Kernel additive modeling for interference reduction in multi-channel music recordings
Thomas Prätzlich, Rachel Bittner, Antoine Liutkus, Meinard Müller

To cite this version:
Thomas Prätzlich, Rachel Bittner, Antoine Liutkus, Meinard Müller. Kernel additive modeling for interference reduction in multi-channel music recordings. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Apr 2015, Brisbane, Australia. <hal-01116686v2>

HAL Id: hal-01116686
https://hal.inria.fr/hal-01116686v2
Submitted on 18 Feb 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
1. INTRODUCTION

When recording musicians during a live performance, sound engineers typically set a microphone for each voice, which may be an instrument or an instrument group such as violin or brass sections in the case of an orchestra. Recordings are often made with the musicians playing together in the same room. See Figure 1 for an illustration. In a typical professional setup, the recording room is equipped with sound absorbing materials and acoustic shields to isolate all the voices as much as possible. However, complete acoustic isolation between the voices is often not possible. In practice and as depicted in Figure 1, each microphone not only records sound from its dedicated voice, but also from all others in the room, resulting in recordings that do not feature isolated signals, but rather mixtures of a predominant voice with all others being audible through what is referred to as interference, bleeding, crosstalk, or leakage. Such interferences are annoying in practice for several reasons. First, interferences greatly reduce the mixing possibilities for a sound engineer, and second, they prevent the removal or isolation of a voice from the recording, which may be desirable, e.g., for pedagogical reasons. A natural question thus arises: is it possible to remove these interferences to get clean, isolated voice signals?

In the past, several studies from the literature have considered this interference reduction problem. In each of these studies, it is assumed that for each voice there is at least one microphone present and that the number of voices and their corresponding microphones are known. While some approaches are based on echo cancellation and adaptive filtering involving estimation of propagation filters between voices and microphones in the time domain [1, 2], others are based on Time-Frequency (TF) approaches, where the recordings are processed using adequate representation such as the Short Term Fourier Transform (STFT). For the latter cases, interference reduction is typically performed through Wiener filtering [3, 4], which has been shown to produce very good results in terms of sound quality at a small computational cost. Interference reduction using Wiener filtering requires an estimate of the clean spectrogram for each voice. While [3] simply assumes that the spectrogram of each recording is already a good estimate for its dedicated voice signal, [4, 5] introduce further temporal constraints on the voices so as to better identify them from the mixture recordings. In [6], Non-Negative Matrix Factorization (NMF, see [7]) is used as a global parametric model for the voice spectrograms.

Interference Reduction (IR) is closely related to the problem of audio source separation, in which the objective is to separate a sound mixture into its constituent components [8]. Audio source separation in general is a very difficult problem where performance is highly dependent on the signals considered. However, recent studies demonstrate that separation methods can be very effective if prior
Let KAMIR to be computationally efficient while maintaining good performance. Furthermore, we introduce an approximation that permits us to exploit this strong prior information. The main contribution of this separation is made by Kernel Additive Modeling (KAM, [10, 11, 12]) to achieve both computational efficiency and did not harm the results. Thus, KAMIR is effectively a generalization of several previous methods; notably, [3] can be understood as a single iteration of KAMIR. Furthermore, we introduce an approximation that permits KAMIR to be computationally efficient while maintaining good performance.

The remainder of the paper is organized as follows: In Section 2 we describe the KAMIR model and the corresponding algorithm. In Section 3, we evaluate KAMIR through a perceptual evaluation test.

2. MODEL AND METHOD

2.1. Notations and model

Let $J$ be the number of voices and $I$ be the number of microphones. We will refer to the signal $x_i$ with $i \in \{1, \ldots, I\}$ from a microphone as a channel. In full generality, the sound produced by a particular voice $j \in \{1, \ldots, J\}$ can be heard in all channels $x_i$. For this reason, we define the image $y_{ij}$ as the contribution of the sound coming from voice $j$ in channel $i$. Channel $x_i$ is then taken as the sum of the corresponding voice images: $x_i = \sum_{j=1}^{J} y_{ij}$. In the proposed technique, we first compute the Short-Term Fourier Transform (STFT) $X_i$ of each channel $x_i$. This yields $I$ matrices of dimension $N_w \times N_t$, where $N_w$ and $N_t$ are the number of frequency bands and the number of frames, respectively. Each point $(\omega, t)$ is called a Time-Frequency (TF) bin. We have:

$$X_i(\omega, t) = \sum_{j=1}^{J} Y_{ij}(\omega, t), \quad \text{(1)}$$

where $Y_{ij}$ denotes the $N_w \times N_t$ STFT of $y_{ij}$.

Following previous work in source separation literature (see e.g. [13]), we assume that all TF bins of the audio signals are independent. Furthermore, we will also make the assumption that the images $\{Y_{ij}\}$ are independent for all $i$ and $j$. This first comes from the common assumption that the voice signals are independent, as they are produced by different physical instruments and acoustic processes. Second, independence across channels effectively boils down to discarding any phase dependencies. Even if it is arguable, we found in practice that discarding phase modeling in our case was both computationally efficient and did not harm the results.

Finally, we choose to model the signals using a Local Gaussian Model (LGM, [13, 14, 15]). In addition to assuming that the TF bins are independent, it assumes that each $Y_{ij}(\omega, t)$ is distributed with respect to a complex isotropic Gaussian distribution:

$$Y_{ij}(\omega, t) \sim \mathcal{N}_c(0, \sigma^2_{ij}(\omega, t)) \quad \text{,} \quad \text{(2)}$$

where the parameter $\sigma^2_{ij}(\omega, t)$ is called the Power Spectral Density (PSD) of $y_{ij}$. Finally, we assume that the PSDs of each voice are equivalent up to a scaling factor $\lambda_{ij}(\omega)$:

$$\sigma^2_{ij}(\omega, t) = \lambda_{ij}(\omega) P_j(\omega, t) \approx \{Y_{ij}(\omega, t)\}^2, \quad \text{(3)}$$

where $P_j(\omega, t) \geq 0$ is the latent PSD of voice $j$ shared across all channels $i$. The scalar $\lambda_{ij}(\omega)$ gives the amount of interference of voice $j$ into channel $i$ at frequency band $\omega$. For this reason, we define the frequency dependent interference matrix as $\Lambda(\omega) = [\lambda_{ij}(\omega)]_{i,j}$. Note that this model amounts to discarding phase dependencies between the different channels, while relating them through a common latent PSD for each voice.

2.2. Separation method

Because $X_i(\omega, t)$ is the sum of $J$ independent complex isotropic Gaussian random variables $Y_{ij}(\omega, t)$ as in (1)), it is itself distributed as a complex isotropic Gaussian:

$$X_i(\omega, t) \sim \mathcal{N}(0, \sum_{j=1}^{J} \lambda_{ij}(\omega) P_j(\omega, t)) \quad \text{,} \quad \text{(4)}$$

If the PSD $P_j$ is known for each $j$, and the frequency-dependent interference matrices $\Lambda(\omega)$ are known, it has been shown [13, 14, 15] that the Minimum Mean-Squared Error (MMSE) estimate $\hat{Y}_{ij}(\omega, t)$ of any $Y_{ij}(\omega, t)$ is given by generalized Wiener filtering:

$$\hat{Y}_{ij}(\omega, t) = \frac{\lambda_{ij}(\omega) P_j(\omega, t)}{\sum_{j' \neq j} \lambda_{ij'}(\omega) P_{j'}(\omega, t)} X_i(\omega, t) \quad \text{,} \quad \text{(5)}$$

where $\hat{\phi}$ denotes a definition and $W_{ij}(\omega, t)$ is called the Wiener gain. However, for a given voice $j$, we are usually not interested in estimating $\hat{Y}_{ij}$ for all recordings $i$. Rather, we want to obtain $Y_{ij}$ only for those microphone channels that were initially positioned to capture this voice. We define the channel selection function $\varphi(j) \subseteq \{1, \ldots, I\}$ that indicates which images of voice $j$ we want to recover. This is to account for possibly complex scenarios where several microphones are assigned to a single voice signal, as in the case of a concert piano. In the following, the channel selection function $\varphi$ is assumed to be known and given by the user (typically a sound engineer) who knows the recording setup and can easily provide this information. Furthermore, we assume that $\varphi(j) \neq \emptyset$ for all $j = 1, \ldots, J$, meaning that each voice is predominant in at least one microphone channel.

Thus, the objective of interference reduction is to estimate all $\{\hat{Y}_{ij}\}_{i \in \varphi(j)}$ for each voice $j$. The resulting waveforms are easily recovered through an inverse STFT. Note that by selecting the “most relevant” images for a specific voice, we also significantly reduce computation time.

2.3. Parameter estimation algorithm

In this section, we describe the KAMIR algorithm for Interference Reduction based on KAM [10, 11, 12]. As input, it takes the STFTs $X_i$ of the recorded signals and the channel selection function $\varphi$, as described in the preceding section. It returns estimates for the desired clean signals $\{\hat{y}_{ij}\}_{i \in \varphi(j)}$ for each voice $j$. To do this, it only needs to estimate the parameters for the Wiener filter (5): the PSDs $P_j$ of the voices and the interference matrices $\Lambda(\omega)$.

In a nutshell, KAMIR alternates between two distinct procedures in an iterative fashion. In a first separation step, the current parameter fitting stage, those separated signals are kept fixed and...
are assumed to be good estimates. The parameters $P_j$ and $\Lambda(\omega)$ are then re-estimated. Finally, the whole procedure is repeated until a stopping criterion is met, which is usually the imposed number of iterations. The KAMIR algorithm is summarized in Algorithm 1. We now discuss initialization and re-estimation of the parameters $P_j$ and $\Lambda(\omega)$.

**Initialization (Algorithm Step 2).** The elements of the interference matrices are initialized with:

$$\forall (i,j,\omega), \lambda_{ij}(\omega) = \begin{cases} 1 & \text{if } i \in \varphi(j) \\ \rho & \text{otherwise}, \end{cases} \tag{6}$$

where the parameter $\rho \in [0,1]$ is called the minimal interference, and corresponds to the minimum amount of interference we expect in any channel. The rationale for this initialization is the following: a voice $j$ which is associated to channel $i$ is given the maximum interference value $\rho$ as it should ideally be fully captured in this channel; a voice $j$ which is not associated to channel $i$ is given the value $\rho$ as it should be only minimally captured in this channel. See Figure 2a for an example initialization of an interference matrix. Figure 2b shows the average interference matrix $\bar{\Lambda} = \frac{1}{J} \sum\Lambda(\omega)$ for an example with $I = 21$ channels and $J = 11$ voices (a) upon initialization, (b) after learning.

At this point, we can optionally apply a 2-dimensional median filter on $P_j$, as in KAM [10, 11, 12] so as to enforce any knowledge we may know concerning a voice’s local regularities. To do this, a user must provide a 2-D binary kernel $k_j$, with which $P_j$ is filtered. See the aforementioned references for examples of the choice of adequate kernels in audio.

**Interference matrix $\Lambda(\omega)$ updates (Algorithm Step 3c).** According to our probabilistic model, the observed noisy channels $\{X_i(\omega,t)\}_{i,\omega,t}$ are independent and distributed according to (4). Given $P_j$, estimating each row $\Lambda_i(\omega) = [\lambda_{i1}(\omega), \ldots, \lambda_{iJ}(\omega)]$ of the matrix $\Lambda(\omega)$ can be done through the classical NMF methodology [7], as:

$$\Lambda_i(\omega) \approx \arg\min_{\lambda_{ij}(\omega) \geq 0} \sum_{t=1}^{N_t} d_\beta \left( \sum_{j=1}^{J} r_j P_j(\omega,t) \right)^2, \tag{8}$$

where $d_\beta$ stands for any appropriate cost-function such as the popular family of $\beta$-divergences, which notably includes Kullback-Leibler ($\beta = 1$) and Itakura-Saito ($\beta = 0$). The main idea of using NMF here is to update $\Lambda(\omega)$ in a multiplicative fashion, so as to enforce nonnegativity. It can be shown that the corresponding update rule for each matrix element $\lambda_{ij}(\omega)$ is:

$$\lambda_{ij}(\omega) \leftarrow \lambda_{ij}(\omega) \cdot \frac{\sum_{t=1}^{N_t} \bar{Y}_{ij}(\omega,t)^{\beta-2} V_i(\omega,t) P_j(\omega,t)}{\sum_{t=1}^{N_t} \bar{V}_i(\omega,t)^{\beta-2} P_j(\omega,t)}. \tag{9}$$

where $V_i(\omega,t) \triangleq |X_i(\omega,t)|^2$ and $\bar{V}_i(\omega,t) \triangleq \sum_{\lambda_{ij}(\omega)} P_j(\omega,t)$. Note that $P_j$ is kept fixed during the NMF updates. After $\Lambda(\omega)$ has been updated, its entries are normalized $^1$ to the interval $[0,1]$. To achieve this, an easy procedure is to first re-scale $P_j(\omega,t)$ by:

$$P_j(\omega,t) \leftarrow P_j(\omega,t) \frac{\lambda_{ij}(\omega)}{\sum_{i=1}^{J} \lambda_{ij}(\omega)}, \tag{10}$$

and then set:

$$\lambda_{ij}(\omega) \leftarrow \max \left[ \rho, \frac{\lambda_{ij}(\omega)}{\sum_{i=1}^{J} \lambda_{ij}(\omega)} \right]. \tag{11}$$

An example of a learned interference matrix is shown in Figure 2b.

**Algorithm 1: KAMIR**

1. **Input:**
   - $X_i(\omega,t)$ for each channel $x_i$
   - Channel selection function $\varphi(j)$ for each voice $j$
   - Minimal interference $\rho$
   - Optional: Kernels $k_j$ for each voice $j$

2. **Initialization**
   - For each $\omega$, initialize $\Lambda(\omega)$ as in (6)
   - For each $j$, for each $i \in \varphi(j)$, $\bar{Y}_{ij}(\omega,t) \leftarrow X_i$

3. **Parameter fitting step**
   - (a) For each $j$: update $P_j$ as in (7)
   - (b) Optional: For each $j$, apply median filter on $P_j$ with kernel $k_j$
   - (c) For each $\omega$, update $\Lambda(\omega)$ using (9)
   - (d) Re-scale $P_j$ with (10) and normalize $\Lambda$ with (11).

4. **Separation step**
   - For each $j$, for each $i \in \varphi(j)$, update $Y_{ij}(\omega,t)$ as in (5)

5. **For another iteration, return to step (3)**

6. **Output:** $Y_{ij}(\omega,t)$ for each $j$, for each $i \in \varphi(j)$

---

$^1$All results reported in Section 3 use the normalization as in Equations (10) and (11). However, we later found that normalizing such that the total weight for each channel equals one (by skipping Equation (10) and changing the denominator in Equation (11) to $\sum_{j=1}^{J} \lambda_{ij}(\omega)$), leads to more reasonable results in $\Lambda$. We suggest that this alternate normalization be used when implementing the algorithm.
2.4. Computationally Effective Approximation

If no kernels $k_i$ are provided to filter the PSD estimates $P_i$ at step 3b of KAMIR, we observe that the Wiener gains $W_{ij}$ in (5) evolve in a very particular fashion during the iterations of KAMIR. To illustrate this, we display the Wiener gains at iterations 2–4 as a function of their value at iteration 1 as scatter plots in Figure 3. We see that there is a clear tendency towards binarization of the Wiener gains with increasing iterations. This can effectively be modeled as applying a sigmoidal function $g$ to the initial Wiener masks $W_{ij}$, as demonstrated by the bold curves in Figure 3. The sigmoidal gain function $g$ is given by:

$$g(x) = 1 - \frac{1}{1 + \exp(s \cdot (x - \tau))},$$

where $s$ and $\tau$ are slope and a threshold parameters, respectively. The slope of the sigmoid function approximates the number of iterations, and $\tau$ is related to the minimal interference $\rho$.

Hence, to simulate applying several iterations of KAMIR, instead of actually making the computations, we can simply replace $W_{ij}(\omega, t)$ in (5) by $g(W_{ij}(\omega, t))$. In this way, only one iteration is needed to update the inference matrices $\Lambda(\omega)$ and proceed to interference reduction. Note that this approximation is only valid if no kernels $k_i$ are provided for median filtering the PSD estimates. This is because median filtering induces dependencies between different TF bins, causing the sigmoidal approximation to become a poor model.

3. EVALUATION

We compared KAMIR’s performance for five different parameter settings with the algorithm from [4] on an orchestra recording featuring 21 microphones with up to 11 voices. Throughout the experiments, we used $\beta = 0$ and omitted the kernel median filtering (Algorithm Step 3b). Because the recording session used a live setup featuring interference in all microphones, the standard evaluation metrics for blind source separation [17, 18] were not applicable, as they require a clean reference signal. Instead, we evaluated the performance of each setting in a perceptual study involving 21 listeners using five different microphone signals (items). Listeners gave two ratings for each item and setting: first, a rating of “how well the reduction of interference was accomplished” (Figure 4a), and second, a rating for the overall quality, considering the success of both the interference reduction and the sound quality (Figure 4b).

As can be seen in Figure 4b, the Kokkinis 2012 algorithm [4] has the tendency to perform slightly better in terms of overall sound quality, especially for items #1 and #4. However, we also see that the different versions of KAMIR perform favorably as well, especially when the model benefits from several microphones for one voice signal. For example, for item #5, the three microphones used for the cello section lead to improvements in the interference reduction without a reduction in the overall quality. See the parameter settings $K3$ and $K4$ compared to the approaches in Figure 4a and b.

Another interesting observation is that applying several iterations and learning the interference matrices $\Lambda(\omega)$ improves the interference reduction of KAMIR in certain scenarios (see items #2, #4, #5 with the parameter settings $K3$, and $K4$ compared with $K2$ in Figure 4a). However, items #1, #3, #4 in Figure 4b indicate an inverse tendency for the overall quality ratings of $K2$, $K3$ and $K4$. This is not surprising, because by suppressing more interferences, artifacts or distortions are more likely to be introduced into the signals. Finally, we want to note that the computationally effective approximation $K5$ performs similarly to the other parameter settings. The sound material of the listening test and an implementation of the algorithm can be found at [19, 20].

4. CONCLUSIONS

In this paper, we have proposed a simple, yet effective algorithm called KAMIR for interference reduction in multi-channel live recordings. It is based on iteratively estimating the spectrograms of the desired clean voices and the interference in the microphone channels. Our evaluation indicates that in addition to being very simple to implement, KAMIR produces separated voices that have good perceptual quality. It furthermore provides a practical way to trade-off interference reduction and distortion, notably through the number of iterations. Finally, we have shown that in some cases, the impact of several iterations of KAMIR can be predicted from the result of its first iteration, yielding a computationally effective technique for interference reduction. Future work includes evaluation of the use of kernels within the KAMIR algorithm.

---

2. We want to thank Elias Kokkinis who has provided the code for his algorithm and a parameter setting specifically tuned for this recording.

3. The recording is an excerpt of the opera “Der Freischütz” recorded by the project Freischütz Digital (www.freischuetz-digital.de) [16].
5. REFERENCES


