HAL
open science

A variance modeling framework based on variational autoencoders for speech enhancement

Simon Leglaive, Laurent Girin, Radu Horaud

To cite this version:

Simon Leglaive, Laurent Girin, Radu Horaud. A variance modeling framework based on variational autoencoders for speech enhancement. MLSP 2018 - IEEE 28th International Workshop on Machine Learning for Signal Processing, Sep 2018, Aalborg, Denmark. pp.1-6, 10.1109/MLSP.2018.8516711 . hal-01832826

HAL Id: hal-01832826

https://inria.hal.science/hal-01832826

Submitted on 12 Jul 2018

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.

A VARIANCE MODELING FRAMEWORK BASED ON VARIATIONAL AUTOENCODERS FOR SPEECH ENHANCEMENT

Simon Leglaive[1] *Laurent Girin*[1,2] *Radu Horaud*[1]

[1]Inria Grenoble Rhône-Alpes, France [2]Univ. Grenoble Alpes, Grenoble INP, GIPSA-lab, France

ABSTRACT

In this paper we address the problem of enhancing speech signals in noisy mixtures using a source separation approach. We explore the use of neural networks as an alternative to a popular speech variance model based on supervised non-negative matrix factorization (NMF). More precisely, we use a variational autoencoder as a speaker-independent supervised generative speech model, highlighting the conceptual similarities that this approach shares with its NMF-based counterpart. In order to be free of generalization issues regarding the noisy recording environments, we follow the approach of having a supervised model only for the target speech signal, the noise model being based on unsupervised NMF. We develop a Monte Carlo expectation-maximization algorithm for inferring the latent variables in the variational autoencoder and estimating the unsupervised model parameters. Experiments show that the proposed method outperforms a semi-supervised NMF baseline and a state-of-the-art fully supervised deep learning approach.

***Index Terms*—** Audio source separation, speech enhancement, variational autoencoders, non-negative matrix factorization, Monte Carlo expectation-maximization

1. INTRODUCTION

Speech enhancement is a classical problem of speech processing, which aims to recover a clean speech signal from the recording of a noisy signal, where the noise is generally considered as additive [1]. In this work we address single-channel speech enhancement, which can be seen as a an under-determined source separation problem, where the sources to be separated are of different nature.

Statistical approaches combining a local Gaussian modeling of the time-frequency signal coefficients with a variance model have attracted a lot of attention in the past few years [2]. Within this framework, non-negative matrix factorization (NMF) techniques are especially popular to structure the time-frequency-dependent signal variance [3]. Recently, deep neural networks were also successfully investigated for speech enhancement, to generate either time-frequency masks or clean power spectrograms from noisy spectrograms [4], or to model the signal variance [5] (here in a multichannel framework). Similarities between NMF and standard autoencoders have been reported in [6]. The recent approach [7] also falls into the variance modeling framework. Very interestingly, the authors developed a semi-supervised single-channel speech enhancement method mixing concepts of Bayesian inference and deep learning, through the use of a variational autoencoder [8].

In the present work, we follow a similar approach, using a speaker-independent supervised Gaussian speech model based on a variational autoencoder. For a given time-frequency point, it mainly consists in modeling the speech variance as a non-linear function of a Gaussian latent random vector, by means of a neural network. Compared with [7], we further introduce a time-frame-dependent gain in order to provide some robustness with respect to the loudness of the training examples. The noise model is Gaussian and based on unsupervised NMF, so as to be free of generalization issues regarding the noisy recording environments. We propose a new Monte Carlo expectation-maximization algorithm [9] for performing semi-supervised speech enhancement. Experimental results show the superiority of our approach compared with a baseline method using NMF, and also compared with a state-of-the-art fully supervised deep learning approach [10].

We first provide in Section 2 a brief presentation of the NMF-based variance modeling framework commonly used in audio source separation. This will help us emphasizing the conceptual similarities shared with the speech model used in this work. In Section 3 we present the model. Our inference algorithm is detailed in Section 4. The optimization problem related to the variational autoencoder training is presented in Section 5. Experiments are described in Section 6 and we finally conclude in Section 7.

2. VARIANCE MODELING FRAMEWORK

We work in the short-term Fourier transform (STFT) domain. For all $(f,n) \in \mathbb{B} = \{0,...,F-1\}\times\{0,...,N-1\}$, where f denotes the frequency index and n the time-frame index, the single-channel speech enhancement problem consists in recovering the clean speech STFT coefficients $s_{fn} \in \mathbb{C}$ from the noisy observed ones $x_{fn} = s_{fn} + b_{fn}$, where $b_{fn} \in \mathbb{C}$ is a noise signal. The source STFT coefficients are modeled as complex circularly symmetric Gaussian random variables [2]:

$$s_{fn} \sim \mathcal{N}_c(0, v_{s,fn}); \qquad b_{fn} \sim \mathcal{N}_c(0, v_{b,fn}), \tag{1}$$

where $\mathcal{N}_c(\mu, \sigma^2)$ denotes the complex proper Gaussian distribution, which is circularly symmetric (rotational invariant in the complex plane) if $\mu = 0$. Under a local stationarity assumption [11], the variances in (1) characterize the short-term power spectral density of the audio source signals. Note that this Gaussian model has been recently extended to other complex circularly symmetric and elliptically contoured distributions [12], such as the symmetric alpha-stable [13] or the Student's t [14] distributions.

A widely used linear variance model for the source signals relies on NMF. For $j \in \{s,b\}$, let $\mathbf{W}_j \in \mathbb{R}_+^{F\times K_j}$ be a matrix containing K_j spectral patterns, and let $\mathbf{H}_j \in \mathbb{R}_+^{K_j\times N}$ be a matrix containing the activations of these spectral patterns over the time frames. At each time-frequency point $(f,n) \in \mathbb{B}$, the variance $v_{j,fn}$ is modeled as follows:

$$v_{j,fn} = (\mathbf{W}_j\mathbf{H}_j)_{f,n} = \mathbf{w}_{j,f}^{\top}\mathbf{h}_{j,n}, \tag{2}$$

This work is supported by the ERC Advanced Grant VHIA #340113.

where $\mathbf{w}_{j,f}^{\top} \in \mathbb{R}_+^{K_j}$ is f-th row of $\mathbf{W}_j$, $\mathbf{h}_{j,n} \in \mathbb{R}_+^{K_j}$ is n-th column of $\mathbf{H}_j$, and $\cdot^{\top}$ denotes the transposition operator. Without any additional constraint on the model parameters, the NMF rank K_j is generally chosen such that $K_j(F+N) \ll FN$. Compared with the unconstrained Gaussian model in (1), the number of parameters to be estimated is therefore considerably reduced, which is of great interest for solving an under-determined problem.

In a semi-supervised speech enhancement setting [15], $\mathbf{W}_s$ is learned on a clean speech database. The variance model in (2) is therefore seen as a linear function of the activations. Then, in the speech enhancement algorithm, only the speech activation matrix $\mathbf{H}_s$ and the noise NMF parameters $\mathbf{W}_b$ and $\mathbf{H}_b$ are estimated from the observation of the noisy mixture signal. It has been shown in [3] that maximum likelihood estimation of the NMF parameters under this Gaussian model amounts to solving an optimization problem involving the Itakura-Saito divergence. An optimal estimate of the speech signal in the minimum mean square error sense is then obtained by Wiener filtering.

3. MODEL

3.1. Supervised Speech Model

In a supervised setting, the variance model in (2) is limited by its linear nature. Moreover, the number of trainable parameter is restricted by the factorization rank, which is usually chosen to be small. In this work we consider a supervised and non-linear modeling of the speech variances. Independently for all $(f,n) \in \mathbb{B}$, we have the following generative model, involving a latent random vector $\mathbf{z}_n \in \mathbb{R}^L$:

$$\mathbf{z}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{I}); \tag{3}$$

$$s_{fn} \mid \mathbf{z}_n; \boldsymbol{\theta}_s \sim \mathcal{N}_c(0, \sigma_f^2(\mathbf{z}_n)), \tag{4}$$

where $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the multivariate Gaussian distribution for a real-valued random vector, $\mathbf{I}$ is the identity matrix of appropriate size, and $\sigma_f^2 : \mathbb{R}^L \mapsto \mathbb{R}_+$, $f \in \{0, ..., F-1\}$, represents a non-linear function parametrized by $\boldsymbol{\theta}_s$. The output of this function is provided by one neuron in the F-dimensional output layer of a neural network, whose weights and biases are denoted by $\boldsymbol{\theta}_s$. In the context of variational autoencoders, the model (3)-(4) is called the *generative network* or *probabilistic decoder* [8].

As in the case of an NMF-based supervised variance model, we assume that the parameters $\boldsymbol{\theta}_s$ are learned on a clean speech database. This is done using a *recognition network* or *probabilistic encoder* in addition to the probabilistic decoder, as will be explained in Section 5. Only the latent random vectors $\{\mathbf{z}_n\}_{n=0}^{N-1}$ will have to be inferred from the observed data. Moreover, the latent dimension L is set much smaller than F.

3.2. Unsupervised Noise Model

As introduced in Section 2, we use an unsupervised NMF-based Gaussian model for the noise. Independently for all $(f,n) \in \mathbb{B}$:

$$b_{fn}; \mathbf{w}_{b,f}, \mathbf{h}_{b,n} \sim \mathcal{N}_c\left(0, (\mathbf{W}_b\mathbf{H}_b)_{f,n}\right). \tag{5}$$

3.3. Mixture Model

The observed mixture signal is modeled as follows for all $(f,n) \in \mathbb{B}$:

$$x_{fn} = \sqrt{g_n} s_{fn} + b_{fn}, \tag{6}$$

where $g_n \in \mathbb{R}_+$ represents a frame-dependent but frequency-independent gain. We introduce this term in order to provide some robustness with respect to the loudness of the training examples used to learn the variational autoencoder parameters $\boldsymbol{\theta}_s$ (this will be illustrated in Section 6). The speech and noise signals are further supposed to be mutually independent given the latent random vectors $\{\mathbf{z}_n\}_{n=0}^{N-1}$, such that for all $(f,n) \in \mathbb{B}$:

$$x_{fn} \mid \mathbf{z}_n; \boldsymbol{\theta}_s, \boldsymbol{\theta}_{u,fn} \sim \mathcal{N}_c\left(0, g_n\sigma_f^2(\mathbf{z}_n) + (\mathbf{W}_b\mathbf{H}_b)_{f,n}\right), \tag{7}$$

where $\boldsymbol{\theta}_{u,fn} = \{\mathbf{w}_{b,f}, \mathbf{h}_{b,n}, g_n\}$ is the set of unsupervised model parameters at time-frequency point (f,n).

4. INFERENCE

Let us introduce the following notations:

- ▷ $\mathbf{x}_n = \{x_{fn}\}_{f=0}^{F-1}$: The set of mixture STFT coefficients for a given time frame $n \in \{0, ..., N-1\}$;
- ▷ $\mathbf{x} = \{x_{fn}\}_{(f,n)\in\mathbb{B}}$: The set of all mixture STFT coefficients;
- ▷ $\mathbf{z} = \{\mathbf{z}_n\}_{n=0}^{N-1}$: The set of all latent variables;
- ▷ $\boldsymbol{\theta}_u = \{\boldsymbol{\theta}_{u,fn}\}_{(f,n)\in\mathbb{B}}$: The set of unsupervised model parameters.

From the set of observations $\mathbf{x}$, our primary goal is to estimate the unsupervised model parameters $\boldsymbol{\theta}_u$, which will serve in the final inference of the speech STFT coefficients. Unfortunately, straightforward maximum likelihood estimation is here intractable. A common alternative then consists in exploiting the latent variable structure of the model to derive an expectation-maximization (EM) algorithm [16]. From an initialization $\boldsymbol{\theta}_u^{\star}$ of the model parameters, it consists in iterating the two following steps until convergence:

- ▷ E-Step: Compute $Q(\boldsymbol{\theta}_u; \boldsymbol{\theta}_u^{\star}) = \mathbb{E}_{p(\mathbf{z}|\mathbf{x};\boldsymbol{\theta}_s,\boldsymbol{\theta}_u^{\star})}[\ln p(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta}_s, \boldsymbol{\theta}_u)]$;
- ▷ M-Step: Update $\boldsymbol{\theta}_u^{\star} \leftarrow \arg\max_{\boldsymbol{\theta}_u} Q(\boldsymbol{\theta}_u; \boldsymbol{\theta}_u^{\star})$.

4.1. E-Step

Due to the non-linear relation between the observations and the latent variables (7), we cannot compute the posterior distribution $p(\mathbf{z} \mid \mathbf{x}; \boldsymbol{\theta}_s, \boldsymbol{\theta}_u^{\star})$ in an analytical form. Therefore we cannot compute the expectation involved in the definition of $Q(\boldsymbol{\theta}_u; \boldsymbol{\theta}_u^{\star})$ at the E-Step. We consequently approximate it by the following empirical average (Monte Carlo approximation):

$$\begin{aligned} Q(\boldsymbol{\theta}_u; \boldsymbol{\theta}_u^{\star}) &\approx \tilde{Q}(\boldsymbol{\theta}_u; \boldsymbol{\theta}_u^{\star}) \\ &\overset{c}{=} -\frac{1}{R}\sum_{r=1}^{R}\sum_{(f,n)\in\mathbb{B}}\Bigg[\ln\left(g_n\sigma_f^2\left(\mathbf{z}_n^{(r)}\right) + (\mathbf{W}_b\mathbf{H}_b)_{f,n}\right) \\ &\qquad + \frac{|x_{fn}|^2}{g_n\sigma_f^2\left(\mathbf{z}_n^{(r)}\right) + (\mathbf{W}_b\mathbf{H}_b)_{f,n}}\Bigg], \end{aligned} \tag{8}$$

where $\overset{c}{=}$ denotes equality up to additive constants with respect to $\boldsymbol{\theta}_u$ and $\boldsymbol{\theta}_u^{\star}$, and $\{\mathbf{z}_n^{(r)}\}_{r=1,...,R}$ is a sequence of samples drawn from the posterior $p(\mathbf{z}_n \mid \mathbf{x}_n; \boldsymbol{\theta}_s, \boldsymbol{\theta}_u^{\star})$ using a Markov Chain Monte Carlo (MCMC) method. Here we use the Metropolis-Hastings algorithm [17]. This approach forms the basis of the Monte Carlo EM (MCEM) algorithm [9]. Note that unlike the standard EM algorithm, it does not ensure that the likelihood increases at each iteration. Nevertheless, some convergence results in terms of stationary point of the likelihood can be obtained under suitable conditions [18].

At the m-th iteration of the Metropolis-Hastings algorithm and independently for all $n \in \{0, ..., N-1\}$, we first draw a sample $\mathbf{z}_n$ from a proposal random walk distribution:

$$\mathbf{z}_n \mid \mathbf{z}_n^{(m-1)}; \epsilon^2 \sim \mathcal{N}(\mathbf{z}_n^{(m-1)}, \epsilon^2\mathbf{I}). \tag{9}$$

Using the fact that this is a symmetric proposal distribution [17], we then compute the acceptance probability α defined by:

$$\alpha = \min\left(1, \frac{p\left(\mathbf{x}_n \mid \mathbf{z}_n; \boldsymbol{\theta}_s, \boldsymbol{\theta}_u^\star\right) p\left(\mathbf{z}_n\right)}{p\left(\mathbf{x}_n \mid \mathbf{z}_n^{(m-1)}; \boldsymbol{\theta}_s, \boldsymbol{\theta}_u^\star\right) p\left(\mathbf{z}_n^{(m-1)}\right)}\right), \tag{10}$$

where $p\left(\mathbf{x}_n \mid \mathbf{z}_n; \boldsymbol{\theta}_s, \boldsymbol{\theta}_u^\star\right) = \prod_{f=0}^{F-1} p(x_{fn} \mid \mathbf{z}_n; \boldsymbol{\theta}_s, \boldsymbol{\theta}_{u,fn}^\star)$ with $p(x_{fn} \mid \mathbf{z}_n; \boldsymbol{\theta}_s, \boldsymbol{\theta}_{u,fn}^\star)$ defined in (7) and $p\left(\mathbf{z}_n\right)$ defined in (3). Then we draw u from the uniform distribution $\mathcal{U}([0,1])$. If $u < \alpha$, we accept the sample and set $\mathbf{z}_n^{(m)} = \mathbf{z}_n$, otherwise we reject the sample and set $\mathbf{z}_n^{(m)} = \mathbf{z}_n^{(m-1)}$. We only keep the last R samples for computing $\tilde{Q}(\boldsymbol{\theta}_u; \boldsymbol{\theta}_u^\star)$ in (8), i.e. we discard the samples drawn during a so called burn-in period.

4.2. M-Step

At the M-step of the MCEM algorithm, we want to maximize $\tilde{Q}(\boldsymbol{\theta}_u; \boldsymbol{\theta}_u^\star)$ in (8) with respect to the unsupervised model parameters $\boldsymbol{\theta}_u$. As usual in the NMF literature [19], we adopt a block-coordinate approach by successively and individually updating $\mathbf{H}_b$, $\mathbf{W}_b$ and $\mathbf{g} = [g_0, ..., g_{N-1}]^\top$, using the auxiliary function technique. We will provide derivation details only for $\mathbf{H}_b$, as the procedure is very similar for the other parameters.

Let $\mathcal{C}(\mathbf{H}_b)$ be the opposite of $\tilde{Q}(\boldsymbol{\theta}_u; \boldsymbol{\theta}_u^\star)$ in (8), seen as a function of $\mathbf{H}_b$ only (other parameters are fixed). Our goal is to minimize $\mathcal{C}(\mathbf{H}_b)$ under a non-negativity constraint.

Definition 1 (Auxiliary function)**.** The $\mathbb{R}_+^{K_b \times N} \times \mathbb{R}_+^{K_b \times N} \mapsto \mathbb{R}_+$ mapping $\mathcal{G}(\mathbf{H}_b, \tilde{\mathbf{H}}_b)$ is an auxiliary function to $\mathcal{C}(\mathbf{H}_b)$ if and only if

$$\forall (\mathbf{H}_b, \tilde{\mathbf{H}}_b) \in \mathbb{R}_+^{K_b \times N} \times \mathbb{R}_+^{K_b \times N}, \quad \mathcal{C}(\mathbf{H}_b) \leq \mathcal{G}(\mathbf{H}_b, \tilde{\mathbf{H}}_b); \tag{11}$$

$$\forall \mathbf{H}_b \in \mathbb{R}_+^{K_b \times N}, \quad \mathcal{C}(\mathbf{H}_b) = \mathcal{G}(\mathbf{H}_b, \mathbf{H}_b). \tag{12}$$

In other words, $\mathcal{G}(\mathbf{H}_b, \tilde{\mathbf{H}}_b)$ is an upper bound of $\mathcal{C}(\mathbf{H}_b)$ which is tight for $\tilde{\mathbf{H}}_b = \mathbf{H}_b$. The original minimization problem can be replaced with an alternate minimization of this upper bound. From an initial point $\mathbf{H}_b^\star$ we iterate:

$$\mathbf{H}_b^\star \leftarrow \underset{\mathbf{H}_b \in \mathbb{R}_+^{K_b \times N}}{\arg\min}\ \mathcal{G}(\mathbf{H}_b, \mathbf{H}_b^\star). \tag{13}$$

This procedure corresponds to the majorize-minimize (MM) algorithm [20], which by construction leads to a monotonic decrease of $\mathcal{C}(\mathbf{H}_b)$. Moreover, its convergence properties are the same as the ones of the EM algorithm [21].

Proposition 1 (Auxiliary function to $\mathcal{C}(\mathbf{H}_b)$)**.**
The function $\mathcal{G}(\mathbf{H}_b, \tilde{\mathbf{H}}_b)$ defined below is an auxiliary function to $\mathcal{C}(\mathbf{H}_b)$.

$$\begin{aligned}\mathcal{G}(\mathbf{H}_b, \tilde{\mathbf{H}}_b) = \frac{1}{R} \sum_{r=1}^{R} \sum_{(f,n)\in\mathbb{B}} \Bigg[& \ln\left(g_n \sigma_f^2\left(\mathbf{z}_n^{(r)}\right) + \left(\mathbf{W}_b \tilde{\mathbf{H}}_b\right)_{f,n}\right) \\ & + \frac{\left(\mathbf{W}_b \mathbf{H}_b\right)_{f,n} - \left(\mathbf{W}_b \tilde{\mathbf{H}}_b\right)_{f,n}}{g_n \sigma_f^2\left(\mathbf{z}_n^{(r)}\right) + \left(\mathbf{W}_b \tilde{\mathbf{H}}_b\right)_{f,n}} \\ & + |x_{fn}|^2 \Bigg(\frac{g_n \sigma_f^2\left(\mathbf{z}_n^{(r)}\right)}{\left(g_n \sigma_f^2\left(\mathbf{z}_n^{(r)}\right) + \left(\mathbf{W}_b \tilde{\mathbf{H}}_b\right)_{f,n}\right)^2} \\ & + \sum_{k=1}^{K_b} \frac{w_{b,fk} \tilde{h}_{b,kn}^2}{h_{b,kn}\left(g_n \sigma_f^2\left(\mathbf{z}_n^{(r)}\right) + \left(\mathbf{W}_b \tilde{\mathbf{H}}_b\right)_{f,n}\right)^2} \Bigg) \Bigg],\end{aligned} \tag{14}$$

where for all $(f,n) \in \mathbb{B}$ and $k \in \{1, ..., K_b\}$, $w_{b,fk} = (\mathbf{W}_b)_{f,k}$ $h_{b,kn} = (\mathbf{H}_b)_{k,n}$ and $\tilde{h}_{b,kn} = (\tilde{\mathbf{H}}_b)_{k,n}$.

Proof. The proof is provided in [22] due to space limitation.

The auxiliary function $\mathcal{G}(\mathbf{H}_b, \tilde{\mathbf{H}}_b)$ is separable in convex functions of the individual coefficients $h_{b,kn}$, $k \in \{1, ..., K_b\}$, $n \in \{0, ..., N-1\}$, which is convenient for updating those parameters. By canceling the partial derivative of $\mathcal{G}(\mathbf{H}_b, \tilde{\mathbf{H}}_b)$ with respect to $h_{b,kn}$, we obtain an update which depends on $\tilde{h}_{b,kn}$. According to the MM algorithm, we can use the fact that $\tilde{h}_{b,kn}$ is equal to the previous value of $h_{b,kn}$ for obtaining the following multiplicative update in matrix form:

$$\mathbf{H}_b \leftarrow \mathbf{H}_b \odot \left[\frac{\mathbf{W}_b^\top \left(|\mathbf{X}|^{\odot 2} \odot \sum_{r=1}^{R} \left(\mathbf{V}_x^{(r)}\right)^{\odot -2} \right)}{\mathbf{W}_b^\top \sum_{r=1}^{R} \left(\mathbf{V}_x^{(r)}\right)^{\odot -1}} \right]^{\odot 1/2}, \tag{15}$$

where $\odot$ denotes element-wise multiplication and exponentiation, matrix division is also element-wise, $\mathbf{V}_x^{(r)} \in \mathbb{R}_+^{F \times N}$ is the matrix of entries $\left(\mathbf{V}_x^{(r)}\right)_{f,n} = g_n \sigma_f^2\left(\mathbf{z}_n^{(r)}\right) + (\mathbf{W}_b \mathbf{H}_b)_{f,n}$, and $\mathbf{X} \in \mathbb{C}^{F \times N}$ is the matrix of entries $(\mathbf{X})_{f,n} = x_{fn}$.

We can straightforwardly apply the same procedure for maximizing $\tilde{Q}(\boldsymbol{\theta}_u; \boldsymbol{\theta}_u^\star)$ with respect to $\mathbf{W}_b$. The resulting multiplicative update rule is given by:

$$\mathbf{W}_b \leftarrow \mathbf{W}_b \odot \left[\frac{\left(|\mathbf{X}|^{\odot 2} \odot \sum_{r=1}^{R} \left(\mathbf{V}_x^{(r)}\right)^{\odot -2} \right) \mathbf{H}_b^\top}{\sum_{r=1}^{R} \left(\mathbf{V}_x^{(r)}\right)^{\odot -1} \mathbf{H}_b^\top} \right]^{\odot 1/2}. \tag{16}$$

Again, in a very similar fashion, the vector of speech gains is updated as follows:

$$\mathbf{g}^\top \leftarrow \mathbf{g}^\top \odot \left[\frac{\mathbf{1}^\top \left[|\mathbf{X}|^{\odot 2} \odot \sum_{r=1}^{R} \left(\mathbf{V}_s^{(r)} \odot \left(\mathbf{V}_x^{(r)}\right)^{\odot -2} \right) \right]}{\mathbf{1}^\top \left[\sum_{r=1}^{R} \left(\mathbf{V}_s^{(r)} \odot \left(\mathbf{V}_x^{(r)}\right)^{\odot -1} \right) \right]} \right]^{\odot 1/2}, \tag{17}$$

where $\mathbf{1}$ is an all-ones column vector of dimension F and $\mathbf{V}_s^{(r)} \in \mathbb{R}_+^{F \times N}$ is the matrix of entries $\left(\mathbf{V}_s^{(r)}\right)_{f,n} = \sigma_f^2\left(\mathbf{z}_n^{(r)}\right)$.

The non-negativity of $\mathbf{H}_b$, $\mathbf{W}_b$ and $\mathbf{g}$ is ensured provided that those parameters are initialized with non-negative entries. In practice, at the M-step of the MCEM algorithm, we perform only one iteration of updates (15), (16) and (17).

4.3. Speech reconstruction

In the following $\boldsymbol{\theta}_u^\star = \{\mathbf{W}_b^\star, \mathbf{H}_b^\star, \mathbf{g}^\star\}$ denotes the set of parameters estimated by the above MCEM algorithm. For all $(f,n) \in \mathbb{B}$, let $\tilde{s}_{fn} = \sqrt{g_n^\star} s_{fn}$ be the scaled version of the speech STFT coefficients as introduced in (6), with $g_n^\star = (\mathbf{g}^\star)_n$. Our final goal is to estimate those coefficients according to their posterior mean:

$$\begin{aligned}\hat{\tilde{s}}_{fn} &= \mathbb{E}_{p(\tilde{s}_{fn} | x_{fn}; \boldsymbol{\theta}_s, \boldsymbol{\theta}_u^\star)}[\tilde{s}_{fn}] \\ &= \mathbb{E}_{p(\mathbf{z}_n | \mathbf{x}_n; \boldsymbol{\theta}_s, \boldsymbol{\theta}_u^\star)} \left[\mathbb{E}_{p(\tilde{s}_{fn} | \mathbf{z}_n, \mathbf{x}_n; \boldsymbol{\theta}_s, \boldsymbol{\theta}_u^\star)}[\tilde{s}_{fn}] \right] \\ &= \mathbb{E}_{p(\mathbf{z}_n | \mathbf{x}_n; \boldsymbol{\theta}_s, \boldsymbol{\theta}_u^\star)} \left[\frac{g_n^\star \sigma_f^2(\mathbf{z}_n)}{g_n^\star \sigma_f^2(\mathbf{z}_n) + (\mathbf{W}_b^\star \mathbf{H}_b^\star)_{f,n}} \right] x_{fn}.\end{aligned} \tag{18}$$

This estimation corresponds to a soft-masking of the mixture signal, similarly as the standard Wiener filtering. Exactly as before, this expectation cannot be computed in an analytical form, but we can approximate it using the same Metropolis-Hastings algorithm as detailed in Section 4.1. Note that this estimation procedure is different than the one proposed in [7]. Here the STFT source coefficients are estimated from their true posterior distribution, without conditioning on the latent variables. The time-domain estimate of the speech signal is finally obtained by inverse STFT and overlap-add.

5. SUPERVISED TRAINING OF THE SPEECH MODEL

For supervised training of the speech model, we use a large dataset of independent clean-speech STFT time frames $\mathbf{s}^{tr} = \{\mathbf{s}_n \in \mathbb{C}^F\}_{n=0}^{N_{tr}-1}$. In our speech enhancement method detailed above, we are only interested in using the generative model defined in (3) and (4). However, in order to learn the parameters $\boldsymbol{\theta}_s$ of this generative model, we also need to introduce a *recognition network* or *probabilistic encoder* $q(\mathbf{z} \mid \mathbf{s}^{tr}; \boldsymbol{\phi})$, which is an approximation of the true posterior $p(\mathbf{z} \mid \mathbf{s}^{tr}; \boldsymbol{\theta}_s)$ [8]. Independently for all $l \in \{0, ..., L-1\}$ and $n \in \{0, ..., N_{tr}-1\}$, $q(\mathbf{z} \mid \mathbf{s}^{tr}; \boldsymbol{\phi})$ is defined similarly as in [7] by:

$$z_{l,n} \mid \mathbf{s}_n; \boldsymbol{\phi} \sim \mathcal{N}\left(\tilde{\mu}_l\left(|\mathbf{s}_n|^{\odot 2}\right), \tilde{\sigma}_l^2\left(|\mathbf{s}_n|^{\odot 2}\right)\right), \tag{19}$$

where $z_{l,n} = (\mathbf{z}_n)_l$ and $\tilde{\mu}_l : \mathbb{R}_+^F \mapsto \mathbb{R}$, $\tilde{\sigma}_l^2 : \mathbb{R}_+^F \mapsto \mathbb{R}_+$ are non-linear functions parametrized by $\boldsymbol{\phi}$. The output of each function is provided by one neuron in the $2L$-dimensional output layer of a neural network, whose weights and biases are denoted by $\boldsymbol{\phi}$.

As explained in [8], the recognition network parameters $\boldsymbol{\phi}$ and the generative network parameters $\boldsymbol{\theta}_s$ can be jointly trained by maximizing the evidence lower-bound (ELBO) defined by:

$$\mathcal{L}(\boldsymbol{\theta}_s, \boldsymbol{\phi}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{s}^{tr};\boldsymbol{\phi})}\left[\ln p\left(\mathbf{s}^{tr} \mid \mathbf{z}; \boldsymbol{\theta}_s\right)\right] - D_{KL}\left(q\left(\mathbf{z} \mid \mathbf{s}^{tr}; \boldsymbol{\phi}\right) \parallel p(\mathbf{z})\right), \tag{20}$$

where $D_{KL}(q \parallel p) = \mathbb{E}_q[\ln(q/p)]$ is the Kullback-Leibler divergence. The first term in the right-hand side of (20) is a reconstruction accuracy term, while the second one is a regularization term. From (3), (4) and (19), the ELBO can be developed as follows:

$$\mathcal{L}(\boldsymbol{\theta}_s, \boldsymbol{\phi}) \overset{c}{=} -\sum_{f=0}^{F-1}\sum_{n=0}^{N_{tr}-1} \mathbb{E}_{q(\mathbf{z}_n|\mathbf{s}_n;\boldsymbol{\phi})}\left[d_{IS}\left(|s_{fn}|^2; \sigma_f^2(\mathbf{z}_n)\right)\right] + \frac{1}{2}\sum_{l=0}^{L-1}\sum_{n=0}^{N_{tr}-1}\left[\ln \tilde{\sigma}_l^2\left(|\mathbf{s}_n|^{\odot 2}\right) - \tilde{\mu}_l\left(|\mathbf{s}_n|^{\odot 2}\right)^2 - \tilde{\sigma}_l^2\left(|\mathbf{s}_n|^{\odot 2}\right)\right], \tag{21}$$

where $d_{IS}(x; y) = x/y - \ln(x/y) - 1$ is the Itakura-Saito (IS) divergence. We note that maximizing the ELBO with respect to the parameters $\boldsymbol{\theta}_s$ of the generative speech model amounts to minimizing a cost function that involves the IS divergence between the observed power spectrogram $|s_{fn}|^2$ and the variance model $\sigma_f^2(\mathbf{z}_n)$. It is interesting to highlight the similarity with the Gaussian generative speech model (1)-(2), where maximum likelihood estimation of its parameters also amounts to minimizing the IS divergence between $|s_{fn}|^2$ and the NMF-based variance parametrization $(\mathbf{W}_s\mathbf{H}_s)_{f,n}$, as proved in [3].

Finally, as the expectation in (21) cannot be computed in an analytical form, we again use a Monte Carlo approximation:

$$\mathbb{E}_{q(\mathbf{z}_n|\mathbf{s}_n;\boldsymbol{\phi})}\left[d_{IS}\left(|s_{fn}|^2; \sigma_f^2(\mathbf{z}_n)\right)\right] \approx \frac{1}{R}\sum_{r=1}^{R} d_{IS}\left(|s_{fn}|^2; \sigma_f^2\left(\mathbf{z}_n^{(r)}\right)\right), \tag{22}$$

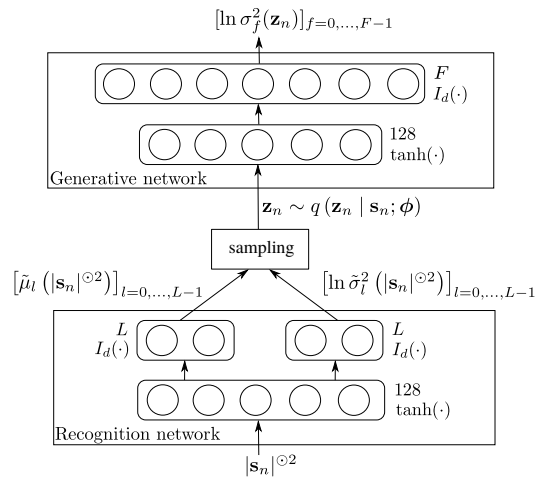

Fig. 1. Structure of our variational autoencoder. All layers are fully connected. Next to each layer we indicate its size along with the activation function that is used.

where for all $r \in \{1, ..., R\}$, $\mathbf{z}_n^{(r)}$ is independently drawn from $q(\mathbf{z}_n \mid \mathbf{s}_n; \boldsymbol{\phi})$ in (19), using the so-called *reparametrization trick* [8]. Independently for all $l \in \{0, ..., L-1\}$, it consists in sampling:

$$z_{l,n}^{(r)} \mid \mathbf{s}_n; \boldsymbol{\phi} \sim \tilde{\mu}_l\left(|\mathbf{s}_n|^{\odot 2}\right) + \tilde{\sigma}_l\left(|\mathbf{s}_n|^{\odot 2}\right)\mathcal{N}(0, 1). \tag{23}$$

Injecting (22) in (21), we obtain an objective function which is differentiable with respect to both $\boldsymbol{\theta}_s$ and $\boldsymbol{\phi}$, and which can be optimized using gradient-ascent-based algorithms.

6. EXPERIMENTS

Database: The supervised speech model parameters are learned from the training set of the TIMIT database [23]. It contains almost 4 hours of speech signals at a 16-kHz sampling rate, distributed over 462 speakers. For the evaluation of the speech enhancement algorithm, we mixed clean speech signals from the test set of the TIMIT database and noise signals from the DEMAND database [24], with various noisy environments: domestic, nature, office, indoor public spaces, street and transportation. We created 168 mixtures at a 0 dB signal-to-noise ratio (one mixture per speaker in the TIMIT test set). Note that both speakers and sentences are different than in the training set.

Baseline methods: Considering that both approaches belong to the variance modeling framework, we first compare our method with a semi-supervised NMF baseline as described in Section 2. The speech NMF dictionary is learned using the training set of the TIMIT database, and only the speech activation matrix and the noise NMF model parameters are estimated from the noisy mixture signals. The speech signal is then recovered by Wiener filtering. We also compare our approach with the fully supervised deep learning method proposed in [10]. In this work, a deep neural network is trained to map noisy speech log-power spectrograms to clean speech log-power spectrograms. The authors showed that their mapping-based approach outperformed other deep learning techniques that rely on the estimation of time-frequency masks. Moreover, the authors used more than 100 different noise types to train their system, which showed to be quite effective in handling unseen noise types. It is

therefore a very relevant method to compare with. We used the Python code provided by the authors.[1]

Parameter settings: The STFT is computed using a 64-ms sine window (i.e. $F = 513$) with 75%-overlap. We compare the performance of our method with the semi-supervised NMF baseline according to the dimension of the latent space L and the rank K_s of the NMF speech model. For fair comparison, we set $L = K_s \in \{8, 16, 32, 64, 128\}$. The rank of the noise model is arbitrarily fixed to $K_b = 10$. Unsupervised NMF parameters are randomly initialized. For the proposed method, the gain vector $\mathbf{g}$ is initialized with an all-ones vector. The iterative algorithms of those two methods are stopped if the improvement in terms of objective function is smaller that 10^{-4} (for our method we monitored $\tilde{Q}(\boldsymbol{\theta}_u; \boldsymbol{\theta}_u^\star)$). The parameters of our MCEM algorithm are the following ones: At each E-Step (see Section 4.1), we run 40 iterations of the Metropolis-Hastings algorithm using $\epsilon^2 = 0.01$ for the proposal distribution, and we discard the first 30 samples as the burn-in period. At the first iteration of the MCEM algorithm we need to initialize the Markov chain of the Metropolis-Hastings algorithm. For that purpose we use the recognition network considering the mixture signal as input because the clean speech signal is not observed: for all $l \in \{0, ..., L-1\}$, $z_{l,n}^{(0)} = \tilde{\mu}_l\left(|\mathbf{x}_n|^{\odot 2}\right)$. Then, at each new E-Step, we use the last sample drawn at the previous E-Step to initialize the Markov chain. For estimating the speech STFT coefficients (see Section 4.3), we run the Metropolis-Hastings algorithm for 100 iterations and discard the first 75 samples.

Variational autoencoder: The structure of our variational autoencoder is represented in Fig. 1. As in [8], we use hyperbolic tangent ($\tanh(\cdot)$) activation functions except for the encoder/decoder output layers, which use identity activation functions ($I_d(\cdot)$). The values of these output layers thus lie in $\mathbb{R}$, which is the reason why we output logarithm of variances. For learning the parameters $\boldsymbol{\theta}_s$ and $\boldsymbol{\phi}$ (see Section 5), we used the Adam optimizer [25] with a step size of 10^{-3}, exponential decay rates for the first and second moment estimates of 0.9 and 0.999 respectively, and an epsilon of 10^{-7} for preventing division by zero. 20% of the TIMIT training set was kept as a validation set, and early stopping with a patience of 10 epochs was used. Weights were initialized using the uniform initializer described in [26]. For reproducibility, a Python implementation of our algorithm is available online.[2]

Results: We evaluate the enhanced speech signal in terms of signal-to-distortion ratio (SDR) [27] and perceptual evaluation of speech quality (PESQ) measure [28]. The SDR is expressed in decibels (dB) and the PESQ score lies between -0.5 and 4.5. We computed the SDR using the mir_eval Python library.[3] We first illustrate the interest of having a time-frame dependent gain g_n as introduced in (6). Inevitably, the variational autoencoder is trained using examples with a limited loudness range. If we force $g_n = 1$ for all $n \in \{0, ..., N-1\}$, we can therefore expect the speech enhancement quality to depend on the speech power in the observed noisy mixture signal. This problem is illustrated in Fig. 2: The SDR indicated by black dots suddenly drops for a scaling factor of the mixture signal greater than 12 dB. It indicates that the power of the underlying clean speech signal in this mixture went out of the power range observed during the training. One solution to this problem could be to perform some kind of data augmentation as in [7]. Here we exploited the statistical modeling framework, by introducing parameters in order to handle some inherent drawbacks of learning-based approaches. The gray diamonds in Fig. 2 indeed show that the time-frame-dependent gains provide a solution to this robustness issue.

[1] https://github.com/yongxuUSTC/sednn

[2] https://github.com/sleglaive/MLSP-2018

[3] https://github.com/craffel/mir_eval

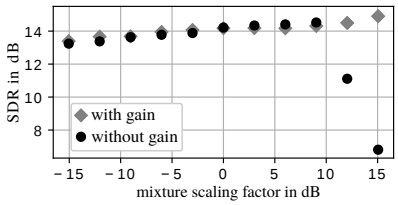

Fig. 2. SDR according to different scalings of the power of one mixture signal in our test dataset. Grey diamonds represent the performance when the vector of time-frame dependent gains $\mathbf{g}$ is updated at the M-Step, and black dots when $\mathbf{g}$ is fixed to an all-ones vector.

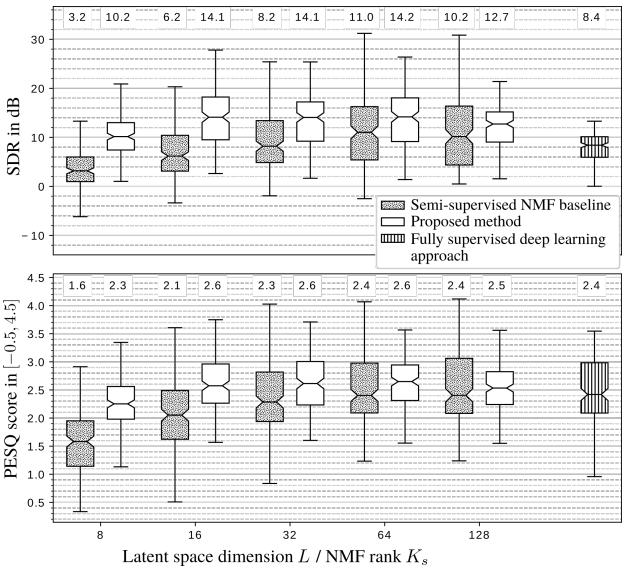

Fig. 3. Enhancement results, the median is indicated above each box.

The speech enhancement results are presented in Fig. 3. We first observe that while the NMF baseline requires a sufficiently large rank $K_s = 64$, our method obtains quite stable results for a latent dimension L greater than or equal to 16. Moreover it always performs better than the NMF baseline in terms of both SDR and PESQ measures. We also see that it outperforms the fully supervised deep learning approach [10]. It is indeed difficult for such kind of approaches to generalize to unseen noises, which strongly justifies the interest of semi-supervised approaches. Audio examples illustrating those results are available online.[4]

We conclude this section with some information about the computational time for enhancing a 2.6 seconds-long mixture, using a central processing unit at 2.5 GHz. One iteration of the proposed MCEM algorithm (with $L = 64$) takes around 2.9 seconds. For the semi-supervised NMF baseline (with $K_s = 64$), one iteration of update rules takes around 4 milliseconds. The fully supervised deep-learning approach, which is non iterative, takes around 7 seconds.

7. CONCLUSION

In this paper we showed that within the variance modeling framework, variational autoencoders are a suitable alternative to super-

[4] https://sleglaive.github.io/demo-mlsp18.html

vised NMF. We proposed a semi-supervised speech enhancement method based on an MCEM algorithm. Experimental results demonstrated that it outperforms both a semi-supervised NMF baseline and a fully supervised deep learning approach. We used a relatively shallow architecture for the variational autoencoder, we can therefore expect even better results by going deeper. Future works include using variational autoencoders for separating similar sources, e.g. multiple speakers. This is a much harder problem and we may have to exploit spatial information provided by multichannel mixtures to address it. Another interesting perspective would be to introduce a temporal model for the latent variables of the variational autoencoder.

8. REFERENCES

[1] Philipos C. Loizou, *Speech enhancement: theory and practice*, CRC press, 2007.

[2] Emmanuel Vincent, Maria G. Jafari, Samer A. Abdallah, Mark D. Plumbley, and Mike E. Davies, "Probabilistic modeling paradigms for audio source separation," in *Machine Audition: Principles, Algorithms and Systems*, Wenwu Wang, Ed., pp. 162–185. IGI Global, 2010.

[3] Cédric Févotte, Nancy Bertin, and Jean-Louis Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis," *Neural computation*, vol. 21, no. 3, pp. 793–830, 2009.

[4] DeLiang Wang and Jitong Chen, "Supervised speech separation based on deep learning: an overview," *arXiv preprint arXiv:1708.07524*, 2017.

[5] Aditya Arie Nugraha, Antoine Liutkus, and Emmanuel Vincent, "Multichannel audio source separation with deep neural networks," *IEEE Trans. Audio, Speech, Language Process.*, vol. 24, no. 9, pp. 1652–1664, 2016.

[6] Paris Smaragdis and Shrikant Venkataramani, "A neural network alternative to non-negative audio models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2017, pp. 86–90.

[7] Yoshiaki Bando, Masato Mimura, Katsutoshi Itoyama, Kazuyoshi Yoshii, and Tatsuya Kawahara, "Statistical speech enhancement based on probabilistic integration of variational autoencoder and non-negative matrix factorization," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2018, pp. 716–720.

[8] Diederik P. Kingma and Max Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2014.

[9] Greg C.G. Wei and Martin A. Tanner, "A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms," *Journal of the American statistical Association*, vol. 85, no. 411, pp. 699–704, 1990.

[10] Yong Xu, Jun Du, Li-Rong Dai, and Chin-Hui Lee, "A regression approach to speech enhancement based on deep neural networks," *IEEE Trans. Audio, Speech, Language Process.*, vol. 23, no. 1, pp. 7–19, 2015.

[11] Antoine Liutkus, Roland Badeau, and Gäel Richard, "Gaussian processes for underdetermined source separation," *IEEE Trans. Signal Process.*, vol. 59, no. 7, pp. 3155–3167, 2011.

[12] Esa Ollila, Jan Eriksson, and Visa Koivunen, "Complex elliptically symmetric random variables-generation, characterization, and circularity tests," *IEEE Trans. Signal Process.*, vol. 59, no. 1, pp. 58–69, 2011.

[13] Antoine Liutkus and Roland Badeau, "Generalized Wiener filtering with fractional power spectrograms," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2015, pp. 266–270.

[14] Kazuyoshi Yoshii, Katsutoshi Itoyama, and Masataka Goto, "Student's t nonnegative matrix factorization and positive semidefinite tensor factorization for single-channel audio source separation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2016, pp. 51–55.

[15] Paris Smaragdis, Bhiksha Raj, and Madhusudana Shashanka, "Supervised and semi-supervised separation of sounds from single-channel mixtures," in *Proc. Int. Conf. Indep. Component Analysis and Signal Separation*, 2007, pp. 414–421.

[16] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the royal statistical society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[17] Christian P. Robert and George Casella, *Monte Carlo Statistical Methods*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[18] K.S. Chan and Johannes Ledolter, "Monte Carlo EM estimation for time series models involving counts," *Journal of the American Statistical Association*, vol. 90, no. 429, pp. 242–252, 1995.

[19] Cédric Févotte and Jérôme Idier, "Algorithms for nonnegative matrix factorization with the β-divergence," *Neural computation*, vol. 23, no. 9, pp. 2421–2456, 2011.

[20] David R. Hunter and Kenneth Lange, "A tutorial on MM algorithms," *The American Statistician*, vol. 58, no. 1, pp. 30–37, 2004.

[21] C.F. Jeff Wu, "On the convergence properties of the EM algorithm," *The Annals of statistics*, pp. 95–103, 1983.

[22] "Supporting document," `http://sleglaive.github.io/documents/sup_doc_MLSP18.pdf`.

[23] John S. Garofolo, Lori F. Lamel, William M. Fisher, Jonathan G. Fiscus, David S. Pallett, Nancy L. Dahlgren, and Victor Zue, "TIMIT acoustic phonetic continuous speech corpus," in *Linguistic data consortium*, 1993.

[24] Joachim Thiemann, Nobutaka Ito, and Emmanuel Vincent, "The Diverse Environments Multi-channel Acoustic Noise Database (DEMAND): A database of multichannel environmental noise recordings," in *Proc. Int. Cong. on Acoust.*, 2013.

[25] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[26] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artif. Intelligence and Stat.*, 2010, pp. 249–256.

[27] Emmanuel Vincent, Rémi Gribonval, and Cédric Févotte, "Performance measurement in blind audio source separation," *IEEE Trans. Audio, Speech, Language Process.*, vol. 14, no. 4, pp. 1462–1469, 2006.

[28] Antony W. Rix, John G. Beerends, Michael P. Hollier, and Andries P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2001, pp. 749–752.