pondéré 2 itérations) à plusieurs algorithmes de l'état de l'art : `OZE`, `DUR` et `HUA` sont basés sur la NMF, `KAM{1,2}` sont des variantes de KAM [5], `RAF{1,2,3}` des variantes de l'algorithme REPET et `UHL{1,2}` des variantes de l'algorithme [12] basé sur les DNNs. La performance est évaluée de façon classique en terme de rapport signal-à-distorsion global (SDR), rapport signal-à-interférence (SIR), rapport image-à-distorsion spatiale (ISR) et rapport signal-à-artefacts (SAR) pour le signal de voix chantée. dnnsep est considérablement meilleur que les méthodes basées sur la NMF, KAM ou REPET et légèrement meilleur que l'autre algorithme basé sur les DNNs, qui n'utilise pas l'information multicanal.

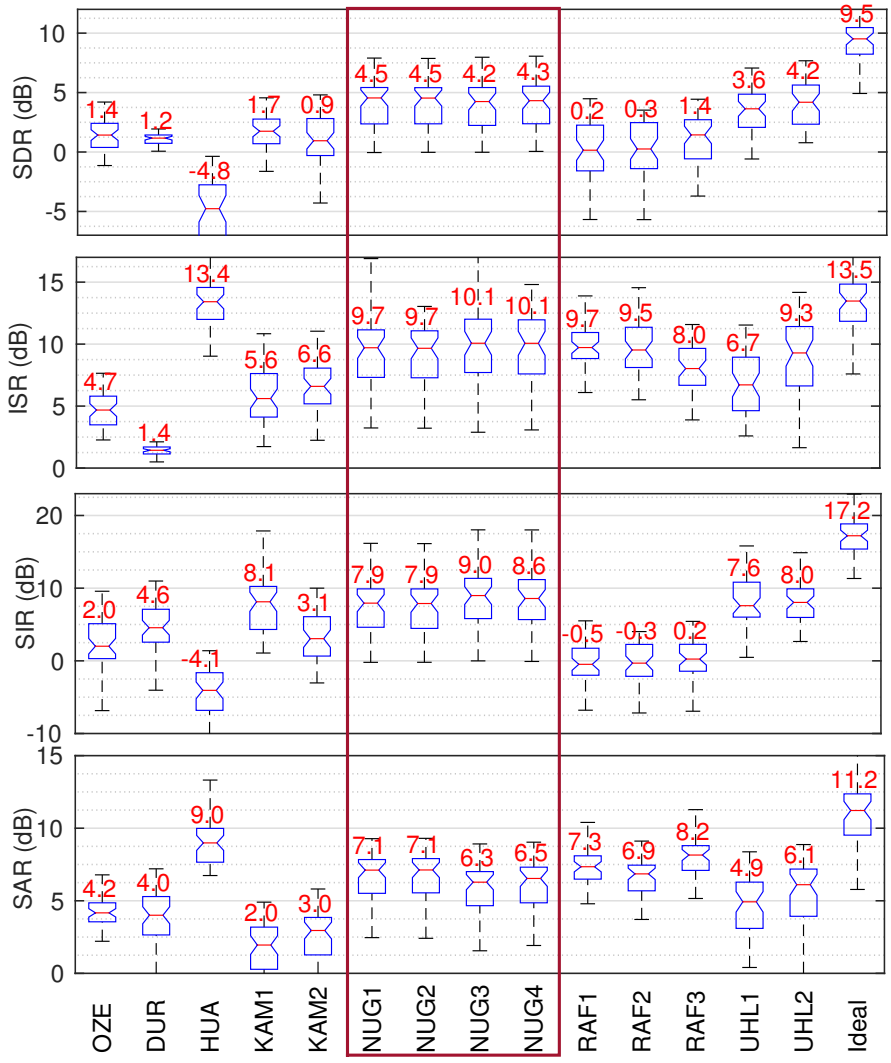

FIGURE 6 – Performance de séparation de la voix chantée sur l'ensemble de test. Les nombres en rouge indiquent la valeur médiane.

Références

[1] Ngoc Q. K. Duong, Emmanuel Vincent, and Rémi Gribonval. Under-determined reverberant audio source separation using a full-rank spatial covariance model. 18(7) :1830–1840, 2010.

[2] Derry Fitzgerald, Antoine Liutkus, and Roland Badeau. Projection-based demixing of spatial audio. *IEEE Transactions on Audio, Speech*

and Language Processing, May 2016.

[3] Po-Sen Huang, Minje Kim, Mark Hasegawa-Johnson, and Paris Smaragdis. Joint optimization of masks and deep recurrent neural networks for monaural source separation. 23(12) :2136–2147, 2015.

[4] Elias K. Kokkinis, Joshua D. Reiss, and John Mourjopoulos. A Wiener filter approach to microphone leakage reduction in close-microphone applications. *IEEE Transactions on Audio, Speech and Language Processing*, 20(3) :767–779, 2012.

[5] A. Liutkus, D. Fitzgerald, Z. Rafii, B. Pardo, and L. Daudet. Kernel additive models for source separation. *IEEE Transactions on Signal Processing*, June 2014.

[6] Antoine Liutkus, Jean-Louis Durrieu, Laurent Daudet, and Gaël Richard. An overview of informed audio source separation. In *WIAMIS*, pages 1–4, Paris, France, 2013.

[7] Antoine Liutkus and Emmanuel Vincent. Démixer la musique. *Interstices*, January 2016.

[8] Aditya Arie Nugraha, Antoine Liutkus, and Emmanuel Vincent. Multichannel audio source separation with deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(10), June 2016.

[9] Aditya Arie Nugraha, Antoine Liutkus, and Emmanuel Vincent. Multichannel music separation with deep neural networks. In *European Signal Processing Conference (EUSIPCO)*, Budapest, Hungary, August 2016.

[10] Thomas Prätzlich, Rachel Bittner, Antoine Liutkus, Juan Bello, and Meinard Müller. Interference reduction for multitrack music recordings. *IEEE Transactions on Audio, Speech and Language Processing*, 2016. submitted.

[11] Thomas Prätzlich, Rachel Bittner, Antoine Liutkus, and Meinard Müller. Kernel additive modeling for interference reduction in multichannel music recordings. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Brisbane, Australia, 2015.

[12] Stefan Uhlich, Franck Giron, and Yuki Mitsufuji. Deep neural network based instrument extraction from music. pages 2135–2139, 2015.

[13] E. Vincent, N. Bertin, R. Gribonval, and F. Bimbot. From blind to guided audio source separation : How models and side information can improve the separation of sound. *IEEE Signal Processing Magazine*, 31(3) :107–115, 2014.