New Methods of Complex Matrix Factorization for Single-Channel Source Separation and Analysis

Brian John King

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2012

Reading Committee:

Les Atlas, Chair

Maya Gupta

Bhiksha Raj

Jay Rubinstein

Program Authorized to Offer Degree:
Department of Electrical Engineering

University of Washington

Abstract

New Methods of Complex Matrix Factorization for Single-Channel Source Separation and Analysis

Brian John King

Chair of the Supervisory Committee:
Professor Les Atlas
Electrical Engineering

Throughout the day, people are constantly bombarded by a variety of sounds. Humans with normal hearing are able to easily and automatically cut through the noise to focus on the sources of interest, a phenomenon known as the "cocktail party effect." This ability, while easy for humans, is typically very challenging for computers. In this dissertation, we will focus on the task of single-channel source separation via matrix factorization, a state-of-the-art family of algorithms.

In this work, we present three primary contributions. First, we explore how cost function and parameter choice affect source separation performance, as well as discuss the advantages and disadvantages of each matrix factorization model. Second, we propose a new model, complex matrix factorization with intra-source additivity, that has significant advantages over the current state-of-the-art matrix factorization models. Third, we propose the complex probabilistic latent component analysis algorithm, which can be used to transform complex-valued data into nonnegative data in such a way that the underlying structure in the complex data is preserved. We also show how these new methods can be applied to single-channel source separation and compare them with the current state-of-the-art methods.

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY

ASR: Automatic Speech Recognition

BSS: Blind Source Separation

CASA: Computational Auditory Scene Analysis

CMF: Complex Matrix Factorization

CPLCA: Complex Probabilistic Component Analysis

EM: Expectation-Maximization

GMM: Gaussian Mixture Model

HMM: Hidden Markov Model

IBM: Ideal Binary Mask

ICA: Independent Component Analysis

IS: Itakura-Saito

ISTFT: Inverse Short-Time Fourier Transform

KL: Kullback-Liebler

LVD: Latent Variable Decomposition

MLE: Maximum-Likelihood Estimation

MM: Maximization-Minimization

MSE: Mean squared error

NCMF: Nonnegative Complex Matrix Factorization

NMF: Nonnegative Matrix Factorization

PCA: Principal Component Analysis

PESQ: Perceptual Evaluation of Speech Quality

PLCA: Probabilistic Latent Component Analysis

PLSA: Probabilistic Latent Semantic Analysis

SAR: Source-to-Artifact Ratio

SDR: Source-to-Distortion Ratio

SIR: Source-to-Interference Ratio

STFT: Short-Time Fourier Transform

SVD: Singular Value Decomposition

TMR: Target-to-Masker Ratio

ACKNOWLEDGMENTS

As I contemplate completing the final chapter of my 26-year journey as a student, I think about all the people who have offered me invaluable knowledge, lessons, support, encouragement, and inspiration. I would first like to thank my academic adviser, professor Les Atlas, for his guidance and support during my entire graduate school career. I would like to thank those on my PhD committee, professors Maya Gupta, Bhiksha Raj, Jay Rubinstein, and Adrian "KC" Lee. They have challenged me to dig deeper into the math and back up everything I said with mathematical proofs, which made my theoretical understanding of my topic much more concrete as a result.

I would like to thank other mentors along the way. I would like to thank Hirokazu Kameoka and Gautham Mysore, for their inspiration and guidance in my PhD research. I would like to thank my mentors, managers, and coworkers during my many internships at Adobe, including Paris Smaragdis, Gautham Mysore, David Salesin, Paul Ellis, Paul Siegel, Jamie Gjerde, Shenzhi Zhang, Charles Van Winkle, Durin Gleaves, and Matt Stegner. I would like to thank professors Bhiksha Raj and Rita Singh for hosting me several times as a student visitor at Carnegie Mellon University. At times, I learned more in those epic 16-hour work-a-thon days than ever before in such a short time. I would like to thank professor Cédric Févotte for inviting me to collaborate with him at Télécom ParisTech during the spring of 2012. During that time, in addition to writing a conference paper, his expertise in the field helped me finish up my PhD research so that I was able to return to Seattle to write my entire dissertation in one quarter. Finally, I am thankful for the Air Force Office of Scientific Research, Adobe, and Télécom ParisTech, who provided me the resources and

flexibility to attend conferences and visit universities around the world to share my knowledge, receive insights from, and collaborate with others.

I would like to thank my fellow labmates and graduate students at the University of Washington. I would like to thank Pascal Clark in particular, for his many insights and ideas regarding my research. I would also like to thank Elliot Saba for helping me put together my first large scale set of complex matrix factorization experiments where we experimented with GPU and cluster computing. Although our heavy computational load caused a minor computer fire and allegedly brought a cluster to its knees and its admin into a fit of rage, we were finally able to complete the experiments, which I used for my first IEEE journal paper. And perhaps even more important than the technical support was the non-technical support during the emotional roller-coaster of graduate school. During the highs and lows of graduate school, including mono, qualifying exams, paper deadlines, negative results, unpredictable funding, and a computer spontaneously combusting, there was always someone to talk with and encourage me.

Finally, I would like to thank my friends and family for their love and support. I would first like to thank my parents for their lifelong support. During the years my parents homeschooled me, they helped me to develop invaluable study skills, a strong work ethic, and a curiosity to explore the unknown. I also thank them for their support and encouragement during my college years. I would like to thank my sister, for helping instill in me a competitive spirit and for her encouragement. I would like to thank my friends, who would encourage me to not give up when times were hard, and for giving me grace when we hung out after a long day when my brain was fried and was probably not that fun to spend time with. I would like to thank my lovely wife and best friend, Karelisa, who has stuck with me through my entire life as a graduate student, first as a friend, then girlfriend, then wife. It is not easy

being a graduate student, but I think it is often more difficult to be married to one. She always offered her love, support, and encouragement. There were some tough times, especially with the financially instability of a full-time graduate student, but she stuck with throughout it all. She has definitely kept her vows of sticking with me "for poorer," and am looking forward to a new season of "for richer." Finally, I would like to thank Jesus Christ, my Lord and savior, for giving me the talent, support, friends, and opportunities that have helped me to succeed, and the strength and encouragement to persevere.

DEDICATION

To Karelisa, my lovely wife and best friend.

Chapter 1

INTRODUCTION

1.1 Problem Statement and Applications

During a typical day, people are exposed to a virtual polyphonic orchestra of sounds coming simultaneously from several sources. From noisy parties with a multitude of conversations all competing to be heard over music, to a "quiet" office with murmuring conversations, taps on keyboards and mice, and the steady hum of an air conditioner, people with normal hearing are able to focus on sources of interest and tune out everything else easily and without thinking. This task of automatic source separation, although trivial and natural for humans, is typically very difficult for computer systems. There have been many breakthroughs over the last few decades since the beginning of the field of digital signal processing, yet low-distortion, source- and location-invariant source separation remains an open problem. One of the most promising recent developments in source separation is nonnegative matrix factorization (NMF), which models a multi-source signal as a linear combination of source-specific building blocks, or "bases". A potential problem with NMF and related algorithms, however, is that they estimate only the magnitude component of the complex-valued short-time Fourier transform (STFT). In the case of overlapping signals, it implicitly assumes that at each time-frequency point in the STFT, individual sources share a common phase, which we will later show in this document to be incorrect. We will also demonstrate how this phase assumption can cause incorrect estimations in both the phase and magnitude of the individual sources, limiting separation performance and introducing audible distortion, and how these problems can be solved by adding phase into the NMF framework [25, 26]. However, complex

matrix factorization, while satisfying superposition in the complex STFT domain, is currently limited in its usefulness due to overparameterization.

The contributions of this dissertation can be divided into three categories. For our first contribution, we present how parameter choice and training methods affect performance in nonnegative and complex matrix factorization-based speech separation. For our second contribution, we present complex matrix factorization with intra-source additivity and its application in single-channel source separation. This new model is a hybrid of nonnegative and complex matrix factorization that combines many of the best characteristics found in both. For the third contribution, we present complex probabilistic latent component analysis, a novel method to transform complex data into nonnegative data lying in a simplex. The complex data's underlying structure is preserved in the new space so that nonnegative matrix factorization methods may be used to model and separate sources. This is useful because it makes it possible to apply many of the advanced nonnegative matrix factorization and probabilistic latent component analysis methods to complex data, such as short-time Fourier transform, in such a way that superposition is satisfied.

Although the algorithms and principles can be applied to a variety of signals and applications, we will be focusing on the task of single-channel source separation of audio. There are many applications where single-channel source separation is integral, such as noise reduction and source isolation for use in hearing aids, cochlear implants, and automatic speech recognition systems. Hearing aids, cochlear implants, and automatic speech recognition systems work well in low-noise environments, but rapidly degrade in the presence of noise or overlapping talkers. Other applications for speech include real-time noise reduction, or offline processing for increasing intelligibility for audio forensics. There are many interesting applications in music as well, such as isolating instruments within a monophonic recording so that effects or panning can be applied to instruments individually or to be used as inputs to a polyphonic transcription system. These are some of the most common applications of single-channel

source separation, but there are many other exciting applications as well as many more that have yet to be discovered.

1.2 Prior Work

There is a rich history of work in DSP related to source separation and the closely-related field of source enhancement, which can be viewed as separating the target and noise sources. This list is not meant to be exhaustive, but is meant to give an overview of the most influential work as well as the most closely-related work in the subject. This section will also focus on single-channel methods, although the most common multi-channel methods will be discussed at the end of the section. The single-channel algorithms can be divided into two broad categories, filter-based and synthesis-based. The common feature with all filter-based methods is that they all modify, or filter the original signal. In contrast, the synthesis-based algorithms all synthesize their separated or enhanced signals from their underlying models. This section will first discuss single-channel filter-based, then single-channel synthesis-based, and then multi-channel algorithms.

The first major contribution of filter-based processing is Wiener filtering [69], which assumes that both the target and noise are additive, stationary, and independent, with known statistical characteristics such as autocorrelation. The signal is enhanced by use of a Wiener filter designed to minimize the mean squared error of the filtered signal. Another related method is spectral subtraction, which subtracts the spectral magnitude of the noise estimate from the mixed signal [4]. Similar to the previous method, the noise is assumed to be additive and typically stationary. If the magnitude of the subtracted signal spectra become negative, due to either poor noise estimation or non-additive phases in the two signals, those points may be set to zero or a minimum nonnegative value [3].

While the previous two filter-based methods typically assume stationarity in both signals, like Wiener filtering, or in just the noise signal, such as spectral subtrac-

tion, the next methods dynamically update their filters to enhance and separate non-stationary signals. The first method is comb filtering, which assumes that at least one of the signals is harmonic [57, 40], which can work for such signals as harmonic speech and instruments. Another model that makes an even broader assumption is coherent modulation filtering, which uses the assumption that at least one of the signals is locally narrowband and that the signals are not overlapping in both time and frequency simultaneously [53, 8]. The final type of models to be discussed in this group are associated with computational auditory scene analysis (CASA) [6]. It is based on the theory that when humans experience a rich audio environment, or scene, they make sense of it by dividing it up into separate audio streams depending on their attention [5]. The task, then, is to develop algorithms to automatically divide a matrix of time-frequency points into disjoint sets representing the streams. One method is to classify each time-frequency point to create a binary mask. Methods include identifying streams based on common onset and offset, harmonicity, and other spectro-temporal information. The goal of these mask-finding methods is to approach the ideal binary mask [68], which is computed as follows:

$$Y_{s,f,t} = \begin{cases} X_{f,t} & \text{if } |X_{s,f,t}| > |X_{r,f,t}|, \text{for all } r \neq s \\ 0 & \text{otherwise} \end{cases} \tag{1.1}$$

The key of the ideal binary mask is that it gives the maximum signal-to-noise ratio possible with a binary mask. There are other CASA-inspired models, such as harmonic-temporal clustering [30], that instead classify each spectro-temporal point as a mixture of sources.

In contrast to filter-based methods, which modify the original signal, the synthesis-based methods instead assume a model for the source or sources, estimate the model's parameters, and synthesize an entirely new signal from the parameters. One popular such model for narrowband and harmonic signals is sinusoidal modeling [49, 35], which analyzes and synthesizes speech with a summation of sinusoids frame-by-frame. Parameters for estimation include sinusoidal frequency, amplitude, and phase. Applying

this technique to speech can exhibit significant noise reduction in harmonic signals, but drawbacks include difficulty in accurately modeling rapid changes in pitch and other parameters, as well as failure to accurately model inharmonic speech. Another broad category of methods use machine learning principles such as Gaussian mixture models (GMM) and hidden Markov models (HMM) to model the different sources [51, 2].

In addition to single-channel methods, there are many methods that require multiple channels. Since the focus of this thesis is single-channel source separation, these methods will not be discussed in depth, but are worth mentioning. The first method is beam-forming, which uses an array of microphones to "focus attention" on a specific location or angle in the environment [15]. The simplest form of this method works by using weighted sums and differences with phase delays between channels to effectively boost the desired source location and cancel out the others. Further improvements can be made by also taking into account the convolutional characteristics of the environment. While beam-forming can work quite well, it requires information about the location of the sources. When the locations are unknown, the problem is known as blind source separation (BSS). The most common method of BSS employs independent component analysis (ICA), which finds the multiplicative or convolutional weights to linearly combine the microphone channels to create signals with the maximum statistical independence [45, 52]. Although there are some methods that attempt to surpass these, typical limitations of ICA include knowing the number of sources *a priori*, requiring the locations of the sources be relatively stationary and a sufficient distance away from each other, and requiring the number of channels equal or exceed the number of sources. In optimal conditions, these multi-channel methods can work quite well, but their inherent limitations, such as source locations and microphone array patterns, point to the need for single-channel methods. Another advantage of single-channel approaches is that many can be modified to take advantage of multi-channel signals if available, but the inverse is not true for multi-

channel methods. Finally, single-channel methods can be used in conjunction with multi-channel methods in a complementary fashion.

In conclusion, it is worth repeating that the above list is not meant to be a comprehensive survey of source separation and enhancement methods, but is meant to give a brief overview of the most well-known methods as well as those most relevant to this dissertation's topics of complex matrix factorization with intra-source additivity, complex probabilistic latent component analysis, and their application to single-channel audio source separation. The three most closely related source separation methods, nonnegative matrix factorization, complex matrix factorization, and probabilistic latent component analysis will be discussed in much more detail in the background section to follow.

Chapter 2
BACKGROUND

This dissertation has its roots in nonnegative matrix factorization, complex matrix factorization, and nonnegative probabilistic latent semantic analysis. In this section, the theory, algorithms, strengths, and weaknesses of these methods will be discussed.

2.1 Nonnegative Matrix Factorization

Nonnegative matrix factorization (NMF) is a matrix decomposition method that approximates a matrix with nonnegative elements as a product of two other matrices with nonnegative elements [33].

$$\begin{gathered} |X| \approx BW, \\ \text{where } X \in \mathbb{R}^{\mathbb{F}\times\mathbb{T}}, B \in \mathbb{R}^{\mathbb{F}\times\mathbb{K}}, W \in \mathbb{R}^{\mathbb{K}\times\mathbb{T}}, \\ X \geq 0, B \geq 0, W \geq 0 \text{ for all elements} \end{gathered} \tag{2.1}$$

The first matrix, B, is commonly referred to as the base matrix, while W is commonly referred to as the weight matrix. The columns in base matrix B, commonly referred to as basis vectors, can be thought of as building blocks. The product of the matrices estimates each column t in X with a linear combination of bases. Therefore, column t in weight matrix W contains the weights approximating the corresponding column in X.

Source separation works because of the summation of the bases embedded in the matrix multiplication. In order to separate the signals after the base B and weight W matrices are calculated, the basis elements and weights corresponding to a specific source are multiplied together (see separation step in Figure 2.1). Typically, the basis

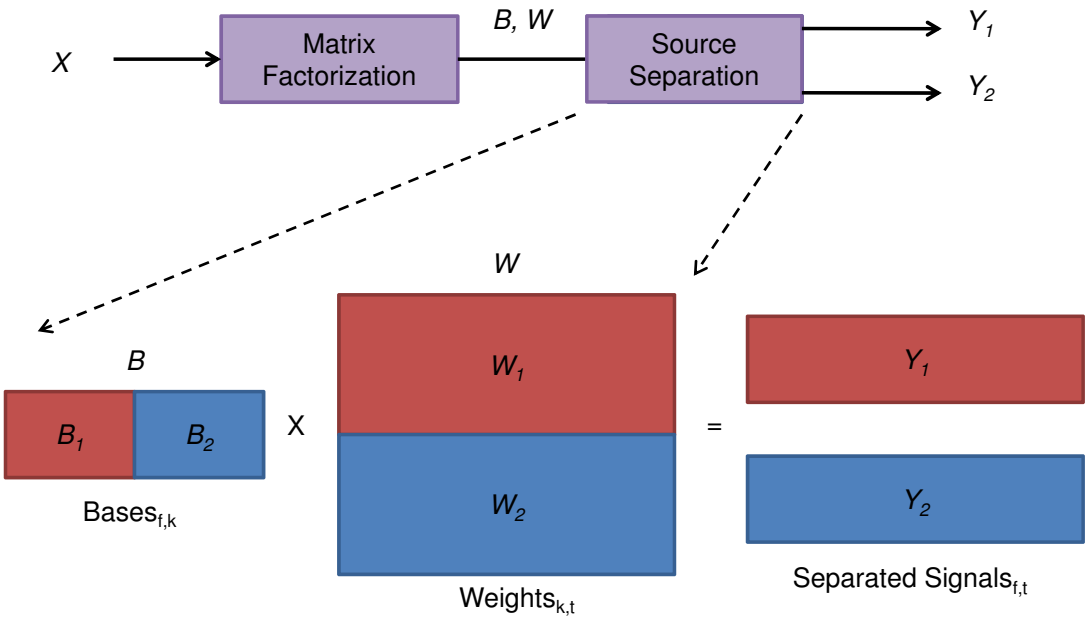

Figure 2.1: Block diagram for separating two sources via matrix factorization. The lower set of blocks detail the separation process.

vectors corresponding to each of the sources are learned from training data. We will discuss training methods further in Section 3.2.

In order to compute the best approximation, the notion of "best" must be defined. The three most common measures are the elemental sum of the squared error, resulting in the squared Frobenius norm,

$$\||X| - BW\|_F = \sum_{f,t} \left(|X_{f,t}| - (BW)_{f,t}\right)^2 \tag{2.2}$$

the Kullback-Leibler (KL) divergence [34],

$$(|X|\,||BW)_{KL} = \sum_{f,t}\left(|X_{f,t}|\log\frac{|X_{f,t}|}{(BW)_{f,t}} - |X_{f,t}| + (BW)_{f,t}\right) \tag{2.3}$$

and the Itakura-Saito (IS) divergence [11],

$$(|X|\,||BW)_{IS} = \sum_{f,t}\left(\frac{|X_{f,t}|}{(BW)_{f,t}} - \log\frac{|X_{f,t}|}{(BW)_{f,t}} - 1\right) \tag{2.4}$$

These cost functions are all special cases of the beta-divergence,

$$D_\beta(c||d) = \begin{cases} \frac{1}{\beta(\beta-1)}\left(c^\beta + (\beta-1)d^\beta - \beta c d^{\beta-1}\right) & \beta \in \{\mathbb{R} - \{0,1\}\} \\ c\log\frac{c}{d} - c + d & \beta = 1 \\ \frac{c}{d} - \log\frac{c}{d} - 1 & \beta = 0 \end{cases} \tag{2.5}$$

β's of 2, 1, and 0 correspond to the squared Frobenius norm, KL divergence, and IS divergence, respectively [7, 12]. Although these three values are the most popular values of β used in NMF, other values can be used as well. In Section 3.1, we will explore how different values of β affect two-talker separation. The β-divergence is a sub-class of the Bregman divergence [12, 20]. The square term in the Frobenius norm places much higher importance on fitting the larger elements well versus the smaller elements, while the logarithm and fraction terms in the KL and IS tend to place a more proportionally even importance of fit. The Frobenius norm version of NMF is closely related to its complex extension, which will be discussed in Section 2.2, and KL divergence has very similar update rules to the basic version of probabilistic latent component analysis, which will be discussed in Section 2.3. Although the IS divergence is an important contribution to the field, it is not as closely related to either complex matrix factorization or probabilistic latent component analysis, which are central topics in this dissertation, so the Frobenius and KL versions will be discussed in more detail than the IS.

Closed-form solutions of NMF finding the global minimum of the cost functions have not been found. Instead, an iterative approach is used. The base and weight

matrices are first seeded with initial values. Next, the base matrix is updated to minimize the cost function with the weight matrix held constant, and then the weight matrix is updated with the base matrix held constant. These two update steps are repeated until the cost function reaches a sufficiently stationary point. Although the algorithm has no guarantee of converging to the global minimum, the local minima located at the algorithm's stationary points are typically satisfactory for the given task. The update equations for the Frobenius norm are

$$B_{f,k} = B_{f,k}\frac{(XW^T)_{f,k}}{(BWW^T)_{f,k}} \tag{2.6}$$

$$W_{k,t} = W_{k,t}\frac{(B^TX)_{k,t}}{(B^TBW)_{k,t}} \tag{2.7}$$

and the update equations for the KL divergence are

$$B_{f,k} = B_{f,k}\frac{\sum_{\tau_1} W_{k,\tau_1}X_{f,\tau_1}/(BW)_{f,\tau_1}}{\sum_{\tau_2} W_{k,\tau_2}} \tag{2.8}$$

$$W_{k,t} = W_{k,t}\frac{\sum_{\omega_1} B_{\omega_1,k}X_{\omega_1,t}/(BW)_{\omega_1,t}}{\sum_{\omega_2} B_{\omega_2,k}} \tag{2.9}$$

These update equations are derived from auxiliary functions of the original cost functions and are guaranteed to decrease with every iteration. An auxiliary function is used when directly minimizing a primary function is not possible. An auxiliary function of a primary function is defined as any function that has a higher value than the original function at every point in its domain, except for at the points where the auxiliary parameter values match the primary parameter values. At these points, the primary and auxiliary function values are equal. Therefore, the auxiliary function defines an upper bound on the primary function. So minimizing the auxiliary function minimizes the upper bound on the primary function, which guarantees nonincreasing updates for the primary cost function. More information on auxiliary functions and their use in matrix factorization can be found in Févotte and Idier [12] and Yang and Oja [70]. Auxiliary functions typically have additional variables to create a function with such characteristics. Figure 2.2 shows the relationship between the auxiliary

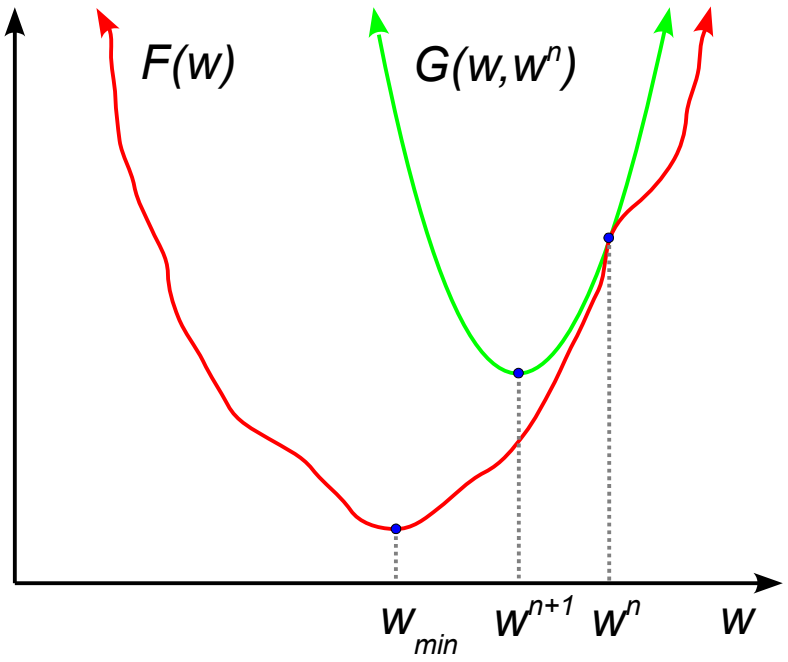

Figure 2.2: Illustration of the relationship between an auxiliary function $G(w, w^n)$ and its primary function $F(w)$.

and primary functions and how when the auxiliary variable a^n is updated to a^{n+1}, the costs of both functions decrease.

The auxiliary function corresponding to the Frobenius norm is

$$G(w, w^n) = F(w^n) + (w - w^n)^T \Delta F(w^n) + \frac{1}{2}(w - w^n)^T D(w^n)(w - w^n) \qquad (2.10)$$

where D is the diagonal matrix

$$D_{a,b}(w^n) = \frac{\delta_{a,b}\left[(B^T B w^n)_a\right]}{w_a} \qquad (2.11)$$

and w is an f-length vector that is a single column in $W \in \mathbb{W}^{\mathbb{K},\mathbb{T}}$. The auxiliary

function corresponding to the KL divergence is

$$G(w, w^n) = \sum_f (x_f \log x_f - x_f) + \sum_{f,t} B_{f,k} w_k - \sum_{f,k} x_f \frac{B_{f,k} w_k}{\sum_m B_{f,m} w_m^t} (\log B_{f,k} w_k - \log \frac{B_{f,k} w_k^t}{\sum_m B_{f,m} w_m^t}) \quad (2.12)$$

For more information on these auxiliary functions, please refer to Lee and Seung [34]. Although it is important to understand the auxiliary function derivations in order to fully understand NMF, the primary reason they are discussed in such depth is to demonstrate the first, and perhaps the greatest disadvantage, of NMF. Finding, proving, and minimizing auxiliary functions for NMF algorithms can be quite challenging, but is often required for developing new extensions. Some of the most popular algorithm extensions include adding the option for convolutive bases [42, 58], shift-invariance [47], and adding an L_1 term as a sparsity constraint to the Frobenius norm [42, 10].

If the goal of NMF is to produce an enhanced time-domain audio signal, instead of synthesizing the separated signals directly from the weights and bases, the basis vectors and weights are used to filter the original signal in the STFT domain,

$$Y_{s,f,t} = \frac{\sum_{k_s} B_{f,k_s} W_{k_s,t}}{\sum_k B_{f,k} W_{k,t}} X_{f,t} \quad (2.13)$$

where k_s denotes $k_s \in K_s$, the set of basis vectors corresponding to source s. This method is typically referred to as the "Wiener method", due to its similarity to Wiener filtering [69]. This method forces the sum of the separated signals to exactly equal the mixture instead of synthesizing directly from the weights and bases. This technique is popular because it is a lossless synthesis technique and typically produces better-sounding audio signals with fewer unnatural-sounding artifacts than the basic method,

$$Y_{s,f,t} = \left(\sum_{k_s} B_{f,k_s} W_{k_s,t} \right) e^{j\phi_{f,t}}, \text{ where } e^{j\phi_{f,t}} = \frac{X_{f,t}}{|X_{f,t}|} \quad (2.14)$$

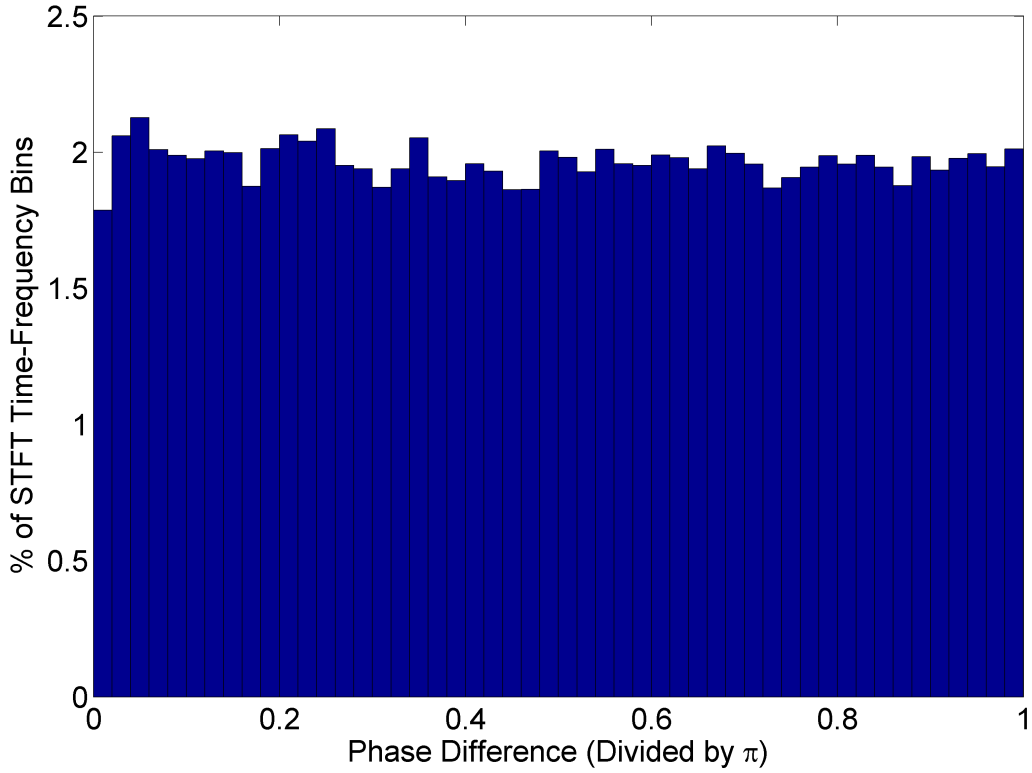

Figure 2.3: Normalized 50-bin histogram of phase angle differences between the STFT's of two speech signals of the same length.

There are several other matrix factorization algorithms, and while these other algorithms will not be explored in detail, some of the most popular methods will be mentioned and compared with NMF. In comparing NMF and principle component analysis (PCA) [23] and the closely-related singular value decomposition (SVD) [9], two key differences are found. The first difference is that, for PCA, the values of the elements of the data and of the two calculated matrices have no restrictions, while with NMF, all three matrices are constrained to be nonnegative and real. Secondly, the columns of the base matrix in PCA are linearly independent, while this is not necessary in NMF.

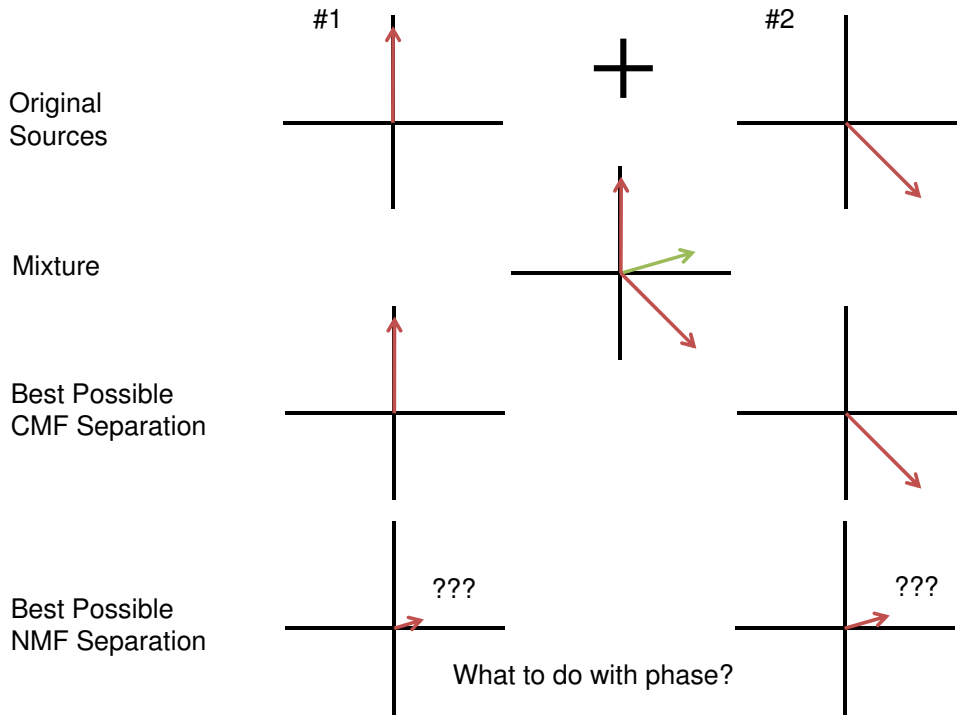

Figure 2.4: Example of how CMF and NMF differ in estimating individual sources in a mixture, shown for a single time-frequency point. Note how CMF can choose the correct phase for the separated signals, while NMF is constrained to use the mixture phase for the separated signals, resulting in both incorrect magnitude and phase estimations.

2.2 Complex Matrix Factorization

When analyzing nonnegative data with components that are solely additive, NMF may be an appropriate method for analysis. However, since NMF requires nonnegative, real data, changes are necessary when analyzing complex-valued data. What typically happens in such cases is that the phase is ignored and NMF is simply performed on the magnitudes of the data. This simplification is valid when all the

components of the data are additive. However, in the case of single-channel audio source separation of overlapping signals, this leads to the assumption that for each time-frequency point, all nonzero sources share the same phase, which is false. To illustrate the fact that arbitrary overlapping signals do not share a common phase, we calculated the STFT's of two equal length speech clips from different talkers and compared their phase (see Figure 2.3). This figure illustrates that the phases of two arbitrary signals are, in fact, not equal, but also uncorrelated. The following equations show how the sum of magnitudes does not equal the magnitude of the sums when the source phases are not equal:

$$X_{f,t} = C_{f,t} + D_{f,t}, \text{ where } |C_{f,t}| > 0, |D_{f,t}| > 0 \tag{2.15}$$

$$|X_{f,t}|e^{j\phi} = |C_{f,t}|e^{j\phi_1} + |D_{f,t}|e^{j\phi_2} \tag{2.16}$$

$$|X_{f,t}| = |C_{f,t}|e^{j(\phi_1-\phi)} + |D_{f,t}|e^{j(\phi_2-\phi)} \tag{2.17}$$

$$|X_{f,t}| \begin{cases} = |C_{f,t}| + |D_{f,t}|, \text{ when } \phi_1 = \phi_2 = \phi_3 \\ \neq |C_{f,t}| + |D_{f,t}|, \text{ otherwise} \end{cases} \tag{2.18}$$

where $C_{f,t}$ and $D_{f,t}$ are the two sources in mixture $X_{f,t}$ at time-frequency point (f, t). Another related problem is in recovering phase for the separated signals. Since magnitude-only spectrum matrices are created in the NMF separation step, no phase information is present for synthesizing a time-domain signal. Usually the original mixed phase is used [58], but doing so often causes audible artifacts in the reconstructed signal. In order to solve the aforementioned problems when using NMF for source separation, NMF has been extended to approximate the complex signal by including phase estimates. The cost function for CMF is the sum of the squared Frobenius norm of the estimation error and an optional sparsity constraint on the

weight matrix W that is controlled by λ and ρ,

$$f(\theta) = \sum_{f,t} |X_{f,t} - Y_{f,t}|^2 + 2\lambda \sum_{k,t} |W_{k,t}|^\rho, \text{ where} \tag{2.19}$$

$$Y_{f,t} = \sum_{s} Y_{s,f,t} \text{ and} \tag{2.20}$$

$$Y_{s,f,t} = \sum_{k_s \in K_s} B_{k_s} W_{k_s,t} e^{j\phi_{k_s,f,t}} \tag{2.21}$$

Although ρ's values can range between $0 < \rho < 2$, a value of 1 is most common, which makes the sparsity constraint an L_1-norm. The three-dimensional phase term indicates that a unique phase is estimated for each frequency, time, and basis index. This method was originally called complex nonnegative matrix factorization by the authors in Kameoka *et al.* [24] because the bases and weights still only contain nonnegative, real elements so that the phase variable holds all of phase information. For the sake of brevity and hopefully to disambiguate further between the nonnegative and complex methods, this algorithm will be referred to as complex matrix factorization (CMF).

In order to find the optimal values $\theta = \{B, W, \phi\}$ to minimize the cost function in eq. 2.21, the following auxiliary function is used:

$$\begin{aligned} f^+(\theta, \bar{\theta}) = &\sum_{k,f,t} \frac{|\bar{X}_{k,f,t} - Y_{k,f,t}|^2}{\beta_{k,f,t}} \\ &+\lambda \sum_{k,t} (\rho \left|\bar{W}_{k,t}\right|^{\rho-2} W_{k,t}^2 + 2\left|\bar{W_{k,t}}\right|^\rho - \rho \left|\bar{W}_{k,t}\right|^\rho) \end{aligned} \tag{2.22}$$

where

$$\sum_{k} \beta_{k,f,t} = 1, \beta_{k,f,t} > 0, \text{ for all elements in } \beta \tag{2.23}$$

$$\sum_{k} \bar{X}_{k,f,t} = Y_{f,t} \tag{2.24}$$

$$\bar{W} \in \mathbb{R}^{K \times T} \tag{2.25}$$

The auxiliary variables introduced are $\bar{\theta} = \{\bar{X}, \bar{W}\}$.

To minimize the CMF cost function, an iterative majorization-minimization (MM) algorithm [12] is used. First, for the majorization step, the auxiliary function is updated,

$$\bar{X}_{k,f,t} = Y_{k,f,t} + \beta_{k,f,t}(X_{f,t} - Y_{f,t}) \tag{2.26}$$

$$\bar{W}_{k,t} = W_{k,t} \tag{2.27}$$

Next, in the minimization step, the primary model parameters that minimize the auxiliary function are updated,

$$\phi_{k,f,t} = phase\left(\frac{\bar{X}_{k,f,t}}{\left|\bar{X}_{k,f,t}\right|}\right) \tag{2.28}$$

$$B_{f,k}^{new} = \frac{\sum_t \frac{W_{k,t}|\bar{X}_{k,f,t}|}{\beta_{k,f,t}}}{\sum_t \frac{W_{k,t}^2}{\beta_{k,f,t}}} \tag{2.29}$$

$$W_{k,t}^{new} = \frac{\sum_f \frac{B_{f,k}|\bar{X}_{k,f,t}|}{\beta_{k,f,t}}}{\sum_f \frac{B_{f,k}^2}{\beta_{k,f,t}} + \lambda\rho(\bar{W}_{k_s,t})^{\rho-2}} \tag{2.30}$$

Finally, β can be any value that satisfies the conditions in eq. 2.23, but its value can affect the number of updates required to reach a satisfactory minimum value. In order to minimize the cost function the most for each iteration, the β is updated via

$$\beta_{s,f,t} = \frac{\sum_{k_s} B_{f,k_s} W_{k_s,t}}{\sum_k B_{f,k} W_{k,t}} \tag{2.31}$$

A summary of the algorithm's steps are below:

1. Initialize B with random values that vary around 1
2. Initialize W with random values that vary around 1
3. Initialize ϕ with phase of observed matrix X
4. Update β using eq. 2.31

5. Update $\bar{X}$ using eq. 2.26

6. Update $\bar{W}$ using eq. 2.27

7. Update ϕ using eq. 2.28

8. Update B using eq. 2.29

9. Update W using eq. 2.30

10. Repeat steps 4-9 until satisfied

The separate sources are synthesized by the following method:

$$Y_{s,f,t} = \sum_{k_s} B_{f,k_s} W_{k_s,t} e^{j\phi_{k,f,t}} \tag{2.32}$$

CMF is the first NMF-related algorithm to estimate the phase and thus have the potential of exact source separation and reconstruction (see Figure 2.4) and has shown promising initial results which we will discuss in Section 3.3. However, also in that section we will point out some theoretical and practical disadvantages of CMF that currently limit its separation performance.

2.3 Probabilistic Latent Component Analysis

An alternate view of matrix decomposition is to use a probabilistic approach. The observation, which in the case of audio is typically the STFT magnitude matrix, is modeled as the two-variable joint probability distribution $P(f, t)$. The process then is to discover the hidden, or latent, variables that could have been used to generate the observed data. This general model is known as latent variable decomposition (LVD). The two most common methods of LVD applied to audio are symmetric and asymmetric probabilistic latent component analysis (PLCA) [22, 60, 54, 61]. PLCA

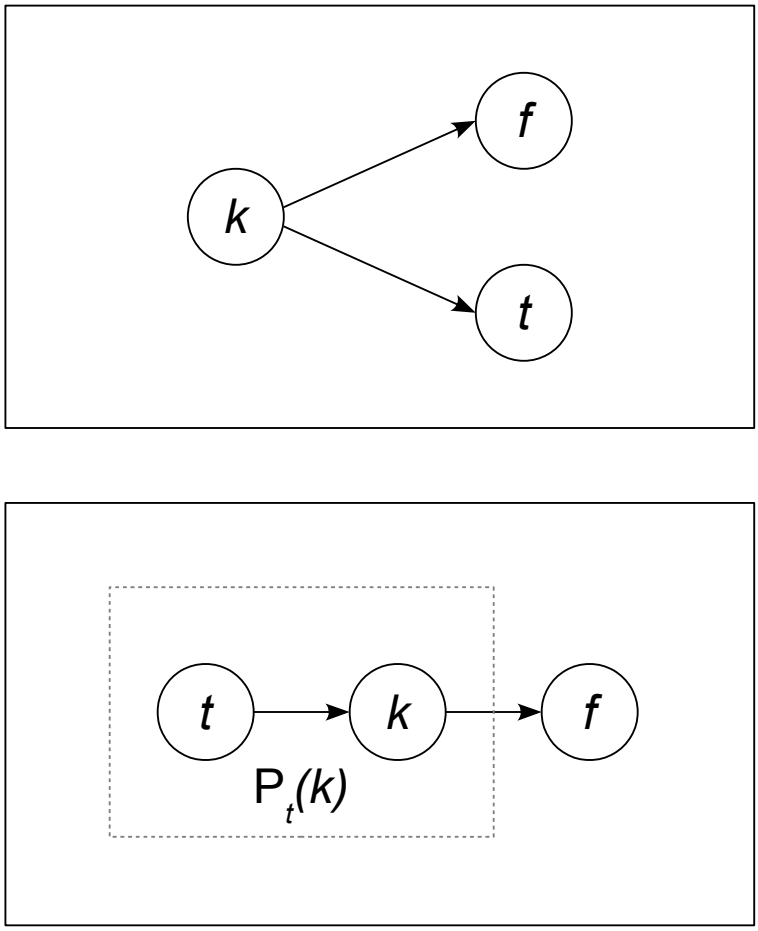

Figure 2.5: Generative models for symmetric (top) and asymmetric PLCA (bottom).

is sometimes referred to in the literature as probabilistic latent semantic analysis (PLSA).

In both types of PLCA, three random variables are used, the observed variables f and t and the latent variable k. Both use models to find the most likely underlying, unknown probability distributions that could have been used to generate the observed distribution $P(f,t)$ using the expectation-maximization method (EM), but they differ in their model specifics. The symmetric PLCA generative model (see Figure 2.5) can be explained by the following event sequence. For each event, first a state k is chosen

from set K with the multinomial distribution probability $P(k)$. Next, two additional values f and t are chosen that are both dependent on the choice of k but independent of each other using the multinomial distributions $P(f|k)$ and $P(t|k)$. Since these distributions are independent, they could also be written as $P(f,t|k) = P(t|k)P(f|k)$. Therefore, the joint probability of the observation can be written as

$$P(f,t) = \sum_{k \in K} P(k)P(f|k)P(t|k) \tag{2.33}$$

Symmetric PLCA is given that name because the probabilities of the two observed variables are conditioned on the same latent component k, thus the symmetry. To explain these three probabilities in a more intuitive manner, $P(k)$ can be considered the probability that a given base is chosen in an observed signal, $P(f|k)$ is the probability distribution of spectra within a particular basis vector, and $P(t|k)$ is the probability distribution of time within a basis vector. While $P(f|k)$ is analogous to the base matrix B in NMF, $P(k)$ and $P(t|k)$ are not represented similarly in NMF. Although symmetric PLCA is very promising and has already been used in audio applications, we will be focusing on asymmetric PLCA because it is the starting point of our research in complex-valued PLCA.

In contrast with PLCA's symmetric generative model, probabilistic latent component analysis uses an asymmetric generative model (see Figure 2.5). PLCA's generative model can thus be explained by the following event sequence. First, pick a time t from the multinomial probability distribution $P(t)$. Next, pick a base k from the multinomial distribution conditioned on t, $P(k|t)$. Finally, pick a frequency f from the multinomial distribution conditioned on k, $P(f|k)$. Therefore, the joint probability of the observation can be written as

$$P(f,t) = P(t) \sum_{k \in K} P(f|k)P(k|t) \tag{2.34}$$

PLCA's generative model is called asymmetric because the base and frequency random variables are dependent on different random variables, unlike the symmetric

PLCA where the observed variables are conditioned on the same latent variable. To explain these three probabilities in a more intuitive manner, $P(t)$ corresponds to the sum of spectral magnitudes at time t as a percentage of the sum of the entire time-frequency matrix, $P(k|t)$ corresponds to the NMF weight matrix W, and $P(f|k)$ corresponds again to the NMF base matrix B. Another way to think about the time variable is that each time t corresponds to a single experiment, during which you draw $P(k|t)$ and $P(f|k)$, and probability $P(t)$ can then be viewed as a weighting term. Also, $P(t)P(k|t)$ can be re-written as $P_t(k)$ and can be viewed as the overall probability of getting state k in the t-th experiment. Although both PLCA models are useful latent variable decompositions for audio processing, as mentioned previously, many more details will be presented about asymmetric PLCA because it is the starting point of our work in extending PLCA for complex numbers and phase estimation. The reason why we have chosen asymmetric PLCA as the starting point instead of symmetric PLCA is because the probabilities in the asymmetric model are, in our opinion, more intuitive for representing STFT's, more closely related with the matrix factorization models, and a more natural fit for extending the model to incorporate temporal and other useful information [36, 38]. In order to determine $P(t)$, $P(k|t)$, and $P(f|k)$ from the normalized observation $X^n_{f,t} = \frac{|X_{f,t}|}{\sum\limits_{f,t}|X_{f,t}|}$, maximum likelihood estimation (MLE) via the EM algorithm is used [54]. The likelihood of the model is

$$\mathcal{P} = \sum_t \sum_f X^n_{f,t} \log P_t(k) \tag{2.35}$$

In order to keep this section more concise, the EM steps for $P_t(k)$ will be explained first in greater detail, and then the resulting EM steps for $P(f|k)$ will be given since the derivations for both are quite similar. The expectation step for $P_t(k)$ is simply computing the posterior probability of the distribution given the observation,

$$P_t(k|f) = \frac{P_t(k)P(f|k)}{\sum_k P_t(k)P(f|k)} \tag{2.36}$$

The maximization step is

$$\begin{aligned}
\mathcal{L} &= E_{\bar{k}|\bar{f};\Lambda} \log P(\bar{f}, \bar{k}) \\
&= E_{\bar{k}|\bar{f};\Lambda} \log \prod_{j,t} P_t(f_j, k_j) \\
&= E_{\bar{k}|\bar{f};\Lambda} \sum_{j,t} \log P_t(f_j, k_j) \\
&= \sum_{j,t} E_{\bar{k}|\bar{f};\Lambda} \log P_t(f_j, k_j) \\
&= \sum_{j,t} E_{\bar{k}|\bar{f};\Lambda} \log P_t(k_j) + \sum_{j,t} E_{\bar{k}|\bar{f};\Lambda} \log P_t(f_j|k_j) \\
&= \sum_{j,t} \sum_{k} P(k|f_j) \log P_t(k_j) + \sum_{j,t} \sum_{k} P(k|f_j) \log P_t(f_j|k_j) \\
&= \sum_{t} \sum_{f} \gamma X^n_{f,t} \sum_{k} P(k|f_j) \log P_t(k_j) \\
&\quad + \sum_{t} \sum_{f} \gamma X^n_{f,t} \sum_{k} P(k|f_j) \log P_t(f_j|k_j)
\end{aligned} \tag{2.37}$$

where Λ is the set of variables to be estimated,

$$\Lambda = \{P(f|k), P_t(k)\} \tag{2.38}$$

The last step in eq. 2.37 changes the summation over draws j to a summation over features f by counting how many times f was observed, and the γ is used as a scaling factor in order for the observed data to be integers, since the data is viewed as a histogram. Since $P_t(k)$ and $P(f|k)$, the two values to be solved for, are proper probabilities, a Lagrangian can be introduced to enforce these constraints,

$$\mathcal{Q} = L + \sum_{t} \tau_t \left(1 - \sum_{k} P_t(k)\right) + \sum_{k} \zeta_k \left(1 - \sum_{f} P(f|k)\right) \tag{2.39}$$

Next, if the partial derivatives of eq. 2.39 with respect to $P_t(k)$ and $P(f|k)$ are calculated and their derivatives are set to zero, the following two relations are found

$$\sum_{f} \gamma X^n_{f,t} P_t(k|f) + \tau_t P_t(k) = 0 \tag{2.40}$$

$$\sum_t \gamma X^n_{f,t} P_t(k|f) + \zeta_k P(f|k) = 0 \tag{2.41}$$

Finally, the two above equations are used to solve for the two equations for $P_t(k)$ and $P(f|k)$,

$$P(f|k) = \frac{\sum_t X^n_{f,t} P_t(k|f)}{\sum_f \sum_t X^n_{f,t} P_t(k|f)} \tag{2.42}$$

$$P_t(k) = \frac{\sum_f X^n_{f,t} P_t(k|f)}{\sum_k \sum_f X^n_{f,t} P_t(k|f)} \tag{2.43}$$

In conclusion, the following process is used for modeling the data via PLCA:

1. Initialize $P_t(k)$ and $P(f|k)$

2. Calculate expectation $P(k|f)$ using eq. 2.36

3. Update $P_t(k)$ and $P(f|k)$ using eqs. 2.42 and 2.43

4. Repeat steps 2-3 until values converge sufficiently (guaranteed by the EM algorithm)

One very interesting finding is that although the methods are very different, the update equations for asymmetric PLCA and the KL version of NMF are equivalent [17, 31, 21]. Although the resulting update equations are equivalent, we would argue that since PLCA was developed in a probabilistic framework with generative models, it is easier to extend the algorithm than its NMF counterpart. For example, entropic priors [54] and hidden Markov models with state-dependent dictionaries [36] have been added to the PLCA framework. In Chapter 5, we will show how complex data can be transformed to lie on a nonnegative simplex. In this form, the data can be modeled and separated by either PLCA or NMF .

Chapter 3

MATRIX FACTORIZATION PERFORMANCE COMPARISONS

In the previous sections, we presented a theory-based argument about the superposition problem introduced when using magnitude-only representations for single-channel source separation, and how that problem is solved via complex analysis and synthesis. In this section, we will present three studies on how parameter choice and training methods affect performance in nonnegative and complex matrix factorization-based speech separation. The first is a study of how nonnegative matrix factorization (NMF) parameters affect single-channel source separation performance [27]. The second and third compare automatic speech recognition (ASR) performance on two-talker mixtures separated with NMF and complex matrix factorization (CMF) [26, 25]. In addition to comparing NMF and CMF, these studies presented and compared two different training methods for matrix factorization.

3.1 Learning How NMF Parameters Affect Performance on Single-Channel Source Separation

NMF has become a popular area of research and has been applied in many diverse fields, including audio. Within audio, it has been applied in a variety of tasks, including single-channel source separation [58], interpolation of missing audio data [62], bandwidth expansion [1], polyphonic transcription [59], and multi-source and noise-robust speech recognition and dynamic time warping [50, 28]. There has been a significant amount of research on these topics, with many new algorithms and parameters being proposed. While it is exciting to see so much work in this field, it has become challenging to choose an NMF algorithm for a particular application because

there are many papers proposing different cost functions and parameters that purport to be the best. At the moment, it can take a significant amount of time to search through the literature and run experiments to find the best parameters for the chosen application. The goal of this work is to help address this challenge. The paper that this section is based on focuses on two popular applications, single-channel separation of speech and interpolation of missing music data [27]. For each, we ran experiments with many different NMF models and parameters. Since this dissertation focuses on single-channel source separation, only this part of the paper will be presented and discussed. In the data analysis, we will discuss how parameters affect performance, provide explanations and hypotheses about the performance trends we observed, as well as present our findings for parameters that perform optimally overall, in a fashion similar to the parameter analysis for NMF-based musical source separation in Fitzgerald *et al.* [14] and NMF-based multi-pitch estimation in Vincent *et al.* [65]. Our goals are to show what models and parameter values work well, as well as help develop an intuition for how parameters affect performance. And although we will focus on just one of the two applications from the conference paper, we hope that this knowledge of how parameters affect performance will lead to an ability to understand, and even predict, how parameter choice will affect the performance of different applications.

3.1.1 Background

As discussed in 2.1, the NMF algorithm decomposes an observed matrix into a product of a basis matrix and weight matrix. When applied to audio, the observed matrix is most often the magnitude or power spectrogram, but can also be other nonnegative time-frequency representations [67]. When applied to a time-frequency representation, the columns of B correspond to spectral basis vectors and the columns of W indicate the weighting of those basis vectors within a particular time window. In audio applications, the nonnegativity constraint on the matrices can often result in meaningful basis vectors, which can be helpful for analysis or processing. For ex-

ample, NMF analysis of a piano excerpt can find basis vectors that each correspond to the spectra of individual notes [59], and NMF analysis of speech can result in phoneme-like bases [58].

Although the basic NMF algorithm is fairly simple, there are many variations that have been proposed in the research literature, including differences in cost function, magnitude exponent, number of bases, and window length. In this section, we will explore how these affect performance, as well as disclose the parameters we found to work best overall. We will now discuss these parameters and provide examples and intuition of how they affect performance. NMF uses an iterative algorithm to updates the basis and / or weight matrices to minimize the given cost function. Thus, different cost functions can produce significantly different results. The most popular cost functions are the squared Frobenius norm [33], the Kullback-Leibler (KL) divergence [33], and the Itakura-Saito (IS) divergence [11]. The squared Frobenius norm, as it minimizes the squared error of the estimate, is sometimes criticized for audio applications because it places too high an importance on modeling higher-energy components, often at the expense of lower-energy components, which are often still important for audio quality and auditory perception [11]. In contrast, the KL divergence places a more equal emphasis on components with higher and lower energy. And finally, the IS divergence has a completely scale-invariant cost function, meaning

$$d(V|\hat{V})_{IS} = d(\alpha V|\alpha\hat{V})_{IS}, \text{ for } \alpha > 0 \tag{3.1}$$

where $\hat{V}$ is the estimation of observed data V. These cost functions are all special cases of the beta-divergence, where β's of $2, 1$, and 0 correspond to the squared Frobenius norm, KL divergence, and IS divergence, respectively [12]. β values between and beyond these three values can be used as well, and will be explored and discussed in this work.

The second parameter we will be examining is the magnitude exponent of the

observation, which raises every element of the STFT magnitude $|X|$ to the power p,

$$V_{f,t} = |X_{f,t}|^p, \text{for } f = 1, ..., F, t = 1, ..., T \tag{3.2}$$

The most common values of p are 1 and 2, which correspond to the magnitude and power spectra of the signal. However, any value of $p > 0$ can be used for any of the cost functions.

Deciding on the number of basis vectors to use is essential for good performance. Choosing too small a number can result in the basis vectors not being able to approximate the data well, but too many can result in over-fitting the data. The final parameter we will be discussing is the window length. If a window is too short, then there may not be enough spectral differentiation between the sources, causing poor source separation and interpolation. But if the window is too long, then the spectra of the signal will be less stationary within a window, which will also result in poor performance.

3.1.2 Theory

The parts-based decomposition in the NMF algorithm lends itself well to source separation. If it is known that disjoint sets of spectral basis vectors correspond to different sources, then the enhanced signal for source s can be synthesized by multiplying together the basis vectors and weights corresponding to that source,

$$\hat{V}_{s,f,t} = \left(\sum_{k \in K_s} B_{f,k} W_{k,t} \right) \tag{3.3}$$

where K_s is the set of spectral basis vectors corresponding to source s. This estimate is used to calculate the time-frequency weights for filtering the original, complex-valued STFT of the mixed signal,

$$Y_{s,f,t} = \left(\frac{\sum_{k_s \in K_s} B_{f,k_s} W_{k_s,t}}{\sum_k B_{f,k} W_{k,t}} \right) X_{ft} = \left(\frac{\hat{V}_{s,f,t}}{\hat{V}_{f,t}} \right) X_{f,t} \tag{3.4}$$

This method ensures that the decomposition is lossless.

One challenge of source separation is how to acquire disjoint sets of basis vectors that represent the different sources well, and will be discussed thoroughly in 3.2. We will be using the most straightforward method for our speech separation experiments, the fully-supervised copy-to-train method [63, 25]. In this method, we will use the magnitude of the STFT of the training data (raised to the magnitude exponent p), with the training data being other utterances from that same speaker found in the mixture. The idea is that the speech from the spectra of the training data will be similar enough to that of the speech in the test data that it can be estimated well by the training data. The copy-to-train method is repeated for all the speakers in the mixture. We chose the fully-supervised method because we had training data for all the speakers. One known method to increase source separation performance is to eliminate all frames with energy below a threshold, because nearly silent frames do not represent the important speech well [63]. Because of this, we set the threshold at 40 dB below the highest-energy frame of each speaker's training data. We chose copy-to-train over factor-to-train for a few reasons. First of all, if the training data is representative of the test data, the former method is guaranteed to result in meaningful basis vectors, while the other methods will not necessarily do so if a poor number of basis vectors is chosen for factorizing the training data. Both too many and too few basis vectors can result in less meaningful basis vectors and poor separation. Also, it was helpful to eliminate any unnecessary variables to focus on analyzing the other variables to make the results clearer. So in these experiments, we will see how cost function, magnitude exponent, and window length affect performance.

We will use the blind source separation evaluation (BSS Eval) toolkit for analyzing and comparing source separation results [66]. This method measures the source-to-interference ratio (SIR), source-to-artifact ratio (SAR) and source-to-distortion ratio (SDR). The SIR measures how much of the interfering sources are left in the separated signal. The SAR measures how much energy is in the signal that is not part of either

the target or interfering signals. The SDR combines the SIR and SAR into one measurement. The SIR results are computed by finding the difference, in dB, of the SIR of the enhanced signal from the SIR of the original mixed signal. The SDR is computed in the same fashion. The SAR is computed simply by the SAR of the enhanced signal. Since there are no artifacts in the original mixture, its SAR is $+\infty$ dB. Thus, if we used this value in the calculation, the resulting SAR difference would always be $-\infty$ dB, which is not helpful for comparisons. In other words, the baseline SIR and SDR are calculated from the original mixture. We will refer to these measurements as relative SIR, absolute SAR, and relative SDR.

3.1.3 Experiments and Results

We had the following goals with these single-channel speech separation experiments. First of all, we wanted to see how parameters (cost function, magnitude exponent, and window length) affected performance. We also wanted to test with a variety of target-to-masker ratios (TMR, where the target is the wanted signal and the masker is the unwanted signal) and speaker combinations to determine whether or not there was an optimal set of parameters for all scenarios, and hypothesize why. The test signals consist of two utterances from different speakers from the TIMIT database [16], sampled at 16 kHz. In order to maximize signal overlap in time, we truncated the length of the mixture to the shorter of the two utterances. We then mixed the utterances at -10, -5, 0, +5, and +10 dB TMR. There were eight speaker combinations of mixed gender, four of female only, and four of male only. For each experiment, we measured the BSS Eval measurements for both speakers in each experiment. Because of this, we had eight target/masker combinations each of F/F, M/M, F/M, and M/F. Each speaker in the TIMIT database has two sentences common to the other speakers and eight unique sentences. We used only the unique sentences. For a given speaker combination, we used seven of the eight sentences for training and the other for the test mixture. For each combination of speakers, we used two different combinations

of test and training signals. So for each set of parameters at each TMR, we ran 24 experiments, resulting in 48 BSS Eval measurements.

In our analysis of the results, we found that one set of parameters performed best overall for our speech separation task. The optimal parameters were a β of 0, a magnitude exponent of 1, and a window size of 512 points (32 ms). We have included some figures that illustrate how parameters affect performance.

Magnitude Power and β

In Figure 3.1, we have plotted the BSS Eval results for varying magnitude powers and the β's of the cost function, with a constant window size of 512 (32 ms) and TMR of 0 dB, averaged over all test sentences. We see that the relative SIR is highest when β equals 0 (Itakura-Saito divergence) and magnitude exponent equals 1.5 and 2. A β of 0 and a magnitude exponent of 2 correspond to the generative model of superimposed Gaussian components advocated in Févotte *et al.* [11]. In the SAR results, we see that as the magnitude exponent increases, the absolute SAR decreases. This is logical because if the magnitude exponent were 0, the separated signals would simply be identical signals that would be half the amplitude of the original signal. There would be no artifacts in that signal, but the SIR improvement would also be 0 dB. As the magnitude exponent increases, more filtering takes place, which can lead to more artifacts and thus a lower SAR. Within a magnitude power, β values of 0 (Itakura-Saito divergence) and 2 (squared Frobenius norm) have the highest absolute SAR, though their SAR values vary less within a given magnitude power than between different powers. The relative SDR, which incorporates both relative SIR and SAR, is maximized when β is 0 and the magnitude exponent is 1.

Window Size and TMR

In Figure 3.2, we see how TMR and window size affect performance. The results plotted are with the optimal parameters discussed in the previous paragraph, which

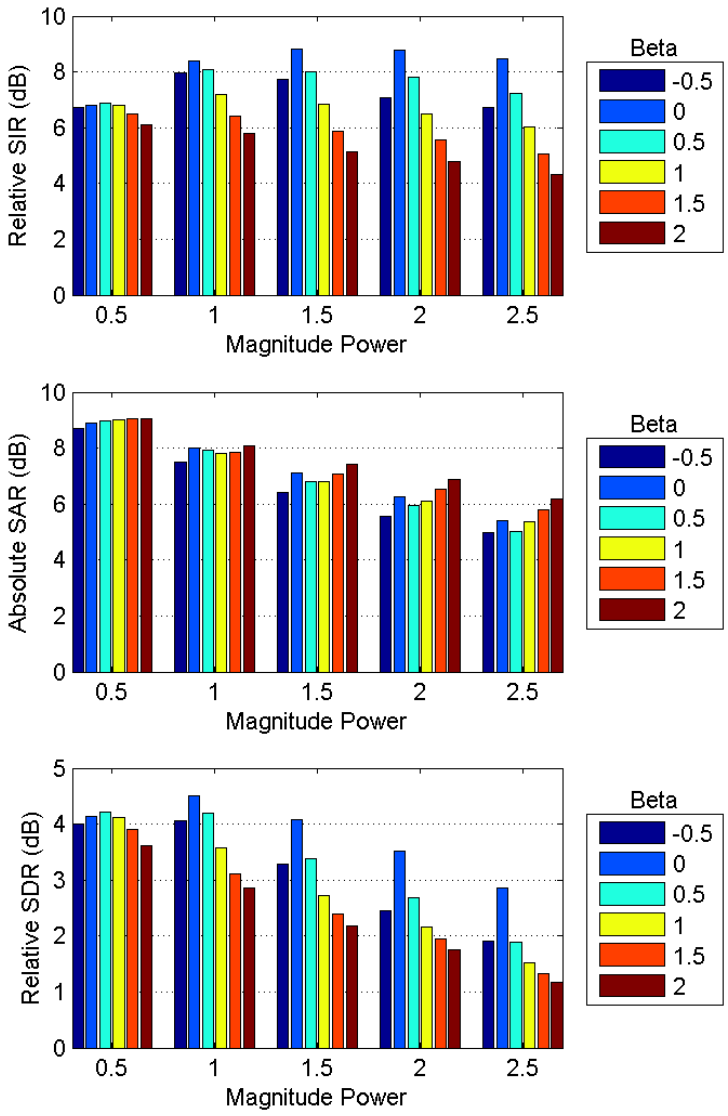

Figure 3.1: BSS Eval results illustrating how β and magnitude exponent affect performance (averaged over all experiments with 0 dB TMR and 512-point (32 ms) window). Note: taller bars are better.

are a β of 0 and a magnitude exponent of 1. We see that within each TMR, a window size of 512 performs the best for both the relative SIR and SDR. The absolute SAR is slightly higher with window sizes of 256 and 2048. What is more interesting in this figure is observing how the BSS Eval measurements are affected by the target

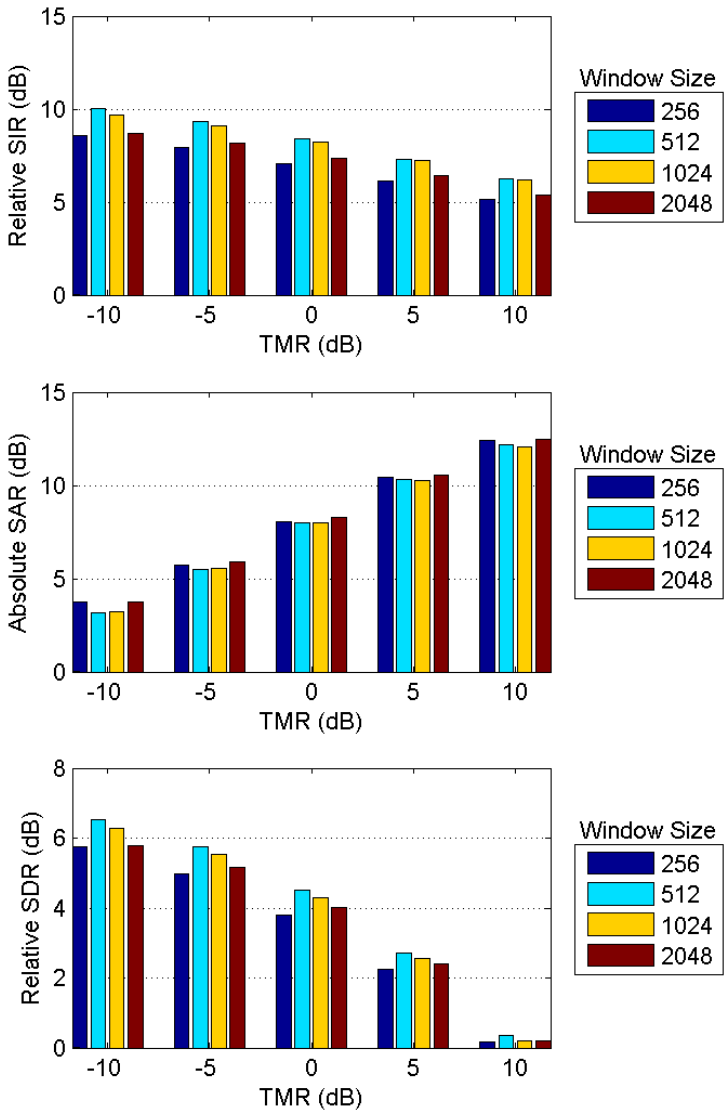

Figure 3.2: BSS Eval results illustrating how window size and TMR affect performance (averaged over all experiments with β of 0 and magnitude exponent of 1). With a 16 kHz sampling rate, windows of 512, 1024, 2048, and 4096 points correspond to lengths of 32, 64, 128, and 256 ms, respectively.

speaker's TMR. As the TMR increases, the resulting relative SIR decreases. This can be explained as follows: if the TMR is very low, it is easier to identify and attenuate the masker, but when the TMR is very high, is it more difficult to identify

and attenuate the masker. For example, if the TMR were 100 dB, it could be very difficult to find the noise, and attenuating the signal would most likely attenuate the desired signal, thus decreasing the relative SIR. The absolute SAR results can be explained as follows: since the SAR measures the ratio of the signal to the artifacts, if the signal level increases and artifacts stay the same level, then the SAR will improve. However, we see that artifacts do not stay constant over the TMR range, so that for every difference of 5 dB, the artifact level changes by about 2.5 dB. We thus see that an TMR increase of 5 dB results roughly in an increase of 2.5 dB absolute SAR. And finally, when analyzing the SDR results, we see that as TMR increases, relative SDR decreases. So at +10 dB, although we are able to still increase relative SIR, the artifacts cause just a small improvement in relative SDR. Although the relative SDR and SIR decrease as TMR increases, we fortunately see that the absolute SDR (which is the sum of the mixture TMR and the relative SDR) and absolute SIR increase as TMR increases.

Speaker Gender and Window Size

In Figure 3.3, we see how the speakers' genders and the window size affect performance. The results plotted are with the optimal parameters discussed in the previous paragraph, which are a β of 0 and a magnitude exponent of 1. We wanted to compare these two variables on the same figure because we had originally hypothesized that the optimal window size would depend on the spectral similarity and the pitch ranges of the speakers. We hypothesized that as spectral similarity increased (same gender), or the pitch ranges decreased (both male speakers), that a longer window would perform better. We found, however, that this was not true. In fact, the 512-point window gives the best overall performance for all three speaker gender combinations (mixed gender, both female, and both male). This is good news because the optimal value of this parameter is not dependent on the speakers' characteristics.

In comparing performance between different sets of speaker genders, we saw that

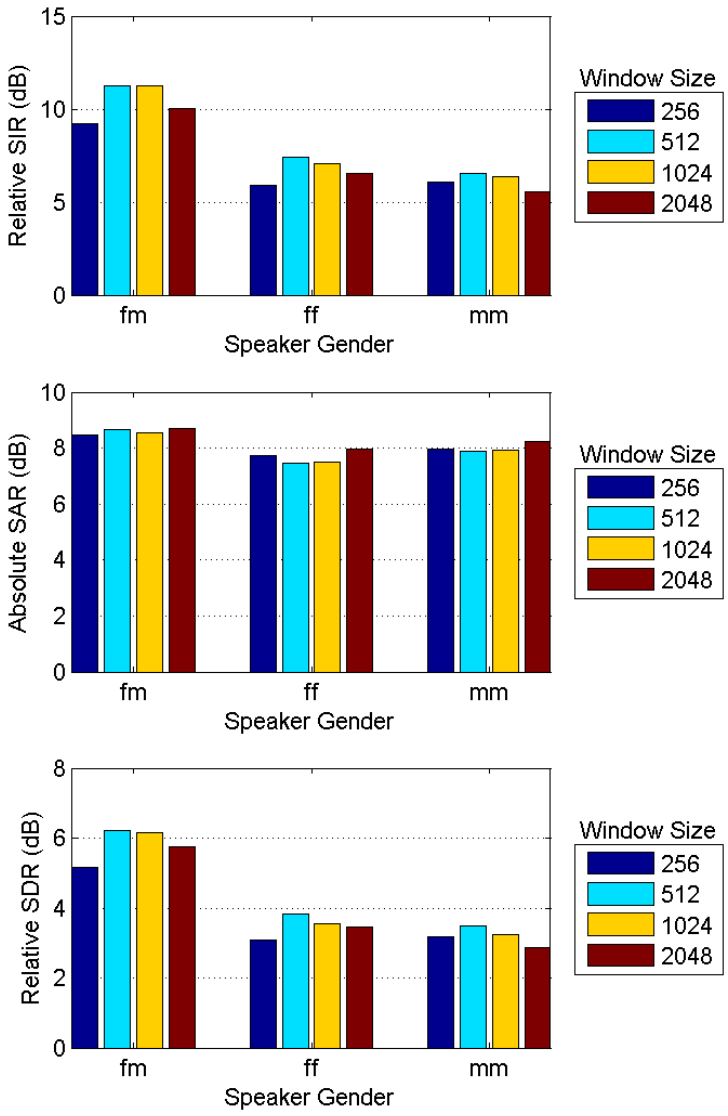

Figure 3.3: BSS Eval results illustrating how window size and speaker gender affect performance (averaged over all experiments with β of 0, magnitude exponent of 1, and TMR of 0).

the mixed-gender sentences typically had higher relative SIR, absolute SAR, and relative SDR scores than sentences with the same gender. This is consistent with our hypothesis that more spectrally different sources would result in better separation. We also see that the relative SIR and SDR are higher for the two-female mixtures than

the two-male mixtures, and the absolute SAR is just slightly lower. We think that the relative SIR and SDR improvements are higher for the female speakers because there will typically be less overlap in the spectra of the two females because their pitch, and thus spacing between harmonics, is greater. Spectral similarity and spectro-temporal overlap are thus two of the best indicators of NMF's source separation performance. NMF will perform better when similarity and overlap are lower and worse when similarity and overlap are higher.

Here is a summary of the single-channel two-talker speech separation results:

- The best overall parameters found for relative SDR were a β of 0, a magnitude exponent of 1, and a window size of 512 points (32 ms).
- The Itakura-Saito divergence ($\beta = 0$) with a magnitude exponent of 1.5 and 2 typically maximize suppression of interfering speakers.
- Increasing magnitude exponent typically increases artifacts.
- Increasing TMR typically increases absolute SIR, absolute SAR, and absolute SDR, and decreases relative SIR, and relative SDR.
- A 512-point (32 ms) window works best overall for all speaker gender combinations.
- Performance is negatively correlated with the sources' spectral similarity and spectro-temporal overlap, which give the following order of performance, from best to worst: mixed gender, both females, and both males.

3.1.4 Conclusion

In this work, we have explored how parameters and test conditions affect performance for NMF-based single-channel speech separation. Specifically, we analyzed

the β, magnitude exponent, and window size parameters and the TMR's and genders of the sources for speech separation. We hope that our goals of explaining how parameters affect performance will not only provide optimal parameter choices for these applications, but that the explanations of the data will also provide a deeper understanding and a better intuition of NMF parameters for other applications.

3.2 Comparing Matrix Factorization Training Methods

In this section, we will discuss matrix factorization for source separation and the significance of training in the process, summarize past training methods, and then present a new training method for matrix factorization [25, 63].

The key to successful separation lies in the quality of the bases. The two most important characteristics of bases used for source separation are having a set that accurately models the signal and having high independence between speaker-specific sub-matrices. The first characteristic, having an accurate model, means that a linear combination of bases must be able to closely approximate the mixed signal. If this does not happen, the resulting synthesized separated signals will not be accurate. The second characteristic, having a high degree of linear independence between speaker-specific sub-matrices, means that the bases from one source should be as different as possible from the bases of the other source. Bases may be linearly dependent from one another, but typical nonnegative and complex matrix factorization techniques do not enforce this condition. If a speaker's bases are significantly different than the others', then each source in the mix will only be able to be accurately represented with bases from the correct speaker, which will allow for maximum source separation. If bases are very similar between speakers, then the signal cannot be separated well. We will now present two training methods and discuss each of their advantages and disadvantages.

3.2.1 No-Train Method

In using matrix factorization for source separation, there have been two main methods for calculating the bases [24]. The first, which we will call the "no-train" method, calculates the base and weight matrices from the mixed signal without a training step. There are two challenges with this method. The first is that the resulting columns in the base matrix are in random order so that it is unknown which base columns correspond to which source. Because of this, basis elements have to be further analyzed to try to determine which source they represent.

The other problem with this method is that the number of basis vectors must be specified before factorization. The difficulty in choosing the number of vectors is that if the count is too small, the basis set consists of spectra that contain multiple speakers or phonemes. If the count is too high, however, the set of frequency vectors in the set may be too general and not speech-like. For example, if the base matrix is complete or overcomplete, one possible basis decomposition would be one containing the identity matrix, which obviously is not useful for multi-talker separation because the bases cannot be divided into speaker-specific sub-matrices. Thus, picking an appropriate number of basis vectors is usually done in an ad hoc manner by time-consuming trial-and-error, which highlights the need for a better solution.

3.2.2 Factorize-to-Train Method

The second previous method for calculating bases, which we will call the "factorize-to-train" method, solves the first problem mentioned above by first calculating the base matrix with clean, single-source training data [50]. If there are N sources in the mixture, then a base matrix is calculated from each clean source during the training. All these bases are then concatenated together in order to form the base matrix for the mixture. This base matrix is then used during factorization of the mixture signal in order to determine the weight matrix for the mixture. Once the weight matrix

is calculated, the separation step is possible because the basis indices are known for each source. The problem with this method is that the number of basis vectors must be specified before factorization in the training step. The difficulty in choosing the number of vectors is that if the count is too small, the basis set consists of spectra that contain multiple speakers or phonemes. If the count is too high, however, the set of frequency vectors in the set may be too general and not speech-like. For example, if the base matrix is overcomplete, one possible basis decomposition would be one containing the identity matrix, which obviously is not useful for multi-talker separation because the bases cannot be divided into speaker-specific sub-matrices. Adding the sparsity constraint can aid in finding meaningful bases, but again, finding the best-performing sparsity factor is non-trivial and further complicates the training process. Thus, picking an appropriate number of basis vectors is usually done in an ad hoc manner by time-consuming trial-and-error, which highlights the need for a better solution.

3.2.3 Copy-to-Train Method

A new method for choosing the base, which we will call the "copy-to-train" method, does not have any of the problems mentioned above [25, 63]. Instead of calculating the base matrices for each source by matrix factorization, the basis vectors are simply set to be the magnitude of each single-source training STFT. In other words, the bases are a "copy" (in STFT matrix form) of the original training data. The number of bases is then simply set by the training data length as well as the window and overlap size of the analysis STFT. This type of training is known as a "lazy learning" method because it does attempt to model the training data with a set of parameters as done by "eager training" methods like factorize-to-train.

This method also solves the problem of having meaningful bases because the phonetic elements in the training data are known to be represented perfectly by a single basis vector and can thus be synthesized by single nonzero weight corresponding to

that basis vector. As long as there is not a significant deviation in the phonetic content between the training and mixed data, the separated signal can also be synthesized by a highly sparse weight matrix which can result in better separation [63]. In conclusion, this approach has the following advantages over factorize-to-train: (1) separation becomes easy as the basis vector indices are known for each source, (2) the problem in choosing the number of bases is eliminated, (3) the basis vectors are known to correspond to the phonetic elements of the speaker and can thus be used to represent the source using a sparse set of weights, and (4) processing the bases is significantly faster since no matrix factorization is needed in training.

3.3 Comparing NMF and CMF in Overlapping Talker Automatic Speech Recognition

We actually conducted two similar experiments. The first was a short pilot study with a smaller dataset to see if CMF had potential [25], while the second was a more in-depth study [26]. In the pilot study, we compared the Frobenius norm with L_1 sparsity versions of nonnegative and matrix factorization and compared the factorize-to-train method with the copy-to-train method. To summarize our findings from that study, we found that automatic speech recognition performance with CMF significantly outperforms both its closest NMF variant as well as the unprocessed mixed signal on overlapping speech mixed at target-to-masker ratios from -6 dB to +6 dB. We also found that the copy-to-train method outperformed the factorize-to-train method for CMF in all cases, but not for NMF.

With these positive initial results, we designed a new experiment with a larger dataset to verify our findings. We used the Boston University Radio News corpus [44] which contains both male and female speakers. For each separation test, we first selected a target speaker utterance at random. The lengths of the utterances ranged between 10 and 66 seconds, with an average of 29 seconds. We next selected a masking clip from a different speaker at random and added it to the original clip

so that the target-to-masker ratio was 0 dB. For training data, we used 60 seconds of clean speech from different clips from the same speakers. We then performed matrix factorization on the mixed signals to obtain the separated target. On each clip, we used the factorize-to-train method for 50, 100, 200, 400, 800, and 1600 bases per speaker as well as the copy-to-train method. By using 60 seconds of training data at a 16 kHz sampling rate, a 1024-point window, and a 50% overlap, our copy-to-train method resulted in 1876 bases per speaker. We chose not to try factorize-to-train with more basis vectors for two reasons. First of all, the CMF algorithm is much more computationally expensive and would have taken an unreasonably long time to run. Additionally, running factorize-to-train to find a higher number of bases than columns of data in the training data would lead to a less sparse solution, hence degrading performance by finding less meaningful bases. We used the synthesis method defined in eq. 2.14 for NMF and the method defined in eq. 2.32 for CMF.

Our test data was composed of 150 clips for a total length of 72 minutes. For all experiments, we used a sparsity factor λ of 0.01, which we observed to work well with this data. We did not choose to include a thorough study on tuning the sparsity factor for two reasons. First of all, we wanted to keep our focus on the two main goals of the paper, comparing CMF with NMF and copy-to-train with factorize-to-train. Secondly, there are already similar studies on sparsity performance with NMF [63, 54, 42].

We used two automatic speech recognition systems, the Stanford Research Institute (SRI) system [64] used in our pilot study as well as the Sphinx-3 system [46]. It is important to note that the results between tests are difficult to compare, since the systems used different acoustic and language models. Also, no significant work was done to boost scores with either recognition system, since we are more concerned in the relative improvement between the unprocessed, NMF, and CMF clips instead of their absolute scores.

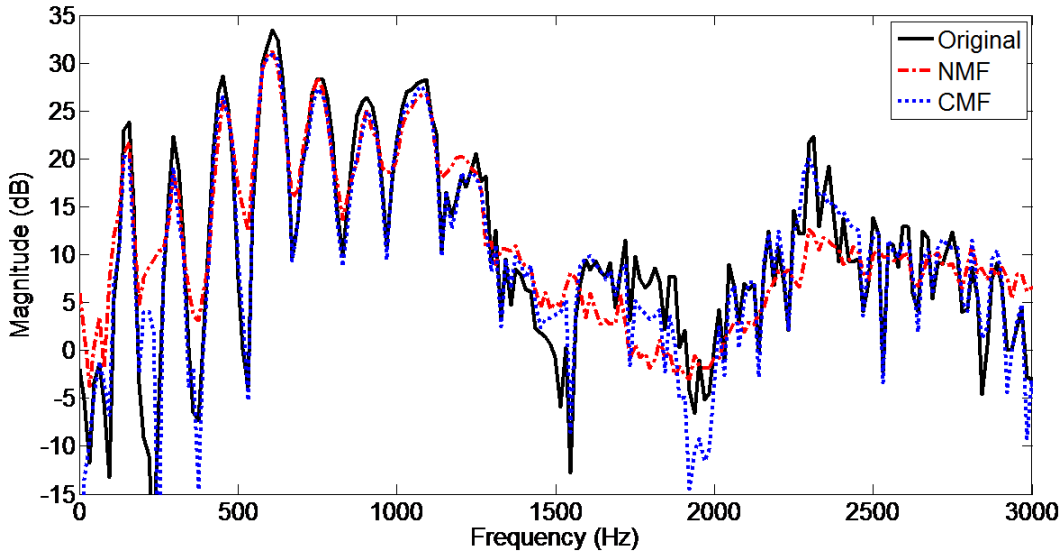

Figure 3.4: Magnitude spectrum of a single timeframe for the original clean signal, the separated via NMF, and separated via CMF (up to 3 kHz).

3.3.1 Complex vs. Nonnegative Matrix Factorization

Before discussing the recognition results, we will first discuss our informal observations of the NMF and CMF separated signals. The CMF separated signal can be described as the target signal sounding very similar to the target in the original mixed signal and the masker attenuated naturally, as if dropping its level with a mixing board. It does not suffer from any "musical noise" or other unnatural artifacts commonly found in separation and enhancement algorithms. In comparing the two processed signals, NMF has a similar target-to-masker ratio to the CMF for 2 kHz and below. The higher frequencies, however, capture much less of the target signal than CMF. Finally, the NMF has noticeably more artifacts including a white noise-like hiss throughout the spectrum.

These differences are illustrated quantitatively in Figure 3.4. In the lower frequencies, CMF and NMF capture the peaks with similar accuracy. The higher frequency peak at 2.3 kHz is better modeled by CMF. And throughout the spectrum, the val-

leys in the target signal are typically better captured by CMF. We hypothesize this additional inharmonic magnitude is caused by the lack of superposition when using NMF to separate complex-valued STFT signals. This inharmonic energy likely manifests itself in the hissy artifacts that we observed in the NMF signals. To summarize our observations, the NMF had more artifacts and poorer high-frequency modeling while the CMF sounded much more natural in its attenuation. We will now discuss the automatic speech recognition results. To compare the unprocessed, NMF, and CMF for each system, we first found the basis vector learning method resulting in the best-performing NMF and CMF for each clip and plotted the mean percentage accuracy of all these clips with error bars indicating the 95% confidence range. With both systems, we saw that NMF performed slightly worse than unprocessed and that CMF significantly outperformed both the unprocessed and NMF (Figure 3.5), results which were consistent with our original pilot study.

From King and Atlas [26], we saw that CMF outperformed both the unprocessed and NMF on average. In order to get more details about the performance within those 150 samples, we also looked at the change in scores from unprocessed to processed for the four target/masker gender combinations (see Figure 3.6). There are a few important points to note in this data. First of all, we see that CMF outperforms the unprocessed and NMF ASR results in all four circumstances, showing that adding phase estimates into the analysis can improve performance over magnitude-only NMF analysis. Secondly, the target talker with a masker of the opposite gender performed better in separation than a masker of the same gender. This was consistent with our hypothesis that source separation would perform better on more dissimilar talkers. Thirdly, for both CMF and NMF, we saw that the increase in ASR scores was the highest for male targets with female maskers. We also saw that unprocessed samples with female targets outperform samples with male targets, which was likely due to the higher harmonics for the female speech being less masked by the male speech and thus being processed more successfully by the ASR system. Following this reasoning,

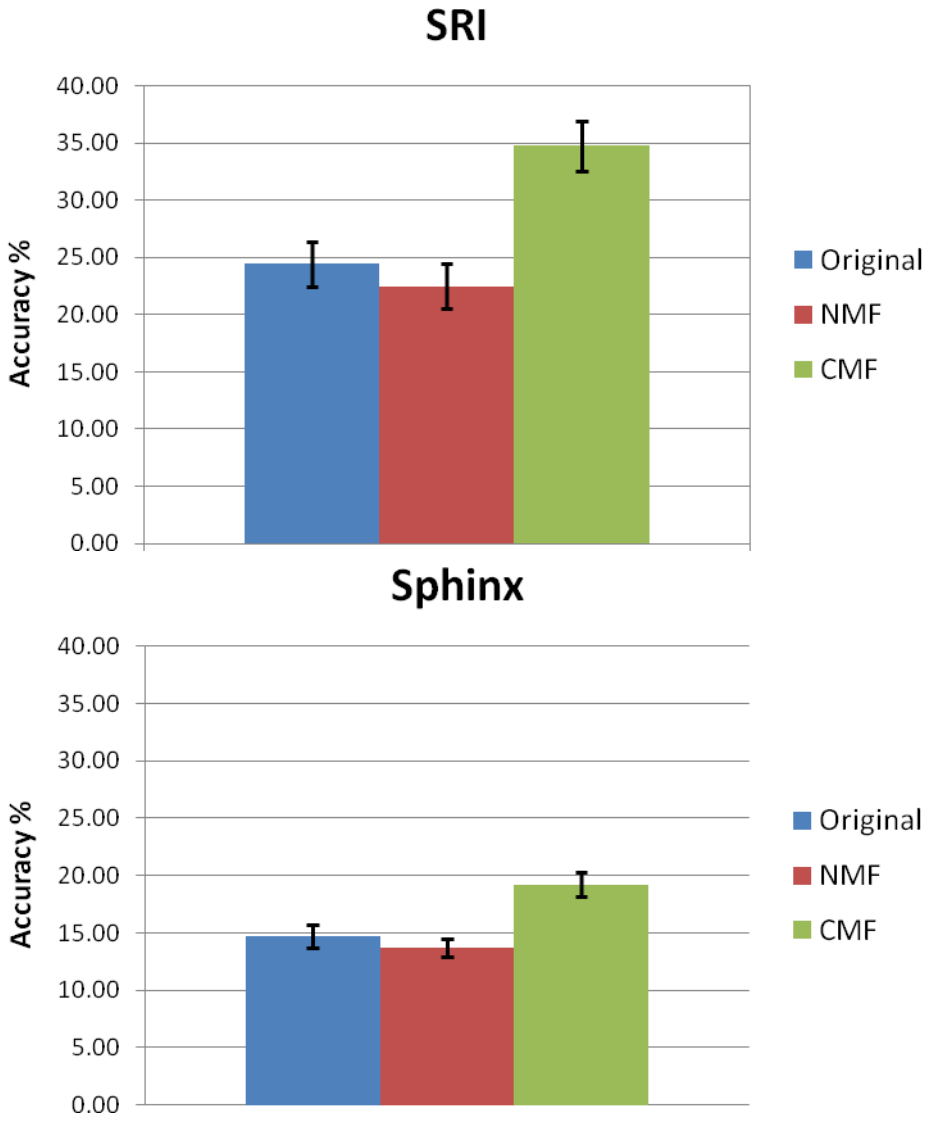

Figure 3.5: ASR results from CMF and NMF. The original signals are two-talker mixtures at 0 dB target-to-masker ratio. Error bars identify 95% confidence interval.

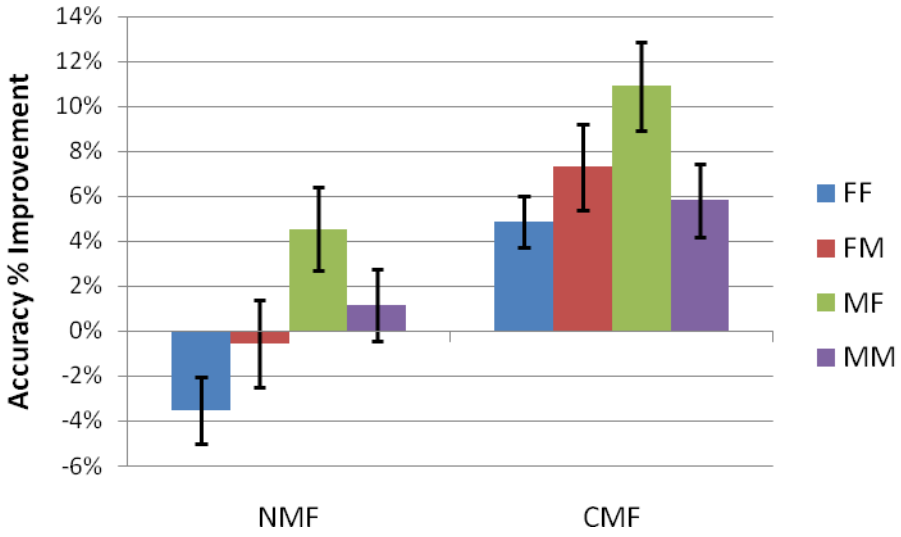

Figure 3.6: Change in absolute ASR recognition score from original mixed signals to separated signal via NMF or CMF for each speaker/target sex combination. For example, "FM" denotes percent change with all signals with a female target and male masker. The original signals are two-talker mixtures at 0 dB target-to-masker ratio. Error bars identify 95% confidence interval.

we hypothesized that since the male targets were doing worse from the beginning, it might be easier to increase their relative score. This hypothesis is consistent with the results, although it may not be the only contributing factor. Of these three points, however, the most significant was the first, that the CMF samples outperformed the NMF and unprocessed samples for all four target/masker combinations.

3.3.2 Factorize-to-Train vs. Copy-to-Train

As previously mentioned, our results comparing unprocessed, CMF, and NMF were consistent with our pilot study. Our results comparing the factorize-to-train and our new copy-to-train method, however, were not consistent with our pilot study [25]. In our pilot study, we saw that when using CMF, the copy-to-train method outperformed the factorize-to-train method for all numbers of bases. With this dataset, we saw something less universal, although the results are fairly consistent with both

recognizers as well as NMF compared with CMF. Figure 3.7 shows the percentage of speech segments with the specified training method performing the best compared with the other training methods on the same original clip. The method that results in the best recognition score the most frequently was factorize-to-train with 1600 bases per speaker, which was the highest number of bases used for factorize-to-train, performing the best almost 35% of the time on average. As the number of basis vectors decreased, the percentage of time that number was best also decreased, with only one exception. The copy-to-train method performed significantly worse than in the pilot study, only performing the best in about 6% of the clips on average.

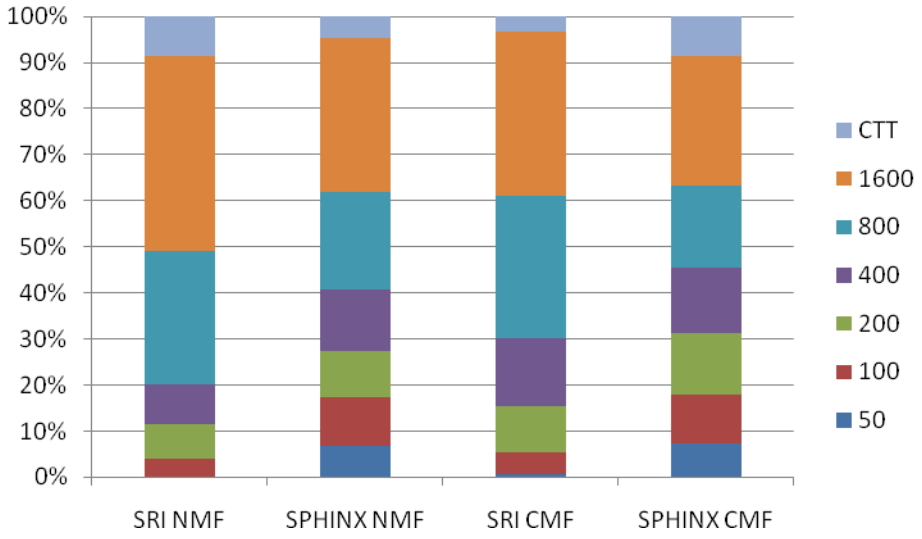

Figure 3.7: Percentage of total samples with the best ASR recognition score using the specified training method. The "CTT" label denotes the copy-to-train method, and the number labels denote the factorize-to-train method with that number of bases trained per talker. The original signals are two-talker mixtures at 0 dB target-to-masker ratio.

We have a few theories about why these results on the training methods differ from those of the pilot study [25]. First of all, the two experiments used speech from different corpora, with the pilot study using data from MSNBC broadcast news and this experiment using the Boston University news corpus [44]. Secondly, in our initial

pilot study, we only used 50, 100, and 200 bases for the factorize-to-train method, which we see from Figure 3.7 do not seem to perform as well as training with more bases. Training with 400, 800, and 1600 bases on the pilot study data may have given results more consistent with our findings. Thirdly, we used a much smaller dataset for the pilot, which likely did not provide enough data for as accurate a distribution of performance as we have collected from our new experiments. Since this experiment contains significantly more data, these results should be given more weight than the small pilot study.

Although these separation results on the new training method were not as clearly promising as the pilot study's results, they were still important as they revealed much about training. The most important result we see from this data is that the method of choosing bases is far from straightforward and is still a key remaining open problem. Our goal was to eliminate the ad hoc method of choosing the number of bases in the factorize-to-train method by using the new copy-to-train method, which we solved. Although the new method is potentially less ad hoc, however, it did not perform as well as the previous factorize-to-train method on this new, larger set of data. So although our new method is not as successful as we had seen in the pilot study, this work is still significant in that it shows the need for a better training method. After all, even the most consistently best-performing method only performs the best less than 35% of the time, which leaves much to be desired.

In conclusion, we demonstrated that incorporating phase estimation via CMF can significantly improve source separation performance by making a controlled comparison with its most closely related variant of NMF, and compared the less parametric and naturally sparse copy-to-train with the commonly used factorize-to-train method. And due to the positive initial CMF results from these studies, we have decided to further research improved models and methods for how to incorporate phase estimates into source separation so that superposition is satisfied. These methods will be presented in the remaining chapters.

Chapter 4

COMPLEX MATRIX FACTORIZATION WITH INTRA-SOURCE ADDITIVITY (CMFWISA): A SIMPLER MODEL THAT SATISFIES SUPERPOSITION

4.1 Model

In Section 2.2, we presented the complex matrix factorization (CMF) model, which extends nonnegative matrix factorization (NMF) model to satisfy superposition by incorporating phase observation and estimation into the algorithm, but is plagued by overparameterization. In Section 3.3, we saw that despite this overparameterization issue, introducing phase into the matrix factorization model can improve source separation performance. Motivated by these results, we present a new model,

$$X_{f,t} \approx \sum_{k_s} B_{f,k_s} W_{k_s,t} e^{\phi_{s,f,t}}, \text{ where} \tag{4.1}$$

$$X \in C^{F \times T}, k_s \in K_s, k \in K, B_{f,k} \geq 0, W_{k,t} \geq 0, \text{ for all elements} \tag{4.2}$$

X is the observed STFT data, B is the set of basis vectors, W is the weight matrix, K is the set of all basis vector indices, and K_s is the set of all basis indices that correspond to source s. This model will be called complex matrix factorization with intra-source additivity (CMFWISA). Although similar to CMF, the key differentiator between CMF and CMFWISA is that for each time-frequency point, CMF estimates a unique phase for each basis vector, while CMFWISA estimates a unique phase for each source. This seemingly small difference, however, is quite significant.

There are four advantages of CMFWISA over CMF. The first is that the CMFWISA model has significantly fewer parameters than the CMF model. For example, consider the task of separating two sources in a fully-supervised case, which entails

finding the best set of weights and phases to model the mixture and separate the sources, given a fixed set of basis vectors for each of the sources. Suppose each source has 1,000 bases. With these conditions, the CMF model would estimate 2,000 phases for each time-frequency point, 1,000 for each source, since each basis vector would have a unique phase. In contrast, the CMFWISA model would estimate two phases for each time-frequency point, one for each source, since each basis vector within a source would share the same phase. So although the CMFWISA still has a higher number of estimated parameters than the observed data, and is therefore overparameterized, it it significantly less overparameterized than CMF.

The second advantage is that the CMFWISA model is a more realistic representation of the physical source. The CMFWISA model assumes that all of the components within a source are additive. For example, when modeling a human speaker with CMFWISA, the phoneme-like basis vectors will be additively combined to estimate the speech. CMF, however, has the flexibility for both adding and canceling out basis vector contributions. In fact, for any three or more basis vectors, phases and weights can be estimated so that they completely cancel each other out, which is an undesirable characteristic. We illustrate this in Figure 4.1, where we use CMF to estimate phases to completely cancel out the nonzero weights and basis vectors. To do this, we fix the basis vectors and the weights, and only allow the phases to update. The base matrix consists of 25 vectors trained on one speaker and 25 trained on another. The weights are all set to one. The observed signal is all zeros. Thus, CMF automatically updates the phases so that this is achieved. Notice how the two estimated are nearly identical in amplitude, but have opposite phase so they cancel each other out when added together. In practice, this particular scenario would not be useful, but this is a useful demonstration to show that overparameterization can end up converging at undesirable solutions.

In the case of a polyphonic source where it may be beneficial to allow the different voices within a source to have different phases, the CMFWISA model can handle this

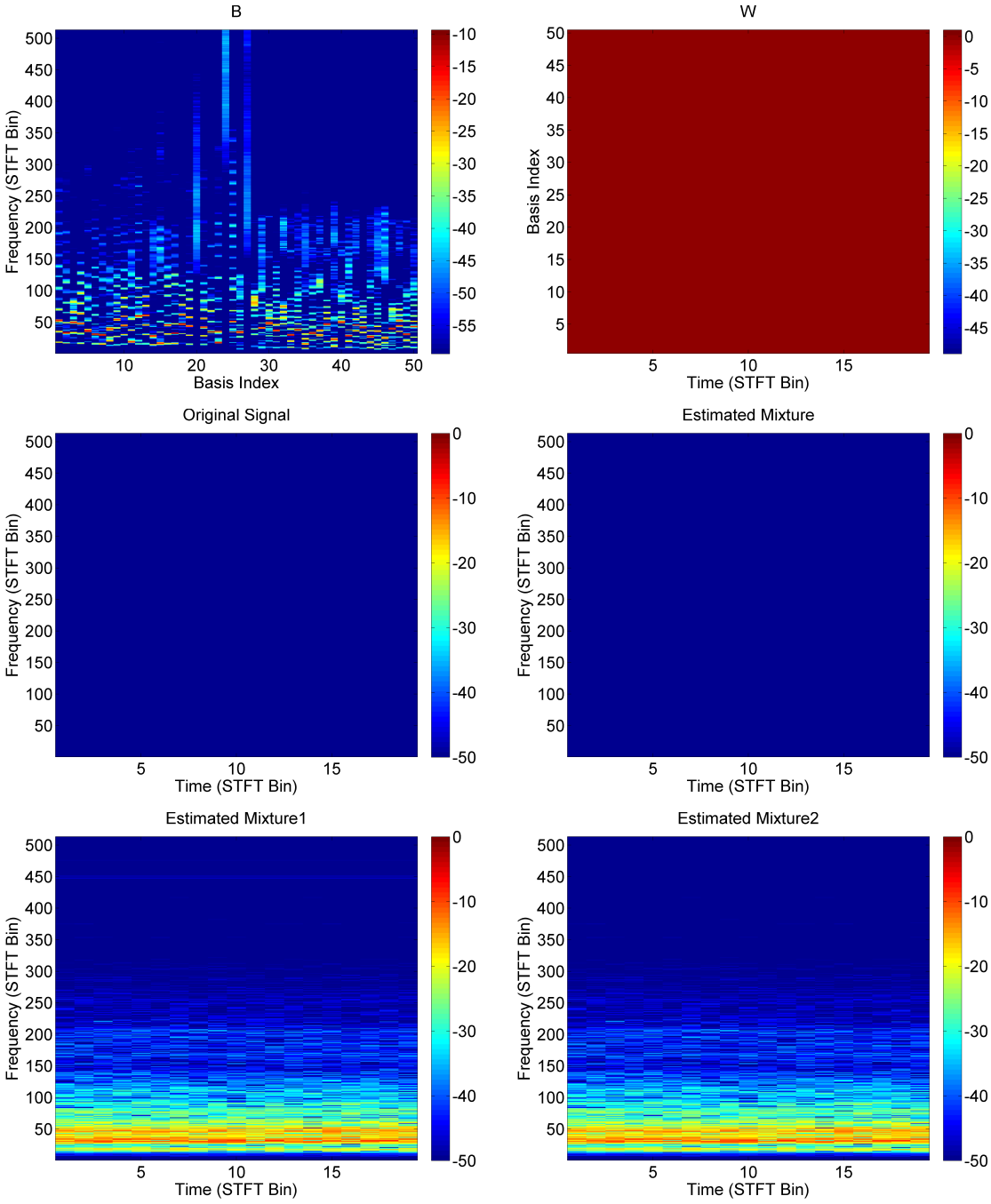

Figure 4.1: Example of how CMF's phase overparameterization can result in an infinite number of solutions. Note: all figures are in dB.

as well by simply modeling each voice as a source. After estimating the source's contributions within a mixture, the voices within a source can be synthesized individually, or added together to approximate the polyphonic source.

The third advantage CMFWISA has over CMF is that since CMFWISA has significantly fewer parameters than CMF, CMFWISA has significant computational advantages in both memory and computation time. Since the phase is the parameter with the largest number of elements, let us compare the memory required to store the phase for an example signal. For this signal, we will be modeling the phase ϕ for two sources with 1,000 basis vectors each over 60 seconds of audio data sampled at 16 kHz with a 50% overlapping window and with double-precision floating values requires

$$60(\text{seconds}) \times 16e^3(\frac{\text{samples}}{\text{second}}) \times 2(\text{overlap factor}) \times 16(\frac{\text{bytes}}{\text{complex double}}) \times 2(\text{sources}) \times 1(\frac{\text{phases}}{\text{source}}) = 61.44 \text{ MB} \tag{4.3}$$

61.44 megabytes is a relatively small amount of memory for a modern computer, and even an hour of data would take up less than 4 gigabytes of memory and could easily be stored with a modern personal computer. Compare this with the memory requirements for CMF,

$$60(\text{seconds}) \times 16e^3(\frac{\text{samples}}{\text{second}}) \times 2(\text{overlap factor}) \times 16(\frac{\text{bytes}}{\text{complex double}}) \times 2(\text{sources}) \times 1e^3(\frac{\text{phases}}{\text{source}}) = 61.44 \text{ GB} \tag{4.4}$$

Although 61.44 GB may be possible with modern super-computers, this is currently out of the realm of possibility for personal computers. In order to calculate CMF, it is usually necessary to resort to computing the CMF model parameters frame-by-frame. After the CMF algorithm converges sufficiently and the individual sources are synthesized, the phases could either be written to the hard drive or discarded. Although this is a workaround at times, not being able to compute matrix factorization on

the entire observed signal has its disadvantages. For example, NMF and probabilistic latent component analysis (PLCA) models that incorporate temporal smoothing [37, 67], hidden Markov models [36], and convolutive models [41, 58] require observing all time windows of the observed STFT signal. These models, though theoretically extendable to CMF, would be computationally impossible with modern hardware. However, such extensions would be possible with the CMFWISA model. Another result of the memory requirements of CMF is that it is typically impossible to find basis vectors for a source using the factorize-to-train method [26] for CMF, since all of a source's training data needs to be observable during the training step. Because of this, NMF or the copy-to-train method is usually used for CMF training. In addition to memory advantages, CMFWISA also has computational speed advantages. Since CMFWISA has fewer parameters to estimate, each iteration is faster because there are fewer computations. Also, fewer parameters tends to result in faster convergence, so fewer iterations may be required for CMFWISA than CMF.

The fourth advantage of CMFWISA over CMF is that it is more consistent with the training methods most commonly used for both methods. As mentioned in the preceding paragraph, factorize-to-train with CMF is typically impossible due to its immense memory requirements for storing the phase for each basis vector. Because of this, CMF will often be performed on the observed mixture with basis vectors determined with the NMF factorize-to-train method [26]. This results in an inconsistent source model, since the NMF during training models the sources as additive and the CMF during separation does not model the sources as additive. This model inconsistency is another factor that limits CMF's performance. In contrast, the CMFWISA model can be easily used in both the training and separation steps. In fact, when used to approximate a single source, CMFWISA reduces to NMF,

$$X_{f,t} \approx \sum_{k_s} B_{f,k_s} W_{k_s,t} e^{s,f,t} \tag{4.5}$$

$$|X_{f,t}|e^{\varphi_{f,t}} \approx \sum_{k_s} B_{f,k_s} W_{k_s,t} e^{\phi_{s,f,t}} \tag{4.6}$$

$$|X_{f,t}| \approx \sum_{k_s} B_{f,k_s} W_{k_s,t}, \text{ since } e^{\varphi_{f,t}} = e^{\phi_{s,f,t}} \tag{4.7}$$

This unification with NMF is another desirable trait with CMFWISA. With CMFWISA, the additivity constraints provided by NMF are satisfied within a single source, while superposition is satisfied between sources because the model allows different sources to have differing phases.

4.2 Cost Function and Algorithm

In the previous section, the CMFWISA model was proposed. The cost function for the associated algorithm is the sum of the squared Frobenius norm of the estimation error and an optional sparsity constraint on the weight matrix W that is controlled by λ and ρ,

$$f(\theta) = \sum_{f,t} |X_{f,t} - Y_{f,t}|^2 + 2\lambda \sum_{k,t} |W_{k,t}|^\rho, \text{ where} \tag{4.8}$$

$$Y_{f,t} = \sum_{s} Y_{s,f,t} \text{ and} \tag{4.9}$$

$$Y_{s,f,t} = \sum_{k_s \in K_s} B_{k_s} W_{k_s,t} e^{j\phi_{s,f,t}} \tag{4.10}$$

Although ρ's values can range between $0 < \rho < 2$, a value of 1 is most common, which makes the sparsity constraint an L_1-norm. In order to find the optimal values $\theta = \{B, W, \phi\}$ to minimize the cost function in eq. 4.10, the following auxiliary function will be used:

$$\begin{aligned} f^+(\theta, \bar{\theta}) = &\sum_{s,f,t} \frac{|\bar{X}_{s,f,t} - Y_{s,f,t}|^2}{\beta_{s,f,t}} \\ &+\lambda \sum_{k,t} (\rho \left|\bar{W}_{k,t}\right|^{\rho-2} W_{k,t}^2 + 2\left|\bar{W_{k,t}}\right|^\rho - \rho \left|\bar{W}_{k,t}\right|^\rho) \end{aligned} \tag{4.11}$$

where

$$\sum_s \beta_{s,f,t} = 1, \beta_{s,f,t} > 0, \text{ for all elements in } \beta \tag{4.12}$$

$$\sum_s \bar{X}_{s,f,t} = X_{f,t} \tag{4.13}$$

$$\bar{W} \in \mathbb{R}^{K \times T} \tag{4.14}$$

This auxiliary function was inspired by the CMF auxiliary function proposed in Kameoka *et al.* [24]. The auxiliary variables introduced are $\bar{\theta} = \{\bar{X}, \bar{W}\}$. A proof to show that eq. 4.11 satisfies the requirements of a proper auxiliary function can be found in Appendix A.1.

To minimize the CMFWISA cost function, an iterative majorization-minimization (MM) algorithm [12] is used. First, for the majorization step, the auxiliary function is updated,

$$\bar{X}_{s,f,t} = Y_{s,f,t} + \beta_{s,f,t}(X_{f,t} - Y_{f,t}) \tag{4.15}$$

$$\bar{W}_{k,t} = W_{k,t} \tag{4.16}$$

Next, in the minimization step, the primary model parameters that minimize the auxiliary function are updated,

$$\phi_{s,f,t} = phase\left(\frac{\bar{X}_{s,f,t}}{|\bar{X}_{s,f,t}|}\right) \tag{4.17}$$

$$B_{f,k_s}^{new} = B_{f,k_s} \frac{\sum_t \frac{W_{k_s,t}|\bar{X}_{s,f,t}|}{\beta_{s,f,t}}}{\sum_t \frac{B_{f,k_s} W_{k_s,t}^2}{\beta_{s,f,t}} + \sum_t \frac{W_{k_s,t} \sum_{k'_s \in (K_s - k_s)} B_{f,k'_s} W_{k'_s,t}}{\beta_{s,f,t}}} \tag{4.18}$$

$$W_{k_s,t}^{new} = W_{k_s,t} \frac{\sum_f \frac{B_{f,k_s}|\bar{X}_{s,f,t}|}{\beta_{s,f,t}}}{\sum_f \frac{B_{f,k_s}^2 W_{k_s,t}}{\beta_{s,f,t}} + \sum_f \frac{B_{f,k_s} \sum_{k'_s \in \{K_s - k_s\}} W_{k'_s,t}}{\beta_{s,f,t}} + \lambda\rho(\bar{W}_{k_s,t})^{\rho-2} W_{k_s,t}} \tag{4.19}$$

The derivations for the updates can be found in Appendix A.2.

Finally, β can be any value that satisfies the conditions in eq. 4.12, but its value can affect the number of updates required to reach a satisfactory minimum value. In order to minimize the cost function the most for each iteration, the β is updated via

$$\beta_{s,f,t} = \frac{\sum_{k_s} B_{f,k_s} W_{k_s,t}}{\sum_k B_{f,k} W_{k,t}} \tag{4.20}$$

A summary of the algorithm's steps are below:

1. Initialize B with random values that vary around 1
2. Initialize W with random values that vary around 1
3. Initialize ϕ with phase of observed matrix X
4. Update β using eq. 4.20
5. Update $\bar{X}$ using eq. 4.15
6. Update $\bar{W}$ using eq. 4.16
7. Update ϕ using eq. 4.17
8. Update B using eq. 4.18
9. Update W using eq. 4.19
10. Repeat steps 4-9 until satisfied

If part or all of one or more of the latent variables in θ are known, it is easy to incorporate them into the algorithm. Simply initialize with the known value and skip its update step. This technique will be employed to in the fully-supervised source separation experiments, where the basis vectors are found in a prior training step.

4.3 Extension - CMFWISA with STFT Consistency Constraints

In the previous section, the CMFWISA model was proposed and its advantages over the CMF model were presented. Although it is a simpler model than CMF with significantly fewer phase parameters, CMFWISA is still over-parameterized. One potential problem with over-parameterization is that there can be several sets of parameters resulting in the same cost function value. It is then impossible to determine which solution is best without additional information. To address this issue, an additional regularization constraint on the STFT consistency of the estimated signals can be added to the cost function. If an STFT of a signal $F_{f,t}$ is consistent, then

$$F_{f,t} = STFT(ISTFT(F_{f,t})) \tag{4.21}$$

An inconsistent STFT is caused by the overlapping regions of its time-domain not containing matching data. If the overlapping data do match, when the windows are added together, the signal remains undistorted. However, if the data does not match up, the overlapping regions end up with components inconsistent with the original signal (see Figure 4.2). In traditional NMF, PLCA, and CMF, the estimated sources within a mixture typically result in inconsistent STFT's. Although there has been much work in the past related to estimating consistent STFT's [19, 43, 48, 39, 18], a new STFT consistency constraint has been proposed for CMF [32]. This constraint consists of the L_2-norm of the STFT consistency error for the STFT of each basis vector contribution,

$$\sum_{k,f,t} \left|C(Y_k)_{f,t}\right|^2, \text{ where} \tag{4.22}$$

$$Y_{k,f,t} = B_{f,k} W_{k,t} e^{j\phi_{k,f,t}} \text{ and} \tag{4.23}$$

$$C(Y_k)_{f,t} = Y_{k,f,t} - STFT(ISTFT(Y_k))_{f,t} \tag{4.24}$$

In this work, the authors showed that introducing this constraint increases the STFT consistency of the estimated signals. This notion of STFT consistency has then been

applied to the CMFWISA model. Instead of enforcing consistency on the STFT of every basis vector contribution, however, it is instead enforced on the STFT of every estimated source,

$$\sum_{s,f,t} \left| C(Y_{s,f,t}) \right|^2, \text{ where} \tag{4.25}$$

$$C(Y_{s,f,t}) = Y_{s,f,t} - STFT(ISTFT(Y_{s,f,t})) \tag{4.26}$$

Enforcing STFT consistency on the source instead of the basis contributions has a significant advantage. Although STFT consistency on the individual basis contributions will lead to consistent sources, it is an unnecessary condition for the basis contributions to be themselves consistent. For example, let's say that a frame of recorded speech can be modeled by a combination of four basis vectors from that source. It makes little sense for the individual component contributions to have to be consistent of their own. Since it is known that the actual source is consistent, what really matters is that the estimated source's STFT is consistent as well.

The cost function for the associated algorithm is the sum of the squared Frobenius norm of the estimation error, an optional sparsity constraint on the weight matrix W that is controlled by λ and ρ, and the squared Frobenius norm of the STFT consistency error of the estimated sources,

$$f(\theta) = \sum_{f,t} \left| X_{f,t} - Y_{f,t} \right|^2 + 2\lambda \sum_{k,t} \left| W_{k,t} \right|^\rho + \gamma \sum_{s,f,t} \left| C(Y_s)_{f,t} \right|^2 \tag{4.27}$$

In order to find the optimal values $\theta = \{B, W, \phi\}$ to minimize the cost function in eq. 4.10, the following auxiliary function will be used.

$$\begin{aligned} f^+(\theta, \bar{\theta}) = & \sum_{s,f,t} \frac{\left| \bar{X}_{s,f,t} - Y_{s,f,t} \right|^2}{\beta_{s,f,t}} \\ & + \lambda \sum_{k,t} (\rho \left| \bar{W}_{k,t} \right|^{\rho-2} W_{k,t}^2 + 2 \left| \bar{W_{k,t}} \right|^\rho - \rho \left| \bar{W}_{k,t} \right|^\rho) \\ & + \gamma \sum_{s,f,t,f',t'} \frac{1}{\delta_{s,f,t,f',t'}} \left| \bar{Z}_{s,f,t,f',t'} - A_{f,t,f',t'} \sum_{k_s} B_{f',k_s} W_{k_s,t'} e^{j\phi_{s,f',t'}} \right|^2 \end{aligned} \tag{4.28}$$

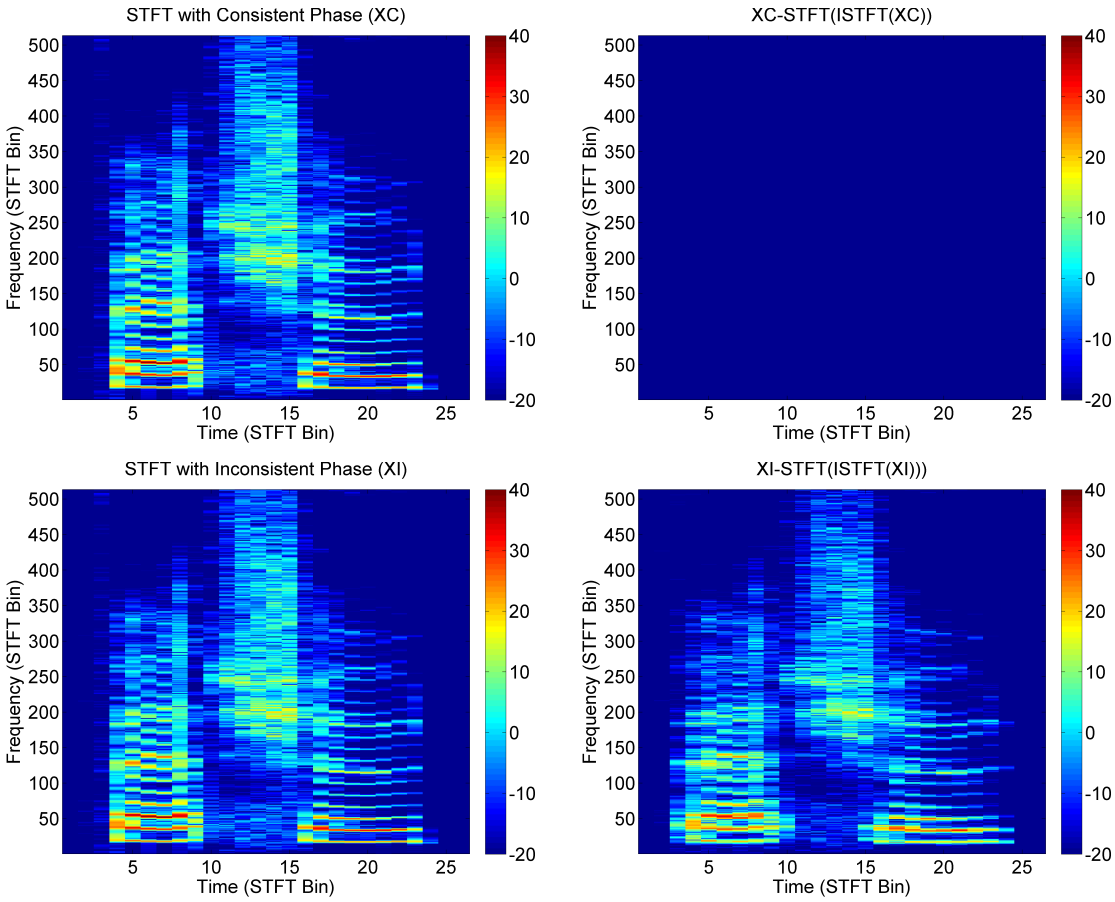

Figure 4.2: Example of how phase affects STFT consistency. Both XC and XI have the same magnitude but differ in phase. XC's phase results in a consistent STFT, while XI's phase results in an inconsistent STFT. Notice how the inconsistent STFT has a nonzero STFT consistency error, $C(XI) = XI - STFT(ISTFT(XI))$ Note: All figures are in dB.

where

$$\sum_{s} \beta_{s,f,t} = 1, \beta_{s,f,t} > 0, \text{ for all elements in } \beta \tag{4.29}$$

$$\sum_{f',t'} \delta_{s,f,t,f',t'} = 1, \delta_{s,f,t,f',t'} > 0, \text{ for all elements in } \delta \tag{4.30}$$

$$\sum_{s} \bar{X}_{s,f,t} = X_{f,t} \tag{4.31}$$

$$\sum_{f',t'} \bar{Z}_{s,f,t,f',t'} = 0 \tag{4.32}$$

$$\bar{W} \in \mathbb{R}^{K \times T} \tag{4.33}$$

$A_{f,t,f',t'}$ is the matrix representation of the STFT consistency error term in eq. 4.26, so that

$$C(Y_s)_{f,t} = \sum_{f',t'} A_{f,t,f',t'} Y_{s,f',t'} \tag{4.34}$$

The auxiliary variables introduced are $\bar{\theta} = \{\bar{X}, \bar{W}, \bar{Z}\}$.

To minimize the cost function, an iterative majorization-minimization (MM) algorithm is again used. First, for the majorization step, the auxiliary function is updated,

$$\bar{X}_{s,f,t} = Y_{s,f,t} + \beta_{s,f,t}(X_{f,t} - Y_{f,t}) \tag{4.35}$$

$$\bar{W}_{k,t} = W_{k,t} \tag{4.36}$$

$$\bar{Z}_{s,f,t,f',t'} = A_{f,t,f',t'} Y_{s,f,t} - \delta_{s,f,t,f',t'} C(Y_s)_{f,t} \tag{4.37}$$

Next, in the minimization step, the primary model parameters that minimize the auxiliary function are updated in succession,

$$\phi_{s,f,t} = phase\left[\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + 2\gamma\left(\sum_{f',t'} \frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}} Y_{s,f,t} - C^H(C(Y_s))_{f,t}\right)\right] \tag{4.38}$$

$$B^{new}_{f,k_s} = B_{f,k_s} \sum_t Re\left[\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + \gamma \sum_{f',t'}\left(\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}} Y_{s,f,t} - C^H(C(Y_s))_{f,t}\right)\right] W_{k_s,t}$$
$$\div\left[\sum_t \left(B_{f,k_s} W^2_{k_s,t} + W_{k_s,t} \sum_{k'_s \in (K_s - k_s)} B_{f,k'_s} W_{k'_s,t}\right)\left(\sum_{f',t'} \frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}} + \frac{1}{\beta_{s,f,t}}\right)\right] \tag{4.39}$$

$$W_{k_s,t}^{new} = W_{k_s,t} \sum_f Re\left[\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + \gamma \sum_{f',t'}\left(\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}} Y_{s,f,t} - C^H(C(Y_s))_{f,t}\right)\right] B_{f,k_s}$$
$$\div\left[\sum_f\left(B_{f,k_s}^2 W_{k_s,t} + B_{f,k_s}\sum_{k'_s\in(K_s-k_s)} B_{f,k'_s}W_{k'_s,t}\right)\left(\sum_{f',t'}\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}} + \frac{1}{\beta_{s,f,t}}\right)\right.$$
$$\left. + \lambda\rho(\bar{W}_{k_s,t})^{\rho-2}W_{k_s,t}\right] \tag{4.40}$$

The derivations for the updates can be found in Appendix B.2.

β can again be any value that satisfies the conditions in eq. 4.29, but its value can affect the number of iterations required to reach a stationary point. In order to achieve satisfactory minimization the fastest (i.e. with the fewest iterations), the β is updated via

$$\beta_{s,f,t} = \frac{\sum_{k_s\in K_s} B_{f,k_s}W_{k_s,t}}{\sum_s\sum_{k_s\in K_s} B_{f,k_s}W_{k_s,t}} \tag{4.41}$$

Also, directly storing or calculating any of the five-dimensional terms is at the edge of tractability. Fortunately, the simplifications for CMF with STFT consistency constraints presented in Le Roux *et al.* [32] can be applied here.

$$\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}} = a, \text{ where} \tag{4.42}$$
$$a = \left(\sum |A_{f,t,f',t'}|\right)^2 \tag{4.43}$$

More details on constructing A can be found in Le Roux [29].

A summary of the algorithm's steps are below:

1. Initialize B with random values that vary around 1

2. Initialize W with random values that vary around 1

3. Initialize ϕ with phase of observed matrix X

4. Update β using eq. 4.41

5. Update $\bar{X}$ using eq. 4.35
6. Update $\bar{W}$ using eq. 4.36
7. Update $\bar{Z}$ using eq. 4.37
8. Update ϕ using eq. 4.38
9. Update B using eq. 4.39
10. Update W using eq. 4.40
11. Repeat steps 4-10 until satisfied

4.4 Experiments

In this section, we will compare the performance of the CMWISA model with NMF and CMF on two-talker source separation. The same training and test data from Section 3.1 was used in these examples, which were 16 mixtures of TIMIT sentences at -10, -5, 0, +5, and +10 dB target-to-masker ratio (TMR). The basis vectors are found using the copy-to-train method on seven sentences of training data for each of the speakers in the mixture. Then, the weights (and phases, if applicable) are estimated for the mixture data. To maintain consistency with the previous experiments using this data, the source-to-distortion ratio (SDR) and source-to-interference ratio (SIR) value are the relative changes in dB from the unprocessed data, and the source-to-artifact ratio (SAR) value is the value of the processed signal.

In the following figures of results, there are three CMFWISA parameters that we will be varying. The first controls the seeding values of the weight matrix, and is reflected in the first number following the type of matrix factorization used (CMFWISA, CMF, or NMF). A 1 indicates that the weight matrix was seeded with the results from the NMF algorithm, and a 0 indicates that the weight matrix was seeded

randomly. The second parameter is whether or not the weight matrix update equation incorporated STFT consistency constraints, and is reflected in the second number. A value of 1 indicates that STFT consistency is incorporated into the weight estimate, and a 0 indicates that it is not. The third parameter is the value of γ which controls the importance of STFT consistency in the cost function and updates, and is reflected in the third number. If γ is nonzero, then the phase update always incorporates STFT consistency, and the second parameter value indicates whether it is incorporated into the weight updates as well. The NMF and CMF parameters are all 0 because they always use a random seed and do not incorporate STFT consistency constraints into the updates. The following parameters are constant throughout the experiments: all use a window size of 512 points, a 50% overlap, and a magnitude exponent of 1. These parameters were chosen because they performed the best in the NMF experiments on the same data that was presented in Section 3.1. For NMF, the squared Frobenius norm was chosen because it is most similar to the CMFWISA and CMF cost functions.

Each set of experiments has two accompanying figures, one with the SDR, SIR, and SAR measurements from the blind source separation evaluation toolbox (BSS Eval) [66] from the signals separated using the SIRMAX synthesis method, and the other using the SARMAX method. The SIRMAX method synthesizes the separated signals simply by taking the inverse STFT of the signal estimated directly from the matrix factorization algorithm. So, the NMF SIRMAX STFT estimate is

$$Y_{s,f,t}^{SIRMAX} = \left(\sum_{k_s} B_{f,k_s} W_{k_s,t} \right) e^{j\phi_{f,t}}, \text{ where } e^{j\phi_{f,t}} = \frac{X_{f,t}}{|X_{f,t}|} \tag{4.44}$$

The CMF STFT estimate is

$$Y_{s,f,t}^{SIRMAX} = \left(\sum_{k_s} B_{f,k_s} W_{k_s,t} \right) e^{j\phi_{k_s,f,t}} \tag{4.45}$$

The CMFWISA STFT estimate is

$$Y_{s,f,t}^{SIRMAX} = \left(\sum_{k_s} B_{f,k_s} W_{k_s,t} \right) e^{j\phi_{s,f,t}} \tag{4.46}$$

As one might imagine, the SIRMAX method typically has better suppression of the interfering signal(s), but has more artifacts than the other method. On the other hand, the SARMAX method typically results in fewer artifacts at the expense of poorer suppression of the interfering signals. The SARMAX synthesis method computes the ISTFT of the following signal:

$$Y_{s,f,t}^{SARMAX} = \frac{\left|Y_{s,f,t}^{SIRMAX}\right|}{\sum_s \left|Y_{s,f,t}^{SIRMAX}\right|} X_{f,t} \tag{4.47}$$

The SARMAX synthesis method ensures a lossless decomposition, which reduces artifacts. The SARMAX is most popular in NMF because it reduces artifacts and typically produces results that sound better to human listeners.

The experimental section for CMFWISA is divided into three sets. The first set of tests evaluates the performance of CMWISA when the phase of the individual sources is known to be estimated correctly. The second set compares the performance of CMFWISA with STFT consistency constraints on both the phase and weight updates with using constraints on only the phase updates. The third set compares CMFWISA with NMF and CMF.

4.4.1 Speech Separation via CMFWISA with Oracle Phase

The first set of tests evaluates the performance of CMWISA when the phase of the individual sources is known to be estimated correctly. In this set, instead of estimating the phase every iteration, the phase of the individual sources is instead used. We call this the "oracle phase." Although using the oracle phase is not typically a possibility in real-world usage, it is used here to isolate the phase and weight estimation steps in order to see how well the model can work if phase is estimated well. This is a good first test because it can be used to estimate the potential of the algorithm, as well as see whether the phase or weight update performance is the limiting factor. NMF separation using the squared Frobenius norm cost function is included as a reference. An example of CMFWISA's performance with oracle phase is shown in Figure 4.3,

and an example of NMF can be seen in Figure 4.4. These figures also clearly illustrate the difference between the SIRMAX and SARMAX synthesis methods. Both of these examples use the same basis vectors and source material and mixing at 0 dB TMR.

The performance of CMWISA with oracle phase and NMF using the SIRMAX synthesis method is seen in Figure 4.5 and using the SARMAX synthesis method is seen in Figure 4.6. The results clearly show that the CMFWISA method significantly outperforms NMF in all conditions and with both synthesis methods. The greater performance differences are seen at lower TMR's, which is sensible since there is greater potential for improvement at lower TMR's. With the SIRMAX method, we see approximately a 30 dB improvement in SIR in all TMR's, while the SAR ranges from 11 dB at the highest TMR (Figure 4.5, top right plot) to 6 dB at the lowest (Figure 4.5, bottom right plot). These combine to give a 0.5 dB SDR improvement for the highest TMR (Figure 4.5, top left plot) up to a 16 dB SDR improvement in the lowest TMR (Figure 4.5, bottom left plot). Again, the reason such a small improvement is seen in the SDR of the highest (+10 dB) TMR is that the SDR of the original signal is already quite high. With the SARMAX method, the SIR ranges from 11 dB at the highest TMR (Figure 4.6, top middle plot) to 16 dB at the lowest (Figure 4.6, bottom middle plot), while the SAR ranges from 12 dB at the highest TMR (Figure 4.6, top right plot) to 7 dB at the lowest (Figure 4.6, bottom right plot). These combine to give a 4.5 dB SDR improvement for the highest TMR (Figure 4.6, top left plot) up to a 12 dB SDR improvement in the lowest TMR (Figure 4.6, bottom left plot).

There are a few significant findings from this set of experiments. First of all, when the phases of the individual sources are known *a priori* or are estimated well, the CMFIWSA method performs very well, up to about 32 dB in SIR, 8 dB in SAR, and 13 dB in SDR. Secondly, the synthesis methods performed as expected, with the SIRMAX synthesis method indeed having the best SIR, while the SARMAX method having the best SAR for a given model.

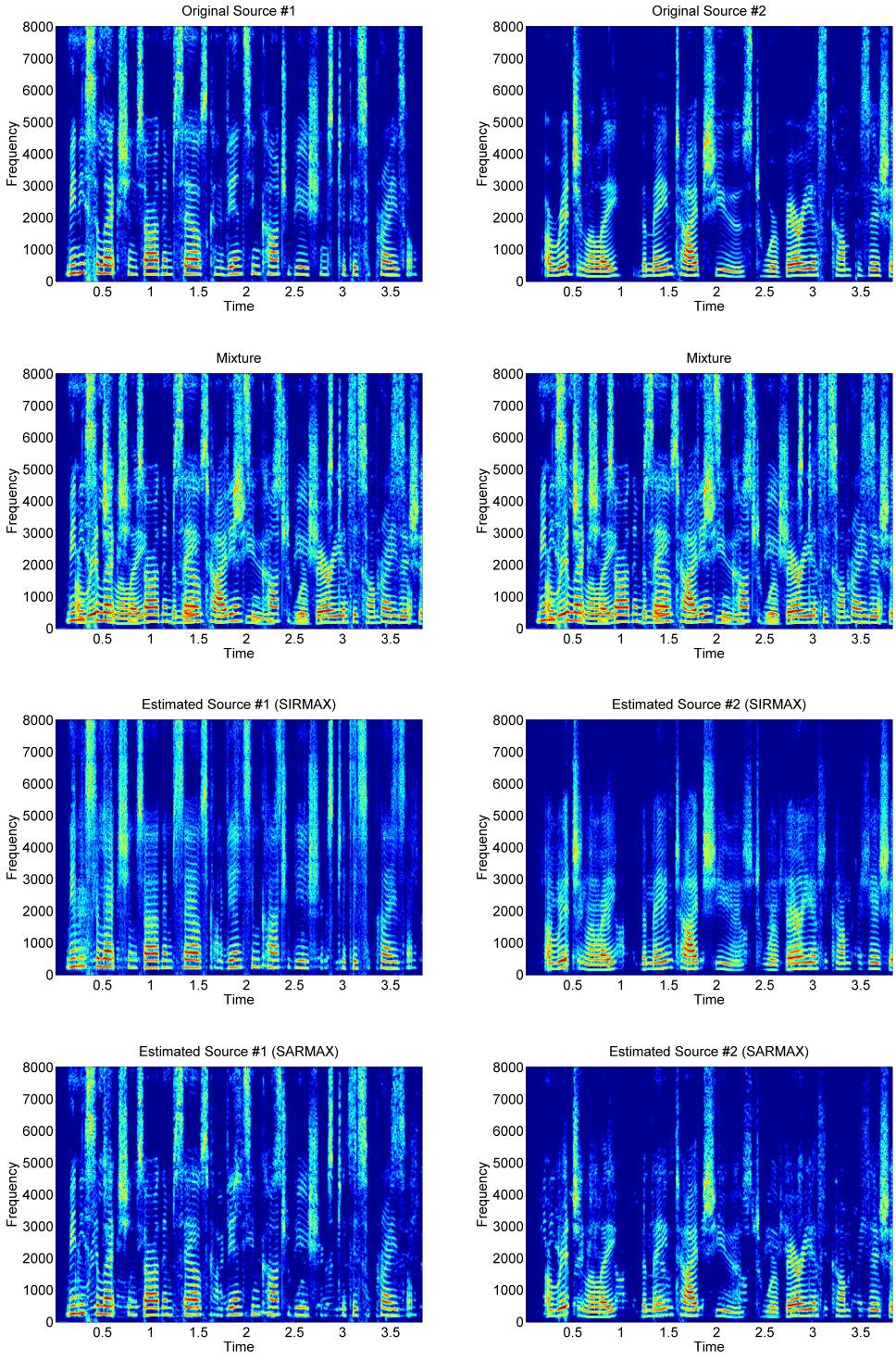

Figure 4.3: Example of CMFWISA-based source separation using oracle phase.

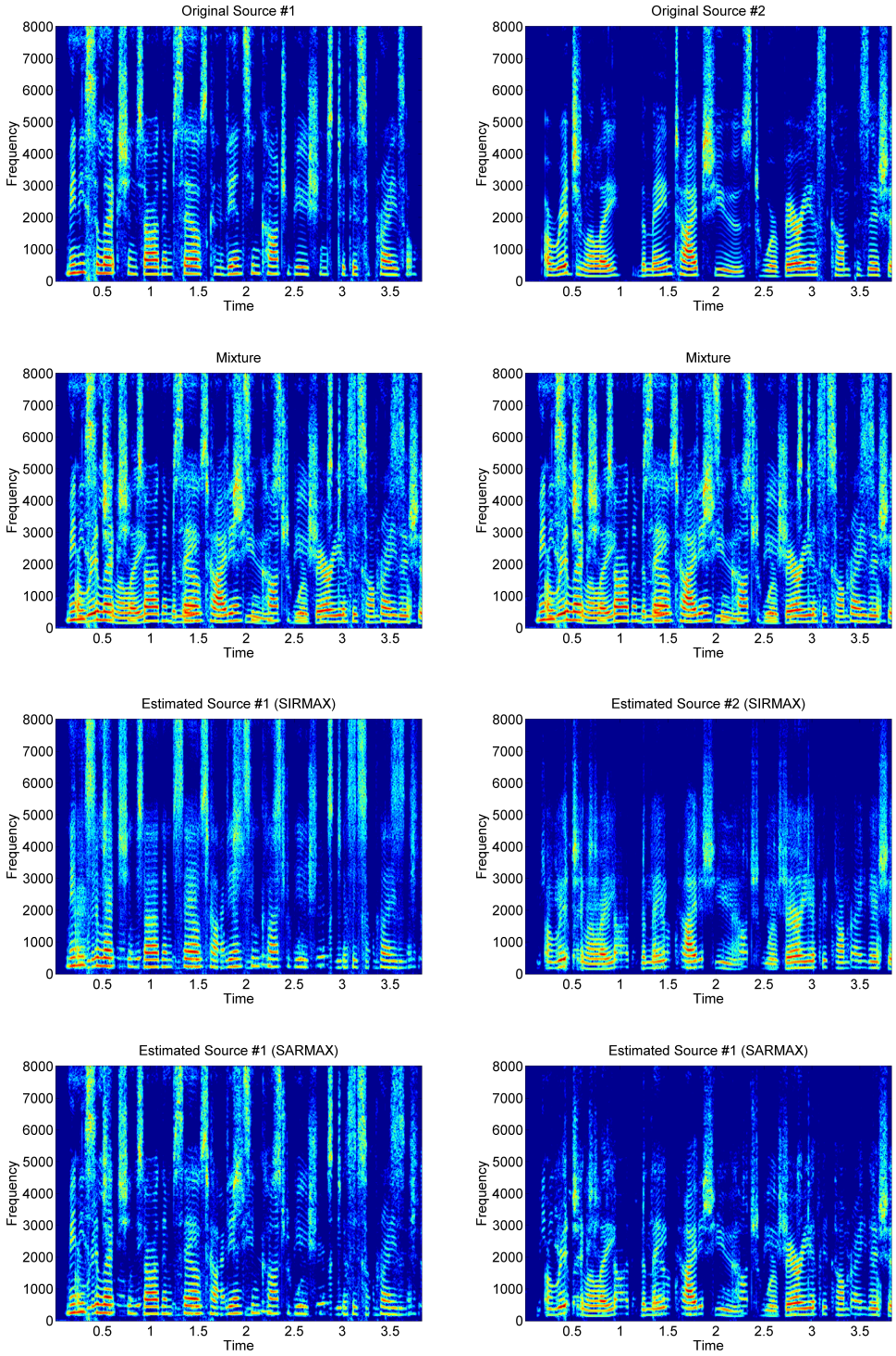

Figure 4.4: Example of NMF-based source separation.

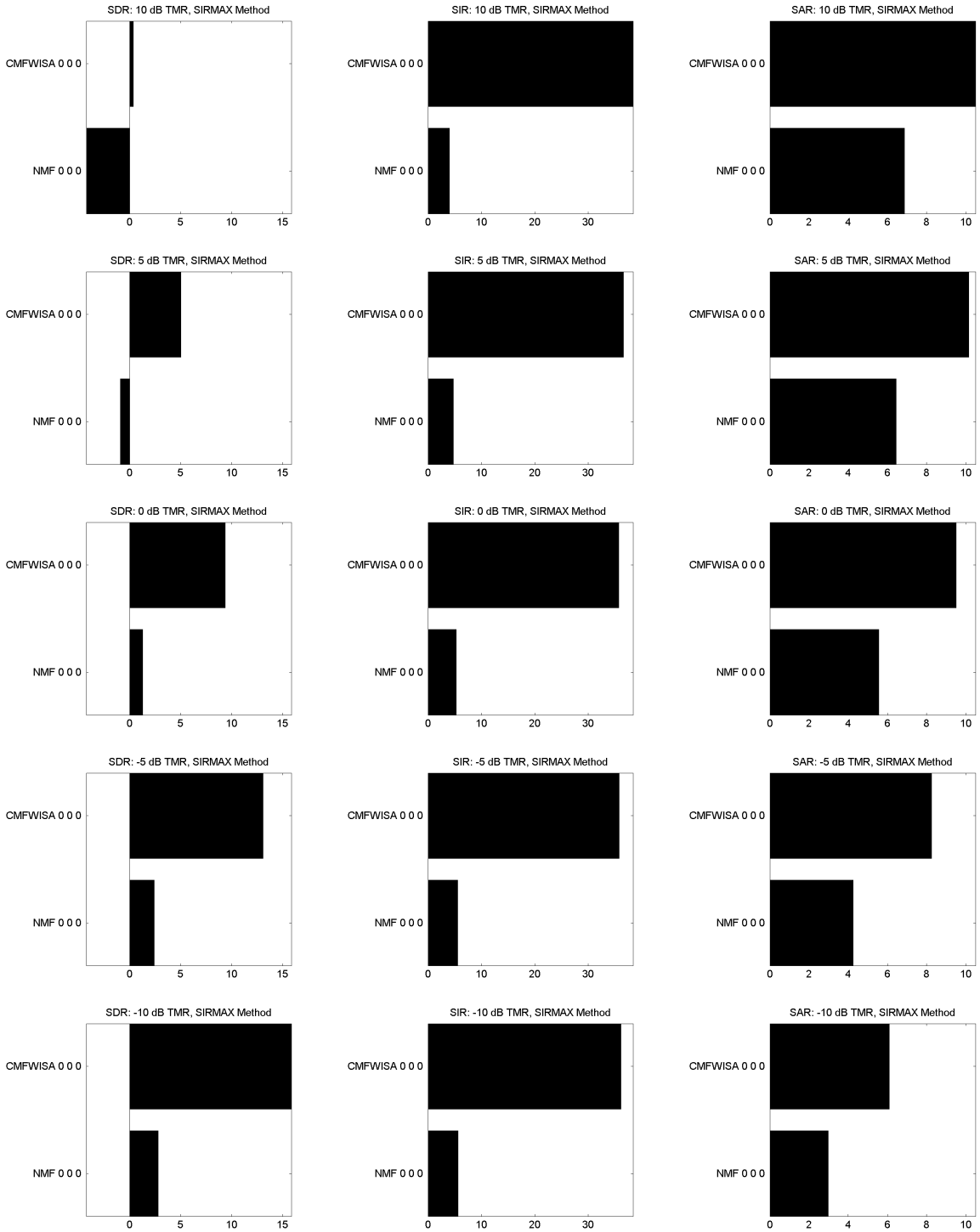

Figure 4.5: BSS Eval measurements for separated sources synthesized with the SIRMAX method (oracle phase).

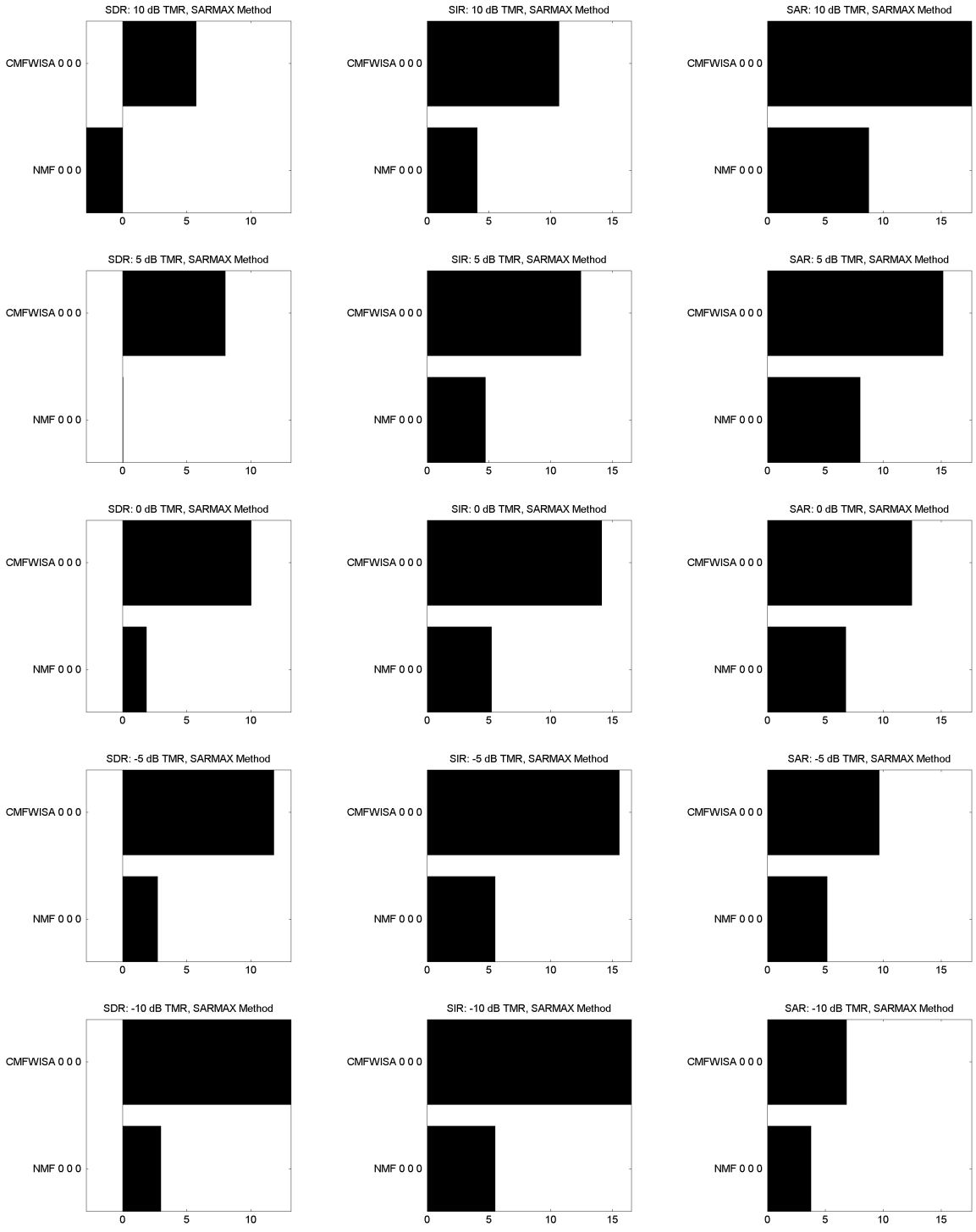

Figure 4.6: BSS Eval measurements for separated sources synthesized with the SARMAX method (oracle phase).

4.4.2 Speech Separation with CMFWISA Using Estimated Phase, With and Without Incorporating STFT Consistency in the Weight Updates

The second set of experiments compares the performance of CMFWISA with STFT consistency constraints on both the phase and weight updates with applying the constraints on only the phase updates. The purpose of these experiments is to see if the STFT consistency constraints applied to the weight and phase updates improve CMFWISA performance. These experiments are only done with the 0 dB TMR case.

Results for CMWISA using the SIRMAX synthesis method are seen in Figure 4.8 and using the SARMAX synthesis method are seen in Figure 4.9. There are two key conclusions from these results. First of all, we see that adding STFT consistency constraints to the weight update equations degrades performance. This result, which was puzzling initially, can be understood with the following explanation. Increasing the value of γ in eq. 4.27 increases the importance of minimizing the squared Frobenius norm of the STFT consistency error,

$$\sum_{s,f,t} \left| \sum_{k_s} B_{f,k_s} W_{k_s,t} e^{j\phi_{s,f,t}} - STFT(ISTFT(\sum_{k_s} B_{f,k_s} W_{k_s,t} e^{j\phi_{s,f,t}}))_{f,t} \right|^2 \tag{4.48}$$

The hope is that this additional term in the cost function will increase the STFT consistency of the estimated sources without much degradation in the estimation of the observed mixture. Unfortunately, what is actually happening is that increasing the importance of STFT consistency degrades the estimation of the mixture more than it helps the estimation of the sources. This is occurring because the STFT consistency terms are causing the values in W to shrink too much. This is permitted within the model because a smaller W also results in a smaller error in this term. In the extreme case, a W of all zeros results in the STFT consistency error in eq. 4.48 to be zero. This is similar to the problem encountered when making the sparsity constraint so high that it causes all the weights to go to zero, and thus minimize the sparsity of W at the expense of the reconstruction error of the observation. This is shown in

Figure 4.7, where as the value of γ increases, the overall energy of the estimated signal decreases significantly. This is also reflected in the results, where we see that as γ increases, the SDR decreases with both synthesis methods. In conclusion, the first key point from this set of experiments is that the STFT consistency constraint in its current form degrades performance when used in the weight update equations.

While the first point is disappointing, the second key result is much more promising. The highest SDR in these experiments is when the STFT consistency constraint is applied to the phase update only. With the SARMAX method, γ's of 10 and 100 both have higher SDR's and SIR's than with a γ of 0, and with the SIRMAX method, γ's of 10 and 100 both have higher SIR's than with a γ of 0. However, the SAR values are slightly lower for γ's of 10 and 100 than 0. Initial analysis indicates that γ may be used as an additional tuning parameter to control whether SIR or SAR are maximized. If this is true, such a parameter may be very useful, as different applications have different signal requirements. This idea will be explored further in the next set of experiments, where the STFT consistency constraints applied to phase updates are applied to all target-to-masker ratio (TMR) values in the test set. To summarize this set of experiments, we see that using the STFT consistency constraints for the phase update equations can improve performance, especially SIR, while using this constraint on the weights causes the weights to become too small and thus degrade performance.

4.4.3 Speech Separation with CMFWISA using Estimated Phase

The third set of experiments sees how STFT consistency constraints affect the performance of CMFWISA, as well as comparing with CMF and NMF as references. Also, we want to see whether seeding with NMF-derived values increases performance. For each mixture, the same set of basis vectors and initial values were used for all the CMFWISA, NMF, and CMF tests. The weight matrix from the NMF results are used as the seeding values for CMFWISA. This was done so that we can see if the

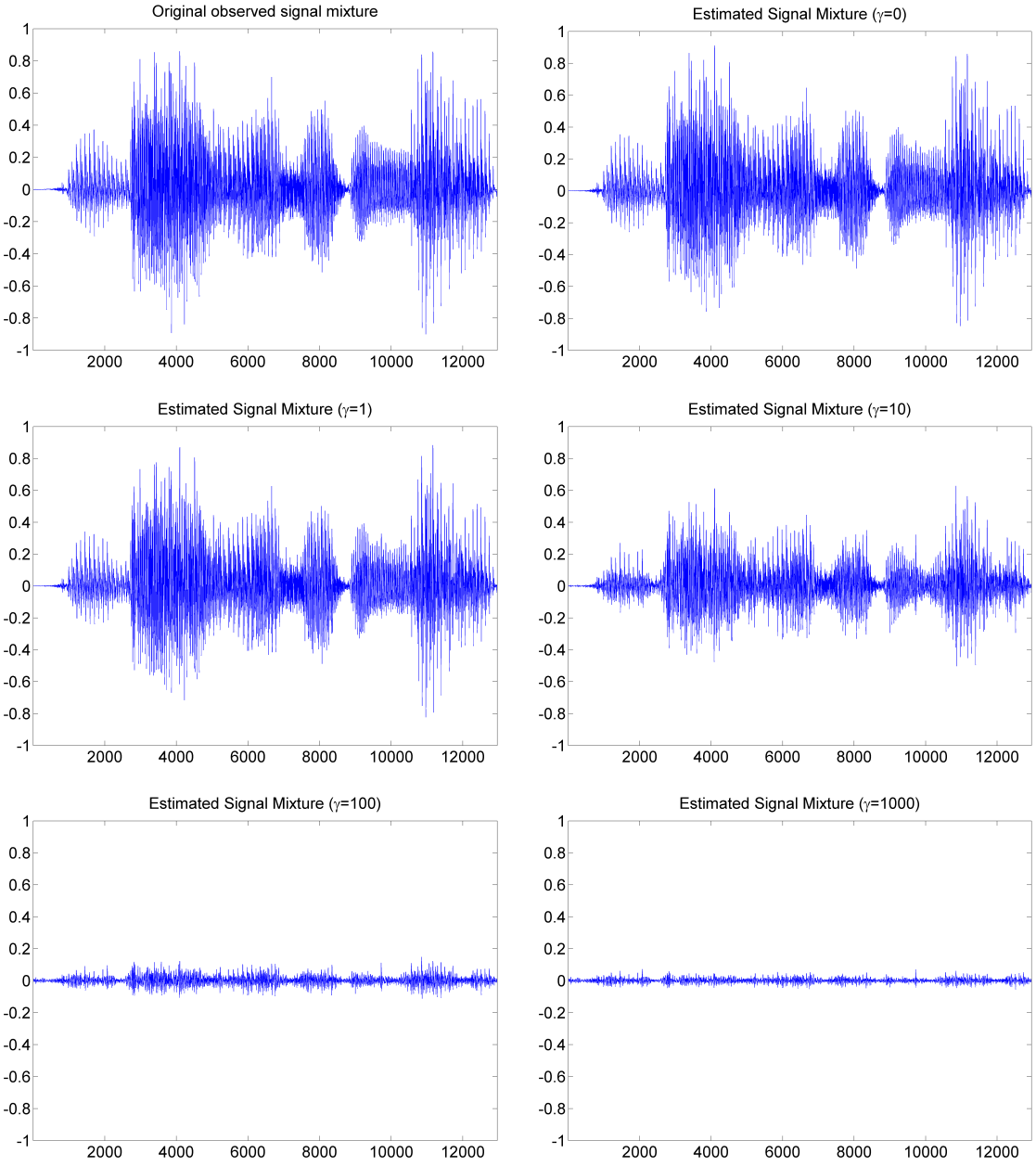

Figure 4.7: Example of how the STFT consistency constraint γ, if applied to the weight updates, can attenuate the estimated signal too much.

CMFWISA improves or degrades performance compared with its NMF starting point. The SIRMAX results are in Figure 4.10 and the SARMAX results are in Figure 4.11.

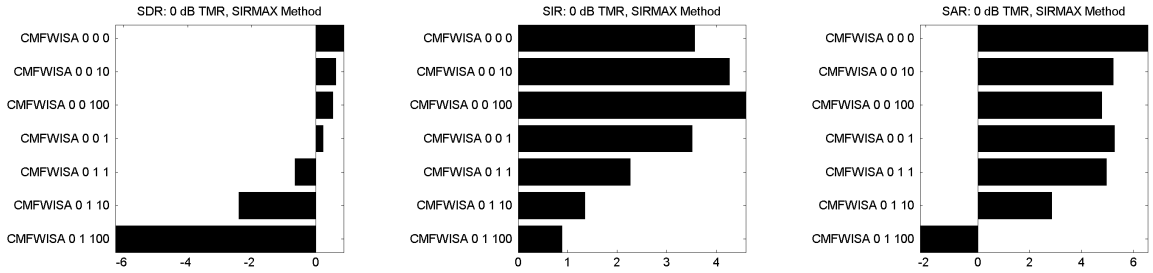

Figure 4.8: BSS Eval measurements for separated sources synthesized with the SIRMAX method (compares performance with and without STFT consistency constraint used in W update).

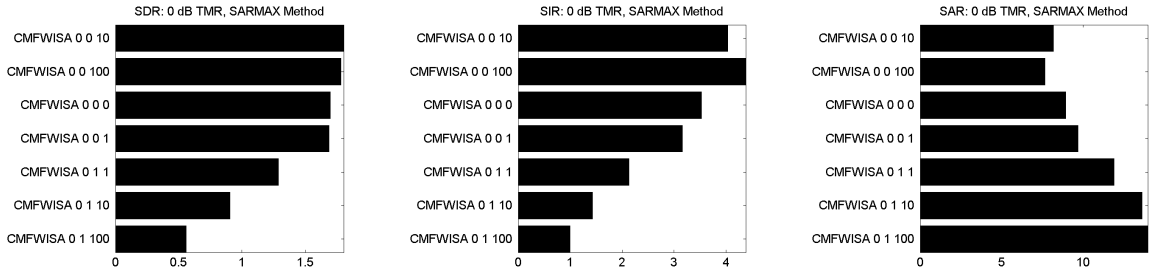

Figure 4.9: BSS Eval measurements for separated sources synthesized with the SARMAX method (compares performance with and without STFT consistency constraint used in W update).

There are a few key results found in these experiments. First of all, we see that CMF consistently has the highest SAR, especially in the SIRMAX synthesis method, but always has the worst SIR. As discussed in Section 2.2, we have seen that CMF has the highest number of parameters due to the phase term $\phi_{k,f,t}$, which allows the mixture data to be fit better than with any other model. However, since there are an unlimited number of parameter values that fit the observed mixture, the algorithm usually ends up in a local minimum that describes the mixture well, but not the individual sources. Thus, there are few artifacts, but poor separation. Just like we have seen before, at higher TMR's, the SAR plays a larger role than SIR in determining SDR, and at lower TMR's, the order is reversed. This is why the CMF has the highest

SDR for the +10 and +5 dB conditions with the SIRMAX synthesis method (Figure 4.10, bottom and second from bottom left plots). So if minimizing artifacts is of highest importance, and computational time and resources is not an issue, then CMF may be the best choice. But if source separation is more important, both CMFWISA and NMF can achieve significantly (>2 dB) better separation performance with very little (<1 dB) increase in artifacts when using the SARMAX synthesis method.

The second key result is how the performance of CMFWISA compared with that of NMF. NMF resulted in the highest SIR with in lower TMR's ($\leq$ 0 dB), and CMFWISA was better in higher TMR's ($\geq$ 5 dB) using the SIRMAX synthesis method (Figure 4.10, middle column plots). CMFWISA, then, lay in the middle between NMF and CMF performance, with an SIR worse than NMF and better than CMF, and an SAR better than NMF and worse than CMF. We will focus on the SARMAX results because it is a more popular method than SIRMAX because it typically produces better-sounding results and a higher SDR. In the SARMAX case, CMFWISA has the highest SDR with the exception of the -10 dB TMR case (Figure 4.11, bottom left plot), where SIR influences the SDR value much more than the SAR. In the higher TMR examples, where SAR dominates SDR, NMF scores the worst and CMFWISA without NMF seeding and with a small STFT consistency constraint value performs best (Figure 4.11, first and second plots in the left column). In the 0 dB TMR and higher scenario, seeding with NMF and using a higher STFT consistency constraint produces the best results (Figure 4.11, middle left plot).

When CMFWISA is seeded with NMF weights (indicated by the first of the three numbers being a 1), it tends to perform more similarly to NMF than without seeding, which is intuitive. It is logical that results from a seeded initialization would be more similar to results from a random initialization. Therefore, CMFWISA with an NMF seed will typically result in a higher SIR and lower SAR than with a random seed. Also, a higher value of γ, which increases the importance of STFT consistency, tends to result in a higher SIR and lower SAR, which was not what we had predicted. We

had hypothesized that making STFT consistency more important on the estimated sources would instead increase SAR as well as SIR. Our theory of why this is happening is that the STFT consistency constraint may be focusing more on removing parts of the signal it deems to not fit with the estimated signal, thus increasing SIR, than adding in missing pieces of the signal, thus decreasing SAR.

In its current form, CMWISA offers flexibility in finding desired results between the high-SIR/low-SAR NMF and the low-SIR/high-SAR CMF. We have shown how the NMF seeding and STFT consistency constraints can be used to find the desired trade-off between SIR and SAR. From the SARMAX results, we see that CMFWISA has the highest SDR in all cases, except in the -10 dB TMR case (Figure 4.11, bottom left plot) where NMF is slightly better, though the difference ($<$.05 dB) is insignificant. And although these results are positive and show great promise, there is much more potential for better performance with CMFWISA. The biggest area for improvement is in phase estimation. In our first set of CMWISA experiments in Section 4.4.1, we saw that if the phase is estimated correctly, CMFWISA can produce results much better than these. So although the STFT consistency constraint improves the SIR performance, we think that there is potential to achieve even better phase estimation that will lead to better separation performance. So although we have proposed many answers and solutions with our work, we have perhaps provided even more new questions and topics for future research.

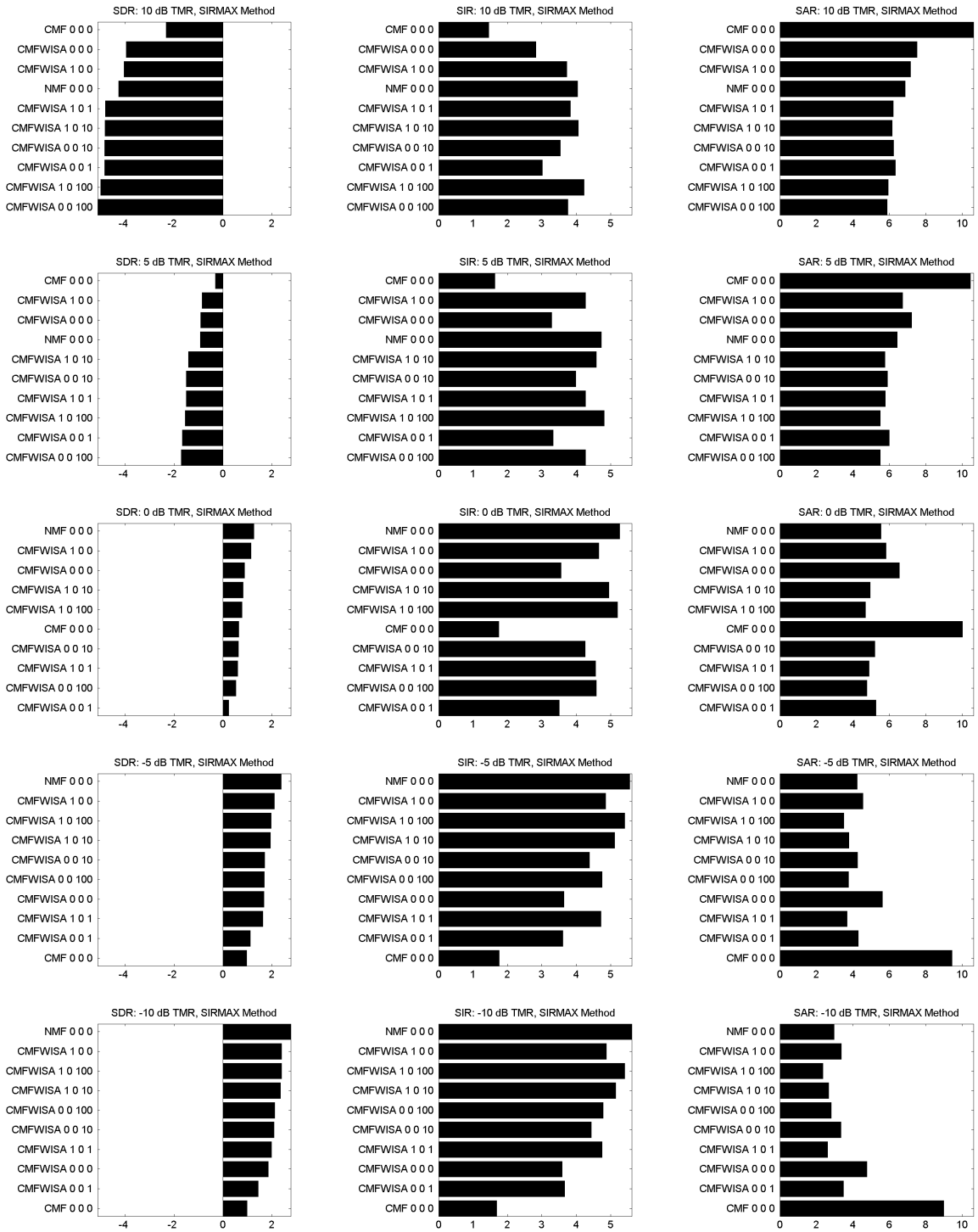

Figure 4.10: BSS Eval measurements for separated sources synthesized with the SIR-MAX method.

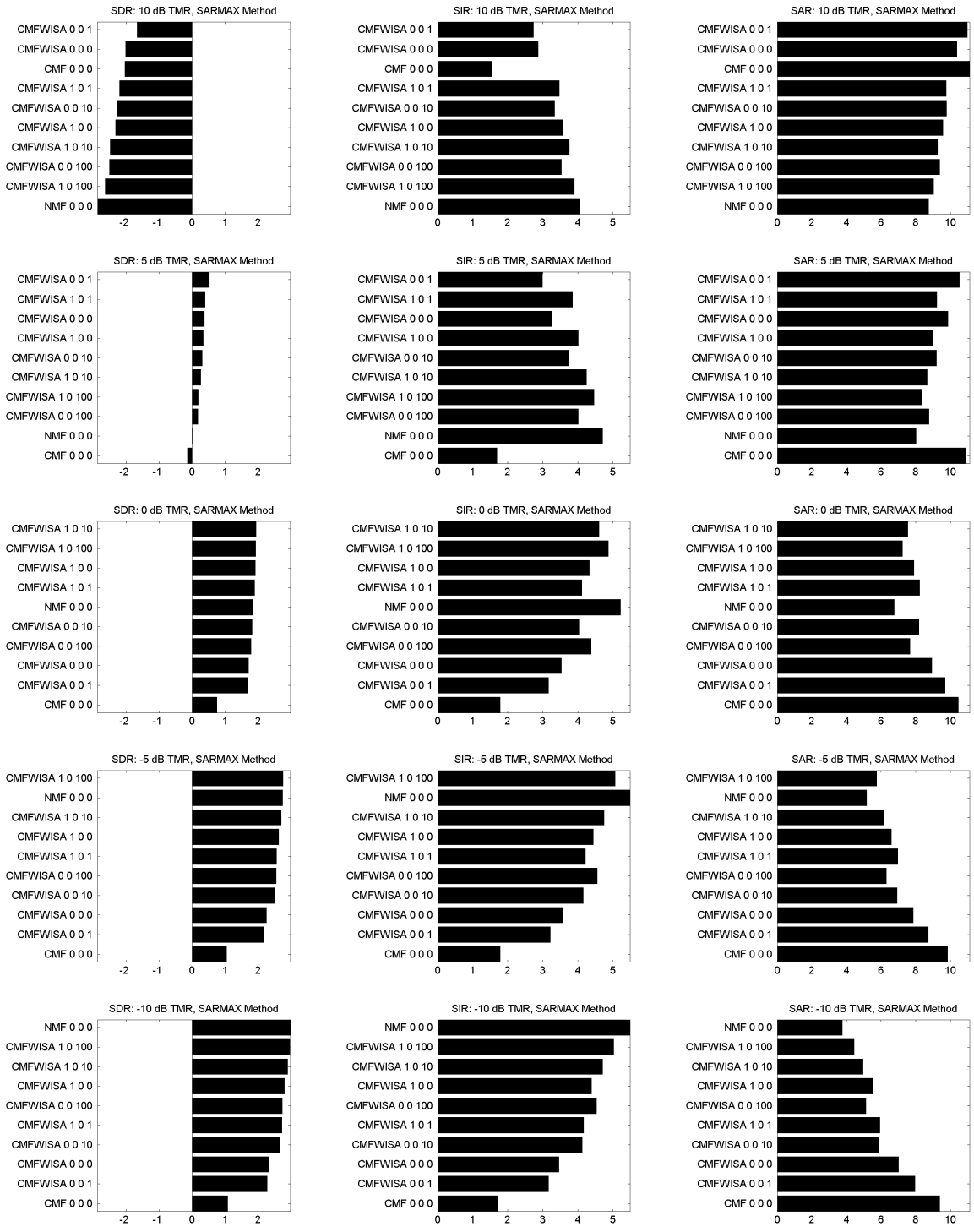

Figure 4.11: BSS Eval measurements for separated sources synthesized with the SARMAX method.

Chapter 5

COMPLEX PROBABILISTIC LATENT COMPONENT ANALYSIS (CPLCA): ENABLING NMF OF COMPLEX DATA THAT SATISFIES SUPERPOSITION

In the previous chapter, we proposed complex matrix factorization with intra-source additivity (CMFWISA), a matrix factorization model that satisfies superposition of overlapping signals. In this chapter, we will propose the complex probabilistic latent component analysis (CPLCA) algorithm, which also satisfies superposition. In addition, we will present how this algorithm can both be used on its own or combined with CMFWISA. The key step of CPLCA is transforming complex data onto a nonnegative simplex in such a way that the underlying data structure is preserved. Once the data is nonnegative, any probabilistic latent component analysis (PLCA) or nonnegative matrix factorization (NMF) method can be used on the transformed data, including source separation.

This chapter begins with presenting a method to transform complex data to a nonnegative simplex. Next, we will present three different models that can be used with the CPLCA algorithm for source separation: one with time-invariant basis vectors and two with time-varying basis vectors. We will conclude with experiments comparing nonnegative and complex CPLCA, as well as comparing CMFWISA and CPLCA.

5.1 Transforming Complex Data to Lie on a Nonnegative Simplex

In this section, we will show how to transform complex data so that it lies in the unit simplex. The unit n-simplex (or standard n-simplex) $\triangle^n$ is defined as the subset of

$\mathbb{R}^{n+1}$ given by

$$\triangle^n = \{(p_0, ..., p_n) \in \mathbb{R}^{n+1} | \sum_{i=0}^{I} p_i = 1 \text{ and } p_i \geq 0 \text{ for all } i\} \tag{5.1}$$

Since data that lies in the unit simplex have dimension positions that are nonnegative, real, and sum to one, data in the simplex can be modeled as a joint probability distribution such as PLCA [22, 56]. The nonnegativity constraint on the data also allows modeling via NMF. By transforming complex-valued data to lie in the unit simplex, we are able to perform PLCA, NMF, or other methods requiring nonnegative, real data. Although there are a variety of methods to transform complex to nonnegative data, the most common method for NMF analysis of STFT data is simply computing the magnitude of the data. This however, loses information and does not preserve the structure of the data, which breaks superposition and other conditions required for high-quality source separation. We will define preservation of the underlying data structure if the following condition is true:

$$\begin{aligned} &\text{if } X^c = B^c W, \\ &\text{then } X^n = B^n W \end{aligned} \tag{5.2}$$

X is the observed data, B is the set of basis vectors, W is the weight matrix. The "c" and "n" superscripts indicate complex and nonnegative data. If this requirement is satisfied, unsupervised CPLCA can be performed as follows:

1. Transform observed complex-valued data to lie within the unit simplex
2. Perform PLCA or NMF on data to learn basis and weight matrices, B^n and W
3. Transform nonnegative-valued basis vectors to be complex, $B^n \rightarrow B^c$
4. Use transformed complex basis vectors B^c and weights W for signal processing in the original complex domain

In order to perform supervised analysis and source separation, instead perform the following:

1. Learn complex-valued basis vectors for all sources, $B^c = [B_1^c, ..., B_N^c]$

2. Transform complex-valued basis vectors to lie within the unit simplex

3. Transform observed complex-valued data to lie within the unit simplex

4. Perform PLCA or NMF on data to learn weights W

5. Use learned complex basis vectors B^c and weights W for signal processing in the original complex domain

In sections 5.2 and 5.3, we will present different methods on how to learn the complex-valued basis vectors for each source.

In order for the unsupervised and supervised separation steps above to work, and for the conditions in eq. 5.2 to be satisfied, the CPLCA algorithm must meet two requirements. First, the algorithm must allow for transforming complex-valued to nonnegative-valued data, and vice-versa. Secondly, the value of W must remain unchanged through the transform.

Recently, an algorithm has been proposed to transform real-valued data to non-negative data lying within the unit simplex [55]. We have extended it to allow transforming complex to simplicial data. Before explaining how to extend to complex, we will first discuss how the algorithm works with real-valued data. First, the real-valued data is transformed to lie on an $N+1$-dimensional hyperplane. This is done by multiplying the data by N $N+1$-dimensional orthonormal vectors. Next, the data is translated and normalized so that it is nonnegative, real, and summable to one. The data now lie within the unit N-simplex.

In order to extend the transform to N-dimensional complex data, the data will first be transformed to a $2N$-dimensional space composed of the complex data's real and imaginary parts such that

$$C = Re(C) + i \times Im(C) \tag{5.3}$$

Alternative complex-to-real data transformations include other orthogonal decompositions such as breaking the complex data into in-phase and quadrature components. The new real-valued data $\begin{bmatrix} Re(C) \\ Im(C) \end{bmatrix}$ can then be transformed by a $2N+1$-dimensional transform as defined above. Then PLCA, NMF, or another method requiring nonnegative data can be used for analysis. In order to convert back to complex data, the $2N + 1$-dimensional nonnegative hyperplane data is inverted back to the $2N$-dimensional real-valued, Cartesian representation, and then finally converted back to the original N-dimensional complex data using the preceding equation. Figure 5.1 provides an illustration of the simplest case, converting one-dimensional complex data into three-dimensional nonnegative data lying on an unit two-simplex.

The following proofs show that although this transform is not linear in all cases, it is linear for data composed of a convex combination of basis vectors.

Given:

$V \in \mathbb{R}^{2F+1 \times 2F}$ is a matrix of zero-sum orthonormal columns [55].

$Z^c \in \mathbb{C}^{F \times T}$ is the input data, and $Z^r = \begin{bmatrix} Re(Z^c) \\ Im(Z^c) \end{bmatrix} \in \mathbb{R}^{2F \times T}$.

$Z^n \in \mathbb{R}^{2F+1 \times T}$ is the data mapped to the simplex.

$P^c \in \mathbb{C}^{F \times G}$ is a parameter for the simplex operator used to convert complex to simplicial data, and $P^r = \begin{bmatrix} Re(P^c) \\ Im(P^c) \end{bmatrix} \in \mathbb{R}^{2F \times G}$. In our applications, the size of G is either the number of basis vectors K or the number of the observed data's time frames T.

The simplex operator that can be used to transform complex to simplicial data is

$$S\{Z^c; P^c\} \triangleq \frac{VZ^r - (VP^r)_{min}}{\sum_k (VP^r - (VP^r)_{min})_{k,1}} \triangleq AZ^r + D \triangleq Z^n \tag{5.4}$$

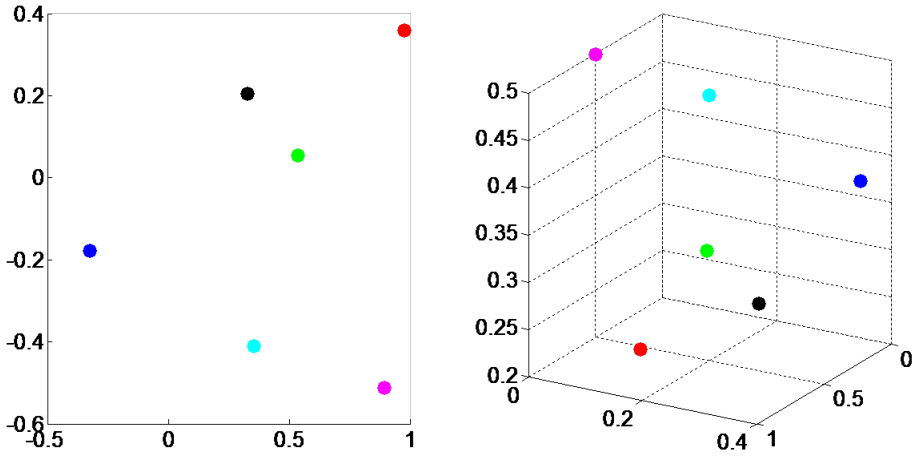

Figure 5.1: Illustration of converting one-dimensional complex data to three-dimensional nonnegative data. The one-dimensional complex data can also be viewed as two-dimensional real-valued data. The one-dimensional case was chosen because higher-dimensional data becomes difficult to display.

where

$$A = \frac{V}{\sum_k \left(VP^r - (VP^r)_{min}\right)_{k,1}} \tag{5.5}$$

$$D = \mathbf{1}_{2F+1\times T} \frac{-(VP^r)_{min}}{\sum_k \left(VP^r - (VP^r)_{min}\right)_{k,1}} = d\mathbf{1}_{2F+1\times T} \tag{5.6}$$

$(\cdot)_{min}$ is the minimum element of the matrix, the summation in the denominator is the sum of one column, and $\mathbf{1}_{2F+1\times T}$ is a matrix of 1's. The term in the denominator is a normalization term, and its summation is over one column of data. However, each column sums to the same value, so the sum can be over any one of the columns. As shown in the right-hand side of eq. 5.4, $S\{\cdot\}$ is an affine operator. Also, $P^n = S\left\{P^c; P^c\right\}$ is guaranteed to lie in the unit simplex.

Lemma 1: suppose some data $X^c = B^c W \in \mathbb{C}^{F\times T}$ with bases $B^c \in \mathbb{C}^{F\times K}$, weights $W \in \mathbb{R}_{\geq 0}^{K\times T}$, where $\sum_k W_{k,t} = 1$ for all $t \in T$, $B^n \triangleq S\left\{B^c; B^c\right\}$, and $X^n \triangleq S\left\{X^c; B^c\right\}$, then

1. $X^n = B^n W$

2. X^n lies in the unit $2F$-simplex

We will start with a proof of the first statement:

$$\begin{aligned} X^n &= S\left\{X^c; B^c\right\} \\ &= AX^c + d\mathbf{1}_{2F+1\times T} \\ &= AB^cW + d\mathbf{1}_{2F+1\times T} \\ &= AB^cW + d\mathbf{1}_{2F+1\times K}W, \text{ since } \sum_k W_{k,t} = 1 \text{ for all } t \\ &= \left(AB^c + d\mathbf{1}_{2F+1\times K}\right)W \\ &= S\left\{B^c; B^c\right\}W \\ &= B^nW \end{aligned} \tag{5.7}$$

Next, we will prove the second statement, that X^n lies in the unit simplex: Since the columns of B^n lie in the unit simplex and the columns of X^n lie in the convex hull of B^n, then the columns of X^n also lie in the unit simplex.

Next, we will look at the conditions necessary for the transform to satisfy superposition.

Lemma 2: since the operator itself is affine, it does not satisfy superposition for all possible values of W. In other words, if

$$\begin{aligned} X^n &\triangleq S\left\{X^c; B^c\right\}, \\ X_1^n &\triangleq S\left\{X_1^c; B^c\right\}, \\ X_2^n &\triangleq S\left\{X_2^c; B^c\right\}, \text{ and} \\ X^c &= X_1^c + X_2^c \end{aligned}$$

then

$$X^n \neq X_1^n + X_2^n$$

Proof:

Given:

$$X^n = S\{X^c; B^c\} \tag{5.8}$$

$$= AX^c + d\mathbf{1}_{2F+1\times T} \tag{5.9}$$

then

$$\begin{aligned} X_1^n + X_2^n &= S\{X_1^c; B^c\} + S\{X_2^c; B^c\} \\ &= AX_1^c + d\mathbf{1}_{2F+1\times T} + AX_2^c + d\mathbf{1}_{2F+1\times T} \\ &= A(X_1^c + X_2^c) + 2d\mathbf{1}_{2F+1\times T} \\ &= A(X^c) + 2d\mathbf{1}_{2F+1\times T} \\ &= X^n + d\mathbf{1}_{2F+1\times T} \end{aligned} \tag{5.10}$$

Thus

$$X^n \neq X_1^n + X_2^n \tag{5.11}$$

However, this is not problematic, as we will show that superposition is satisfied for CPLCA on an arbitrary number of overlapping signals as long as each point in the data is a convex combination of the basis vectors from all the sources.

Lemma 3: if the following conditions are true

$$\begin{aligned} X^n &\triangleq S\{X^c; B^c\} \\ X^c &= X_1^c + X_2^c \\ X^c &= B^c W \\ X_1^c &= B_1^c W_1 \\ X_2^c &= B_2^c W_2 \\ B &= \begin{bmatrix} B_1 & B_2 \end{bmatrix} \\ W &= \begin{bmatrix} W_1 \\ W_2 \end{bmatrix} \end{aligned}$$

$$K_s \text{ is the number of basis vectors for source } s$$

$$\sum_k W_{k,t} = 1 \text{ for all } t \tag{5.12}$$

then

$$X^n = S\left\{B_1^c; B^c\right\} W_1 + S\left\{B_2^c; B^c\right\} W_2 \tag{5.13}$$

Proof:

$$\begin{aligned}
&S\left\{B_1^c; B^c\right\} W_1 + S\left\{B_2^c; B^c\right\} W_2 \\
&\quad = \left(AB_1^c + d\mathbf{1}_{2F+1\times K_1}\right) W_1 + \left(AB_2^c + d\mathbf{1}_{2F+1\times K_2}\right) W_2 \\
&\quad = AB_1^c W_1 + AB_2^c W_2 + d\mathbf{1}_{2F+1\times K_1} W_1 + d\mathbf{1}_{2F+1\times K_2} W_2 \\
&\quad = A\left(B_1^c W_1 + B_2^c W_2\right) + d\mathbf{1}_{2F+1\times K} W \\
&\quad = AX^c + d\mathbf{1}_{2F+1\times T}, \text{ since } W = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix}, \text{ and } \sum_k W_{k,t} = 1 \text{ for all } t \\
&\quad = X^n
\end{aligned} \tag{5.14}$$

Thus

$$X^n = S\left\{B_1^c; B^c\right\} W_1 + S\left\{B_2^c; B^c\right\} W_2 \tag{5.15}$$

This key property allows us to use CPLCA to separate signals despite the simplex transform not satisfying superposition.

To summarize, the complex-to-unit simplex operator has the following key properties:

- The underlying data structure is preserved:
  if $X^c = B^c W$ and $\sum_k W_{k,t} = 1$ for all t, then $X^n = B^n W$

- The operator is invertible:
  if $\sum_k W_{k,t} = 1$ for all t, then $Z^c = S^{-1}\left\{S\left\{Z^c; B^c\right\}; B^c\right\}$, for any $Z^c \in \mathbb{C}^{F\times T}$

- The operator satisfies superposition for all values if W is a convex sum at all $t \in T$:

$$\text{if } X^n \triangleq S\left\{X^c; B^c\right\}, X^c = X_1^c + X_2^c,$$
$$X_1^c = B_1^c W_1, X_2^c = B_2^c W_2, \text{ and } \sum_k W_{k,t} = 1 \text{ for all } t$$

  then

$$X^n = S\left\{B_1^c; B^c\right\} W_1 + S\left\{B_2^c; B^c\right\} W_2 \tag{5.16}$$

  This key property allows us to use CPLCA to separate signals despite the simplex transform not satisfying superposition in all cases.

In Shashanka [55], the author used the proposed real-to-nonnegative simplex operator for unsupervised PLCA analysis. Performing unsupervised analysis makes the following implicit assumptions on the data:

- $W_{k,t} = 1$ for all t
- Basis vectors are time-invariant

These assumptions are appropriate for some data, but not audio STFT data. Roughly speaking, the first assumption would mean that all STFT frames would have the same energy. This would not allow for modeling a signal's dynamics. We will now discuss how to extend the CPLCA algorithm to allow for changes in energy. The second assumption will be discussed in Section 5.3, where we will propose two models for CPLCA that have time-varying basis vectors.

As shown in eq. 5.14, CPLCA only works if for each time frame, the weights sum to one. There are two modifications to the algorithm in order to ensure a convex combination of weights regardless of the frame's energy. The first addresses the case

if the summation of weights is smaller than 1, and the second addresses the case if the summation of weights is larger than 1. Traditionally, in PLCA and NMF, the basis vectors are normalized,

$$\sum_f B_{f,k} = 1 \text{ for all } k, \text{ and } B \geq 0 \text{ for all elements} \tag{5.17}$$

This solves the BW scaling ambiguity issue as well as enabling one to enforce a meaningful sparsity constraint on the weights. For nonnegative data, in order to use a convex combination of basis vectors, the data of each column must also sum to one. If the frame does not sum to one, it is trivial for it to be normalized by simply dividing the frame's components by the sum of that frame's components. In the case of real and complex data, however, this normalization is not as straightforward. The reason is that since the components are not constrained to be additive, some parts of the components may partially or completely cancel each other out when added. We will now give an example to illustrate the two necessary modifications for the supervised CPLCA algorithm. For simplicity, the example will use real-valued data, but the same methods apply to complex-valued data. Let us say that we know the following data X^r was generated by the known basis vectors B^r:

$$\begin{gathered} X^c = B^c W \\ \begin{bmatrix} .2 & .08 & .3 \\ .1 & .04 & .0 \end{bmatrix} = \begin{bmatrix} .2 & .2 \\ .4 & -.2 \end{bmatrix} \begin{bmatrix} .5 & .2 & .5 \\ .5 & .2 & 1 \end{bmatrix} \end{gathered} \tag{5.18}$$

For supervised PLCA, the goal is then to find W. The first step in the CPLCA algorithm is to convert to nonnegative data using the simplex operator. By doing so,

we find the following values:

$$S\{B^c;B^c\} = \begin{bmatrix} .6443 & .3943 \\ .3557 & .1057 \\ 0 & 0.5 \end{bmatrix}$$

$$S\{X^c;B^c\} = \begin{bmatrix} .5193 & .4077 & .5498 \\ .2307 & .2923 & .1168 \\ .25 & .3 & .3333 \end{bmatrix}$$

$$S\{B^c;B^c\}W = \begin{bmatrix} .5193 & .2077 & .7165 \\ .2307 & .0923 & .2835 \\ .25 & .1 & .5 \end{bmatrix}$$

So if the transformation succeeded, then $S\{X^c;B^c\} = S\{B^c;B^c\}W$. But we see that only the first frame's weights are correct. The reason was that the other frames did not have a convex combination of weights. In order to fix the problem where the weight combination values are too small, a basis vector of zeros is added to B^r. This allows the weight corresponding to the no-energy basis vector to assume any value without changing the original data. This allows for a convex combination of basis vectors in the simplex space.

$$X^c = B^cW$$

$$\begin{bmatrix} .2 & .08 & .3 \\ .1 & .04 & .0 \end{bmatrix} = \begin{bmatrix} .2 & .2 & 0 \\ .4 & -.2 & 0 \end{bmatrix} \begin{bmatrix} .5 & .2 & .5 \\ .5 & .2 & 1 \\ 0 & .6 & 0 \end{bmatrix} \tag{5.19}$$

$$S\{B^c;B^c\} = \begin{bmatrix} .6443 & .3943 & .3333 \\ .3557 & .1057 & .3333 \\ 0 & 0.5 & .3333 \end{bmatrix}$$

$$S\{X^c; B^c\} = \begin{bmatrix} .5193 & .4077 & .5498 \\ .2307 & .2923 & .1168 \\ .25 & .3 & .3333 \end{bmatrix}$$

$$S\{B^c; B^c\}W = \begin{bmatrix} .5193 & .4077 & .7165 \\ .2307 & .2923 & .2835 \\ .25 & .3 & .5 \end{bmatrix}$$

By adding in the basis vector of zeros, the estimation of the first frame has been fixed. So adding a frame of zeros solves the problem if $\sum_k W < 1$. In order to fix the problem occurring in the third frame, the data is simply scaled so that a convex combination of weights can be used. Again, it is not a problem if the data is scaled smaller than necessary, because this will simply result in the "zero weight", the weight corresponding to the vector of zeros, increasing to accommodate. This normalization of the third frame can then be used to scale back once the NMF or PLCA has finished and the data is transformed back the original domain. For example, scale the third frame down by 2:

$$X^c = B^c W$$

$$\begin{bmatrix} .2 & .08 & .15 \\ .1 & .04 & .0 \end{bmatrix} = \begin{bmatrix} .2 & .2 & 0 \\ .4 & -.2 & 0 \end{bmatrix} \begin{bmatrix} .5 & .2 & .25 \\ .5 & .2 & .5 \\ 0 & .6 & .25 \end{bmatrix} \tag{5.20}$$

$$S\{B^c; B^c\} = \begin{bmatrix} .6443 & .3943 & .3333 \\ .3557 & .1057 & .3333 \\ 0 & 0.5 & .3333 \end{bmatrix}$$

$$S\{X^c; B^c\} = \begin{bmatrix} .5193 & .4077 & .4416 \\ .2307 & .2923 & .2251 \\ .25 & .3 & .3333 \end{bmatrix}$$

$$S\left\{B^c; B^c\right\} W = \begin{bmatrix} .5193 & .4077 & .4416 \\ .2307 & .2923 & .2251 \\ .25 & .3 & .3333 \end{bmatrix}$$

By scaling the third frame down so that a convex combination of weights can be used for estimation, the structure is preserved. And because of the inclusion of the basis vector of zeros, it is not necessary to get the scaling perfectly, because the zero weight will allow the algorithm to accommodate for lower frame energies. Conversely, the same effect could be achieved by scaling up the basis vectors instead of scaling down the data. To summarize, the extended supervised CPLCA algorithm has the following steps:

1. Learn complex-valued basis vectors for all sources, $B^c = [B_1^c, ..., B_N^c]$

2. Add a basis vector of zeros to accommodate low-energy frames

3. Scale down high-energy data frames or scale up basis vectors to ensure a convex combination of weights

4. Transform observed complex-valued basis vectors to lie within the unit simplex using operator defined in eq. 5.4

5. Perform PLCA or NMF on data to learn weights W

6. Use learned complex basis vectors B^c and weights W for signal processing in the original complex domain.

7. Rescale data frames if they were normalized in step 3

Now that the CPLCA algorithm and complex-to-nonnegative simplex transform have been introduced, we will introduce three models that might be used within the CPLCA framework.

5.2 CPLCA with Time-Invariant Basis Vectors

The first model proposed for use with the CPLCA algorithm uses time invariant basis vectors:

$$X_{f,t} \approx \sum_k B_{f,k} W_{k,t}, \text{ where } B \in \mathbb{C}^{F \times K}, W \in \mathbb{R}_{\geq 0}^{K \times T}$$
$$\text{i.e. } X_{f,t} \approx \sum_k \hat{B}_{f,k} W_{k,t} e^{j\phi_{f,k}}, \text{ where } \hat{B} \in \mathbb{R}_{\geq 0}^{F \times K}, W \in \mathbb{R}_{\geq 0}^{K \times T} \tag{5.21}$$

Each model will be written two different ways. The first is with the phase term combined with the basis vectors, and the second is with the phase term separate from the basis vectors. Also, the three models for use with CPLCA will be described with NMF terms, but can be used with PLCA as well.

This model is most similar to the traditional NMF and PLCA models, because the basis vectors do not vary with time in either. This model can be used when the phase within data observations is known and fixed. An example of appropriate data would be in time-synchronous events, where the data observations are synchronized with the data sampling. In general, this model would not be a good fit for STFT data. For example, let us assume that a set of time-invariant basis vectors were able to perfectly represent an observed STFT. If that data were shifted slightly in time, this would result in a linear phase shift in the observed STFT, and the basis vectors would no longer be able to fit the data well.

The process for unsupervised CPLCA with in this version would be the following:

1. Transform observed complex-valued data to lie within the unit simplex using $X^n = S\{X^c, X^c\}$

2. Perform PLCA or NMF on data to learn basis and weight matrices, B^n and W

3. Transform nonnegative-valued basis vectors to be complex, $B^n \rightarrow B^c$ using $B^n = S\{B^c, X^c\}$

4. Use tranformed complex basis vectors B^c and weights W for signal processing in the original complex domain

In order to perform supervised analysis and source separation, instead perform the following:

1. Learn complex-valued basis vectors for all sources, $B^c = [B^c_1, ..., B^c_N]$

2. Transform observed complex-valued basis vectors to lie within the unit simplex using $B^n = S\{B^c, B^c\}$

3. Transform observed complex-valued data to lie within the unit simplex using $X^n = S\{X^c, B^c\}$

4. Perform PLCA or NMF on data to learn weights W

5. Use learned complex basis vectors B^c and weights W for signal processing in the original complex domain

In this first model, the steps are the same for those presented in the previous section. For the next two models, some steps will be added.

5.3 CPLCA with Time-Varying Basis Vectors

5.3.1 Basis-Dependent Phases

The second model proposed for use within the CPLCA uses time-varying basis vectors with basis-dependent phases:

$$X_{f,t} \approx \sum_k B_{f,k,t} W_{k,t},$$

$$\text{where } B \in \mathbb{C}^{F \times K}, W \in \mathbb{R}^{K \times T}_{\geq 0}, |B_{f,k,t}| = |B_{f,k,t'}| \text{ for all } k, f, t, t'$$

$$\text{i.e. } X_{f,t} \approx \sum_k \hat{B}_{f,k} W_{k,t} e^{j\phi_{f,k,t}}, \text{ where } \hat{B} \in \mathbb{R}^{F \times K}_{\geq 0}, W \in \mathbb{R}^{K \times T}_{\geq 0} \tag{5.22}$$

This model is most similar to complex matrix factorization (CMF), where a phase is estimated for each time, frequency, and basis vector. And like CMF, this model is a good fit for STFT data because it allows enough freedom within the phase to model the time-varying phase of an audio STFT, but is also similarly overparameterized, so that there are potentially an infinite number of solutions for a given set of data.

Since the basis vectors are time-varying, the CPLCA algorithm must be run on the data frame-by-frame. Therefore, the unsupervised method is not possible. In order to perform supervised analysis and source separation, perform the following:

1. Learn nonnegative-valued basis vectors for all sources, $B^n = [B_1^n, ..., B_N^n]$.

2. Initialize W and ϕ to be used for estimating observed data.

3. Pick the next time t in the observed data. If t is the final time frame, return to first frame. If the algorithm has converged sufficiently, then skip to step 9.

4. Use CMF updates to solve for $\phi_{f,k,t}$. The phase can also be solved for the entire observed signal if computational constraints permit [24]. Incorporating STFT consistency constraints is optional [32].

5. Combine phase and basis vectors to construct complex basis vectors $B_{f,k,t}^c = B_{f,k}^n e^{j\phi_{f,k,t}}$ and transform to lie within the unit simplex using $B_t^n = S\{B_t^c; B_t^c\}$.

6. Transform current frame's observed complex-valued data to lie within the unit simplex using $X_t^n = S\{X_t^c; B_t^c\}$.

7. Perform PLCA or NMF on data to learn weights for frame t, W_t.

8. Use learned complex basis vectors B_t^c and weights W_t for model estimate Y_t in the original complex domain. Return to step 3.

9. Use learned basis vectors B^n, phase ϕ, and weights W for signal processing in the original complex domain.

5.3.2 Source-Dependent Phases

The third model proposed for use within the CPLCA uses time-varying basis vectors with source-dependent phases:

$$\begin{aligned} X_{f,t} &\approx \sum_k B_{f,k,t} W_{k,t}, \text{ where } B \in \mathbb{C}^{F\times K}, W \in \mathbb{R}_{\geq 0}^{K\times T}, \\ |B_{f,k,t}| &= |B_{f,k,t'}|, phase\left(B_{f,k_s,t}\right) = \phi_{f,s,t}, \text{ for all } k, f, t, t' \\ \text{i.e. } X_{f,t} &\approx \sum_k \hat{B}_{f,k} W_{k,t} e^{j\phi_{f,s,t}}, \text{ where } \hat{B} \in \mathbb{R}_{\geq 0}^{F\times K}, W \in \mathbb{R}_{\geq 0}^{K\times T} \end{aligned} \tag{5.23}$$

This model is most similar to complex matrix factorization with intra-source additivity (CMFWISA), where a phase is estimated for each time, frequency, and source. And like CMFWISA, this model is a good fit for STFT data because it allows enough freedom within the phase to model the time-varying phase within an audio STFT. This model is a better fit for source separation than CMF, because of the same reasons CMFWISA is better than CMF, including less overparameterization, a consistent model with training and separation, faster computation time, and a smaller memory footprint.

Since the basis vectors are time-varying, the CPLCA algorithm must be run on the data frame-by-frame. Therefore, the unsupervised method is not possible. In order to perform supervised analysis and source separation, perform the following:

1. Learn nonnegative-valued basis vectors for all sources, $B^n = [B_1^n, ..., B_N^n]$.

2. Initialize W and ϕ to be used for estimating observed data.

3. Pick the next time t in the observed data. If t is the final time frame, return to first frame. If the algorithm has converged sufficiently, then skip to step 9.

4. Use CMFWISA update to solve for $\phi_{f,s,t}$. Incorporating STFT consistency constraints is optional (algorithm details found in Section 4.3. The phase can also be solved for the entire observed signal if computational constraints permit.

5. Combine phase and basis vectors to construct complex basis vectors $B^c_{f,k_s,t} = B^n_{f,k_s} e^{j\phi_{f,s,t}}$ and transform to lie within the unit simplex using $B^n_t = S\{B^c_t; B^c_t\}$.

6. Transform current frame's observed complex-valued data to lie within the unit simplex using $X^n_t = S\{X^c_t; B^c_t\}$.

7. Perform PLCA or NMF on data to learn weights for frame t, W_t.

8. Use learned complex basis vectors B^c_t and weights W_t for model estimate Y_t in the original complex domain. Return to step 3.

9. Use learned basis vectors B^n, phase ϕ, and weights W for signal processing in the original complex domain.

5.4 Experiments

In this section, we will present two CPLCA experiments. The first compares the performance of complex and nonnegative PLCA when the basis vectors are complex and known. The second compares the performance of CMFWISA and CPLCA when the phases of the original sources are known.

5.4.1 Comparing Complex and Nonnegative PLCA with Known Bases

In this experiment, we compare the performance of complex and nonnegative PLCA when the basis vectors are complex and known. We synthesized the data using a basis matrix $B \in \mathbb{C}^{20\times 10}$ and a weight matrix $W \in \mathbb{R}^{10\times 10}_{\geq 0}$ (Figure 5.2, top row). The observed mixture data was thus $X^c \in \mathbb{C}^{20\times 10}$. We divided the mixture into two

sources, the first source containing the data contributions of the first five basis vectors, and the second source containing the data of the last five basis vectors. The weights were randomly generated and did not sum to one within each time frame. The basis vectors were also random, and each had a different phase.

For this experiment, the goal was to compare separation performance between complex and nonnegative PLCA using known basis vectors. Thus, only the weights were estimated from the observed data. First, we performed complex CPLCA. Using the known basis vectors, we converted the observed complex data (Figure 5.2, bottom left) to lie in the unit simplex (Figure 5.2, bottom right). All of the figures in this section are plotted in dB. Since the basis vectors are time invariant, we used the first proposed CPLCA model (eq. 5.21). Since the data frames were not originally composed of a convex combination of basis vectors, we used the two methods described in Section 5.1 to ensure a convex combination. The two methods were adding a basis vector of zeros and scaling any data frames that were too large to make sure their summed weights were not greater than one. We then performed asymmetric PLCA on the simplex data to estimate the weights, and then synthesized estimates of the complex mixture and sources.

To perform nonnegative PLCA, we simply took the absolute value and normalized the mixture data and basis vectors, and performed PLCA with them to find the weights. We then synthesized the mixture and sources using the SIRMAX method (eq. 4.44).

We then compared the estimated weights, mixture, and sources for each method. In Figure 5.3, we see the original and estimated weights and mixture data, and in Figure 5.4, we see the original and estimated sources. In comparing the estimated complex and nonnegative values with their actual values, we see that the complex is indistinguishable from the correct value, while the estimates are somewhat similar, but significantly different. The difference is also apparent when comparing objectively. The mean squared error (MSE) of the two estimated sources for nonnegative PLCA

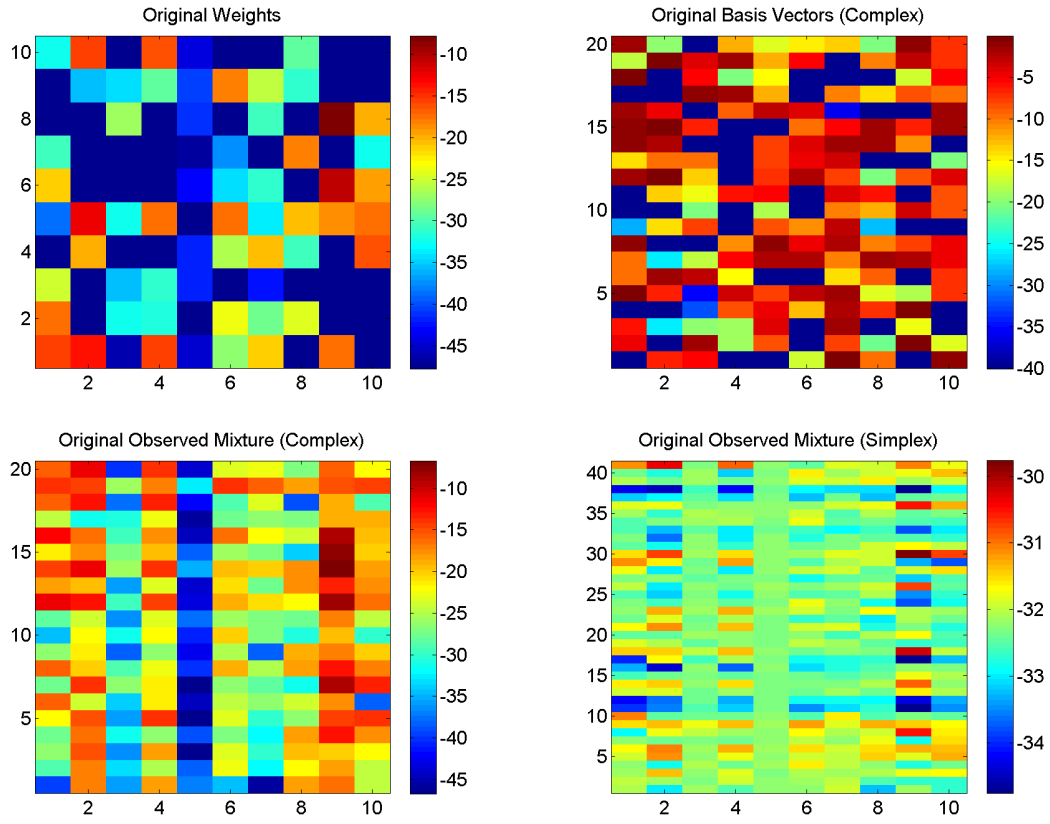

Figure 5.2: Original data. Note: All figures are in dB. "(Complex)" indicates that the figure's data is complex, and "(Simplex)" indicates the data is simplicial.

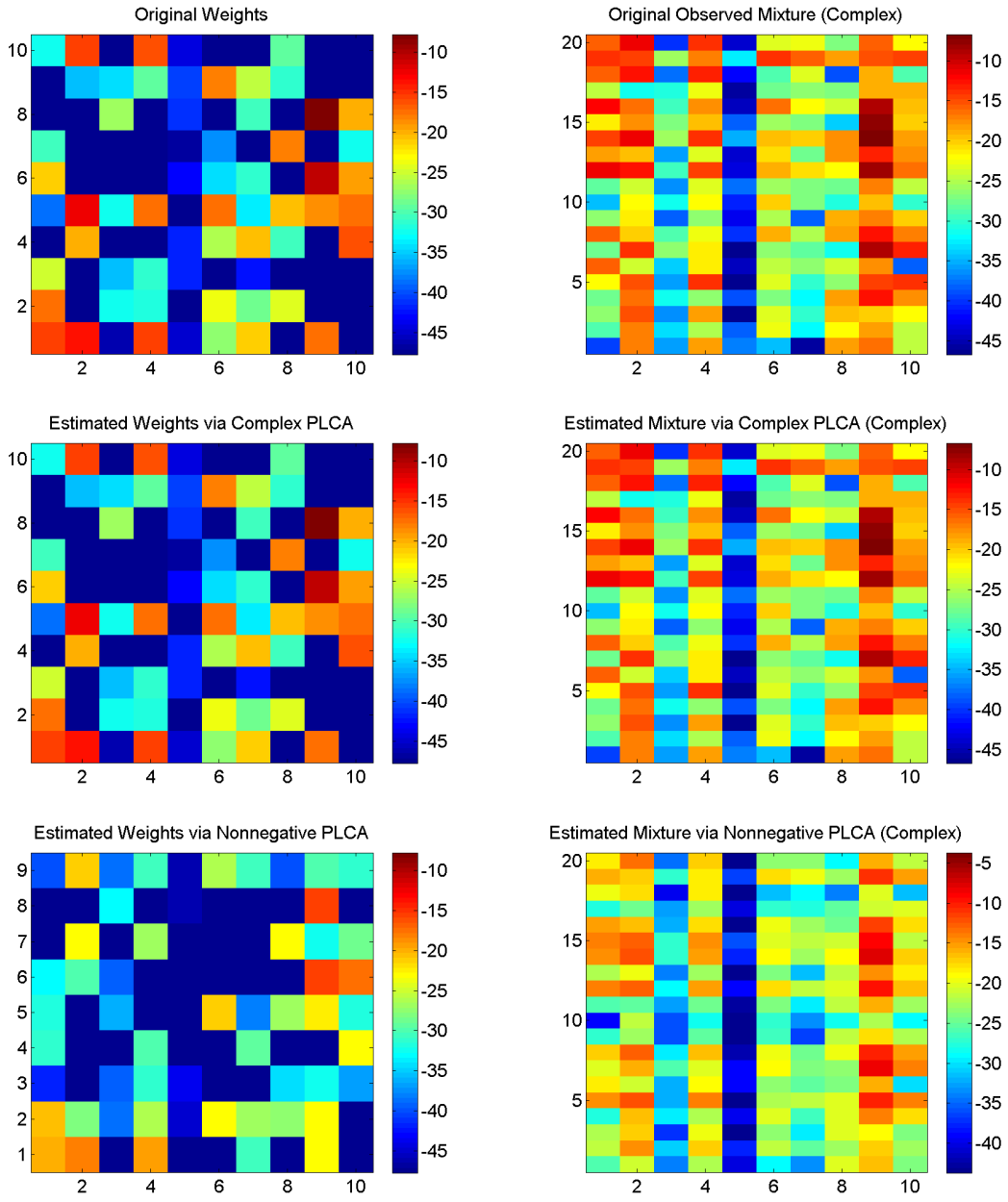

Figure 5.3: Results of complex and nonnegative PLCA. Note: All figures are in dB. "Complex" indicates that the figure's data is complex.

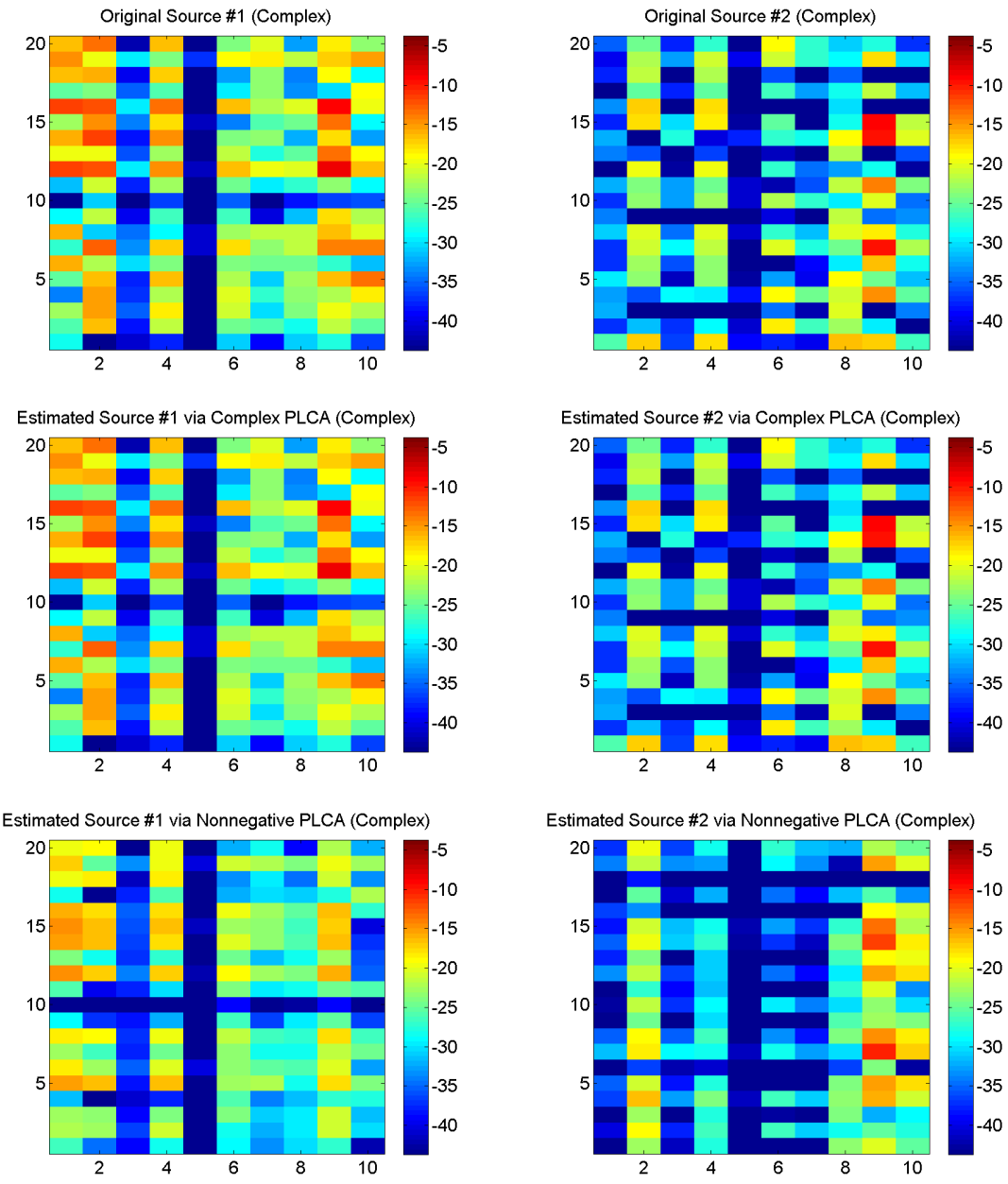

Figure 5.4: Results of complex and nonnegative PLCA. Note: All figures are in dB. "(Complex)" indicates that the figure's data is complex.

is -3.05 dB, while the MSE for the CPLCA estimates is -83.11 dB. So in this example, CPLCA is more than 80 dB better than nonnegative PLCA in source estimation and separation. The reason why the difference is so striking is because there is such a significant overlap between the two sources. This degree of overlap, though not typical in two-talker mixture, could be encountered in audio mixtures with many sources, such as a recording of a lively party, a city street, or an orchestra performance. Therefore, this is an excellent example of the advantages that satisfying superposition provides in mixture of complex data, such as audio STFT's.

5.4.2 Comparing CPLCA and CMFWISA with Known Bases

In this set of experiments, we compared the performance of CPLCA with CMFWISA on two-talker separation when the phases of the individual sources is known. They are similar to the experiments in Section 4.4.1, where we compared CMFWISA and NMF with oracle phase. In this experiment, we compare the CMFWISA results with three different CPLCA variations. For each of the three CPLCA variations, we use the third model, presented in Section 5.3.2, which has time-varying basis vectors with source-dependent phases. This model is the best fit for the audio STFT data because it allows flexibility within the phase while still keeping the number of estimated parameters down. The three different CPLCA variations were on which cost function to use in the simplex space. We used the squared Frobenius norm, the Kullback-Liebler divergence, and the Itakura-Saito divergence NMF cost functions. We wanted to see how performance varies between using different cost functions within the simplex space. And again, since the data frames were not originally composed of a convex combination of basis vectors, we used the two methods to ensure a convex combination. The two methods were adding a basis vector of zeros and scaling any data frames that were too large to make sure their summed weights were not greater than one.

We used the same supervised methods for learning the basis vectors of the sources

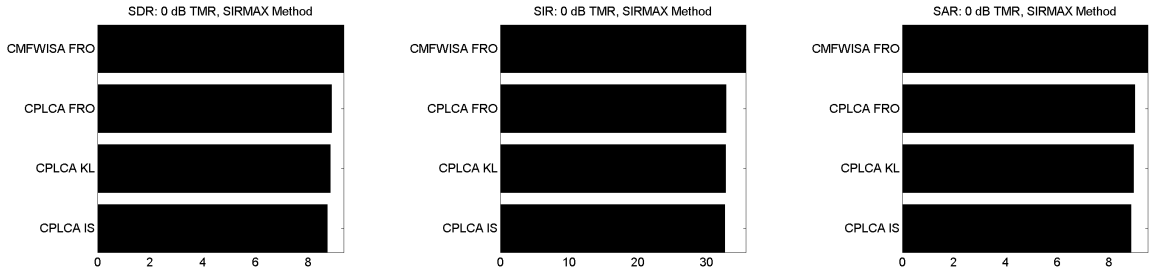

Figure 5.5: BSS Eval measurements for separated sources synthesized with the SIRMAX method.

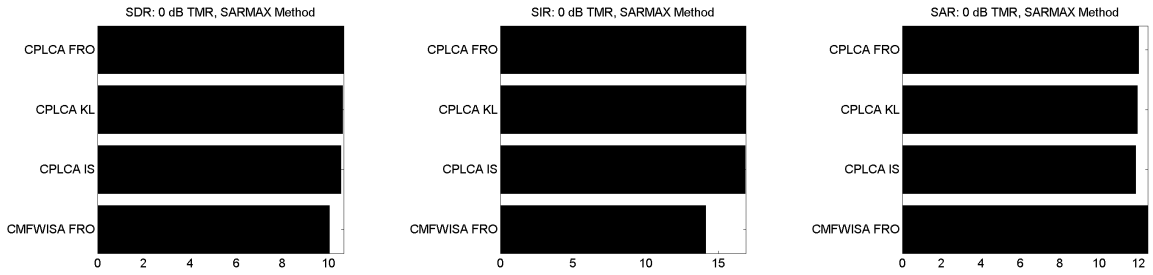

Figure 5.6: BSS Eval measurements for separated sources synthesized with the SARMAX method.

seen in Section 4.4.1 and other tests. Since this CPLCA model have time-varying phases, we estimate the model's weights frame-by-frame. In order to create the complex-valued basis vectors needed for CPLCA, we used the known phases for the sources for each frame. We then transform the frame's complex STFT data and basis vectors to the simplex space, perform NMF using the chosen cost function, and then synthesize that frame's source and mixture estimates with the estimated weights and complex basis vectors using both the SIRMAX (eq. 4.44) and SARMAX (eq. 4.47) methods. The CMFWISA results were computed the same as in Section 4.4.1.

The blind source separation evaluation (BSS Eval) results can be seen in Figures 5.5 and 5.6. There are a few key observations to be made from the results. The first is that the differences in performance between the three CPLCA variations is very slight.

Our hypothesis for this is due to the new structure of the data in the simplex space. Although the distances and relative positions are preserved in the new simplex space, the absolute positions are not. The squared Frobenius norm places more importance on fitting high-valued components, which the KL and IS divergences place more equal importance on higher and lower values. When converting complex values to lie on a simplex, a high-amplitude component will not necessarily have a high amplitude in the new simplex space. So even though the three cost functions still minimize distance in the new space, their priority in data-fitting are different in the two spaces. That may be causing their separation results to be similar. The next observation we have is that CMFWISA has the highest source-to-distortion ration (SDR) of the four in the SIRMAX synthesis set, but the lowest SDR in the SARMX synthesis set. In the SIRMAX case, CMFWISA has the highest SDR, source-to-distortion ratio (SIR), and source-to-artifact ratio (SAR). The most significant difference is in the SIR, where CMFWISA is 2.8 dB better than the CPLCA results. However, in the SARMAX synthesis case, CMFWISA has the worst SDR and SIR, but the best SAR. Again, the most significant difference is in the SIR, where CMFWISA is 2.7 dB worse than the CPLCA results.

To summarize these results, we see that the both the CMFWISA and CPLCA models perform very well in the oracle phase conditions, with CPLCA performing slightly better in the SARMAX case and slightly worse in the SIRMAX case, compared with CMFWISA. These results were consistent with our hypothesis that their performance would be similar. However, we believe we would achieve superior performance if we were to use some of the more advanced PLCA methods, such as incorporating temporal information via hidden Markov models with state-dependent dictionaries [36]. This chapter is just an introduction to CPLCA, and there is much more to be explored with it in the future.

Chapter 6

CONCLUSION

In this dissertation, we explored the topic single-channel source separation via matrix factorization. We discussed three state-of-the-art methods, nonnegative matrix factorization (NMF), probabilistic latent component analysis (PLCA), and complex matrix factorization (CMF), and presented their advantages and disadvantages. There are three major contributions in this work. The first contribution is exploring how choices in cost function, method of training, magnitude exponent of observed data, and window size affect performance (Chapter 3). The second contribution is proposing complex matrix factorization with intra-source additivity (CMFWISA), a model that has several advantages compared with previous models. For example, unlike NMF and PLCA, the model satisfies superposition, which allows for better separation performance. It also has many advantages over CMF, including less overparameterization, consistent training and analysis models, a more intuitive physical interpretation, faster computation time, and lower computational memory requirements. We also showed how we can incorporate STFT consistency constraints to be used as a regularizer to further address the overparameterization issue. We then compared performance of CMFWISA with NMF and CMF on two-talker separation. The third contribution is presenting complex probabilistic latent component analysis (CPLCA). We shows how the CPLCA algorithm can be used to transform complex data to nonnegative data lying in the unit simplex in such a way to preserve the data's structure. This method makes it possible to transform the complex data and basis vectors to lie in a nonnegative simplex, analyze the simplicial data using PLCA or NMF, and then use the weights found in the simplex analysis for processing in

the original complex domain. This method allows us to be able to apply many of the generative and probabilistic models and methods requiring nonnegative data to transformed complex data. We then conducted two experiments, one showing the significant improvements in performance over the traditional, nonnegative PLCA in separating complex overlapping data, and the second showing how CPLCA can perform similarly to CMFWISA in source separation when the phases of the sources are known.

Both CMFWISA and CPLCA were introduced for the first time in this dissertation. Although the fundamentals of each and some initial experiments were presented, there are many more questions to be answered and future topics to be studied. Some exciting future CMFWISA research topic include new methods for estimating phase, incorporating beamforming and other complementary methods in multi-channel CMFWISA, and better methods for training and choosing the best number of basis vectors. Multi-channel CMFWISA will likely yield some significant performance increases for a couple reasons. First of all, more observed data will significantly decrease estimation problems caused by overparameterization. Secondly, phase information is essential in multi-channel beamforming, so using this phase information will likely significantly increase performance over multi-channel NMF approaches that simply analyze differences in magnitude only [13]. Some future CPLCA research topics include using advanced models such as incorporating entropic priors and temporal information, enabling phase estimation within the simplex space to eliminate the need to switch between the complex and simplex spaces during phase estimation, and exploring how the different cost functions used in the simplex space affect the data fit, analysis, and separation performance in the original complex space. We hope that you have found the content interesting and have been inspired to explore the exciting, new frontiers of CMFWISA and CPLCA.

BIBLIOGRAPHY

[1] D. Bansal, B. Raj, and P. Smaragdis. Bandwidth expansion of narrowband speech using non-negative matrix factorization. In *INTERSPEECH*, 2005.

[2] L. Benaroya and F. Bimbot. Wiener based source separation with HMM/GMM using a single sensor. In *Proc. ICA*, pages 957–961, 2003.

[3] M. Berouti, R. Schwartz, and J. Makhoul. Enhancement of speech corrupted by acoustic noise. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages 208–211, 1979.

[4] S. Boll. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on Signal Processing*, 27(2):113–120, 1979.

[5] A.S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. The MIT Press, 1994.

[6] G.J. Brown and M. Cooke. Computational auditory scene analysis. *Computer Speech & Language*, 8(4):297–336, 1994.

[7] A. Cichocki, H. Lee, Y.D. Kim, and S. Choi. Non-Negative matrix factorization with α-divergence. *Pattern Recognition Letters*, 29(9):1433–1440, 2008.

[8] Pascal Clark. *Effective Coherent Modulation Filtering and Interpolation of Long Gaps in Acoustic Signals*. MS thesis, University of Washington, 2008.

[9] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21:1253–1278, 2000. ACM ID: 354398.

[10] J. Eggert and E. Korner. Sparse coding and NMF. In *IEEE International Joint Conference on Neural Networks*, volume 4, pages 2529–2533, 2004.

[11] C. Févotte, N. Bertin, and J.L. Durrieu. Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis. *Neural Computation*, 21(3):793–830, 2009.

[12] C. Févotte and J. Idier. Algorithms for nonnegative matrix factorization with the beta-divergence. *Neural Computation*, 13(3):1–24, 2010.

[13] C. Févotte and A. Ozerov. Notes on nonnegative tensor factorization of the spectrogram for audio source separation: statistical insights and towards self-clustering of the spatial cues. In *International conference on exploring music contents*, pages 102–115, 2010.

[14] D. Fitzgerald, M. Cranitch, and E. Coyle. On the use of the beta divergence for musical source separation. In *IET Irish Signals and Systems Conference*, 2009.

[15] S. Gannot, D. Burshtein, and E. Weinstein. Signal enhancement using beamforming and nonstationarity with applications to speech. *IEEE Transactions on Signal Processing*, 49(8):1614–1626, 2001.

[16] J.S. Garofolo, L.F. Lamel, W.M. Fisher, J.G. Fiscus, D.S. Pallett, and N.L. Dahlgren. *DARPA TIMIT Acoustic Phonetic Continuous Speech Corpus*. NIST, 1993.

[17] E. Gaussier and C. Goutte. Relation between PLSA and NMF and implications. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, page 601602, New York, NY, USA, 2005. ACM.

[18] D. Griffin and J. Lim. Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 32(2):236–243, 1984.

[19] M. Hayes, J. Lim, and A. Oppenheim. Signal reconstruction from phase or magnitude. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 28(6):672–680, 1980.

[20] R. Hennequin, B. David, and R. Badeau. Beta-divergence as a subclass of bregman divergence. *IEEE Signal Processing Letters*, 18(2):83 –86, February 2011.

[21] M. Hoffman. Poisson-uniform nonnegative matrix factorization. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012.

[22] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1):177–196, 2001.

[23] J.E. Jackson. *A User's Guide to Principal Components*. Wiley-Interscience, 1991.

[24] H. Kameoka, N. Ono, K. Kashino, and S. Sagayama. Complex NMF: a new sparse representation for acoustic signals. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009.

[25] B. King and L. Atlas. Single-channel source separation using simplified-training complex matrix factorization. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2010.

[26] B. King and L. Atlas. Single-Channel source separation using complex matrix factorization. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(8):2591–2597, 2011.

[27] B. King, C. Févotte, and P. Smaragdis. Optimal cost function and magnitude power for NMF-based speech separation and music interpolation. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2012.

[28] B. King, P. Smaragdis, and G.J. Mysore. Noise-Robust dynamic time warping using PLCA features. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012.

[29] J. Le Roux. *Exploiting Regularities in Natural Acoustical Scenes for Monaural Audio Signal Estimation, Decomposition, Restoration and Modification*. PhD thesis, The University of Tokyo & Université Paris VI Pierre et Marie Curie, 2009.

[30] J. Le Roux, H. Kameoka, N. Ono, A. de Cheveigne, and S. Sagayama. Single channel speech and background segregation through harmonic-temporal clustering. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 279 –282, October 2007.

[31] J. Le Roux, H. Kameoka, N. Ono, and S. Sagayama. On the interpretation of I-divergence-based distribution-fitting as a maximum-likelihood estimation problem. Technical Report METR 2008-11, The University of Tokyo, March 2008.

[32] J. Le Roux, H. Kameoka, E. Vincent, N. Ono, K. Kashino, and S. Sagayama. Complex NMF under spectrogram consistency constraints. In *Acoustical Society of Japan Autumn Meeting*, September 2009.

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

[34] D.D. Lee and H.S. Seung. Algorithms for non-negative matrix factorization. In *Neural Information Processing Systems (NIPS)*, volume 13, pages 556—562, 2001.

[35] R. McAulay and T. Quatieri. Speech analysis/synthesis based on a sinusoidal representation. *IEEE Transactions on Audio, Speech, and Signal Processing*, 34(4):744–754, 1986.

[36] G.J. Mysore. *A Non-Negative Framework for Joint Modeling of Spectral Structures and Temporal Dynamics in Sound Mixtures*. PhD thesis, Stanford University, 2010.

[37] G.J. Mysore and P. Smaragdis. Relative pitch estimation of multiple instruments. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 313–316, 2009.

[38] J. Nam, G.J. Mysore, and P. Smaragdis. Sound recognition in mixtures. In *Latent Variable Analysis and Signal Separation*, volume 7191 of *Lecture Notes in Computer Science*, pages 405–413. Springer Berlin / Heidelberg, 2012.

[39] S. Nawab, T. Quatieri, and J. Lim. Signal reconstruction from short-time fourier transform magnitude. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 31(4):986–998, 1983.

[40] A. Nehorai and B. Porat. Adaptive comb filtering for harmonic signal enhancement. *IEEE Transactions on Signal Processing*, 34(5):1124–1138, 1986.

[41] P.D. O'Grady and B.A. Pearlmutter. Convolutive non-negative matrix factorisation with a sparseness constraint. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 427–432, 2006.

[42] P.D. O'Grady and B.A. Pearlmutter. Discovering speech phones using convolutive non-negative matrix factorisation with a sparseness constraint. *Neurocomputing*, 72(1-3):88–101, 2008.

[43] A. Oppenheim and J. Lim. The importance of phase in signals. *Proceedings of the IEEE*, 69(5):529–541, 1981.

[44] M. Ostendorf, P. Price, and S. Shattuck-Hufnagel. The boston university radio corpus. *Linguistic Data Consortium*, 1996.

[45] L. Parra and C. Spence. Convolutive blind separation of non-stationary sources. *IEEE Transactions on Speech and Audio Processing*, 8(3):320–327, 2000.

[46] P. Placeway, S. Chen, M. Eskenazi, U. Jain, V. Parikh, B. Raj, M. Ravishankar, R. Rosenfeld, K. Seymore, M. Siegler, et al. The 1996 hub-4 sphinx-3 system. In *DARPA Speech Recognition Workshop*, pages 85–89, 1997.

[47] V.K. Potluru, S.M. Plis, and V.D. Calhoun. Sparse shift-invariant NMF. In *IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, pages 69–72, 2008.

[48] T. Quatieri and A. Oppenheim. Iterative techniques for minimum phase signal reconstruction from phase or magnitude. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 29(6):1187–1193, 1981.

[49] T.F. Quatieri and R.G. Danisewicz. An approach to co-channel talker interference suppression using a sinusoidal model for speech. *IEEE Transactions on Audio, Speech, and Signal Processing*, 38(1):56–69, 1990.

[50] B. Raj, R. Singh, and P. Smaragdis. Recognizing speech from simultaneous speakers. In *INTERSPEECH*, 2005.

[51] S.T. Roweis. One microphone source separation. In *Neural Information Processing Systems (NIPS)*, pages 793–799, 2000.

[52] H. Saruwatari, S. Kurita, K. Takeda, F. Itakura, T. Nishikawa, and K. Shikano. Blind source separation combining independent component analysis and beamforming. *EURASIP Journal on Applied Signal Processing*, 2003:11351146, 2003. ACM ID: 1283362.

[53] S.M. Schimmel, L. Atlas, and K. Nie. Feasibility of single channel speaker separation based on modulation frequency analysis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages 605–608, 2007.

[54] M. Shashanka. *Latent Variable Framework for Modeling and Separating Single Channel Acoustic Sources*. PhD thesis, Boston University, 2007.

[55] M. Shashanka. Simplex decompositions for real-valued datasets. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2009.

[56] M. Shashanka, B. Raj, and P. Smaragdis. Probabilistic latent variable models as nonnegative factorizations. *Computational Intelligence and Neuroscience*, pages 1–9, 2008.

[57] U.C. Shields. *Separation of added speech signals by digital comb filtering.* MS thesis, MIT, 1970.

[58] P. Smaragdis. Convolutive speech bases and their application to supervised speech separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(1):1–12, 2007.

[59] P. Smaragdis and J.C. Brown. Non-Negative matrix factorization for polyphonic music transcription. In *IEEE IEEE Workshop on Applications of Signal Processing to Audio and Acoustics WASPAA*, 2003.

[60] P. Smaragdis, B. Raj, and M. Shashanka. A probabilistic latent variable model for acoustic modeling. In *Neural Information Processing Systems (NIPS)*, 2006.

[61] P. Smaragdis, B. Raj, and M. Shashanka. Sparse and shift-invariant feature extraction from non-negative data. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 2069–2072, 2008.

[62] P. Smaragdis, B. Raj, and M. Shashanka. Missing data imputation for spectral audio signals. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2009.

[63] P. Smaragdis, M. Shashanka, and B. Raj. A sparse non-parametric approach for single channel separation of known sounds. In *Neural Information Processing Systems (NIPS)*, 2009.

[64] A. Stolcke, B. Chen, H. Franco, V.R.R. Gadde, M. Graciarena, M.Y. Hwang, K. Kirchhoff, A. Mandal, N. Morgan, X. Lei, T. Ng, M. Ostendorf, K. Sönmez, A. Venkataraman, D. Vergyri, W. Wang, J. Zheng, and Q. Zhu. Recent innovations in speech-to-text transcription at SRI-ICSI-UW. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1729–1744, 2006.

[65] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):528–537, 2010.

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

[67] T. 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, 2007.

[68] D.L. Wang. On ideal binary mask as the computational goal of auditory scene analysis. In *Speech Separation by Humans and Machines*, pages 181–197. 2005.

[69] N. Wiener. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. The MIT Press, 1964.

[70] Z. Yang and E. Oja. Unified development of multiplicative algorithms for linear and quadratic nonnegative matrix factorization. *IEEE transactions on neural networks*, 22(12):1878–1891, 2011. PMID: 22010147.

Appendix A

COMPLEX MATRIX FACTORIZATION WITH INTRA-SOURCE ADDITIVITY

A.1 Auxiliary Function

In 4.1, the CMFWISA primary and auxiliary models, $f(\theta)$ and $f^{+}(\theta,\bar{\theta})$, were presented:

$$f(\theta)=\sum_{f,t}\left|X_{f,t}-Y_{f,t}\right|^{2}+2\lambda\sum_{k,t}\left|W_{k,t}\right|^{\rho},\ \text{where} \tag{A.1}$$

$$Y_{f,t}=\sum_{s}Y_{s,f,t}\ \text{and} \tag{A.2}$$

$$Y_{s,f,t}=\sum_{k_s\in K_s}B_{k_s}W_{k_s,t}e^{j\phi_{s,f,t}} \tag{A.3}$$

$$\begin{aligned}f^{+}(\theta,\bar{\theta})=&\sum_{s,f,t}\frac{\left|\bar{X}_{s,f,t}-Y_{s,f,t}\right|^{2}}{\beta_{s,f,t}}\\&+\lambda\sum_{k,t}(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}^{2}+2\left|\bar{W}_{k,t}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho})\end{aligned} \tag{A.4}$$

where

$$\sum_{s}\beta_{s,f,t}=1,\beta_{s,f,t}>0,\ \text{for all elements in}\ \beta \tag{A.5}$$

$$\sum_{s}\bar{X}_{s,f,t}=X_{f,t} \tag{A.6}$$

$$\bar{W}\in\mathbb{R}^{K\times T} \tag{A.7}$$

and $0 < \rho < 2$. The auxiliary variables introduced are $\bar{\theta} = \{\bar{X}, \bar{W}\}$. The auxiliary variable values found to minimize the auxiliary function are

$$\bar{X}_{s,f,t} = Y_{s,f,t} + \beta_{s,f,t}(X_{f,t} - Y_{f,t}) \tag{A.8}$$

$$\bar{W}_{k,t} = W_{k,t} \tag{A.9}$$

In this section, the cost function proposed to be an auxiliary function for CMWISA will be shown to satisfy the requirement of being an auxiliary function. The requirement for a function $f^{+}(\theta, \bar{\theta})$ to be an auxiliary function of another function $f(\theta)$ is

$$\text{argmin}_{\bar{\theta}} f^{+}(\theta, \bar{\theta}) = f(\theta) \tag{A.10}$$

If a solving a primary cost function is impossible or intractable, an auxiliary function can be used to indirectly minimize the original function. It can be easily shown that minimizing an auxiliary function is guaranteed to at lead to non-increasing values of the primary function [34].

First, the requirement will be proved for $\bar{X}_{s,f,t}$.

$$\begin{aligned} f^{+}(\theta, \bar{\theta}) = & \sum_{s,f,t} \frac{|\alpha_{s,f,t} X_{f,t} - Y_{s,f,t}|^2}{\beta_{s,f,t}} \\ & + \lambda \sum_{k,t} (\rho \, |\sigma_{k,t}|^{\rho-2} \, W_{k,t}^2 + 2 \, |\sigma_{k,t}|^{\rho} - \rho \, |\sigma_{k,t}|^{\rho}) \end{aligned} \tag{A.11}$$

is an auxiliary function if

$$\begin{aligned} & \sum_{s} \alpha_{s,f,t} = 1 \\ & \sum_{s} \beta_{s,f,t} = 1, \beta_{s,f,t} > 0, \text{ for all elements in } \beta \end{aligned} \tag{A.12}$$

Now, the minimizing value for α will be shown to result in the value given for $\bar{X}_{s,f,t}$. To find the minimum value of the first term of the auxiliary function, the

following Lagrangian will be minimized:

$$\begin{aligned}\mathcal{L}[\alpha] =& \sum_s \frac{1}{\beta_{s,f,t}} |\alpha_{s,f,t} X_{f,t} - Y_{s,f,t}| - \sigma_{f,t} \left(\sum_s \alpha^*_{s,f,t} - 1 \right) \\ =& \sum_s \frac{1}{\beta_{s,f,t}} \left(\alpha^*_{s,f,t} X^*_{f,t} \alpha_{s,f,t} X_{f,t} - \alpha^*_{s,f,t} X^*_{f,t} Y_{s,f,t} - \alpha_{s,f,t} X_{f,t} Y^*_{s,f,t} + Y^*_{s,f,t} Y_{s,f,t} \right) \\ & - \sigma_{f,t} \left(\sum_s \alpha^*_{s,f,t} - 1 \right) \end{aligned} \tag{A.13}$$

Next, solve for α by setting its partial derivative to 0:

$$\begin{aligned} 0 = \frac{\delta \mathcal{L}[\alpha]}{\delta \alpha^*_{s,f,t}} &= \frac{1}{\beta_{s,f,t}} \left(X^*_{f,t} \alpha_{s,f,t} X_{f,t} - X^*_{f,t} Y_{s,f,t} \right) - \sigma_{f,t} \\ \alpha_{s,f,t} &= \frac{1}{|X_{f,t}|^2} \left(X^*_{f,t} Y_{s,f,t} + \beta_{s,f,t} \sigma_{f,t} \right) \end{aligned} \tag{A.14}$$

Since $\sum_s \alpha_{s,f,t} = 1$ and $\sum_s \beta_{s,f,t} = 1$,

$$\begin{aligned} \sum_s \alpha_{s,f,t} = 1 &= \sum_s \frac{1}{|X_{f,t}|^2} \left(X^*_{f,t} Y_{s,f,t} + \beta_{s,f,t} \sigma_{f,t} \right) \\ 1 &= \frac{1}{|X_{f,t}|^2} \left(\sigma_{f,t} + \sum_s X^*_{f,t} Y_{s,f,t} \right) \\ |X_{f,t}|^2 &= \sigma_{f,t} + \sum_s X^*_{f,t} Y_{s,f,t} \\ |X_{f,t}|^2 &= \sigma_{f,t} + X^*_{f,t} Y_{f,t} \\ \sigma_{f,t} &= |X_{f,t}|^2 - X^*_{f,t} Y_{f,t} \end{aligned} \tag{A.15}$$

Next, replace the value of σ found in eq. A.15 into A.14:

$$\alpha_{s,f,t} = \frac{1}{|X_{f,t}|^2} \left(X^*_{f,t} Y_{s,f,t} + \beta_{s,f,t} \left(|X_{f,t}|^2 - X^*_{f,t} Y_{f,t} \right) \right) \tag{A.16}$$

Finally, replace the value of α found in eq. A.16 into the first term of the auxiliary

function in eq. A.11:

$$\begin{aligned} f^{+}(Y,\bar{X}) &= \sum_{s_f,t} \frac{1}{\beta_{s,f,t}} \left| \left[\frac{1}{|X_{f,t}|^2} \left(X^*_{f,t} Y_{s,f,t} + \beta_{s,f,t} \left(|X_{f,t}|^2 - X^*_{f,t} Y_{f,t} \right) \right) \right] X_{f,t} - Y_{s,f,t} \right|^2 \\ &= \sum_{s_f,t} \frac{1}{\beta_{s,f,t}} \left| \frac{X^*_{f,t} X_{f,t} Y_{s,f,t}}{|X_{f,t}|^2} + \frac{\beta_{s,f,t} X_{f,t} |X_{f,t}|^2}{|X_{f,t}|^2} + \frac{\beta_{s,f,t} X^*_{f,t} X_{f,t} Y_{f,t}}{|X_{f,t}|^2} - Y_{s,f,t} \right|^2 \\ &= \sum_{s_f,t} \frac{1}{\beta_{s,f,t}} \left| Y_{s,f,t} + \beta_{s,f,t} X_{f,t} + \beta_{s,f,t} Y_{f,t} - Y_{s,f,t} \right|^2 \\ &= \sum_{f,t} \left| X_{f,t} + Y_{f,t} \right|^2 \end{aligned} \tag{A.17}$$

This shows that the value solved for α in eq. A.16 that minimizes the auxiliary function simplifies to the primary function. Finally, solving for $\alpha_{s,f,t} X_{f,t}$ gives the value of $\bar{X}_{s,f,t}$,

$$\begin{aligned} \alpha_{s,f,t} X_{f,t} &= \frac{1}{|X_{f,t}|^2} \left(X^*_{f,t} Y_{s,f,t} + \beta_{s,f,t} \left(|X_{f,t}|^2 - X^*_{f,t} Y_{f,t} \right) \right) X_{f,t} \\ &= \frac{X^*_{f,t} X_{f,t} Y_{s,f,t}}{|X_{f,t}|^2} + \beta_{s,f,t} \left(\frac{|X_{f,t}|^2 X_{f,t}}{|X_{f,t}|^2} - \frac{X^*_{f,t} X_{f,t} Y_{f,t}}{|X_{f,t}|^2} \right) \\ &= \frac{|X_{f,t}|^2 Y_{s,f,t}}{|X_{f,t}|^2} + \beta_{s,f,t} \left(X_{f,t} - \frac{|X_{f,t}|^2 Y_{f,t}}{|X_{f,t}|^2} \right) \\ &= Y_{s,f,t} + \beta_{s,f,t} \left(X_{f,t} - Y_{f,t} \right) \end{aligned} \tag{A.18}$$

which is the value given in eq. A.8.

Next, the minimizing value of $\bar{W}_{k,t}$ will be shown. The proof for the second term of the auxiliary function, the term relating to W sparsity, can be found in Kameoka *et al.* [24], so it will just be summarized here.

The goal is to find a quadratic function $f^{+}(W_{k,t}, \bar{W}_{k,t})$ of the form $aU^2_{k,t} + d$ that is tangent to $|W_{k,t}|^{\rho}$ at point $\bar{W}_{k,t}$. The slope of $aW^2_{k,t} + d$ at point $\bar{W}_{k,t}$ is $2a\bar{W}_{k,t}$, while the slope of $|W_{k,t}|^{\rho}$ at that same point is $\mathrm{sgn}(\bar{W}_{k,t}) \rho \left| \bar{W_{k,t}} \right|^{\rho-1}$. Solving for a and

d will provide the auxiliary function terms. First, the two slopes must be equal.

$$\begin{aligned} 2a\bar{W}_{k,t} &= \text{sgn}\left(\bar{W}_{k,t}\right)\rho\left|\bar{W}_{k,t}\right|^{p-1} \\ a &= \frac{1}{2}\text{sgn}\left(\bar{W}_{k,t}\right)\rho\left|\bar{W}_{k,t}\right|^{\rho-2} \\ a &= \frac{1}{2}\rho\left|\bar{W}_{k,t}\right|^{\rho-2}, \text{ since } \bar{W}_{k,t} > 0 \text{ for all elements} \end{aligned} \tag{A.19}$$

Second, the quadratic must pass through point $(\bar{W}_{k,t}, \left|\bar{W}_{k,t}\right|^{\rho})$,

$$\begin{aligned} \bar{W}_{k,t}^{\rho} &= \frac{1}{2}\rho\left|\bar{W}_{k,t}\right|^{\rho-2}\bar{W}_{k,t}^{2} + d \\ d &= \bar{W}_{k,t}^{\rho} - \frac{1}{2}p\left|\bar{W}_{k,t}\right|^{\rho} \end{aligned} \tag{A.20}$$

So the auxiliary function for the sparsity condition is

$$f^{+}(W,\bar{W}) = \lambda\sum_{k,t}(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}^{2} + 2\left|\bar{W}_{k,t}\right|^{\rho} - \rho\left|\bar{W}_{k,t}\right|^{\rho}) \tag{A.21}$$

and is minimized when

$$\bar{W}_{k,t} = W_{k,t} \tag{A.22}$$

A.2 Update Equations

In this section, derivations for the update equations for B_{f,k_s}, $W_{k_s,t}$, and $\phi_{s,f,t}$ are shown. The first step is to expand the norm terms into summation terms, which will make the following steps easier to follow.

$$\begin{aligned} f^{+}(\theta,\bar{\theta}) =& \sum_{s,f,t}\frac{|\bar{X}_{s,f,t} - Y_{s,f,t}|^{2}}{\beta_{s,f,t}} \\ &+\lambda\sum_{k,t}(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}^{2} + 2\left|\bar{W_{k,t}}\right|^{\rho} - \rho\left|\bar{W}_{k,t}\right|^{\rho}) \\ =& \sum_{s,f,t}\frac{\bar{X}_{s,f,t}\bar{X}_{s,f,t}^{*}}{\beta_{s,f,t}} \end{aligned}$$

$$
\begin{aligned}
&-\sum_{s,f,t}\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}}\sum_{k_s}B_{f,k_s}W_{k_s,t}e^{-j\phi_{s,f,t}}\\
&-\sum_{s,f,t}\frac{\bar{X}^*_{s,f,t}}{\beta_{s,f,t}}\sum_{k_s}B_{f,k_s}W_{k_s,t}e^{j\phi_{s,f,t}}\\
&+\sum_{s,f,t}\frac{1}{\beta_{s,f,t}}\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}e^{-j\phi_{s,f,t}}\right)\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}e^{j\phi_{s,f,t}}\right)\\
&+\lambda\sum_{k,t}(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W^2_{k,t}+2\left|\bar{W_{k,t}}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho})\\
&=\sum_{s,f,t}\frac{\bar{X}_{s,f,t}\bar{X}^*_{s,f,t}}{\beta_{s,f,t}}\\
&-2\sum_{s,f,t}\frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}}\sum_{k_s}B_{f,k_s}W_{k_s,t}\\
&+\sum_{s,f,t}\frac{1}{\beta_{s,f,t}}\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}e^{-j\phi_{s,f,t}}\right)\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}e^{j\phi_{s,f,t}}\right)\\
&+\lambda\sum_{s}\sum_{k_s}\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W^2_{k,t}+2\left|\bar{W_{k,t}}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho}\right)
\end{aligned}
\tag{A.23}
$$

$Re(\cdot)$ is the real part of the value, where $A \in C = Re(A) + j \times Im(A)$.

A.2.1 B Updates

First, the derivation for B_{f,k_s} is shown. First, the expanded auxiliary function's partial derivative is calculated.

$$
\begin{aligned}
\frac{\delta f^+(\theta,\bar{\theta})}{\delta B_{f,k_s}} &= 0\\
&-2\sum_{f}\frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}}W_{k_s,t}\\
&+2\sum_{f}\frac{W_{k_s,t}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\beta_{s,f,t}}\\
&+2\sum_{f}\frac{B_{f,k_s}W^2_{k_s,t}}{\beta_{s,f,t}}
\end{aligned}
\tag{A.24}
$$

Next, the partial derivative value is set to 0 in order to find the value of B that results in the minimum value of the auxiliary function. The terms are arranged into a multiplicative update H so that

$$B_{f,k_s}^{new} = B_{f,k_s} H_{f,k_s} \tag{A.25}$$

Assuming that the current estimate of B is nonnegative, the values of H must also be nonnegative to maintain nonnegativity in the new value of B^{new}. In order to do this, the terms of the auxiliary function's partial derivative are arranged in a fraction term so that all the terms are nonnegative. The negative values are put in the numerator with their signs reversed to positive. The positive values are put into the denominator [34]. The multiplicative update for B, then, is

$$B_{f,k_s}^{new} = B_{f,k_s} \frac{\sum_t \frac{Re(\bar{X}_{s,f,t} e^{-j\phi_{s,f,t}}) W_{k_s,t}}{\beta_{s,f,t}}}{\sum_t \frac{B_{f,k_s} W_{k_s,t}^2}{\beta_{s,f,t}} + \sum_t \frac{W_{k_s,t} \sum_{k_s' \in (K_s - k_s)} B_{f,k_s'} W_{k_s',t}}{\beta_{s,f,t}}} \tag{A.26}$$

Due to the value of ϕ found in A.2.3, the numerator is guaranteed to be nonnegative.

A.2.2 W Updates

Next, the derivation for $W_{k_s,t}$ is shown. First, calculate the expanded auxiliary function's partial derivative.

$$\begin{aligned}\frac{\delta f^{+}(\theta,\bar{\theta})}{\delta W_{k_s,t}} &= 0 \\ &\quad - 2\sum_f \frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}} B_{f,k_s} \\ &\quad + 2\sum_f \frac{B_{f,k_s}\sum_{k'_s\in\{K_s-k_s\}} B_{f,k'_s}W_{k'_s,t}}{\beta_{s,f,t}} \\ &\quad + 2\sum_f \frac{B_{f,k_s}^2 W_{k_s,t}}{\beta_{s,f,t}} \\ &\quad + 2\lambda\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}\right)\end{aligned} \tag{A.27}$$

Next, the partial derivative is similarly set to zero and its terms are similarly arranged into a multiplicative update so that

$$W_{k_s,t}^{new} = W_{k_s,t}H_{k_s,t} \tag{A.28}$$

The multiplicative update for W, then, is

$$W_{k_s,t}^{new} = W_{k_s,t}\frac{\sum_f \frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})B_{f,k_s}}{\beta_{s,f,t}}}{\sum_f \frac{B_{f,k_s}^2 W_{k_s,t}}{\beta_{s,f,t}} + \sum_f \frac{B_{f,k_s}\sum_{k'_s\in(K_s-k_s)} B_{f,k'_s}W_{k'_s,t}}{\beta_{s,f,t}} + \lambda\rho(\bar{W}_{k_s,t})^{\rho-2}W_{k_s,t}} \tag{A.29}$$

Similarly, due to the value of ϕ found in Section A.2.3, the numerator is guaranteed to be nonnegative.

A.2.3 ϕ Updates

Next, the derivation for $\phi_{s,f,t}$ is shown. First, the auxiliary function's partial derivative is calculated.

$$\begin{aligned}\frac{\delta f^{+}(\theta,\bar{\theta})}{\delta\phi_{s,f,t}} &= \sum_{k_s}\frac{1}{\beta_{s,f,t}}\bar{X}_{s,f,t}B_{f,k_s}W_{k_s,t}e^{-j\phi_{s,f,t}} - \sum_{k_s}\frac{1}{\beta_{s,f,t}}\bar{X}^{*}_{s,f,t}B_{f,k_s}W_{k_s,t}e^{j\phi_{s,f,t}} \\ &= Im\left(\frac{\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}}}{\beta_{s,f,t}}\sum_{k_s}B_{f,k_s}W_{k_s,t}\right)\end{aligned} \tag{A.30}$$

$Im(\cdot)$ is the imaginary component of a value, where $A \in C = Re(A) + j \times Im(A)$. This next step differs from the method used to update B and W. Instead of deriving multiplicative updates as before, $\phi_{s,f,t}$ can be solved directly by finding the value of $\phi_{s,f,t}$ that sets the partial derivative to 0. In order to set the imaginary value in the partial derivative to zero, the phases of the terms within the parenthesis must be either 0 or π. One will correspond to a local minimum and the other will correspond to the local maximum. In order to set the phase to zero, set $\phi_{s,f,t}$ to be the complex conjugate of the rest of the term,

$$\begin{aligned}\phi_{s,f,t} &= phase\left(\frac{\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}}}{\beta_{s,f,t}}\sum_{k_s} B_{f,k_s}W_{k_s,t}\right)\\ &= phase\left(\bar{X}_{s,f,t}\right)\end{aligned} \tag{A.31}$$

Eq. A.31 does not need a negative sign for the complex conjugate because there is a negative already on the $\phi_{s,f,t}$ term in eq. A.30.

In order to determine if this is a local maximum or minimum, the phase solution in eq. A.31 will be plugged into A.23,

$$\begin{aligned}f^{+}(\theta,\bar{\theta}) =& \sum_{s,f,t}\frac{\bar{X}_{s,f,t}\bar{X}^{*}_{s,f,t}}{\beta_{s,f,t}}\\ &-2\sum_{s,f,t}\frac{Re(\bar{X}_{s,f,t}e^{-j\left(phase\left(\bar{X}_{s,f,t}\right)\right)})}{\beta_{s,f,t}}\sum_{k_s} B_{f,k_s}W_{k_s,t}\\ &+\sum_{s,f,t}\frac{1}{\beta_{s,f,t}}\left(\sum_{k_s} B_{f,k_s}W_{k_s,t}e^{-j\left(phase\left(\bar{X}_{s,f,t}\right)\right)}\right)\left(\sum_{k_s} B_{f,k_s}W_{k_s,t}e^{j\left(phase\left(\bar{X}_{s,f,t}\right)\right)}\right)\\ &+\lambda\sum_{s}\sum_{k_s}\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W^{2}_{k,t}+2\left|\bar{W_{k,t}}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho}\right)\\ =& \sum_{s,f,t}\frac{\bar{X}_{s,f,t}\bar{X}^{*}_{s,f,t}}{\beta_{s,f,t}}\\ &-2\sum_{s,f,t}\frac{\left|\bar{X}_{s,f,t}\right|^{2}}{\beta_{s,f,t}}\sum_{k_s} B_{f,k_s}W_{k_s,t}\end{aligned}$$

$$+\sum_{s,f,t}\frac{1}{\beta_{s,f,t}}\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}\right)^2$$
$$+\lambda\sum_{s}\sum_{k_s}\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}^2+2\left|\bar{W_{k,t}}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho}\right) \tag{A.32}$$

On the other hand, if the phase was set to

$$\begin{aligned}\phi_{s,f,t}&=phase\left(\frac{\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}}}{\beta_{s,f,t}}\sum_{k_s}B_{f,k_s}W_{k_s,t}\right)+\pi\\&=phase\left(\bar{X}_{s,f,t}\right)+\pi\end{aligned} \tag{A.33}$$

then the cost function would be,

$$\begin{aligned}f^{+}(\theta,\bar{\theta})=&\sum_{s,f,t}\frac{\bar{X}_{s,f,t}\bar{X}_{s,f,t}^{*}}{\beta_{s,f,t}}\\&-2\sum_{s,f,t}\frac{Re(\bar{X}_{s,f,t}e^{-j\left(phase\left(\bar{X}_{s,f,t}\right)+\pi\right)})}{\beta_{s,f,t}}\sum_{k_s}B_{f,k_s}W_{k_s,t}\\&+\sum_{s,f,t}\frac{1}{\beta_{s,f,t}}\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}e^{-j\left(phase\left(\bar{X}_{s,f,t}\right)+\pi\right)}\right)\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}e^{j\left(phase\left(\bar{X}_{s,f,t}\right)+\pi\right)}\right)\\&+\lambda\sum_{s}\sum_{k_s}\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}^2+2\left|\bar{W_{k,t}}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho}\right)\\=&\sum_{s,f,t}\frac{\bar{X}_{s,f,t}\bar{X}_{s,f,t}^{*}}{\beta_{s,f,t}}\\&-2\sum_{s,f,t}\frac{\left|\bar{X}_{s,f,t}\right|^2e^{-j\pi}}{\beta_{s,f,t}}\sum_{k_s}B_{f,k_s}W_{k_s,t}\\&+\sum_{s,f,t}\frac{1}{\beta_{s,f,t}}\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}\right)^2\\&+\lambda\sum_{s}\sum_{k_s}\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}^2+2\left|\bar{W_{k,t}}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho}\right)\\=&\sum_{s,f,t}\frac{\bar{X}_{s,f,t}\bar{X}_{s,f,t}^{*}}{\beta_{s,f,t}}\\&+2\sum_{s,f,t}\frac{\left|\bar{X}_{s,f,t}\right|^2}{\beta_{s,f,t}}\sum_{k_s}B_{f,k_s}W_{k_s,t}\end{aligned}$$

$$+\sum_{s,f,t}\frac{1}{\beta_{s,f,t}}\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}\right)^2$$
$$+\lambda\sum_{s}\sum_{k_s}\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}^2+2\left|\bar{W_{k,t}}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho}\right) \tag{A.34}$$

The difference between using the phase of $Y_{s,f,t}$ and the phase + π is that the subtraction in the second term of the cost function (eq. A.32) becomes an addition (eq. A.34), so the phase of $Y_{s,f,t}$ as seen in eq. A.31 is correct as it clearly minimizes the auxiliary function, while A.33 maximizes the function.

Appendix B

COMPLEX MATRIX FACTORIZATION WITH INTRA-SOURCE ADDITIVITY AND STFT CONSISTENCY CONSTRAINTS

B.1 Auxiliary Function

In Section 4.3, the CMFWISA primary and auxiliary models, $f(\theta)$ and $f^{+}(\theta,\bar{\theta})$, were presented:

$$f(\theta)=\sum_{f,t}\left|X_{f,t}-Y_{f,t}\right|^{2}+2\lambda\sum_{k,t}\left|W_{k,t}\right|^{\rho}+\gamma\sum_{s,f,t}\left|C(Y_{s})_{f,t}\right|^{2},\ \text{where} \tag{B.1}$$

$$Y_{f,t}=\sum_{s}Y_{s,f,t}\ \text{and} \tag{B.2}$$

$$Y_{s,f,t}=\sum_{k_{s}\in K_{s}}B_{k_{s}}W_{k_{s},t}e^{j\phi_{s,f,t}} \tag{B.3}$$

$$\begin{aligned}f^{+}(\theta,\bar{\theta})=&\sum_{s,f,t}\frac{\left|\bar{X}_{s,f,t}-Y_{s,f,t}\right|^{2}}{\beta_{s,f,t}}\\&+\lambda\sum_{k,t}(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}^{2}+2\left|\bar{W_{k,t}}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho})\\&+\gamma\sum_{s,f,t,f',t'}\frac{1}{\delta_{s,f,t,f',t'}}\left|\bar{Z}_{s,f,t,f',t'}-A_{f,t,f',t'}\sum_{k_{s}}B_{f',k_{s}}W_{k_{s},t'}e^{j\phi_{s,f',t'}}\right|^{2}\end{aligned} \tag{B.4}$$

where

$$\sum_{s}\beta_{s,f,t}=1,\beta_{s,f,t}>0,\ \text{for all elements in}\ \beta \tag{B.5}$$

$$\sum_{f',t'}\delta_{s,f,t,f',t'}=1,\delta_{s,f,t,f',t'}>0,\ \text{for all elements in}\ \delta \tag{B.6}$$

$$\sum_{s} \bar{X}_{s,f,t} = X_{f,t} \tag{B.7}$$

$$\sum_{f',t'} \bar{Z}_{s,f,t,f',t'} = 0 \tag{B.8}$$

$$\bar{W} \in \mathbb{R}^{K \times T} \tag{B.9}$$

$A_{f,t,f',t'}$ is the matrix representation of the STFT consistency error term in eq. 4.26, so that

$$C(Y_s)_{f,t} = \sum_{f',t'} A_{f,t,f',t'} Y_{s,f',t'} \tag{B.10}$$

and $0 < \rho < 2$. The auxiliary variables introduced are $\bar{\theta} = \{\bar{X}, \bar{W}, \bar{Z}\}$. The auxiliary variable values found to minimize the auxiliary function are

$$\bar{X}_{s,f,t} = Y_{s,f,t} + \beta_{s,f,t}(X_{f,t} - Y_{f,t}) \tag{B.11}$$

$$\bar{W}_{k,t} = W_{k,t} \tag{B.12}$$

$$\bar{Z}_{s,f,t,f',t'} = A_{f,t,f',t'} Y_{s,f',t'} - \delta_{s,f,t,f',t'} C(Y_s)_{f,t} \tag{B.13}$$

In this section, the cost function proposed to be an auxiliary function for CMWISA will be shown to satisfy the requirement of being an auxiliary function. The requirement for a function $f^{+}(\theta, \bar{\theta})$ to be an auxiliary function of another function $f(\theta)$ is

$$\text{argmin}_{\bar{\theta}} f^{+}(\theta, \bar{\theta}) = f(\theta) \tag{B.14}$$

In A.1, the first and second terms of the auxiliary function were shown to meet the requirements of an auxiliary function and that $\bar{X}$ and $\bar{W}$ were their minimizing auxiliary variable values. In this section, only the new term containing $\bar{Z}$ needs to be proved to be an auxiliary function of the STFT consistency constraint.

Now, the minimizing value for $\bar{Z}$ will be derived. To find the minimum value of the auxiliary function, the following Lagrangian will be minimized:

$$\begin{aligned}\mathcal{L}[\bar{Z}_{s,f,t,f',t'}] &= \sum_s \frac{1}{\delta_{s,f,t,f',t'}} \left|\bar{Z}_{s,f,t,f',t'} - A_{f,t,f',t'} Y_{s,f',t'}\right|^2 - \sigma_{f,t}\left(\sum_s \bar{Z}^*_{s,f,t,f',t'}\right) \\ &= \sum_s \frac{1}{\delta_{s,f,t,f',t'}} (\bar{Z}^*_{s,f,t,f',t'}\bar{Z}_{s,f,t,f',t'} - \bar{Z}^*_{s,f,t,f',t'} A_{f,t,f',t'} Y_{s,f',t'} \\ &\quad - \bar{Z}_{s,f,t,f',t'} A^*_{f,t,f',t'} Y^*_{s,f',t'} + A^*_{f,t,f',t'} Y^*_{s,f',t'} A_{f,t,f',t'} Y_{s,f',t'}) \\ &\quad - \sigma_{f,t}\left(\sum_s \bar{Z}^*_{s,f,t,f',t'}\right) \end{aligned} \tag{B.15}$$

Next, solve for $\bar{Z}$ by setting its partial derivative to 0:

$$\begin{aligned} 0 = \frac{\delta \mathcal{L}[\bar{Z}_{s,f,t,f',t'}]}{\delta \bar{Z}^*_{s,f,t,f',t'}} &= \frac{1}{\delta_{s,f,t,f',t'}}\left(\bar{Z}_{s,f,t,f',t'} - A_{f,t,f',t'} Y_{s,f',t'}\right) - \sigma_{f,t} \\ \sigma_{f,t} &= \frac{1}{\delta_{s,f,t,f',t'}}\left(\bar{Z}_{s,f,t,f',t'} - A_{f,t,f',t'} Y_{s,f',t'}\right) \\ \bar{Z}_{s,f,t,f',t'} &= \sigma_{f,t}\delta_{s,f,t,f',t'} + A_{f,t,f',t'} Y_{s,f',t'} \end{aligned} \tag{B.16}$$

Since $\sum_{f',t'} \bar{Z}_{s,f,t,f',t'} = 0$ and $\sum_{f',t'} \delta_{s,f,t,f',t'} = 1$,

$$\begin{aligned} \sum_{f',t'} \bar{Z}_{s,f,t,f',t'} = 0 &= \sum_{f',t'} (\sigma_{f,t}\delta_{s,f,t,f',t'} + A_{f,t,f',t'} Y_{s,f',t'}) \\ 0 &= \sigma_{f,t} + \sum_{f',t'} A_{f,t,f',t'} Y_{s,f',t'} \\ \sigma_{f,t} &= -C(Y_s)_{f,t} \end{aligned} \tag{B.17}$$

Next, replace the value of σ found in eq. B.17 into B.16:

$$\begin{aligned} \bar{Z}_{s,f,t,f',t'} &= \delta_{s,f,t,f',t'}\left(-C(Y_s)_{f,t}\right) + A_{f,t,f',t'} Y_{s,f',t'} \\ &= A_{f,t,f',t'} Y_{s,f',t'} - \delta_{s,f,t,f',t'} C(Y_s)_{f,t} \end{aligned} \tag{B.18}$$

This shows that the minimizing value for $\bar{Z}$ is the value given in eq. B.8.

Finally, replace the value of $\bar{Z}$ into the third term of the auxiliary function in eq. B.4:

$$\begin{aligned} f^{+}(Z,\bar{Z}) &= \gamma \sum_{s,f,t,f',t'} \frac{1}{\delta_{s,f,t,f',t'}} \left|(A_{f,t,f',t'}Y_{s,f',t'} - \delta_{s,f,t,f',t'}C(Y_s)_{f,t}) - A_{f,t,f',t'}Y_{s,f',t'}\right|^2 \\ &= \gamma \sum_{s,f,t,f',t'} \frac{1}{\delta_{s,f,t,f',t'}} \left|-\delta_{s,f,t,f',t'}C(Y_s)_{f,t}\right|^2 \\ &= \gamma \sum_{s,f,t} \left|C(Y_s)_{f,t}\right|^2 \end{aligned} \tag{B.19}$$

This shows that the value solved for $\bar{Z}$ in eq. B.8 that minimizes the auxiliary function simplifies to the primary function.

B.2 Update Equations

In this section, derivations for the update equations for B_{f,k_s}, $W_{k_s,t}$, and $\phi_{s,f,t}$ are shown. The first step is to expand the norm terms into summation terms, which will make the following steps easier to follow.

$$\begin{aligned} f^{+}(\theta,\bar{\theta}) &= \sum_{s,f,t} \frac{\left|\bar{X}_{s,f,t} - Y_{s,f,t}\right|^2}{\beta_{s,f,t}} \\ &+ \lambda \sum_{k,t} (\rho \left|\bar{W}_{k,t}\right|^{\rho-2} W_{k,t}^2 + 2\left|\bar{W_{k,t}}\right|^{\rho} - \rho \left|\bar{W}_{k,t}\right|^{\rho}) \\ &+ \gamma \sum_{s,f,t,f',t'} \frac{1}{\delta_{s,f,t,f',t'}} \left|\bar{Z}_{s,f,t,f',t'} - A_{f,t,f',t'} \sum_{k_s} B_{f',k_s} W_{k_s,t'} e^{j\phi_{s,f',t'}}\right|^2 \\ &= \sum_{s,f,t} \frac{\bar{X}_{s,f,t}\bar{X}^{*}_{s,f,t}}{\beta_{s,f,t}} \\ &- \sum_{s,f,t} \frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} \sum_{k_s} B_{f,k_s} W_{k_s,t} e^{-j\phi_{s,f,t}} \\ &- \sum_{s,f,t} \frac{\bar{X}^{*}_{s,f,t}}{\beta_{s,f,t}} \sum_{k_s} B_{f,k_s} W_{k_s,t} e^{j\phi_{s,f,t}} \\ &+ \sum_{s,f,t} \frac{1}{\beta_{s,f,t}} \left(\sum_{k_s} B_{f,k_s} W_{k_s,t} e^{-j\phi_{s,f,t}}\right) \left(\sum_{k_s} B_{f,k_s} W_{k_s,t} e^{j\phi_{s,f,t}}\right) \end{aligned}$$

$$
\begin{aligned}
&+\lambda\sum_{k,t}(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}^{2}+2\left|\bar{W_{k,t}}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho})\\
&+\gamma\sum_{s,f,t,f',t'}\frac{\bar{Z}_{s,f,t,f',t'}^{*}\bar{Z}_{s,f,t,f',t'}}{\delta_{s,f,t,f',t'}}\\
&+\gamma\sum_{s,f,t,f',t'}\frac{\bar{Z}_{s,f,t,f',t'}^{*}A_{f,t,f',t'}\sum_{k_s}B_{f',k_s}W_{k_s,t'}e^{j\phi_{s,f',t'}}}{}\\
&+\gamma\sum_{s,f,t,f',t'}\frac{\bar{Z}_{s,f,t,f',t'}A_{f,t,f',t'}^{*}\sum_{k_s}B_{f',k_s}W_{k_s,t'}e^{-j\phi_{s,f',t'}}}{\delta_{s,f,t,f',t'}}\\
&+\gamma\sum_{s,f,t,f',t'}\frac{A_{f,t,f',t'}^{*}A_{f,t,f',t'}\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}e^{-j\phi_{s,f,t}}\right)\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}e^{j\phi_{s,f,t}}\right)}{\delta_{s,f,t,f',t'}}\\
&=\sum_{s,f,t}\frac{\left|\bar{X}_{s,f,t}\right|^{2}}{\beta_{s,f,t}}\\
&-2\sum_{s,f,t}\frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}}\sum_{k_s}B_{f,k_s}W_{k_s,t}\\
&+\sum_{s,f,t}\frac{1}{\beta_{s,f,t}}\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}\right)^{2}\\
&+\lambda\sum_{s}\sum_{k_s}\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}^{2}+2\left|\bar{W_{k,t}}\right|^{\rho}-\rho\left|\bar{W}_{k,t}\right|^{\rho}\right)\\
&+\gamma\sum_{s,f,t,f',t'}\frac{\left|\bar{Z}_{s,f,t,f',t'}\right|^{2}}{\delta_{s,f,t,f',t'}}\\
&+2\gamma\sum_{s,f,t,f',t'}\frac{Re\left(\bar{Z}_{s,f,t,f',t'}A_{f,t,f',t'}^{*}\sum_{k_s}B_{f',k_s}W_{k_s,t'}e^{-j\phi_{s,f',t'}}\right)}{}\\
&+\gamma\sum_{s,f,t,f',t'}\frac{\left|A_{f,t,f',t'}\right|^{2}\left(\sum_{k_s}B_{f',k_s}W_{k_s,t'}\right)^{2}}{\delta_{s,f,t,f',t'}}
\end{aligned}
\tag{B.20}
$$

B.2.1 B Updates

First, the derivation for B_{f,k_s} is shown. First, the expanded auxiliary function's partial derivative is calculated.

$$
\begin{aligned}
\frac{\delta f^{+}(\theta,\bar{\theta})}{\delta B_{f,k_s}} &= 0 \\
&\quad - 2\sum_{t}\frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}}W_{k_s,t} \\
&\quad + 2\sum_{t}\frac{W_{k_s,t}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\beta_{s,f,t}} \\
&\quad + 2\sum_{t}\frac{B_{f,k_s}W_{k_s,t}^2}{\beta_{s,f,t}} \\
&\quad + 0 \\
&\quad + 0 \\
&\quad - 2\gamma\sum_{t,f',t'}\frac{Re\left(\bar{Z}_{s,f,t,f',t'}A^{*}_{f,t,f',t'}W_{k_s,t}e^{-j\phi_{s,f',t'}}\right)}{\delta_{f,t,f',t'}} \\
&\quad + 2\gamma\sum_{t,f',t'}\frac{\left|A_{f,t,f',t'}\right|^2 W_{k_s,t}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\delta_{s,f,t,f',t'}} \\
&\quad + 2\gamma\sum_{f,f',t'}\frac{\left|A_{f,t,f',t'}\right|^2 B_{f,k_s}W_{k_s,t}^2}{\delta_{s,f,t,f',t'}} \\
&= -2\sum_{t}\frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}}W_{k_s,t} \\
&\quad + 2\sum_{t}\frac{W_{k_s,t}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\beta_{s,f,t}} \\
&\quad + 2\sum_{f}\frac{B_{f,k_s}W_{k_s,t}^2}{\beta_{s,f,t}} \\
&\quad - 2\gamma\sum_{t,f',t'}\frac{Re\left(A_{f,t,f',t'}Y_{s,f,t}A^{*}_{f,t,f',t'}W_{k_s,t}e^{-j\phi_{s,f',t'}} - (C^{H}(C(Y_s)))_{f,t}\right)}{\delta_{f,t,f',t'}} \\
&\quad + 2\gamma\sum_{t,f',t'}\frac{\left|A_{f,t,f',t'}\right|^2 W_{k_s,t}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\delta_{s,f,t,f',t'}} \\
&\quad + 2\gamma\sum_{f,f',t'}\frac{\left|A_{f,t,f',t'}\right|^2 B_{f,k_s}W_{k_s,t}^2}{\delta_{s,f,t,f',t'}}
\end{aligned}
$$

$$
\begin{aligned}
&= -2\sum_{t}\frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}}W_{k_s,t} \\
&+ 2\sum_{t}\frac{W_{k_s,t}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\beta_{s,f,t}} \\
&+ 2\sum_{t}\frac{B_{f,k_s}W_{k_s,t}^2}{\beta_{s,f,t}} \\
&- 2\gamma\sum_{t,f',t'}\frac{Re\left(A_{f,t,f',t'}Y_{s,f,t}A^*_{f,t,f',t'}W_{k_s,t}e^{-j\phi_{s,f',t'}} - (C^H(C(Y_s)))_{f,t}W_{k_s,t}\right)}{\delta_{f,t,f',t'}} \\
&+ 2\gamma\sum_{t,f',t'}\frac{|A_{f,t,f',t'}|^2 W_{k_s,t}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\delta_{s,f,t,f',t'}} \\
&+ 2\gamma\sum_{t,f',t'}\frac{|A_{f,t,f',t'}|^2 B_{f,k_s}W_{k_s,t}^2}{\delta_{s,f,t,f',t'}}
\end{aligned}
\tag{B.21}
$$

Next, the partial derivative value is set to 0 in order to find the value of B that results in the minimum value of the auxiliary function. The terms are arranged into a multiplicative update H so that

$$B^{new}_{f,k_s} = B_{f,k_s}H_{f,k_s} \tag{B.22}$$

Assuming that the current estimate of B is nonnegative, the values of H must also be nonnegative to maintain nonnegativity in the new value of B^{new}. In order to do this, the terms of the auxiliary function's partial derivative are arranged in a fraction term so that all the terms are nonnegative. The negative values are put in the numerator with their signs reversed to positive. The positive values are put into the denominator [34]. The multiplicative update for B, then, is

$$B_{f,k_s}^{new} = B_{f,k_s} \sum_t Re\left[\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + \gamma \sum_{f',t'}\left(\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}} Y_{s,f,t} - C^H(C(Y_s))_{f,t}\right)\right] W_{k_s,t}$$
$$\div\left[\sum_t \left(B_{f,k_s} W_{k_s,t}^2 + W_{k_s,t} \sum_{k'_s \in (K_s - k_s)} B_{f,k'_s} W_{k'_s,t}\right)\left(\sum_{f',t'} \frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}} + \frac{1}{\beta_{s,f,t}}\right)\right] \tag{B.23}$$

Due to the value of ϕ found in B.2.3, the numerator is guaranteed to be nonnegative.

B.2.2 W Updates

Next, the derivation for $W_{k_s,t}$ is shown. First, calculate the expanded auxiliary function's partial derivative.

$$\begin{aligned}
\frac{\delta f^+(\theta,\bar{\theta})}{\delta W_{k_s,t}} = 0 \\
&- 2\sum_f \frac{Re(\bar{X}_{s,f,t} e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}} B_{f,k_s} \\
&+ 2\sum_f \frac{B_{f,k_s} \sum_{k'_s \in \{K_s - k_s\}} B_{f,k'_s} W_{k'_s,t}}{\beta_{s,f,t}} \\
&+ 2\sum_f \frac{B_{f,k_s}^2 W_{k_s,t}}{\beta_{s,f,t}} \\
&+ 2\lambda\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2} W_{k,t}\right) \\
&+ 0 \\
&- 2\gamma \sum_{f,f',t'} \frac{Re\left(\bar{Z}_{s,f,t,f',t'} A^*_{f,t,f',t'} B_{f,k_s} e^{-j\phi_{s,f',t'}}\right)}{\delta_{f,t,f',t'}} \\
&+ 2\gamma \sum_{f,f',t'} \frac{\left|A_{f,t,f',t'}\right|^2 B_{f,k_s} \sum_{k'_s \in \{K_s - k_s\}} B_{f,k'_s} W_{k'_s,t}}{\delta_{s,f,t,f',t'}} \\
&+ 2\gamma \sum_{f,f',t'} \frac{\left|A_{f,t,f',t'}\right|^2 B_{f,k_s}^2 W_{k_s,t}}{\delta_{s,f,t,f',t'}}
\end{aligned}$$

$$
\begin{aligned}
&= -2\sum_{f}\frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}}B_{f,k_s}\\
&+ 2\sum_{f}\frac{B_{f,k_s}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\beta_{s,f,t}}\\
&+ 2\sum_{f}\frac{B^2_{f,k_s}W_{k_s,t}}{\beta_{s,f,t}}\\
&+ 2\lambda\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}\right)\\
&- 2\gamma\sum_{f,f',t'}\frac{Re\left(A_{f,t,f',t'}Y_{s,f,t}A^*_{f,t,f',t'}B_{f,k_s}e^{-j\phi_{s,f',t'}}-(C^H(C(Y_s)))_{f,t}\right)}{\delta_{f,t,f',t'}}\\
&+ 2\gamma\sum_{f,f',t'}\frac{\left|A_{f,t,f',t'}\right|^2B_{f,k_s}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\delta_{s,f,t,f',t'}}\\
&+ 2\gamma\sum_{f,f',t'}\frac{\left|A_{f,t,f',t'}\right|^2B^2_{f,k_s}W_{k_s,t}}{\delta_{s,f,t,f',t'}}\\
&= -2\sum_{f}\frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}}B_{f,k_s}\\
&+ 2\sum_{f}\frac{B_{f,k_s}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\beta_{s,f,t}}\\
&+ 2\sum_{f}\frac{B^2_{f,k_s}W_{k_s,t}}{\beta_{s,f,t}}\\
&+ 2\lambda\left(\rho\left|\bar{W}_{k,t}\right|^{\rho-2}W_{k,t}\right)\\
&- 2\gamma\sum_{f,f',t'}\frac{Re\left(A_{f,t,f',t'}Y_{s,f,t}A^*_{f,t,f',t'}B_{f,k_s}e^{-j\phi_{s,f',t'}}-(C^H(C(Y_s)))_{f,t}B_{f,k_s}\right)}{\delta_{f,t,f',t'}}\\
&+ 2\gamma\sum_{f,f',t'}\frac{\left|A_{f,t,f',t'}\right|^2B_{f,k_s}\sum_{k'_s\in\{K_s-k_s\}}B_{f,k'_s}W_{k'_s,t}}{\delta_{s,f,t,f',t'}}\\
&+ 2\gamma\sum_{f,f',t'}\frac{\left|A_{f,t,f',t'}\right|^2B^2_{f,k_s}W_{k_s,t}}{\delta_{s,f,t,f',t'}}
\end{aligned}
\tag{B.24}
$$

Next, the partial derivative is similarly set to zero and its terms are similarly

arranged into a multiplicative update so that

$$W_{k_s,t}^{new} = W_{k_s,t} H_{k_s,t} \tag{B.25}$$

The multiplicative update for W, then, is

$$\begin{aligned} W_{k_s,t}^{new} = W_{k_s,t} \sum_f Re\left[\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + \gamma \sum_{f',t'} \left(\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}} Y_{s,f,t} - C^H(C(Y_s))_{f,t}\right)\right] B_{f,k_s} \\ \div \left[\sum_f \left(B_{f,k_s}^2 W_{k_s,t} + B_{f,k_s} \sum_{k'_s \in (K_s - k_s)} B_{f,k'_s} W_{k'_s,t}\right)\left(\sum_{f',t'} \frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}} + \frac{1}{\beta_{s,f,t}}\right) \right. \\ \left. + \lambda\rho(\bar{W}_{k_s,t})^{\rho-2} W_{k_s,t}\right] \end{aligned} \tag{B.26}$$

Similarly, due to the value of ϕ found in B.2.3, the numerator is guaranteed to be nonnegative.

B.2.3 ϕ Updates

Next, the derivation for $\phi_{s,f,t}$ is shown. First, the expanded auxiliary function's partial derivative is calculated.

$$\begin{aligned} \frac{\delta f^+(\theta,\bar{\theta})}{\delta \phi_{s,f,t}} =& \sum_{k_s} \frac{1}{\beta_{s,f,t}} \bar{X}_{s,f,t} B_{f,k_s} W_{k_s,t} e^{-j\phi_{s,f,t}} \\ & - \sum_{k_s} \frac{1}{\beta_{s,f,t}} \bar{X}^*_{s,f,t} B_{f,k_s} W_{k_s,t} e^{j\phi_{s,f,t}} \\ & + \gamma \sum_{f',t'} \frac{1}{\delta_{s,f,t,f',t'}} \bar{Z}_{s,f,t,f',t'} A^*_{f,t,f',t'} B_{f,k_s} W_{k_s,t} e^{-j\phi_{s,f,t}} \\ & + \gamma \sum_{f',t'} \frac{1}{\delta_{s,f,t,f',t'}} \bar{Z}^*_{s,f,t,f',t'} A_{f,t,f',t'} B_{f,k_s} W_{k_s,t} e^{j\phi_{s,f,t}} \\ =& Im\left(\frac{\bar{X}_{s,f,t} e^{-j\phi_{s,f,t}}}{\beta_{s,f,t}} \sum_{k_s} B_{f,k_s} W_{k_s,t}\right) \\ & + Im\left(2\gamma \sum_{f',t'} \frac{1}{\delta_{s,f,t,f',t'}} \bar{Z}_{s,f,t,f',t'} A^*_{f,t,f',t'} B_{f,k_s} W_{k_s,t} e^{-j\phi_{s,f,t}}\right) \end{aligned}$$

$$
\begin{aligned}
&= Im\left(\frac{2\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}}}{\beta_{s,f,t}}\sum_{k_s}B_{f,k_s}W_{k_s,t}\right) \\
&+ Im\left[2\gamma e^{-j\phi_{s,f,t}}\left(\sum_{f',t'}\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}}Y_{s,f,t} - C^H(C(Y_s))_{f,t}\right)\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}\right)\right]
\end{aligned}
\tag{B.27}
$$

$Im(\cdot)$ is the imaginary component of a value, where $A \in C = Re(A) + j \times Im(A)$. This next step differs from the method used to update B and W. Instead of deriving multiplicative updates as before, $\phi_{s,f,t}$ can be solved directly by finding the value of $\phi_{s,f,t}$ that sets the partial derivative to 0. In order to set the imaginary value in the partial derivative to zero, the phases of the terms within the parenthesis must be either 0 or π. One will correspond to a local minimum and the other will correspond to the local maximum. In order to set the phase to zero, set $\phi_{s,f,t}$ to be the complex conjugate of the rest of the term,

$$
\begin{aligned}
\phi_{s,f,t} &= phase\left[\frac{2\bar{X}_{s,f,t}}{\beta_{s,f,t}}\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}\right)\right. \\
&\qquad \left. + 2\gamma\left(\sum_{f',t'}\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}}Y_{s,f,t} - C^H(C(Y_s))_{f,t}\right)\left(\sum_{k_s}B_{f,k_s}W_{k_s,t}\right)\right] \\
&= phase\left[\frac{2\bar{X}_{s,f,t}}{\beta_{s,f,t}} + 2\gamma\left(\sum_{f',t'}\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}}Y_{s,f,t} - C^H(C(Y_s))_{f,t}\right)\right] \\
&= phase\left[\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + \gamma\left(\sum_{f',t'}\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}}Y_{s,f,t} - C^H(C(Y_s))_{f,t}\right)\right]
\end{aligned}
\tag{B.28}
$$

Eq. B.28 does not need a negative sign for the complex conjugate because there are a negative signs already on the $\phi_{s,f,t}$ terms in eq. B.27.

In order to determine if this is a local maximum or minimum, the phase solution

in eq. B.28 will be plugged into B.20,

$$
\begin{aligned}
f^{+}(\theta,\bar{\theta}) = & \sum_{s,f,t} \frac{\left|\bar{X}_{s,f,t}\right|^2}{\beta_{s,f,t}} \\
& -2\sum_{s,f,t} \frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}} \sum_{k_s} B_{f,k_s} W_{k_s,t} \\
& +\sum_{s,f,t} \frac{1}{\beta_{s,f,t}} \left(\sum_{k_s} B_{f,k_s} W_{k_s,t}\right)^2 \\
& +\lambda \sum_{s} \sum_{k_s} \left(\rho \left|\bar{W}_{k,t}\right|^{\rho-2} W_{k,t}^2 + 2\left|\bar{W_{k,t}}\right|^{\rho} - \rho\left|\bar{W}_{k,t}\right|^{\rho}\right) \\
& +\gamma \sum_{s,f,t,f',t'} \frac{\left|\bar{Z}_{s,f,t,f',t'}\right|^2}{\delta_{s,f,t,f',t'}} \\
& +\gamma 2 \sum_{s,f,t,f',t'} \frac{Re\left(\bar{Z}_{s,f,t,f',t'} A^{*}_{f,t,f',t'} \sum_{k_s} B_{f',k_s} W_{k_s,t'} e^{-j\phi_{s,f,t}}\right)}{\delta_{s,f,t,f',t'}} \\
& +\gamma \sum_{s,f,t,f',t'} \frac{\left|A_{f,t,f',t'}\right|^2 \left(\sum_{k_s} B_{f',k_s} W_{k_s,t'}\right)^2}{\delta_{s,f,t,f',t'}}
\end{aligned}
\tag{B.29}
$$

In order to simplify the equation, all terms without *phi* can be removed, and all

common factors can be divided out to form $g^{+}(\theta,\bar{\theta})$,

$$
\begin{aligned}
g^{+}(\theta,\bar{\theta}) =& \sum_{s,f,t} \frac{Re(\bar{X}_{s,f,t}e^{-j\phi_{s,f,t}})}{\beta_{s,f,t}} \\
&+\gamma \sum_{s,f,t,f',t'} \frac{Re\left(\bar{Z}_{s,f,t,f',t'}A^{*}_{f,t,f',t'}e^{-j\phi_{s,f,t}}\right)}{\delta_{s,f,t,f',t'}} \\
&= \sum_{s,f,t} Re\left[\left(\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + \gamma \sum_{f',t'} \frac{\bar{Z}_{s,f,t,f',t'}A^{*}_{f,t,f',t'}}{\delta_{s,f,t,f',t'}}\right)e^{-j\phi_{s,f,t}}\right] \\
&= \sum_{s,f,t} Re\left[\left\{\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + \gamma \sum_{f',t'} \left(\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}}Y_{s,f,t} - C^{H}(C(Y_s))_{f,t}\right)\right\}e^{-j\phi_{s,f,t}}\right] \\
&= \sum_{s,f,t} Re\Bigg[\left\{\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + \gamma \sum_{f',t'} \left(\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}}Y_{s,f,t} - C^{H}(C(Y_s))_{f,t}\right)\right\} \\
&\quad \times e^{-j\times phase\left\{\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}}+\gamma\left(\sum_{f',t'}\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}}Y_{s,f,t}-C^{H}(C(Y_s))_{f,t}\right)\right\}}\Bigg] \\
&= -\sum_{s,f,t} \left|\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + \gamma \sum_{f',t'} \left(\frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}}Y_{s,f,t} - C^{H}(C(Y_s))_{f,t}\right)\right| \qquad \text{(B.30)}
\end{aligned}
$$

On the other hand, if the phase was set to

$$
\begin{aligned}
e^{j\phi_{s,f,t}} =& \, phase\Bigg[\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}}\left(\sum_{k_s} B_{f,k_s}W_{k_s,t}\right) \\
&+ \gamma\left(\sum_{f',t'} \frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}}Y_{s,f,t} - C^{H}(C(Y_s))_{f,t}\right)\left(\sum_{k_s} B_{f,k_s}W_{k_s,t}\right)\Bigg] \\
=& \, phase\left[\frac{\bar{X}_{s,f,t}}{\beta_{s,f,t}} + \gamma\left(\sum_{f',t'} \frac{|A_{f,t,f',t'}|^2}{\delta_{s,f,t,f',t'}}Y_{s,f,t} - C^{H}(C(Y_s))_{f,t}\right)\right] + \pi \qquad \text{(B.31)}
\end{aligned}
$$

then the subtraction in the second term of the cost function (eq. B.30) becomes an addition, so the phase found in eq. B.28 is correct as it clearly minimizes the auxiliary function, while B.31 maximizes the function.

VITA

From a very young age, Brian was fascinated with music and technology. While in kindergarten, his father bought a Commodore 64 for the family, and Brian was hooked. In second grade, after his teacher played the guitar on the first day of class, he rushed home and begged his parents for a guitar. Ten years and thousands of hours of playing the guitar and computer later, he chose to major in electrical engineering and concentrate on digital signal processing, which fused his two lifelong passions. He first attended Cal Poly in San Luis Obispo, completing his BS in electrical engineering and a minor in music technology. Excited to learn more, he continued with graduate school in the department of electrical engineering at the University of Washington. During his time in graduate school, he was advised by Les Atlas. Realizing that he enjoyed working with others, he sought out such opportunities. During his time in graduate school, he had chances to collaborate with teams at several universities and companies, from which he learned invaluable technical knowledge as well as experiencing other cultures. He completed his MS in 2008 and his PhD in 2012. After graduation, he will be joining the XBOX Kinect team at Microsoft as an audio researcher and engineer.