# Blind Source Separation of Sparse Sources with Attenuations and Delays

## A Novel Approach for the Under-Determined Case

by

Rayan Saab

B.E., The American University of Beirut, 2003

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Applied Science

in

The Faculty of Graduate Studies

(Electrical and Computer Engineering)

The University Of British Columbia

August 2005

© Rayan Saab, 2005

# Abstract

Separation of sources is an important problem in signal processing where one tries to extract two or more underlying signals from their recorded mixtures. Blind source separation is the problem of extracting sources armed only with the knowledge of the observable mixtures and necessarily, some assumptions on the underlying sources or their statistics. Applications of blind source separation abound, from EEG and fMRI in the field of neuroscience, to speech and audio recognition and separation, to face recognition, financial series analysis and communications.

In this thesis we explore blind source separation in the case where there are more sources than available mixtures, i.e. the under-determined case. We take into account both attenuations and delays in the mixing process, utilizing sparsity of the sources for demixing. We provide the theoretical framework for source separation and present simulation results to validate our method.

There are existing techniques that solve the blind source separation problem for instantaneous under-determined mixtures, and ones that solve the anechoic under-determined problem for two mixtures only. The proposed technique is novel in that it is the first to solve the blind source separation problem in a general anechoic setting where no restrictions are put on the number of mixtures, and no assumptions are made on the number of sources.

# Contents

Abstract . . . . . . . . . . . . . . . . . . . . . . . . . . . . ii

Contents . . . . . . . . . . . . . . . . . . . . . . . . . . . . iv

List of Tables . . . . . . . . . . . . . . . . . . . . . . . . . vii

List of Figures . . . . . . . . . . . . . . . . . . . . . . . . ix

Acknowledgements . . . . . . . . . . . . . . . . . . . . . . . xii

**I Thesis** **1**

**1 Introduction** . . . . . . . . . . . . . . . . . . . . . . . . 2

1.1 Prevalent Mixing Models . . . . . . . . . . . . . . . . . . 3

1.2 The BSS Story So Far . . . . . . . . . . . . . . . . . . . . 4

1.2.1 Instantaneous Mixing . . . . . . . . . . . . . . . . . 5

1.2.2 Anechoic Mixing . . . . . . . . . . . . . . . . . . . 9

1.2.3 Echoic Mixing . . . . . . . . . . . . . . . . . . . . 11

1.3 The Two-Step Approach . . . . . . . . . . . . . . . . . . . 13

1.4 Organization of the Thesis . . . . . . . . . . . . . . . . . . . . . . . 14

1.5 Contributions of the Thesis . . . . . . . . . . . . . . . . . . . . . . 15

**2 Background** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

2.1 Transform of Choice . . . . . . . . . . . . . . . . . . . . . . . . . 18

2.1.1 Fourier Transform . . . . . . . . . . . . . . . . . . . . . . . 19

2.1.2 The Short Time Fourier Transform . . . . . . . . . . . . 20

2.1.3 Examples Using the STFT . . . . . . . . . . . . . . . . . 24

2.1.4 Sparsity in the STFT Domain . . . . . . . . . . . . . . . 25

2.2 Sparsity and the $l^0$ - $l^1$ Equivalence . . . . . . . . . . . . . . . . 28

2.2.1 Mathematical Formulation . . . . . . . . . . . . . . . . . 31

2.2.2 Performance Analysis on solving $P_1$ . . . . . . . . . . . . 36

2.2.3 Section Summary and Perspective . . . . . . . . . . . . 40

2.3 K-means Clustering . . . . . . . . . . . . . . . . . . . . . . . . . 41

2.3.1 The K-means Algorithm . . . . . . . . . . . . . . . . . . 42

2.3.2 An Example of Clustering using the K-means Algorithm 43

2.4 BSS Performance Measures . . . . . . . . . . . . . . . . . . . . . 44

**3 Under-determined BSS of Anechoic Mixtures** . . . . . . . . . 49

3.1 Mixing Model and Problem Definition . . . . . . . . . . . . . . 49

3.2 Blind Mixing Model Recovery . . . . . . . . . . . . . . . . . . . 52

3.2.1 STFT Sparsity . . . . . . . . . . . . . . . . . . . . . . . . . 52

3.2.2 Feature Vector Extraction . . . . . . . . . . . . . . . . . . 52

3.2.3 Inherent Assumptions and Limitations . . . . . . . . . 55

3.3 Blind Source Extraction . . . . . . . . . . 57

3.3.1 Sparsity and $l^1$ Minimization . . . . . . . . . . 58

3.3.2 Solving $P_1$ . . . . . . . . . . 59

3.3.3 Source Separation . . . . . . . . . . 61

3.3.4 Effects of Changing the Number of Sources . . . . . . . . . . 62

3.4 Interference Suppression and Distortion Reduction . . . . . . . . . . 63

**4 Experiments and Results** . . . . . . . . . . 66

4.1 Synthetic Mixtures, Random Mixing Parameters . . . . . . . . . . 67

4.2 Synthetic Mixtures, Anechoic Room Mixing Parameters . . . . . . . . . . 74

4.3 Real Mixtures . . . . . . . . . . 76

**5 Conclusions and Future Work** . . . . . . . . . . 77

**Bibliography** . . . . . . . . . . 79

# List of Tables

1.1 The Various Blind Source Separation Problems and the Main Contributions Solving Each Case . . . . . . . . . . . . . . . . 13

2.1 Properties of the Fourier Transform . . . . . . . . . . . . . . . . 19
2.2 The Percentage of Coefficients Needed to Represent Various Percentages of the Total Signal Power . . . . . . . . . . . . . . 28
2.3 Percent of times where the extracted sources using $l^1$ minimization matches actual sources. . . . . . . . . . . . . . . . . 39
2.4 Signal to Noise Ratio (SNR) in dB of extracted sources using $l^1$ minimization. . . . . . . . . . . . . . . . . . . . . . . . . . 39
2.5 K-means cluster centers extracted using Euclidean and City-block distance measures. . . . . . . . . . . . . . . . . . . . . . 43

4.1 Mixing parameters . . . . . . . . . . . . . . . . . . . . . . . . 67
4.2 Demixing Performance with mixtures of 5 sources, $p = 0.8$, $\hat{n} = 6$ : SIR . . . . . . . . . . . . . . . . . . . . . . . . . . 71
4.3 Demixing Performance with mixtures of 5 sources, $p = 0.8$, $\hat{n} = 6$ : SAR . . . . . . . . . . . . . . . . . . . . . . . . . . 72

4.4 Demixing Performance with mixtures of 5 sources, $p = 0.8$, $\hat{n} = 6$ : SDR . . . 72

4.5 Demixing Performance with 5 mixtures of 10 sources, $p = 0.8$, $\hat{n} = 12$ . . . 73

4.6 Average Demixing Performance (60 various runs) with 2 mixtures of 3 sources, $\hat{n} = 3$ . . . 75

4.7 Average Demixing Performance (60 various runs) with 3 mixtures of 3 sources, $\hat{n} = 3$ . . . 75

4.8 Average Demixing Performance (60 various runs) with 3 mixtures of 4 sources, $\hat{n} = 4$ . . . 75

4.9 Average Demixing Performance (60 various runs) with 3 mixtures of 5 sources, $\hat{n} = 5$ . . . 75

4.10 Average Demixing Performance (60 various runs) with 3 mixtures of 6 sources, $\hat{n} = 5$ . . . 76

4.11 Demixing Performance (in dB) with 2 real mixtures of 2 sources, $p = 0.75$, $\hat{n} = 4$ . . . 76

# List of Figures

1.1 Scatter plots of two instantaneous mixtures, in the time domain (left) and in the STFT domain (right). The STFT was done with a window length of $64ms$ and an overlap of 50%. Note that the line orientations, and therefore the mixing parameters, in the figure on the right are clear. . . . . . . . . . . 7

2.1 The various sub-stages used in each of the two steps of our proposed technique. After transforming the data, feature vectors are extracted and used to estimate the mixing parameters via clustering. Both the mixing parameters and the mixtures in the transform domain are then used in the second step of the algorithm to recover the sources. . . . . . . . . . . . . . . . 17

2.2 STFT grids using two different window sizes. The one on the left has better time resolution, while the one on the right has better frequency resolution. . . . . . . . . . . . . . . . . . . . . . 23

2.3 Fourier transform magnitudes of $s_1(t)$ (Top) and $s_2(t)$ (Bottom). Note that both transforms are very similar and that all timing information is lost. . . . . . . . . . . . . . . . . . . . . . 26

2.4 STFT of $s_1(t)$ (Top) and $s_2(t)$ (Bottom). Note that the timing information is preserved, where it can be seen that at t=1, one sinusoid switches off while the other switches on. . . . . . . . . 26

2.5 Average cumulative power of the time domain signals, frequency (Fourier) domain signals and STFT of speech for window sizes of 32ms, 64ms and 128ms. The STFT with 32ms and 64ms window length exhibit a sparser representation of the data (more power in fewer coefficients). The original time domain representation and the frequency domain representation both exhibit very low sparsity. . . . . . . . . . . . . . . . 29

2.6 Average probability density of the absolute value of 50 speech signals in time domain (a), and of the absolute value of the same 50 sources in time-frequency domain (b). Note how much more sparse the TF data are. In both sub-figures, the values have been normalized to the range [0 1] . . . . . . . . . . . . . 30

2.7 Two matrices $A_1$ and $A_2$. Note that while $rank(A_1) = rank(A_2) = m = 3$, $spark(A_1) = m + 1 = 4 \neq spark(A_2) = 2$. . . . . . . . 35

2.8 Success Rate (in %) of the solution to $P_1$ capturing the correct sources when we have 8 mixtures and 10 sources. . . . . . . . 40

2.9 An example with four different 'classes' (in color) and the extracted cluster centers using $l^1$ distance (black crosses) . . . . 44

3.1 3-D view of real (crosses) and estimated (circles) parameters as recovered from the K-means clustering stage. The algorithm was run on 3 simulated mixtures of 5 sources (m=3, n=5), with the user solving for 6 sources. Note the proximity of the real to the estimated parameters. Also note the estimated source parameter that does not correspond to any real one. Displayed are the 3-dimensional normalized attenuation parameters. The delay parameters have not been included. . . 56

3.2 An example case with 2 mixtures and 3 extracted sources. The best $l^1$ decomposition as proposed in [29] and [9] is shown. . . 60

4.1 The spectrogram of one of the original sources (a), one of the mixtures (b), and the corresponding extracted source (c) from 4 mixtures of 5 sources, when the user estimates the existence of 6 sources. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 68

4.2 The original sources (a), delayed and attenuated mixtures (b),and extracted sources (c), from 4 mixtures of 5 sources, when the user estimates the existence of 6 sources. . . . . . . . 69

4.3 The Spectrogram of one of the mixtures. The TF plane of the mixture is not sparse, making the separation harder. . . . . . 73

# Acknowledgements

I would like to thank both my supervisors Dr. Rafeef Abugharbieh and Dr. Martin McKeown for their guidance and support, and for the long meetings where they had to bear my ramblings. I would also like to thank Dr. Özgür Yilmaz for his useful insights into the method proposed in this thesis. Special thanks go to Mohsen without whose headphones no audio blind source separation would have been possible. Also I am grateful to Starbucks and the Starbucks 'crew' for the many fun/wasted hours spent there. I can't forget to thank my parents, Nawal and Taan, without whom I would neither be in this world nor doing what I am doing now, and of course my brother Roham, whom I don't need to have a reason to thank. I would also like to thank my friend Hassan, for no particular reason. Last but not least, I want to thank Lina for every particular reason.

I would not like to thank the Vancouver weather for making me run after buses, or wait for them, in the cold and rain one too many times.

# Part I

# Thesis

# Chapter 1

# Introduction

Blind Source Separation (BSS) can be described as the problem of extracting underlying original sources from a number of observations where the sources are mixed together in some way. An example would be discerning what the various speakers are saying in a crowded room. Humans can do that with relative ease, being able to focus in and listen to what a particular person is saying. Machines on the other hand need sophisticated algorithms to do even a moderate job of it. Thus, solving the BSS problem usually entails making some assumptions about the underlying sources and about the mixing model. Generally, the more restrictive the assumptions and the model, the easier is the solution at the cost of the algorithm being less applicable to more general problems. Over the last few years, BSS algorithms have been developed for a wide variety of models, ranging from linear, anechoic, and echoic on one hand to over-determined, even-determined and under-determined on the other. O'grady et al. [39] provided a good literature review of the available methods in blind source separation over the range of assumptions made and models used. In the coming few pages we provide a similar review.

Assuming that we have $m$ mixtures of $n$ sources, the BSS problem is defined as overdetermined if $m > n$, even-determined if $m = n$ and underdetermined if $m < n$. Moreover, the mixing is considered instantaneous if only attenuations but no delays between the sources and the sensors are considered. The mixing is considered anechoic if both attenuations and delays are considered but no reverberations are taken into account. Finally, the echoic mixing model takes all attenuations, delays, and reverberations due to multiple paths (or echoes) into account. Naturally, as the model starts to include more parameters and as the number of sensors becomes less than the number of sources, the problem becomes harder and more involved. Finally, the models can be made more realistic by incorporating noise, usually modeled as white and gaussian [39].

## 1.1 Prevalent Mixing Models

As previously mentioned, generally three mixing models prevail in the blind source separation literature. They are as follows: [39]:

$$\text{Instantaneous: } x_i(t) = \sum_{j=1}^{n} a_{ij} s_j(t), \tag{1.1}$$

$$\text{Anechoic: } x_i(t) = \sum_{j=1}^{n} a_{ij} s_j(t - \delta_{ij}), \tag{1.2}$$

$$\text{Echoic: } x_i(t) = \sum_{k=1}^{L} \sum_{j=1}^{n} a_{ij}^{k} s_j(t - \delta_{ij}^{k}), \tag{1.3}$$

where $x_i(t)$, $i = 1, \dots, m$ is the mixture observed at the $i^{th}$ receiver, $s_j(t)$ is the $j^{th}$ source, $j = 1, \dots, n$. The instantaneous mixing model (1.1) is the simplest one taking only the attenuations $a_{ij}$ between each $j^{th}$ source and $i^{th}$ receiver into account. The anechoic model (1.2) is more realistic in most scenarios taking both the attenuations $a_{ij}$ and the delays $\delta_{ij}$ between the sources and receivers into consideration. Finally, the echoic model (1.3) considers multiple paths from sources to receivers as well. Thus, $a_{ij}^k$ and $\delta_{ij}^k$ are the attenuations and delays associated with the $j^{th}$ source, $i^{th}$ receiver and $k^{th}$ path, where up to $L$ paths are considered. In what follows, we will present a literature review of the methods used to solve each of the BSS problems, when the mixing is under-determined, even-determined, and over-determined.

## 1.2 The BSS Story So Far

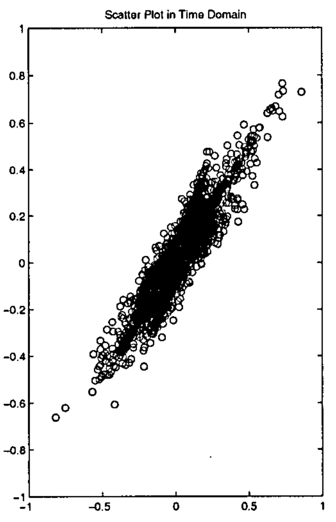
In this section we will discuss the prevalent trends in estimating the mixing parameters for the three mixing models presented. We separate this stage from the source recovery stage because only in the even-determined case is recovery of the mixing matrix sufficient for source recovery. In the under-determined case further processing will be required. However, since many approaches directed at the even-determined case perform mixing matrix estimation and source estimation simultaneously we might also allude to that in what follows.

### 1.2.1 Instantaneous Mixing

For instantaneous mixing- especially in the even-determined case- a powerful tool that has found increasing use is independent component analysis (ICA). First expressed by Jutten and Herault [23] and [24], then developed in an information maximization framework by Bell and Sejnowski [6], ICA assumes statistical independence of the sources and tries to extract $n$ sources from $n$ recorded mixtures, i.e. it assumes $m = n$. This method, from an information theoretic point of view, tries to minimize mutual information and, from a statistical signal processing point of view, tries to maximize independence to extract the mixing matrix [39] (or its inverse) by using higher-order cumulants. Moreover, rather than estimate the mixing matrix, ICA approaches generally try to estimate the unmixing matrix which leads to easier numerical computations and faster convergence [39]. Several researchers have investigated ICA using various criteria for independence. Hyvarinen and Oja [20] developed the fastICA approach which maximizes non-gaussianity as a measure of independence. Some other algorithms that rely on second order statistics [7] exist, and are based on diagonalization of a whitened covariance matrix.

To extend the ICA approach, Lewicki and Sejnowski [28], and Lee [27] then expanded it into the over-complete (under-determined) case, where there are more sources than available recorded mixtures. They use a maximum a posteriori approach to estimate the mixing matrix. The common factor in ICA approaches is the assumption of statistical independence of the sources and of a linear mixing model used to perform the separation. A very good survey of such techniques can be found in [21].

Other approaches, tackling the under-determined mixing scenario ( [9], [47], [29] and [30]) assume sparsity of the sources in some transform domain, as well as a linear mixing model to solve the BSS problem for instantaneous mixtures with more sources than mixtures. All these methods exploiting sparsity use one form or another of the following observation. Note that in equation 1.1, if all sources are zero except the $j^{th}$ one, then we will have $x_i(t) = a_{ij}s_j(t)$. In other words if we were to plot all the recorded mixtures against one another on a scatter plot, we would obtain a line for over source $s_j$ whose orientation is given by the $a_{ij}$'s. Figure 1.1 shows two scatter plots. The first one is obtained by plotting the time-domain mixtures against each other, and the second is obtained by plotting the STFT mixtures. Note that three different line orientations can be seen in the transform domain scatter plot, and what remains is to estimate the mixing parameters by identifying the line orientations. One approach that can be used to identify the lines in such a scatter plot is to use a clustering technique. For example, [47] estimates the mixing matrix as follows. They normalize their data vectors to have unit norm, and map them onto a hemisphere of the unit hyper-sphere (to avoid having two clusters for every source, one on each hemisphere). A fuzzy C-means clustering stage will follow to estimate the cluster centers which are used as columns of the mixing matrix [39]. Such approaches generally use constrained $l^1$ minimization for separation, which they prove also maximizes



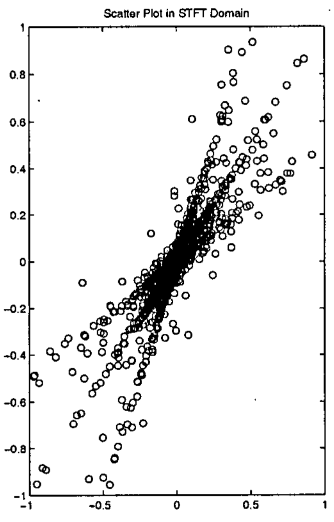


Figure 1.1: Scatter plots of two instantaneous mixtures, in the time domain (left) and in the STFT domain (right). The STFT was done with a window length of $64ms$ and an overlap of 50%. Note that the line orientations, and therefore the mixing parameters, in the figure on the right are clear.

sparsity.

Separation techniques in the instantaneous case depend on whether the system is even-determined (or over-determined) on one hand or under-determined on the other. Note that equation 1.1 in matrix notation becomes $\mathbf{x}(t) = A\mathbf{s}(t)$, where $A$ is the matrix composed of the weights $a_{ij}$. In the even-determined case, extracting estimates $\mathbf{s_{est}(t)}$ of the sources $\mathbf{s}(t)$, is done by a linear transformation: $\mathbf{s_{est}(t)} = W\mathbf{x}(t)$, where $W = A_{est}^{-1}$, and $A_{est}$ is the estimated mixing matrix. The over-determined case can be transformed to the even-determined one by applying some dimension reduction technique like Principal Component Analysis (PCA). On the other hand, separation in the under-determined mixing scenario usually relies on some sparsity assumptions. Thus, $m$-sources are extracted at each time point and the others are set to zero, effectively reducing the problem (point-wise) to an even-determined one. This is done by using constrained $l^1$ minimization, as in [9] and [30], for example.

The constrained $l^1$ minimization problem can be formulated using a probabilistic framework is as follows. Assume that we have our estimate $A_{est}$ of the mixing matrix $A$, obtained using one of the methods mentioned previously (scatter plots, clustering...), and that the sources follow a Laplacian distribution and are independent. Thus, the prior p.d.f. of $s_{est}$ would be:

$$P(s_{est}) = \frac{1}{2b} e^{-|s_{est}|/b}. \tag{1.4}$$

The problem would be to find the estimates $s_{est}$ as such, under the constraint $\mathbf{x}(t) = A_{est}\mathbf{s_{est}}(t)$:

$$\begin{aligned}
\mathbf{s_{est}} &= \arg\max_{\mathbf{s_{est}}} P(\mathbf{s_{est}}(t)|A_{est}, \mathbf{x}(\mathbf{t})) \\
&= \arg\max_{\mathbf{s_{est}}} P(\mathbf{x}(\mathbf{t})|A_{est}, \mathbf{s_{est}}(\mathbf{t}))P(\mathbf{s_{est}}(t)) \\
&= \arg\max_{\mathbf{s_{est}}} P(\mathbf{s_{est}}(t)) \\
&= \arg\max_{\mathbf{s_{est}}} e^{-\ {}_{i=1}^{n} |s_{est}|} \\
&= \arg\min_{\mathbf{s_{est}}} \sum_{i=1}^{n} |s_{est}|
\end{aligned}$$

Reintroducing the constraint the problem becomes:

$$P_1 : \text{find } \arg\min_{\mathbf{s}_{\mathbf{est}}} \|\mathbf{s}_{\mathbf{est}}\|_1, \text{ subject to } A_{est}\mathbf{s}_{\mathbf{est}} = \mathbf{x}, \tag{1.5}$$

The aim of presenting this derivation here is to show how by formulating and solving the $P_1$ problem presented above, we are assuming both sparsity and independence of the sources. Methods like the ones presented in [9] and [30] separate the sources in the under-determined instantaneous case by solving $P_1$. Later on, we shall encounter a complex version of the $P_1$ problem, but as an approximation to a harder non-convex problem, which we formulate to solve for sources in the under-determined *anechoic* scenario.

### 1.2.2 Anechoic Mixing

There are several approaches that deal with both attenuation and delays in the case of more sources than mixtures. Anemuller, Sejnowski, and Makeig [3] proposed a complex independent component analysis technique, which extracts an equal number of sources from mixtures in each of various separate spectral bands to solve the BSS problem for electroencephalographic data. However, this entails identifying whether sources extracted from different spectral bands correspond to each other or not, and may also involve solving a permutation problem. Jourjine et al [22], and Rickard and Yilmaz [45] developed an algorithm called the Degenerate Unmixing Estimation Technique (DUET) that exploits sparsity in the Short Time Fourier Transform

(STFT) domain, and uses masking to extract multiple sources from only two mixtures. The assumption, which they call *W-Disjoint Orthogonality*, is that in the time-frequency (TF) domain, only one source is active at every point. Therefore, what remains is to assign every TF point to its respective source and transform the signals back to the time domain. To that end, a two-dimensional histogram of relative attenuations and delays between the two mixtures is constructed. Due to the approximate *W-Disjoint Orthogonality* of the sources, several peaks, each corresponding to one source will appear in the histogram. Detecting those peaks is thus equivalent to estimating the mixing parameters. Next, TF points are assigned to their closest peaks, and a TF mask is constructed for each source. For a certain source, if the mask value is 1 at a certain TF point, the value of that TF point is set to the value of the corresponding point in one of the mixtures. The process is repeated for all the sources in the TF domain, and finally the extracted sources are transformed back into the time domain.

Some other algorithms that rely on similar concepts for separation of multiple sources from *two* mixtures were developed as well. For example [8], uses ideas from the instantaneous case to estimate the mixing parameters. A scatter plot technique is used to approximate the *attenuations* by estimating line orientations using a kernel density method. The method works by dividing the scatter plot in a radial grid from 0 to $\pi$ radians, and assigning weights to the bins on the grid based on the proximity of the data points to the bin directions. Combining the results from all the bins on the grid yields a kernel density function whose local maxima correspond to the attenuations of the mixing process. Once that is done the delays are estimated using the real and imaginary values of the STFT points. Delay values for a certain source are iteratively changed until the kernel function is maximized. The procedure is repeated for all the sources and this way the delay parameters are estimated as well. After the amplitudes and delays are extracted, Bofill uses *second order cone programming*, a technique that allows for $l^1$ minimization in the complex domain, to recover the sources in the TF plane. Finally, the sources are transformed back into the time domain.

Also, [11] proposes a method that uses a minimum mean square error estimator in the frequency domain to estimate the signal spectra from noisy observations and then applies K-means clustering to identify the sources. This method also uses simple masking based on the assumption that only one source is active at every time-frequency point to perform the separation, and is very similar to DUET in that sense.

Despite the fact that DUET and the approaches of [8] and [11] are confined to the case of two mixtures ($m = 2$), they open the door for algorithms such as the one we present in this thesis to solve the problem of anechoic mixing even when we have more than two mixtures available.

### 1.2.3 Echoic Mixing

The most involved of the three mixing models discussed in section 1.1, echoic mixing is a challenging BSS problem. One of the earliest developments that made attempts at this possible is the PhD thesis of Lambert [26] and subsequently [25] that built an 'FIR matrix algebra' and used it for blind separation of convolutive mixtures. Lambert provided methods for inverting FIR filters, and used them to determine the inverse of the mixing process. Generally, estimating the inverse mixing process is done in the Fourier (frequency) domain, where convolutions are transformed to multiplications [39]. This transforms the problem into the instantaneous mixing problem at every frequency cell. One then has to solve multiple instantaneous mixing problems. This idea was used by Smaragdis [42], who extended the Infomax approach of Bell and Sejnowski [6] to complex data allowing for separation of convolved mixtures using a powerful statistical approach to be accomplished. On the other hand this introduces the problems of scaling and permutation. A source component extracted at a particular frequency might not be extracted in the same bin at another frequency, and it might be scaled differently. Thus, techniques have to be used to match the various sources across frequency bins. Errors in solving the permutation and scaling problems could lead to interference between extracted sources and artifacts within each source, respectively.

We will not go much further into the literature of echoic mixing BSS since all available algorithms currently are unable to solve the under-determined problem.

A brief overview of the work that has been done in the various problems associated with blind source separation by various researchers is presented in table 1.1. It shows that for the problem of under-determined BSS in the

Table 1.1: The Various Blind Source Separation Problems and the Main Contributions Solving Each Case

| **Number of Mixtures** | **Mixing Model** | | |
|---|---|---|---|
| | Instantaneous | Anechoic | Echoic |
| over/even-determined ($m \leq n$) | [23], [6], [21], [7],... | [22], [45], [8],... | [26], [42] |
| under-determined ($m = 2$) | [32], [27], [9], [46], [30],... | [22], [45], [8],... | |
| under-determined (any $m \geq 2$) | [27], [9], [46], [30],... | Our contribution | |

anechoic case, solutions have only been proposed for the two-sensor case, while no solutions have been proposed for the under-determined echoic case.

The contribution presented in this theses is a technique that solves the under-determined anechoic BSS problem without restricting the number of mixtures to two. We call this technique the Blind Anechoic Under-determined Source Separation technique (BAUSS).

## 1.3 The Two-Step Approach

An important contribution by Theis and Lang [43] formalizes a two step approach to over-complete (under-determined) blind source separation. They assert that for solving the under-determined BSS problem, recognizing the mixing parameters is insufficient, and further processing is needed to extract the sources. The solution to the problem is thus in two steps, the first of which is blind mixing model recovery and the second is blind source recovery. In [9] and [30] such an approach is used, whereas [28] fuses the two steps into one.

In our work, we follow the two step approach. We illustrate a demixing algorithm for anechoic mixtures with delays, utilizing the STFT as our transform domain, and perform separation using a geometric interpretation of $l^1$ minimization.

## 1.4 Organization of the Thesis

Following the introduction, we provide a brief overview of the mathematics and the algorithms we are going to use throughout. We discuss the STFT and its properties, as well as constrained $l^1$ minimization followed by an overview of K-means clustering. We then discuss the metrics we use to evaluate the performance of the BSS algorithm. Next, we discuss the proposed method for solving the anechoic BSS problem in the under-determined case. We begin with a mathematical formalization of the problem. We then explain how we utilize sparsity of mixtures in the STFT domain to extract feature vectors. We describe our feature vectors and the reasoning behind selecting them, combining the feature vectors used in [45], [9] and [30]. We show how by using clustering in the feature space we are able to extract the mixing model parameters. Next, we explain our algorithm for recovering the unknown sources by capitalizing on the mixing parameters we extracted and the sparsity of the sources in the transform domain. We show how this step is a generalization of the $l^1$ minimization procedure employed in [9] and [30] to the case where delays are included in the mixing model. We also present an enhancement of the algorithm outlined above aimed at suppressing interference and reducing distortions caused by errors in estimating the mixing matrix, failure of the sparsity assumption, or high levels of noise. Finally, we report the results we obtained running the algorithm on both synthetic and real mixtures and we present our conclusions and outline directions for future work.

## 1.5 Contributions of the Thesis

One can see from table 1.1 that the existing techniques for under-determined blind source separation are restricted to the instantaneous case and to the anechoic case with two mixtures only. In this thesis we present a technique that can extract sources in an anechoic environment without the need for as many mixtures as sources, and without limiting the number of mixtures to two. Obviously this is important, since scenarios may arise and experiments can be designed with multiple sensors detecting multiple sources. One would naturally want to utilize all the available information rather than use only two mixtures to perform the separation.

Our results indicate that the technique we present in this thesis improves on the results achieved by DUET, with gains in performance even in the case of two mixtures while successfully extending under-determined BSS to the more general anechoic setting.

# Chapter 2

# Background

In this chapter, we will cover the theoretical background material used in this work. Recall that we are using a two step approach to solve the BSS problem. In fact, each of these two steps contains various processing sub-stages, as can be seen in Figure 2.1. Because we use a variety of tools and methods for blind source separation of mixtures of delayed and attenuated sources in these various sub-stages, we believe it will be useful to first survey these methods. In section 2.1 we provide an overview of the desirable properties governing the choice of transform, and present the STFT in this context. We also cover implementation issues such as sampling and inversion. We show how certain signals, such as speech, exhibit the desirable property of sparsity in this transform domain. In section 2.2, we explain why sparsity is desirable and how it allows us to perform separation by focusing on the mathematical basis behind our technique for separation, constrained $l^1$ minimization. Section 2.3 is dedicated to an overview of k-means clustering, the technique that we subject our feature vectors to, in order to extract the mixing parameters. Finally, section 2.4 aims at introducing the performance

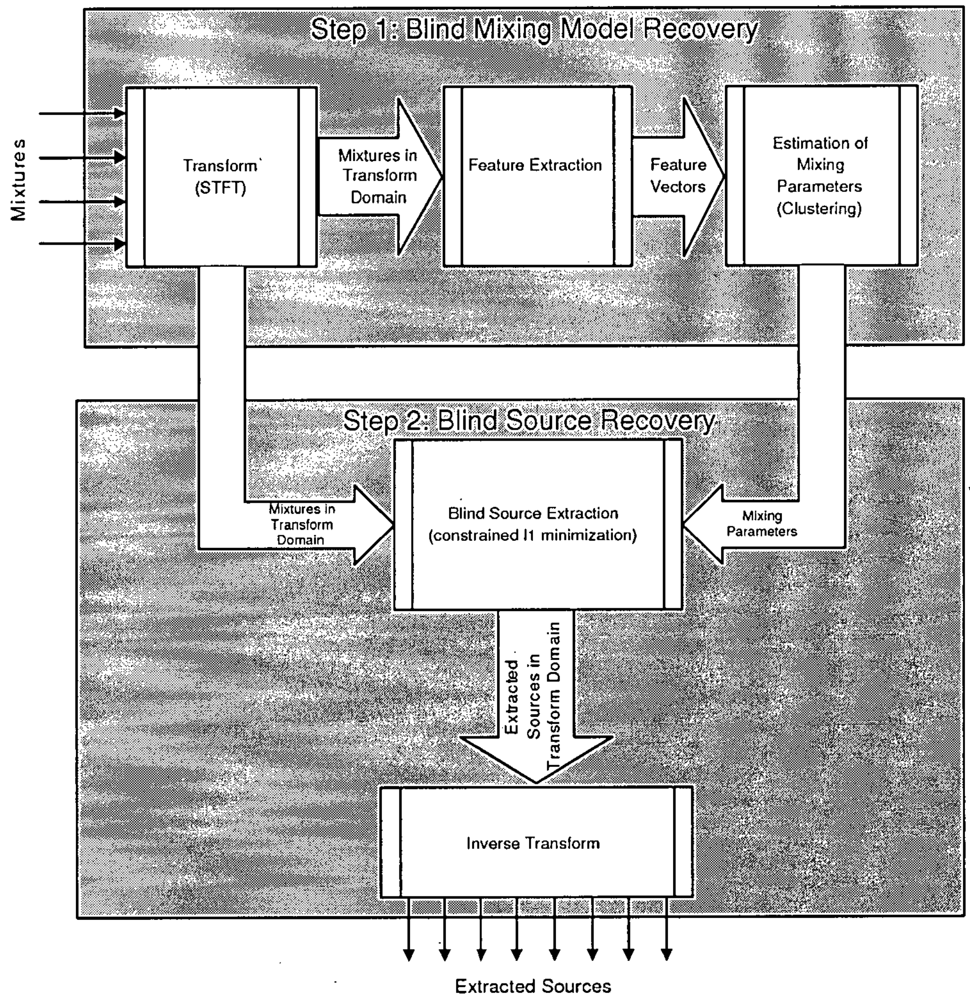

Step 1: Blind Mixing Model Recovery
Mixtures
Transform (STFT)
Mixtures in Transform Domain
Feature Extraction
Feature Vectors
Estimation of Mixing Parameters (Clustering)
Step 2: Blind Source Recovery
Mixtures in Transform Domain
Blind Source Extraction (constrained l1 minimization)
Mixing Parameters
Extracted Sources in Transform Domain
Inverse Transform
Extracted Sources

Figure 2.1: The various sub-stages used in each of the two steps of our proposed technique. After transforming the data, feature vectors are extracted and used to estimate the mixing parameters via clustering. Both the mixing parameters and the mixtures in the transform domain are then used in the second step of the algorithm to recover the sources.

measures we use to assess the quality of our separation technique.

## 2.1 Transform of Choice

As a first step in solving the anechoic BSS separation problem, we need to transform the data to an appropriate domain. The transform of choice must have certain properties that make blind mixing model recovery and blind source recovery easier. While one could choose from a variety of transforms such as complex wavelets and wavelet packets, or even over-complete dictionaries of bases (combinations of several transforms), we choose to use the STFT for a number of reasons. Mainly, signals like speech are sparse in that domain. Moreover, the STFT is linear and the STFT of a delayed signal is a complex constant multiplied by the STFT of the original signal. Of some importance as well is the fact that the STFT is not very computationally intensive.

We shall capitalize on most of these properties in chapter 3 to extract the appropriate features and subsequently the mixing parameters as well as to recover the sources. In what follows we will present the STFT as a logical extension to the regular Fourier Transform, briefly discussing the afore mentioned properties as well as implementation issues.

Table 2.1: Properties of the Fourier Transform

| Property | Function | Fourier Transform |
|---|---|---|
| Translation/Time delay | $f(t-\tau)$ | $e^{-j\omega\tau}\hat{f}(\omega)$ |
| Linearity | $af_1(t)+bf_2(t)$ | $a\hat{f}_1(\omega)+b\hat{f}_2(\omega)$ |

### 2.1.1 Fourier Transform

The Fourier transform of a function $f(t)$ is given by:

$$\hat{f}(\omega) = \int_{-\infty}^{+\infty} f(t)e^{-j\omega t}dt \tag{2.1}$$

Furthermore, its inverse is given by

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \hat{f}(\omega)e^{j\omega t}d\omega. \tag{2.2}$$

The Fourier transform, which is a continuous and bounded function of frequency ($\omega$) measures how much oscillations there are at each ($\omega$). Two important properties of the Fourier transform, i.e. linearity and translation, that are relevant to this work are reported in table 2.1. Both properties can be derived by a simple change of variables in the Fourier integral.

Another important property of the Fourier transform, known as Parseval's relation is:

$$\int_{-\infty}^{+\infty} |f(t)|^2dt = \frac{1}{2\pi}\int_{-\infty}^{+\infty} |\hat{f}(\omega)|^2d\omega. \tag{2.3}$$

Put into words, equation (2.3) simply states that the Fourier transform conserves 'power', i.e. power in the time domain is equal to power in the frequency domain.

As we are concerned with digital signals, we are thus interested in the discrete counterpart of the Fourier transform. The N-point discrete Fourier transform (DFT) of a digital signal $f[n]$ is defined as:

$$\hat{f}[k] = \sum_{n=0}^{N-1} f[n] e^{-jk2\pi n/N}, \tag{2.4}$$

and its inverse is defined as:

$$f[n] = \frac{1}{N} \sum_{k=0}^{N-1} \hat{f}[k] e^{jn2\pi k/N}. \tag{2.5}$$

Here also Parseval's relation holds, as do the properties of the Fourier transform discussed previously in table 2.1. As for implementation, the DFT of a signal can be calculated by means of the Fast Fourier Transform (FFT) via $O(NlogN)$ operations. For this work, this brief overview of the Fourier transform will suffice; a more extensive discussion of both the Fourier transform and the DFT, as well as of their properties can be found in [37], [40], or any good book on signal processing.

### 2.1.2 The Short Time Fourier Transform

Shortcomings of the Fourier transform, namely its assumption of stationarity of the signal, prevent it from being a very useful tool for the analysis of real world time-varying signals. A more reasonable assumption in that case would be stationarity over short periods of time. Motivated by that, we now consider the Short Time Fourier Transform.

The Short Time Fourier Transform (STFT) of a function $f(t)$, also known as Windowed Fourier Transform (WFT), can be expressed mathematically as follows:

$$F^W[f](\tau,\omega) \quad = \quad \int_{-\infty}^{+\infty} f(t)g(t-\tau)e^{-j\omega t}dt, \tag{2.6}$$

where g(t) is a real and symmetric window, normalized so that $\|g\| = 1$ and so that $\|g(t-\tau)e^{-j\omega t}\| = 1$. Note here that the term $g(t-\tau)e^{-j\omega t}$ in equation 2.6 is simply a translation (time shift) of $g(t)$ by $\tau$ and a frequency modulation (frequency shift) by $\omega$ [37].

Gabor introduced this transform to measure the frequency variations of sound [37], in 1946. One can observe that it represents a one dimensional signal by a two dimensional one, showing the changes in frequency content over time. Its inverse is given by:

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} F^W[f](\tau,\omega)g(t-\tau)e^{j\omega t}d\tau d\omega. \tag{2.7}$$

Moreover, it conserves the energy of the signal:

$$\int_{-\infty}^{+\infty} |f(t)|^2 dt = \frac{1}{2\pi}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} |F^W[f](\tau,\omega)|^2 d\tau d\omega. \tag{2.8}$$

For a proof of the reconstruction relation and the energy preservation relation, one can refer to [37]. The discrete version of the STFT follows from the continuous one in a manner similar to how the DFT follows from the Fourier Transform.

$$F^W(f)[k,l] = \sum_{n=0}^{N-1} f[n]g[n-k]e^{-j2\pi nl/N} \tag{2.9}$$

Also its inverse is given by:

$$f[n] = \frac{1}{N}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} F^W(f)[k,l]g[n-k]e^{j2\pi nl/N} \tag{2.10}$$

As Previously mentioned, the STFT transforms a one dimensional signal into a two dimensional one in Time-Frequency (TF) space. It basically consists of performing the Fourier transform on a windowed segment of the signal, sliding the window by one sample and then repeating. Computationally, performing the STFT to obtain a representation of the signal at every $k$ and $l$, i.e. $\forall k,l$ such that $0 \leq k < N, 0 \leq l < N$ requires $O(N^2 logN)$ operations, and is thus more expensive than the regular Fourier transform. However, the properties of the Fourier transform illustrated in table 2.1 hold for the STFT as well. This is obvious from the fact that the STFT is nothing more than the Fourier transform performed on short, windowed, overlapping portions of the signal.

Also worth noting is the fact that there is a time-frequency resolution tradeoff in the choice of window-length with the STFT. The longer the window is, the higher the frequency resolution and the lower the time resolution [37]. Furthermore, the STFT is a highly redundant TF representation. To remove this redundancy one usually samples the STFT on a rectangular grid. Figure 2.8 shows two examples of sampled STFT grids, one for a transform with a higher time resolution and the other with a higher frequency resolution.

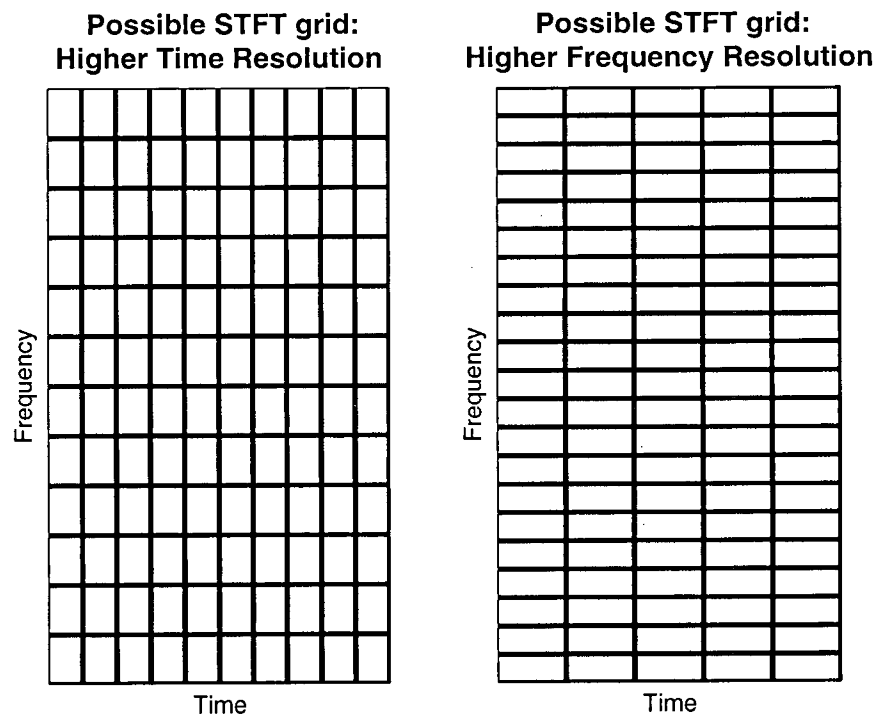


Figure 2.2: STFT grids using two different window sizes. The one on the left has better time resolution, while the one on the right has better frequency resolution.

If sampling is used, then perfect reconstruction can be achieved only if certain conditions hold [1]:

1. Number of frequencies at which the STFT is calculated is at least equal

to the window length.

2. Sampling is done at a rate of at least $2B$ samples per second (to avoid aliasing), where $B$ is the bandwidth of the window used.

Condition 2 stems from the fact that the window chosen can be viewed as a filter of bandwidth $B$ in the frequency domain; to avoid aliasing, one has to sample the signal at least at twice that frequency.

Thus, one reconstruction procedure called the overlap-add technique [1], which shall be used in this work is as follows.

**Overlap-add Method:**

1. Perform the inverse Fourier transform on one frame.
2. Multiply the resultant time domain portion of the signal by the window used.
3. Move to the next frame and go back to step 1, repeating until all frames have been transformed back into the time domain.
4. Shift each time domain signal relative to the previous one by the overlap amount and add all of them together.

### 2.1.3 Examples Using the STFT

The importance of the STFT stems from the fact that it does not assume stationarity of the signal to be transformed except over the support of the chosen window. On the other hand the Fourier transform assumes that the signal is stationary over its entire duration. Figure 2.3 shows the Fourier transform magnitudes of two signals $s_1(t)$ and $s_2(t)$:

$$\begin{aligned} s_1(t) &= \sin(2\pi f_1 t) + \sin(2\pi f_2 t), \text{ for } t < 2 \\ s_2(t) &= \begin{cases} 2\sin(2\pi f_1 t) & \text{, for } t \leq 1 \\ 2\sin(2\pi f_2 t) & \text{, for } 1 < t < 2 \end{cases} \end{aligned}$$

where $f_1 = 60Hz$ and $f_2 = 120Hz$

One can see that the Fourier transforms of these two signals contain no timing information. On the other hand their STFT magnitudes illustrated in figure 2.4 clearly shows the switching nature of $s_2(t)$ and the stationary nature of $s_1(t)$.

### 2.1.4 Sparsity in the STFT Domain

While one could choose from a variety of 'sparsifying' transformations, we have opted to use the STFT for the reasons previously mentioned. In the previous section, we defined the STFT, its inverse, as well as conditions for perfect reconstruction. Having the mathematical properties of the STFT, we can now move on to show how certain signals, such as speech, exhibit sparsity in the STFT domain.

Sparsity is the opposite of density; it usually refers to the number or percentage of zero (or more loosely, near zero) coefficients in a given matrix or data-set. In [13] sparsity means that most coefficients 'do not differ

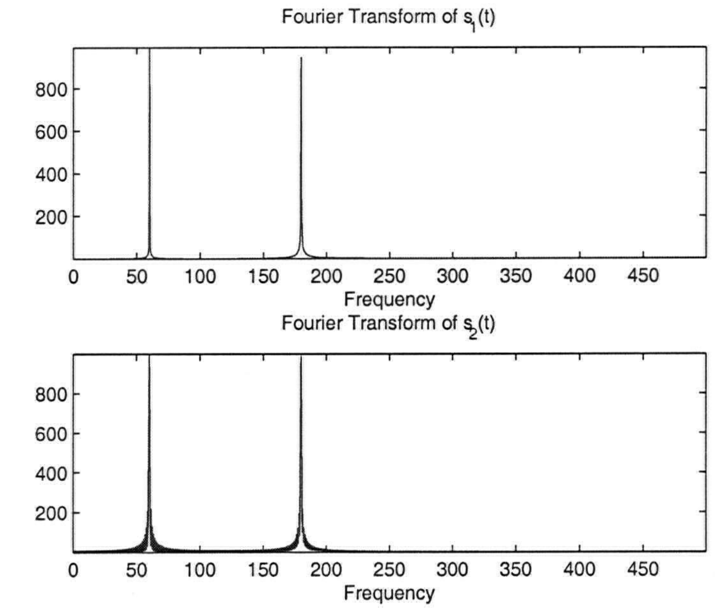


Figure 2.3: Fourier transform magnitudes of $s_1(t)$ (Top) and $s_2(t)$ (Bottom). Note that both transforms are very similar and that all timing information is lost.

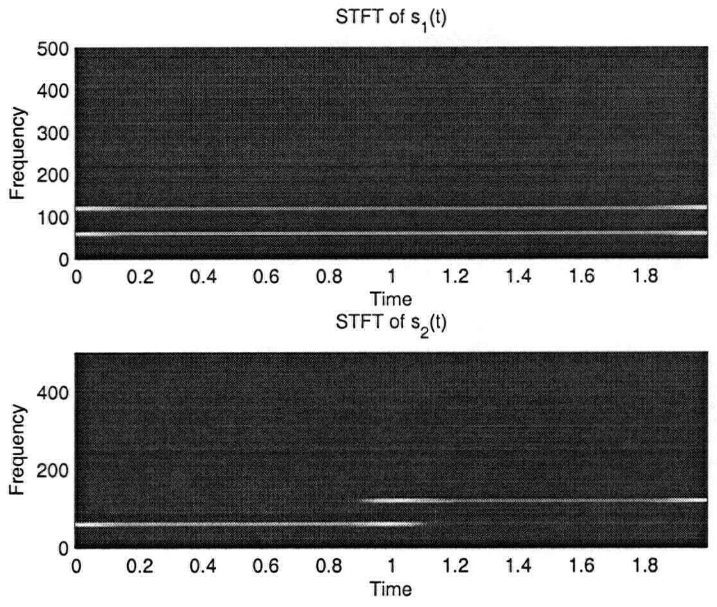


Figure 2.4: STFT of $s_1(t)$ (Top) and $s_2(t)$ (Bottom). Note that the timing information is preserved, where it can be seen that at t=1, one sinusoid switches off while the other switches on.

significantly from zero'. Furthermore, a distribution is defined as sparse 'if most of the probability mass lies close to zero'. That means that probably, a few coefficients will hold most of the signal power. Thus, sparse data representations are obviously very important in coding because they provide good coding efficiency [13]. In blind source separation, sparsity is of great importance as well. Cardoso [10] noted that the accuracy with which the mixing parameters in a BSS model can be estimated is a function of how non-gaussian the sources are. A consequence of that is that sparse sources improve the accuracy in the estimation of the mixing parameters. Of equal importance is that given sparser sources, the quality of separation achieved is higher [46], [47]. This leads to the conclusion that a transformation that yields a sparse representation of the data is desirable, both for estimating the mixing parameters correctly, and for performing separation.

In an experiment to illustrate the sparsity of speech in the STFT domain, we use 50 speech sources of 50000 samples each, sampled at 16000Hz from the TIMIT database and transform them to the STFT domain using 3 different window sizes of 32ms, 64ms, and 128ms with an overlap factor of 50% and using a Hamming window. Next, we sort the STFT coefficients of each in decreasing order of power, and plot their average cumulative powers, which can be seen in figure 2.5. Also plotted on the same figure are the average cumulative power of both the unaltered time domain sources and of their Fourier transforms. Table 2.2 shows the percentage of coefficients needed to represent 90%, 95%, 98% and 99% of the total signal power, using the STFT with varying window sizes, the time domain signal and Fourier transformed signals. We can see that the STFT window-sizes of 32ms and 64ms demonstrate similar performance, with the latter being superior in terms of sparsity, capturing 98% of the total signal power with only approximately 9% of the coefficients. For further illustration of the sparsity exhibited by speech in the STFT domain, figure 2.6 shows the normalized histograms of the same 50 speech sources in the time domain and in the STFT domain. It is obvious from this figure as well that the transform domain exhibits a much sparser signal representation, as can be seen from the fact that the magnitudes of most coefficients are concentrated near zero.

Table 2.2: The Percentage of Coefficients Needed to Represent Various Percentages of the Total Signal Power

| Percentage of the Total Power | **Percentage of Points Needed** | | | | |
|---|---|---|---|---|---|
| | STFT: 32ms | STFT: 64ms | STFT: 128ms | Fourier Domain | Time Domain |
| 90.000 | 2.6781 | **2.2656** | 2.6999 | 16.954 | 14.098 |
| 95.000 | 5.0167 | **4.5710** | 5.5212 | 24.604 | 23.756 |
| 98.000 | 9.4100 | **9.1439** | 11.032 | 34.560 | 38.970 |
| 99.000 | **13.650** | 13.697 | 16.428 | 41.640 | 51.104 |

## 2.2 Sparsity and the $l^0$ - $l^1$ Equivalence

In this section we discuss, under the assumption of sparsity, the solution to constrained optimization in the case of an over-complete dictionary, i.e. one where we have more than enough vectors to span the space. Balan

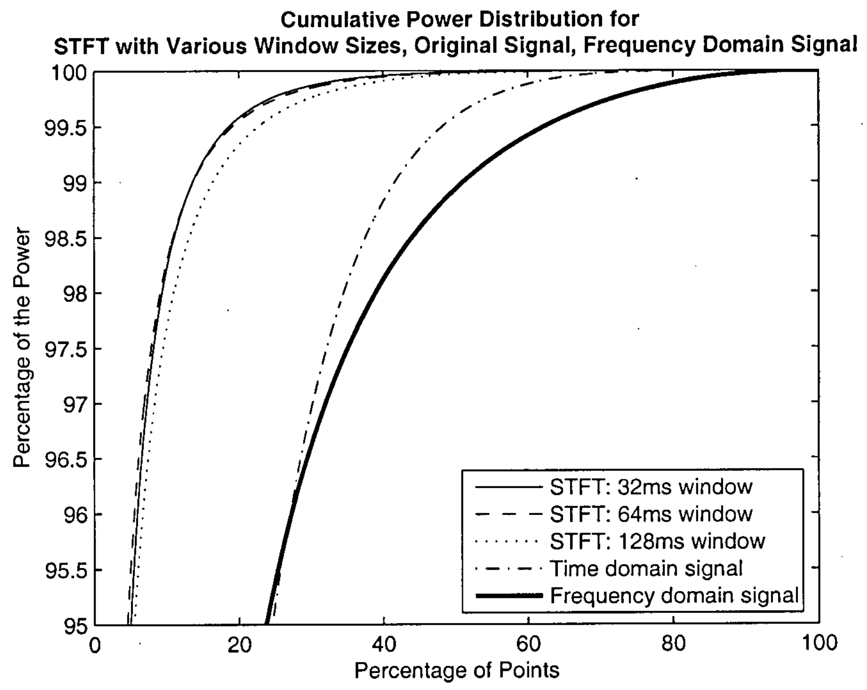
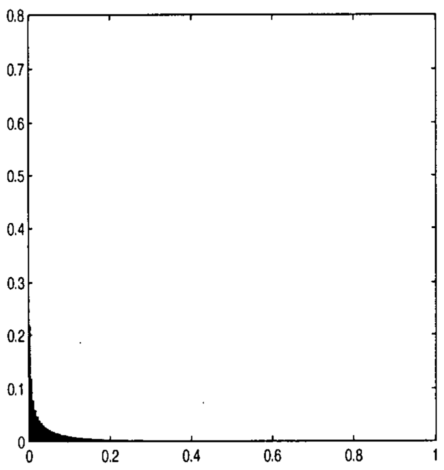
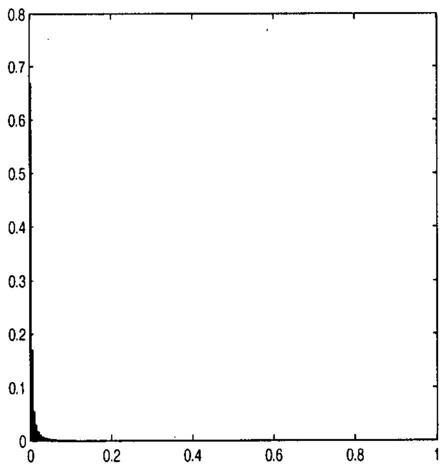


Figure 2.5: Average cumulative power of the time domain signals, frequency (Fourier) domain signals and STFT of speech for window sizes of 32ms, 64ms and 128ms. The STFT with 32ms and 64ms window length exhibit a sparser representation of the data (more power in fewer coefficients). The original time domain representation and the frequency domain representation both exhibit very low sparsity.



(a) average probability density of the magnitudes of 50 time-domain speech signals



(b) average probability density of the magnitudes of the STFT of the same signals, with a window length of 64ms and 50% overlap

Figure 2.6: Average probability density of the absolute value of 50 speech signals in time domain (a), and of the absolute value of the same 50 sources in time-frequency domain (b). Note how much more sparse the TF data are. In both sub-figures, the values have been normalized to the range [0 1]

et al [4] define the estimation of a sparse signal representation as one with the minimum number of nonzero components. Finding the solution to such a problem is hard. Fortunately, under certain assumptions and conditions, mostly related to the sparsity of the signals, equivalent easier problems can be defined and solved. Several important papers ( [15], [14], [4], [30]) discuss such methods for finding the sparsest possible solution to constrained optimization problems.

To that end, it is important to define $l^p$ norms, for $p > 0$ [37], [4] as :

$$\|f\|_p = \left( \sum_n |f(n)|^p \right)^{1/p}$$

and

$$\|f\|_0 = |supp(f)| \text{ for } p = 0,$$

where $supp(f) = \{k|f[k] \neq 0\}$ is the support of $f$ and $|S|$ is the cardinal of the discrete set $S$. In other words, the $l^0$ norm is simply a count of the non-zero terms that $f[k]$ contains. Intuitively, minimizing this number will maximize sparsity. Thus, we now recognize the equivalence between solutions maximizing sparsity and those minimizing the $l^0$ norm.

## 2.2.1 Mathematical Formulation

To mathematically formalize the problem we are dealing with, we present the following. Consider a set of $n$ $m$-dimensional vectors $a_j$, $j = 1, \ldots, n$ similar to [17], with $n > m$ and let $A$ be the $(m \times n)$ matrix with these vectors as its columns:

$$A = [\mathrm{a}_1|\mathrm{a}_2|\dots|\mathrm{a}_n]. \tag{2.11}$$

Using a set of $n$ weights (or variables) s, any linear combination of $A$ can be written in matrix notation as

$$\mathrm{x} = A\mathrm{s}. \tag{2.12}$$

In general, given $\mathbf{x}$ and $A$, it is not possible to recover a unique solution for s without making any further assumptions, as equation 2.12 is comprised of a set of $m$ linear equations with $n$ unknowns, and $n > m$. However, if s contains only a few non-zero terms, then it may be possible to recover s by trying to find the sparsest possible solution to equation 2.12, i.e. solving:

$$P_0 : \text{find s, such that } A\mathrm{s} = \mathrm{x} \text{ and } \|\mathrm{s}\|_0 \text{ is minimized.} \tag{2.13}$$

Equivalently:

$$P_0 : \text{find } \arg\min_s \|\mathrm{s}\|_0, \text{ subject to } A\mathrm{s} = \mathrm{x}. \tag{2.14}$$

As [17] notes, solving 2.14 is generally not feasible in general since it requires a combinatorial approach, where one tries all possible combinations of columns of $A$. Thus, it is convenient to find an equivalent, or even similar problem, that is easier or more straight-forward to solve. As previously mentioned, several authors ( [15], [14], [17], [44], [4], [30], [36]) have investigated the use of the $l^1$ norm as a substitute to the $l^0$ norm, to obtain the following problem:

$$P_1 : \text{find } \arg\min_{s} \|\mathrm{s}\|_1, \text{ subject to } A\mathrm{s} = \mathbf{x}. \tag{2.15}$$

$P_1$ can be solved efficiently by linear programming in the real case, and by second order cone programming [33] in the complex case, which makes it more preferable to solve than the $P_0$ problem. The seminal paper by Donoho and Huo [15] proves that the solution to 2.14 and 2.15 are unique and identical as long as the underlying realizations of s are sufficiently sparse. This result is proved for the case of the mixing matrix $A$ composed of two orthogonal bases. Donoho and Elad [14] later extend these results to the case of general non-orthogonal dictionaries. They derive sufficient conditions defining *how sparse* the underlying sources should be for the solution to be:

1. unique, and
2. attainable via $l^1$ minimization.

They introduce the notion of *spark*, the smallest number of linearly *dependent* columns of a matrix, and derive the sufficient condition for the solution to 2.15 to be *unique*. First, however, to highlight the difference between the rank and spark of matrices we present a simple example in figure 2.7, where two matrices $A_1$ and $A_2$ have the same rank but different sparks.

Going back to the uniqueness condition, it is established whenever the number of non-zero terms in s is less than *spark*$(A)/2$ [15], [14]:

$$\|s\|_0 < \frac{\text{spark}(A)}{2}. \tag{2.16}$$

It is not hard to see that the spark of an $(m \times n)$ matrix $A$ satisfies the following [14]:

$$2 \leq \text{spark}(A) \leq \text{Min}(n, \text{rank}(A) + 1) \tag{2.17}$$

Thus, in the case of under-determined mixing:

$$n > m \Rightarrow \text{Min}(n, \text{rank}(A) + 1) = \text{rank}(A) + 1. \tag{2.18}$$

Thus, depending on the properties of $A$, solving 2.14, will yield a unique answer for s, as long as the number of nonzero components of s is less than $\frac{m+1}{2}$. This can be seen by substituting $m+1$ as the upper bound of spark(A) in 2.17, and then assuming that spark(A) is at its upper bound, substituting it in 2.16. In other words, if the matrix $A$ is well behaved, and its spark is bigger than its rank, we could simultaneously extract $\frac{m+1}{2}$ active sources, while still maximizing sparsity, assuming we could solve problem $P_0$. This means that in a BSS context, from two mixtures we could extract two sources (at every point!), and from 5 mixtures we can extract 3 sources that solve 2.14.

$$A_1 = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad ; \quad A_2 = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure 2.7: Two matrices $A_1$ and $A_2$. Note that while $rank(A_1) = rank(A_2) = m = 3$, $spark(A_1) = m + 1 = 4 \neq spark(A_2) = 2$.

Thus far we have discussed the conditions pertaining to the uniqueness of the solution to $P_0$. We will now discuss the conditions pertaining to its equivalence to the solution of $P_1$. Donoho and Elad [14], and also Gribonval and Nielsen [18] independently derived very similar results related to $M(A)$, the *mutual coherence* of $A$, where

$$M(A) := \max_{i \neq j} |\langle \mathrm{a}_i, \mathrm{a}_j \rangle|. \tag{2.19}$$

More specifically, they conclude that if:

$$\|s\|_0 < f(A) = \frac{[1 + 1/M(A)]}{2}, \tag{2.20}$$

then the solution to 2.14 can be obtained by solving 2.15. It is worth mentioning here that for any $A$, where $(n > m)$, $Spark(A) > \frac{1+1/M(A)}{2}$ [14]. This means that if inequality 2.20 is indeed satisfied, the solution to $P_1$ (equation 2.15) is necessarily unique and the sparsest. Note that these results apply to both the real $\mathbb{R}^m$ and the complex spaces $\mathbb{C}^m$ as generalized by [44].

The theoretical results we have presented so far in this section are important, because they prove that the BSS problem we formulate in the coming chapters is tractable provided sufficiently many of the unknown sources are zero at every Time-Frequency point.

Another comment on 2.15, is that solving it does not require a priori knowledge of which sources are zero or even if any of them are zero. We just know that if sufficiently many of them *are* simultaneously zero, the solution to 2.15 matches the solution to 2.14 and is unique. Moreover, the solution to 2.15 is easily achievable via linear programming in the real case and via second order cone programming [33] in the complex case. In this work, and in the context of over-determined BSS with delays, we will run into these constrained optimization problems. To solve them, we adopt an approach based on a geometric interpretation of the $l^1$ norm, which we will present in a subsequent section.

### 2.2.2 Performance Analysis on solving $P_1$

The results presented in the previous section provide sufficient conditions for the uniqueness and recoverability of $\mathbf{s}$ from $A$ and $\mathbf{x}$ in the under-determined case, i.e. when we have more unknowns than equations. More importantly these conditions are either hard to compute (complexity of spark computation) or rather stringent. For instance, note that the condition in equation 2.20 relates to $M(A)$, which in turn relates to the two 'closest' vectors of $A$. In other words, if $A$ contains two vectors $\mathbf{a_i} \approx \mathbf{a_j}$ that are very close to one another, then $|\langle \mathrm{a_i}, \mathrm{a_j} \rangle| \approx 1$, and $M(A)$ is at least equal to that. This consequently means that $f(A)$ of equation 2.20 is equal to or slightly larger than one. The effect is that the solutions to the minimization problems $P_0$ (2.14) and $P_1$ (2.15) are only guaranteed to be identical if only one term in $\mathrm{s}$ is nonzero. Li et al. [31] make an observation similar to this one and offer simulations and a probabilistic analysis into the recoverability of $\mathrm{s}$ in the case where $\mathbf{x}$, $A$, and $\mathrm{s}$ are all real. Their results show that even when the conditions set in equation 2.20 are not met, we may still be able to extract the correct solution to 2.14 by solving 2.15.

Bearing in mind that in the BSS problem, we have no guarantee that the sources are maximally sparse, except via transforming them into the STFT, it is worth running some simulations to determine the recoverability of the sources under certain looser assumptions on sparsity. Thus, we proceed to conduct our own recoverability analysis on sparse *complex* sources, and complex mixing matrices trying to answer the following questions:

- Can we still recover good estimates of the sources if the conditions for uniqueness of the $P_0$ solution and equivalence to $P_1$ solution are not met?
- What if the sources we are looking for are not solutions to $P_0$?
- How sparse do the sources have to be for us to be able to extract them?
- To what extent will the extraction be succesful?

To get an idea about the answers to these questions, we generate random complex mixing matrices from a uniform distribution. We also generate random complex sources from an exponential distribution (for the sake of ensuring some kind of sparsity). Furthermore, we fix the number of sources to 10, and repeat our simulations 1000 times for each of the following settings. We vary the number of mixtures available $m$ from 2 to 10 and force the number of active sources, i.e. non-zero sources to be $l$, letting $l$ vary from 1 to $m$. We then extract our estimate of the sources during each iteration using the 'geometric approach' to solving $P_1$ which will be outlined in a latter section. Finally, we calculate the percentage of times when the solution to $P_1$ is identical to the actual sources we are trying to extract, and we report the results in table 2.3. Table 2.3, verifies that for any number of mixtures available, if only one source is active, the solution to $P_1$ matches the sources. Also as the number of mixtures available increases, the approach can extract an increasing number of sources correctly. For example, with 8 mixtures available, we can extract up to 3 active sources perfectly and up to 5 active sources 85% of the time. Naturally, when we have 10 available mixtures we can solve for the 10 sources perfectly, regardless of how many of them are active, since the system of equations 2.12 is now even-determined. Table 2.4 shows the average signal to noise ratios obtained with the same experiment. It aims at proving that even if we do not have perfect reconstruction, the errors that are introduced, still allow for acceptable signal to noise ratios. Figure 2.8 shows how the success rate decreases when we attempt to solve

$P_1$ for 7 mixtures of 10 sources as the number of active sources varies from 1 to 7.

Table 2.3: Percent of times where the extracted sources using $l^1$ minimization matches actual sources.

| Available | Number of Active Sources (l) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Mixtures ($m$) | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 2 | 100 | 10.2 | - | - | - | - | - | - | - | - |
| 3 | 100 | 44.3 | 5.6 | - | - | - | - | - | - | - |
| 4 | 100 | 64.7 | 23 | 4.4 | - | - | - | - | - | - |
| 5 | 100 | 92.1 | 61.3 | 26.2 | 5 | - | - | - | - | - |
| 6 | 100 | 94.6 | 72.8 | 43.4 | 20.3 | 4.8 | - | - | - | - |
| 7 | 100 | 100 | 98.2 | 86.1 | 58.4 | 31.1 | 8.8 | - | - | - |
| 8 | 100 | 100 | 99.9 | 98.1 | 87.8 | 70.4 | 41.9 | 17.6 | - | - |
| 9 | 100 | 100 | 100 | 99.9 | 99.2 | 95.6 | 84.5 | 64.8 | 36.9 | - |
| 10 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |

Table 2.4: Signal to Noise Ratio (SNR) in dB of extracted sources using $l^1$ minimization.

| Available | Number of Active Sources (l) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Mixtures ($m$) | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 2 | 303 | 2.6 | - | - | - | - | - | - | - | - |
| 3 | 301 | 6.6 | 3.1 | - | - | - | - | - | - | - |
| 4 | 297 | 9.4 | 5.8 | 4 | - | - | - | - | - | - |
| 5 | 281 | 16.1 | 10.2 | 7.6 | 5.4 | - | - | - | - | - |
| 6 | 285 | 16.6 | 11.9 | 9.1 | 7.6 | 6.1 | - | - | - | - |
| 7 | 296 | 283 | 27.7 | 18.9 | 13.8 | 11.2 | 8.9 | - | - | - |
| 8 | 281 | 294 | 54.2 | 29.2 | 20.4 | 16.6 | 13.5 | 11.7 | - | - |
| 9 | 285 | 286 | 285 | 50.8 | 34.9 | 28.7 | 22.1 | 18.4 | 15.4 | - |
| 10 | 288 | 287 | 276 | 288 | 288 | 288 | 284 | 287 | 287 | 287 |

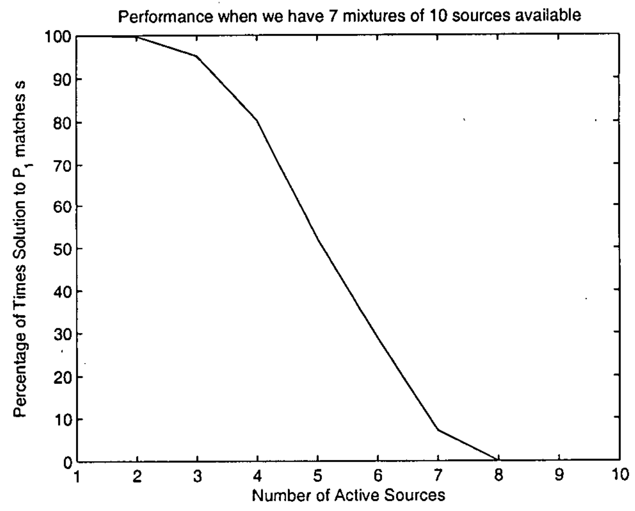


Figure 2.8: Success Rate (in %) of the solution to $P_1$ capturing the correct sources when we have 8 mixtures and 10 sources.

### 2.2.3 Section Summary and Perspective

Recall that in this work we are dealing with blind source separation, in the case where we have more sources than available mixtures. The aim of this section was to show the following:

*Given a transformation that yields a very sparse representation* $\mathbf{s}$ *of the sources in such a way as to satisfy equation 2.20, and given both the mixing matrix* $A$ *(under-determined with* $m < n$*), and the observation vector* $\mathbf{x}$ *in the transform domain , we can recover* $\mathbf{s}$ *precisely, via solving* $P_1$*, and then inverse transform to obtain the sources.*

*Also, even if the conditions outlined for uniqueness and equivalence are* *not met, or if the solution we are looking for is not necessarily the sparsest possible, we can still expect good recoverability of sources by solving $P_1$ as the results of table 2.3 indicate.*

In blind source separation, the only information we have is the observation vector $\mathbf{x}$. Necessarily we have to make some assumptions on the sources to be able to extract them. In the previous section, we demonstrated that for speech signals the STFT is a very sparse representation. Thus, for the purpose of separation of speech sources, we already have established a 'sparsifying' transformation and the observation vector(s) $\mathbf{x}$. What remains is for us to be able to formulate the problem in the same form as that of $P_1$, i.e. as a linear mixing, and -once that is done- to be able to estimate the mixing matrix $A$. Then, we can proceed to extract the sources by solving $P_1$ .

## 2.3 K-means Clustering

In the previous sections we have proven that the STFT is a sparsifying transform for signals like speech. We have also shown that given sparse sources, we can solve under-determined mixing problems if both the mixing matrix and the observation vectors (mixtures) are known. In the case of blind source separation we only have the observation vectors and hence have to estimate the mixing matrix. Fortunately, sparsity of the sources in the transform domain allows us to extract and use certain feature vectors and estimate the mixing matrix by clustering these vectors. The feature vectors we select, and the rationale for using them will be explained in a later chapter. In this section we will briefly discuss k-means clustering and some minor variations of it, which we shall use subsequently.

### 2.3.1 The K-means Algorithm

Perhaps one of the most widely used clustering algorithms available to us, k-means clustering [35] is attractive because of its simplicity and relative speed. K-means partitions data (possibly multi-dimensional) into K-mutually exclusive clusters [38], returning indices indicating which cluster each data point belongs to and the cluster centers. Moreover, it performs partitioning by minimizing the distances within each cluster, while maximizing the distances between clusters [38].

The k-means algorithm is described in any good book on classification or pattern recognition. Here we will present it in a manner similar to [34].

**The k-means Algorithm:**

1. Initialize the first K-cluster centers.
2. Assign each sample point (or vector) to its nearest cluster according to some distance measure.
3. Compute the new cluster centers as the average of the points in that cluster (if using the Euclidean distance).
4. If any cluster center has changed, repeat steps 2 and 3, otherwise terminate.

Table 2.5: K-means cluster centers extracted using Euclidean and City-block distance measures.

| distance measure | k-means Extracted Cluster Centers | | | |
|---|---|---|---|---|
| cityblock | $(4.976, -2.999)$ | $(4.024, 3.988)$ | $(-4.025, 1.996)$ | $(0.006, 1.997)$ |
| Euclidean | ( 4.983,−3.033) | (4.133, 4.029) | ( −4.164,1.988) | (0.034,2.018) |

Possible choices of distance measures that k-means, and the respective cluster center updates one can use are [38]:

- Euclidean distance ($l^2$ norm) $\Rightarrow$ new cluster center is the mean of the points in that cluster
- City block distance ($l^1$ norm) $\Rightarrow$ new cluster center is the median of the points in that cluster

### 2.3.2 An Example of Clustering using the K-means Algorithm

As an example of using the k-means clustering algorithm, we simulate 4 clusters where each is sampled from a 2-dimensional Laplacian distribution with the following cluster centers (5,-3), (4,4), (-4, 2), and (0,2), and we run the k-means algorithm using the two distance measures outlined in the previous subsection to obtain the cluster centers reported in table 2.5. The 4 clusters and the extracted centers using the city-block distance measure can be seen in figure 2.9.

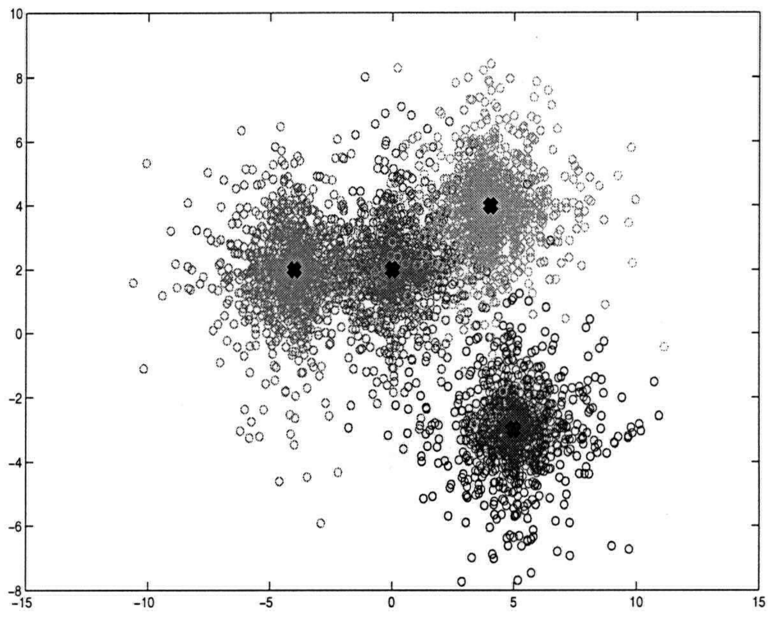


Figure 2.9: An example with four different 'classes' (in color) and the extracted cluster centers using $l^1$ distance (black crosses)

## 2.4 BSS Performance Measures

In this section we will discuss the performance measures we selected to assess the proposed algorithm. Concerns for the evaluation of BSS performance were first outlined by Schobben et. al in [41]. These were later developed mathematically by Gribonval et. al in [18] to measure the quality of blind source separation from more than one aspect. They incorporate several measures of distortion: interference, algorithmic artifacts, and noise. They also include a measure of the total distortion incurred from all these contributions combined. Thus the measures introduced by [18] are: Signal to Interference Ratio, Signal to Artifact Ratio, Signal to Distortion Ratio, and Signal to Noise Ratio. In this work we use these measures to quantify the quality of separation of the proposed algorithm and thus it is necessary to provide a brief overview of how they were derived in [18].

The argument presented [18] is that a simple measure of distortion of a certain signal $s$, estimated by $s_{est}$, cannot be done using the standard relative distortion measure: $D_1 = \frac{\|s - s_{est}\|}{\|s_{est}\|}$, because of certain intrinsic ambiguities of BSS techniques. The relevant ambiguity here is that of scaling. In other words, a BSS algorithm can only extract underlying sources up to a multiplicative constant (and usually an error), i.e. $s_{est} = cs + \epsilon$. Under such ambiguities, the distortion measure $D_1$, would highly penalize such scaling and would thus be inappropriate. As an alternative Gribonval et. al [18] suggested the following total distortion measure:

$$D_{total} = \frac{\|s_{est}\|^2 - |\langle s_{est}, s\rangle|^2}{|\langle s_{est}, s\rangle|^2}, \tag{2.21}$$

where $\langle x(t), y(t)\rangle = \sum_t x(t)y(t)$, is the inner product of x and y [18]. Thus projecting $s_{est}$ on $s$ and calling the remaining orthogonal error term $\epsilon_{total}$, we obtain [18]: $s_{est} = \langle s_{est}, s\rangle\, s + \epsilon_{total}$.

As previously mentioned, this total error term can be further decomposed into various contributions [18] due to interference from other sources, artifacts forced by the algorithm, and noise. Thus we obtain [18]:

$$\epsilon_{total} = \epsilon_{interference} + \epsilon_{artifacts} + \epsilon_{noise}. \tag{2.22}$$

To be able to explicitly define each of these contributions, we must first define the following [18]:

- $\mathbf{P_s}$, the orthogonal projector onto the subspace spanned by all the sources.
- $\mathbf{P_{s,n}}$, the orthogonal projector onto the subspace spanned by all the sources and the noise signals combined.

Now, for a particular source $s_k$ and its estimate $s_{k,est}$, the various errors are defined in [18] as:

- $\epsilon_{interference} = \mathbf{P_s} s_{k,est} - \langle s_{k,est}, s_k \rangle s_k$
- $\epsilon_{noise} = \mathbf{P_{s,n}} s_{k,est} - \mathbf{P_s} s_{k,est}$.
- $\epsilon_{artifact} = s_{k,est} - \mathbf{P_{s,n}} s_{k,est}$

This leads to the following definitions of the various distortions and the total distortion [18]:

- $D_{interference} = \frac{\|\epsilon_{interference}\|^2}{|\langle s_k, s_{k,est} \rangle|^2}$
- $D_{noise} = \frac{\|\epsilon_{noise}\|^2}{|\langle s_k, s_{k,est} \rangle s_k + \epsilon_{interference}|^2}$
- $D_{artifact} = \frac{\|\epsilon_{artifact}\|^2}{|\langle s_k, s_{k,est} \rangle s_k + \epsilon_{interference} + \epsilon_{noise}|^2}$

- $D_{total} = \frac{\|s_{k,est}\|^2 - |\langle s_{k,est}, s_k\rangle|^2}{|\langle s_{est,k}, s_k\rangle|^2}$.

Finally, [18] defines the Signal to Distortion Ratio (SDR), Signal to Interference Ratio (SIR), Signal to Noise Ratio (SNR) and Signal to Artifact Ratio (SAR):

- $SDR = 10\log_{10} D_{total}^{-1}$
- $SIR = 10\log_{10} D_{interference}^{-1}$
- $SNR = 10\log_{10} D_{noise}^{-1}$
- $SAR = 10\log_{10} D_{artifact}^{-1}$.

Note that computing these distortion measures involves computing $\mathbf{P}_{\mathbf{s}} s_{k,est}$ and $\mathbf{P}_{\mathbf{s},\mathbf{n}} \mathbf{s}_{\mathbf{k},\mathbf{est}}$. Not making any assumptions about the orthogonality of the original sources [18] notes that:

$$\mathbf{P}_{\mathbf{s}} s_{k,est} = \sum_{i=1}^{n} c_l s_l = \mathbf{c}^T \mathbf{s}, \tag{2.23}$$

where $\mathbf{c} = conj(G)^{-1}\mathbf{d}_m$, $G = \mathbf{s}\mathbf{s}^H$, and $\mathbf{d}_k = (\langle s_{k,est}, s_j\rangle)_{j=1}^{n}$ is a vector of the projections of $s_{k,est}$ on all the sources. Also, assuming that all the noise signals at the receivers are orthogonal (uncorroloated) to one another and to all the source signal, [18] obtains:

$$\mathbf{P}_{\mathbf{s},\mathbf{n}} s_{k,est} = \mathbf{P}_{\mathbf{s}} s_{k,est} + \sum_{i=1}^{m} \langle s_{k,est}, n_i\rangle\, n_i / \|n_i\|^2, \tag{2.24}$$

where $n_i$ is the noise affecting the $i^{th}$ mixture. Matlab routines to compute all these measures presented in [18] are available online [16], and were used in this thesis.

# Chapter 3

# Thesis Contribution

In this chapter we mathematically formulate the problem of BSS in an anechoic environment and explain the approach taken to recovering the sources.

## 3.1 Mixing Model and Problem Definition

First we describe the anechoic mixing model. Given $n$ sources $s_1(t), \dots, s_n(t)$, and $m$ mixtures $x_1(t), \dots, x_m(t)$, such that

$$x_k(t) = \sum_{j=1}^{n} a_{kj} s_j(t - \delta_{kj}), \; k = 1, 2, \dots, m \tag{3.1}$$

where $a_{kj}$ and $\delta_{kj}$ are scalar attenuation coefficients and time delays associated with the path from the $j^{\text{th}}$ source to the $k^{\text{th}}$ receiver, respectively. Without loss of generality we set $\delta_{1j} = 0$ for $j = 1, \dots, n$.

$$\hat{x}_k(\omega, \tau) = \sum_{j=1}^{n} a_{kj} \hat{s}_j(\omega, \tau) e^{-i\omega\delta_{kj}}, \; k = 1, 2, \dots, m \tag{3.2}$$

The short time Fourier transform (STFT) of a function $s$ with respect to a fixed *window function* $W$ is defined as

$$F^W[s](\tau,\omega) := \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} W(t-\tau)s_j(t)e^{-i\omega t}dt \tag{3.3}$$

which we shall usually refer to as $\hat{s}(\tau,\omega)$. Throughout this chapter , we shall use

$$\begin{aligned} F^W[s_j(\cdot-\delta)](\tau,\omega) &= \exp(-i\omega\delta)F^W[s_j](\tau-\delta,\omega) \\ &\approx \exp(-i\omega\delta)F^W[s_j](\tau,\omega). \end{aligned} \tag{3.4}$$

This assumption is realistic as long as the window function $W$ is chosen appropriately. We have briefly touched on this issue in section 2.1.2. A more detailed discussion can be found in [5].

*Main Problem.* Given $x_1, x_2, \ldots, x_m$, estimate $s_1, \ldots, s_n$ in the general case where $n \geq m$ and $n$ is unknown . Taking the STFT of $x_1$,. . . ,$x_m$ with respect to an appropriate window function $W$ and using (3.4), the mixing model (3.2) reduces to

$$\hat{\mathrm{x}}(\tau,\omega) = A(\omega)\hat{\mathrm{s}}(\tau,\omega), \tag{3.5}$$

where

$$\hat{\mathrm{x}} = [\hat{x}_1 \ldots \hat{x}_m]^T,\ \hat{\mathrm{s}} = [\hat{s}_1 \ldots \hat{s}_n]^T, \tag{3.6}$$

and

$$A(\omega) = \begin{bmatrix} 1 & \dots & 1 \\ a_{21}e^{-i\omega\delta_{21}} & \dots & a_{2n}e^{-i\omega\delta_{2n}} \\ \vdots & \vdots & \vdots \\ a_{m1}e^{-i\omega\delta_{m1}} & \dots & a_{mn}e^{-i\omega\delta_{mn}} \end{bmatrix}. \tag{3.7}$$

From this point on, instead of the continuous STFT, we shall use the equivalent discrete counterpart, i.e., we shall consider the samples of the STFT of $s$ on a lattice in the time-frequency plane given by

$$\hat{s}_j[k,l] = \hat{s}_j(k\tau_0, l\omega_0) \tag{3.8}$$

where $\tau_0$ and $\omega_0$ are the time-frequency lattice parameters. The equivalence is nontrivial and only true if the family $\{e^{il\omega_0 t}W(t-k\tau_0) : k,l \in \mathbb{Z}\}$ constitutes a *Gabor frame* for the signal space of interest [12]. For this, one needs an appropriately chosen window function $W$ with sufficiently small $\tau_0$ and $\omega_0$ as discussed in section 2.1.2. Note that, in the discrete framework, the mixing model can now be written as

$$\hat{\mathrm{x}}[k,l] = A(l\omega_0)\hat{\mathrm{s}}[k,l]. \tag{3.9}$$

with $\hat{\mathrm{x}}, \hat{\mathrm{s}}$ as in (3.6), and $A$ as in (3.7).

To solve the above BSS problem, we propose a 2-stage algorithm, as introduced in Section 1.3. In the first stage, we use a clustering approach aimed at estimating the mixing parameters, i.e., all attenuation and delay coefficients. In the second stage, we utilize the mixing matrices, formed by our mixing parameters, to perform demixing in the TF plane.

## 3.2 Blind Mixing Model Recovery

### 3.2.1 STFT Sparsity

In order to estimate the mixing parameters of our model, we utilize the fact that time-frequency representations of speech signals are sparse, thus few coefficients will capture most of the signal power [45]. We have verified this assumption in section 2.6.

### 3.2.2 Feature Vector Extraction

Having represented our mixtures in the TF domain, we now proceed to construct a $2m-1$ dimensional feature space where the first $m$ coordinates are the normalized attenuations of the mixtures and the remaining $m-1$ dimensions are the delays of the mixtures relative to the first one. Note that we omit the delay of the first mixture relative to itself.

Let $\hat{\mathbf{x}}$ be as in (3.6). First, at each point $[k,l]$ on the TF lattice, we define the normalized attenuation vector

$$\hat{\mathbf{x}}_{\mathrm{at}}[k,l] := \frac{1}{\|\hat{\mathbf{x}}[k,l]\|} \left[ \begin{array}{ccc} |\hat{x}_1| & \dots & |\hat{x}_m| \end{array} \right] [k,l]. \tag{3.10}$$

Here $\|\cdot\|$ denotes the Euclidean norm. Note that the resulting $\mathbf{x}_{\mathrm{at}}[k,l]$

correspond to points on the unit sphere of $\mathbb{R}^m$.

Next, we calculate the complex phases $-l\omega_0\hat{\Delta}_{j1}[k,l]$ of mixtures $\hat{x}_j$, $j = 2,\ldots,m$ relative to the mixture $\hat{x}_1$ at each TF point $[k,l]$, as in [45]. More precisely,

$$\hat{\Delta}_{j1}[k,l] := -\frac{1}{l\omega_0}\angle\frac{\hat{x}_j[k,l]}{\hat{x}_1[k,l]}. \tag{3.11}$$

Finally, we append the $m$-dimensional feature vector $\mathbf{x}_{\text{at}}$ defined in (3.10) to obtain:

$$\begin{aligned}&\mathbf{F}[k,l]\\ &:= \left[\ \left|\frac{\hat{x}_1[k,l]}{\|\hat{\mathbf{x}}[k,l]\|}\right| \ \cdots\ \left|\frac{\hat{x}_m[k,l]}{\|\hat{\mathbf{x}}[k,l]\|}\right| \ \cdots\ \hat{\Delta}_{21}[k,l] \ \cdots\ \hat{\Delta}_{m1}[k,l]\ \right].\end{aligned} \tag{3.12}$$

Note that if at a TF point $[k,l]$ only one source, say $s_J$, is active, i.e., $\hat{s}_J[k,l] \neq 0$ and $\hat{s}_j[k,l] = 0$ for $j \neq J$, the feature vector will reduce to

$$\begin{aligned}\mathbf{F}[k,l] &= \mathbf{F}_J\\ &:= \left[\ \left|\frac{a_{1J}}{C}\right| \ \cdots\ \left|\frac{a_{mJ}}{C}\right| \ \cdots\ \delta_{2J} \ \cdots\ \delta_{mJ}\ \right],\end{aligned} \tag{3.13}$$

where $C = (\sum_{i=1}^m a_{iJ}^2)^{1/2}$. In this case $F_J$ does not depend on $[k,l]$, and is completely determined by the mixing parameters in the $J$th column of the mixing matrix, given in (3.7). Moreover, the converse is also true, i.e., given $F_J$, one can extract the mixing parameters. Therefore, if the sources have disjoint TF representations, the feature vector $F[k,l]$ corresponding to any TF point $[k, l]$, at which at least one source is active, will be in the set $\{F_1, \ldots, F_n\}$. Once we obtain this set, we can compute the mixing parameters using (3.13).

In practice, it is unrealistic to expect that the sources have disoint TF representations. However, as discussed in Section 3.2.1 as well as in [45], speech signals have sparse Gabor expansions. Therefore, it is highly likely that there will be an abundance of TF points at which one source is dominant. In other words, there will be several points in the TF plane where one source has a high contribution while the other sources have a near zero contribution. Thus in the feature space, points $F[k, l]$ will tend to cluster around $F_j, j = 1, \ldots, n$, the coordinates representing the columns of the mixing matrix. At this stage, having extracted the feature vectors of all significant TF points –TF points at which the mixtures $|\hat{x}_j|$ are not smaller than a threshold–, we perform K-means clustering to obtain the cluster centers and consequently the parameters of the mixing model. In summary:

**Parameter Estimation Algorithm:**

1. Compute the mixture vector $\hat{\mathrm{x}}[k, l]$, as in (3.6) at every TF point $[k, l]$.

2. Compute the corresponding feature vector $F[k, l]$, as in (3.12), at every TF point $[k, l]$.

3. Peform some clustering algorithm (e.g., K-means) to find the $n$ cluster centers in the feature space. The cluster centers will yield estimates of the mixing parameters.

4. Compute the columns of the mixing matrix using the parameters found in step 3. Normalize each column so that its Euclidean norm is 1. (Note that the renormalization only rescales the attenuations, and does not affect the delays.)

Figure 3.1 shows a three dimensional view of the extracted cluster centers and the real cluster centers from 3 mixtures of 5 sources.

### 3.2.3 Inherent Assumptions and Limitations

The proposed parameter estimation algorithm will yield a meaningful estimate of the mixing parameters only if certain assumptions hold. First, due to the periodicity of the complex exponential and to avoid phase indeterminacy we assume that

$$|\omega\delta_{ij}| < \pi, \tag{3.14}$$

for all $i, j$ and every $\omega$. This amounts to assuming that

$$|\delta_{max}| < \pi/\omega_{max}, \tag{3.15}$$

where $\delta_{max}$ is the largest delay in the system and $\omega_{max}$ is the highest frequency present. If $\omega_{max} = \omega_s/2$, where $\omega_s$ is the sampling frequency, then this means that our maximum allowed delay is one sample [45]. This entails that the spacing between any two microphones be limited to $d < 2\pi c/\omega_s$, where $c$ is the speed of sound [45]. Note that we do not need to know the actual spacing between the microphones - only that it is within the bound. Second,

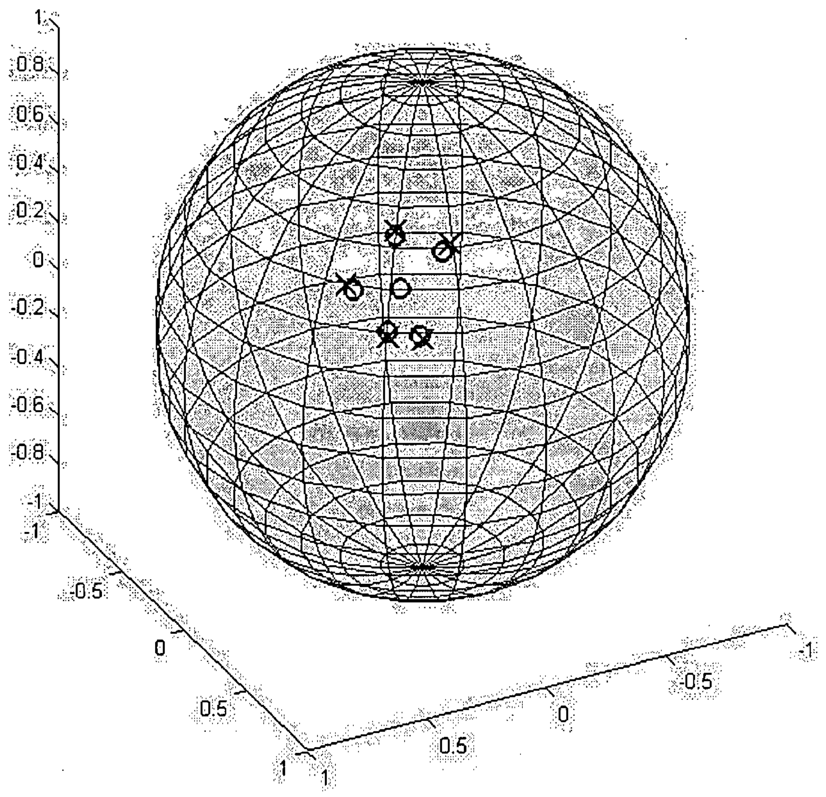


Figure 3.1: 3-D view of real (crosses) and estimated (circles) parameters as recovered from the K-means clustering stage. The algorithm was run on 3 simulated mixtures of 5 sources (m=3, n=5), with the user solving for 6 sources. Note the proximity of the real to the estimated parameters. Also note the estimated source parameter that does not correspond to any real one. Displayed are the 3-dimensional normalized attenuation parameters. The delay parameters have not been included.

due to the same problem of phase indeterminacy we also assume that all the attenuations $a_{ij}$ present in the mixing model are positive. This is due to the following equality,

$$ae^{-i\omega\delta} = -ae^{-i(\pm\pi+\omega\delta)} = -ae^{-i\omega(\delta\pm\pi/\omega)}, \tag{3.16}$$

which leads to two possible attenuation-delay pairs for every entry in the feature vectors of (3.13). We avoid these problems in this work by assuming delays that are limited to one sample at most and positive attenuations. The positive attenuation assumption holds for anechoic audio mixtures.

## 3.3 Blind Source Extraction

Having obtained the cluster centers, we must now use them to extract the sources. The cluster centers can be used to construct the mixing matrix $\tilde{A}[l]$:

$$\tilde{A}[l] = [\tilde{a}_1[l]|\tilde{a}_2[l]|\dots|\tilde{a}_n[l]], \tag{3.17}$$

where the $\tilde{a}_i[l]$'s are the columns of the mixing matrix and are constructed from the cluster centers in a manner following that of equation (3.7).

$$\tilde{A}[l] = \begin{bmatrix} \tilde{a}_{11}e^{-il\omega_0\tilde{\delta}_{11}} & \dots & \tilde{a}_{1n}e^{-il\omega_0\tilde{\delta}_{1n}} \\ \tilde{a}_{21}e^{-il\omega_0\tilde{\delta}_{21}} & \dots & \tilde{a}_{2n}e^{-il\omega_0\tilde{\delta}_{2n}} \\ \vdots & \vdots & \vdots \\ \tilde{a}_{m1}e^{-il\omega_0\tilde{\delta}_{m1}} & \dots & \tilde{a}_{mn}e^{-il\omega_0\tilde{\delta}_{mn}} \end{bmatrix}. \tag{3.18}$$

Next, we want to estimate the individual sources $s_1, s_2, ..., s_n$. Equivalently, we want to find $[\hat{s}_1^e[k,l], \ldots, \hat{s}_n^e[k,l]]^T =: \hat{\mathbf{s}}^{\mathrm{e}}[k,l]$ such that:

$$\tilde{A}[l]\hat{\mathbf{s}}^{\mathrm{e}}[k,l] = \hat{\mathbf{X}}[k,l]. \tag{3.19}$$

Thus at each TF point [k,l], we have $m$ equations (corresponding to the $m$ available mixtures that we have), with $n > m$ unknowns $(\hat{s}_1^e[k,l], \ldots, \hat{s}_n^e[k,l])$. Assuming that this system of equations is consistent, it has infinitely many solutions.

## 3.3.1 Sparsity and $l^1$ Minimization

The importance of sparsity of the sources in the transform domain stems from the fact that it means that only a few of them will have large STFT coefficients at a given TF point. Thus, in addition to satisfying (3.19), $\hat{\mathbf{s}}^{\mathrm{e}}$ should also be sparse. Motivated by this, we consider:

$$P_0 : \text{find } \hat{\mathbf{s}}^{\mathrm{e}}, \text{such that } \tilde{A}\hat{\mathbf{s}}^{\mathrm{e}} = \hat{\mathbf{X}} \text{ and } \sum_{i=1}^{n} |\hat{\mathbf{s}}_i^e[k,l]|^0 \text{ is minimized} \tag{3.20}$$

and

$$P_1 : \text{find } \hat{\mathbf{s}}^{\mathrm{e}}, \text{such that } \tilde{A}\hat{\mathbf{s}}^{\mathrm{e}} = \hat{\mathbf{X}} \text{ and } \sum_{i=1}^{n} |\hat{\mathbf{s}}_i^e[k,l]|^1 \text{ is minimized} \tag{3.21}$$

Seeking the sparsest possible solution to (3.19) is equivalent to solving $P_0$, as discussed in section 2.1.4. Recalling that solving $P_0$ is difficult and noting that the solution is very sensitive to noise, we consider $P_1$, the closest convex optimization problem to $P_0$ [15]. The solution to $P_1$ is not the sparsest possible solution in general. However, it will yield a minimally spread solution [4], and, at least in certain cases, one can prove that the solution to $P_1$ is the sparsest possible as discussed in section 2.2. Since we are considering general anechoic mixing and using the STFT domain, our mixing matrix is both complex and frequency dependant and for our application, we are not necessarily interested in the sparsest possible solution. We are interested in obtaining reasonable estimates of the sources in the TF plane. Motivated by this and the argument presented in section 2.1.4, we will solve $P_1$ as a tool to extract our sources.

### 3.3.2 Solving $P_1$

Several methods, such as linear programming, exist for solving (3.21). In this work we adopt an approach similar to that of [46]. In [46], which deals with a real and constant mixing matrix $A$, a simple geometrical approach to solving the optimization problem is proposed. They state that when the columns of the mixing matrix are normalized to the unit sphere, the optimal solution to (3.21) will include at most $m$ of the original sources. They call the solution incorporating these sources the *shortest path* decomposition from the origin to the point. In [43], it is proved that finding the shortest path as presented in [46] amounts to minimizing the $l^1$ norm. This shortest path may only include directions available in the columns of the mixing matrix.

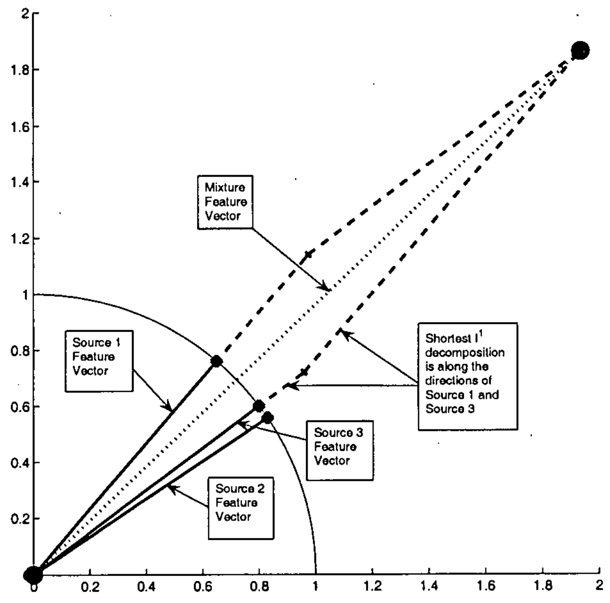


Figure 3.2: An example case with 2 mixtures and 3 extracted sources. The best $l^1$ decomposition as proposed in [29] and [9] is shown.

Thus, we are trying to find a 'best' $l^1$ decomposition of an m-dimensional vector by using the best $m$ out of $n$ available vectors. Given that the optimal $l^1$ solution satisfying the constraint will contain at most contributions from $m$ sources, we can now proceed to formulate a system of m-equations and m-unknowns and solve it at each TF point. Figure 3.2 shows the optimal $l^1$ decomposition of a 2-dimensional vector at some arbitrary TF point into contributions from the two sources that leads to the shortest path. The lengths of the contributions from each source will be the extracted magnitude of the STFT of those sources at this particular TF point.

### 3.3.3 Source Separation

Given that all the extracted time delays $\Delta$ were referenced to the first mixture, we reference our mixtures to the first one as well. We do that by subtracting at each TF point, the phase of the first mixture from the phases of the other mixtures. We must now select the m-sources that are most likely to be active at any given TF point. Selecting the first of these sources is easy; it is the one whose cluster center vector is closest to the mixture feature-vector. We select the other m-1 vectors by a process of trial and error. We solve the m equations of (3.19) with all possible combinations having $m$ active sources. We then select the combination that gives us the smallest $l^1$ norm satisfying (3.21).

**Remark:** The parameter estimation part of our algorithm is based on the assumption that one source is active at most TF points, so that clustering yields good estimates. On the other hand, the source separation stage relaxes this assumption and, depending on the number of available mixtures, will work even if many sources are active at some of the TF points.

In other words, we are constructing and solving a system of m-equations of m-unknowns at each time-frequency point [k,l]:

$$\hat{x}[k,l] = \tilde{A}_r[l]\hat{\mathrm{s}}_r^e[k,l] \tag{3.22}$$

with the vector $\hat{\mathbf{s}}_r^e$ being the vector of the m unknown sources we are solving for at this stage. $\tilde{A}_r[l]$, is the reduced estimated mixing matrix of size $m \times m$ composed of the columns of $\tilde{A}$ corresponding to the active sources. $\hat{x}[k,l]$, which is a column vector containing the STFT values of the mixtures at $[k,l]$ is known. Also, $\tilde{A}_r[l]$, can be constructed from the attenuation and damping parameters extracted in the clustering step. The solution thus is:

$$\begin{aligned} \hat{s}_r[k,l] &:= \tilde{A}_r^{-1}[l]\hat{x}[k,l] \qquad (3.23) \\ \hat{s}_v[k,l] &:= 0; \end{aligned}$$

where $v$ indexes the $n-m$ sources assumed to be not active at $[k,l]$. We solve (3.23) for all possible combinations of $m$ out of $n$ available sources, and choose the solution that yields the minimum $l^1$ norm. Finally, we correct the phase of the sources by preserving their magnitude and adding the phase of the first mixture since all delay features were referenced to the first mixture. We finally calculate the inverse transform using the overlap-add method as outlined in section 2.1.2 to obtain our estimate of the sources.

### 3.3.4 Effects of Changing the Number of Sources

In real applications, the number of sources contributing to the mixture is typically unknown a priori. In [30], it is proven that an overestimate of the number of sources will yield no obvious degradation in the performance. The argument is that if a user estimates $\hat{n} > n$ sources, the estimated mixing matrix $\tilde{A}'$ of size $m \times \hat{n}$ will contain estimates of all the columns of the real mixing matrix $A$, in addition to columns not corresponding to the mixing parameters of any source. In the case of overestimating the number of sources, the optimization problem (3.21) will change to:

$$P_1' : \text{find } \hat{\mathrm{s}'}^{\mathrm{e}}, \text{such that } \tilde{A}'\hat{\mathrm{s}'}^{\mathrm{e}} = \hat{\mathrm{X}} \text{ and } \sum_{i=1}^{\hat{n}} |\tilde{\mathrm{s}'}_i^{e}[k,l]|^1 \text{ is minimized} \qquad (3.24)$$

and the solution to (3.24) will be $\hat{\mathrm{s}'}^{e} = [\tilde{s}^T, \Delta\tilde{s}^T]^T$ where $\tilde{s}$ is sufficiently close to $\hat{s}$ and $\Delta\tilde{s}$ is sufficiently close to zero. Our results will show that this conclusion, reached for a real and constant mixing matrix, holds for our variable complex mixing matrix as well.

## 3.4 Interference Suppression and Distortion Reduction

The algorithm discussed extracts n-sources from m-mixtures. It accomplishes that by extracting m-sources at each time-frequency point which minimize the $l^1$ norm. The correct recovery of a source at a TF point depends on some assumptions:

1. No more than m-sources are active at that TF point.

2. The columns of the mixing matrix are extracted correctly in the mixing model recovery stage.

3. There is no noise affecting the mixtures.

If the above assumptions hold, then our decomposition of the mixtures into their source contributions will be good. We avoid the problem of having more than m-active sources at each TF point since we cannot solve for more than m-unknowns using m-equations while still satisfying the $l^1$ sparsity assumption. By using a suitable window size for the STFT, this ensures that we obtain a sparse representation of our data. In cases where more than m-sources are active at a large number of TF points, then the problem will become intractable to our approach, since the sparsity assumption will be violated, and $l^1$ minimization may no longer be appropriate. This scenario though is highly unlikely with speech sources where a reasonable number of mixtures is available and we will show in the results section that our algorithm does not encounter this problem. A more important issue is that it is likely that the clustering stage will not perfectly yield the columns of the mixing matrix. It is therefore possible that our estimates of the sources might be negatively affected by this. Also, under the sparsity assumption, it is very likely that there are less than m-sources active at many TF points. In that case, the combination of an error in the estimates of the mixing directions, coupled with the fact that several sources are inactive, and the existence of noise might lead us to falsely assign contributions to sources that are silent. These contributions, which the algorithm assigns as source activity, could in fact be due to projections of contributions from other sources, and possibly due to noise. To circumvent this problem, we introduce a power ratio parameter that the user sets, based on noise level or assumed difficulty of separation. Accordingly, after resolving the contribution of each of the sources, i.e. solving (3.23), we inspect each source's contribution to the total power of all sources at that TF point. We preserve the $k$ highest sources, where $1 < k < m$ which collectively contribute to at least $p\%$ of the total power. We then redistribute the power of the truncated sources among the remaining ones, while still maintaining the power percentages. The argument behind this is that if a source is inactive, noise will still project on its direction, and give a contribution albeit a small one. Thus, to get rid of these unwanted small contributions, we introduce this parameter. To recap,

**Interference Suppression Algorithm:** At each TF point:

1. Sort the source estimates in decreasing order.
2. Preserve the first (highest) $k$ sources that contribute to at least $p\%$ of the total power.
3. Set the remaining estimates to zero.
4. Normalize the sources so that they have the same total power as calculated prior to this stage.

# Chapter 4

# Experiments and Results

We evaluate the performance of our algorithm by presenting results on both simulated and real mixtures. To assess the quality of the separation we utilize the performance measures suggested in [18] where three measures are proposed: Source to Distortion Ratio (SDR), Source to Interference Ratio (SIR) and Source to Artifact Ratio (SAR). Recall from section 2.4 that SAR measures the distortions due to algorithmic or numerical artifacts such as "unnatural zeros" in the STFT. SIR, on the other hand measures the interference still present due to other sources in an extracted source. Finally, SDR is a measure of all types of distortion, whether artifacts, interference or noise. In [18] it is claimed that informal listening tests correlate well with the nature of the perceived distortion as quantified by the SIR and SAR measures. Our own informal listening tests confirm this claim.

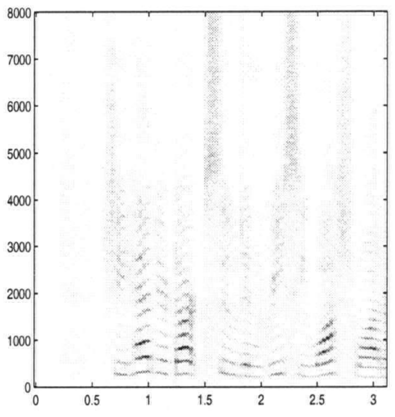
## 4.1 Synthetic Mixtures, Random Mixing Parameters

The algorithm was first tested on synthetic mixtures of 5 sources, of which two are speech and three are music. We generated 5 mixtures comprised of delayed and attenuated versions of these sources with the mixing parameters randomly selected and shown in Table 4.1.

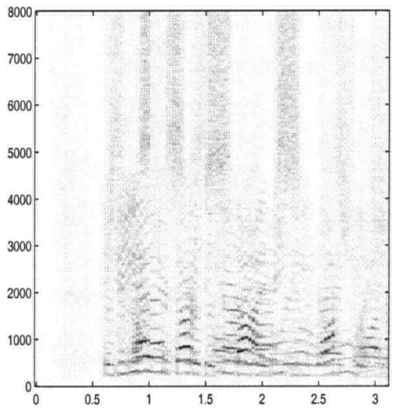
Table 4.1: Mixing parameters

| | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ |
|---|---|---|---|---|---|
| $a_{1i}$ | 0.57 | 0.55 | 0.42 | 0.32 | 0.28 |
| $a_{2i}$ | 0.26 | 0.41 | 0.49 | 0.52 | 0.46 |
| $a_{3i}$ | 0.42 | 0.20 | 0.55 | 0.52 | 0.74 |
| $a_{4i}$ | 0.37 | 0.58 | 0.47 | 0.32 | 0.25 |
| $a_{5i}$ | 0.54 | 0.40 | 0.24 | 0.51 | 0.33 |
| $\delta_{2i}$ | 0.01 | −0.62 | 0.08 | 0.72 | 0.80 |
| $\delta_{3i}$ | 0.42 | −0.61 | −0.70 | 0.71 | 0.64 |
| $\delta_{4i}$ | −0.14 | 0.36 | 0.40 | 0.19 | 0.29 |
| $\delta_{5i}$ | −0.39 | −0.39 | −0.24 | −0.01 | 0.64 |

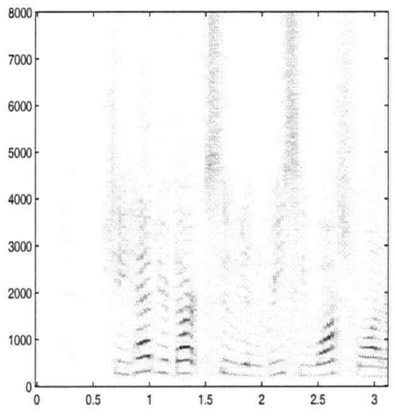
Figure 4.1. shows the TF decomposition of one of the sources, one of the mixtures, and the corresponding extracted source, when using four mixtures for separation. Figure 4.2. shows all the original sources, mixtures, and extracted sources, when using 4 mixtures to perform the separation. Tables 4.2, 4.3 and 4.4 show the demixing performance based on SIR, SAR, and SDR respectively. All the reported results are obtained by running the algorithm with the user parameters $\hat{n} = 6$ and $p = 0.8$, where $\hat{n}$ and $p$ are as defined in sections 3.3.4 and 3.4. Each column in these tables indicates the performance



(a) Original Source Spectrogram

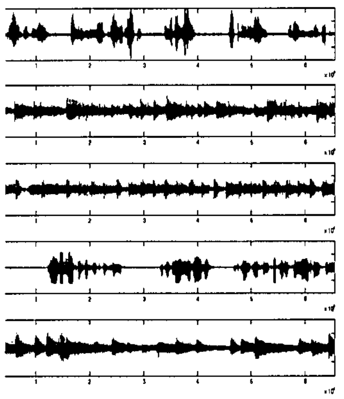


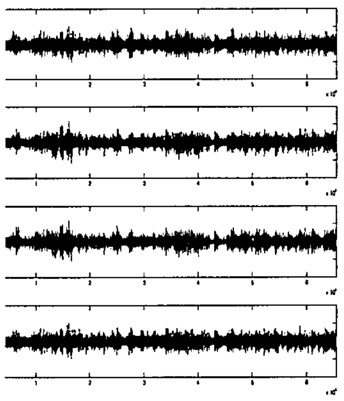
(b) Mixture 1 Spectrogram



(c) Extracted Source Spectrogram

Figure 4.1: The spectrogram of one of the original sources (a), one of the mixtures (b), and the corresponding extracted source (c) from 4 mixtures of 5 sources, when the user estimates the existence of 6 sources.


(a) Original Sources


(b) Mixtures

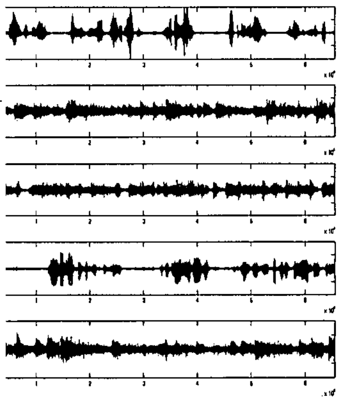

(c) Extracted Sources

Figure 4.2: The original sources (a), delayed and attenuated mixtures (b),and extracted sources (c), from 4 mixtures of 5 sources, when the user estimates the existence of 6 sources.

before demixing as well as when using either 5, 4, 3, or 2 of the available mixtures for demixing; the last column reports the corresponding results obtained using DUET, the anechoic demixing technique briefly discussed in chapter 1. From the tabulated results, one can observe from the last row of Table 4.2 that even when we are demixing 5 sources from just 2 mixtures, there is a mean gain in SIR of over 16 dB, which is quite significant, with the average gain reaching over 30 dB when utilizing all five available mixtures. The improvement in the SIR indicates that the demixing has been successful. This highlights the importance of our proposed technique for achieving sizable improvement in performance by utilizing all the available information at hand. Similarly when looking at the SDR, which measures all the distortions that the extracted source signal has incurred, a gain ranging from 3dB to over 16dB is achieved when using two to five mixtures respectively. The fact that the demixing performance improves with the use of more information is obvious, but that does not take away from the importance of quantifying it. The interesting point to note, however, is the degradation of the SAR upon separation from approximately 20 to 13 dB with the use of all five mixtures and to 0 dB with the use of just two. This can be attributed to the fact that our algorithm is non-linear in nature, and acts on the TF transform of the mixtures, extracting the sources in that domain. This introduces numerical artifacts, such as unnatural zeros or non-smooth transitions in the STFT of the sources, which are reflected by the SAR values reported. Moreover, upon comparison of the results of SDR, SIR and SAR obtained by applying our algorithm on 2 mixtures only, with the results obtained by applying DUET, we note that our algorithm on average outperforms DUET in every single criterion (last 2 columns of Tables 4.2,4.3, and 4.4).

Table 4.2: Demixing Performance with mixtures of 5 sources, $p = 0.8$, $\hat{n} = 6$ : SIR

| Source | *Signal to Interference Ratio (SIR) in dB* | | | | | |
|---|---|---|---|---|---|---|
| | Before Demixing | After Demixing with BAUSS | | | | After Demixing with DUET |
| | | Number of Mixtures Used for Demixing | | | | |
| | | 5 | 4 | 3 | 2 | |
| $s_1$ | −7.4246 | 32.4152 | 25.6948 | 18.7628 | 12.4747 | 9.1474 |
| $s_2$ | −4.0727 | 37.8785 | 18.1916 | 18.9401 | 10.7621 | 4.3941 |
| $s_3$ | 0.5228 | 17.9173 | 13.1723 | 22.6597 | 17.7238 | 5.8776 |
| $s_4$ | −2.5051 | 29.8467 | 19.1309 | 10.8810 | 16.3982 | 16.868 |
| $s_5$ | −5.0683 | 17.4729 | 9.0028 | 12.3569 | 6.3972 | 4.4940 |
| *mean* | −3.7096 | 27.1061 | 17.0385 | 16.7201 | 12.751 | 8.1561 |
| *mean gain* | 0 | 30.8157 | 20.7481 | 20.4297 | 16.4608 | 11.8657 |

As an experiment to push the limit of the algorithm, we tested it on 5 mixtures of 10 sources, using random delays and attenuations. We used the same 5 sources of the previous experiment, adding 5 other speech sources. The resulting mixtures were not sparse, even in the TF plane, as can be seen from figure 4.3., which makes the separation harder. The algorithm performs quite well even in this scenario as the results in Table 4.5 indicate. The average gains in SIR, SAR and SDR of approximately 18.8, 4.9 and 12.3 dB respectively, attest that the algorithm has performed remarkably well in a scenario where there is a much higher number of sources than mixtures. In fact, all but the last extracted source were recovered fairly successfully and

Table 4.3: Demixing Performance with mixtures of 5 sources, $p = 0.8$, $\hat{n} = 6$ : SAR

| Source | *Signal to Artifact Ratio (SAR) in dB* | | | | | |
|---|---|---|---|---|---|---|
| | Before Demixing | After Demixing with BAUSS | | | | After Demixing with DUET |
| | | Number of Mixtures Used for Demixing | | | | |
| | | 5 | 4 | 3 | 2 | |
| $s_1$ | 20.6601 | 11.6572 | 10.6055 | 7.7185 | 0.5058 | 1.6507 |
| $s_2$ | 22.8100 | 12.6903 | 7.2641 | 7.2175 | −5.1410 | −1.6890 |
| $s_3$ | 22.8100 | 14.5325 | 7.9361 | 3.5051 | 2.3721 | −0.3896 |
| $s_4$ | 20.6601 | 16.0863 | 13.3220 | 7.4717 | 4.6826 | 0.97023 |
| $s_5$ | 17.4866 | 12.0881 | 5.4458 | 6.8700 | −2.3201 | −3.2989 |
| *mean* | 20.8854 | 13.4109 | 8.9147 | 6.5566 | 0.0199 | −0.5503 |
| *mean gain* | 0 | −7.4745 | −11.9707 | −14.3288 | −20.8655 | −21.4357 |

Table 4.4: Demixing Performance with mixtures of 5 sources, $p = 0.8$, $\hat{n} = 6$ : SDR

| Source | *Signal to Distortion Ratio (SDR) in dB* | | | | | |
|---|---|---|---|---|---|---|
| | Before Demixing | After Demixing with BAUSS | | | | After Demixing with DUET |
| | | Number of Mixtures Used for Demixing | | | | |
| | | 5 | 4 | 3 | 2 | |
| $s_1$ | −7.4684 | 11.6184 | 10.4617 | 7.3366 | 0.0132 | 0.5124 |
| $s_2$ | −4.1042 | 12.6765 | 6.8662 | 6.8832 | −5.5927 | −3.7573 |
| $s_3$ | 0.4746 | 12.8450 | 6.6403 | 3.4295 | 2.1765 | −.21348 |
| $s_4$ | −2.5630 | 15.9030 | 12.2679 | 5.6028 | 4.3072 | 0.7737 |
| $s_5$ | −5.1687 | 10.9243 | 3.4960 | 5.5963 | −3.6670 | −5.1217 |
| *mean* | −3.7659 | 12.7934 | 7.9464 | 5.7697 | −0.5525 | −1.9455 |
| *mean gain* | 0 | 16.5594 | 11.7124 | 9.5356 | 3.2134 | 1.8204 |

Table 4.5: Demixing Performance with 5 mixtures of 10 sources, $p = 0.8$, $\hat{n} = 12$

| SIR (dB) | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ | $s_8$ | $s_9$ | $s_{10}$ | **mean** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| mixtures | −8.12 | −5.78 | 3.88 | −9.71 | −3.74 | 0.929 | 0.754 | −8.17 | −0.926 | −5.58 | −3.65 |
| extracted sources | 13.7 | 11.8 | 26.3 | 14.7 | 16.4 | 16.6 | 15.5 | 18 | 16.2 | 2.16 | 15.1 |
| gain | 21.8 | 17.5 | 22.4 | 24.4 | 20.1 | 15.6 | 14.7 | 26.2 | 17.1 | 7.74 | **18.8** |
| **SAR (dB)** | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ | $s_8$ | $s_9$ | $s_{10}$ | **mean** |
| mixtures | 2.63 | 4.08 | 2.63 | 4.08 | 3.27 | −4.14 | −0.89 | −0.89 | −3.16 | −3.16 | 0.446 |
| extracted sources | 1.72 | 5.03 | 2.79 | 3.76 | 8.69 | 6.69 | 9.24 | 7.03 | 6.51 | 1.87 | 5.33 |
| gain | −0.91 | 0.946 | 0.163 | −0.327 | 5.42 | 10.8 | 10.1 | 7.92 | 9.67 | 5.03 | **4.89** |
| **SDR(dB)** | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ | $s_8$ | $s_9$ | $s_{10}$ | **mean** |
| mixtures | −10.2 | −7.52 | −0.71 | −11.3 | −5.97 | −7.4 | −4.91 | −12 | −7.69 | −11.2 | −7.89 |
| extracted sources | 1.28 | 3.96 | 2.77 | 3.28 | 7.93 | 6.18 | 8.22 | 6.64 | 5.97 | −2.19 | 4.4 |
| gain | 11.5 | 11.5 | 3.48 | 14.5 | 13.9 | 13.6 | 13.1 | 18.6 | 13.7 | 9.01 | **12.3** |

the speakers' sentences could be discerned without difficulty. One other thing to note from the results of this experiment is that there was an improvement even in the SAR values. A possible reason for this is that due to the high number of sources, the number of points in the TF plane that the algorithm sets to zero is less; thus, the extracted sources exhibit less artifact.

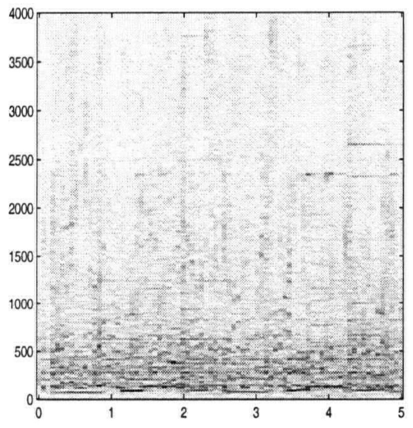


Figure 4.3: The Spectrogram of one of the mixtures. The TF plane of the mixture is not sparse, making the separation harder.

## 4.2 Synthetic Mixtures, Anechoic Room Mixing Parameters

In addition to the experiments mentioned in the previous section, we performed additional ones where the mixing parameters are derived from an anechoic room model as used in [8]. The simulated scenarios involved three microphones and several sources placed around the room. We repeated experiments extracting 3, 4, 5 and 6 sources from 3 mixtures (60 repetitions each) and compared the results to those of DUET. The results can be seen in tables 4.6, 4.7, 4.8, 4.9, and 4.10. There is an obvious advantage in the use of the proposed algorithm, stemming from the use of the information present in all mixtures. One other thing to note is that we ran the algorithm feeding it the correct cluster centers (and hence the correct mixing parameters) in the case of three mixtures and six sources, with p=0, and obtained an average SDR of 5.4341dB, which is close to the 5.23dB obtained with the estimated cluster centers and p=0.6, as reported in table 4.10. We believe this validates the extra process we perform to reduce distortion as explained in section 3.4. As for the selection of the parameter $p$, from the results reported throughout this section, it seems that values in the range of $0.6 - 0.8$, generally give good performance of the algorithm.

Table 4.6: Average Demixing Performance (60 various runs) with 2 mixtures of 3 sources, $\hat{n} = 3$

| | BAUSS | | | DUET |
|---|---|---|---|---|
| | $p = 1$ | $p = 0.8$ | $p = 0.6$ | |
| **SIR (dB)** | 20.291 | 22.164 | **25.783** | 24.279 |
| **SAR (dB)** | **10.679** | 10.590 | 9.7515 | 9.2522 |
| **SDR (dB)** | **9.6128** | 9.4363 | 9.0542 | 8.7223 |

Table 4.7: Average Demixing Performance (60 various runs) with 3 mixtures of 3 sources, $\hat{n} = 3$

| | BAUSS | | | DUET |
|---|---|---|---|---|
| | $p = 1$ | $p = 0.8$ | $p = 0.6$ | |
| **SIR (dB)** | 19.730 | 28.919 | **40.191** | 22.622 |
| **SAR (dB)** | **14.225** | 14.082 | 13.033 | 8.2051 |
| **SDR (dB)** | 12.711 | **13.774** | 13.007 | 7.9123 |

Table 4.8: Average Demixing Performance (60 various runs) with 3 mixtures of 4 sources, $\hat{n} = 4$

| | BAUSS | | | DUET |
|---|---|---|---|---|
| | $p = 1$ | $p = 0.8$ | $p = 0.6$ | |
| **SIR (dB)** | 16.991 | 21.196 | **25.918** | 17.436 |
| **SAR (dB)** | 10.718 | **10.655** | 10.401 | 7.2363 |
| **SDR (dB)** | 8.9965 | 9.6780 | **9.8149** | 4.6757 |

Table 4.9: Average Demixing Performance (60 various runs) with 3 mixtures of 5 sources, $\hat{n} = 5$

| | BAUSS | | | DUET |
|---|---|---|---|---|
| | $p = 1$ | $p = 0.8$ | $p = 0.6$ | |
| **SIR (dB)** | 13.984 | 16.601 | **19.970** | 12.982 |
| **SAR (dB)** | 6.8408 | **6.9605** | 6.8718 | 4.9712 |
| **SDR (dB)** | 5.5626 | 6.1441 | **6.3774** | 2.3890 |

Table 4.10: Average Demixing Performance (60 various runs) with 3 mixtures of 6 sources, $\hat{n} = 5$

| | BAUSS | | | DUET |
|---|---|---|---|---|
| | $p = 1$ | $p = 0.8$ | $p = 0.6$ | |
| **SIR (dB)** | 13.081 | 15.035 | **17.751** | 11.953 |
| **SAR (dB)** | 5.7519 | 5.8750 | **5.9576** | 3.7166 |
| **SDR (dB)** | 4.3056 | 4.7992 | **5.2393** | 1.9419 |

## 4.3 Real Mixtures

To test the proposed techniques in a real setting, we used the real anechoic mixtures posted on [19], which have two sources and two microphones. The microphones are placed 35cm apart, and the sources are placed 60$^o$ degrees to the left of the microphones and 2m on the mid-perpendicular of the microphones respectively [19], [2]. It is worth noting that while we know a priori that we have two mixtures with two sources involved, we run the algorithm completely assuming that there are four sources. As discussed in the previous section and as our results indicate, this does not degrade the separation performance. Table 4.11 shows that our algorithm slightly outperforms the algorithm detailed in [2] and whose audio separation results are in [19]. Our proposed algorithm does slightly better in SIR and by approximately 3dB in SAR and SDR.

Table 4.11: Demixing Performance (in dB) with 2 real mixtures of 2 sources, $p = 0.75$, $\hat{n} = 4$

| | SIR [2] | SIR (BAUSS) | SAR [2] | SAR (BAUSS) | SDR [2] | SDR (BAUSS) |
|---|---|---|---|---|---|---|
| $s_1$ | 26.232 | 41.277 | 4.5363 | 7.8223 | 4.4967 | 7.8200 |
| $s_2$ | 55.410 | 41.249 | 5.6433 | 7.9978 | 5.6433 | 7.9955 |
| *mean* | 40.821 | **41.263** | 5.0898 | **7.9101** | 5.0700 | **7.9077** |

# Chapter 5

# Conclusions and Future Work

In this thesis, we have presented a novel blind source separation algorithm that can extract more sources than available mixtures in the anechoic case where both delays and attenuations play a part in the mixing model. The BSS technique presented combines the strengths of $l^1$ minimization approaches with those of DUET, an approach that relies on TF masking. The technique relies on first extracting feature vectors of time-frequency points and utilizing them in a clustering stage to extract the parameters of the mixing model. This blind mixing model recovery stage is followed by a blind source extraction stage based on $l^1$ minimization, as justified by the sparsity of the sources. We perform the constrained $l^1$ norm minimization step with a technique known as shortest path decomposition. Furthermore, we presented an enhancement to the algorithm based on preserving only a percentage of the extracted signal power to reduce undesirable effects that we attribute to noise and clustering errors.

We presented experimental results based on synthetic mixtures and real mixtures in anechoic environments. Our experiments highlighted the robustness of the algorithm to user set parameters and to lack of knowledge of the real number of sources a priori, thereby generalizing similar results obtained for instantaneous mixing. We measured the performance of our algorithm in terms of SDR, SIR and SAR. We showed that while it necessarily introduces artifacts due to its non-linear nature, we showed that it generally performs favorably with respect to all criteria. Our use of the preserved power ratio parameter enables the user to balance the types of distortion incurred, between artifact and interference. The optimal choice of this parameter, and its relationship to the estimated number of sources used for demixing and the actual number of sources remains a topic for further research.

The technique presented in this thesis would be appropriate for non-audio applications as well. Thus, other topics for future research include blind source separation applied to such mixtures as electroencephalography (EEG) recordings. Prevalent demixing techniques in that field assume an instantaneous mixing model, and there may be a lot to be gained by assuming a more general model utilizing both attenuations and propagation delays. Methods would have to be devised to generalize the proposed technique to be able to handle negative attenuations, or handle the phase indeterminacy problem before it could be successfully applied to EEG recordings.

# Bibliography

[1] J. B. Allen and L. R. Rabiner. A unified approach to short-time Fourier analysis, synthesis. In *Proc. IEEE*, pages 1558–1564, November 1977. Published as Proc. IEEE, volume 65, number 11.

[2] J. Anemuller and Birger Kollmeier. Adaptive separation of acoustic sources for anechoic conditions: a constrained frequency domain approach. *Speech Commun.*, 39(1-2):79–95, 2003.

[3] J. Anemuller, T.J. Sejnowski, and S. Makeig. Complex independent component analysis of frequency domain electroencephalographic data. In *Neural Networks*, pages 16:1311–1323. 2003.

[4] R. Balan, J. Rosca, and S. Rickard. Equivalence principle for optimization of sparse versus low-spread representations for signal estimation in noise. *International Journal of Imaging Systems and Technology, special issue on Blind Source Separation and Deconvolution in Imaging and Image Processing*, to appear 2005.

[5] R. Balan, J. Rosca, S. Rickard, and J. O'Ruanaidh. The influence of windowing on time delay estimates. In *Conf. on Info. Sciences and*

*Systems (CISS)*, volume 1, pages WP1–(15–17), Princeton, NJ, March 2000.

[6] A.J. Bell and T.J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159, 1995.

[7] A. Belouchrani, K. Abed-Meraim, J.F. Cardoso, and E. Moulines. A blind source separation technique based on second order statistics. *IEEE transactions on signal processing*, 45(2).

[8] P. Bofill. Underdetermined blind separation of delayed sound sources in the frequency domain. *Neurocomputing*, 55:627–641, 2003.

[9] P. Bofill and M. Zibulevsky. Blind separation of more sources than mixtures using sparsity of their short-time Fourier transform. In *International Workshop on Independent Component Analysis and Blind Signal Separation (ICA)*, pages 87–92, Helsinki, Finland, June 19–22 2000.

[10] J.F. Cardoso. Blind signal separation: Statistical principles. *Proceedings of IEEE, Special Issue on Blind System Identification and Estimation*, pages 2009–2025, October 1998.

[11] C. Choi. Real-time binaural blind source separation. pages 567–572, Nara, Japan, April 2003.

[12] I. Daubechies. *Ten Lectures on Wavelets.* SIAM, Philadelphia, PA, 1992.

[13] M. Davies and N. Mitianoudis. Simple mixture model for sparse overcomplete ica. *Vision, Image and Signal Processing, IEE Proceedings*, 15(Issue 1), 2004.

[14] D. Donoho and M. Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via $l^1$ minimization. In *Proc. Natl. Acad. Sci. USA 100 (2003), 2197-2202.*

[15] D.L. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Trans. Inf. Theory*, 47:2845–2862, 2001.

[16] Action Jeunes Chercheurs du GDR ISIS (CNRS). Ressources pour la sparation de signaux audiophoniques. *http://www.ircam.fr/anasyn/ISIS/.*

[17] J.J. Fuchs. On sparse representations in arbitrary redundant bases. *IEEE Transactions on Information Theory*, 50(6):1341–1344, June 2004.

[18] R. Gribonval and M. Nielsen. On sparse representations in arbitrary redundant bases. *IEEE Transactions on Information Theory*, 49(12):3320–3325, December 2003.

[19] *http : //medi.unioldenburg.de/demo/demo_separation.html.*

[20] A. Hyvärinen and E. Oja. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9(7):1483–1492, 1997.

[21] A. Hyvarinen and E. Oja. Indpendent component anlysis: Algorithms and applications. *Neural Networks*, 13(4-5):411–430, 2000.

[22] A. Jourjine, S. Rickard, and Ö. Yılmaz. Blind separation of disjoint orthogonal signals: Demixing N sources from 2 mixtures. In *Proc. ICASSP2000, June 5-9, 2000, Istanbul, Turkey*, June 2000.

[23] C. Jutten and J. Herault. Blind separation of sources, part i: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 24:1–10, 1991.

[24] C. Jutten, J. Herault, P. Comon, and E.Sorouchiary. Blind separation of sources, parts i,ii and iii. *Signal Processing*, 24:1–29, 1991.

[25] R. Lambert and A. Bell. Blind separation of multiple speakers in a multipath environment. In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP97)*, pages 423–426, Munich, Germany, April 1997.

[26] R.H. Lambert. *Multichannel blind deconvolution: FIR matrix algebra and separation of multipath mixtures.* PhD thesis, Univ. Southern California, Los Angeles, CA, 1996.

[27] T-W. Lee, M. Lewicki, M. Girolami, and T. Sejnowski. Blind source separation of more sources than mixtures using overcomplete representations. *IEEE Signal Proc. Letters*, 6(4):87–90, April 1999.

[28] M.S. Lewicki and T.J. Sejnowski. Learning overcomplete representations. In *Neural Computation*, pages 12:337–365, 2000.

[29] Y. Li, A. Cichocki, and S. Amari. Sparse component analysis for blind source separation with less sensors than sources. In *Proceedings of 4th International Symposium on Independent Component Analysis and Blind Signal Separation (ICA2003)*, pages 89–94, Kyoto, Japan, April 2003. Riken, ICA.

[30] Y. Li, A. Cichocki, S. Amari, S. Shishkin, J. Cao, and F. Gu. Sparse representation and its applications in blind source separation. In *Seventeenth Annual Conference on Neural Information Processing Systems (NIPS-2003)*, Vancouver, December 2003.

[31] Yuanqing Li, Andrzej Cichocki, and Shun ichi Amari. Analysis of sparse representation and blind source separation. *Neural Comput.*, 16(6):1193–1234, 2004.

[32] J.-K. Lin, D. G. Grier, and J. D. Cowan. Feature extraction approach to blind source separation. In *IEEE Workshop on Neural Networks for Signal Processing (NNSP)*, pages 398–405, Amelia Island Plantation, Florida, September 24–26 1997.

[33] M.S. Lobo, L. Vanderberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

[34] Carl Grant Looney. *Pattern recognition using neural networks: theory and algorithms for engineers and scientists.* Oxford University Press, Inc., New York, NY, USA, 1997.

[35] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

[36] D.M. Malioutov, M. Cetin, and A.S. Willsky. Optimal sparse representations in general overcomplete bases. In *Proc. ICASSP2004, May, 2004, Montreal, Canada*, May 2004.

[37] S. Mallat. *A wavelet tour of signal processing.* Academic Press, 1998.

[38] Documentation of Matlab statistics toolbox.

[39] P. O'Grady, B. Pearlmutter, and S. Rickard. Survey of sparse and non-sparse methods in source separation. *International Journal of Imaging Systems and Technology, special issue on Blind Source Separation and Deconvolution in Imaging and Image Processing*, to appear 2005.

[40] A.V. Oppenheim, R.W. Schafer, and J.R. Buck. *Discrete-time signal processing (2nd ed.).* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1999.

[41] D. Schobben, K. Torkkola, and P. Smaragdis. Evaluation of blind signal separation methods. In *First International Workshop on Independent*

*Component Analysis and Blind Signal Separation*, Aussois, France, January 11–15 1999.

[42] P. Smaragdis. Blind separation of convolved mixtures in the frequency domain. *Neurocomputing*, 22:21–34, 1998.

[43] F.J. Theis and E.W. Lang. Formalization of the two-step approach to overcomplete bss. In N. Younan, editor, *Proc. 4th Intern. Conf. on Signal and Image Processing (SIP'02) (Hawaii)*, 2002.

[44] J.A. Tropp. Recovery of short, complex linear combinations via $l^1$ minimization. *IEEE Transactions on Information Theory*, 51(4):1568–1570, April 2005.

[45] O. Yilmaz and S. Rickard. Blind source separation of speech mixtures via time-frequency masking. *IEEE Transactions on Signal Processing*, 52(7):1830–1847, July 2004.

[46] M. Zibulevsky, B.A. Pearlmutter, P. Bofill, and P. Kisilev. Blind Source Separation by Sparse Decompostion of a Signal Dictionary. In S. Roberts and R. Everson, editors, *Independent Component Analysis: Principles and Practice*, chapter 7. Cambridge, 2001.

[47] Michael Zibulevsky and Barak A. Pearlmutter. Blind source separation by sparse decomposition in a signal dictionary. *Neural Computation*, 13(4):863–882, 2001.