A Combined Statistical and Machine Learning Approach For Single Channel Speech Enhancement

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Hung-Wei Tseng

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Zhi-Quan Luo

May, 2015

© Hung-Wei Tseng 2015
ALL RIGHTS RESERVED

Acknowledgements

I would like to express my sincerest gratitude to my advisor, Dr. Zhi-Quan Luo, whose expertise, understanding, and patience guided me throughout my graduate studies. Dr. Luo gave me the freedom to explore my research interests while simultaneously providing instruction that enabled me to learn from my mistakes.

In addition to Dr. Luo, I would like to extend my thanks to my other committee members, Dr. Nikos D. Sidiropoulos, Dr. Andrew J. Oxenham and Dr. Jarvis D. Haupt, for their encouragement and valuable comments to my thesis.

This thesis would not have been possible without the help from my collaborators and labmates. In particular, I would like to thank Dr. Srikanth Vishnubhotlas for his comments about psychoacoustics, Dr. Mingyi Hong for his insightful theoretical suggestions, and Ruoyu Sun for his fruitful discussions with me about research topics.

A special thanks goes to the congregation of the Evangelical Formosan Church of the Twin Cities (EFCTC) , my spiritual family in the United States, for their sincere friendship and continuous prayers.

Last but not the least, I would like to thank my parents Chin-Hsing Tseng and Cheng-Lin Tung, along with my younger brother Hung-Yi Tseng. It was through their unconditional love and support that I was able to overcome many difficulties.

Dedication

TO THE ALMIGHTY GOD, THE SAVIOR OF MY LIFE

“Then Jesus said, Did I not tell you that if you believe, you will see the glory of God? John 11:40 ”

Abstract

In this thesis, we study the single-channel speech enhancement problem, the goal of which is to recover a desired speech from a monaural noisy recording. Speech enhancement is a focal issue to study due to is widespread usage in speech-related applications, such as hearing aids, mobile communications, and speech recognition systems.

Three speech enhancement algorithms are proposed. In the first algorithm, the Wiener Non-negative Matrix Factorization (WNMF), we combine the traditional Wiener filtering and the NMF into a single optimization problem. The objective is to minimize the mean square error, similar to Wiener filtering, and the constraints ensure the enhanced speeches are sparsely representable by the speech model learned by NMF. WNMF is novel because it utilizes NMF to capture the speech-specific structure while simultaneously leveraging it, thus improving the Wiener filtering. For the second algorithm, we propose a Sparse Gaussian Mixture Model (SGMM) that extends the traditional NMF and the Gaussian model. SGMM better captures the complex structure of speech than the traditional NMF. To control for overrepresentation of SGMM, we impose sparsity in order to ensure that only a few Gaussian models are simultaneously active. Computationally, it is achieved by using a l_0-norm in the constraint of the maximum-likelihood (ML) estimation. The contribution of SGMM is in solving the constrained ML estimation, which has a closed form update even with the non-convex and non-smooth l_0-norm constraint. The final algorithm proposed is the Sparse NMF + Deep Neural Network (SNMF-DNN), in which we treat speech enhancement as a supervised regression problem - the goal being to estimate the optimal enhancement gain. SNMF, originally designed for source separation, is used to extract features from the noisy recording. DNN is subsequently trained to estimate the optimal enhancement gain. Although our system is simple and does not require any sophisticated handcrafted features, we are able to demonstrate a substantial improvement in both intelligibility and enhanced speech quality.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

In this thesis, we consider an old but important problem, i.e., enhancing target speech from a monaural noisy recording. This problem is important because of its broad engineering applications, such as hearing aids, mobile communications, and robust speech recognition. In these systems, speech enhancement is usually applied before any further audio processing. Moreover, when there is only one microphone, as is the usual case for most cost-effective devices today, effective monaural speech enhancement is of great need. Despite decades of study [5], improving both the enhanced speech quality and the speech intelligibility still remains a challenging task, especially when the noise level is high or when the noise is non-stationary. Monaural speech enhancement is difficult because it is intrinsically an underdetermined system, i.e., with only one noisy mixture. Our aim is to separate speech and noise. The goal of this thesis is to develop effective and efficient algorithms for monaural speech enhancement.

1.2 System Model and Time-Frequency Representations

In this section, we present the time-domain system model that is used in this thesis and in most existing works. Also, we present two commonly used time-frequency representations, the spectrogram and the cochleagram.

1.2.1 System Model

We consider the simple additive noisy model:

$$y(t) = x(t) + v(t) \tag{1.1}$$

where $y(t)$, $x(t)$, and $v(t)$ denote, respectively, the noisy observation, the desired clean speech, and the additive noise at time frame t. Eq. (1.1) assumes only one path from the sound source to the microphone, and assumes the additive noise. In real world, however, there are multiple paths with different delays, which is commonly called the reverberant model. However, if there exist a dominant path, (1.1) will be a good approximation of the reverberant model. Moreover, if the impulse response from the sound source to the microphone can be accurately approximated, the reverberant model can be transformed into (1.1) by equalization. Therefore, we use (1.1) in this thesis for simplicity.

However, the time domain model (1.1) is difficult to analyze. Most speech enhancement algorithms work on the time-frequency (TF) representation, which better presents the characteristics of speech and noise. Two commonly used TF representations are the spectrogram and the cochleagram [6].

1.2.2 Spectrogram Representation

To convert the time-domain signal $x(t)$ into the spectrogram $\mathbf{X}$, we apply the short-time-Fourier-transform (STFT) as illustrated in Fig. 1.1. A segment of the time domain signal is extracted, for example, 512 samples is used in this thesis. It corresponds to 32-ms for a signal sampled at 16 kHz. Then, 512 point Fast-Fourier-Transform (FFT) along with the Hamming windowing are used to convert the time-domain sample into the frequency domain. There is some overlap between segments, e.g., 50% is used here. Due to the symmetry of FFT, only the first 257 FFT coefficients are kept and are stored as the column of the spectrogram matrix $\mathbf{X}$. Therefore, $\mathbf{X}$ has dimension 257 by N, where N is the number of FFT frames. To convert back to the time domain, we apply the inverse FFT on $\mathbf{X}$ followed by an overlap-add operation.

Spectrogram is the most widely used TF representation because of its computational efficiency and interpretable presentation. It is well-known that a N pint FFT can be computed in $O(N \log N)$. In Fig. 1.2, we show the waveform and the spectrogram of

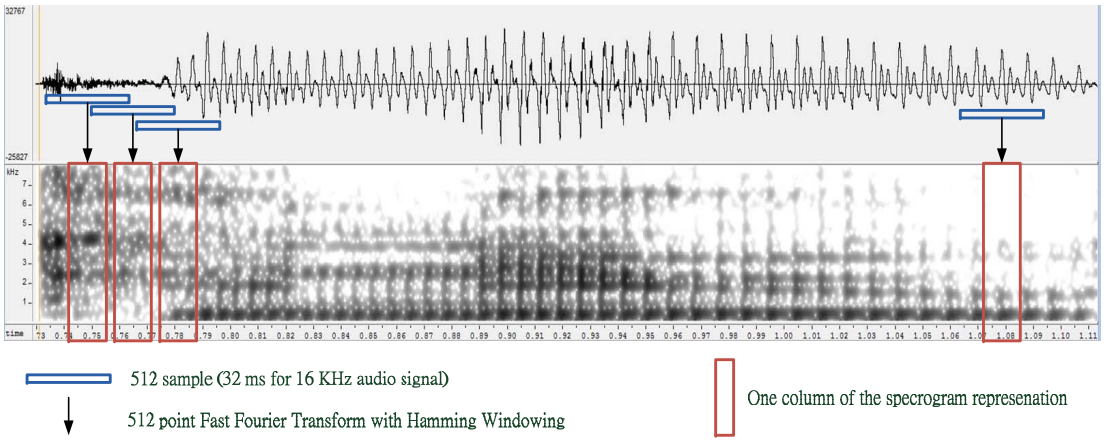


Figure 1.1: Spectrogram computation

a clean speech and a noisy speech. It is clear that speech-specific structures are more pronounced in the spectrogram domain than in the time-domain.

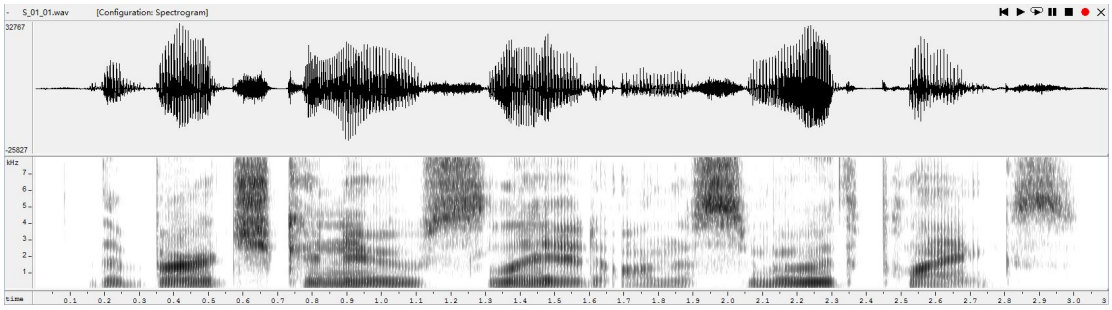
We can rewrite the system model (1.1) in the spectrogram domain:

$$\mathbf{Y} = \mathbf{X} + \mathbf{V} \tag{1.2}$$

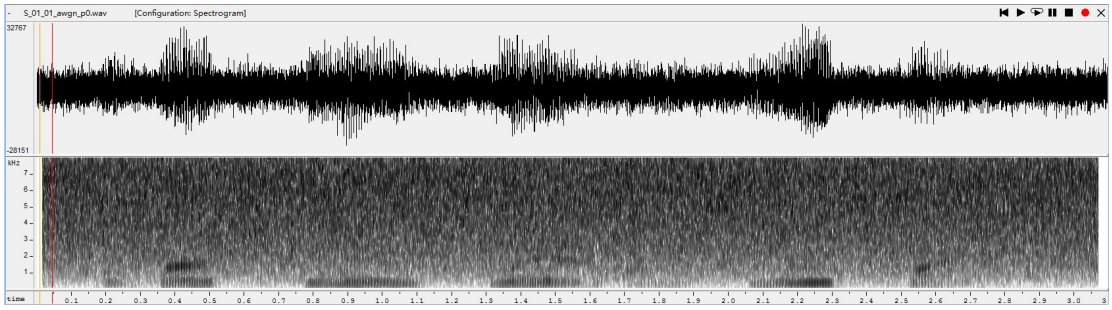
where $\mathbf{Y} \in \mathcal{C}^{K \times N}$ denotes the spectrogram representation of the noisy observation. K denotes the number of frequency bins and N denotes the number of time frames. $\mathbf{X} \in \mathcal{C}^{K \times N}$ and $\mathbf{V} \in \mathcal{C}^{K \times N}$ denotes, respectively, the spectrogram of the clean speech and the additive noise.

1.2.3 Cochleagram Representation

The cochleagram is another TF representation that is mostly used in Computer Auditory Scene Analysis (CASA) [6]. Fig. 1.3 shows the diagram for computing the cochleagram. In the first step, audio signal $x(t)$ is passed through a 64-channel Gammatone filterbank with center frequencies spanning from 50 Hz to 8 kHz on the equivalent rectangular bandwidth rate scale. It generates 64 audio streams, denoted by $x_1(t), x_2(t), \cdots, x_{64}(t)$, centered at different frequency. To calculate the cochleagram, the 64 streams are divided



(a) clean speech



(b) 0 dB noisy speech (Additive White Gaussian Noise)

Figure 1.2: Waveform and Spectrogram representation of a clean speech and a noisy speech.

into 20-ms segments with 10-ms overlap. Energy in each segments is calculated and then forms a TF representation called cochleagram.

Compared with spectrogram, the main difference is the frequency spacing between channels. In the spectrogram, the frequency spacing is linear, which is about 31.25 Hz in our previous setup. However, the frequency spacing in the cochleagram is logarithmic. The lower frequency bins has smaller spacing, and the higher frequency bins has larger spacing. This is used to approximate the human auditory system. Due to the non-linear spacing, the cochleagram representation has only 64 channels, while the spectrogram has 257 channels.

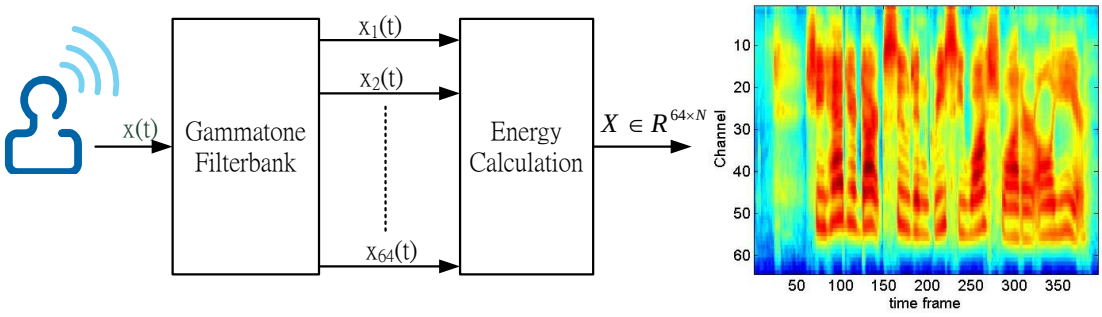


Figure 1.3: Cochleagram computation.

1.3 Existing Algorithms

In this section, we survey three major approaches for speech enhancement.

1. Bayesian-based approach
2. Non-negative Matrix Factorization (NMF)-based approach
3. Ideal Masking (IM)-based approach

We will present the basic ideas of these approaches and comment on their pros and cons. Our proposed algorithms presented in Chap. 3 - 5 are developed upon these ideas. From a historical point of view, Bayesian-based algorithms have been studied since 1950. Researchers began to apply NMF for speech enhancement around 2000. The first practically implementable IM-based algorithm that demonstrate substantial performance improvement was published in 2009 [7].

1.3.1 Bayesian-based Approach

Considering the spectrogram model (1.2), Bayesian-based algorithms aim to recover $\mathbf{X}$ from $\mathbf{Y}$ by using statistical inference. Generally speaking, Bayesian-based approach is divided into three steps. In the first step, the speech signal $\mathbf{X}$ and the noise signal $\mathbf{V}$ are modeled statistically. We use $\eta_{k,n}^2$ and $\sigma_{k,n}^2$ to denote the variances of speech and noise at frequency bin k and time frame n, respectively. In the second step, $\eta_{k,n}^2$ and

$\sigma^2_{k,n}$ are estimated from $\mathbf{Y}$. In the last step, $\mathbf{X}$ is estimated based on certain statistical criteria. Bayesian-based algorithms (e.g., [1,3,8–13]) differ from each in how these three steps are implemented. Table 1.1 summarizes the difference between these algorithms.

Table 1.1: Summary of Bayesian-based algorithms. WF denotes the Wiener filtering, DD denotes the Directed-Directed approach [1], and MS denotes the Minimum-Statistics approach [2]. "Gauss" denotes the complex Gaussian distribution, while "Non-Gauss" denotes distributions other than Gaussian.

	Speech		Noise	*A priori* SNR	Noise variance		Estimator		
	Gauss	Non-Gauss	Gauss	DD [1]	Known	MS [2]	MMSE (DFT)	MMSE (amplitude)	MAP
WF	v		v	v	v		v		
[1]	v		v	v	v			v	
[3]	v		v	v	v			v(log)	
[8]	v		v	v	v			v(square)	v
[9]		v	v	v	v				v
[10]		v	v	v		v	v		
[11]		v	v	v	v			v	v
[12]		v	v	v	v			v	
[13]		v	v	v		v	v	v	

Statistical Model of Speech and Noise

Let $\mathbf{x}_{k,n}$ and $\mathbf{v}_{k,n}$ be the complex DFT coefficient of speech and noise at frequency k and time frame n. The goal is to model $\mathbf{x}_{k,n}$ and $\mathbf{v}_{k,n}$ as random variables. While the noise signal being Gaussian distributed is well-justified theoretically (due to the central limit theorem) and experimentally, the speech signal, on the other hand, does not follow a Gaussian distribution in practice. It is observed that heavy-tail distributions, such as Laplace distribution or Gamma distribution, better model the speech signal. For example, [9] models the amplitude of speech, $\|\mathbf{x}_{k,n}\|$, as a particular super-Gaussian distribution which can be specialized to Laplace and Gamma distribution. The phase of $\mathbf{x}_{k,n}$ is modeled as an uniform random variable between 0 and 2π. In [10, 12, 13], the real part and the imaginary part of $\mathbf{x}_{k,n}$ are modeled as independent Laplace or Gamma distribution. Changing the statistical modeling changes the probability density function of $\mathbf{Y}$, which will result in different statistical estimators.

Speech and Noise Variance Estimation

While the mean of $\mathbf{x}_{k,n}$ and $\mathbf{v}_{k,n}$ are usually assumed to be zero, their variances, $\eta_{k,n}^2$ and $\sigma_{k,n}^2$, are non-zero and need to be estimated. If the noise is stationary, i.e., $\sigma_{k,n}^2 = \tilde{\sigma}_k^2$ for all n, the noise variance can be reliably estimated by the initial few frames where only noise is present.

$$\tilde{\sigma}_k^2 = \frac{1}{M} \sum_{n=1}^{M} \|\mathbf{y}_{k,n}\|^2 \tag{1.3}$$

where we assume only noise is present during the first M frame. This is the reason why several papers [1, 3, 8, 9, 11, 12] assume the noise variance is known. However, real-world noises are typically non-stationary. The most well-known noise tracking algorithm is the Minimum-Statistics (MS) algorithm proposed in [2]. MS algorithm estimates the noise variance by tracking the spectra minima of each frequency bins. Recently, another noise tracking algorithm based on Minimum Mean-Square Error (MMSE) has been proposed [4]. Compared with the MS algorithm, the MMSE-based algorithm has a shorter tracking delay and thus is more suitable for fast-varying noises.

Instead of estimating the speech variance $\eta_{k,n}^2$ directly, most algorithms estimate the *a priori* SNR:

$$\xi_{k,n} := \frac{\eta_{k,n}^2}{\sigma_{k,n}^2} \tag{1.4}$$

Since $\sigma_{k,n}^2$ is known, $\eta_{k,n}^2$ can be easily computed from $\xi_{k,n}$. In almost every paper, the Decision-Directed (DD) approach [1] defined below is adopted for *a priori* SNR estimation.

$$\xi_{k,n} = \alpha \frac{\|\hat{\mathbf{x}}_{k,n-1}\|^2}{\sigma_{k,n}^2} + (1-\alpha) \max\left(\frac{\|\mathbf{y}_{k,n}\|^2}{\sigma_{k,n}^2} - 1, 0 \right) \tag{1.5}$$

where α is the temporal smoothing factor, which is usually set to 0.98. $\hat{\mathbf{x}}_{k,n-1}$ denotes the estimated complex DFT at the same frequency bin in the *previous* frame. The DD approach (1.5) estimates the *a priori* SNR by heavily relying on the temporal continuity. As a result, the annoying musical background noise can be significantly reduced. However, it will make the enhanced speech sound "muffled", hence affect the intelligibility. Moreover, if $\hat{\mathbf{x}}_{k,n-1}$ is wrongly estimated, the DD approach will propagate the estimation error to frame n.

Bayesian Estimator

Once the speech variance and the noise variance have been estimated, $\mathbf{x}_{k,n}$ can be estimated using different statistical criteria. We use $\theta_{k,n}$ to denote the phase of $\mathbf{y}_{k,n}$.

- MMSE of the complex DFT:

$$\hat{\mathbf{x}}_{k,n} = \mathbb{E}\left[\mathbf{x}_{k,n} \mid \mathbf{y}, \sigma^2_{k,n}, \eta^2_{k,n}\right]$$

- MMSE of the DFT amplitude:

$$\begin{aligned}\hat{\mathbf{x}}_{k,n} &= \hat{x}_{k,n}\exp(j\theta_{k,n}) \\ \hat{x}_{k,n} &= \mathbb{E}\left[\|\mathbf{x}_{k,n}\| \mid \mathbf{y}, \sigma^2_{k,n}, \eta^2_{k,n}\right]\end{aligned}$$

- MMSE of the DFT log amplitude:

$$\begin{aligned}\hat{\mathbf{x}}_{k,n} &= \hat{x}_{k,n}\exp(j\theta_{k,n}) \\ \hat{x}_{k,n} &= \exp\left(\mathbb{E}\left[\log\left(\|\mathbf{x}_{k,n}\|\right) \mid \mathbf{y}, \sigma^2_{k,n}, \eta^2_{k,n}\right]\right)\end{aligned}$$

- MMSE of the DFT amplitude square:

$$\begin{aligned}\hat{\mathbf{x}}_{k,n} &= \hat{x}_{k,n}\exp(j\theta_{k,n}) \\ \hat{x}_{k,n} &= \sqrt{\mathbb{E}\left[\|\mathbf{x}_{k,n}\|^2 \mid \mathbf{y}, \sigma^2_{k,n}, \eta^2_{k,n}\right]}\end{aligned}$$

- Maximum A Posteriori (MAP)

$$\hat{\mathbf{x}}_{k,n} = \arg\max_{\mathbf{x}_{k,n}} \quad p(\mathbf{x}_{k,n} \mid \mathbf{y}_{k,n}, \sigma^2_{k,n}, \eta^2_{k,n})$$

  where $p(\mathbf{x}_{k,n} \mid \mathbf{y}_{k,n}, \sigma^2_{k,n}, \eta^2_{k,n})$ denotes the probability density function.

Depending on different statistical assumptions, the difficulty of computing these estimators may vary greatly. While some are simple closed form [1], some require complicated numerical integration and special functional evaluation [10–13].

Pros and Cons of Bayesian-based Approach

Two major advantages of Bayesian-based approach are the unsupervised implementation and computational simplicity. Bayesian algorithms only require the knowledge of the speech and the noise variances, which can be unsupervisedly estimated from the noisy recording. In other words, Bayesian algorithms can be applied to any noisy speech without supervised training. In terms of complexity, the dominating factor is the evaluation of the Bayesian estimator, which can usually be computed in closed form or by table lookup. Moreover, they usually deliver satisfactory enhancement performance if the noise level is small enough and is stationary enough. Therefore, Bayesian algorithms are currently the main stream algorithms.

The disadvantage of Bayesian algorithms, however, also come from its unsupervised statistics estimation. When the noise level is high or when the noise is highly non-stationary, pure unsupervised algorithms produce erroneous statistics estimation. It significantly degrades the enhancement performance and produces unpleasant artifacts. To achieve a better performance, supervised methods that leverage the structure of speech and noise are needed.

1.3.2 NMF-based Approach

In this section, we will give a short review of the NMF-based approach for speech enhancement. First, we will review the basic idea of NMF. Followed by it, we will survey some existing works that apply NMF in single-channel speech enhancement or source separation. Source separation is also considered because speech enhancement can actually be viewed as a special case of source separation, if we view speech and noise as two sources.

NMF and Its Sparse Variant

NMF factorizes a non-negative matrix $X \in \mathcal{R}_+^{K\times N}$ into a product of a dictionary matrix $D \in \mathcal{R}_+^{K\times M}$ and a gain matrix $G \in \mathcal{R}_+^{M\times N}$:

$$X \approx D \times G \tag{1.6}$$

Let x_n, d_m, and g_n denotes the nth, mth, and nth column of X, D, and G, respectively. Eq. (1.6) means x_n is approximated as a weighted sum of d_m with weight $g_{m,n}$:

$$x_n \approx \sum_m g_{m,n} d_m \tag{1.7}$$

where $g_{m,n}$ is the mth entry of the vector g_n. $\{d_m\}_{\forall m}$ are also called as *atoms*, since they serve as the basic blocks for synthesizing x_n. In other words, NMF models the data by learning a dictionary D which can well approximate x_n through non-negative linear combination.

To learn a dictionary that meets (1.6), NMF solves the following optimization problem:

$$(D, G) = \arg \min_{D \geq 0, G \geq 0} d(X \| DG) \tag{1.8}$$

where $d(X\|\hat{X})$ is a cost function that measures the discrepancy between X and $\hat{X}$. Common cost functions include Eucldean distance [14], Kullback-Leibler divergence [14], Itakura-Saito divergence [15], and the negative likelihood of data in the probabilistic NMF [16]. Using different discrepancy measures changes the characteristics of the dictionary, and the best discrepancy measure is application-dependent.

Sparse NMF (SNMF) is a special variant of NMF that further restricts the combination weights g_n to be sparse. In other words, x_n lies in a low-dimensional conic hull defined by the *few* active atoms that correspond to the *few* non-zero indices of g_n. However, different x_n may has different sparse weight g_n. Due to the sparse representation, SNMF allows the dictionary to be overcomplete and still being discriminative. Assuming using Euclidean distance as the discrepancy measure, one popular training scheme for SNMF is as follows:

$$\begin{aligned} (D, G) \quad = \quad & \arg \min_{D \geq 0, G \geq 0} \frac{1}{2} \|X - DG\|_F^2 + \beta \|G\|_1 \\ & \text{s.t. } \|d_m\| \leq 1, \forall m \end{aligned} \tag{1.9}$$

where $\|G\|_1 = \sum_{i,j} |G_{i,j}|$ denotes the sum of all absolute values of entries of G. It is known as the L_1 norm of G, if we view G as a two-dimensional vector. To avoid scaling ambiguity, each column of D is constrained to has a norm that is less than or equal to one. Without the bounded norm constraint, one can arbitrarily scale up D and scale

down G such that DG remains the same and $\|G\|_1$ decreases arbitrarily close to zero. This scaling will decrease the objective (1.9) without either learning a better dictionary or increasing sparsity. Using L_1 norm regularization is known to promote sparsity [17]. This is because L_1 norm is the tightest convexification of the L_0 norm, which is defined as the number of non-zero entries.

NMF-based Approaches for Speech Enhancement or Source Separation

NMF has been used for speech enhancement (e.g., [18–21]) and audio source separation (e.g., [22–24]). When applying NMF, the first step is to train a dictionary for speech and/or noise. To train a dictionary for speech, training utterances are first transformed into the TF representation. For example, let $\mathbf{X}_i \in \mathcal{R}^{K \times N_i}$ be the complex spectrogram of the ith training utterance, and let $X = [|\mathbf{X}_1|, |\mathbf{X}_2|, \cdots]$ denotes the spectrogram magnitude of *all* training utterances. To capture the spectral structure of X, (1.8) or (1.9) is used to learn a dictionary. A dictionary for noise can be trained in the same way. The learned dictionaries are then used for source separation or speech enhancement. Two most important techniques used by researchers are the *sparsity* of the gain matrix and the *temporal information* of speech. However, these two techniques were used differently by different researchers. Here I summarize a few, but it is by no means comprehensive.

Sparsity is widely exploited in existing works. For example, in [22], Schmidt and Olsson use sparsity for separating multiple speech sources from a single microphone recording. In the training stage, a speaker-dependent dictionary is trained using sparse NMF (1.9). In the separation stage, they decompose the received mixture by using the learned dictionary. Take 2 speaker scenario as an example. Let D_1 be the dictionary for the first speaker and D_2 be the one for the second speaker. They decompose the mixture Y into two terms, $D_1\hat{G}_1$ and $D_2\hat{G}_2$, where $\hat{G}_1$ and $\hat{G}_2$ are obtained by the following optimization.

$$[\hat{G}_1, \hat{G}_2] = \arg \min_{G_1 \geq 0, G_2 \geq 0} \|Y - D_1 G_1 - D_2 G_2\|^2 + \lambda \|G_1\|_1 + \lambda \|G_2\|_1 \tag{1.10}$$

Due to the way the dictionaries are trained, $D_1\hat{G}_1$ is treated as the first speaker signal and $D_2\hat{G}_2$ is treated as the second speaker signal. This algorithm is fully supervised,

and heavily relies on sparsity for separating the two sources. D_1 is trained such that utterances of speaker 1 can be sparsely represented. It is possible that D_2 can also represent it; however, it is less likely that it will also be *sparsely* representable by D_2. Therefore, when promoting sparsity in the separation, we are able to separate the utterances of different speakers from a single mixture.

Some researchers exploit sparsity in a semi-supervised fashion. In [18], only the noise dictionary (D_2) is trained offline. In the enhancement stage, the authors optimized (1.10) with respect to G_1, G_2, and D_1. In this setup, the speech dictionary (D_1) is learned *online*, therefore makes it semi-supervised. Similar idea has also been used by Duan *et al.* in [20]. However, Duan uses probabilistic NMF as the discrepancy measure, and they updated the speech dictionary only when speech is present, which requires a Voice Activity Detector (VAD).

Temporal information is undoubtedly a very important characteristics for natural sounds. In [23], Virtanen exploits the temporal continuity by adding a regularizer that penalizes the scaled successive difference of the gain matrix G:

$$c_m \|g_{m,n} - g_{m,n-1}\|^2 \tag{1.11}$$

where $c_m = \sqrt{\frac{1}{N}\sum_{n=1}^{N} g_{m,n}^2}$. This penalization will make the gain matrix G to be slow varying in time. Since $x_n = Dg_n$, the estimated sound source, the x_n, will also be temporally continuous.

Wilson *et al.* [25] also capture the temporal information of X through G like Virtanen. However, the mathematical approach is different. Let $g_{m,:}$ denotes the mth row of G. $g_{m,:}$ captures the temporal variation of the dictionary atom d_m. Wilson *et al.* model the logarithm of $g_{m,:}$ as a multi-variate Gaussian distribution. Temporal information of G is therefore captured in the mean and the covariance of the multi-variate Gaussian distribution. Both mean and variance are learned in the training stage. In the enhancement stage, while keeping the mean and the covariance unchanged, the negative log-likelihood of $g_{m,:}$ is used as an additional regularizer in (1.8).

Mysore *et al.* [24] propose a non-negative factorial hidden Markov model (N-FHMM) to capture the spectro-temporal information. It is a temporal extension of the probabilistic NMF [16]. In N-FHMM, speech spectrum magnitude in a given time frame is modeled as a linear combination of the atoms from one (out of many) dictionary.

Which dictionary to use in a given time frame is defined as the "state" of that time frame. To capture the temporal characteristics, N-FHMM models the state transition by a first order Markov model. In the separation stage, the authors minimize the negative likelihood of the probabilistic NMF, while taking the Markov model into account.

Mohammadiha *et al.* [21] also use the probabilistic NMF framework as in [24]. However, Mohammadiha *et al.* use a Kalman filter-like approach to leverage the temporal information. They view g_n as the "state" in time frame n, and assume g_n follows an Lth order vector autoregressive model:

$$g_n = \sum_{l=1}^{L} A_m g_{n-l} + u_n \tag{1.12}$$

where u_n denotes the state transition noise. Taking the state transition (1.12) into account, the authors modify the probabilistic NMF by incorporating a prediction step similar to the Kalman filter.

Pros and Cons of NMF-based Approach

The major advantage of NMF-based approach is that enhancement across different frequency bins is performed *jointly*, wheres in Bayesian-based approach it is performed *independently*. In NMF, the dictionary captures the spectrum characteristics of speech and noise. In other words, the structure across frequency bins is captured in the dictionary learned by NMF. In the enhancement stage, the cross-frequency information is naturally leveraged. However, in most Bayesian algorithms the variances are estimated independently across frequency bins. Therefore, enhancement in different frequency bins are performed independently.

Two major disadvantages of NMF-based approach are the lack of statistical optimality and the high computational complexity. Unlike Bayesian-based approach, most NMF-based approaches perform enhancement or separation by minimizing an objective without well-justified statistical criteria (e.g., mean square error, MAP). Moreover, most NMF-based approaches are computationally more demanding than Bayesian-based algorithms due to the dictionary training step. However, thanks to the advance of computing technologies, the computational complexity of NMF will not be a prohibiting factor.

1.3.3 IM-based Approach

Recent works [7, 26–29] that use Ideal Masking (IM) technique for speech enhancement are reviewed in this section. These algorithms contain a training stage and a testing stage. In the training stage, they train a model for estimating the ideal mask. In the testing stage, the trained model is applied to produce the estimated mask, which is then used for generating the enhanced speech. Despite the subtle differences, these algorithms share a similar system design. We will first present the general system design, and then present the differences.

The General System Design of IM-based Approach

Most IM-based algorithms follow the system design shown in Fig. 1.4. In the training stage, the clean speech $x(t)$ and the additive noise $v(t)$ are transformed into the TF representation through the *analysis* and the *TF* calculation module. The purpose of the analysis module is to separate the audio stream into several subbands. For example, the Gammatone filterbank is used for cochleagram computation. The two TF representations, X and V, are used to calculate the ideal mask R. Different researchers define the "ideal mask" differently. The major computational goal in the training stage is to train a system, which is composed of a *feature extraction* step and a *mask estimation* step, that can reliably estimate the ideal mask. Different researchers design these two blocks differently. The estimated mask $\hat{R}$ is then used to produce the enhanced speech $\hat{x}$. In the testing stage where only the noisy speech is available, the enhancement is performed by applying the trained system to the noisy speech.

Differences between Different IM-based Algorithms

In this section, we highlight the differences between existing IM-based algorithms. Please see Table 1.2 for the comparison summary.

In the first step, time-domain audio sources are decomposed into different frequency bands through the *analysis* module. In [7], they use a 25-channel filterbank with mel-frequency spacing. While [28] uses a 32-channel Gammatone filterbank, [26, 27, 29] use a 64-channel Gammatone filterbank. The more the frequency bands, the finer the frequency resolution. However, it increases the computational complexity for mask

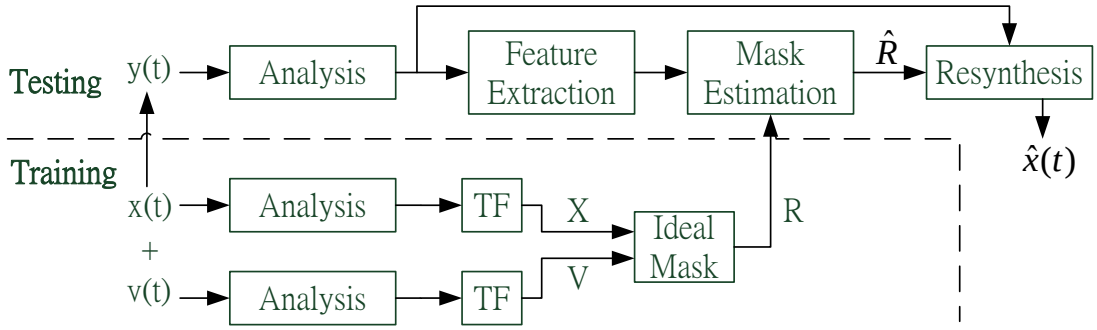


Figure 1.4: Block diagram of the general system design of IM-based approach.

estimation, since the dimension of the mask for each time frame equals to the number of frequency bins.

In the second step, *TF* representation is computed from the output of the filterbank. In short, it calculates the energy within a short period of time. For example, [7] calculates the energy within 32 ms, while others [26–29] consider 20 ms duration.

Let $X \in \mathcal{R}_+^{K\times N}$ and $V \in \mathcal{R}_+^{K\times N}$ be the TF representation of the speech signal and the noise signal, where K is the number of frequency bands, and N denotes the number of time frames. For example, using the setup in [26], K equals to 64 and each time frame is equivalent to 20 ms segments. There are several ways to define the ideal mask. For example, the Ideal Binary Mask (IBM) is typically defined as the dominance of each TF unit:

$$B_{k,n} = \begin{cases} 1, & X_{k,n} \geq V_{k,n} \\ 0, & X_{k,n} < V_{k,n} \end{cases} \tag{1.13}$$

Below is a popular definition for a soft ratio mask, though other definitions have been proposed:

$$R_{k,n} = \frac{X_{k,n}}{X_{k,n} + V_{k,n}} \tag{1.14}$$

The main idea of these ideal masks is to indicate which TF units are speech-dominant and which are noise-dominant. If we can estimate the masks accurately, the enhancement can be performed by preserving the speech-dominant units and attenuating the

Table 1.2: Summary of IM-based algorithms. Please refer to Table B.1 for the full name of the acronyms below.

	Analysis	TF	Ideal Mask	Feature per TF unit (dimension)	Mask Estimation
[7]	25-channel filterbank	energy in 32 ms	binary	AMS(45)	GMM + Bayesian
[26]	64-channel filterbank	cochleagram (energy in 20 ms)	binary	ACF(6)+AMS(45)	SVM
[27]	64-channel filterbank	cochleagram	binary	AMS(15)+RASTA-PLP(13) +MFCC(31)+Pitch-based(6)	DNN+SVM
[28]	32-channel filtrerbank	cochleagram	binary	MRCG(256)	NN
[29]	64-channel filterbank	cochleagram	soft ratio	AMS(15)+RASTA-PLP(13) +MFCC(31)+Pitch-based(6) +GF(1)	DNN

noise-dominant units.

The feature extraction and the mask estimation are arguably the two most important components for any classification system. All existing works use domain-specific features that are originally designed for CASA purpose, e.g., AMS, RASTA-PLP, MFCC, etc. To calculate these features, an extensive domain knowledge about speech is needed. Except for [28] which uses the same 256-dimensionanl feature for all 32 frequency bins, other works extract distinct features for *each* TF unit. As for the mask estimator, several state-of-art estimators have been considered, e.g., the Gaussian Mixture Modeling along with Bayesian inference [7], the support vector machine [26], and the recently popular deep neural network [27,29]. Different estimators have their own pros and cons. While SVM is computationally efficient, it generally performs worse than a well-trained DNN. However, training a DNN typically requires a large training set and a careful implementation of the back propagation with some optimization techniques.

Pros and Cons of IM-based Approach

The major advantage of IM-based algorithms in comparison to most Bayesian-based and NMF-based algorithms is the significant improvement in intelligibility. However,

the improved intelligibility comes with the price of increased training complexity. To reliably estimate the ideal mask, IM-based algorithms require a lot of training data and a powerful classifier. For example, a DNN is used as the classifier in [29]. The training complexity can be several orders of magnitude higher than NMF-based and Bayesian-based algorithms. However, it is worth noting that as the training complexity gets higher, the complexity in the testing stage is actually comparable to NMF-based algorithms. This is because "applying" the classifier is computationally efficient.

1.4 Summary of the Proposed Algorithms

In this thesis, we propose three algorithms for single-channel speech enhancement. These algorithms improve upon the existing works by properly combining or modifying techniques from different approach. Below is a short summary, which presents the key points and highlights the major contribution of each algorithm.

1. Wiener Non-negative Matrix Factorization (WNMF):
   WNMF combines Wiener filtering and NMF for speech enhancement. In WNMF, we use NMF to train a speech model that captures the distinct speech characteristics in the spectrogram domain. In the testing stage, we formulate it as a constrained optimization problem, where the objective is to minimize the MSE, which is basically Wiener filtering, and the constraints enforce the enhanced speech to be sparsely representable by the speech model. In other words, we refine the Wiener filtering by leveraging the speech-specific characteristics captured in the speech model.

   The contribution of WNMF is the way in which we combine Wiener filtering and NMF. In conventional Wiener filtering, no speech-specific structure is leveraged. However, WNMF intelligently leverages the speech structure by using the speech model learned by NMF as the constraint. As a result, the performance improved significantly. Compared to other NMF-based algorithms, WNMF is different in how NMF is used. In most NMF-based algorithms, they perform enhancement from a source separation perspective. Therefore, a speech model and a noise model need to be pre-trained. However, we use NMF to refine the Wiener filtering. Therefore, only a speech model is needed. Since in most practical scenarios, the

testing noise is not accessible in the training phase. Pretraining a noise dictionary is usually not possible.

2. Sparse Gaussian Mixture Model (SGMM):
   Sparse Gaussian Mixture Model (SGMM) generalizes the traditional complex Gaussian model and the NMF model. In comparison to existing models, SGMM better captures the complex speech structure while avoids being overly representative. We apply SGMM to monaural speech enhancement, and propose a semi-supervised enhancement algorithm. In the enhancement stage, only the speech model (the SGMM) is learned in a supervised manner. The noise model is learned using the existing noise variance tracking module.

   The major contribution of SGMM is how we impose sparsity. Unlike most sparsity-promoting algorithms that use a regularization in the objective, we enforce it in the constraint by using a l_0-norm. This constraint explicitly enforce the sparsity. We propose an efficient iterative algorithm to train the SGMM. Despite the fact that l_0-norm is non-convex and non-smooth, the proposed algorithm admits a closed form update and generates non-increasing objective values. This is mainly due to the special structure of the objective in the SGMM training.

3. Sparse Non-negative Matrix Factorization + Deep Neural Network (SNMF-DNN):
   SNMF-DNN is an IM-based algorithm. In particular, we use sparse NMF (SNMF) to extract features from the noisy speech, and use a DNN to perform classification and regression. In its basic version, the SNMF-DNN-Binary, we use both the speech and the noise data to train the model. However, the need for noise training data is dropped in our second version, the SNMF-DNN-Ratio. We integrate an online noise tracking module in SNMF-DNN-Ratio such that it can adapt to unseen noises. Due to the high complexity in training a DNN, the training stage of SNMF-DNN is time-consuming. However, the complexity in the testing stage is actually comparable to the other NMF-based algorithms. Moreover, our simulation result shows that SNMF-DNN is able to substantially improve not only the quality, but also the intelligibility of the noisy speech.

   In the existing IM-based algorithms, they use handcrafted features that require

extensive domain knowledge. However, in SNMF-DNN we show that a comparable performance can be achieved by using a simple SNMF for feature extraction. SNMF requires only minimal domain knowledge and can be efficiently implemented. Moreover, using NMF as a way for feature extraction appears to be new in the speech enhancement community. All the NMF-based algorithms use NMF for monaural source separation.

Chapter 2

Preliminaries

In this chapter, we present various performance measures for speech enhancement algorithms. Although subjective listening is still the ultimate criterion for evaluating the speech enhancement performance, it is very time consuming and difficult to standardize. Therefore, objective measures have been proposed. We will discuss several commonly-used objective measures, which will be used for performance evaluation in this thesis. We divide the objective measures into three categories. In the first category, we discuss measures designed for evaluating the perceived speech quality. The second category evaluates the perceived speech intelligibility. While the previous two categories focus on the human perception perspective, the third category takes the Blind Audio Source Separation (BASS) perspective.

2.1 Quality Measure

Improving the perceived speech quality is the main objective for speech enhancement. High quality is usually described as "natural, light speech degradation, and little noticeable background noise". Though there are many quality measures, we focus on two particular metrics, the Perceptual Evaluation of Speech Quality (PESQ) [30] and the composite measures [31], due to their popularity and high correlation with subjective evaluation.

2.1.1 PESQ

PESQ [30] is designed for telecommunication and is accepted as the International Telecommunication Union (ITU-T) recommendation P.862. Compared to the old measures such as the Perceptual Speech Quality Measure (PSQM) [32] and the model based on measuring normalizing blocks [33], PESQ better models a wider range of communication network conditions, including analogue connections, codecs, packet loss and variable delay. To compute the PESQ, the reference signal and the compared signal are first appropriately equalized and aligned. Then, auditory transformation and cognitive modeling is used to compute the PESQ value. PESQ ranges between 0.5 (bad quality) and 4.5 (high quality). Interested readers may refer to [30] for more implementation detail. Despite that PESQ has been proposed 15 years ago, it is still one of the most competitive objective measures, according to the subjective study conducted in 2008 [31]. The MATLAB implementation provided in [5] is used for calculating PESQ in this thesis.

2.1.2 Composite Measures

In [31], researchers propose three composite measures for measuring three aspects of the perceived speech quality, i.e., the signal distortion, the background noise, and the overall quality. These measures, which are denoted as C_{sig}, C_{bak}, and C_{ovl}, are obtained by properly combining four existing measures, i.e., the Log Likelihood Ratio [34], the PESQ [30], the weighted-slope spectral [35], and the segmental SNR. C_{sig} is used to measure the degree of signal distortion. 5 means very natural with no degradation, and 1 means very unnatural and very degraded. C_{bak} measures the intrusiveness of the background noise, where 5 denotes the background noise is not noticeable and 5 denotes very conspicuous and intrusive. The last composite measure, C_{ovl}, is used to evaluate the overall perceived quality. The higher the value, the better the perceived quality. The composite measures give us a more comprehensive evaluation than using a single quality measure. The MATLAB implementation provided in [5] is used for calculating the composite measures.

2.2 Intelligibility Measure

Besides improving the speech quality, improving the speech intelligibility is another main objective for speech enhancement. In some communication applications where the major purpose is to *understand* the target speech, improving the intelligibility is even more important than the quality. In this section, we will present two intelligibility measures, the I_3 [36] and the Short-Time Objective Intelligibility measure (STOI) [37].

2.2.1 I_3

I_3 ranges between 0 (low intelligibility) and 1 (high intelligibility). It is an improved version the speech intelligibility index (SII) (ANSI S3.5-1997). SII is developed to measure the intelligibility of the noisy speech subjecting to linear filtering and additive noise. It first divides the clean speech and the noisy speech into 20 bands. In each band, the weighed average of SNRs is computed. SII is obtained by using the Band-Importance Functions (BIFs) to weight the SNRs of each band. However, SII does not model the effect of non-linear distortion during the communication transmission. Also, it cannot be applied to non-stationary noises, such as the competing talker scenario, since SII is calculated using the long-term average spectra. To fix these issues, Kates *et al.* [36] proposed I_3 by properly modifying SII. There are two major differences between I_3 and SII. First, while SII calculates the SNRs of each band, I_3 uses the Signal-to-Distortion Ratio computed by the coherence function between the processed speech and the clean speech. Second, I_3 first compute three intermediate measures, CSII (high), CSII (mid), CSII (low), based on the Root-Mean-Square (RMS) level of the clean speech. The I_3 measure is obtained by properly combining the three intermediate measures. According [38], I_3 is highly correlated (correlation coefficient equals to 0.92) with the subjective evaluation. The MATLAB implementation provided in [5] is used for calculating I_3.

2.2.2 STOI

Most intelligibility measures are developed for speeches under natural degradation (e.g., additive noise, reverberation, filtering and clipping). They are not designed for evaluating the intelligibility of speeches *after* noise reduction algorithms. In order to compare

different enhancement algorithms, measures that can evaluate the intelligibility after applying enhancement algorithms is of great need. STOI proposed in [37] is developed for this purpose. To compute STOI, the enhanced speech and the reference speech are first transformed into TF representations. After proper normalization and clipping, TF-dependent correlation coefficients between the two are computed. STOI is computed by first averaging the correlation coefficients over time and over frequency bins, and then using logistic function to map to the range between 0 and 1. The higher the STOI, the better the intelligibility. Despite its simplicity, it shows the highest correlation coefficient with the subjective evaluation for predicting the intelligibility of noisy speeches processed by conventional noise reduction algorithms [37]. The implementation provided in [37] is used in this thesis.

2.3 Blind Audio Source Separation Measure

Single-channel speech enhancement can be viewed as a special case of BASS, while one source is the desired speech and the other source is the noise. Therefore, we also use the BASS measures proposed in [39] for evaluating the performance of speech enhancement algorithms. Three measures are proposed in [39], which are the Source-to-Distortion Ratio (SDR), the Signal-to-Interference Ratio (SIR), and the Source-to-Artifacts Ratio (SAR). Let s be the vector of the desired speech in the time domain, v be the additive noise, and y be the noisy observation. To compute SDR, SIR, and SAR, the enhanced speech $\hat{y}$ in the time domain is first decomposed as a sum of three components:

$$\hat{y} = \hat{s} + \hat{v} + n$$

where $\hat{s}$ is the speech component, $\hat{v}$ is the noise component, and n is the residual artifacts. The three measures are defined as follows:

$$\begin{aligned}
\text{SDR} &:= 10 \times \log_{10} \left\{ \frac{\|\hat{s}\|^2}{\|\hat{v} + n\|^2} \right\} \\
\text{SIR} &:= 10 \times \log_{10} \left\{ \frac{\|\hat{s}\|^2}{\|\hat{v}\|^2} \right\} \\
\text{SAR} &:= 10 \times \log_{10} \left\{ \frac{\|\hat{s} + \hat{v}\|^2}{\|n\|^2} \right\}
\end{aligned}$$

The higher the SDR, SIR, and SAR, the better the enhanced performance. By using three SNR-related measures, we can better understand the performance of the enhanced speech. Compared to the quality and intelligibility measures which take human auditory system into account, these BASS measures take a pure mathematical point of view. Another difference is that SDR, SIR, and SAR are calculated in the time domain, while other perceptual-related measures are usually calculated in the TF domain. In this thesis, we use BASS measures to replace the conventional SNR measure. The MATLAB toolbox provided by [39] is used in our simulation.

Chapter 3

Wiener Non-negative Matrix Factorization

In this chapter, we propose a WNMF algorithm that combines Wiener filtering and non-negative matrix factorization (NMF). The basic idea is to learn a speech model via NMF in the training stage. In the enhancement stage, the trained speech model is used to refine the Wiener filtering result. Depending on how the model is trained and how it is used, we propose two variants, i.e., the causal WNMF and the non-causal WNMF. In the following, we will first present the two variants, and then compare their enhancement performance.

3.1 Causal WNMF

Causal WNMF is divided into two stages, the dictionary training stage and the enhancement stage. In the dictionary training stage, we use a sparse NMF to learn a dictionary that can sparsely represent speech spectrums. In the enhancement, the enhanced speech is obtained by minimizing the mean square error (i.e., Wiener filtering) subject to sparse NMF constraint. If the speech and the noise variances are estimated correctly, Wiener filtering alone is good enough. However, both variances are estimated with error in practice. Leveraging sparse NMF within the Wiener filtering framework helps to mitigate the artifacts of inexact variance estimation. This will be observed in the simulation.

3.1.1 Dictionary Learning Stage

Let $X \in \mathcal{R}_+^{K\times N}$ denotes the collections of clean speech spectral magnitude, where K denotes the number of frequency bins and N denotes the number of time frames. In the dictionary training stage, sparse NMF is used to learn a dictionary $D \in \mathcal{R}_+^{K\times M}$ that can sparsely represent X, where M is the total number of atoms in the dictionary:

$$\begin{aligned}(D,G) &= \arg\min_{D\geq 0, G\geq 0} \frac{1}{2}\|X-DG\|_F^2 + \beta\|G\|_1 \\ &\quad \text{s.t. } \|d_m\| \leq 1, \forall m\end{aligned} \tag{3.1}$$

where $d_m \in \mathcal{R}_+^{K\times 1}$ denotes the mth column of the dictionary, and

$$\|G\|_1 = \sum_{m,n} |G_{m,n}|$$

The L_1 norm regularization of G is known to promote sparsity [17]. Unlike other NMF-based algorithms, which require training dictionaries for both speech and noise, we only train a speech dictionary. The reason is because the noise part is being taken cared of by the Wiener filter. The speech dictionary is used to "refine" the Wiener filter result. This will be more clear when we introduce the enhancement stage.

To solve (3.1), we tailor the Block Successive Upper-bound Minimization (BSUM) framework [40] for this problem. In a nutshell, D and G are alternatively updated by constructing an auxiliary upper bound. Moreover, both updates admit closed form, which makes the training process computationally efficient. The overall training algorithm is summarized in Alg. 1, where $\sigma_{max}(A)$ denotes the largest singular value of the matrix A, and $\mathcal{P}_{\mathcal{A}}(\cdot)$ denotes the projection to the set $\mathcal{A}$. The two projection operators needed are:

$$\mathcal{P}_+ = \{X \mid G_{m,n} \geq 0,\ \forall m,n\}$$
$$\mathcal{P}_{\|\cdot\|\leq 1} = \{X \mid \|X_m\|_2 \leq 1,\ \forall m\}$$

It is not difficult to show that both projection operators can be performed in closed form. Moreover, according to the BSUM theory [40], the iterates generated by Alg. 1 converge to the set of stationary points of the original problem (3.1).

Algorithm 1 Sparse Dictionary Learning for solving (3.1)

1: Initialize D randomly s.t. $\|d_m\|_2 \leq 1,\ \forall m$
2: **repeat**
3: G update:

$$G = \mathcal{P}_+ \left(G - \frac{1}{\tau_G} D^T (DG - X) \right)$$

where $\tau_G = \sigma^2_{max}(D)$
4: D update:

$$D = \mathcal{P}_{\|\cdot\| \leq 1} \left(\mathcal{P}_+ \left(D - \frac{1}{\tau_D} (DG - X) G^T \right) \right)$$

where $\tau_D = \sigma^2_{max}(G)$
5: **until** some convergence criterion is met

3.1.2 Enhancement Stage

In this section, we will present the enhancement stage of the causal WNMF that is summarized in Alg. 2. The main contribution is step 5, where we combine Wiener filtering with sparse NMF as a constrained optimization problem.

In step 1 of Alg. 2, we transform the noisy recording into the spectrogram domain. In step 3 to step 5, we calculate the estimated speech spectrum $\hat{x}_n$ as soon as we receive $\mathbf{y}_n$. Therefore, it is a causal algorithm. In step 3, we estimate the noise variance at all frequency bins in time frame n. This can be done using the minimum statistics-based algorithm proposed in [2] or the MMSE-based algorithm proposed in [4]. In step 4, we estimate the *a priori* SNR $\xi_{k,n}$ and the *a posteriori* SNR $\gamma_{k,n}$. In (3.3), we use the DD method for estimating the *a priori* SNR, while the forgetting factor $\alpha_{k,n}$ is determined adaptively (3.4). This adaptive rule is proposed in [41], and works well in our context.

In step 5, we intelligently combine the Wiener filter with the speech dictionary, which we call the "Sparsity-based Wiener plus Dictionary Learning" (SWDL). SWDL, as shown in (3.5), minimizes the mean square while enforcing the enhanced speech spectrum to be sparsely representable by the speech dictionary. Minimizing the mean square error subject to (3.6) is exactly Wiener filtering. To leverage the speech structure, we require the enhanced spectrum to be representable by D in (3.7). Moreover, we want

Algorithm 2 Causal WNMF

Require: : noisy speech $y(t)$, dictionary D, sparsity parameter λ

1: $\mathbf{Y} = Y \bigodot \exp(j\theta) = \text{STFT}(y(t))$

2: **for** $n = 1 \rightarrow N$ **do**

3: Estimating noise power $\sigma^2_{k,n}$ for all $k = 1, \cdots, K$

4: Estimating $\xi_{k,n}$ and $\gamma_{k,n}$ or all $k = 1, \cdots, K$

$$\begin{aligned} \gamma_{k,n} &= \frac{Y^2_{k,n}}{\sigma^2_{k,n}} && (3.2) \\ \xi_{k,n} &= \alpha_{k,n}\frac{\hat{X}^2_{k,n-1}}{\sigma^2_{k,n}} + (1-\alpha_{k,n})\bar{\gamma}_{k,n} && (3.3) \\ \alpha_{k,n} &= \frac{1}{1+\left(\frac{\bar{\gamma}_{k,n}-\xi_{k,n-1}}{1+\bar{\gamma}_{k,n}}\right)^2} && (3.4) \\ \bar{\gamma}_{k,n} &= \max[\gamma_{k,n}-1, 0] \end{aligned}$$

5: Estimating $\hat{x}_{k,n}$

$$\begin{aligned} \hat{x}_n &= D \times \hat{g}_n \\ \hat{g}_n &= \text{SWDL}(\{y_{k,n}, \gamma_{k,n}, \eta_{k,n}\}_{k=1}^K, D, \lambda) \qquad [Eq.(3.12)] \end{aligned}$$

6: **end for**

7: Enhanced STFT: $\hat{\mathbf{X}} = \hat{X} \bigodot \exp(j\theta)$

8: **return** Enhanced speech: $\hat{x}(t) = \text{IFFT}(\hat{\mathbf{X}})$

the representation to be sparse, therefore we penalize the L_1 norm of the gain coefficient in the objective. λ controls the sparsity level of g, i.e., the larger the λ, the sparser the g. We will discuss how to select λ in the simulation section. In SWDL, the speech dictionary is used to refine the result of Wiener filtering by using the fact that clean speech spectrum magnitude can be sparsely represented by D. Therefore, a "good" enhanced spectrum should also be sparsely representable by D.

$$
\begin{aligned}
[\hat{x}_n, \hat{h}_n, \hat{g}_n] &= \arg\min_{x,h,g\geq 0} \sum_{k=1}^{K} J_{MSE}(h_{k,n}) + \beta\|g\|_1 && (3.5)\\
&\text{s.t.} \quad x = h \bigodot y_n && (3.6)\\
&\qquad x = D \times g, \quad g \geq 0 && (3.7)
\end{aligned}
$$

where $J_{MSE}(h_{k,n}) = \mathbb{E}\left[\frac{1}{2}|h_{k,n}\mathbf{x}_{k,n} - \mathbf{y}_{k,n}|^2\right]$ denotes the mean square error.

At the first glance, (3.5) seems to be difficult to solve since it has two linear constraints and a non-negativity constraint. Due to the special structure of this problem, these two linear constraints can actually be removed and SWDL can be reduced to a simple non-negative quadratic minimization that can be efficiently solved. The first linear constraint (3.6) defines a one-to-one mapping between x and h. Therefore, it can be eliminated by representing h in terms of x:

$$h_{k,n} = \frac{x_k}{y_{k,n}}, \ \forall k = 1, \cdots, K \tag{3.8}$$

By substituting (3.8) into (3.5), we can rewrite SWDL as follows:

$$
\begin{aligned}
[\hat{x}_n, \hat{g}_n] &= \arg\min_{x,h,g\geq 0} \sum_{k=1}^{K} \frac{1}{2}(\eta_{k,n}^2 + \sigma_{k,n}^2)\left(\frac{x_k}{y_{k,n}} - \frac{\xi_{k,n}}{1+\xi_{k,n}}\right)^2 + \beta\|g\|_1 && (3.9)\\
&\text{s.t.} \quad x = D \times g, \quad g \geq 0 && (3.10)
\end{aligned}
$$

Let us define the weighted L_2 norm of a vector $x \in \mathcal{R}^{K\times 1}$ as follows:

$$\|x\|_u^2 = \sum_{k=1}^{K} u_k |x_k|^2 \tag{3.11}$$

We can further remove the last linear constraint and transform SWDL into a non-negative quadratic minimization problem as (3.12). Non-negative quadratic minimization is a well-studied problem in optimization community, and can be solved efficiently by various algorithms including the Nesterov's optimal first order method [42].

$$
\begin{aligned}
\hat{x}_n &= D \times \hat{g}_n \\
\hat{g} &= \arg\min_{g\geq 0} \frac{1}{2}\|Dg - x_n^w\|_{u_n}^2 + \beta\|g\|_1 && (3.12)
\end{aligned}
$$

where $u_n = [\frac{1+\xi_{1,n}}{\gamma_{1,n}}, \cdots, \frac{1+\xi_{K,n}}{\gamma_{K,n}}]^T$, $x_n^w = [\frac{\xi_{1,n}}{1+\xi_{1,n}} y_{1,n}, \cdots, \frac{\xi_{K,n}}{1+\xi_{K,n}} y_{K,n}]$

In step 7, we use the $\hat{X}$ obtained in step 5 as the enhanced spectrogram magnitude, and use the phase of the noisy spectrogram, denoted by θ, as the enhanced spectrogram phase. In step 8, the enhanced time-domain speech signal is obtained by the inverse FFT (IFFT).

3.2 Non-causal WNMF

Here we present the non-causal WNMF algorithm. The main idea is similar to its causal counterpart, that is, using a speech dictionary that captures the speech structure to refine the Wiener filtering result. In non-causal WNMF, however, dictionaries are used to capture the spectro-temporal speech structures over *multiple* STFT frames, while the dictionary in the causal WNMF only captures the speech structure in a *single* STFT frame. When combining the multi-frame dictionaries with Wiener filter, the resulting algorithm becomes non-causal. Since multi-frame dictionaries capture more information than a single-frame dictionary, we expect a superior performance of non-causal WNMF. It is indeed observed in our numerical simulation.

Non-causal WNMF is still divided into two stages, the dictionary training stage and the enhancement stage. The algorithm in both stages can be viewed as an generalization of the causal WNMF.

3.2.1 Dictionary Learning Stage (MMSNMF)

The dictionary training algorithm for non-causal WNMF, which we named as "Multi-class Multi-frame Sparse Non-negative Matrix Factorization" (MMSNMF), is summarized in Alg. 3. It is named as "Multi-class Multi-frame" because in step 1 we combine multiple STFT frames, namely $\{x_{n-2}, x_n, x_{n+2}\}$, into a spectrum patch $\bar{x}_n$. Second, we classify $\bar{x}_n$ into one of the four classes. Followed by it, we use sparse NMF to learn a dictionary for each class. We will elaborate more on the classification part, since it is the main difference compared to previous dictionary training scheme.

Classification in step 1 of Algorithm 3

In most NMF-based algorithms, dictionary is trained to capture the speech structure within a *single* STFT frames. In other words, the temporal structure across *multiple* STFT frames is not captured in the dictionary. In MMSNMF, we want to capture the

Algorithm 3 MMSNMF training

Require: training spectra $\{x_n\}_{n=1}^N$, sparsity parameter β^C and number of dictionary atoms M^C for each class C, where C denotes one of the four classes $\{C_{LL}, C_{LH}, C_{HL}, C_{HH}\}$

Ensure: Dictionaries for four classes: $\{D^{LL}, D^{LH}, D^{HL}, D^{HH}\}$

1: **Classification**:

1. Combine three spectra into a spectrum patch $\bar{x}_n = [x_{n-2}; x_n; x_{n+2}]$
2. For each spectrum patch $\bar{x}_n$, classify it into one of the four classes according to (3.15).

2: **for** each class C **do**

3: Let $\bar{X}^C$ denotes the collection of all spectrum patches in class C.

4: Sparse NMF: Learn a dictionary D^C for class C by solving sparse NMF problem.

$$(D^C, G^C) = \arg \min_{D \geq 0, G \geq 0} \frac{1}{2} \|\bar{X}^C - DG\|_F^2 + \beta^C \|G\|_1 \quad \text{s.t. } \|d_m\| \leq 1, \forall m \tag{3.13}$$

5: **end for**

temporal structure across time as well as the spectral structure across frequency bands. For this purpose, we first concatenating three spectral frames into a *spectrum patch* $\bar{x}_n$:

$$\bar{x}_n = [x_{n-2}; x_n; x_{n+2}] \tag{3.14}$$

The reason for considering $(n-2, n, n+2)$ instead of $(n-1, n, n+1)$ is because $(n-2, n, n+2)$ corresponds to three non-overlapping time-domain speech segments, since we use 50% overlapped STFT. Clearly, $\bar{x}_n$ contains richer spectro-temporal information than a single spectrum frame x_n, and the dimension of $\bar{x}_n$ is three times larger than x_n.

Instead of using sparse NMF to train a dictionary for all spectrum patches, we decide to classify spectrum patches into four classes, and train a dictionary for each class. The purpose of the classification is two-fold, *i).* to build a better dictionary and *ii).* to reduce training complexity. Due to the classification, dictionaries can now be specialized to represent its own class of spectrum patches, which makes it better than using a single dictionary to represent all spectrum patches. Since each dictionary is only responsible for sparsely representing a specific type of spectrum patches, a smaller size can be used. This will reduces the training complexity. Moreover, dictionaries for each classes can be trained in parallel, which reduces the total training time further.

To classify $\bar{x}_n$, we first divide x_n into two types: the *low-frequency dominant frame (LF)* if the energy in low frequency bands ($0-3$ kHz) is larger than the energy in high frequency bands ($3-8$ kHz), and the *high-frequency dominant frame (HF)* otherwise. Accordingly, we can classify $\bar{x}_n$ into four classes depending on the spectral transition from x_{n-2} to x_{n+2}:

$$\begin{cases} C_{LL}, & x_{n-2} \text{ is LF and } x_{n+2} \text{ is LF} \\ C_{LH}, & x_{n-2} \text{ is LF and } x_{n+2} \text{ is HF} \\ C_{HL}, & x_{n-2} \text{ is HF and } x_{n+2} \text{ is LF} \\ C_{HH}, & x_{n-2} \text{ is HF and } x_{n+2} \text{ is HF} \end{cases} \tag{3.15}$$

Obviously, spectrum patches from different classes have different spectro-temporal characteristics. As an illustration, we show the spectrogram of one sentence in Fig. 3.1. Four rectangles in the figure represent four different spectrum patches, each containing three TF frames. For example, the leftmost spectrum patch belongs to class HL since it changes from HF to LF, and the rightmost spectrum patch belong to class HH.

For each class C, we will use the sparse NMF algorithm in (1) to train a dictionary $D^C \in \mathcal{R}^{3K\times M}$. We denote the four dictionaries by $\{D^{LL}, D^{LH}, D^{HL}, D^{HH}\}$.

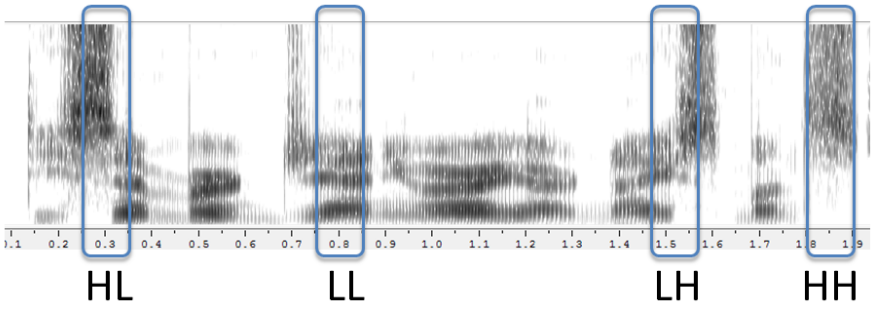


Figure 3.1: A clean speech TF representation that illustrates the concept of spectrum patches and the ST classes defined in (3.15).

3.2.2 Enhancement Stage

Here we present the enhancement algorithm of the non-causal WNMF summarized in Alg. 4. Assuming the four speech dictionaries have been trained, the first step of the

enhancement is to transform the noisy recording into the spectrogram. The noise variance and the speech variance are estimated in step 2 and 3, respectively. The main steps are step 4 and step 5. Step 4 estimates the class of the spectrum patch $\bar{x}_n$, and the enhanced spectrum magnitude is estimated in step 5. We will present these two steps in detail.

Algorithm 4 Non-causal WNMF

Require: noisy speech $y(t)$, dictionaries for four classes $\{D^{LL}, D^{LH}, D^{HL}, D^{HH}\}$, sparsity parameter $\{\lambda_n\}$

1: $\mathbf{Y} = Y \bigodot \exp(j\theta) = \text{STFT}(y(t))$
2: Estimate noise variance $\sigma^2_{k,n}$, for all $k = 1, \cdots, K$ and $n = 1, \cdots, N$
3: Estimate speech variance $\eta^2_{k,n}$ for all for all $k = 1, \cdots, K$ and $n = 1, \cdots, N$

$$\eta^2_{k,n} = \alpha\eta^2_{k,n-1} + (1-\alpha)\max\left(|\mathbf{y}_{k,n}|^2 - \sigma^2_{k,n}, 0\right) \tag{3.16}$$

4: **Class estimation:** Estimate C_n, which is the class for time frame n
5: **WF-MMSNMF:** Estimate $\hat{X}$ by solving WF-MMSNMF [Eq. (3.19) - Eq. (3.21)]
6: Enhanced STFT: $\hat{\mathbf{X}} = \hat{X} \bigodot \exp(j\theta)$
7: **return** Enhanced speech: $\hat{x}(t) = \text{IFFT}(\hat{\mathbf{X}})$

Step 4: Class estimation

Estimating C_n is equivalent to determining whether x_{n-2} and x_{n+2} belongs to LF or HF defined in (3.15). Let k_0, k_3, and k_8 denotes the frequency bands that correspond to 0 kHz, 3 kHz, and 8 kHz, respectively. By definition (3.15), we say x_n belongs to LF if

$$\sum_{k=k_0}^{k_3} |x_{k,n}|^2 \geq \sum_{k=k_3}^{k_8} |x_{k,n}|^2, \tag{3.17}$$

otherwise x_n belongs to HF. However, the true $x_{k,n}$ is not known during the enhancement process. Therefore, we propose to use the expected value instead. The left hand side of

(3.17) becomes the following:

$$\begin{aligned} & \mathbb{E}\left[\sum_{k=k_0}^{k_3} |x_{k,n}|^2 \mid y_{k,n}, \sigma_{k,n}, \eta_{k,n}, \text{for all } k\right] \\ = & \sum_{k=k_0}^{k_3} \mathbb{E}\left[|x_{k,n}|^2 \mid y_{k,n}, \sigma_{k,n}, \eta_{k,n}\right] \\ = & \sum_{k=k_0}^{k_3} \frac{\xi_{k,n}}{1+\xi_{k,n}} \left(\frac{1}{\gamma_{k,n}} + \frac{\xi_{k,n}}{1+\xi_{k,n}}\right) y_{k,n}^2 \end{aligned} \tag{3.18}$$

Eq. (3.18) comes from the Gaussian assumption of both the speech and the noise; see [43] for more detail. Similar calculation can be done for the right hand side. Based on the estimation result of x_{n-2} and x_{n+2}, we can estimate C_n by the definition (3.15)

Step 5: WF-MMSNMF

Below shows how to obtain the enhanced speech spectrum $\hat{x}_{k,n}$ by using Wiener filtering and leveraging the four speech dictionaries $\{D^{LL}, D^{LH}, D^{HL}, D^{HH}\}$.

$$\hat{x}_{k,n} = \frac{\hat{\xi}_{k,n}}{1+\hat{\xi}_{k,n}} \tag{3.19}$$

$$\hat{\xi}_{k,n} = \frac{|\hat{h}_{k,n} y_{k,n}|^2}{\sigma_{k,n}^2} \tag{3.20}$$

$$(\hat{H}, \hat{G}) = \arg\min_{H,G} \sum_{k,n} J_{MSE}(h_{k,n}) + \sum_n \lambda_n \|g_n\|_1 \tag{3.21}$$

$$\text{s.t.} \quad \bar{x}_n(H) = D^{C_n} g_n, \tag{3.22}$$

$$g_n \geq 0, \quad 0 \leq h_{k,n} \leq 1$$

where $\bar{x}_n(H) = [h_{n-2} \bigodot y_{n-2}; h_n \bigodot y_n; h_{n+2} \bigodot y_{n+2}]$ denotes the spectrum patch after applying gain H to the noisy spectrum Y. C_n denotes the class for time frame n, which we assume has been estimated.

In (3.21), minimizing the mean square error in the first term promotes $\hat{H}$ to be statistical optimal. This is basically the same as Wiener filtering. In (3.22) we enforce the enhanced spectrum patch $\bar{x}_n(H)$, which is a linear function of H, to be sparsely

representable by the trained dictionary D^{C_n} by: *i)* constraining the enhanced speech to satisfy $\bar{x}_n(H) = D^{C_n} g_n$, and *ii)* penalizing the sparsity-inducing L_1 norm of the coefficient vector g_n. If the instantaneous speech variance $\eta_{k,n}^2$ and the noise variance $\sigma_{k,n}^2$ are estimated perfectly, Wiener filtering alone is good enough. There is no need to impose dictionary constraints. In practice, both speech variance and noise variance are estimated with error that leads to noticeable artifacts. To mitigate these artifacts, we propose to constrain the estimated spectrum patch $\bar{x}_n(H)$ to be sparsely representable by the dictionary D^{C_n}. In the training stage, we already trained dictionaries such that *clean spectrum patches* can be sparsely represented by the dictionaries. It means sparse representability necessarily captures the spectro-temporal information of speech. By enforcing the sparse representability on $\bar{x}_n(H)$, we ensure the enhanced spectrum will be "similar" to the clean spectrum. As a result, the artifacts due to inaccurate variance estimation is mitigated and the enhancement performance is improved.

The non-causalness of WF-MMSNMF comes from (3.22), where h_n appears in $\bar{x}_{n-2}(H)$, $\bar{x}_n(H)$, and $\bar{x}_{n+2}(H)$. As a result, $\hat{h}_n$ depends on not only the past samples, but also the future samples. This non-causalness also prevents us from transforming (3.21) into a simple non-negative quadratic minimization as in the causal counterpart. Therefore, we propose an efficient inexact alternating direction of multipliers (ADMM) for solving (3.21), which will be presented shortly.

Experimentally, we found out that a post Wiener filtering by using $\xi_{k,n} = \frac{|\hat{h}_{k,n} y_{k,n}|^2}{\sigma_{k,n}^2}$ improves the performance. Post processing has been used before [44]. The reason for improved performance may be due to the fact that with limited number of dictionary atoms, some fine details of speech spectrum cannot be well captured using the NMF. Doing post processing partially reduces this artifact.

Inexact ADMM for solving (3.21)

To efficiently solve (3.21), we propose an inexact variant of the Alternating Direction Method of Multiplier (ADMM [45]), which admits closed form updates and is guaranteed to converge to a global optimal solution. ADMM is a popular algorithm for solving optimization with linear equality constraint. Let $L(H, G, U)$ denote the the augmented Lagragian function, where U is the dual variable and $\rho > 0$ denotes the constraint

violation parameter.

$$\begin{aligned} L(H,G,U) &= \sum_n \Bigg\{ \sum_k J_{MSE}(h_{k,n}) + \lambda_n \|g_n\|_1 + \langle u_n, \bar{x}_n(H) - D^{C_n} g_n \rangle \\ &+ \frac{\rho}{2} \|\bar{x}_n(H) - D^{C_n} g_n\|^2 \Bigg\} \end{aligned} \tag{3.23}$$

In iteration r, tthe traditional ADMM is composed of three steps, i.e., the primal updates (3.24) and (3.25), and the dual update (3.26).

$$G^{r+1} = \arg\min_{G \geq 0} L(H^r, G, U^r) \tag{3.24}$$

$$H^{r+1} = \arg\min_{0 \leq H \leq 1} L(H, G^{r+1}, U^r) \tag{3.25}$$

$$u_n^{r+1} = u_n^r + \rho \left(\bar{x}_n(H^{r+1}) - D^{C_n} g_n^{r+1} \right), \quad \forall n \tag{3.26}$$

Because of the structure of $L(H,G,U)$, updating H is separable across k and n and can be computed in closed form. For notational convenience, let us define $[z_{n-2}^n; z_n^n; z_{n+2}^n] \equiv D^{C_n} g_n^{r+1} - \frac{1}{\rho} u_n^r$. Each z_m^n is a $K \times 1$ vector, with $z_{k,m}^n$ denotes its kth frequency component. Updating H can be calculated as follows:

$$\begin{aligned} h_{k,n}^{r+1} &= \arg\min_{0 \leq h_{k,n} \leq 1} \frac{1}{2} \left[(1-h_{k,n})^2 \eta_{k,n}^2 + \mu h_{k,n}^2 \sigma_{k,n}^2 \right] \\ &\quad \frac{\rho}{2} \Big[(h_{k,n} y_{k,n} - z_{k,n}^n)^2 + (h_{k,n} y_{k,n} - z_{k,n}^{n+2})^2 \\ &\quad (h_{k,n} y_{k,n} - z_{k,n}^{n+4})^2 \Big] \end{aligned} \tag{3.27}$$

$$= \left[\frac{(\xi_{k,n} + \mu) h_{k,n}^W + 3\rho\gamma_{k,n} \frac{z_{k,n}^r}{y_{k,n}}}{\xi_{k,n} + \mu + 3\rho\gamma_{k,n}} \right]_0^1 \tag{3.28}$$

where $z_{k,n}^{r+1} = \frac{1}{3}(z_{k,n}^n + z_{k,n}^{n+2} + z_{k,n}^{n-2})$; $\xi_{k,n}$ and $\gamma_{k,n}$ denotes the *a posteriori* SNR and *a priori* SNR, respectively; $h_{k,n}^W = \frac{\xi_{k,n}}{1+\xi_{k,n}}$ is the Wiener filter gain; $[\cdot]_0^1$ denotes the projection operator to the interval between 0 and 1.

The G update (3.24) is separable across n but has no closed form solution. Another iterative algorithm is needed for solving (3.24), which make the traditional ADMM a double-loop algorithm. To gain computational efficiency, we propose to solve the G update *inexactly* by minimizing another local upper-bound function which leads to

closed form solution.

$$\begin{aligned} g_n^{r+1} &= \arg\min_{g_n \geq 0} \frac{\rho}{2}\|\bar{x}_n(H^r) - D^{C_n} g_n\|^2 \\ &\qquad +\langle u_n, \bar{x}_n(H^r) - D^{C_n} g_n\rangle + \lambda_n \|g_n\|_1 \\ &= \arg\min_{g_n \geq 0} \frac{\rho}{2}\|v_n^r - D^{C_n} g_n\|^2 + \lambda_n \|g_n\|_1 \qquad (3.29) \\ &\approx \arg\min_{g_n \geq 0} \rho\langle \nabla^r, g_n - g_n^r\rangle + \frac{\rho L}{2}\|g_n - g_n^r\|^2 + \lambda_n \|g_n\|_1 \qquad (3.30) \\ &= \left[g_n^r - \frac{1}{L}\nabla^r - \frac{\lambda_n}{L\rho} e\right]^+ \qquad (3.31) \end{aligned}$$

where $v_n^r = \bar{x}_n(H^r) + \frac{1}{\rho} u_n^r$, $\nabla^r = -D^{C_n}\left(v_n^r - D^{C_n} g_n^r\right)$, e is an all-one vector, and

$$L > L_n, \forall n \; L_n \text{ is the largest eigenvalue of } D^{C_n}(D^{C_n})^T \qquad (3.32)$$

Since the gradient of (3.29) is Lipschitz continuous with constant L_n, (3.30) is therefore a global upper-bound of (3.29).

The overall inexact ADMM is shown in Alg. 5. It is composed of two primal updates and one dual update. However, unlike the traditional ADMM that solves the G update (3.24) exactly, the inexact ADMM solves it inexactly. As a result, the inexact ADMM is computationally more efficient. Nonetheless, we have not yet establish the theoretical convergence of the inexact ADMM. It turns out that the inexact ADMM also converges to the global optimal solution just like the traditional ADMM [45]. This is proved in Thm. 1. In sum, the proposed inexact ADMM is not only computationally efficient, but also converges with theoretical guarantee. We mention that the idea of inexact ADMM has been studied and applied in other context before [46–48], but using it for speech enhancement appears to be new.

Algorithm 5 Inexact ADMM for solving Eq. (3.21)

1: **for** iteration r **do**
2: G update: closed form as shown in Eq. (3.31)
3: H update: closed form as shown in Eq. (3.28)
4: U update: closed form as shown in Eq (3.26)
5: **end for**

Theorem 1. *For any $\rho > 0$, let $\{H^r, G^r, U^r\}$ denote the sequence generated by Algorithm 5. Starting from any initial point $\{H^0, G^0, U^0\}$, $\{H^r, G^r\}$ converges to a primal optimal solution of* (3.21)*, and U^r converges to a dual optimal solution.*

Proof: The basic idea is to show that the iterates generated by Alg. 5 converges to a primal-dual optimal point of (3.21), i.e., $\{H^\infty, G^\infty, U^\infty\}$ satisfies the Karush−Kuhn−Tucker (KKT) condition of (3.21). Though this proving technique is similar to the standard ADMM proof [45, 49], we still provide a complete and self-contained proof in Appendix C for the ease of readers.

3.3 Simulation for WNMF

In this section, we study the performance of WNMF and other commonly used algorithms under various noisy conditions.

3.3.1 Experiment Setup

The TIMIT database [50] was used for both the dictionary training and the performance evaluation. All sentences are re-sampled at 16 kHz, and are segmented into 30 ms duration frames using a Hamming window with 50 % overlap (thus, a spectrum patch contains 90 msec speech). A 512 point FFT/IFFT are used for time-frequency analysis and synthesis operations (thus $K = 257$).

In the enhancement state, 320 male sentences and 160 female sentences are randomly selected from the TIMIT "test" subset. Computer generated Additive white Gaussian noise (AWGN) and two real world noises (street and babble) from the NOISEX-92 [51] are added to each test sentence at four different SNRs ($-5, 0, 5, 10$ dB). We use the implementation by Prof. Loizou [5] to generate noisy speech with different SNRs. It first determines the active speech level of clean speech signals by using the method B of ITU-T P.56 [52], and then the noise sample is approximately scaled and added to the clean speech to obrain the desired SNR.

The TIMIT "train" subset is used for training dictionaries. In the causal WNMF, we train a single universal dictionary of size $K \times 512$ for all speakers. The sparsity parameter β in (3.1) is experimentally determined to be 0.001. In the non-causal WNMF the dictionary size M of the four dictionaries are fixed to 480, and the sparsity parameter β^C (3.13) is set to 0.01.

In the enhancement stage, we compare four algorithms including two baseline Bayesian-based algorithms, i.e., the Wiener filtering and the LSTSA [3], and two variants of

WNMF. For the causal WNMF, the sparsity parameter λ in (3.12) is automatically determined such that the difference between the enhanced speech and noisy speech roughly equals the estimated noise power. For the non-causal WNMF, we consider two cases. In the first case, we use the "estimated" class, i.e., the class of each spectrum patch used in step 4 of Alg. 4 is estimated by the proposed scheme. This is the practical scenario since class label is not known during the enhancement stage. In the second case we use the "true" class. Alg. 4 is still used, but the estimated class is changed to the ground truth class, which requires access to the clean speech. Even though it is impractical, it can be used to evaluate the performance loss due to class estimation error. The sparsity parameter λ_n in (3.21) is fixed as follows:

$$\lambda_n = 0.01 \times \sqrt{\sum_k \eta_{k,n}^2 + \sigma_{k,n}^2} \tag{3.33}$$

To evaluate the performance of enhanced speech quality, we use the PESQ [30]. It ranges from 1 (poor quality) to 5 (good quality). As for speech intelligibility, we use the I3 [36]. I3 is an objective metric that assesses the intelligibility of the processed speech, and ranges from 0 to 1. The higher the better.

3.3.2 Property of the Learned Dictionaries in Non-causal WNMF

In this section, we study the property of the dictionaries learned in non-causal WNMF. Specifically, we will show that spectrum patches which correspond to different phoneme transitions are represented by different sparse subsets of dictionary atoms. In other words, it means the sparse representation of the dictionary atoms indeed capture the spectro-temporal characteristics of speech signal.

First, we collect three types of spectrum patches that all belong to the spectrum-temporal class LL, but correspond to different phoneme transitions. Let $\bar{X}_{/iy/\to/r/}$ denote the collection of spectrum patches that changes from $/iy/$ to $/r/$; define $\bar{X}_{/iy/\to/aa/}$ and $\bar{X}_{/iy/\to/ae/}$ similarly. Though these three spectrum patches all belong to LL, they have different temporal dynamics because the last phoneme is different. For example, as shown in Fig. 3.2, $/r/$ typically demonstrates a low third formant (F_3), while $/aa/$ and $/ae/$ do not. Furthermore, $/r/$ is often characterized by lower energy compared to $/aa/$ and $/ae/$. Therefore, although $\bar{X}_{/iy/\to/r/}$, $\bar{X}_{/iy/\to/aa/}$, and $\bar{X}_{/iy/\to/ae/}$ all belong to the same ST class, they exhibit very different spectro-temporal structures.

In the second step, we want to find a sparse subset of atoms from D^{LL} that can well represent $\bar{X}_{/iy/\rightarrow/r/}$. For this purpose, we solve the sparse coding problem:

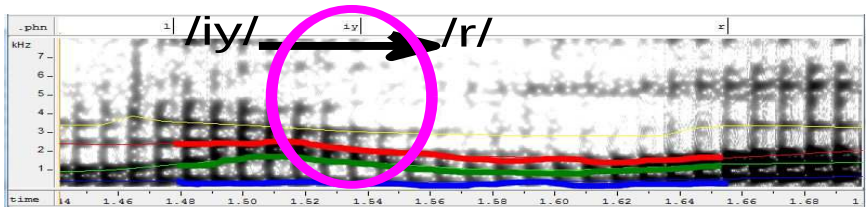
$$[G^*] = \arg\min_{G\geq 0} \frac{1}{2}\|\bar{X}_{/iy/\rightarrow/r/} - D^{LL}G\|_F^2 + \beta\|G\|_1$$

where D^{LL} has already been trained with $M = 480$ and $\beta = 0.01$. We select 50 out of the total 480 rows from G^* that correspond to the 50 largest row sum. These 50 rows thus correspond to 50 most important dictionary atoms for representing $\bar{X}_{/iy/\rightarrow/r/}$. We use $D^{LL}_{/ir/\rightarrow/ir/}$ to denote the collection of these 50 atoms.

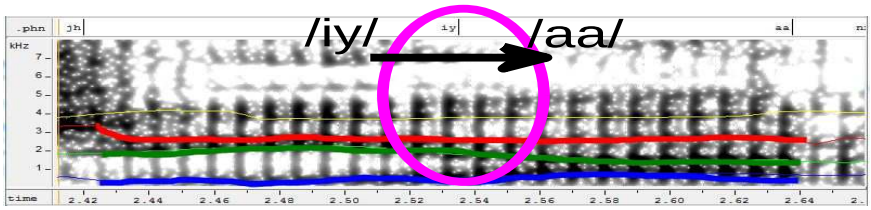
In the third step, we use $D^{LL}_{/iy/\rightarrow/r/}$ to approximate all three phoneme transitions, and calculate the representation error defined in (3.34):

$$\text{Error } (\bar{x}_n) = \min_{g_n} \frac{\|\bar{x}_n - D^{LL}_{/iy/\rightarrow/r/}g_n\|^2}{\|\bar{x}_n\|^2} \tag{3.34}$$

where $\bar{x}_n$ denotes the speech patch that corresponds to any of the three temporal dynamics. Fig. 3.3 shows the histogram of the approximation errors for the three phoneme transitions.



(a) /iy/ → /r/



(b) /iy/ → /aa/

Figure 3.2: Temporal formant patterns in different phoneme transitions. The first, second, and the third formant are drawn in bold blue, green, and red respectively.

Fig. 3.3 shows that using $D^{LL}_{/iy/\rightarrow/r/}$ to approximate $\bar{X}_{/iy/\rightarrow/aa/}$ and $\bar{X}_{/iy/\rightarrow/ae/}$ results in a much higher error than for $\bar{X}_{/iy/\rightarrow/r/}$. In other words, the special sparse subset, $D^{LL}_{/iy/\rightarrow/r/}$, has specifically captured the temporal dynamics of $(/iy/ \rightarrow /r/)$, but not that of the other transitions. Similarly, other temporal dynamics can be captured by different sparse subsets of the learned dictionary. In other words, the sparsity of the linear combination indeed can discriminate different spectro-temporal patterns.

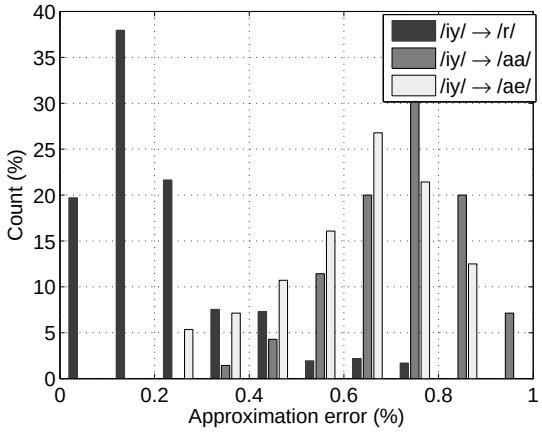


Figure 3.3: Histogram of representation error for using $D^{LL}_{/iy/\rightarrow/r/}$ to represent $\bar{X}_{/iy/\rightarrow/r/}$, $\bar{X}_{/iy/\rightarrow/aa/}$, and $\bar{X}_{/iy/\rightarrow/r/}$. $D^{LL}_{/iy/\rightarrow/r/}$ is obtained by selecting the best 50 atoms from D^{LL} that can best represent $\bar{X}_{/iy/\rightarrow/r/}$

3.3.3 Simulation Result

In Figure 3.4, we present the enhancement result of 5 algorithms under various noisy conditions. Y-axis denotes the average improvement over the non-processed speech; the higher the better. As a reference, the mean metric value of non-processed speech is shown on the figure directly. X-axis denotes different SNRs. In most of the cases, the proposed WNMF, including both the causal and the non-causal version, outperforms both Wiener filtering and LSTSA in terms of both PESQ and I3. This is expected because WNMF leverages speech-specific structure, while the other two do not. For example, in -5 dB AWGN case, using non-causal enhancement algorithm with true class has about 20% PESQ improvements and 80% I3 improvements, while LSTSA only has 7% PESQ improvements and 25% I3 improvements.

Comparing the causal WNMF with the non-causal WNMF, the non-causal version has a better performance. It is reasonable since non-causal WNMF use both past and future speech samples for enhancement.

As for the non-causal WNMF, it is not surprising that using the true class label results in a better performance than using the estimated class label. The performance gap is more pronounced when the SNR is low. This is because class estimation is more erroneous in the low SNR scenario. However, non-causal WNMF with estimated class still outperforms the other algorithms (Wiener, LSTSA, and the causal WNMF) in most of the cases. If a more sophisticated class estimation algorithm is used, the performance gap can be further reduced.

To better visualize the performance, we show the spectrogram of a sample sentence in Fig. 3.5.

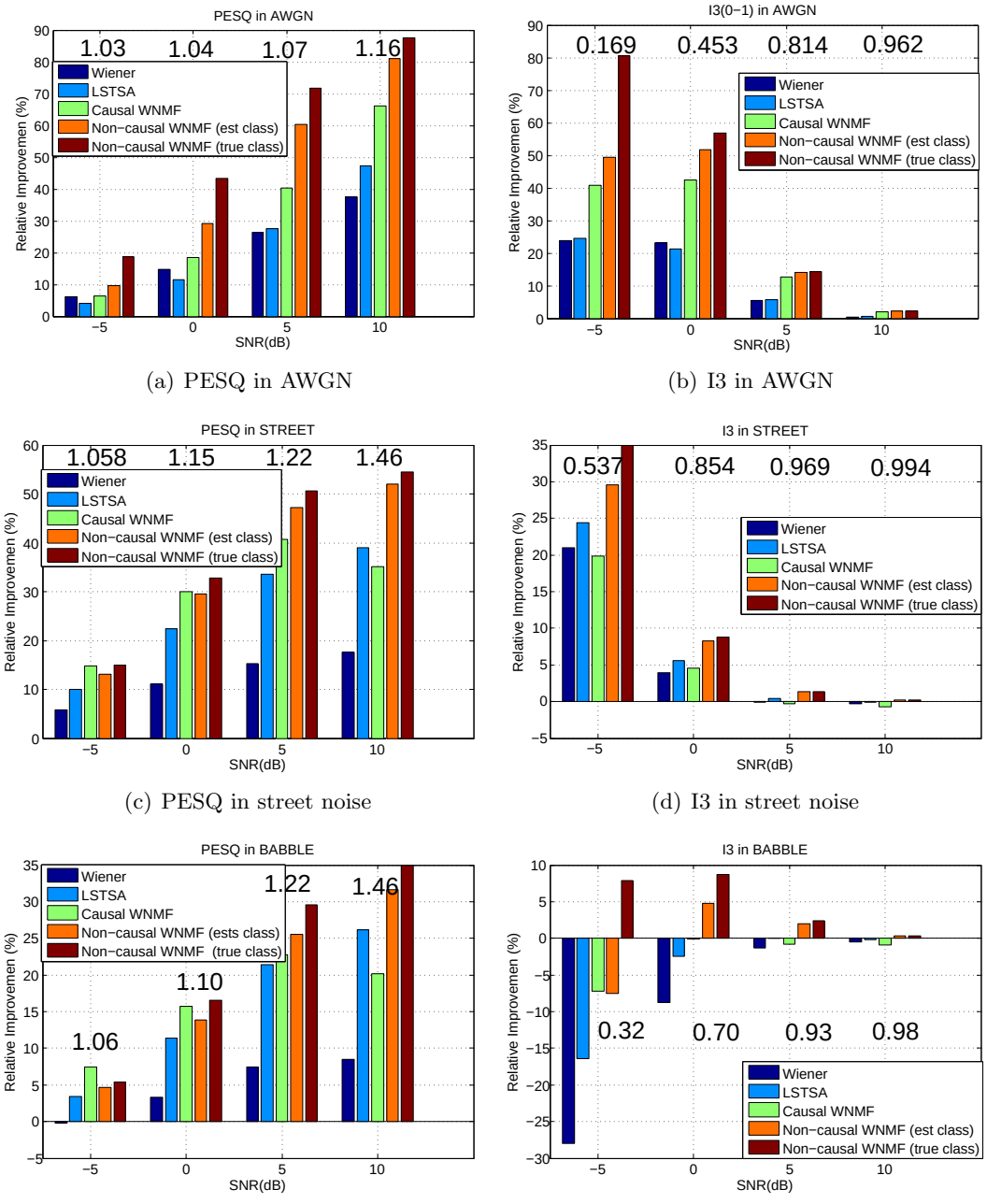
3.3.4 Effect of The Parameters λ, M, and β in Non-causal WNMF

In the previous section, the parameters of the non-causal WNMF are fixed to its default values. In this section, we study the performance when the parameter values change. See Table 3.1 for the definition of λ, M, and β. For simplicity, we only consider the non-causal WNMF with true class label.

Table 3.1: Parameters of Non-causal WNMF

Stage	Parameter	Default value	Equation
Train	$\beta^C = \beta,\ \forall C$	$\beta = 0.01$	(3.13)
	$M^C = M,\ \forall C$	$M = 480$	(3.13)
Enhancement	$\lambda_n = \lambda\sqrt{\sum_k \sigma_{k,n}^2 + \eta_{k,n}^2}$	$\lambda = 0.01$	(3.19)

Fig. 3.6 presents the SDR value when varying the sparsity parameter λ in the enhancement stage, while keeping the dictionary training stage the same. Intuitively, the larger the λ, the sparser the g_n. Since noise is unlikely to be sparsely representable by the dictionaries, making g_n sparse helps removing noise. However, when λ is too large, the MSE term in (3.19) is de-emphasized because too much weight is putting on the L_1 norm term. The optimal λ is thus noise-type dependent and SNR dependent, which is observed in Fig. 3.6(a). When comparing different noise types, the optimal



(a) PESQ in AWGN

(b) I3 in AWGN

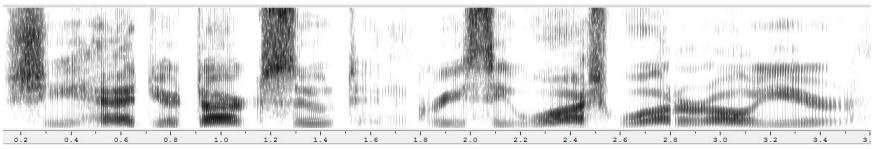
(c) PESQ in street noise

(d) I3 in street noise

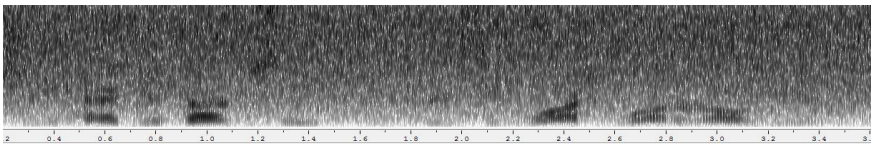
(e) PESQ in babble noise

(f) I3 in babble noise

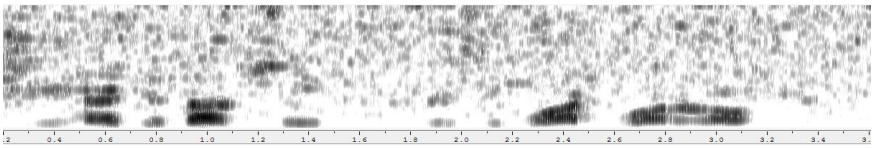
Figure 3.4: Performance (PESQ/I3) at different SNRs under various noise conditions (AWGN, street, babble). Y-axis represents the average metric improvement over non-processed speech. Numbers on each figures shows the average metric values of the non-processed speech.



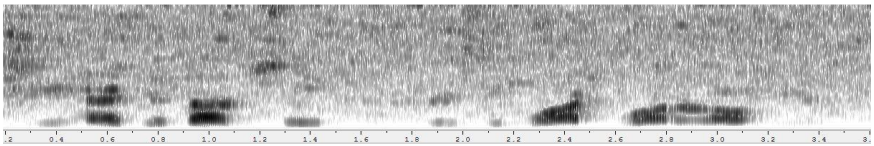
(a) Clean



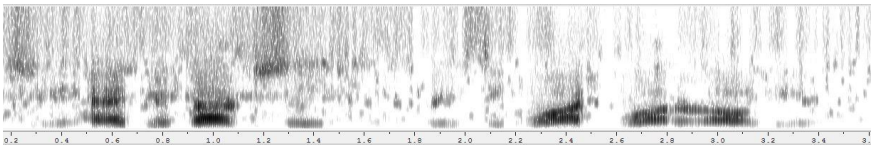
(b) Unprocessed (0 dB AWGN)



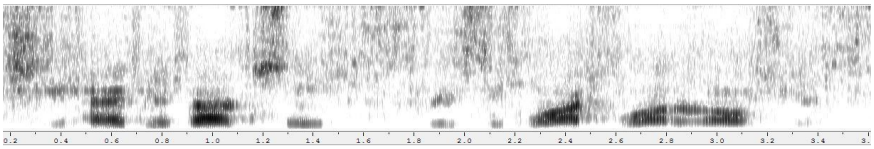
(c) Wiener filtering



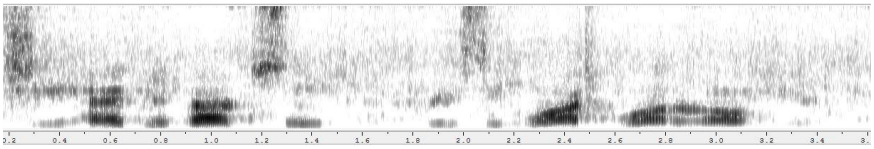
(d) LSTSA



(e) Causal WNMF



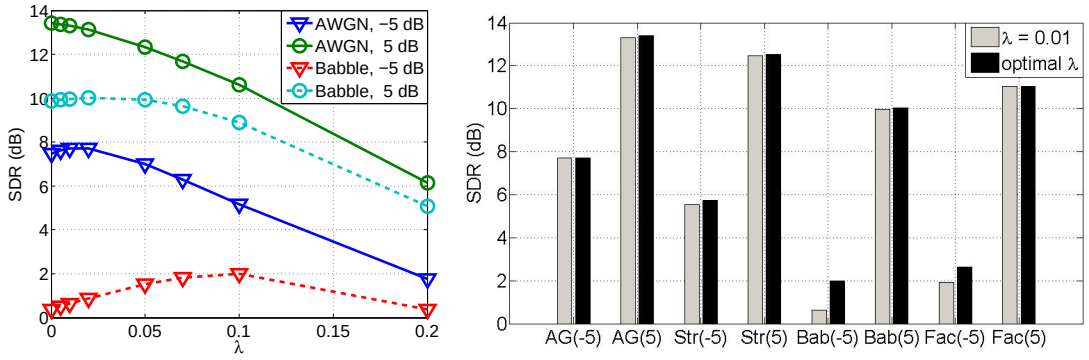
(f) Non-causal WNMF (est class)



(g) Non-causal WNMF (true class)

Figure 3.5: Spectrogram of an sample male utterance is shown in Fig 3.5(a). It is corrupted by 0 dB AWGN as shown in Fig. 3.5(b). Enhancement results of different algorithms are shown from Fig. 3.5(c) - 3.5(g). In each figure, the horizontal axis denotes time in second, and the vertical axis denotes frequency bins (0 Hz - 8 kHz).

λ is larger in Babble noise than in AWGN. This is reasonable because Babble noise is spectrally more similar to human speech than AWGN; therefore, babble noise is more likely to be represented by the learned dictionaries. In order to separate the babble noise from the desired speech, we need to make g_n sparser by increasing λ. Since AWGN is less likely to be represented by the speech dictionary, a small λ suffices to separate AWGN from speech. From Fig. 3.6(b), we see WNMF is not very sensitive to λ. Fixing λ equals to 0.01 gives us a reasonably good SDR value comparing to the optimal λ. The optimal λ is found by enumerating over all λ between 0 and 0.2.

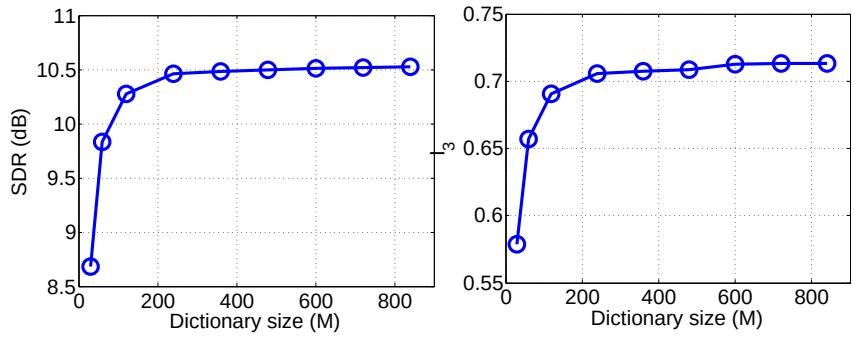


(a) (b)

Figure 3.6: Performance under different noisy conditions with different λ. Fig. 3.6(a) shows the SDR change with respect to λ. Fig. 3.6(b) compares the SDR when λ is fixed to 0.01 or when λ is optimally chosen for each noisy condition. (M, β) is fixed to $(480, 0.01)$.

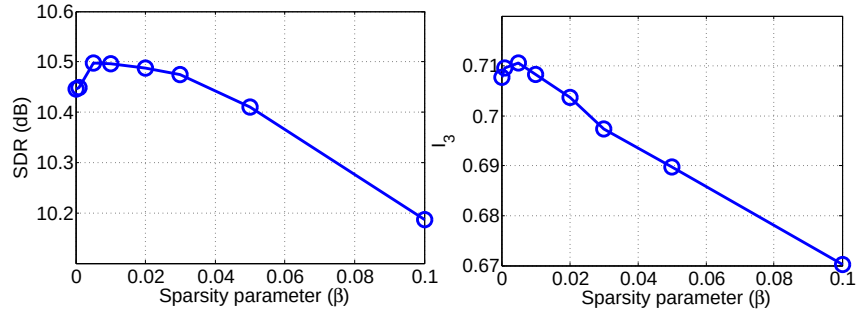
In Fig. 3.7, we observe that the performance of non-causal WNMF improves when the dictionary size M increases. This is expected since larger dictionary can theoretically capture richer speech structures. However, large dictionary size also leads to increased computational complexity.

In Fig. 3.8, we present the performance when varying the sparsity parameter β in the dictionary training stage. Experimentally, a small β, even as small as 0, results in good performance. This is because the non-negativity of D and G enforces only linear *constructive* combination, which is another way to promote sparsity.



(a) (b)

Figure 3.7: Performance of non-causal WNMF under 0 dB AWGN with different dictionary size M. Both β and λ are fixed to 0.01



(a) (b)

Figure 3.8: Performance of non-causal WNMF under 0 dB AWGN with different sparsity parameter β in the dictionary training stage. M is fixed to 480, and λ is fixed to 0.01.

Chapter 4

Sparse Gaussian Mixture Model

4.1 Sparse Gaussian Mixture Model (SGMM)

In this section, we present the SGMM for speech enhancement. We will first describe the system model, and then give a short review of the equivalence between the complex Gaussian model and the Itakura-Satio (IS) NMF. Afterwards, we will present the proposed SGMM and establish an efficient training algorithm. Finally, we will present a semi-supervised enhancement algorithm that apply the SGMM.

4.1.1 System Model

Let $y(t)$, $s(t)$, and $v(t)$ denote, respectively, the noisy observation, the clean speech, and the additive noise at time index t. Assuming $y(t) = s(t) + v(t)$, we can rewrite the equivalent spectrogram domain model using the Short-Time-Fourier-Transform (STFT):

$$\mathbf{Y} = \mathbf{S} + \mathbf{V} \in \mathcal{C}^{K \times N}$$

where K denotes the number of frequency bins and N denotes the number of time frames. Let $\hat{\mathbf{S}}$ be the estimated speech spectrogram, the time-domain enhanced speech $\hat{s}(t)$ is then obtained by an inverse Fourier transform and the overlap-add approach. Traditionally, speech spectrum $\mathbf{s}_n \in \mathcal{C}^{N \times 1}$ and noise spectrum $\mathbf{v}_n \in \mathcal{C}^{N \times 1}$ at time frame

n are assumed to obey the complex Gaussian distribution:

$$p(\mathbf{s}_n) = \mathcal{N}_c\left(\mathbf{s}_n \mid, 0, \sigma_n^s\right) \tag{4.1}$$

$$p(\mathbf{v}_n) = \mathcal{N}_c\left(\mathbf{v}_n \mid, 0, \sigma_n^v\right) \tag{4.2}$$

where $\sigma_n^s = [\sigma_{1,n}^s, \cdots, \sigma_{K,n}^s]$ denotes the speech variances at time frame n. Same definition holds for σ_n^v.

4.1.2 Equivalence: IS-NMF and complex Gaussian Model

In [15], the speech variance σ_n^s is further assumed to be a non-negative combination of a dictionary $W \in \mathcal{R}^{K \times M^s}$:

$$\sigma_n^s = W h_n, \quad h_n \geq 0 \tag{4.3}$$

where $h_n \in \mathcal{R}^{M^s \times 1}$ is the *gain* coefficient; M^s denotes the size of the dictionary.

In practice, W and $H = [h_1, \cdots, h_N]$ are unknown and need to be estimated. The most well-known statistical estimation scheme is arguably the Maximum Likelihood (ML) estimation. Interestingly, references [15, 53] have shown that performing ML estimation of W and H is equivalent to performing IS-NMF; see the result below.

Theorem 2. *[15, Theorem 1] Let* $\mathbf{S} = [\mathbf{s}_1, \cdots, \mathbf{s}_N]$*, and that* $\mathbf{s}_n$ *satisfies the complex Gaussian model* (4.1)–(4.3)*. Define* S *as the matrix with entries* $s_{k,n} = \|\mathbf{s}_{k,n}\|^2$*. Then, the ML estimation of* W *and* H *is equivalent to performing IS-NMF on* V*. That is, the ML estimates* W^* *and* H^* *are obtained by*

$$[W^*, H^*] = \arg \min_{W \geq 0, H \geq 0} d_{\mathrm{IS}}\left(S \mid WH\right).$$

where $d_{\mathrm{IS}}(X \mid Y) = \sum_{k,n} \left\{ \frac{x_{k,n}}{y_{k,n}} - \log \frac{x_{k,n}}{y_{k,n}} - 1 \right\}$

In other words, the IS-NMF, an objective for matrix factorization, actually has a very good statistical interpretation. It motivates us to find the matrix factorization counterpart of the more advanced statistical model, e.g., Gaussian mixture model.

4.1.3 Sparse Gaussian Mixture Model

One issue with the Gaussian model (4.1) – (4.3) is the necessity of trading-off the representation power with the discrimination power. That is, dictionaries with large

M^s are capable of representing more speech signals, but at the same time they may represent the noise signals equally well, resulting less discriminative power.

To improve this trade-off, we propose to model the distribution $p(\mathbf{s}_n)$ as a Gaussian mixture with certain *sparse* structure (hence the name SGMM), expressed below:

$$p(\mathbf{s}_n) = \sum_{i=1}^{I} \phi_n^i \mathcal{N}_c\left(\mathbf{s}_n \mid 0, W^i h_n^i\right) \tag{4.4}$$

$$1 \geq \phi_n^i \geq \epsilon, \quad \|\phi_n\|_1 = 1, \quad \|\phi_n - \epsilon e\|_0 \leq a. \tag{4.5}$$

where $\phi_n = [\phi_n^1, \cdots, \phi_n^I] \in \mathcal{R}^{I\times 1}$; $\|\cdot\|_1$ and $\|\cdot\|_0$ represent the l_1 norm and the l_0 norm, respectively; ϵ is a small positive constant, and e is a all one vector with dimension I.

In (4.4), we model $\mathbf{s}_n$ as a mixture of I different Gaussian models. ϕ_n^i denotes the probability that $\mathbf{s}_n$ is drawn from model i and $\{W^i, h_n^i\}$ denotes the corresponding dictionary and the gain coefficient. We want W^i to capture the spectrum structure of certain *fundamental components* of speech such as a phoneme. In each time frame n, there may be only one dominating fundamental component, say W^{j_n}. However, due to the continuity of speech signal, $\mathbf{s}_n$ is not only represented by W^{j_n}, but also affected by the nearby fundamental components $\{W^{j_{n-m}}, \cdots, W^{j_{n+m}}\}$, where m is a small positive number that represents the continuity across time. To model this phenomenon, we use the Gaussian mixture model (4.4) with ϕ_n^i denoting the *weight* of each fundamental components. Since the complex Gaussian model (4.1)– (4.3) is a special case of our model (by setting $I = 1$), it is expected that the mixture model (when $I > 1$) can better capture the complex structure of speech signal.

To maintain the discriminative power of the model, we adopt the sparse coding approach by imposing the restriction that only a few of the weights $\{\phi_n^i\}_{i=1}^I$ are non-zero; cf. (4.5). Indeed, such requirement is met by setting $\epsilon = 0$, and $a << I$, which makes sure that only a sparse combination of the I models can be used at a given time frame n. Note that the sets of active models used in different time frames can be different. In our simulation, we set ϵ to be a small positive constant because the proposed algorithm described in Sec. 4.1.4 requires $\epsilon > 0$.

4.1.4 Training SGMM for speech

The proposed training algorithm, summarized in Alg. 6, takes a Majorization Minimization (MM) approach by iteratively minimizing certain upper bound functions of the objective (4.4). Due to our special construction, we are able to minimize the upper bound function even with the nonconvex sparsity constraint (4.5). This is a very special case since most of the optimizations are NP-hard the l_0 norm is present. As a result, the proposed training algorithm is guaranteed to produce a sequence of non-increasing and convergent objectives [40].

Algorithm 6 Algorithm for training SGMM (4.6)

1: **for** iteration r **do**
2: Construct an upper bound L^r by (4.7)
3: Update ϕ_n^i by (4.9) and (4.10)
4: Update h_n^i by (4.12)
5: Update W^i by (4.13)
6: **end for**

Let $\{\mathbf{s}_n\}_n^N$ be the complex spectrums of training utterances. The SGMM is learned by the ML paradigm, where ϵ and a are the given parameters of the model.

$$\begin{aligned} \arg \min_{\phi_n^i, W^i \geq 0, h_n^i \geq 0} & \sum_{n=1}^{N} \left\{ -\log \left[\sum_{i=1}^{I} \phi_n^i \mathcal{N}_c \left(\mathbf{s}_n \mid 0, W^i h_n^i \right) \right] \right\} \\ \text{s.t.} \qquad & \phi_n^i \geq \epsilon, \quad \|\phi_n\|_1 = 1, \quad \|\phi_n - \epsilon\|_0 \leq a \end{aligned} \tag{4.6}$$

where

$$\mathcal{N}_c \left(\mathbf{s}_n \mid 0, \sigma_n^i \right) = \sum_{k=1}^{K} \frac{1}{\sqrt{2\pi\sigma_{k,n}^i}} \exp \left[-\frac{|\mathbf{s}_{k,n}|^2}{2\sigma_{k,n}} \right]$$

$$\sigma_n^i = W^i h_n \in \mathcal{R}_+^{K \times 1}$$

Define $\{\theta_n = \{\phi_n^i, W^i, h_n^i\}_{i=1}^I\}_{n=1}^N$ to be the variables of SGMM. By using the convexity of $-\log(\cdot)$, it is not difficult to show that $L^r(\{\theta_n\})$ defined below is an upper bound of

the objective function [54]:

$$\begin{aligned} L^r(\{\theta_n\}) &= \sum_{n,i} \left\{ -\lambda_n^i \log \phi_n^i - \lambda_n^i \log \left[\mathcal{N}_n \left(\mathbf{s}_n \mid 0, W^i h_n^i \right) \right] \right\} \\ &\stackrel{\circ}{=} \sum_{n,i} \left\{ -\lambda_n^i \log \phi_n^i + \lambda_n^i d_{\mathrm{IS}} \left(s_n \mid W^i h_n^i \right) \right\}. \end{aligned} \tag{4.7}$$

where r denotes the iteration number, $s_n = [s_{1,n}, \cdots, s_{K,n}]$, $s_{k,n} = \|\mathbf{s}_{k,n}\|^2$, and the notation $\stackrel{\circ}{=}$ denotes an equality up to an additive constant. In the above expression, λ_n^i is a constant defined by the variables in the *previous* iteration:

$$\lambda_n^i = \frac{\tilde{\phi}_n^i \mathcal{N}_n \left(\mathbf{s}_n \mid 0, \tilde{W}^i \tilde{h}_n^i \right)}{\sum_{j=1}^I \tilde{\phi}_n^j \mathcal{N}_n \left(\mathbf{s}_n \mid 0, \tilde{W}^j \tilde{h}_n^j \right)}$$

where the superscript $\sim$ signifies that the variable is obtained in the previous iteration. In the following, we will present the algorithmic steps for optimizing L^r with respect to different variables, i.e., $\{\phi_n^i, h_n^i, W^i\}$ subject to constraints (4.5).

Fixing $\{h_n^i, W^i\}$, L^r is separable across variables $\{\phi_n^i\}$. By a change of variable, $q_n^i = p_n^i - \epsilon$, the n-th subproblem becomes

$$\begin{aligned} P_0^n : \arg \min_{\{q_n^i\}_{\forall i}} \quad & -\sum_i \lambda_n^i \log \left(q_n^i + \epsilon \right) \\ \text{s.t. } & q_n^i \geq 0, \quad \|q_n\|_1 = 1 - \epsilon I, \quad \|q_n\|_0 \leq a. \end{aligned} \tag{4.8}$$

It is now clear why the proposed algorithm requires $\epsilon > 0$. If $\epsilon = 0$, (4.8) becomes ill-posed because for all feasible $\{q_n^i\}$, the objective value equals to ∞. To solve (4.8), we first observe that without the cardinality constraint, P_0^n can be solved by the well-known water-filling algorithm; see [55, Sec.10]. Somewhat surprisingly, the following result says that adding the cardinality constraint $\|q_n\|_0 \leq a$ even simplifies the solution.

Theorem 3. *Without loss of generality, we assume $\{\lambda_n^i\}$ in problem P_0^n satisfies $\lambda_n^1 \geq \lambda_n^2 \geq \cdots \lambda_n^I \geq 0$. Then $\{\hat{q}_n^i\}_{i=1}^I$ defined below is the global optimal solution of problem*

P_0^n.

$$[\hat{q}_n^1, \cdots, \hat{q}_n^a] = \arg \min_{q_n^1, \cdots, q_n^a} -\sum_{i=1}^{a} \lambda_n^i \log\left(q_n^i + \epsilon\right) \tag{4.9}$$

$$s.t.\ q_n^i \geq 0, \quad \sum_{i=1}^{a} q_n^i = 1 - \epsilon I, \quad \forall i = 1, \cdots, a$$

$$\hat{q}_n^j = 0, \quad \forall j > a. \tag{4.10}$$

Moreover, (4.9) *can be solved by the water-filling algorithm:*

$$\hat{q}_n^i = \begin{cases} \lambda_n^i \nu - \epsilon, & \lambda_n^i \nu \geq \epsilon \\ 0, & \text{otherwise} \end{cases}$$

where ν *is the value that ensures* $\sum_{i=1}^{a} \max\{\lambda_n^i \nu - \epsilon, 0\} = 1 - \epsilon I$.

Proof. Let $\{q_n^{i,*}\}$ be the optimal solution of P_0^n. Using the monotonicity of λ_n^i, we can show that $\{q_n^{i,*}\}$ must satisfy the following rule:

$$\text{if } q_n^{i,*} > 0, \text{ then } q_n^{j,*} > 0, \ \forall j < i.$$

It means only $\{q_n^{1,*}, \cdots, q_n^{a,*}\}$ can be non-zero, and the rest $\{q_n^{i,*}\}_{i=a+1}^{I}$ have to be zero. By substituting this fact into P_0^n, we get (4.9) and (4.10). □

Next, let us look at the subproblem for $\{h_n^i\}$. Clearly L^r is separable with respect to each h_n^i, so this subproblem decomposes into $N \times I$ independent IS-NMF:

$$\arg \min_{h_n^i \geq 0} d_{IS}\left(s_n \mid \tilde{W}^i h_n^i\right), \quad \forall n, i. \tag{4.11}$$

The multiplicative update (4.12) proposed in [56] can be used to solve (4.11)

$$h_n^i = \tilde{h}_n^i \odot \left[\frac{(\tilde{W}^i)^T \left[\left(\tilde{W}^i \tilde{h}_n^i \right)^{\cdot -2} \odot s_n \right]}{(\tilde{W}^i)^T \left(\tilde{W}^i \tilde{h}_n^i \right)^{\cdot -1}} \right]^{\cdot 1/2}, \forall n, i \tag{4.12}$$

Optimizing L^r with respect to W^i is separable across i but not across n. For each i, we apply the MM approach again and obtain the multiplicative update shown in (4.13).

$$W^i = \tilde{W}^i \odot \left[\frac{\left[(\tilde{W}^i \tilde{H}^i)^{\cdot -2} \odot S \right] \times \text{diag}(\{\lambda_n^i\}_{n=1}^N) \times (\tilde{H}^i)^T}{(\tilde{W}^i \tilde{H}^i)^{\cdot -1} \times \text{diag}(\{\lambda_n^i\}_{n=1}^N) \times (\tilde{H}^i)^T} \right]^{\cdot 1/2} \tag{4.13}$$

where $\text{diag}(v)$ denotes a diagonal matrix with vector v being the diagonal entries. $X^{\cdot a}$ denotes raising each entry of the matrix X to the power a.

Algorithm 7 Speech variance estimation (4.18)

1: **for** iteration r **do**
2: Construct an upper bound L^r:

$$L^r = \sum_i -\lambda_n^i \log \phi_n^i + \lambda_n^i d_{\text{IS}}\left(y_n \mid W^i h_n^i + \hat{\sigma}_n^v\right)$$

where $y_n := [y_{1,n}, \cdots, y_{K,n}]$ and $y_{k,n} = \|\mathbf{y}_{k,n}\|^2$, and

$$\lambda_n^i = \frac{\tilde{\phi}_n^i \mathcal{N}_n\left(\mathbf{y}_n \mid 0, W^i \tilde{h}_n^i + \hat{\sigma}_n^v\right)}{\sum_{j=1}^I \tilde{\phi}_n^j \mathcal{N}_n\left(\mathbf{y}_n \mid 0, W^j \tilde{h}_n^j + \hat{\sigma}_n^v\right)}$$

3: Update ϕ_n^i by (4.9) and (4.10)
4: Update h_n^i:

$$h_n^i = \tilde{h}_n^i \odot \left[\frac{(W^i)^T \left[\left(W^i \tilde{h}_n^i + \hat{\sigma}_n^v\right)^{.-2} \odot y_n\right]}{(W^i)^T \left(W^i \tilde{h}_n^i + \hat{\sigma}_n^v\right)^{.-1}}\right]^{.1/2}$$

5: **end for**
6: Estimate speech variance $\hat{\sigma}_{k,n}^s$ by (4.15)

4.1.5 Applying SGMM for monaural speech enhancement

In this section, we present the proposed enhancement algorithm that combines the SGMM for speech modeling and the noise tracking module for noise modeling.

Let us assume that the speech spectrum $\mathbf{s}_n$ obeys the SGMM (4.4)–(4.5), and the noise spectrum $\mathbf{v}_{k,n}$ obeys a complex Gaussian distribution. Then we can write the probability density function of the noisy complex spectrum $\mathbf{y}_n = \mathbf{s}_n + \mathbf{v}_n$:

$$p(\mathbf{y}_n) = \sum_{i=1}^I \phi_n^i \mathcal{N}_c\left(\mathbf{y}_n \mid 0, W^i h_n^i + \hat{\sigma}_n^v\right) \tag{4.14}$$

where W^i is the dictionary learned in the SGMM training stage, and $\hat{\sigma}_n^v \in \mathcal{R}^{K\times 1}$ is the noise variance estimated by the MMSE-based noise tracking algorithm [4]. Suppose the distribution $p(\mathbf{y}_n)$ in (4.14) is fully specified, we can estimate the speech variance:

$$\hat{\sigma}_{k,n}^s = \sum_{i=1}^I \phi_n^i \left[W^i h_n^i\right]_k \tag{4.15}$$

Then, Wiener filtering (4.17) together with the DD approach (4.16) can be used to produced the enhanced spectrum.

$$\xi_{k,n} = \alpha \frac{|\hat{\mathbf{x}}_{k,n-1}|^2}{\hat{\sigma}^v_{k,n}} + (1-\alpha)\frac{\hat{\sigma}^s_{k,n}}{\hat{\sigma}^v_{k,n}} \tag{4.16}$$

$$\hat{\mathbf{x}}_{k,n} = \frac{\xi_{k,n}}{1+\xi_{k,n}}\mathbf{y}_{k,n} \tag{4.17}$$

where $0 \leq \alpha \leq 1$ denotes the temporal smoothing constant.

Since $\{\phi^i_n, h^i_n\}$ are unknown, ML estimation (4.18) is used:

$$\begin{aligned} \arg \min_{\phi^i_n, h^i_n \geq 0} \; & -\log\left[\sum_{i=1}^{I} \phi^i_n \mathcal{N}_c\left(\mathbf{y}_n \mid 0, W^i h^i_n + \hat{\sigma}^v_n\right)\right] \\ & \text{s.t. } \phi^i_n \geq \epsilon, \quad \|\phi_n\|_1 = 1, \quad \|\phi_n - \epsilon\|_0 \leq a \end{aligned} \tag{4.18}$$

Due to the similarity with (4.6), we can use the same technique to derive update rules for ϕ^i_n and h^i_n; see Alg. 7.

There are two reasons for using the noise tracking algorithm instead of the conventional ML approach [18,20,24,57]. First, we observe that the noise tracking algorithm [4] can give a reasonablely good estimate. Second, using it makes the proposed algorithm causal, while most semi-supervised algorithms are non-causal due to the ML estimation of noise dictionary.

4.2 Simulation for SGMM

4.2.1 Experiment Setup

The TIMIT data set [50] is used as the speech corpus, and five types of noises from NOISEX-92 [51], namely, babble, factory 1, factory 2, street and tank, are taken as the noise sources. All audio files are sampled at 16 kHz, and segmented into 30-ms duration using a Hamming window with 50% overlap. A 512 point FFT/IFFT was used for the TF analysis and synthesis operations (thus, $K = 257$). Noisy mixtures are obtained by mixing a clean sentence with one type of noises at three different SNRs (-5, 0, 5 dB).

We use the composite measures, $\{C_{sig}, C_{bak}, C_{ovl}\}$, proposed in [31] to measure the quality of the enhanced speech due to its high correlation with human subjective test. C_{sig} measures the degree of speech signal distortion, C_{bak} measures the background

noise intrusiveness, and C_{ovl} measures the overall quality. All measures range from 1 to 5, while 1 being the worse and 5 being the best.

4.2.2 Implementation

Here we present the implementation of various algorithms. The parameters for different algorithms are summarized in Table 4.1.

The first algorithm is the well-known Wiener filter (WF). The WF is fully unsupervised because the speech variance is estimated using the DD approach (4.19), and the noise variance $\hat{\sigma}^v_{k,v}$ is estimated from the noisy mixture using the MMSE-based approach [4]:

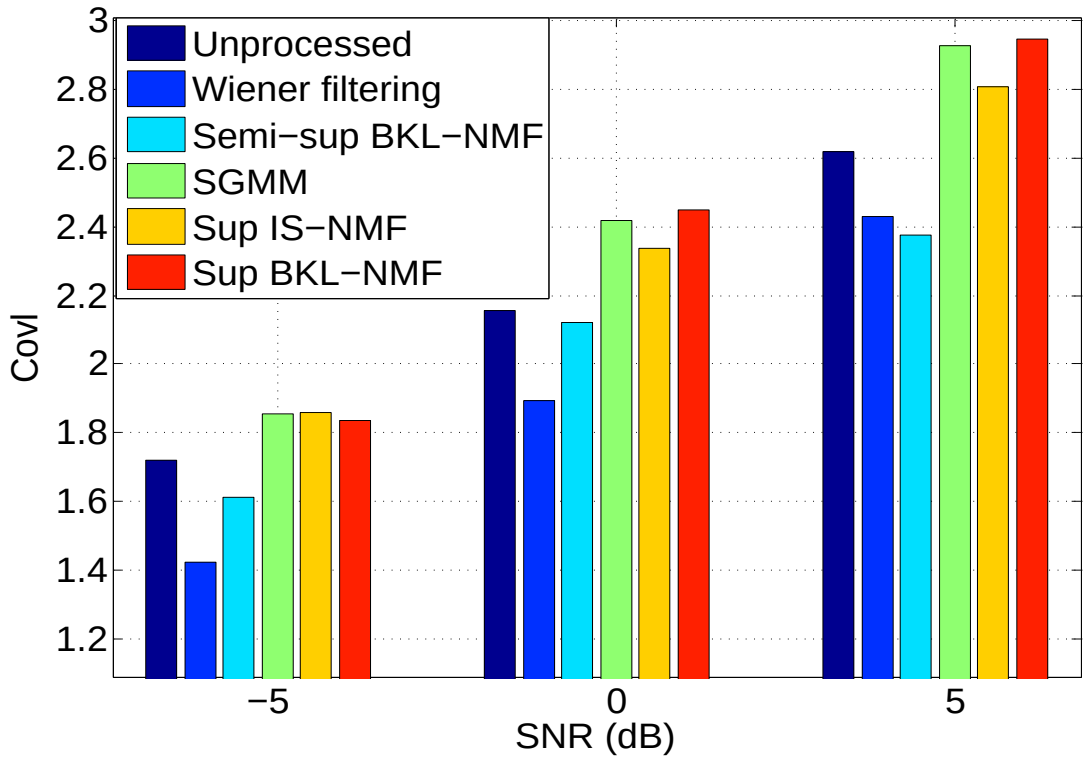
$$\hat{\sigma}^s_{k,n} = \alpha|\hat{\mathbf{x}}_{k,n-1}|^2 + (1-\alpha)\max\left\{|\mathbf{y}_{k,n}|^2 - \hat{\sigma}^v_{k,n}\right\}. \tag{4.19}$$

Two semi-supervised algorithms are considered, the proposed SGMM algorithm and the semi-supervised version of the BKL-NMF algorithm [57]. In SGMM, a is set to 3 in order to model the effect from the frame before and the frame after. The other parameters are experimentally found to yield a good performance. In semi-sup BKL-NMF, a total of I speech dictionaries each of size $K \times M^s$ are learned offline. The noise dictionary $W^v \in \mathcal{R}^{K \times M^v}$, however, is learned online from the noisy mixture using ML estimation. In the enhancement stage, semi-sup BKL-NMF first separates the noisy mixture into a speech component and a noise component. These components are then used to construct the Wiener gain (4.17) for producing the enhanced speech.

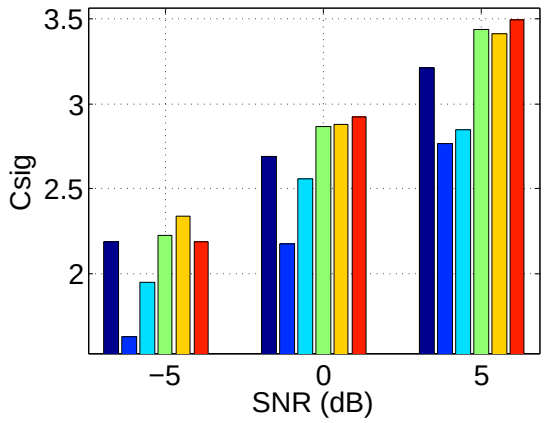
The two supervised algorithms considered are the IS-NMF [56] and the supervised version of BKL-NMF [57]. In sup IS-NMF, a speech dictionary with size $K \times M^s$ and a noise dictionary with size $K \times M^v$ are constructed in the training stage. It means that the algorithm must know the testing noise type in the training stage, and therefore is called "supervised" algorithm. Similarly, the noise dictionary in sup BKL-NMF is also trained a priori.

4.2.3 Enhancement Result

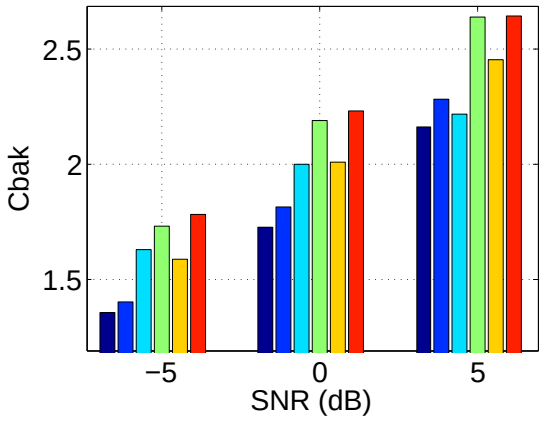
Fig. 4.1 presents the enhancement result at three different input SNRs. Audio samples are available at [58]. First, we observe that the proposed SGMM algorithm achieves a comparable performance with the two supervised algorithms, Sup IS-NMF and Sup



(a) C_{ovl}



(b) C_{sig}



(c) C_{bak}

Figure 4.1: Enhanced speech quality comparison between algorithms using three objective metrics. All values are averaged across 192 randomly selected sentences and across 5 different noise types, namely, babble, factory 1, factory 2, street, and tank.

Table 4.1: Parameters used in different algorithms

WF	SGMM				Sup IS-NMF [56]		BKL-NMF [57] (Sup & Semi-sup)		
α	I	M^s	a	ϵ	M^s	M^v	I	M^s	M^v
0.98	20	30	3	10^{-5}	30	10	20	10	10

BKL-NMF, even though SGMM is only a semi-supervised algorithm which does not require the knowledge of noise type. Note that after the speech and noise variances are estimated, all three algorithms use *the same* Wiener gain (4.17) to perform enhancement. Therefore, we can conclude that the variances estimated by our algorithm are as good as those estimated by the other two supervised algorithms. This is due to the combination of the proposed SGMM and the noise variance estimation module [4]. Moreover, our algorithm is more practical and less computational intensive, since no pre-trained noise dictionaries are needed.

Second, the SGMM clearly outperforms the semi-sup BKL-NMF across all SNRs. We attribute this to the way how sparse regularization is enforced. While semi-sup BKL-NMF uses a non-convex regularization in the objective to promote sparsity, we uses an l_0 norm constraint to explicitly enforce sparsity. Even though both approaches are non-convex, our approach has the advantage that it is parameter-free: there is no sparsity related regularization coefficient to be tuned. In our experience, we found that the performance of semi-sup BKL-NMF is sensitive to the regularization coefficient, which is difficult to select optimally. However, the proposed SGMM algorithm is found to be robust to the l_0 norm bound a. For example, setting $a = 3$ yields a consistent high-quality enhancement in all scenarios.

Third, the SGMM approach yields a better performance than the unsupervised Wiener filtering, as expected. Note that both algorithms use *i).* the same algorithm [4] for tracking the noise variances, *ii).* the same Wiener gain (4.17) for producing the enhanced spectrum. Therefore the observed performance gap is a result of different ways that the speech variances are estimated. While the baseline Wiener filtering uses (4.19) without any speech modeling, SGMM leverages the pre-trained speech model to estimate the speech variance.

Chapter 5

Sparse Non-negative Matrix Factorization + Deep Neural Network

In this chapter, we present the proposed SNMF-DNN algorithm which includes two versions. The first version presented in Sec. 5.1 is called the SNMF-DNN-Binary. It estimates the ideal binary mask by using sparse NMF for feature extraction and a DNN for classification. Simulation result for SNMF-DNN-Binary is presented in Sec. 5.2. The second version presented in Sec. 5.3 is called the SNMF-DNN-Ratio. It is an improved version of SNMF-DNN-Binary because of the following reasons.

1. SNMF-DNN-Ratio does not require noise training data, while SNMF-DNN-Binary requires.

2. SNMF-DNN-Ratio uses a DNN to estimate the Ideal Ratio Mask (IRM), while SNMF-DNN-Binary estimates the Ideal Binary Mask (IBM). IBM is a binary version of IRM, and thus is easier to estimate.

3. In our simulation, SNMF-DNN-Ratio results in a significantly better enhanced quality while at the same time maintains a comparable intelligibility as SNMF-DNN-Binary.

Sec. 5.4 shows the simulation of SNMF-DNN-Ratio.

5.1 SNMF-DNN-Binary

In this section, we present the SNMF-DNN-Binary algorithm, a new IBM-based speech enhancement algorithm. We demonstrate that, without requiring much speech-specific domain knowledge during the entire process, a superior enhancement performance can be achieved by using a simple feature extraction step followed by a DNN classifier. In particular, we use the standard Sparse NMF (SNMF) to extract features from the noisy mixture. This step is simple and easy to implement. Though SNMF has been applied to soft gain calculation before [18, 24, 59, 60], this is the first time it is used for IBM estimation. The SNMF step is followed by a DNN classifier, which is a neural network with more than one hidden layers. Here we use a DNN rather than a SVM for classification because the former achieves a better performance in our experiment, which is in line with other reports [61, 62].

5.1.1 System Description

We first provide an overview of the proposed system shown in Fig. 5.1. The details will be described latter. As a proof of concept, we consider a matched-noise condition, i.e., the noise types and the input SNRs in both the training and testing stages are the same. For each type of noise, a system like Fig. 5.1 is trained independently. The proposed system differs from [26, 28, 63] in how the feature extraction and the classification are performed.

In the training stage, time-domain signals are transformed into the T-F representation by passing through an *analysis front-end* and a *cochleagram* computation [6]. Then, we apply SNMF, a sparse variant of NMF originally designed for object recognition [64], to extract features. SNMF is divided into a *dictionary training* step and a *sparse coding* step. In the dictionary training step, a speech dictionary D^s and a noise dictionary D^v are learned from the speech cochleagram S and the noise cochleagram V, respectively. These dictionaries are used to extract features from the noisy cochleagram Y in the sparse coding step. The extracted features together with the true IBM are used to train the *DNN classifier*.

In the testing stage, noisy speech is first transformed into the cochleagram representation, and is then passed through the sparse coding block for feature extraction.

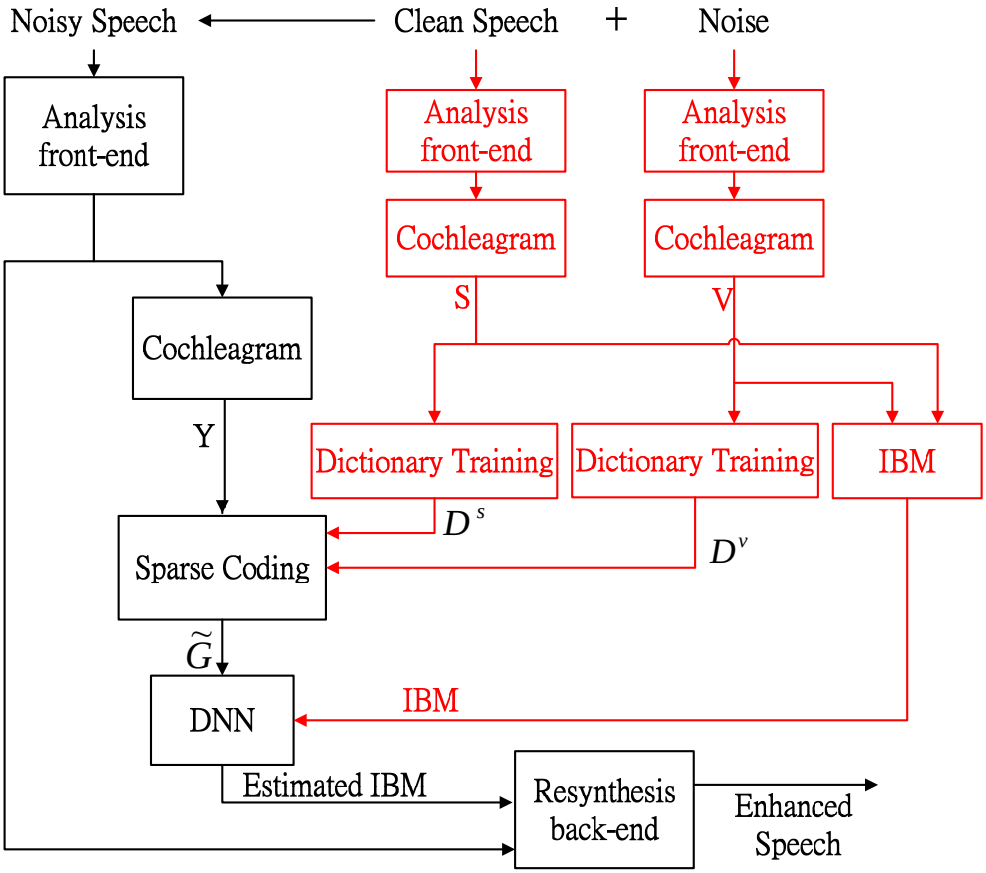


Figure 5.1: Block diagram of the SNMF-DNN-Binary algorithm. Modules highlighted in red are those modules used only in the training stage; modules in black are those used in both training and testing stage.

Different from the training stage, the sparse coding here uses the dictionaries already trained in the training stage, and therefore does not require access to the actual speech and noise. Taking the extracted features as input, the trained DNN generates the estimated IBM. The *resynthesis back-end* takes the estimated IBM and produces the enhanced speech.

Analysis front-end, Cochleagram, IBM, and Resynthesis back-end

In the analysis front-end, audio files sampled at 16 kHz are passed through a 64-channel Gammatone filterbank with center frequencies spanning from 50 Hz to 8 kHz on the

equivalent rectangular bandwidth rate scale. The output of each filter is divided into 20-ms segments with 10-ms overlap. Energy in each segment is calculated and then forms a T-F matrix called cochleagram. We use $Y \in \mathcal{R}_+^{64\times N}$ to denote the cochleaagram of the noisy speech, where N represents the total number of time frames. Let S and V denote the cochleagram of clean speech and noise, respectively.

The ideal binary mask matrix $B \in \mathcal{R}_+^{64\times N}$ is defined based on whether a T-F unit is either speech-dominant or noise-dominant:

$$B_{k,n} = \begin{cases} 1, & \text{if SNR}_{k,n} \geq LC \\ 0, & \text{otherwise} \end{cases}$$

where $\text{SNR}_{k,n} = \frac{S_{k,n}}{V_{k,n}}$ denotes the local SNR, and LC denotes the local SNR criterion which is set to -5 dB. Clearly, calculating IBM requires access to S and V, which can only be done in the training stage but not the testing stage.

In the resynthesis step, the estimated IBM (which will be explained later) is used to binarily weight the Gammatone filterbank output of each channel. The speech-dominant regions are kept intact while the noise-dominant regions are discarded. All the 64 weighted streams are then summed to produce the final enhanced speech. We use the implementation from Wang's group[1] for the analysis front-end, cochleagram calculation, and the resynthesis back-end.

Learning Speech and Noise Dictionary

We first present the idea and the computational algorithm for the dictionary learning step in SNMF, and then show how to specialize it to speech and noise. Dictionary learning factorizes a non-negative matrix $X \in \mathcal{R}_+^{K\times N}$ into the product of a dictionary $D \in \mathcal{R}_+^{K\times M}$ and a sparse code $G \in \mathcal{R}_+^{M\times N}$:

$$X \approx D \times G \tag{5.1}$$

where M denotes the size of the dictionary. Columns of D are called *atoms*. Since G is sparse, only a *few* atoms suffice to represent X. In other words, the sparse code G necessarily contains important information about X. Computationally, factorization

[1] See the code available at `http://www.cse.ohio-state.edu/pnl/`.

(5.1) is achieved by solving (5.2):

$$\min_{D\geq 0, G\geq 0} d_{\mathrm{IS}}(X \mid DG) + \lambda \sum_{m,n} \log(\epsilon + G_{m,n}) \tag{5.2}$$

where $d_{\mathrm{IS}}(\cdot \mid \cdot)$ denotes the Itakura-Satio (IS) divergence [54] between matrices A and B:

$$d_{\mathrm{IS}}(A \mid B) = \sum_{k,n} \left\{ \frac{A_{k,n}}{B_{k,n}} - \log \frac{A_{k,n}}{B_{k,n}} - 1 \right\}.$$

The second term in (5.2) is a sparsity-promoting regularization [65]. The larger the λ, the sparser the G. Moreover, ϵ is a small positive constant to make the term inside logarithm strictly positive. Similar to [54], (5.2) can be solved by:

$$D \leftarrow D \odot \left\{ \frac{\left[(DG)^{\cdot -2} \odot X\right] G^T}{(DG)^{\cdot -1} G^T} \right\}^{\cdot \frac{1}{2}} \tag{5.3}$$

$$G \leftarrow G \odot \left\{ \frac{D^T \left[(DG)^{\cdot -2} \odot X\right]}{D^T (DG)^{\cdot -1} + \frac{\lambda}{\epsilon + G}} \right\}^{\cdot \frac{1}{2}} \tag{5.4}$$

Besides being easy to implement, these update rules generate non-increasing objective values and converge theoretically [40].

A speech dictionary $D^s \in \mathcal{R}_+^{64\times M^s}$ is trained by substituting S for X in (5.2), while a noise dictionary $D^v \in \mathcal{R}_+^{64\times M^v}$ is learned by replacing X with V. Only one *universal* speech dictionary is trained, while one noise dictionary is trained for *each* noise type. The speech dictionary is expected to work well for *all* utterances regardless of the gender, dialects, and content. Therefore, we set $M^s > 64$ with $\lambda > 0$ in order to ensure the speech dictionary is comprehensive enough. As for noise dictionary, we use $M^v < 64$ with $\lambda = 0$ because D^v is responsible for characterizing only *one* noise type.

Sparse Coding

By assuming a linear model, i.e., $Y \approx S + V$, S and V can be estimated from Y via sparse coding:

$$\min_{G^s\geq 0, G^v\geq 0} d_{\mathrm{IS}}\left(Y \mid D^s G^s + D^v G^v\right) + \beta \sum_{m,n} \log\left(\epsilon + G^s_{m,n}\right) \tag{5.5}$$

where D^s and D^v are the previously learned dictionaries and ϵ is a small positive constant. Problem (5.5) can be efficiently solved by a similar update as in (5.4). By the ways that D^s and D^v are trained, we have $D^s G^s \approx S$ and $D^v G^v \approx V$. If both approximations are accurate, then the IBM can be reliably estimated by a simple thresholding:

$$\hat{B}_{k,n} = \begin{cases} 1, & \text{if } \frac{[D^s G^s_n]_k}{[D^v G^v_n]_n} \geq LC \\ 0, & \text{otherwise} \end{cases}, \tag{5.6}$$

which can be viewed as applying a simple linear classifier on the sparse code $G_n := [G^s_n; G^v_n] \in \mathcal{R}_+^{(M^s+M^v)\times 1}$ for each channel k. However, if the approximation is inaccurate, then more advanced classifiers such as SVM or DNN can improve the performance. To include more *temporal information*, we concatenate two adjacent frames to construct the final feature:

$$\tilde{G} := \left\{ \tilde{G}_n \in \mathcal{R}_+^{3(M^s+M^v)\times 1} \mid [G_{n-1}; G_n; G_{n+1}], \forall\ n \right\} \tag{5.7}$$

Deep Neural Network

We train a DNN for classification. For time frame n, it takes $\tilde{G}_n$ as the input and takes $[B_{1,n}; \cdots; B_{64,n}] \in \mathcal{R}_+^{64\times 1}$ as the label. The output of the network, the estimated IBM $[\hat{B}_{1,n}; \cdots; \hat{B}_{64,n}]$, is used to resynthesize the enhanced speech. In particular, we use a 5-layer Rectified Linear Unit (ReLU) [66] network, which uses a rectified activation function, $\max(0, z)$, as the non-linear layer. Compared with traditional sigmodial or hyperbolic tangent non-linearity, ReLU achieves a better performance without any unsupervised pre-training [67]. Training of the network is performed by minimizing the cross entropy loss over the training set with pure back propagation algorithm.

Rationale of the System Design

In our design, we perform IBM estimation in the cochleagram domain instead of the conventional spectrogram domain. There are two main reasons behind our choice of the working domain. First, the cochleagram only requires 64 channels to provide a fine enough frequency resolution for a 16 kHz speech, while the conventional spectrogram usually requires at least 256 channels. The channel reduction comes from the non-uniform frequency spacing in the cochleagram which mimics the human perception.

From a classification point of view, estimating a mask with dimension 64 is easier and thus more desirable than a mask with dimension 256. Second, despite the smaller dimensionality, masking on the cochleagram domain still results in high quality enhanced speech [63], provided the mask is accurately estimated.

The SNMF is chosen for feature extraction, mainly due to its simplicity and effectiveness. Existing works use sophisticated features to find an accurate mask. The design and extraction of these features heavily rely on extensive domain knowledge and human fine tuning [7, 26, 28, 63]. In contrast, our SNMF based approach is conceptually much simple. However, despite the simplicity, the SNMF is capable of extracting the desired speech and noise information, even when the noise is non-stationary [18, 60]. This desirable property is the basis for achieving an accurate IBM estimation.

The DNN is used for classification, mainly due to its huge success in various complicated machine learning tasks [61, 62]. Certainly, other classifiers such as SVM can also be used, but experimentally we have found DNN to achieve the best result.

5.2 Simultion for SNMF-DNN-Binary

In this section, we present the simulation result for the SNMF-DNN-Binary algorithm, the first version of the SNMF-DNN algorithm.

5.2.1 Experiment Setup

The TIMIT data set [50] was used as the speech corpus. Three types of noises from NOISEX-92 [51], namely, street, factory, babble, are taken as noise sources. Audio files are resampled at 16 kHz. Noisy mixture is obtained by mixing a sentence with one type of noise at -5 dB. In the training stage, a universal speech dictionary is trained using 500 sentences from the "train" subset of TIMIT. For each noise type v, a separate noise dictionary D^v is trained using a randomly selected 30-second noise segment. Further, a DNN is trained using 4000 noisy sentences obtained by mixing utterances from the "train" subset and randomly selected noise segments of type v. In the testing stage, the "test" subset is combined with each of the three noise sources for performance evaluation.

To quantify the classification performance, we use the HIT-FA criterion, which has

been shown to correlate well to human speech intelligibility [7]. Here HIT represents the percentage of correctly predicted speech-dominant T-F units while FA is the percentage of wrongly predicted noise-dominant T-F units. To assess the performance of the enhanced speech, we use the Perceptual Evaluation of Speech Quality score (PESQ) [30] for evaluating the quality and the Short-Time Objective Intelligibility measure (STOI) [37] for intelligibility. PESQ ranges from 0.5 to 4.5, while STOI takes its value between -1 and 1. Both measures are known to correlate well to human perception. The higher the value, the better the performance.

To evaluate the performance of DNN in our proposed approach, we compare two alternative classifiers: the simple thresholding (5.6) and the linear SVM (LSVM), assuming that all classifiers use the same set of features extracted from SNMF. Though kernel SVM generally achieves a better performance than LSVM, the high complexity of kernel SVM makes it prohibitive in our setup, where both the feature dimensionality and the number of samples are big. In SNMF + LSVM, while $\tilde{G}_n$ (5.7) remains as the feature, we train 64 LSVMs as the classifiers for the 64 channels [68]. Moreover, two other classification-based systems proposed in Kim *et al.* [7] and Chen *et al.* [28] are also compared. For Kim's system, we use the values reported in [26], and use the results reported in [28] (Table 1 and 2) for Chen's system. Both systems are compared because they also consider a matched-noise condition and use similar front-end and back-end structures.[2] Therefore, we can focus on the influence of feature extraction and classification. Further, to compare the quality and intelligibility of the enhanced speech, we implement a commonly-used statistical-based algorithm by using the MMSE algorithm [4] for noise tracking and the Log Short-Time Spectrum Amplitude (LSTSA) estimator [3] for gain calculation. For notational convenience, this algorithm is referred to as LSTSA.

5.2.2 Parameters for Dictionary Training and Sparse Coding

The speech dictionary is trained by solving (5.2), where X is replaced by the clean speech cochleagram S. The dictionary size M^s is set to 512, λ is set to 0.01, and ϵ is set to 10^{-5}. We perform 500 iterations of the multiplicative updates (5.3) and (5.4), which

[2] Chen's system uses a 32-channel Gammatone filterbank in the analysis front-end, and sets the LC value to -10 dB. [26] modifies Kim's system to use exactly the same front-end and back-end as ours.

takes about 1 hour when using a MATLAB implementation on a cluster computer.[3]

For each noise type v, the corresponding noise dictionary D^v is trained by solving problem (5.2), where X is replaced by the noise cochleagram. For different noise dictionaries, their sizes M^v are chosen according to the "complexity" of the corresponding noise sources. For example we use the largest number of atoms for the babble noise dictionary, as it is clearly the most complex one among all three types of noises. Other parameters for the noise dictionary training as well as the sparse coding are given in Table 5.1.

In (5.5), the regularization parameter β in each noise type is determined by first solving (5.5) for a large number of potential candidates[4] of β, and then picking the one with the best separation performance (or the smallest $d_{IS}(S \mid D^s G^s) + d_{IS}(V \mid D^v G^v)$). Interestingly, the resulting β makes intuitive sense: when M^v is large, the corresponding noise dictionary D^v is more likely to represent the speech. As a result smaller β should be used so that the speech dictionary can represent the speech source as much as possible.

Table 5.1: Parameters of noise dictionaries and sparse coding stage

	Street	Factory	Babble
M^v in (5.2)	5	15	25
ϵ in (5.2)	10^{-5}	10^{-5}	10^{-5}
β in (5.5)	0.15	0.04	0.02
ϵ in (5.5)	10^{-5}	10^{-5}	10^{-5}

5.2.3 Parameters for DNN

We use a 5 layer ReLU network [66] with $3 \times M$ neurons in the input layer, 1024 neurons in the 3 hidden layers, and 64 neurons in the output layer. Here, M denotes the total number of dictionary atoms. For example, the network used in babble noise has $3 \times (512 + 25)$ neurons in the input layer. The drop out probability [66] in the input layer is set to 0.2, and it is changed to 0.5 in the hidden and output layer. To avoid form diverging, we constraint the L_∞ norm of the network weight to be less than 0.1.

[3] A Linux-based system with 8 Intel Sandy bridge E5-2670 processors (2.6 GHz) and 64 GB memory.

[4] $\beta \in [0.001, 0.01, 0.02, 0.04, 0.06, 0.08, 0.1, 0.12, 0.15, 0.2, 0.3, 0.5]$

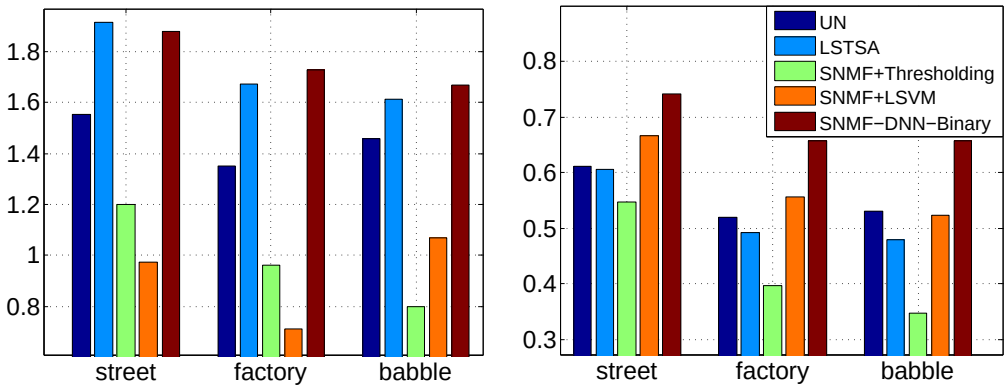
500 epoches is used for supervised training, which takes 30 hours to run on a cluster computer.[3]

5.2.4 Enhancement Results

In Table 5.2, we present the classification performance under different noise conditions at −5 dB SNR. The "x" means there is no available data. The differences between these systems are the feature extraction step and/or the classification step. The front-end and back-end are the same. The first three rows of Table 5.2 show a monotonic performance improvement when the stronger classifier is used. When the SNR is as low as −5 dB, SNMF itself is unable to produce a reliable decomposition, and therefore SNMF+Thresholding cannot deliver satisfactory result. When a classifier is used, we see a better performance by using DNN than using LSVM. Also, it is clear that the proposed system outperforms the system proposed by Chen *et al.* and Kim *et al.* Our results also suggest that when the DNN is used as the classifier, a simple SNMF based feature extraction is sufficient for classification. Note that [63] also uses DNN for classification, but only the 0 dB scenario was considered, where most normal-hearing listeners have near perfect recognition rate.

Table 5.2: Classification results for different systems at −5 dB SNR under 3 noise types. Bold faced letters denote the best result.

		Street	Factory	Babble
SNMF-DNN-Binary	HIT	**86.5**%	**78.1**%	**75.8**%
	FA	**12.9**%	**9.0**%	**13.9**%
	HIT-FA	**73.6**%	**69.1**%	**61.9**%
SNMF+LSVM	HIT	73.9%	65.7%	58.8%
	FA	**12.9**%	11.4%	18.7%
	HIT-FA	61.0%	54.3%	40.1%
SNMF+Thresholding	HIT	59.4%	35.3%	43.2%
	FA	25.0%	17.5%	31.7%
	HIT-FA	34.4%	17.8%	11.5%
Chen *et al.* [28]	HIT	x	70%	62%
	FA	x	7%	13%
	HIT-FA	x	63%	49%
Kim *et al.* [7] (from Table 1 of [26])	HIT	x	57.3%	53.8%
	FA	x	26.7%	27.1%
	HIT-FA	x	30.6%	26.6%



(a) PESQ (b) STOI

Figure 5.2: Performance comparison of the enhanced speech at −5 dB SNR under street, factory, and babble noises. "UN" denotes the unprocessed noisy mixture. "LSTSA" denotes using LSTSA [3] with MMSE noise variance tracking [4]. SNMF+Thresholding and SNMF+LSVM denotes the system that uses the same structure as the proposed system, but changes the classifiers from DNN to thresholding and LSVM, respectively. All values are averaged over 300 randomly selected sentences from the "test" data set.

Fig. 5.2 evaluates the quality and the intelligibility of the enhanced speech by different approaches. Audio samples are available at [69]. In terms of both PESQ and STOI, the proposed system outperforms the thresholding and the LSVM counterparts. This is consistent with the results in Table 5.2. Compared to LSTSA, the proposed approach achieves a comparable speech quality to LSTSA. This is very encouraging, as it shows that a classification-based system is capable of achieving high speech quality, although its primary target is to improve the speech intelligibility. Further, we observe from Fig. 5.2(a) that the advantage of the proposed approach over LSTSA is more pronounced when the noise becomes increasingly more non-stationary (from "street" to "babble"). We attribute this to the use of SNMF in the feature extraction step, as this technique is known to work well when the noise source contains distinct features, and is robust to the non-stationarity [18, 24, 59, 60]. Fig. 5.2(b) compares the speech intelligibility of the enhanced speech. As expected, LSTSA is not able to improve the speech intelligibility, while the proposed system has a relatively significant improvement. This is consistent

with what is known in the literature, that traditional enhancement methods cannot improve speech intelligibility [70], while a properly trained classification-based algorithm can [7, 26, 28, 63].

5.3 SNMF-DNN-Ratio

5.3.1 Introduction

In this section, we present the SNMF-DNN-Ratio, the second version of SNMF-DNN. Comparing these two versions, SNMF-DNN-Ratio is more practical because it does not require noise training data. In terms of the performance, SNMF-DNN-Ratio yields a better enhanced quality than SNMF-DNN-Binary, while maintains a comparable enhanced intelligibility. This will be verified in the simulation section.

Difference between SNMF-DNN-Ratio and other IM-based algorithm

The proposed SNMF-DNN-Ratio algorithm differs from the existing algorithms in several noted ways. First, the role of NMF has been changed: it is no longer directly used for separation [18, 20, 23, 24, 44, 57, 60], but is used for feature extraction. In most NMF-based algorithms, NMF separates the noisy observation into a speech component and a noise component. The speech component is directly used as the enhanced speech. This approach works well if both the speech model and the noise model are properly trained. However, when models are not ideal, e.g., when there is a mismatch between the training data and the testing data, the performance degrades significantly. To improve the robustness against model mismatch, we view both components as features and use another DNN for mask estimation. Using DNN, errors caused by model mismatch can be largely corrected, leading to significantly improved the performance. Second, unlike other semi-supervised NMF algorithms [20, 57] that estimate the noise dictionary online, we use a noise tracking algorithm [4] and completely get rid of the noise dictionary. Third, our algorithm differs from the IBM-based algorithms [7, 27–29] in how the features are extracted. Most of the existing works use sophisticated handcrafted features from the Computer Auditory Scene Analysis (CASA) [6]. For example, the Amplitude Modulation Spectrograms (AMS) is used in [7]. A composite feature that

includes AMS, Relative Spectral Transform and Perceptual Linear Prediction (RASTA-PLP), Mel-Frequency Cepstral Coefficients (MFCC) and pitch-based features are used in [27]. In contrast, our proposed algorithm uses SNMF to extract features, which is computationally efficient and requires minimum modeling tuning.

5.3.2 The SNMF-DNN-Ratio Algorithm

In this section, we present the proposed enhancement algorithm. First, we provide an overview of the SNMF-DNN-Ratio algorithm shown in Fig. 5.3. The implementation details of the proposed system are discussed latter.

Overview of SNMF-DNN

We consider an additive noisy model:

$$y(t) = s(t) + v(t) \tag{5.8}$$

where $y(t)$, $s(t)$, and $v(t)$ denote, respectively, the noisy observation, the clean speech, and the additive noise at time t. The proposed SNMF-DNN algorithm is composed of a training stage and an enhancement stage. In the training stage, the goal is to train a system to reliably estimate the IRM from the noisy observation $y(t)$. In the enhancement stage, we apply the trained system on $y(t)$ to calculate the estimated IRM, and use it to produce the enhanced speech. The main contribution of this work is to demonstrate that by properly combining SNMF and DNN, IRM can be reliably estimated.

The training stage is composed of four steps plus one optional step. In the training stage, the clean speech, the additive noise, and the noisy mixture are available. Since the true IRM, which will be used for DNN training, can be calculated from the clean speech and the additive noise, it is also available. In the first step, all signals are transformed into a TF representation called *cochleaagram* [6]. In Fig. 5.3, $\{Y, S, V\} \in \mathcal{R}_{+}^{K\times N}$, denotes the cochleagram representation of the noisy speech, the clean speech, and the additive noise, respectively. K denotes the number of frequency bins; N denotes the number of time frames. In the second step, a *noise estimation* module [4] is used to estimate V from Y, denoted as $\hat{V} \in \mathcal{R}_{+}^{K\times N}$. In the third step, sparse NMF, which is composed of a *dictionary training* module and a *sparse coding* module, is used to extract

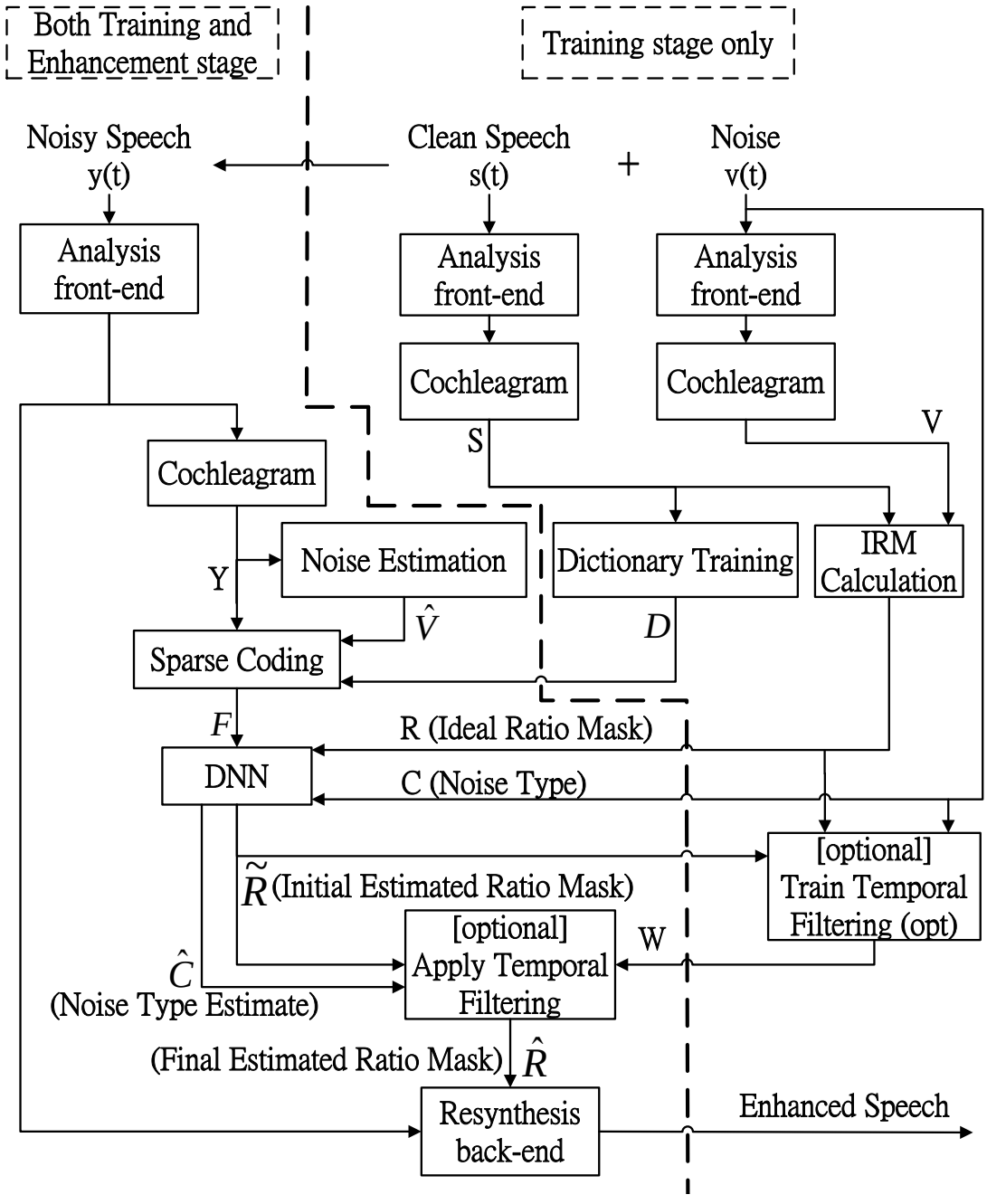


Figure 5.3: Block diagram of the proposed SNMF-DNN system. Modules on the right are those modules used only in the training stage; modules on the left are those used in both training and enhancement stage.

the features F for the noisy speech, which will latter be used to estimate the IRM. In the dictionary training module, a universal speech dictionary $D \in \mathcal{R}_+^{K\times M}$ is trained to capture the characteristics of speech in the cochleagram domain, where M denotes the size of the dictionary. The sparse coding module uses the trained speech dictionary D and the noise estimate $\hat{V}$ to generate the feature F. Using the extracted feature, a *DNN* is trained to estimate both the IRM and the noise type. To better leverage the temporal information, we provide an optional *temporal filtering* step, where a temporal filter W is used to produce the final mask $\hat{R}$. If the temporal filtering is not used, we simply set $\hat{R} = \tilde{R}$. Though enabling the temporal filtering generally improves the performance, it increases the overall system delay. Therefore, we leave it as an option for users.

In the enhancement stage, features are extracted from $y(t)$ by using the *noise estimation* module and the *sparse coding* module. The extracted feature is then fed to the trained DNN to obtain the estimated ratio mask $\tilde{R}$. If the optional temporal filtering is enabled, the final ratio mask $\hat{R}$ is obtained by filtering $\tilde{R}$ with W. Otherwise, we set $\hat{R} = \tilde{R}$ and use it as the final mask. The enhanced speech is obtained by a *resynthesis back-end*, where $\hat{R}$ is used as the weighting coefficient.

We will present the implementation details of each module.

Analysis front-end, Cochleagram, IRM Calculation, and Resynethsis back-end

In the *analysis front-end*, audio files sampled at 16 kHz are passed through a 64-channel Gammatone filterbank with center frequencies spanning from 50 Hz to 8 kHz on the equivalent rectangular bandwidth rate scale. To calculate the *cochleagram*, the output of each filterbank is divided into 20-ms segments with 10-ms overlap. Energy in each segment is calculated and then forms a T-F representation called *cochlegaram* [6]. For example, the matrix $Y \in \mathcal{R}_+^{K\times N}$ denotes the cochleagram representation of the noisy speech, where $K = 64$ denotes the number of channels, and N denotes the number of time frames with each frame corresponds to a 20-ms duration segment.

The signal model in (5.8) can then be re-written in the cochleagram domain:

$$Y \approx S + V \tag{5.9}$$

where S and V respectively represent the signal and the noise in T-F representation.

The approximation in (5.9) comes from the energy calculation step in the cochleagram computation, which is just an approximate linear operation. Assuming both S and V are known, the *IRM*, denoted as $R \in \mathcal{R}_+^{K\times N}$, is defined as follows:

$$R_{k,n} = \frac{S_{k,n}}{S_{k,n} + V_{k,n}} \tag{5.10}$$

where $k \in \{1, \cdots, K\}$ denotes the frequency bin and $n \in \{1, \cdots, N\}$ denotes the time frame.

Let $\hat{R} \in \mathcal{R}_+^{K\times N}$ be the estimated IRM. The *resynthesis back-end* uses $\hat{R}$ to weight the Gammatone filterbank output of each channel. All the 64 weighted streams are then summed to produce the final enhanced speech. We note that in this work, the implementation from Dr. DeLiang Wang research group is used for the analysis front-end, cochleagram calculation, and the resynthesis back-end[5] .

Noise Estimation

Given the noisy mixture Y, we apply the MMSE-based *noise power estimation* algorithm [4] to estimate the noise cochleagram V. The estimated noise, denoted as $\hat{V}$, will be used in the sparse coding module for feature calculation. Compared to the algorithm based on minimum statistics [2], the MMSE-based algorithm has a shorter tracking delay and therefore performs better for non-stationary noises.

Dictionary Training (SNMF)

The purpose of *dictionary training* is to learn a speech model that captures the speech characteristics. Mathematically, it factorizes the speech cochleagram $S \in \mathcal{R}_+^{K\times N}$ into a product of a speech dictionary $D \in \mathcal{R}_+^{K\times M}$ and a sparse coefficient $G \in \mathcal{R}_+^{M\times N}$:

$$S \approx D \times G \tag{5.11}$$

where M denotes the size of the dictionary. The columns of D are called *atoms*, since they are the building blocks for approximating the speech cochleagram. G being sparse means only a *few* atoms are sufficient to represent S. The speech dictionary is usually

[5] See the codes available at http://web.cse.ohio-state.edu/pnl/.

over-complete, i.e., $M \geq K$, in order to well represent different speech signal. SNMF usually performs better than the traditional non-sparse NMF [18, 57, 71, 72].

Computationally, (5.11) is achieved by solving

$$\min_{D\geq 0, G\geq 0} d_{\mathrm{IS}}\left(S \mid DG\right) + \lambda \sum_{m,n} \log\left(\epsilon + G_{m,n}\right) \tag{5.12}$$

where $d_{\mathrm{IS}}(A \mid B)$ denotes the Itakura-Satio (IS) divergence [71] between matrices A and B:

$$d_{\mathrm{IS}}(A \mid B) = \sum_{k,n} \left\{ \frac{A_{k,n}}{B_{k,n}} - \log \frac{A_{k,n}}{B_{k,n}} - 1 \right\}.$$

The reason for choosing the IS divergence as the discrepancy measure is because of it is scale invariance, i.e., $d_{\mathrm{IS}}(c \times A \mid c \times B) = d_{\mathrm{IS}}(A \mid B)$ for any $c > 0$. This property is desirable for signal with huge dynamic range such as speech. Discrepancy measures lacking this property, such as the Euclidean distance, will focus on the large-amplitude frames and ignores the small-amplitude frames, since the large-amplitude frames dominate the objective. The second term in (5.11) is the sparsity-promoting regularization [65] and $\lambda > 0$ is a tunable coefficient. The larger the λ, the sparser the G. The small constant $\epsilon > 0$ is used to make the argument inside logarithm strictly positive. Using the Block Successive Upper-bound Minimization (BSUM) approach [40, 71], problem (5.12) can be solved by alternatingly updating D and G:

$$D \leftarrow D \odot \left\{ \frac{\left[(DG)^{\cdot -2} \odot S\right] G^T}{(DG)^{\cdot -1} G^T} \right\}^{\cdot \frac{1}{2}} \tag{5.13}$$

$$G \leftarrow G \odot \left\{ \frac{D^T \left[(DG)^{\cdot -2} \odot S\right]}{D^T (DG)^{\cdot -1} + \frac{\lambda}{\epsilon + G}} \right\}^{\cdot \frac{1}{2}}. \tag{5.14}$$

Besides being easily implementable, these update rules generate non-increasing objective values and are guaranteed to converge to a stationary point of (5.12), according to the general theorem in [40, Theorem 2].

Sparse Coding (SNMF)

Sparse coding is used to generate the feature for estimating IRM. Let $Y_n \in \mathcal{R}_+^{K\times 1}$ and $\hat{V}_n \in \mathcal{R}_+^{K\times 1}$ be the cochleagram of the noisy mixture and the noise estimate at time

frame n. Assuming the speech dictionary D is given, sparse coding decomposes Y_n into a sum of speech component $DG_n \in \mathcal{R}_+^{K\times 1}$ and a noise component $h_n\hat{V}_n$, i.e.,

$$Y_n \approx DG_n + h_n\hat{V}_n \tag{5.15}$$

where $G_n \in \mathcal{R}_+^{M\times 1}$ and $h_n \in \mathcal{R}_+$ are obtained by solving the sparse optimization:

$$\min_{G_n\geq 0, h_n\geq 0} d_{\mathrm{IS}}\left(Y_n \mid DG_n + h_n\hat{V}_n\right) + \beta\sum_m \log\left(\epsilon + G_{m,n}\right). \tag{5.16}$$

Eq. (5.16) is formulated based on the following three assumptions.

A1. Eq. (5.9) is true, where a noisy cochleagram can be approximately written as a sum of speech and noise.

A2. The noise estimate $\tilde{V}_n$ is a good approximation of the true noise V_n up to nonnegative scaling.

A3. The trained speech dictionary D can indeed represent the speech signal with a sparse gain G_n.

If these assumptions are all satisfied, the estimated speech component DG_n will be a good approximation of the true speech component S, and $h_n\hat{V}_n$ will be close to V_n.

Similar to the dictionary training, (5.16) can be efficiently solved by applying the MM technique:

$$G_n \leftarrow G_n \odot \left\{ \frac{(D)^T\left[(DG_n + h_n\hat{V}_n)^{\cdot -2}\odot Y_n\right]}{(D)^T(DG_n + h_n\hat{V}_n)^{\cdot -1} + \frac{\lambda}{\epsilon + G_n}} \right\}^{\cdot\frac{1}{2}} \tag{5.17}$$

$$h_n \leftarrow h_n \odot \left\{ \frac{\hat{V}_n^T\left[(DG_n + h_n\hat{V}_n)^{\cdot -2}\odot Y_n\right]}{\hat{V}_n^T(DG_n + h_n\hat{V}_n)^{\cdot -1}} \right\}^{\cdot\frac{1}{2}} \tag{5.18}$$

Again, the general theory [40] shows these update rules generate non-increasing objective values and converge to a stationary point of the original problem (5.16).

If Assumptions A1-A3 are all satisfied, we will have $DG_n \approx S_n$ and $h_n\hat{V}_n \approx V_n$. In this case, IRM can be reliably estimated using the following expression:

$$\hat{R}_{k,n} = \frac{[DG_n]_k}{[DG_n]_k + [h_n\hat{V}_n]_k}. \tag{5.19}$$

Since D is a universal speech dictionary, $\tilde{F}_n \triangleq [G_n; h_n\hat{V}_n] \in \mathcal{R}_+^{(M+K)\times 1}$ provides all the necessary information for estimating the ratio mask. In other words, we can view $\tilde{F}_n$ as a feature for estimating the mask. However, if the approximations are inaccurate, a more powerful estimator such as a DNN is needed to tranform $\tilde{F}_n$ to produce high quality estimates. To include more *temporal information*, we concatenate S consecutive frames to form the final feature:

$$F_n = [\tilde{F}_{n-\lfloor S/2 \rfloor}; \cdots; \tilde{F}_n; \tilde{F}_{n+\lfloor S/2 \rfloor}] \in \mathcal{R}_+^{S(M+K)\times 1} \tag{5.20}$$

where $\lfloor x \rfloor$ is the floor operation.

Deep Neural Network

For time frame n, let $R_n \triangleq [R_{1,n}; \cdots; R_{K,n}] \in \mathcal{R}_+^{K\times 1}$ denotes the IRM of all channels, and let $C_n \triangleq [C_{n,1}; \cdots, C_{n,J}] \in \mathcal{R}_+^{J\times 1}$ be the noise type, where J denotes the total number of training noise types and $\forall j \in \{1, \cdots, J\}$,

$$C_{n,j} = \begin{cases} 1, & \text{if time frame } n \text{ belongs to noise type } j \\ 0, & \text{otherwise} \end{cases}$$

In the training stage, the goal is to train a DNN that takes F_n (5.20) as the input feature and $\{R_n; C_n\}$ as the target label, and outputs an estimated IRM $\tilde{R}_n \in \mathcal{R}_+^{K\times 1}$ and an estimated noise label $\tilde{C}_n \in \mathcal{R}_+^{J\times 1}$. In the enhancement stage, F_n is first computed from the noisy observation. Then, F_n is sent to the trained DNN for estimating the mask and the noise type.

Our DNN takes a Rectified linear Unit (ReLU) [66] structure, which uses a rectified activation, $\max(0, z)$, as the non-linear hidden layer and a sigmodial function as the output layer. Compared to traditional neural networks that use sigmodial or hyperbolic tangent function as the hidden layer, ReLU achieves a better performance without unsupervised pre-training [67]. Training the network is performed by minimizing the cross entropy loss over the training set in a purely supervised fashion. Optimization techniques, dropout [73], AdaGrad [74] and infinity-norm bound, are combined with the standard mini-batch Stochastic Gradient Descent for better training performance. Interested readers may refer to the Appendix D for the implementation details.

Temporal Filtering (optional)

Temporal filtering aims to further improve the performance of $\tilde{R}$ by incorporating more temporal information. Theoretically, we can increase S in (5.20) to leverage more temporal information. In practice, however, increasing S may be prohibitive because the larger the S, the higher the feature dimension, and thus the higher the training complexity. Therefore, we provide a simple alternative by doing a post processing on $\tilde{R}$.

In *training temporal filtering*, a temporal filter for each noise type is trained. It is noise-dependent because different noise types have distinct spectral-temporal characteristics. Let $\mathcal{N}^j \triangleq \{n \mid C_{n,j} = 1\}$ be the collection of time frames that corresponds to noise type j. Define $\tilde{R}_n^L \in \mathcal{R}_+^{KL\times 1}$ to be the temporal extension of $\tilde{R}_n$:

$$\tilde{R}_n^L = \left[\tilde{R}_{n-\lfloor L/2\rfloor}; \cdots ; \tilde{R}_n; \cdots ; \tilde{R}_{n+\lfloor L/2\rfloor}\right] \tag{5.21}$$

where $L \geq S$ is the size of the temporal window. For noise type j, the optimal temporal filter $W^j \in \mathcal{R}^{K\times KL}$ is calculated using the least square criterion:

$$\begin{aligned} W^j &= \arg\min_{W^j} \sum_{n\in\mathcal{N}^j} \left\| R_n - W^j \tilde{R}_n^L \right\|^2 \\ &= \left[\sum_{n\in\mathcal{N}^j} R_n (\tilde{R}_n^L)^T\right] \left[\sum_{n\in\mathcal{N}^j} \tilde{R}_n^L (\tilde{R}_n^L)^T\right]^{-1} \end{aligned}$$

In other words, W^j leverages L adjacent frames to better estimate the true ratio mask.

In *applying temporal filtering*, we use the trained filters $W \triangleq \{W^j\}_{j=1}^J$ and the estimated noise type $\hat{C}$ to produce the final estimated ratio mask:

$$\hat{R}_n = \sum_{j=1}^{J} \tilde{C}_{n,j} W^j \tilde{R}_n^L$$

where $\tilde{C}_{n,j}$ and $\tilde{R}_n^L$ are calculated from the output of the DNN. If the optional temporal filtering is disabled, the output of the network, $\tilde{R}_n$, is used as the final mask.

5.3.3 Revisit the SNMF-DNN-Ratio: Design Rationale and Comparison with Existing Works

In this section, we revisit the proposed SNMF-DNN-Ratio algorithm by explaining our design choices in each module and comparing with other similar works [7, 27–29, 75].

Mask Domain

In our design, we choose to perform mask estimation in the 64-channel cochleagram domain instead of the conventional spectrogram domain for two reasons. First of all, in our experience, the spectrogram usually requires 256 channels for a 16 kHz speech signal in order to provide finer enough frequency resolution. From a computational point of view, estimating a mask with only 64 dimension is much easier than a mask with 256 dimension. Second, if the mask is reliably estimated, performing masking in the cochleagram domain suffices to provide a substantial performance improvement [27–29, 75]. However, we want to point out that the proposed system can also be implemented in the spectrogram domain by simply changing the cochleagram computation into the spectrogram computation.

Mask Type

Though IRM (5.10) is used as the target mask in our design, using other mask types is also possible. In [7, 27, 28, 75], the authors use the IBM, which is the binary version of IRM:

$$B_{k,n} = \begin{cases} 1, & R_{k,n} \geq c \\ 0, & R_{k,n} \leq c \end{cases} \tag{5.22}$$

where c denotes a given threshold. For example, c is usually set to -10.4 dB for -5 dB noisy speech [76]. In [29], the authors also investigate another mask called SR-IRM, which is the square root of $R_{k,n}$. In our experiment, we observed that while three masks yield comparable enhanced intelligibility, using IRM as the target mask produces the best enhanced quality.

Feature Extraction

In most existing works [7,27–29], handcrafted CASA-based features are used. The design of these CASA-based features are highly non-trivial and requires extensive domain-knowledge. In contrast, SNMF extracts features with only minimal domain knowledge and can be easily implemented using (5.12) and (5.16). Moreover, the effectiveness of the features extracted by SNMF is validated experimentally in our simulation (to be shown shortly). In fact, sparse coding has been successfully applied in pattern

recognition [77, 78]. Sparse coding is effective because many signals can be naturally represented sparsely under certain transformation. For example, images are known to be sparse under the Discrete Fourier Transform (DCT). Suppose the sparsifying transformation such as the DCT for images is known, then sparse coding finds the corresponding sparse coefficient. It is observed that using the sparse coefficient as the feature, the performance of pattern recognition is comparable, if not better, than using sophisticated hand-crafted features [77, 78]. When the sparsifying transformation is not known *a priori*, the dictionary training algorithm (5.12) can be used to learn the transformation.

In our recent work [75], SNMF is also used to extract features for binary mask estimation. However, [75] requires a pre-trained speech dictionary and a noise dictionary, while in this work only one pre-trained speech dictionary is required. To train a noise dictionary, [75] assumes the access to the enhancement noise during the training stage. In practice, this information is not available, so in this work we use a noise tracking algorithm instead of the noise dictionary training. Our resulting algorithm is therefore applicable to situation without the knowledge of the noise type.

Estimation by DNN

Similar to the recent works [27, 29, 75], a DNN is trained for estimating the IRM. In an ideal case where the separation (5.15) obtained by SNMF is nearly perfect, there is no need for DNN since using the naive estimation rule (5.19) is accurate enough. However, when the approximation error is non-negligible, robust estimators are needed. Among possible estimators, we choose the DNN because of its superior performance in image and audio applications [62]. As the number of layer increases, DNN extracts more and more abstract features, which is argued to be more robust to noise [62]. Despite its superior performance, training DNN is computationally expensive, especially for a large network with a large quantity of training data. Thanks to the recent advance of computing technology, we are able to train our system within a reasonable amount of time. For example, our network contains more than 5 million parameters and more than 2 billion training data is used. Using the GPU computing technology, we can train such a system within 5 days.[6] While training DNN is time consuming, applying

[6] All codes are implemented in Matlab with GPU computing enabled.

DNN, on the contrary, is computationally cheap. In our experiment, our algorithm has a comparable enhancement complexity as the compared NMF-based algorithm.

Temporal Filtering

We introduce an optional temporal filtering in our algorithm. In our experiment, temporal filtering generally improves the overall performance. However, temporal filtering increases the delay from $\lfloor \frac{S}{2} \rfloor$ to $\lfloor \frac{L}{2} \rfloor$, where S is the number of consecutive frames in the feature extraction (5.20), and L is the temporal extension (5.21). In applications where a longer delay is acceptable, user can enable the temporal filtering option. Otherwise, the output of the DNN can be used to produce the enhanced speech directly.

5.4 Simulation for SNMF-DNN-Ratio

In this section, we evaluate the performance of the SNMF-DNN-Ratio algorithm along with other competing algorithms. First, we present the the simulation setup. Second, the implementation detail of all the compared algorithms are discussed. Under different training-enhancement conditions, we demonstrate the superior performance of SNMF-DNN-Ratio by comparing with other competing algorithms. To better understand the masks estimated by various algorithms, we quantitatively study the mask estimation error. Since SNMF-DNN-Ratio has several design parameters in both the training stage and the enhancement stage, we also study the effect of these hyper parameters.

5.4.1 Experiment Setup

The TIMIT database [50] is used as the speech corpus. TIMIT database is divided into two non-overlapping subset, the train subset and the test subset. The train subset is used for training the algorithm, and the test subset is used for performance evaluation. The computer-generated Speech Shape Noise (SSN) and 7 real-world noises from the NOISEX [51] are used as the noise sources. Both speech and noise are re-sampled at 16 kHz. The implementation provided by [5] is used to generate noisy speeches. It first determines the active speech level of the clean speech using the method B of the ITU-T P56 [52]. Then, noise sample is appropriately scaled and add to the clean speech to obtain the noisy speech at the desired SNR.

Although human listening is till the ultimate criterion for evaluating the performance, subjective evaluation is time-consuming and is difficult to standardized. As an alternative, we use three widely-accepted objective measures, the Perceptual Evaluation of Speech Quality sore (PESQ) [30], the Short-Time Objective Intelligibility (STOI) [37], and the Source-to-Distortion Ratio (SDR) [39], as the performance metrics.

PESQ and STOI are used to approximate the subjective evaluation of the enhanced speech quality and intelligibility, respectively. While taking the auditory system into account, PESQ estimates the enhanced speech quality by comparing the loudness difference between the clean speech and the enhanced speech. PESQ ranges between 0.5 (bad quality) and 4.5 (good quality). STOI estimates the perceived speech intelligibility by comparing the signal in the TF domain. It ranges between 0 and 1; the higher the value, the better the intelligibility. In the subjective test conducted in [37], STOI shows a higher correlation coefficient with the subjective listening than the other 5 commonly-used intelligibility measures.

Since monaural speech enhancement is a special case of source separation, we also use SDR, a metric designed for source separation, to evaluate the performance. To compute SDR, the enhanced speech is first decomposed into a speech part and a non-speech part. SDR is defined as the ratio between the two in the dB scale. Therefore, higher SDR means better separation performance.

For any algorithm that involves both training and enhancement stages, it is important to ask the question: what happens if the data used in the training stage do not match those in the enhancement stage. Ideally a good algorithm should not be too sensitive to such mismatch. To answer this question, in our experiment we consider three scenarios, each representing different levels of mismatch between the data used in the training and the enhancement parameters.

S1. Match SNR (-5 dB), Match Noise: In this condition, we consider the perfect match scenario, where the noise type and the SNR level (set to -5 dB) are the same in both stages. Note that the actual speech utterances and noise samples are different across two stages, since the training stage uses the "train" subset and the enhancement stage uses the "test" subset. In the simulation, the result is averaged over three noise types, i.e., babble, factory, and street.

S2. Match SNR (−5 dB), Unmatched Noise: This condition is used to study the performance loss due to noise type mismatch, while the SNR remains at −5 dB. Five noises including SSN, babble, factory, destroyer engine, and street, are used to train the algorithms. In the enhancement stage, we evaluate the average performance on two *unseen* noise types, i.e., factory 2 and tank.

S3. Unmatched SNR (0 dB), Unmatched Noise: This is the most challenging condition, where both the SNR and the noise types are different. The training stage is the same as the previous case, where five noises at −5 dB SNR are used. However, in the enhancement stage, we evaluate the performance on factory 2 and tank noises at 0 dB SNR.

5.4.2 Algorithm Implementation

In our experiments eight enhancement algorithms are compared. In the following, we present the implementation detail of each algorithm. Further, whenever applicable we comment on the implementation differences for algorithms under three training-enhancement scenarios S1–S3.

1. UN: The unprocessed speech is used as the performance lower bound.

2. MMSE-GMA: Among all unsupervised algorithms, MMSE-GMA algorithm proposed in [79] is considered as one of the state-of-the-art. In the spectrogram domain, it assumes that the noise signal follows a complex Gaussian distribution and the speech signal follows a generalized Gamma distribution. The noise power is estimated using the noise tracking algorithm [4], which is also the one used in our algorithm. The Decision-Directed approach [3] is adopted for estimating the speech variance. To balance the residual noise and the signal distortion, the hyper parameters, γ and ν, are set to 2 and 0.1, respectively [79]. Since MMSE-GMA is an unsupervised algorithm, no training stage is needed.

3. BKL-NMF: BKL-NMF [57] is a special variant of NMF that combines one universal speech dictionary with a noise-dependent dictionary. In the training stage, it first trains 20 speaker-dependent dictionaries for 20 randomly selected speakers.

Each speaker-dependent dictionary contains 10 atoms. The universal speech dictionary is obtained by concatenating all 20 dictionaries. It is called the *universal* speech dictionary because they assume utterances of any unseen speaker can be well approximated by linearly combining atoms from the 20 speaker-dependent dictionaries. In the "Match SNR (-5 dB), Match Noise" condition, a 10-atom noise dictionary is trained in the training stage. Then, it is combined with the universal speech dictionary for performing enhancement. In the other two conditions where a pre-trained noise dictionary is not available, BKL-NMF adaptively learns the noise dictionary from the noisy sentences directly. However, as we will verify in the simulation, the enhancement performance degrades if the noise dictionary is not a priori trained. In our experiment, we set the sparsity parameter λ used in Eq. (2) of [57] to 50 in the matched condition and to 20 in the other two unmatched conditions.

4. GFCC-DNN: GFCC-DNN uses a similar structure as SNMF-DNN, except it uses GFCC [28] as the feature. GFCC calculates the cepstral coefficient from the output of the Gammatone filter bank. It has been verified to be a good feature for IBM/IRM estimation [28]. Therefore, we compare with GFCC-DNN in order to validate our previous claim that features extracted by SNMF is good. If the performance of SNMF-DNN is better than GFCC-DNN, we can conclude that SNMF produces a better feature than GFCC. We use a 31-dimension GFCC [28]. To incorporate longer contextual information, we concatenate the GFCC of 5 adjacent frames similar to (5.20). We will defer our discussion on the structure of DNN, as it is the same as what we have used in SNMF-DNN. For simplicity, no temporal filtering is used.

5. SNMF-ERM: SNMF-ERM also uses a similar structure as SNMF-DNN. However, instead of using a DNN to perform IRM estimation, SNMF-ERM uses the direct estimation scheme (5.19) to estimate the ratio mask. Therefore, by comparing SNMF-ERM and SNMF-DNN, we can study the performance gain due to the use of DNN.

6. SNMF-DNN-Ratio: SNMF-DNN-Ratio denotes the proposed algorithm without the optimal temporal filtering step. Table 5.3 summarizes the parameters used for

Table 5.3: Parameters for SNMF-DNN-Ratio and SNMF-DNN-Ratio (TF)

Dictionary Training			Sparse Coding		Temporal Filter	DNN	
M	λ	ϵ	β	S	L	D	N
512	0.01	10^{-5}	0.005	5	11	5	1024

producing the simulation result. These values are experimentally found to yield a good performance. The effect of changing these values will be discussed latter. The size of the speech dictionary M is set to 512 and the sparsity parameter λ is fixed to 0.01. In the sparse coding step (5.16), we found that $\beta = 0.005$ produces the best separation performance, i.e., a minimum $d(S_n \mid DG_n) + d(V_n \mid h_n\hat{V}_n)$ value. We combine 5 consecutive frames to form the final feature (5.20). We use D to denote the depth of the network. For example, $D = 5$ means a 5 layer ReLU network with one input layer, three hidden layer and one output layer is used. N is the number neurons per layer. These values are the same for all three training-enhancement conditions.

7. SNMF-DNN-Ratio (TF): This represents the proposed algorithm with temporal filtering. Compared to SNMF-DNN-Ratio, the only difference is the post temporal filtering, where $L = 11$ frames are concatenated to better leverage the temporal information.

8. IRM: In IRM, we use the *true* ratio mask in the cochleagram domain to perform enhancement. Since the true ratio mask requires access to both clean speech and noise, IRM is not implementable in practice. However, it serves as the performance upper bound for our approach.

5.4.3 Enhancement Performance Comparison

In Table 5.4, we summarize the enhancement result of various algorithms under three training-enhancement conditions. Generally speaking, the proposed algorithms, SNMF-DNN-Ratio and SNMF-DNN-Ratio (TF), outperform other competing algorithms in all training-enhancement setups. Audio samples are available at [80].

When the training and enhancement conditions match (cf scenario S1) , SNMF-DNN-Ratio outperforms MMSE-GMA in all three metrics. Even in the most challenging case where both the SNR and the noise type are different (cf scenario S3), SNMF-DNN-Ratio still have a substantial gain in intelligibility and a comparable enhanced quality. MMSE-GAM, however, even produces a worse intelligibility than the unprocessed speech. We attribute this to the use of supervised training. Even if the training and enhancement conditions do not match, SNMF-DNN-Ratio still can apply the relevant information learned in the training stage to improve the performance.

If we compare SNMF-DNN-Ratio with two other supervised algorithms, BKL-NMF and SNMF-ERM, we observe that SNMF-DNN-Ratio always yields the best performance. We believe this is due to the use of DNN. Despite the subtle difference, both BKL-NMF and SNMF-ERM separate the noisy mixture into a speech part and a noise part by leveraging sparsity. Unfortunately, sparsity alone results in an erroneous separation and is not enough for a high-quality IRM estimation, especially when the training condition does not match the enhancement condition. To correct the estimation error, a noise-robust estimator such as a DNN is needed. By using a large quantity of training data, e.g., more than $20,000$ noisy sentences with each of length 3 to 5 seconds long, the DNN automatically learns the mapping from the erroneous separation to IRM. Therefore, when applying DNN after SNMF, the estimation errors are corrected and the mask performance is significantly improved. Despite the high complexity in training the DNN, SNMF-DNN-Ratio has similar complexity as BKL-NMF in the enhancement stage, since applying DNN in the enhancement stage is computationally cheap.

Comparing SNMF-DNN-Ratio with other IBM/IRM-based systems [7, 27–29], the main difference is how the features are extracted. To study the quality of the features extracted by SNMF, we compare with GFCC-DNN, which uses a similar structure as SNMF-DNN-Ratio, but the DNN is fed with a different set of features (i.e., the GFCC). According to Table 5.4, while GFCC-DNN and SNMF-DNN-Ratio both provide substantial performance improvement over MMSE-MGA, BKL-NMF, and SNMF-ERM, SNMF-DNN performs consistently better in all conditions. In other words, the features extracted by SNMF is comparable, if not better, to GFCC, a known good feature for IBM estimation.

In the two versions of the proposed algorithm, applying a temporal filter is most

Table 5.4: Performance comparison under different training-enhancement conditions. All values are averaged over 192 randomly selected sentences. Among all implementable algorithms, the best two values are shown in bold face.

	Match SNR (−5 dB) Match Noise			Match SNR (−5 dB) Unmatched Noise			Unmatched SNR (0 dB) Unmatched Noise		
	PESQ	STOI	SDR	PESQ	STOI	SDR	PESQ	STOI	SDR
UN	1.45	0.55	-5.31	1.66	0.65	-5.33	2.00	0.75	-0.43
MMSE-GMA [79]	1.62	0.51	1.02	**2.10**	0.64	5.40	**2.46**	0.74	9.47
BKL-NMF [57]	1.47	0.53	0.78	1.90	0.62	2.65	2.21	0.72	6.32
GFCC-DNN	1.86	0.64	4.36	2.05	0.68	6.02	2.40	0.78	9.66
SNMF-ERM	1.39	0.51	-1.74	1.76	0.62	2.21	2.16	0.73	6.67
SNMF-DNN-Ratio	**1.97**	**0.69**	**4.61**	**2.12**	**0.73**	**6.59**	**2.47**	**0.82**	**10.40**
SNMF-DNN-Ratio (TF)	**1.96**	**0.69**	**4.88**	2.08	**0.73**	**6.69**	2.45	**0.82**	**10.39**
IRM	2.65	0.84	6.65	2.85	0.86	8.73	3.09	0.90	11.73

beneficial when the training and the enhancement conditions match. When they do not match, applying a temporal filter may slightly degrade the performance. Experimentally, we observed that applying a temporal filter essentially *smooths* out the estimated mask. This can be seen form Fig. 5.5(d) and Fig. 5.5(e), which shows the masks estimated by SNMF-DNN-Ratio and SNMF-DNN-Ratio (TF). Thus, if the training and the enhancement conditions match, the temporal filter is well trained and the smoothing operation improves the performance. However, when there is a mismatch, smoothing operation actually degrades the performance.

Now, we investigate the quantitative performance of the masks estimated by various algorithms. Let $R_{k,n}$ and $\hat{R}_{k,n}$ be the IRM and the estimated mask at channel k and time frame n. The error in dB scale is defined as follows:

$$\text{Error}_{k,n} = 10 \log_{10} \left(|R_{k,n} - \hat{R}_{k,n}| \right)$$

To better analyze the estimation error, we plot the average error versus the true ratio mask in Fig. 5.4. The masks estimated by four algorithms are presented in Fig. 5.5.

From Fig. 5.4 and Fig. 5.5, it is clear that SNMF-ERM performs the worst, which

means using SNMF alone is not sufficient for a reliable mask estimation. Large estimation error also leads to a poor enhancement performance, which is confirmed from Table 5.4.

Comparing SNMF-DNN and GFCC-DNN, the two algorithms where the DNN is used, we see that the error performance is quite different across various IRM values. In the speech dominant region (where IRM is larger than -10.4 dB, cf. (5.22)), SNMF-DNN outperforms GFCC-DNN and the performance gap increases as IRM increases. In the noise dominant region, however, SNMF-DNN is slightly better than GFCC-DNN when IRM is less than -25 dB and is slightly worse than GFCC-DNN when IRM is larger than -25 dB. In other words, SNMF-DNN outperforms GFCC-DNN when IRM is either close to 1 or close to 0. We believe the difference of error distribution makes SNMF-DNN perform better than GFCC-DNN (see Table 5.4 for performance comparison). To explain this phenomenon, we use the subjective evaluation result reported in [81]. In their work, they study the relative importance of IBM to perceived speech intelligibility in noise. In short, their result shows that in the speech dominant regions, the larger the speech power, the more important the TF unit. In the noise dominant regions, however, it is the TF unit with the largest noise power that is the most important. In other words, the estimation error that happens in regions which IRM is either close to 0 and 1 is more harmful to intelligibility than regions which IRM is between 0 and 1. Using this result, we can now understand why the different estimation error pattern of SNMF-DNN and GFCC-DNN leads to a different overall performance.

5.4.4 Effect of the Parameters

In SNMF-DNN-Ratio, there are several parameters that can be adjusted. In this section, we study the effect of these parameters. For simplicity, we consider a matched condition where -5 dB factory noise is used in both the training and the enhancement stage. No temporal filtering is used. Unless otherwise stated, SNMF-DNN-Ratio is trained using the values reported in Table 5.3.

1. Effect of the Mask Type: As discussed in Sec. 5.3.3, there are three possible target masks, the IRM (5.10), the IBM (5.22), and the square root of IRM (SR-IRM). According to Table 5.5, we observe that using IRM as the target mask

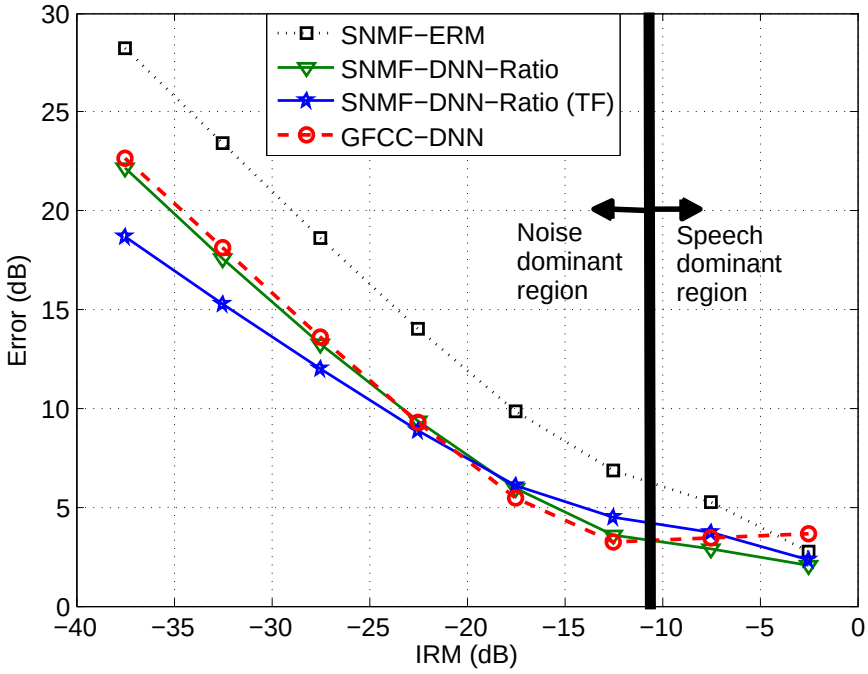
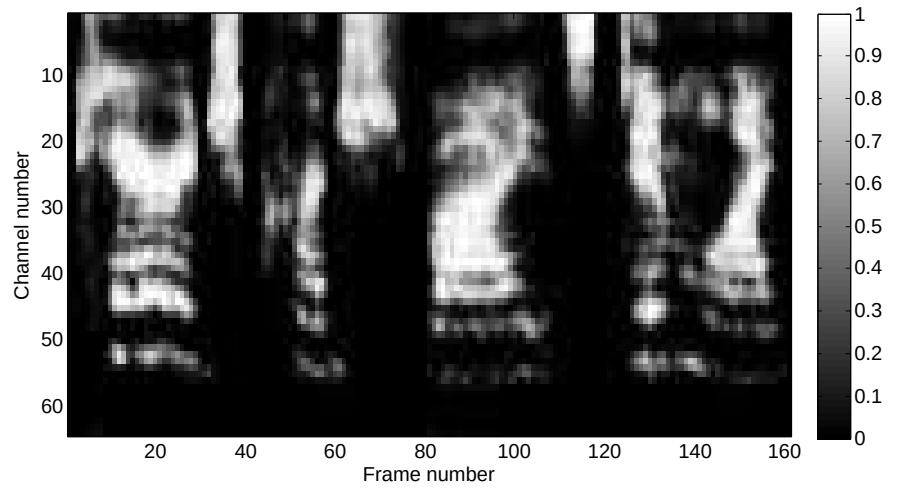


Figure 5.4: Average (SSN, babble, factory, destroyer engine, and street) mask estimation error at −5 dB SNR.

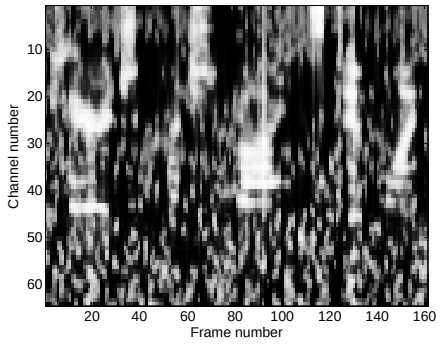
results in the best performance. This result seems to contradict with the result in [29], which claims SR-IRM yields the best performance. We believe this is due to the difference of the feature. In their work, they combine several CASA-based features to form the overall feature. In our system, however, we use SNMF to extract features. Since the features are different, it is possible that the *best* mask type for both algorithms are different.

Table 5.5: Performance comparison of SNMF-DNN-Ratio with different mask types under matched training-enhancement condition (−5 dB factory noise).

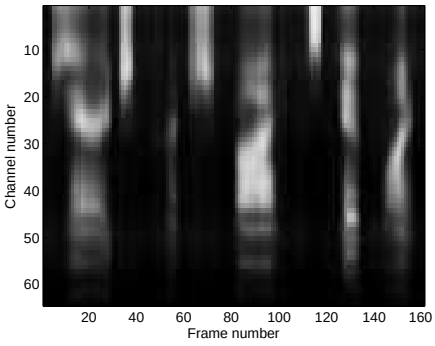
	IBM	SR-IRM	IRM
PESQ	1.67	1.79	1.96
STOI	0.66	0.66	0.67
SDR	1.84	3.23	4.78



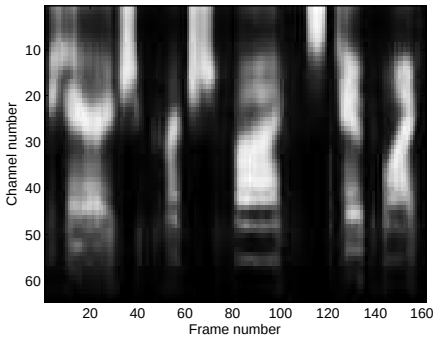
(a) IRM



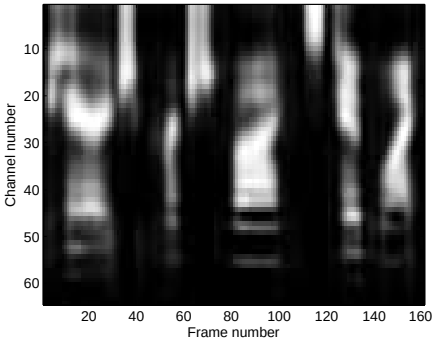
(b) SNMF-ERM



(c) GFCC-DNN

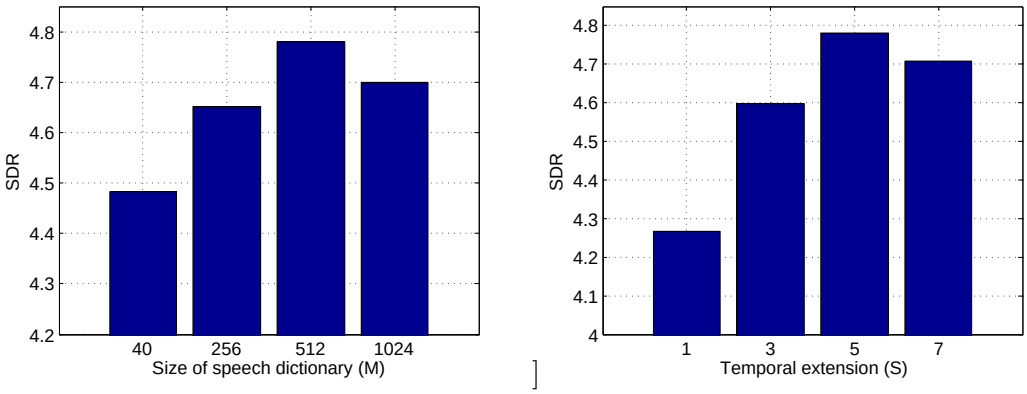


(d) SNMF-DNN-Ratio



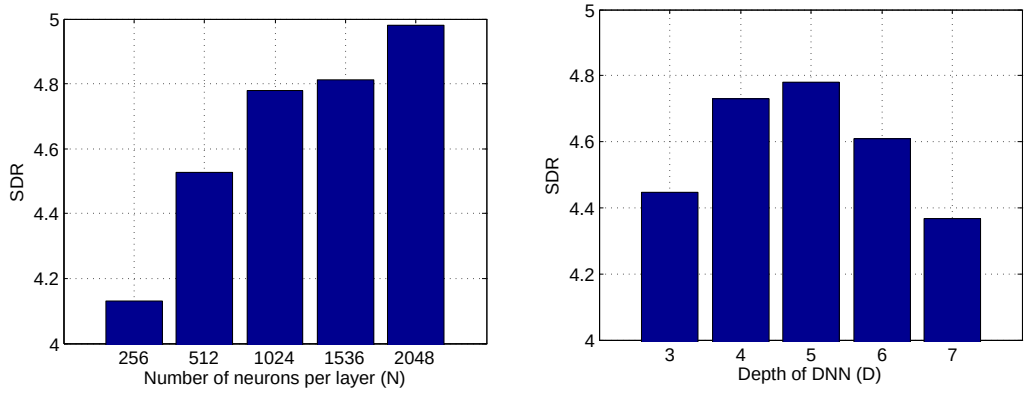
(e) SNMF-DNN-Ratio (TF)

Figure 5.5: Masks estimated by different algorithms under −5 dB factory noise. The top panel denotes the ideal ratio mask, while the remaining panels shows the estimated mask by different algorithms.



(a) (b)

Figure 5.6: SDR of SNMF-DNN-Ratio with different size of speech dictionary and different length of temporal extension under matched training-enhancement setup (−5 dB Factory noise).



(a) (b)

Figure 5.7: SDR of SNMF-DNN-Ratio with different DNN structure, i.e., different number of neurons and different depth, under matched training-enhancement setup (−5 dB Factory noise).

2. Effect of Dictionary Learning and Sparse Coding: Fig. 5.6 presents the performance of SNMF-DNN-Ratio when the speech dictionary size M and the temporal extension S change. Except for $M = 40$, where λ in (5.12) is set to 0, λ is fixed to 0.01 for all other M in order to promote sparsity. Conceptually, when M increases, more speech-specific information can be captured and thus a better performance is anticipated. However, a larger M also means a higher computational complexity. Under the same computational constraint, it is not always beneficial to increase M, which is reflected in Fig. 5.6(a). This argument also explains Fig. 1. It is worth noting that the largest performance improvement comes from $S = 1$ to $S = 3$. This is intuitively reasonable since the closest adjacent frames have the highest temporal correlation, and thus should be most beneficial for IRM estimation.

3. Effect of DNN: Two parameters that control the structure of DNN are N and D, which denotes the number of neurons per layer and the depth of the network, respectively. Fig. 5.7 presents enhancement performance if N or D is changed. First of all, when N increases, the enhancement performance always increases, although the training complexity also increases at the same time. However, this is not the case for the depth, where the performance decreases when the network becomes too deep. We conjecture that this is because increasing the depth makes the network more difficult to train, when compared with only increasing the width. However, more work needs to be done to verify our conjecture.

Chapter 6

Comparison of Three Proposed Algorithms

In the previous chapters, we have presented three algorithms and compared their performance with existing algorithms. In this chapter, we compare the performance of the three proposed algorithms. In Sec. 6.1, the experiment setup is presented. Sec. 6.2 shows the overall performance comparison.

6.1 Experiment Setup

6.1.1 Speech and Noise Corpus

The TIMIT database [50] is used as the speech corpus. Its "train" subset is used in the training stage, if the algorithm requires a training stage. The "test" subset is used for evaluating the performance. The NOISEX [51] database are used as the noise source. Since some of the compared algorithms are supervised algorithms, we consider the following two training-testing scenarios.

1. Matched condition: In this condition, we consider the perfect match scenario, where the noise type and the SNR level are the same in both the training stage and the testing stage. In our simulation, we average over three noises, i.e., babble, factory, and street, at -5 dB SNR.

2. Unmatched condition: This is a practical scenario where the noise used in the training stage and in the testing stage are different. In the testing stage, we test the algorithm on the "factory 2" and the "tank" noise from the NOISEX [51] at -5 dB SNR. In the training stage, different algorithms may use different noise types. However, the noise type used in the testing stage is not included.

In this section, we only test the algorithms at -5 dB SNR. In our previous studies, we observed that our algorithms are most effective when SNR is around -5 dB. When the SNR is 0 dB or higher, traditional Bayesian-based algorithms such as LSTSA suffice to provide a satisfactory performance. In this case, using our proposed algorithms increases the computational complexity without providing a significant performance gain. However, when the SNR is as low as -10 dB, most algorithms, including ours, fail to deliver acceptable performance. Therefore, we focus on -5 dB in this section.

6.1.2 Objective Metrics

The Perceptual Evaluation of Speech Quality (PESQ) [30] is used to measure the enhanced speech quality. The Short-Time Objective Intelligibility measure STOI [37] is used to measure the enhanced speech intelligibility. Please refer to Chap. 2 for more details about these metrics.

6.1.3 Compared Algorithms

The following algorithms are compared. Name inside the parenthesis denotes the abbreviation used in the simulation.

1. Unprocessed speech (UN): The unprocessed noisy speech is used as the performance baseline.

2. Wiener filtering (Wiener): Wiener filtering is undoubtedly the most widely used single channel speech enhancement algorithm due to its simplicity. In our implementation, the decision-directed (DD) approach (1.5) proposed in [1] is used for estimating the *a priori* SNR. The MMSE-based noise tracking algorithm [4] is adopted for noise variance estimation.

3. Causal WNMF (C-WNMF): This is the causal version of the WNMF algorithm that we proposed in Chap. 3.1. In the training stage, Alg. 1 trains a universal speech dictionary. The size of the dictionary M is set to 512, and the sparsity parameter β is fixed to 0.01. In the testing stage, Alg. 2 is applied for enhancement. The sparsity parameter in (3.12) is set to 0.03.

4. Non-causal WNMF (N-WNMF): N-WNMF is the non-causal version of the WNMF. Alg. 3 trains a set of speech dictionaries, and Alg. 4 performs the enhancement. N-WNMF is non-causal, i.e., it can only perform enhancement after receiving the whole sentence. While C-WNMF trains a single speech dictionary, N-WNMF trains four speech dictionaries with each of size 480 atoms. Please refer to Table 3.1 for the parameters used in N-WNMF.

5. Sparse Gaussian Mixture Model (SGMM): SGMM is the second proposed algorithm which extend the traditional NMF and Gaussian model. Alg. 6 is used to train the SGMM for speech. Chap. 4.1.5 explains how to use the trained SGMM for speech enhancement. The parameters are summarized in Table 4.1.

6. SNMF-DNN-Binary (SDB): SDB is the third proposed algorithm. The block diagram is shown in Fig. 5.1. It is a supervised algorithm that combines SNMF and DNN for estimating the ideal binary mask. In the matched condition, the parameters reported in Table 5.1 are used to train the system for each noise type. In the unmatched condition, we simply use the system trained with factory noise for enhancing noisy speeches.

7. SNMF-DNN-Ratio (SDR): SDR is the last proposed algorithm, where the overall system is shown in Fig. 5.3. For simplicity, we do not add the temporal filtering. For both the matched and unmatched conditions, the parameters are the same and are summarized in Table 5.3. However, in the unmatched scenario, we use five noises from the NOIXEX [51], including SSN, babble, factory, destroyer engine, and street, to train the system.

Table 6.1 summarizes the differences between the proposed algorithms. The only algorithm that requires the noise training data is the SDB. This is because SDB uses the noise data to build a noise dictionary. In other algorithms, we use the noise tracking

Table 6.1: Differences between the proposed algorithms. SDR requires noise training data, but the noise type does not need to be the same as the testing noise type.

	Wiener	C-WNMF	N-WNMF	SGMM	SDB	SDR
Speech training data	no	yes	yes	yes	yes	yes
Noise training data	no	no	no	no	yes	no*
System delay (ms)	30	30	whole sentence	60	60	90
Training complexity	zero	low	low	low	high	high
Testing complexity	low	medium	medium	medium	medium	medium
Model complexity	low	medium	medium	medium	high	high

algorithm [4] to estimate the noise component. In terms of the system delay, non-causal WNMF can only perform enhancement after receiving the whole sentence. This is because of the coupling constraint in (3.22). However, the whole sentence delay can be relaxed to 1 to 2 seconds delay in practice. In terms of the training complexity, the two SNMF-DNN algorithms are the highest. This is because training a DNN is very time consuming. In our design, we use a 5 layer network with 1024 neurons per layer. To avoid over-fitting, we us more than 24 hours of speech utterances to train the DNN. Even with a GPU implementation, it takes days to train a system. However, in the testing stage, the complexity is comparable to SGMM and WNMF. This is because *applying* a network is computationally simple, while *training* a network is highly complicated. As for the model complexity, the two SNMF-DNN algorithms again have the highest complexity. This is because we have to store the parameters of the network, in addition to the speech dictionaries. The other algorithms, however, only need to store the speech dictionaries.

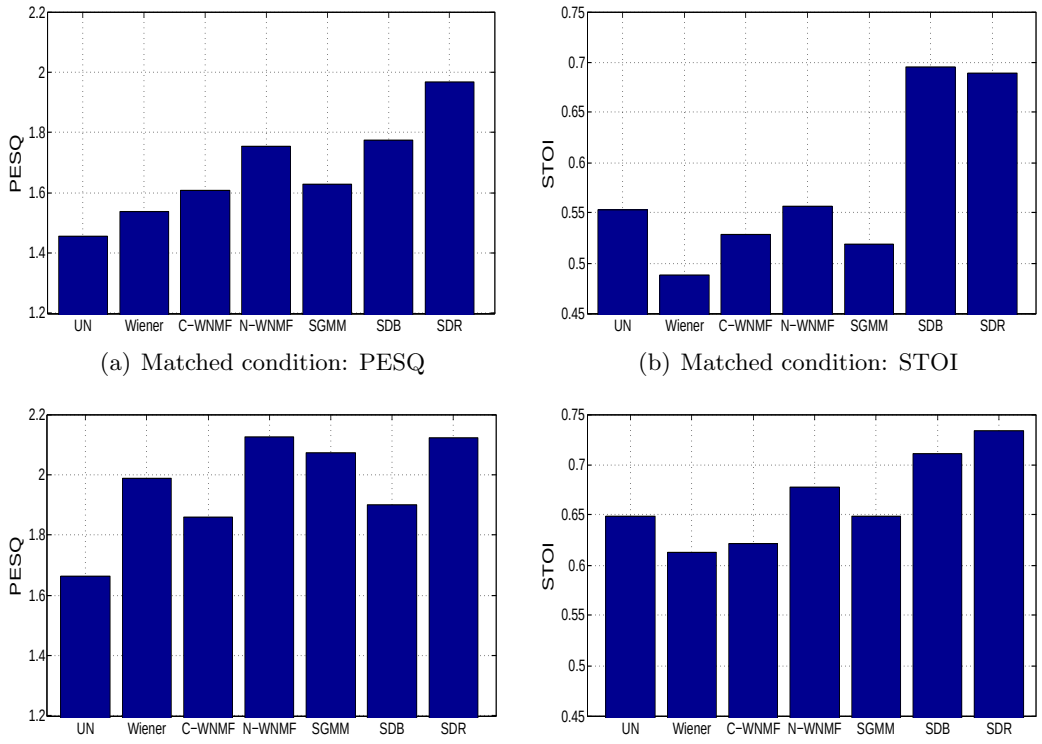
6.2 Enhancement Result

Fig. 6.1 shows the simulation result for both the matched and unmatched condition. In terms of the enhanced speech quality, SNMF-DNN-Ratio and Non-causal WNMF are the best. It is worth noting that both algorithms are non-causal. In non-causal WNMF, the non-causality comes from the coupling of adjacent spectrum frames in the learned dictionary. For SNMF-DNN-Ratio, two future frames, which corresponds to

30-ms delay, are included for the feature extraction. Since adjacent frames are highly correlated, leveraging future frames should help in improving the performance. However, the non-causality of both algorithms make them unsuitable for real-time applications.

From the intelligibility point of view, SNMF-DNN-Binary and SNMF-DNN-Ratio perform the best This is in line with the exist works [29], i.e., if the mask, either the binary mask or the ratio mask, can be reliably estimated, the perceived intelligibility can be significantly improved. However, the price to pay is the complexity. In comparison with WNMF and SGMM, SNMF-DNN is significantly more complex due to the usage of a DNN. However, a powerful estimator is necessary in the success of SNMF-DNN. In Chap. 5 we have shown that if the estimator is changed to a less powerful one, such as a linear SVM, the performance substantially decreases.

The performance of SGMM is in between causal WNMF and non-causal WNMF. Similar to the causal WNMF, SGMM can also be implemented in real time. The main difference between them is how the speech signal is being modeled. In WNMF, we use the classical sparse NMF to model the speech dictionary. In SGMM, we use the newly proposed SGMM, which is an extension of the classical NMF model. This may be the reason why SGMM performs slightly better than the causal WNMF.



(a) Matched condition: PESQ

(b) Matched condition: STOI

(c) Unatched condition: PESQ

(d) Unatched condition: STOI

Figure 6.1: Enhancement result in both the matched and unmatched conditions. Algorithms on each figure denotes, from left to right, the unprocessed speech, the Wiener filtering, the causal WNMF, the non-causal WNMF, the SGMM, the SNMF-DNN-Binary, and the SNMF-DNN-Ratio.

Chapter 7

Conclusion

Single-channel speech enhancement is an important problem due to its widespread application in many speech-related systems. The goal of enhancement is to improve both speech quality and the speech intelligibility. It is intrinsically difficult due to its undertermined nature. Properly leveraging prior information of both speech and noise is essential in developing effective enhancement algorithms.

In this thesis, we proposed three algorithms based on different statistical criteria and machine learning paradigms. In WNMF, we combined Wiener filtering with SNMF. This algorithm intelligently leverages the speech-specific structure within the Wiener filtering framework, which is very different from existing NMF-based algorithms in which NMF is utilized for source separation. Furthermore, our algorithm extends the Wiener filtering by considering speech structure.

The second algorithm, SGMM, generalizes the standard IS-NMF model. It models speech signal as a Gaussian mixture. Only a few Gaussian models can be activated simultaneously by imposing an l0 norm constraint in the formulation. Due to the sparsity constraint, SGMM can remain discriminative even when the size of the dictionary is large. By contrast, the standard IS-NMF inevitably becomes over-representative when the dictionary size increases. Additionally, we proposed an efficient algorithm to train the SGMM, which admits a closed form update in each step.

In the final proposed algorithm, SNMF-DNN estimated the ideal mask by using SNMF to extract features while applying a DNN for mask estimation. What separates SNMF-DNN from the existing IM-based algorithms is the feature extraction. While

existing IM-based algorithms use sophisticated features, we use a simple SNMF for feature extraction. SNMF is easy to implement and it requires no extensive speech domain knowledge. In our simulation, we demonstrated that SNMF-DNN is capable of delivering substantial performance enhancement.

Due to the time constraints, we were unable to investigate the following areas and propose these topics them for future research.

1. Multi-channel speech enhancement: Modern devices are increasingly equipped with multiple microphones. It is of our interest to see how our proposed algorithms can be integrated into a multi-channel speech enhancement setup. In particular, we are interested to see how to combine the proposed approaches with the beamforming techniques.

2. Implementation complexity: In terms of performance, SNMF-DNN is undoubtedly superior. However, it is also the most complex algorithm. For example, the DNN model we use contains more than 10^6 parameters. While large devices such as laptops or cell phones have enough memory to store a large model, it is a problem for smaller devices such as hearing aids. SNMF-DNN will be more practical if we can reduce the model size while still preserving superior performance.

3. Reverberant recording: In this thesis, we assume a simple additive observation model. However, real-life recording always suffers from multi-path effects. We are interested in seeing how to modify the proposed algorithms to the reverberant noisy recording setup.

References

[1] Yariv Ephraim and David Malah. Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 32(6):1109 – 1121, dec 1984.

[2] Rainer Martin. Noise power spectral density estimation based on optimal smoothing and minimum statistics. *IEEE Transactions on Speech and Audio Processing*, 9(5):504–512, 2001.

[3] Yariv Ephraim and David Malah. Speech enhancement using a minimum mean-square error log-spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech and Signal Processing (ICASSP)*, 33(2):443 – 445, Apr. 1985.

[4] Timo Gerkmann and Richard C. Hendriks. Unbiased mmse-based noise power estimation with low complexity and low tracking delay. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(4):1383–1393, 2012.

[5] Philipos C. Loizou. *Speech Enhancement: Theory and Practice*. CRC, 2 edition, February 2013.

[6] DeLiang Wang and Guy J. Brown. *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications*. Wiley-IEEE Press, 2006.

[7] Gibak Kim, Yang Lu, Yi Hu, and Philipos C. Loizou. An algorithm that improves speech intelligibility in noise for normal-hearing listeners. *The Journal of the Acoustical Society of America*, 126(3):1486–1494, 2009.

[8] Patrick J. Wolfe and Simon J. Godsill. Efficient alternatives to the ephraim and malah suppression rule for audio signal enhancement. *EURASIP Journal on Advances in Signal Processing*, 2003(10):1043–1051, 1900.

[9] Thomas Lotter and Peter Vary. Speech enhancement by map spectral amplitude estimation using a super-gaussian speech model. *EURASIP J. Appl. Signal Process.*, 2005:1110–1126, January 2005.

[10] Rainer Martin. Speech enhancement based on minimum mean-square error estimation and supergaussian priors. *IEEE Transactions on Speech and Audio Processing*, 13(5):845 – 856, sept. 2005.

[11] Ioannis Andrianakis and Paul R. White. Mmse speech spectral amplitude estimators with chi and gamma speech priors. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages III–III, 2006.

[12] Bin Chen and Philipos C. Loizou. A laplacian-based {MMSE} estimator for speech enhancement. *Speech Communication*, 49(2):134 – 143, 2007.

[13] Jan S. Erkelens, Richard C. Hendriks, Richard Heusdens, and Jesper Jensen. Minimum mean-square error estimation of discrete fourier coefficients with generalized gamma priors. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(6):1741–1752, Aug. 2007.

[14] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 13:556–562, 2001.

[15] Cédric Févotte, Nancy Bertin, and Jean-Louis Durrieu. Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis. *Neural computation*, 21(3):793–830, 2009.

[16] Paris Smaragdis, Bhiksha Raj, and Madhusudana Shashanka. A probabilistic latent variable model for acoustic modeling. *Advances in models for acoustic processing, NIPS*, 148, 2006.

[17] Emmanuel J. Candés and Michael B. Wakin. An introduction to compressive sampling. *Signal Processing Magazine, IEEE*, 25(2):21–30, 2008.

[18] Mikkel N. Schmidt, Jan Larsen, and Fu-Tien Hsiao. Wind noise reduction using non-negative sparse coding. In *IEEE Workshop on Machine Learning for Signal Processing*, pages 431 –436, Aug. 2007.

[19] Mikkel N. Schmidt and Jan Larsen. Reduction of non-stationary noise using a non-negative latent variable decomposition. In *Machine Learning for Signal Processing, 2008. MLSP 2008. IEEE Workshop on*, pages 486 –491, oct. 2008.

[20] Zhiyao Duan, Gautham J. Mysore, and Paris Smaragdis. Speech enhancement by online non-negative spectrogram decomposition in non-stationary noise environments. In *INTERSPEECH*, 2012.

[21] Nasser Mohammadiha, Paris Smaragdis, and Arne Leijon. Prediction based filtering and smoothing to exploit temporal dependencies in nmf. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013.

[22] Mikkel N. Schmidt and Rasmus K. Olsson. Single-channel speech separation using sparse non-negative matrix factorization. *Interspeech"06, Int. Conf. Spoken Lang. Process.*, 2006.

[23] Tuomas Virtanen. 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, march 2007.

[24] Gautham J. Mysore and Paris Smaragdis. A non-negative approach to semi-supervised separation of speech from noise with the use of temporal dynamics. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 17–20, 2011.

[25] Kevin W. Wilson, Bhiksha Raj, and Paris Smaragdis. Regularized non-negative matrix factorization with temporal dependencies for speech denoising. In *INTERSPEECH*, pages 411–414, 2008.

[26] Kun Han and DeLiang Wang. An svm based classification approach to speech separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4632–4635, May 2011.

[27] Yuxuan Wang and DeLiang Wang. Towards scaling up classification-based speech separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(7):1381–1390, July 2013.

[28] Jitong Chen, Yuxuan Wang, and DeLiang Wang. A feature study for classification-based speech separation at very low signal-to-noise ratio. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014.

[29] Yuxuan Wang, Arun Narayanan, and DeLiang Wang. On training targets for supervised speech separation. Technical report, Technical Report OSU-CISRC-2/14-TR05, Department of Computer Science and Engineering, The Ohio State University, 2014.

[30] 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 *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 749 –752 vol.2, 2001.

[31] Yi Hu and Philipos C. Loizou. Evaluation of objective quality measures for speech enhancement. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(1):229 –238, jan. 2008.

[32] John G Beerends and Jan A. Stemerdink. A perceptual speech-quality measure based on a psychoacoustic sound representation. *Journal of Audio Engineering Society*, 42(3):115–123, 1994.

[33] Stephen Voran. Objective estimation of perceived speech quality. i. development of the measuring normalizing block technique. *IEEE Transactions on Speech and Audio Processing*, 7(4):371–382, Jul 1999.

[34] Schuyler R Quackenbush, Thomas Pinkney Barnwell, and Mark A Clements. *Objective measures of speech quality*. Prentice Hall, 1988.

[35] Dennis H. Klatt. Prediction of perceived phonetic distance from critical-band spectra: A first step. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 7, pages 1278–1281, May 1982.

[36] James M. Kates and Kathryn H. Arehart. Coherence and the speech intelligibility index. *The Journal of the Acoustical Society of America*, 117:2224, 2005.

[37] Cees H. Taal, Richard C. Hendriks, Richard Heusdens, and Jesper Jensen. An algorithm for intelligibility prediction of time-frequency weighted noisy speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2125–2136, Sept 2011.

[38] Jianfen Ma, Yi Hu, and Philipos C. Loizou. Objective measures for predicting speech intelligibility in noisy conditions based on new band-importance functions. *The Journal of the Acoustical Society of America*, 125(5):3387–3405, 2009.

[39] Emmanuel Vincent, Rémi Gribonval, and Cédric Févotte. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1462–1469, 2006.

[40] Meisam Razaviyayn, Mingyi Hong, and Zhi-Quan Luo. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 23(2):1126–1153, 2013.

[41] Md. Kamrul Hasan, Sayeef Salahuddin, and M. Rezwan Khan. A modified a priori snr for speech enhancement using spectral subtraction rules. *Signal Processing Letters, IEEE*, 11(4):450 – 453, april 2004.

[42] Yu. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer, 2003.

[43] Patrick J. Wolfe and Simon J. Godsill. Simple alternatives to the ephraim and malah suppression rule for speech enhancement. In *Proceedings of the 11th IEEE Signal Processing Workshop on Statistical Signal Processing*, pages 496–499, 2001.

[44] Nasser Mohammadiha, Timo Gerkmann, and Arne Leijon. A new linear mmse filter for single channel speech enhancement based on nonnegative matrix factorization. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 45–48, 2011.

[45] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, January 2011.

[46] Gong Chen and Marc Teboulle. A proximal-based decomposition method for convex minimization problems. *Mathematical Programming*, 64(1-3):81–101, 1994.

[47] Xiaoqun Zhang, Martin Burger, and Stanley Osher. A unified primal-dual algorithm framework based on bregman iteration. *Journal of Scientific Computing*, 46(1):20–46, 2011.

[48] Xiangfeng Wang and Xiaoming Yuan. The linearized alternating direction method of multipliers for dantzig selector. *SIAM Journal on Scientific Computing*, 34(5):A2792–A2811, 2012, http://epubs.siam.org/doi/pdf/10.1137/110833543.

[49] Jonathan Eckstein and DimitriP. Bertsekas. On the douglasrachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1-3):293–318, 1992.

[50] John Garofolo, Lori Lamel, William Fisher, Jonathan Fiscus, David Pallett, Nancy Dahlgren, and Victor Zue. DARPA TIMIT acoustic phonetic continuous speech corpus, 1993.

[51] Andrew Varga and Herman J.M. Steeneken. Assessment for automatic speech recognition: Ii. noisex-92: A database and an experiment to study the effect of additive noise on speech recognition systems. *Speech Communication*, 12(3):247 – 251, 1993.

[52] ITU-T P.56. Objective measurement of active speech level ITU-T recommendation p.56., 1993.

[53] Laurent Benaroya, Raphael Blouet, Cedric Févotte, and Israel Cohen. Single sensor source separation using multiple-window stft representation. In *Proc. International Workshop on Acoustic Echo and Noise Control (IWAENC06), Paris, France*, 2006.

[54] Cédric Févotte and Jérôme Idier. Algorithms for nonnegative matrix factorization with the β-divergence. *Neural Computation*, 23(9):2421–2456, 2011.

[55] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[56] Cédric Févotte and Jérôme Idier. Algorithms for nonnegative matrix factorization with the β-divergence. *Neural Computation*, 23(9):2421–2456, 2011.

[57] Dennis L. Sun and Gautham J. Mysore. Universal speech models for speaker independent single channel source separation. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 141–145, May 2013.

[58] Hung-Wei Tseng. Icassp'2015 companion website. `http://ohandyya.wix.com/bioinfo#!sgmm/cny2`, 2014.

[59] Nasser Mohammadiha, Paris Smaragdis, and Arne Leijon. Supervised and unsupervised speech enhancement using nonnegative matrix factorization. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(10):2140–2151, Oct 2013.

[60] Hung-Wei Tseng, Srikanth Vishnubhotla, Mingyi Hong, Xiangfeng Wang, Jinjun Xiao, Zhi-Quan Luo, and Tao Zhang. A single channel speech enhancement approach by combining statistical criterion and multi-frame sparse dictionary learning. In *Proc. Interspeech*, 2013.

[61] Geoffrey Hinton., Li Deng, Dong Yu, George E. Dahl, Abdel rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, Nov 2012.

[62] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, Aug 2013.

[63] Yuxuan Wang and DeLiang Wang. Towards scaling up classification-based speech separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(7):1381–1390, July 2013.

[64] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[65] Emmanuel J. Candés, Michael B. Wakin, and Stephen P. Boyd. Enhancing sparsity by reweighted l1 minimization. *Journal of Fourier Analysis and Applications*, 14(5-6):877–905, 2008.

[66] George E. Dahl, Tara N. Sainath, and Geoffrey E. Hinton. Improving deep neural networks for lvcsr using rectified linear units and dropout. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 8609–8613, May 2013.

[67] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the ICML*, 2013.

[68] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[69] Hung-Wei Tseng. Icassp'2015 companion website. `http://ohandyya.wix.com/bioinfo#!nmf-dnn-ibm/cqse`, 2014.

[70] Yi Hu and Philipos C. Loizou. A comparative intelligibility study of single-microphone noise reduction algorithms. *The Journal of the Acoustical Society of America*, 122(3):1777–1786, 2007.

[71] Augustin Leféevre, Francis Bach, and Cédric Févotte. Itakura-saito nonnegative matrix factorization with group sparsity. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 21–24, May 2011.

[72] Hung-Wei Tseng, Srikanth Vishnubhotla, Mingyi Hong, Jinjun Xiao, Zhi-Quan Luo, and Tao Zhang. A novel single channel speech enhancement approach by combining wiener filter and dictionary learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013.

[73] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[74] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, July 2011.

[75] Hung-Wei Tseng, Mingyi Hong, and Zhi-Quan Luo. Combining sparse nmf with deep neural network: A new classification-based approach for speech enhancement. In *Submitted to IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015.

[76] Ning Li and Philipos C. Loizou. Factors influencing intelligibility of ideal binary-masked speech: Implications for noise reduction. *The Journal of the Acoustical Society of America*, 123(3):1673–1682, 2008.

[77] Rajat Raina, Alexis Battle, Lee Honglak, Benjamin Packer, and Andrew Y. Ng. Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the ICML*, pages 759–766, New York, NY, USA, 2007. ACM.

[78] John Wright, Yi Ma, Julien Mairal, Guillermo Sapiro, Thomas S. Huang, and Shuicheng Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE*, 98(6):1031–1044, June 2010.

[79] Jan S. Erkelens, Richard C. Hendriks, Richard Heusdens, and Jesper Jensen. Minimum mean-square error estimation of discrete fourier coefficients with generalized gamma priors. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(6):1741–1752, 2007.

[80] Hung-Wei Tseng. Companion website. `http://ohandyya.wix.com/bioinfo#!nt-nmf-dnn-irm/cxy3`, 2015.

[81] Chengzhu Yu, Kamil K. Wójcicki, Philipos C. Loizou, John H. L. Hansen, and Michael T. Johnson. Evaluation of the importance of time-frequency contributions to speech intelligibility in noise. *The Journal of the Acoustical Society of America*, 135(5):3007–3016, 2014.

Appendix A

Notations

The following notations will be used through out the thesis. Other notations will be explicitly specified when necessary.

- An upper case letter denotes a matrix, and a lower case letter denotes either a vector or a scalar depending on the context.
- Bold face represents complex-valued quantities.
- x_n denotes the nth column of matrix X
- $x_{k,n}$ denotes the (k, n) entry of matrix X
- $|X|$ denotes a matrix with the same dimension as X, but the entry value of $|X|$ equals to the absolute value of X.
- $\|X\|_1 := \sum_{k,n} |x_{k,n}|$ denotes the sum of absolute values of all entries.
- $\|X\|_F := \sqrt{\sum_{k,n} |x_{k,n}|^2}$ denotes the Frobenius norm of matrix X.
- $\mathcal{C}^{K\times N}$ denotes the set of all $K \times N$ matrices with complex entry values.
- $\mathcal{R}^{K\times N}$ denotes the set of all $K \times N$ matrices with real entry values.
- $\mathcal{R}_+^{K\times N}$ denotes the set of all $K \times N$ matrices with non-negative entry values.
- $\bigodot$, $\geq$, and $\div$ denote entry-wise multiplication, entry-wise "greater or equal to", and the entry-wise division, respectively.
- $X^{\cdot\alpha}$ denotes raising each entry of the matrix X to the power α

Appendix B

Acronyms

Table B.1: Acronyms

Acronym	Meaning
SNR	Signal-to-Noise Ratio
NMF	Non-negative Matrix Factorization
SVM	Support Vector Machine
DNN	Deep Neural Network
TF	Time-Frequency representation
STFT	Short-Time-Fourier-Transform
FFT	Fast-Fourier-Transform
CASA	Computer Auditory Scene Analysis
GMM	Gaussian Mixture Model
SGMM	Sparse Gaussian Mixture Model
PESQ	Perceptual Evaluation of Speech Quality
PSQM	Perceptual Speech Quality Measure
STOI	Short-Time Objective Intelligibility
BASS	Blind Audio Source Separation
SDR	Signal-to-Distortion Ratio
SIR	Signal-to-Interference Ratio

Continued on next page

Table B.1 – continued from previous page

Acronym	Meaning
SAR	Signal-to-Artifacts Ratio
MMSE	Minimum Mean Square Error
DFT	Discrete Fourier Transform
HMM	Hidden Markov Model
IM	Ideal Mask(ing)
AMS	Amplitude Modulation Spectrogram
ACF	AutoCorrelation Function
RASTA-PLP	RelAtive Spectral TrAnsform and Perceptual Linear Prediction
MFCC	Mel-Frequency Cepstral Coefficient
MRCG	Multi-Resolution Cochleagram
GF	Gammatone Filterbank power spectra

Appendix C

Proof of Theorem 1

For the ease of presentation, we re-write (3.21) as follows:

$$\begin{aligned} &\min\ f(h) + p(g) \\ &\text{s.t. } Qh = Mg, h \in \mathcal{H}, g \in \mathcal{G} \end{aligned} \tag{C.1}$$

where h and g denote the concatenation of $\{h_{k,n}\}$ and $\{g_{m,n}\}$, respectively. $f(h)$ corresponds to $\sum_{k,n} J_{MSE}(g_{k,n})$ and $p(g)$ corresponds to $\sum_n \lambda_n \|g_n\|_1$. $Qh = Mg$ denotes the linear constraints in (3.21). $\mathcal{H}$ denotes the box constraint of h, and $\mathcal{G}$ denotes the non-negative constraint of g.

Let $w := (g, h, u)$, where u denotes the dual variable. The proof can be divided into two steps. In the first step, we will prove that $w^r := (g^r, h^r, u^r)$ generated by Algorithm 5 converges to a limit point w^∞. Second, we will show that w^∞ satisfies the optimality condition of (C.1), as shown in (C.2) below.

$$\begin{pmatrix} g - g^\infty \\ h - h^\infty \\ u - u^\infty \end{pmatrix}^T \begin{pmatrix} \partial p(g^\infty) + M^T u^\infty \\ \partial f(x^\infty) - Q^T u^\infty \\ Qh^\infty - Mg^\infty \end{pmatrix} \geq 0, \forall g \in \mathcal{G}, h \in \mathcal{H}, u \tag{C.2}$$

To prove the first step, we use three conditions, i) $f(h)$ is strongly convex and $p(h)$ is convex, ii) the optimality condition (C.2), and iii) the optimality condition (C.3),

which is the result of the three updates in Algorithm 5.

$$\begin{cases} \langle g - g^{k+1}, \partial p(g^{k+1}) + \rho M^T(Mg^k - Qh^k + \frac{u^k}{\rho}) + \frac{1}{\tau}(g^{k+1} - g^k)\rangle \geq 0, \quad \forall g \in \mathcal{G} \\ \langle h - h^{k+1}, \partial f(h^{k+1}) + \rho Q^T(Qh^{k+1} - Mg^{k+1} - \frac{u^k}{\rho})\rangle \geq 0, \quad \forall h \in \mathcal{H} \\ \langle u - u^{k+1}, Qh^{k+1} - Mg^{k+1} - \frac{1}{\beta}(u^k - u^{k+1})\rangle \geq 0, \quad \forall u \end{cases}, \text{ where } \tau = \frac{1}{\rho L} \tag{C.3}$$

Let w^* denote any primal-dual optimal solution pair of (C.1). Define a matrix $H = \begin{pmatrix} \frac{1}{\tau}I - \rho M^T M & 0 & 0 \\ 0 & \rho Q^T Q & 0 \\ 0 & 0 & \frac{1}{\rho}I \end{pmatrix}$, where $\tau = \frac{1}{\rho L}$. According to the definition of L in (3.32), $\frac{1}{\tau}I - \rho M^T M$ is a positive definite matrix. We first want to prove that the sequence $\{V^r\}$, associated with w^r, in monotonically non-increasing.

$$\begin{aligned} V^r &:= \|w^r - w^*\|_H^2 = (w^r - w^*)^T H(w^r - w^*) \\ &= \|g^r - g^*\|^2_{\frac{1}{\tau}I - \rho M^T M} + \|h^r - h^*\|^2_{\rho Q^T Q} + \|u^r - u^*\|^2_{\frac{1}{\rho}I} \end{aligned}$$

After rearranging (C.3), we can obtain the first inequality below.

$$\begin{aligned} & (w^* - w^{r+1})^T H(w^k - w^{r+1}) \\ \leq & \begin{pmatrix} g^* - g^{r+1} \\ h^* - h^{r+1} \\ u^* - u^{r+1} \end{pmatrix}^T \begin{pmatrix} \partial p(g^{r+1}) + M^T u^{r+1} \\ \partial f(h^{r+1}) - Q^T u^{r+1} \\ Qh^{r+1} - Mg^{r+1} \end{pmatrix} && \text{(C.4)} \\ & + \begin{pmatrix} g^* - g^{r+1} \\ h^* - h^{r+1} \end{pmatrix}^T \begin{pmatrix} -\rho M^T Q(h^r - h^{r+1}) \\ \rho Q^T Q(h^r - h^{r+1}) \end{pmatrix} && \text{(C.5)} \\ \leq & (u^{r+1} - u^r)^T Q(h^r - h^{r+1}) && \text{(C.6)} \\ \leq & 0 && \text{(C.7)} \end{aligned}$$

Using the fact that $p(g)$ and $h(h)$ are convex functions, and also the fact that w^* satisfies optimality condition (C.2), we can show that (C.4) is less than or equal to 0. (C.5) can be shown to be equal to (C.6) by using the u^{r+1} update rule and $Qh^* = Mg^*$. Combine these two forms the second inequality. The last inequality is obtained by the inference of the h optimality condition in (C.3). Further,

$$
\begin{aligned}
V^{r+1} &= \|w^{r+1} - w^*\|_H^2 \\
&= \|w^r - w^* + w^{r+1} - w^r\|_H^2 \\
&= \|w^r - w^*\|_H^2 + 2(w^r - w^*)^T H(w^{r+1} - w^r) \\
&\quad + \|w^{r+1} - w^r\|_H^2 \\
&\leq V^r - \|w^{r+1} - w^r\|_H^2
\end{aligned}
$$

As a result, $\{V^r\}$ is an non-increasing sequence, and $\lim_{r\to\infty} \|w^{r+1} - w^r\|_H^2 = 0$. Since $\frac{1}{\tau}I - \rho M^T M$ and $\frac{1}{\rho}I$ are positive definite matrix, we have

$$
\lim_{r\to\infty} \|g^r - g^{r+1}\| = 0, \quad \lim_{u\to\infty} \|u^r - u^{r+1}\| = 0
$$

$\{g^r\}$ and u^r are therefore Cauchy sequences. By the completeness of real numbers, $\{g^r\}$ and u^r converges to a limit point, denoted by g^∞ and u^∞. Since $f(h)$ is strongly convex, h update in Algorithm 5 has a unique solution. Therefore, $\{h^r\}$ also converges to a limit point h^∞. This complete the first step of the proof.

In the second step, we want to show that w^∞ satisfies the optimality (C.2). This can be easily proved by taking $k \to \infty$ in (C.3). ■

Appendix D

DNN Training

Algorithm 8 summarizes the implementation detail of the DNN training used in the SNMF-DNN-Ratio algorihtm (Chap 5.3). In essence, it is a stochastic gradient descent (SGD) algorithm with three additional optimization techniques, dropout [73], AdaGrad [74], and infinity-norm bound.

At each iteration, a mini-batch of size m are selected. These samples are propagated through the network (D.1) - (D.3). While the inner layer uses the rectified activation (D.4) as the non-linear mapping, the output layer uses the traditional sigmoidal function (D.5). To train the network, the first optimization technique we use is "dropout", which is implemented by multiplying the neurons in each layer with a random vector b_n^d, with each of its components drawn i.i.d. from a Bernoulli distribution Bernoulli$(1-p)$. In other words, each link is randomly *dropout* with probability p. Besides being easily implementable, the dropout technique prevents the network from overfitting the training set and yields a better generalization performance [73]. Then, standard back-propagation is used to calculate the gradient (D.6), while the cross-entropy (D.7) is the training objective. In (D.8), the network weight W_r^d is updated by first using the AdaGrad [74] as the adaptive step size, and then projecting to the infinity-norm ball. Compared with constant step size used in vanilla SGD, AdaGrad uses different step size for each coordinate, which improves the convergence speed both theoretically and experimentally [74]. The projection operation is to ensure the boundedness of the network weight.

Algorithm 8 DNN training with dropout, AdaGrad, and infinity-norm bound

Require:

- Parameter: dropout rate p, step size α, ∞-norm bound c, mini-batch size m
- Data: feature-label pair $\{F_n, L_n\}_{n=1}^N$

Ensure: network weight $\{W^d\}_{d=1}^D$

Calculating the scaled infinity-norm bound: $c_p = \frac{c}{1-p}$

for iteration $r = 0$ to $R-1$ **do**

Randomly selecting a few samples $\mathcal{N}_r \subset \{1, \cdots, N\}$, where $\mathcal{N}_r$ has size m.

Calculating neuron values with dropout [73] for all $n \in \mathcal{N}_r$.

$$h_n^0 = F_n \tag{D.1}$$

$$h_n^d = \sigma^{Hid}(W_r^d(b_n^{d-1} \odot h_n^{d-1})), \quad 1 \le d < D \tag{D.2}$$

$$h_n^D = \sigma^{Out}(W_r^D(b_n^{D-1} \odot h_n^{D-1})) \tag{D.3}$$

where

$$\sigma^{Hid}(v) \triangleq \max(v, 0) \tag{D.4}$$

$$\sigma^{Out}(v) \triangleq \frac{1}{1+e^{-v}} \tag{D.5}$$

and b_n^d is a vector of Bernoulli random variables each of which has probability $1-p$ of being 1.

Calculating gradients using back propagation

$$g_r^d \triangleq \frac{1}{m} \frac{\partial}{\partial W_r^d} \sum_{n \in \mathcal{N}_r} E(L_n, h_n^D) \tag{D.6}$$

where

$$E(a, b) \triangleq -\left[\sum_j a_j \log(b_j) + (1-a_j)\log(1-b_j)\right] \tag{D.7}$$

Updating W_r^d with AdaGrad [74] and ∞-norm bound

$$W_{r+1}^d = \prod_{\|\cdot\|_\infty \le c_p} \left[W_r^d - \frac{\alpha}{\sqrt{\sum_{i=1}^r (W_i^d)^{\cdot 2}}} \odot g_r^d\right] \tag{D.8}$$

where

$$\prod_{\|\cdot\|_\infty \le c_p} [v] \triangleq \max(\min(v, c_p), -c_p)$$

end for

Weight scaling

$$W^d = (1-p) W_R^d, \quad \forall d = 1, \cdots, D$$