Spring 2013

Advances in blind source separation

Christopher Osterwise

Follow this and additional works at: https://scholarsmine.mst.edu/doctoral_dissertations

Department: Electrical and Computer Engineering

Recommended Citation
Osterwise, Christopher, "Advances in blind source separation" (2013). Doctoral Dissertations. 3.
https://scholarsmine.mst.edu/doctoral_dissertations/3

This thesis is brought to you by Scholars' Mine, a service of the Missouri S&T Library and Learning Resources. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact scholarsmine@mst.edu.
ADVANCES IN BLIND SOURCE SEPARATION

by

CHRISTOPHER THOMAS PAUL OSTERWISE

A DISSERTATION

Presented to the Faculty of the Graduate School of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in

ELECTRICAL ENGINEERING

2013

Approved by

Dr. Steven Grant, Advisor
Dr. Daryl Beetner
Dr. Randy Moss
Dr. Kurt Kosbar
Dr. David Grow
ABSTRACT

Blind Source Separation (BSS) is a classic problem that attempts to separate unknown sources from observed mixtures. A common framework of this is the Cocktail Party problem, where multiple individuals speaking in a noisy room are recorded by a configuration of microphones, with the challenge being to split the observed signals into individual conversations. This dissertation covers the evaluation of an existing algorithm known as Cascaded ICA with Intervention Alignment (CICAIA), the establishment of a new parameter for characterizing blind mixtures, and three new algorithms, of which two are capable of separating any number of sources given more microphones than sources.

Evaluating the performance of the existing algorithm CICAIA reveals that the “reverberation time” parameter most commonly used to describe a mixing environment is actually insufficient to characterize the performance of a given algorithm. This leads to the new descriptor: sparseness. Its formulation is defined, and additional evaluations show its importance in algorithm characterization, while at the same time describing the strengths and weaknesses of CICAIA—namely, it has mediocre performance of about 13 dB SIRI in good conditions, but performs better than others in high noise or low microphone spacing. Modifying the trigger for intervention alignment to a more sensitive version creates the new algorithm Cascaded ICA with Demixing Intervention (CICADI). This proves to have better performance, roughly 20 dB SIRI, in good conditions, while still performing well in adverse conditions.

With three or more sources, neither CICAIA nor CICADI produce more than 5 dB of separation. A new framework is developed to separate additional sources using redundant information present in overdetermined setups. The first approach, Inter-frequency Correlation with Microphone Diversity (ICMD), efficiently chooses a set of microphones to produce a determined mixture that provides the best separation. The microphone set is selected at one frequency, and maintained for subsequent frequencies until a new set is necessary. The second approach, ICA with Triggered Principal component analysis (ITP), extracts the principal components of the overdetermined mixture, resulting in a determined mixture with minimal noise. Both approaches utilize an efficient detection algorithm to determine when separation has failed at a given frequency, and corrects the separation at that bin before proceeding to the next. Both algorithms produce roughly 15 dB of SIRI in good conditions.
ACKNOWLEDGMENTS

First, I am grateful to my advisor, Dr. Steven Grant, for giving me the opportunity and direction (and also the funding) to complete my PhD degree at Missouri University of Science and Technology. The road to get here was a little longer than I expected, but the experiences were well worth the time.

To Dr. Kosbar, for early education and practical experiments in communications systems. To Dr. Beetner, for my employment in the EMC, where I participated in a few interesting projects, and gained several useful skills. To Dr. Moss, for my involvement in the dermatology research group, where I could try my hand at signal processing of other signal types. Finally, to Dr. Grow, whose linear algebra assistance was directly responsible for the creation of CICADI. For these actions in particular, and their help in general, I would like to thank the members of my committee.

Dr. Grant and I would like to thank the authors Buchner, Aichner, and Kellermann, of [14]; Pham, Servière, and Boumaraf, of [22]; Parra and Spence of [23]; and Rahbar and Reilly of [25], for generously sharing their source or object code. Having these algorithms made understanding their papers much easier, and made possible the realistic performance evaluations between approaches.

Finally, I would like to thank my friends and family for all they have done in my aid, but it would take far more than the space I have here. My friends, including Matthew Vezeau, Colin Stagner, and the others, were fundamentally essential in helping me unwind after rushing from deadline to deadline. In particular, I am forever indebted to my parents, Robert and Linda Osterwise, and to my loving wife, Cecilia, for their continued support in achieving my goals.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF ILLUSTRATIONS</td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>ix</td>
</tr>
<tr>
<td>NOMENCLATURE</td>
<td>x</td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1. HISTORY</td>
<td>3</td>
</tr>
<tr>
<td>1.2. CURRENT STATE</td>
<td>5</td>
</tr>
<tr>
<td>1.3. INDEPENDENT COMPONENT ANALYSIS</td>
<td>9</td>
</tr>
<tr>
<td>1.4. CICAIA</td>
<td>14</td>
</tr>
<tr>
<td>1.4.1. Frequency Oversampling and Effect on Correlation</td>
<td>15</td>
</tr>
<tr>
<td>1.4.2. Cascade ICA</td>
<td>17</td>
</tr>
<tr>
<td>1.4.3. Intervention Alignment</td>
<td>18</td>
</tr>
<tr>
<td>1.5. OVERDETERMINED CASE</td>
<td>22</td>
</tr>
<tr>
<td>1.5.1. Diversity Combining</td>
<td>23</td>
</tr>
<tr>
<td>1.5.2. Principal Extraction</td>
<td>24</td>
</tr>
<tr>
<td>1.6. QUANTIFYING PERFORMANCE</td>
<td>24</td>
</tr>
<tr>
<td>1.6.1. Environment</td>
<td>24</td>
</tr>
<tr>
<td>1.6.2. Evaluation Criteria</td>
<td>26</td>
</tr>
<tr>
<td>2. SPARSITY</td>
<td>29</td>
</tr>
<tr>
<td>2.1. ALGORITHMS UNDER TEST</td>
<td>30</td>
</tr>
<tr>
<td>2.2. INPUT NOISE</td>
<td>33</td>
</tr>
<tr>
<td>2.3. MICROPHONE SPACING</td>
<td>33</td>
</tr>
<tr>
<td>2.4. SPARSITY</td>
<td>35</td>
</tr>
<tr>
<td>2.5. REVERBERATION</td>
<td>37</td>
</tr>
<tr>
<td>2.6. REAL ROOM EXPERIMENT</td>
<td>38</td>
</tr>
<tr>
<td>2.7. DISCUSSION</td>
<td>40</td>
</tr>
</tbody>
</table>
## LIST OF ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1. System Diagram</td>
<td>1</td>
</tr>
<tr>
<td>1.2. Two Independent Sources</td>
<td>12</td>
</tr>
<tr>
<td>1.3. Observed Mixtures</td>
<td>13</td>
</tr>
<tr>
<td>1.4. Whitened Mixtures</td>
<td>13</td>
</tr>
<tr>
<td>1.5. Separated Outputs</td>
<td>14</td>
</tr>
<tr>
<td>1.6. Magnitude of $\mathbf{R}_{xx}$ with increasing overlap ($K$)</td>
<td>16</td>
</tr>
<tr>
<td>1.7. Beamforming for Ordering Outputs</td>
<td>18</td>
</tr>
<tr>
<td>1.8. Beam pattern for two sources</td>
<td>19</td>
</tr>
<tr>
<td>1.9. Beam Pattern for three sources, three microphones</td>
<td>20</td>
</tr>
<tr>
<td>1.10. Graph of iterations needed in each Frequency Bin</td>
<td>21</td>
</tr>
<tr>
<td>1.11. Calculating $T_{60}$ from a room impulse response</td>
<td>26</td>
</tr>
<tr>
<td>2.1. Performance vs. Noise in various spacings</td>
<td>34</td>
</tr>
<tr>
<td>2.2. Performance vs. RIR Sparsity</td>
<td>36</td>
</tr>
<tr>
<td>2.3. Performance vs. RIR reverberation time</td>
<td>37</td>
</tr>
<tr>
<td>2.4. Simulation versus Observation, with Microphone Spacing of 16 cm</td>
<td>40</td>
</tr>
<tr>
<td>2.5. Simulation versus Observation, with Microphone Spacing of 8 cm</td>
<td>41</td>
</tr>
<tr>
<td>3.1. Performance of CICADI vs. Noise</td>
<td>45</td>
</tr>
<tr>
<td>3.2. Performance of CICADI vs. Sparsity</td>
<td>47</td>
</tr>
<tr>
<td>3.3. Simulation versus Observation on CICADI</td>
<td>48</td>
</tr>
<tr>
<td>4.1. Performance vs. Noise—Three Sources</td>
<td>49</td>
</tr>
<tr>
<td>5.1. Range of condition values vs. one selected from random sample</td>
<td>55</td>
</tr>
<tr>
<td>5.2. Condition of two indicators over frequency, of one mixture of speech</td>
<td>57</td>
</tr>
<tr>
<td>5.3. Position (in meters) of sources and 9 microphones in simulated room</td>
<td>62</td>
</tr>
<tr>
<td>5.4. Performance with 3 sources, 9 microphones</td>
<td>62</td>
</tr>
<tr>
<td>5.5. Performance with 5 sources, 9 microphones</td>
<td>63</td>
</tr>
<tr>
<td>5.6. Measured performance with 5 sources, 10 microphones</td>
<td>64</td>
</tr>
<tr>
<td>5.7. Complexity of Overdetermined Algorithms</td>
<td>67</td>
</tr>
<tr>
<td>6.1. Eigenvalues in an Instantaneous Mixture</td>
<td>69</td>
</tr>
</tbody>
</table>
6.2. Metrics for a $N_s$ estimation (true $N_s = 4$) ............................................................... 71

6.3. Estimated $N_s$ from thirty random frequency bins.................................................... 72
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1. Details of Conditions in Figure 1.10</td>
<td>22</td>
</tr>
<tr>
<td>2.1. Table of Parameters</td>
<td>32</td>
</tr>
<tr>
<td>2.2. Room Dimensions for Sparsity Test</td>
<td>35</td>
</tr>
</tbody>
</table>
# NOMENCLATURE

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSS</td>
<td>Blind Source Separation</td>
</tr>
<tr>
<td>$N_s$</td>
<td>Number of sources</td>
</tr>
<tr>
<td>$N_i$</td>
<td>Number of inputs</td>
</tr>
<tr>
<td>$s_i(t)$</td>
<td>Source Signal $i$</td>
</tr>
<tr>
<td>$x_j(t)$</td>
<td>Observed signal from microphone $j$</td>
</tr>
<tr>
<td>RIR</td>
<td>Room Impulse Response</td>
</tr>
<tr>
<td>$a_{ji}$</td>
<td>RIR from source $i$ to microphone $j$</td>
</tr>
<tr>
<td>$\zeta_j(t)$</td>
<td>Isotropic noise at microphone $j$</td>
</tr>
<tr>
<td>$t$</td>
<td>Sample index</td>
</tr>
<tr>
<td>$s(t)$</td>
<td>Source vector</td>
</tr>
<tr>
<td>$x(t)$</td>
<td>Observed (mixed) signal vector</td>
</tr>
<tr>
<td>$y(t)$</td>
<td>Output vector of reconstructed sources</td>
</tr>
<tr>
<td>$\zeta(t)$</td>
<td>Isotropic noise at microphones</td>
</tr>
<tr>
<td>$A(\tau)$</td>
<td>Mixing Matrix</td>
</tr>
<tr>
<td>$W(\tau)$</td>
<td>Separation Matrix</td>
</tr>
<tr>
<td>$L$</td>
<td>Length of mixing/demixing filters</td>
</tr>
<tr>
<td>$\star_k$</td>
<td>Element of signal/filter at frequency $k$</td>
</tr>
<tr>
<td>$\Pi_k$</td>
<td>Permutation ambiguity matrix</td>
</tr>
<tr>
<td>$D_k$</td>
<td>(diagonal) Scaling ambiguity matrix</td>
</tr>
<tr>
<td>$\Gamma_k$</td>
<td>Postfilter to solve scaling ambiguity</td>
</tr>
<tr>
<td>ICA</td>
<td>Independent Component Analysis</td>
</tr>
<tr>
<td>$Q_k$</td>
<td>Whitening Matrix</td>
</tr>
<tr>
<td>$\hat{x}_k(t)$</td>
<td>Whitened input</td>
</tr>
</tbody>
</table>
\( \tilde{W}_k \) (un-whitened) Separation matrix; separates \( \tilde{x}_k \)

\( W_k \) Whitened separation matrix; separates \( x_k \)

CICAIA Cascade ICA with Intervention Alignment

\( K \) Frequency oversampling factor

PCA Principal Component Analysis

\( T_{60} \) Reverberation time

SIR Source to Interference Ratio

SDR Signal to Distortion Ratio

SIRI SIR Improvement

\( d \) Microphone spacing for \( N_i = 2 \) case

ICMD Interfrequency Correlation with Microphone Diversity

\( R_{xy}(k-1,k) \) Autocorrelation matrix of output between bins \( k-1 \) and \( k \)

\( \kappa(\cdot) \) Condition number of a matrix

ITP ICA with Triggered PCA

\( V_k \) Linear transform matrix for ITP
1. INTRODUCTION

The objective of Blind Source Separation (BSS) is to separate an observed mixture of source signals, without foreknowledge of either the sources, or the mixing environment. It is a long-standing problem that, despite several recent advancements, is far from having a definitive solution.

Figure 1.1 shows the diagram for the application of BSS. $N_s$ statistically-independent sources are present in a room with $N_i$ microphones (algorithm inputs). Each source $s_i(t)$ couples to each microphone $x_j(t)$ by a room impulse response (RIR), modeled as an FIR filter $a_{ji}$, where it is corrupted by isotropic noise $\zeta_j(t)$. The intent of the BSS system is to recover the unknown sources at the outputs $y_i(t)$.

The mathematical model for this is formed by stacking the signals into vectors and performing matrix convolution, as per the following equation:

$$\mathbf{x}(t) = \sum_{\tau=0}^{L_x} \mathbf{A}(\tau) \mathbf{s}(t-\tau) + \zeta(t)$$ \hspace{2cm} (1.1)

and the solution is found by formulating and convolving with a separating filter:

$$\mathbf{y}(t) = \sum_{\tau=0}^{L_y} \mathbf{W}(\tau) \mathbf{x}(t-\tau)$$ \hspace{2cm} (1.2)

where
• $t$ is the sample index
• $s(t) = [s(1, t) \cdots s(N_s, t)]^T$ is the vector of $N_s$ sources,
• $x(t) = [x(1, t) \cdots x(N, t)]^T$ is the observed signal from $N_i$ microphones,
• $y(t) = [y(1, t) \cdots y(N_s, t)]^T$ is the output of reconstructed sources,
• $\zeta(t) = [\zeta(1, t) \cdots \zeta(N_s, t)]^T$ is the isotropic noise at the microphones,
• $A(t)$ is the $N_i \times N_s \times L_A$ mixing matrix, and
• $W(t)$ is the $N_s \times N_i \times L_W$ demixing matrix.

As the system is blind, the values of $s(t)$, $\zeta(t)$, and $A(t)$ are unknown to the separation algorithm; only $x(t)$ is observed. Certain assumptions, however, can be made. The sources are assumed to be statistically independent, and reasonably spatially separate. This assumption is justified in the theoretical condition of the environment: multiple groups conversing within a room. While each conversation will be concurrent with others, each will consist of only one speaker at a time; therefore, the speakers will be spatially distinct. It can be assumed without loss of generality that the mixing and demixing filters are of equal length ($L_A = L_W = L$). So long as $L$ is sufficiently high, the filters will be correctly identified, though additional zeros will be appended. It is also frequently assumed that the number of sources and microphones are equal ($N_s = N_i$); however, as will be seen, this is not always the case. This work will assume both values are already known. $N_i$, a property of the hardware, is easily provided to the BSS algorithm. $N_s$ must be identified by other means. Appendix D covers one such possible method.

This dissertation will cover several advances made in solving the BSS problem. Among these are three new algorithms, including assessments of their strengths and vulnerabilities, and a new framework for describing the challenge of a mixture. The remainder of section 1 will cover the existing state of the literature, and a few terms necessary to describe the challenges and success of algorithms. In addition, it will cover Cascaded Independent Component Analysis (ICA) with Intervention Alignment (CICAIA), a BSS algorithm from which much of this work progressed. Section 2 will
discuss the performance of this and several other benchmark algorithms, drawing the conclusion that a descriptor of sparsity, which is frequency omitted in literature, is necessary. Section 3 discusses a new algorithm based on CICAIA, with improvements under good conditions. Section 4 covers the existing algorithms’ limitations in the presence of additional sources. Section 5 describes and benchmarks two new algorithms that prove capable of separating an arbitrary number of sources in an unknown configuration. Section 6 describes a preprocessor that can estimate the number of sources present in an overdetermined mixture.

Contributions to the field begin with section 2. In particular, these include enhancements to an existing algorithm (CICAIA), a measure of sparsity of room impulse responses, and three new algorithms (CICADI, ICMD, and ITP). The sparsity measure reveals a more relevant measure of the difficulty of a blind mixture. The two last algorithms are capable of separating an arbitrary (known) number of sources in a room with unknown geometry, so long as the microphones outnumber the sources. The preprocessor to estimate $N_s$ is a modification of a post-processor, meant to run after all frequency bins have been separated. Its implementation here allows for ICMD and ITP to be run on a blind mixture where the number of sources is unknown.

1.1. HISTORY

Seeds of source separation were planted as far back as 1953, with a paper from Professor Colin Cherry [1] where he demonstrated that the human ability to separate speech mixtures was a form of statistical processing. He challenged the engineering community to “design a machine whose reaction, in response to speech stimuli, would be analogous to that of a human being” in reference to separation. The gauntlet was not quickly taken up, but was eventually considered an active problem again in the 1980s, with the advent of blind adaptive techniques [2]. The subject fully germinated by the 1990’s, gaining attention in a relatively short time. In 1991 the problem was considered unknown to much of the signal processing community [3], but by 1999 an international workshop arose on the topic [2].

Early interest in BSS arose in two separate fields simultaneously: communications signal processing, and neurology. The communications signal
processing community saw blind mixtures as part of multi-channel communications, including wireless transmission and audio processing. The most famous of these is the “cocktail party problem”: trying to separate the voices of several people speaking simultaneously in the same room. Neuroscientists, on the other hand, found structures in the brain that were designed specifically to separate components of received signals. Naturally, they saw the human brain was capable of solving the cocktail party problem with great efficacy. Another example involves the signals of proprioception, which effect human coordination, including the type (contraction vs. expansion) and rate of muscle movement. These signals are mixed in generation, and are transmitted as such to the motor cortex where they are separated into estimates of speed and direction [2].

Both fields made significant progress on the problem. Pierre Comon, of the signal processing community, analyzed the setup of an instantaneous linear mixture (equation (1.1) with $L_A = 0$), formally defining the problem around 1991 [3]. In his efforts, he found that such a mixture could be definitively solved by the technique of Independent Component Analysis (ICA) [4]. About the same time, Linsker, studying Neural Computing, found that the application of a nonlinear excitation function used in a system designed to calculate principal components led to the extraction of statistically independent components instead [5], [6]. This new Infomax algorithm was originally designed to improve data flow through a computational intelligence structure, but Bell and Sejnowski demonstrated its use in the field of ICA in 1995 [7]. The Hyvärinen fixed-point algorithm for ICA was generated a few years later, in 1997 [8], [9], and is now simply referred to as FastICA. This is described in section 1.3. There are, as a result, many different implementations of ICA to perform separation of an instantaneous mixture. Unfortunately, these do not solve the case of convolutive mixtures, which make them insufficient by themselves for the general case.

As with many similar papers, this work focuses on solving the BSS problem in the context of the cocktail party problem. Here, mixing filters can be very long, on the order of 2000 taps even with a low sampling rate of 8 kHz, and the inputs suffer from the presence of background noise. There have been many proposed solutions to this case alone.
One of the earliest publications on blind source separation in the field of communications [10], [11], attempted to induce separation by diagonalizing cross-correlations for multiple time delays. While this worked well for the instantaneous case, it was proven that such an approach to convolutive mixtures produced solutions that were non-unique [12], and additional effort was necessary. At first, computational complexity made time-domain techniques impractical. Nevertheless, efficient time-domain separation algorithms have arisen. The algorithm of Douglas, Sawada, and Makino [13] extends Hyvärinen’s FastICA algorithm to work at multiple time-lags. The Trinicon family of algorithms [14], by Kellerman, Buchner, and Aichner, exploit the non-whiteness, non-Gaussianity, and non-stationarity of certain source signals (e.g. speech) to solve the uniqueness problem.

1.2. CURRENT STATE

A majority of the algorithms for the cocktail party problem operate in the frequency domain. They apply weighted over-lap add (WOLA) with the short-time Fourier transform (STFT) to the inputs, changing the effective mixing model. Instead of treating the environment as a multiple-input multiple-output system with memory, it allows the system to be handled as a series of instantaneous mixtures at each frequency. Equations (1.1) and (1.2) can then be re-written as

\[
\begin{align*}
  x_k(t) &= A_k s_k(t) + \zeta_k(t) \\
  y_k(t) &= W_k x_k(t)
\end{align*}
\]

(1.3) (1.4)

where \( x_k \) is an element of a signal or filter at frequency bin \( k \).

This simplifies the task of finding coupled \( L \)-length demixing filters in the time domain, to finding \( L \) or more independent demixing matrices, \( W_k \), of \( N_s \cdot N_t \) elements each. Since \( x_k(t) \) is an instantaneous mixture in this model, ICA is perfect for this task. For optimum separation, it is desired to find \( W_k \) such that

\[
W_k A_k = I_{N_s}
\]

(1.5)

where \( I_{N_s} \) is the identity matrix. Unfortunately, as far back as when he defined the problem, Comon [3] noted that the instantaneous case suffers from uncertainty which
made the exact amplitude and order of outputs impossible to resolve. In the frequency
domain, this degrades (1.5) to

$$W_k A_k = \Pi_k D_k$$  \hspace{1cm} (1.6)

for each frequency bin $k$, where $\Pi_k$ is a permutation matrix (a row permutation of the
identity matrix), and $D_k$ is a diagonal (scaling) matrix. This problem is commonly
referred to in two parts: the permutation ambiguity, and the scaling ambiguity. The latter
states that if $D_k$ is not constant over all frequencies, the resulting output will be a filtered
version of the desired signals. A number of different algorithms have been shown to
adequately correct this. Two are filter shortening [15] and filter shaping [16]. The
typical method of solving this is the Minimal Distortion Principle [17], which removes
any additional filtering on the signal produced by the de-mixing algorithm, leaving only
what is applied by the mixing environment. This is accomplished by adding postfilters
$\Gamma_k$ on the separated signals while still in the time-frequency domain, which are
calculated as

$$\Gamma_k = \text{diag}\{W_k^{-1}\}$$  \hspace{1cm} (1.7)

where $\text{diag}\{A\}$ returns the matrix $A$ with all off-diagonal elements set to zero. The
resulting output is then

$$y_i(t) = \Gamma_k W_k x_i(t).$$  \hspace{1cm} (1.8)

Assuming perfect separation in each frequency bin, and temporarily assuming
zero noise, the above equation results in a system where each output $j$ sees only one (the
$i^{th}$) source, filtered by the room impulse response:

$$y(j,t) = \sum_{\tau=0}^{L_j} a_j(\tau) s(i,t - \tau) \text{ for one } i.$$  \hspace{1cm} (1.9)

Handling the permutation ambiguity—ensuring $\Pi_k$ is constant over all $k$—is a
more vital task. If $\Pi_k$ is not the same for all $k$, then the signals are still mixed when
converted back to the time domain. For good separation, the permutation matrices must
be “aligned.” This is still an open problem and the focus of much of the modern
frequency-domain BSS literature. The permutation ambiguity is usually seen as the
limiting factor for the performance of a frequency-domain BSS algorithm in a realistic scenario [18].

In the literature, permutations between frequency bins are often aligned by exploiting known properties of system geometry and/or inter-frequency correlation. Some algorithms [19]-[20] use the microphone geometry to extract direction of arrival (DOA) or time-difference of arrival (TDOA), and align outputs based on these values. These methods work well even in noise. They are also robust against error contamination: that is, one frequency bin that cannot be properly ordered will not negatively impact other frequencies. They do, however, pose restrictions on the geometry of the system, such as requiring a microphone array with minimum dimensions and known spacing, or sources with sufficient spatial distribution. For example, Kurita, Saruwatari, and Kajita [21] evaluate the de-mixing matrix’s response as a beamformer in each frequency bin to align permutations. This requires microphones in a linear array with known spacing. Ono et. al. [20] use an innovative system of matched pairs of microphones scattered about the recording environment, enabling impressive separation of multiple sources by inter-pair time delay of arrival (TDOA). This does, however, require several pairs of microphones, each with matched properties and known spacing.

Other methods [22]-[23] use inter-frequency correlation—the similarity of some features of the signal in different frequencies from a common source—to align the order of the outputs. This principal can be extended further, such as correlating frequency bins with external patterns like lip movement drawn from video [24]. These techniques generally produce good results regardless of system geometry, but are often more susceptible to noise. In addition, because of how the correlation is usually measured, inputs must be long enough to support the statistical independence of the signals’ envelopes. The algorithm of Pham, Servière, and Boumaraf [22] creates “profiles” from the envelope of the output at each frequency bin, and aligns these to an updating global average. Rahbar and Reilly [25] attempt a similar approach, but directly compare the spectrum modulation of adjacent frequency ranges in a dyadic sorting method.

Several recent methods ([26], [27]) exploit both properties to align permutations. They combine the benefits of both methods, but share the system geometry restrictions of the first category as well as the signal length requirements of the second. Sawada, et. al.
[27] directly integrate both DOA estimation and inter-frequency correlation to determine the most probable alignment at each frequency.

All algorithms that rely on inter-frequency correlation, in full or in part, must address the issue that a failed separation or improper output order of one bin can adversely affect the alignment of other bins. In most algorithms, the performance loss is caused by erosion to the basis of comparison. This leads to ambiguity in the cost function for permutation correction, resulting in more misalignment. In CICAIA [26] the effect was obvious: a permutation error in one bin could immediately cascade to the remaining frequencies. Fortunately, CICAIA’s beamforming-based Intervention Alignment detected and corrected these events. Pham, Servière, and Boumaraf [22] loop through the entire frequency range several times, with the intent that as long as the majority of bins are ordered correctly, the remaining will align with sufficient iteration. In Mazur and Mertins’s approach, permutation errors lead to clusters of frequencies with a common alignment. Most of their algorithms ([28], [29]) revolve around marking the bins where separation is expected to have failed, and returning later to align the groups. In a similar fashion, the algorithms in this work share a common trait of detecting when separation has failed; however, when this occurs, the problem is immediately rectified by activating microphone diversity.

Other, recent approaches have attempted to escape the permutation problem altogether. A recent algorithm has been developed known as independent vector analysis (IVA) [30], [31], which is theoretically unaffected by the permutation ambiguity. Instead of disjoint separations at each frequency bin, IVA performs separation on the entire time-frequency representation at once, treating sources as dependent vectors rather than frequency components. Recent improvements to the algorithm have reduced its computational complexity [32], allowing for implementation on mobile platforms [33].

Other innovative approaches construct a framework where the sources have a sparse representation, then utilize recent sparse reconstruction algorithms to produce the sources. These are especially prevalent in attempting to solve the case where the number of sources exceeds the number of microphones. Asaei et. al. [34] approach BSS by splitting a room into a grid of $G$ locations, each of which may or may not have a source. $G$ far exceeds the number of sources, which can exceed the number of microphones.
They then construct a length-$G$ vector of one source per location, and attempt to recover this vector over time. This forms the sources as spatially sparse, and entwines source recovery with geometry estimation. Likewise, making the assumption that signals are sparse in the time-frequency (TF) domain allows for separation by TF masking. Yilmaz [35] uses this as the sole method of separation, while Sawada [36] employs it as just the last of a two-stage separation procedure.

1.3. INDEPENDENT COMPONENT ANALYSIS

In general, ICA is motivated by the central limit theorem: the sum of several independent distributions (e.g. speech) resembles a Gaussian distribution. The inputs to ICA are assumed to be linear combinations of independent sources, and that the combination is invertible. Finding linear combinations of these inputs that look the least Gaussian will result in independent sources, as any other combination that leaves the output still mixed will more closely resemble the normal distribution. As Comon discovered, this technique is sufficient to separate instantaneous mixtures of sources. Speech has a super-Gaussian distribution (meaning its probability distribution function has a narrower central peak and thicker tails), and is easy to extract from an instantaneous mixture via ICA. Unfortunately, recordings of speech in a reverberant environment are convolutive mixtures, not instantaneous. They contain the original sources and their many echoes, leading to far more than $N_s$ signals to be separated, and most of them statistically correlated. Applying the STFT to convert these signals to the time-frequency domain changes the problem into one of independent mixtures, each with only $N_s$ sources.

The four algorithms covered in this work all use Hyvärinen’s Fast ICA [37] algorithm—a specific implementation of the ICA solution—for separating individual frequency bins. Like most ICA algorithms, [37] assumes the input is white and zero-mean, and enforces this by multiplying the input by a whitening matrix $Q_k$. That is,

$$\tilde{x}_k(t) = Q_k x_k(t)$$

s.t. $\mathbb{E}\{\tilde{x}_k\tilde{x}_k^H\} = I$ 

(1.10)
The whitening matrix is calculated via eigendecomposition of the sample autocorrelation matrix of \( x_k(t) \):

\[
R_{xx,k} = \frac{1}{T} \sum_{t=1}^{T} x_k(t)x_k^H(t)
\]

\[
Q_k = \Lambda_k^{-1/2}E_k^H
\]

where \( R_{xx,k} \) is the autocorrelation matrix of \( x_k \), \( E_k \) is a unitary matrix of its eigenvectors, and \( \Lambda_k \) is a diagonal matrix of the corresponding eigenvalues.

The Hyvärinen algorithm then calculates separation vectors \( \tilde{w}_{k,m} \) such that the outputs

\[
y_k(m,t) = \tilde{w}_{k,m}^H \tilde{x}_k(t), \quad m = 1, \ldots, N, \quad (1.12)
\]

are the least Gaussian and have unit variance, and such that the separation vectors are orthogonal. The non-gaussianity is achieved by maximizing the value of a nonlinear function \( G(\cdot) \) applied to \( y_k(m,t) \). The latter two constraints are met by orthonormalizing the vector with respect to the other separation vectors. That is,

\[
\arg \max \sum_{m=1}^{N} \mathbb{E} \left\{ G \left( \tilde{w}_{k,m}^H \tilde{x}_k(t) \right) \right\}, \quad \text{s.t.} \quad \mathbb{E} \left\{ \left( \tilde{w}_{k,i}^H \tilde{x}_k \right) \left( \tilde{w}_{k,j}^H \tilde{x}_k \right) \right\} = \delta_{ij}
\]

where \( \delta_{ij} = 1 \) for \( i = j \) and \( \delta_{ij} = 0 \) for \( i \neq j \). \( G(\cdot) \) is chosen here as \( G(y) = \log(y + \epsilon) \), with \( \epsilon \) being any arbitrary small positive value, (0.1 in the implementation of this work).

The separation vectors are then collected into columns of \( \tilde{W}_k \).

These constraints can be met with a gradient descent approach. The update at each iteration is made by

\[
\tilde{w}_{k,m} = \mathbb{E} \left\{ \tilde{x}_k(t) y_k^*(m,t) g \left( |y_k(m,t)|^2 \right) \right\} - \mathbb{E} \left\{ g \left( |y_k(m,t)|^2 \right) + |y_k(m,t)|^2 g' \left( |y_k(m,t)| \right) \right\} \tilde{w}_{k,m}, \quad (1.14)
\]

where \( g(\cdot) \) and \( g'(\cdot) \) are the first and second derivatives of \( G(\cdot) \), respectively. Here, [37] suggests iterating (1.14) on one \( \tilde{w}_{k,m} \), ensuring unit norm after each iteration, before
moving on to the next vector. This configuration is modified slightly. Instead, each iteration consists of applying (1.14) to every $\tilde{w}_{k,m}$ in $\tilde{W}_k$, then orthonormalizing the matrix via eigendecomposition after each iteration:

$$\tilde{W}_k^H \tilde{W}_k = E_k A_k E_k^H \quad (1.15)$$

$$\tilde{W}_k = \tilde{W}_k E_k A_k^{1/2} E_k^H$$

$\tilde{W}_k$ is considered converged when it does not change significantly between iterations.

The separation matrix $W_k$ is constructed as

$$W_k = \tilde{W}_k^H Q_k \quad (1.16)$$

and the output is formed by

$$y_k(t) = W_k x_k(t) = \tilde{W}_k^H Q_k x_k(t) \quad (1.17)$$

It is standard convention in BSS literature to ignore the creation of the whitening matrix, absorbing it into $W_k$ (which is usually output from ICA). For details of the proposed algorithms, however, the distinction between whitening and separation matrices must be maintained. This work will modify the notation of [37] slightly, and refer to $\tilde{W}_k$ as the (un-whitened) separation matrix, which is calculated by iterations of ICA, and $W_k = \tilde{W}_k^H Q_k$ as the whitened separation matrix, which approximates the inverse of $A_k$.

The behavior of ICA can best be visualized by a simple example. Consider two independent, uniformly distributed variables $s_1$ and $s_2$, in vector $s = [s_1, s_2]^T$, each with probability distribution function (pdf)

$$f(s_1) = f(s_2) = \begin{cases} \frac{1}{2} & -1 \leq s \leq 1 \\ 0 & \text{else} \end{cases} \quad (1.18)$$

Figure 1.2 plots $s_1$ against $s_2$. It is easy to see that having knowledge of the value of $s_1$ gives no insight to the value of $s_2$, and vice-versa. The two variables are independent.
Consider an arbitrary mixing matrix

$$\mathbf{A} = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix},$$

and its resulting mixture $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^T = \mathbf{x} = \mathbf{A} \mathbf{s}$. According to the central limit theorem, as mixtures, $x_1$ and $x_2$ are mixtures of independent variables, they will have pdfs that more closely resemble a normal distribution. The probability distribution of $x_1$ is trapezoid in shape, obtained from the convolution of $2s_1$ and $3s_2$, as

$$f(x_1) = \begin{cases} \frac{x_1 + 5}{24} & -5 \leq x \leq -1 \\ \frac{1}{6} & -1 \leq x \leq 1 \\ \frac{x_1 - 1}{24} & 1 \leq x \leq 5 \\ 0 & \text{else} \end{cases}.$$  \ (1.20)

The distribution of $x_2$ is similar in shape, though with different bounds. Plotting $x_1$ against $x_2$ produces the diamond of Figure 1.3. It is easily visible that $x_1$ and $x_2$ are not independent—knowing the value of one gives information about (specifically, the bounds of) the other.
As per the first stage of ICA, $Q_k$ is calculated, and the observed mixture is whitened to produce Figure 1.4.
In Figure 1.4, the similarity between the whitened inputs and the sources is quite visible. The linear transform to produce separation with only two whitened sources simplifies down to a simple rotation to restore the independence. After this is applied, the two signals become Figure 1.5:

![Separated Outputs](image)

**Figure 1.5. Separated Outputs**

After separation, the outputs are once again independent. However, the two ambiguities have shown their effect. Fixing the output variance to unity changes the range of the signal from $\pm 1$ to $\pm \sqrt{3}$. Also, careful inspection reveals that the output graph is rotated by $90^\circ$ counterclockwise, meaning that $y_2 \propto s_1$ and $y_1 \propto -s_2$.

### 1.4. CICAIA

The idea to immediately correct a suspected permutation error—which came to be known as intervention alignment—came originally from an algorithm developed by Dr. Steven Grant and Dr. Peng Xie [38]. The approach was later finalized as Cascade-initialized ICA with Intervention Alignment (CICAIA) [26]. The algorithm used three
techniques to first inhibit and then correct any permutation errors: frequency oversampling, cascade initialization of ICA, and intervention alignment.

1.4.1. Frequency Oversampling and Effect on Correlation. Frequency oversampling is the process of creating additional bins between existing frequencies. These are equivalent to resolving the discrete Fourier transform at rational non-integer frequency bins. To produce these, we start with the basic DFT equation, (1.21), which shows the traditional method for converting an $N$-length time-domain signal into $N$ frequency bins.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-j2\pi kn/N} \quad \text{for } k = 0, 1, \ldots, N-1$$

Normally, the frequency index $k$ is an integer between 0 and $N-1$. In frequency oversampling, we partition this domain into $M = KN$ bins, using the new index $m$, where $m \in [0, KN - 1]$, and $k = m / K$. We define the integer $K$ as the frequency oversampling factor. This produces the new equation:

$$X_m = \sum_{n=0}^{N-1} x_n e^{-j2\pi mn/KN} \quad \text{for } m = 0, 1, \ldots, KN - 1$$

which we simplify to:

$$X_m = \sum_{n=0}^{M-1} \tilde{x}_n e^{-j2\pi mn/M}$$

$$\tilde{x}_n = \begin{cases} x_n & n < N \\ 0 & \text{elsewhere} \end{cases}$$

This maintains the usefulness of Fast Fourier Transform, as the $\tilde{x}$ vector can be created by simply zero-padding the $x$ vector.

When a signal of length $N$ is converted to the frequency domain using a $KN$-point FFT, it produces a representation that is smoothed in the frequency domain. This smoothing occurs due to points inserted in the frequency response that are interpolations of the surrounding data.

When a signal is taken from the time domain to the time-frequency domain via a series of short-term Fourier transforms (STFT), and this frequency interpolation is exercised, it creates a series of bins with overlapping frequencies. Any cross-section of this data taken at a specific point in time will appear oversampled in the frequency
domain. Since each of these new frequency bins are interpolations of the surrounding bins, each bin is now correlated to its neighbors.

Figure 1.6 shows the correlation between these subbands. An eight-second long speech signal, $s(t)$, was converted into time-frequency domain representation, $s_k(t)$ using a series of STFTs with a Hamm window. For demonstration, the data window length was kept to $L = 8$, and the value of $K$ varied from 1 to 8. Frequency bins after $K(1+L/2)$ were discarded. The auto-correlation matrix $R_{ss}$ was computed from the remaining bins. The images were created by mapping the magnitude of the auto-correlation matrix to pixel brightness: a lighter shade indicates a higher correlation. For the first two images, the frequency bin for each column is labeled.

![Image](image.png)

Figure 1.6. Magnitude of $R_{ss}$ with increasing overlap ($K$)
White squares are of high magnitude.

In the first image ($K = 1$), there is no oversampling in the frequency domain. Since the source file is a speech signal, there is already some inter-frequency dependence. Frequency bins 0 and 1 are correlated, as is bin 3 with both 2 and 4. However, there is effectively no correlation between bins 1 and 2. As frequency oversampling factor $K$ increases, each frequency bin becomes increasingly correlated with its neighbors. At $K = 1$, bins 1 and 2 are statistically uncorrelated. However, at $K = 2$, bin 2 is correlated with 3, which in turn is correlated with 4. It is this statistical correlation between
adjacent frequency bins that will be exploited to reduce the occurrence of permutations during separation.

1.4.2. Cascade ICA. During separation, CICAIA uses the two-stage permutation control unit from [38]. Traditional frequency-domain approaches to BSS initialize ICA with the same matrix in all frequencies. In CICAIA, ICA is initialized at each frequency using the converged de-mixing matrix from the previous frequency bin, which helps ensure it converges to the same order of outputs. The classic argument against this approach is the belief that errors propagate through subsequent frequencies, leading to blocks of incorrectly-permuted frequencies. CICAIA’s separation matrix in every frequency is therefore checked for permutation against the previous bin, and corrected if necessary.

The proposed method utilizes the fixed-point ICA algorithm proposed by Hyvärinen [8], described in detail in section 1.3, with a modification made to determine the complete demixing matrix \( W_k \) at each update (rather than determining \( W_k \) one column at a time). The separation matrix is declared to be converged when it stops changing “significantly” after each iteration—that is, when the magnitude of the update vector drops below a threshold. When this occurs, the demixing matrix can be compared to the previous frequency bin to determine permutation.

To determine if the demixing matrix has been permuted, a simple, quick-to-compute metric is necessary. As stated, there is some cross-correlation between most neighboring frequency bins of a real-world signal. Because of this correlation, the coefficients of separation matrices in adjacent frequency bins will not change significantly. The magnitude of distance between these coefficients thus provides a simple, quick, and efficient distance metric to determine whether a permutation exists. This is defined as:

\[
d_{CICAIA}(k) = \frac{1}{N_s N_r} \sum_{i,j} \left| \frac{w_{i,j; k}}{\|W_k\|_2} - \frac{w_{i,j; k-1}}{\|W_{k-1}\|_2} \right|
\]  

(1.24)

where \( w_{i,j; k} \) is the element of \( W_k \) in the \( i^{th} \) row and \( j^{th} \) column, and \( \|\cdot\|_2 \) is the induced \( \ell_2 \) (spectral) norm. If this metric is above a threshold, a permutation is suspected, and an
intervention alignment step is taken. If $d_{\text{CICA}}(k)$ is small enough, the output for that frequency is considered aligned, and the process moves to the next frequency.

1.4.3. Intervention Alignment. The intervention alignment step is based on Kurita’s beamforming technique [21]. This treats the $W_k$ as a collection of beamforming vectors, and using knowledge about the geometry of the microphone array, determines the direction of arrival (DOA) of each source. The sources are then sorted based on this estimate.

Consider the case of two sources recorded by a microphone array (Figure 1.7).

The beam pattern of an array is a measure of its gain as a function of a signal’s input direction. When the demixing matrix at any frequency is suspected of having the wrong output order, the beam patterns of each separation vector (each column of $W_k$) are calculated. Example beam patterns are shown below, in Figure 1.8. These were calculated with two input sources (such as depicted in Figure 1.7) and two microphones, using the separation matrix at a frequency of 975 Hz.

ICA produces a separation matrix designed to extract one source while suppressing others. The first demixing vector admits the source from $65^\circ$ and suppresses the source from $110^\circ$, while the second does the converse. At higher frequencies, the
beam pattern becomes much more complex. Still, the pattern follows as per the above example: the effect of the suppression—the null at a particular angle—is far more visible than the actual “peak.” Additionally, the peak will not always align perfectly with the DOA of a source; however, DOAs from suppressed sources will be near a trough. Therefore, the order of outputs is chosen not by the angle of the best gain, but by the angles at which other sources are suppressed.

Unfortunately, with the current usage, this does not scale well with additional sources. Consider the case of three microphones and three sources, shown in Figure 1.9. Here, the three sources are at 45°, 90°, and 135°, respectively. The permutation is properly aligned so that the \( j \)-th output corresponds to the \( j \)-th source, but the level of separation is not very high (i.e., low SIR at this output frequency), especially for the first and third sources. As a result, the gain at 45° for the first output is only slightly higher that at 90° or 135°; similarly for the third output. If the outputs were ordered as order of troughs (like Figure 1.7), source 2 (Separation Vector 2) would be the first output, followed by source 1 and then 3. However, if the trough at 135° dropped below that at 45°, it would be re-ordered to output 3. The peaks at each arrival angle are not well-
enough defined to be useful for permutation alignment. The only way to resolve the ambiguity is with *a-priori* knowledge of the angles of the sources. Indeed, existing algorithms that align permutations based on beamforming usually perform an analysis over the large frequency ranges or multiple sets of microphones to generate statistics, then align the outputs based on these [21]. This approach is incompatible with the intervention alignment paradigm.

![Figure 1.9: Beam Pattern for three sources, three microphones](image)

The combined effects of frequency oversampling and cascade initialization (CI) help drastically reduce the processing time of the algorithm. To demonstrate, a two-source, two-microphone simulation was run, tracking the number of ICA iterations
necessary to perform separation in each frequency bin. The algorithm was run once without CI, once with CI and no oversampling, and once with CI and an oversampling factor of 8. The iterations needed for each bin in the three conditions are displayed in Figure 1.10, and these results are tallied in Table 1.1.

![Graph of iterations needed in each Frequency Bin](image)

Each iteration of ICA is a significant number of calculations, and most algorithms then require applying permutation alignment algorithms that are even more complex. More in-depth analysis of computational complexity is postponed until section 5.4, but in summary, reducing the number of ICA iterations necessary by cascade initialization significantly reduces the overall processing time of the algorithm. Meanwhile, while frequency oversampling increases the number of ICA iterations slightly, its significant
reduction in suspected permutations means less intervention alignment, which leads to further reductions in processing time.

<table>
<thead>
<tr>
<th>Initialization</th>
<th>K</th>
<th>ICA Iterations</th>
<th>Suspected</th>
<th>Actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>a) Static</td>
<td>1</td>
<td>2207</td>
<td>379</td>
<td>225</td>
</tr>
<tr>
<td>b) Cascade</td>
<td>1</td>
<td>1297</td>
<td>264</td>
<td>0</td>
</tr>
<tr>
<td>c) Cascade</td>
<td>8</td>
<td>4530</td>
<td>51</td>
<td>0</td>
</tr>
</tbody>
</table>

1.5. OVERDETERMINED CASE

A special case of blind source separation exists when the number of microphones exceeds the number of sources \( N_j > N_r \). In this scenario, the additional information present provides several options for separation. Publications on classical BSS algorithms greatly outnumber those on overdetermined blind source separation (OBSS); however, the latter is actually more common in application. Consider the following: determined BSS algorithms assume an equal number of sensors and sources. In practical applications, a hardware realization with \( N_j \) inputs will be able to separate up to \( N_j \) active sources. However, most of the time, there will be fewer active sources than the system is designed to handle. During this time, classical BSS algorithms discard the extra inputs. OBSS algorithms, on the other hand, take advantage of the extra information.

In 1999, Westner and Bove [39] showed that elements of the time-domain separation matrix \( W(\tau) \) (from equation (1.2)) have a simpler structure and less energy if the mixture is overdetermined. This facilitated its calculation in the time domain. As a result, while classic BSS is usually handled in the time-frequency domain, a disproportionate number of OBSS algorithms are strictly time-domain approaches [40-43]. Because of the convolutive mixtures, these methods do not utilize ICA in the
traditional sense. For the general frequency-domain approach, the task becomes how to best convert the information from the overdetermined mixture to one that can be processed by complex ICA. The two most common methods for this are diversity combining, and principal extraction.

1.5.1. Diversity Combining. This is a proven technique used to improve the reliability of a wireless connection. It is the concept of using more antennas than signals, and selecting a subset of antennas from which to receive. This is accomplished either by switching between antennas when the current one fails, actively polling all antennas to see which has the best signal, or some more complex form of constructing an output from the available measurements. In diversity’s simplest form, multiple antennas are distributed so that if a transceiver loses sight of one, it can connect with another.

Microphone diversity is a direct application of this technique to the audible frequencies. It is the first method considered here for exploiting the overdetermined case. Assuming the number of sources $N_s$ is known, at each frequency only $N_s$ microphones are fed to ICA to form the output. By this approach, an algorithm can use one of the $\binom{N_m}{N_s}$ sub-matrices of $A_k$ that are better conditioned. This is equivalent to choosing one set of $N_s$ microphones out of the $N_m$ available.

One problem that arises is how to determine which set of microphones to use. Sawada, et al, use a geometric approach [44] to change from wide microphone spacing at low frequencies, to narrow spacing at higher frequencies. Koutras [45] and Zhang [46] simultaneously process all combinations of microphones for each frequency. The former creates a weighted sum from the outputs of each collection; while the latter uses intra-frequency correlation as a measure of independence to select a single set to use for the output. Since the number of possible combinations grows rapidly with the number of microphones, these last two algorithms are both very computationally complex. Zhang attempts to reduce the complexity by only examining sets of microphones with inter-microphone spacing different from the one currently in use. Koutras does not lower the complexity, but proposes a parallel hardware structure to handle it.

Section 5.1 will cover the proposed algorithm Inter-frequency Correlation with Microphone Diversity (ICMD), a far more efficient method to extract, from an
overdetermined case, mixtures that are well-suited for separation. It uses cascade-
initialized ICA (persisting from CICAIA [26]) to reduce output permutation changes,
aligns permutations using energy profiles (which are an efficient permutation alignment
mechanism developed in [22]), and supplements this with a new (low-complexity) direct
cross-correlation method to ensure the current bin has successfully separated. Energy
profiles form the core mechanism by which permutations are aligned for this algorithm
and the next; as a result, they and the algorithm for which they were defined are
described in Appendix A.

1.5.2. Principal Extraction. Microphone diversity reduces $N_i$ inputs down to
$N_s$ signals by discarding those with the worst input mixtures; Principal Extraction
performs the same reduction by linear transform, keeping the best components of all the
input mixtures. A more elaborate explanation of Principal Component Analysis (PCA) is
included as Appendix C. Joho, Mathis, and Lambert [40] demonstrated that after
applying PCA to a noisy instantaneous over-determined mixture, the first $N_s$ principal
components form a determined mixture of the sources with sufficient information for
separation, but with less noise. This approach was applied to a convolutive mixture in
[47], and later [48]. The general approach is to apply PCA to the $N_m$ signals in each
frequency bin to calculate the $N_s$ principal signals, and then apply ICA to them to
produce the separated output.

Many of the advances shown in ICMD can be applied to this approach; resulting
in the new algorithm ICA with Triggered PCA (ITP). This will be covered in section 5.2.

1.6. QUANTIFYING PERFORMANCE

Understanding the effectiveness of a blind source separation task requires
comparing the success of separation versus its difficulty.

1.6.1. Environment. How difficult it is to separate a mixture of speech signals
depends less on the voices themselves and more on the environment in which they were
mixed. By most classical BSS techniques, two similar voices at two distant locations are
easier to separate than two very different voices close to each other. Specifically, three
things need to be known about the environment in order to determine the difficulty of separation.

The first is the reverberation time \( T_{60} \), measured in milliseconds, which measures the time required for echoes to attenuate by 60 dB. This is affected both by the size of the room and the reflective surfaces inside of it. The larger the room, and the more reflective the floor, ceiling, and walls are, the higher the reverberation time. In a real room, this can be measured by specialized equipment. In simulations, this is calculated from the room impulse response. In a process called reverse integration, the RIR is reversed in time, squared, numerically integrated, time-reversed again, and converted into dB scale to form the *Schroeder plot*, \( \zeta_j(t) \). Specifically, this is calculated from impulse response \( a_{ji}(t) \) by:

\[
\zeta_j(t) = 10 \log_{10} \int_{0}^{L} a_{ji}^2(\tau) d\tau.
\]

The slope of \( \zeta_j(t) \), after the direct path, determines how fast the echo decays by 60 dB. This delay, in milliseconds, is the \( T_{60} \). Often, an estimated or simulated impulse response is not sufficiently long to observe a linear decay of 60 dB. As a result, the \( T_{60} \) is frequently approximated as twice the length of time for echoes to decay by 30 dB. The reverberation time of a room should be relatively constant regardless of where in the room it is measured; therefore, the \( T_{60} \) of a simulated room is the average of the reverberation times of all component RIRs.

In Figure 1.11, the Schroeder plot, calculated from the example impulse response, decays by 30 dB in 0.15 s (150 ms); therefore, its \( T_{60} = 300 \) ms. Note that due to decay, the simulated RIR is rounded to zero after 200 ms; this is the reason for the Schroeder plot’s rapid falloff at that index.

Real-life recordings always consist of some background noise. Thermal noise on sensors, wind, and indistinct sources add together at the microphones to form isotropic noise. Equation (1.1) models this with the additive \( \zeta(t) \) term. Its strength is usually expressed as signal-to-noise ratio (SNR), and typical values for it range between 30 and 0 dB.
Finally, environment geometry has an impact on the difficulty of a mixture. It is possible to have a large room with acoustically absorbent walls have the same $T_{60}$ as a small room with bare walls, and these two will produce speech mixtures with very different results—as section 2.4 will show. Therefore, to completely describe a mixture, a description of the physical dimensions of the environment and surface absorption coefficients should be included.

It bears mentioning that these details—$T_{60}$, SNR, and room dimensions—describe how difficult it is to separate a mixture. These should not be provided to the BSS algorithm itself, or else the algorithm would no longer be blind. For such an algorithm, the only inputs should be the observed mixture, the expected number of sources, and for some algorithms, the geometry of the sensors.

1.6.2. Evaluation Criteria. Several performance indices have been proposed for quantifying the performance of BSS algorithms, however the Signal Separation Evaluation Campaign (SiSEC) [50] helped establish a standard of four metrics [51], of which only two are essential. These criteria are useful in that they apply to all mixtures and algorithms. Moreover they do not require knowledge of the mixing or demixing filters—only the source and output signals ($s(t)$ and $y(t)$) are necessary, which make
them easy to calculate. All four are measured as powers relative to the strength of the ideal separated output, and are labeled in decibels.

The performance criteria in [51] are based on the assumption that every input is a mixture of spatial images of sources, \( s_{ij}^{\text{img}}(t) \):

\[
x(i,t) = \sum_{j=1}^{N_s} s_{ij}^{\text{img}}(t).
\]

(1.26)

where each spatial image is the source passed through its associated RIR:

\[
s_{ij}^{\text{img}}(t) = \sum_{\tau} a_{ji}(t-\tau)s(j,t).
\]

(1.27)

The output from a BSS algorithm is the estimate of the source image \( \hat{s}_{ij}^{\text{img}}(t) \), and can be considered a combination of the signal from the intended source, other sources (including isotropic noise and residual sources), and artifacts introduced by the algorithm [52]. This is represented as:

\[
\hat{s}_{ij}^{\text{img}}(t) = \hat{s}_{ij}^{\text{img}}(t) + e_{ij}^{\text{spat}}(t) + e_{ij}^{\text{artif}}(t) + e_{ij}^{\text{interf}}(t).
\]

(1.28)

where \( e_{ij}^{\text{spat}}(t) \), \( e_{ij}^{\text{artif}}(t) \), and \( e_{ij}^{\text{interf}}(t) \) are respectively the error components caused by filtering distortion, artifacts, and interference.

From these, three energy ratios follow: the source Image to Spatial distortion Ratio (ISR), the Signal to Artifacts Ratio (SAR), and the Signal to Interference Ratio (SIR).

The first two are source Image to Spatial distortion Ratio (ISR), which measures the relative power of environmental effects (e.g. reverberation), and Source to Artifacts Ratio (SAR), which measures the relative power of distortion added by the BSS algorithm. These are infrequently reported. Their exact formulations are:

\[
\text{ISR}_j = 10\log_{10} \frac{\sum_{i=1}^{N_s} \sum_{t} s_{ij}^{\text{img}}(t)^2}{\sum_{i=1}^{N_s} \sum_{t} e_{ij}^{\text{spat}}(t)^2}.
\]

(1.29)

\[
\text{SAR}_j = 10\log_{10} \frac{\sum_{i=1}^{N_s} \sum_{t} \left[ s_{ij}^{\text{img}}(t) + e_{ij}^{\text{spat}}(t) + e_{ij}^{\text{artif}}(t) \right]^2}{\sum_{i=1}^{N_s} \sum_{t} e_{ij}^{\text{artif}}(t)^2}.
\]

(1.30)

The Source to Interference Ratio (SIR) is one of two criteria more commonly used. It describes the power left over from other sources, and is a good measure of the actual separation performance of the algorithm. Outputs with a high SIR will sound like
only one source, whereas low SIR outputs will still resemble multiple voices at once. It
is formulated as such:

$$SIR_j = 10 \log_{10} \frac{\sum_{i=1}^{N_j} \sum_t \left[ s_{ij}^\text{img} (t) + e_{ij}^\text{spat} (t) \right]^2}{\sum_{i=1}^{N_j} \sum_t \left[ e_{ij}^\text{interf} (t) \right]^2}. \quad (1.31)$$

The Signal to Distortion Ratio (SDR) is the combined power of all previous
effects relative to the strength of the desired output—an overall gauge of the total error in
the output, and a rough gauge of its acoustic “quality.” Its calculation is therefore:

$$SDR_j = 10 \log_{10} \frac{\sum_{i=1}^{N_j} \sum_t s_{ij}^\text{img} (t)^2}{\sum_{i=1}^{N_j} \sum_t \left[ e_{ij}^\text{spat} (t) + e_{ij}^\text{interf} (t) + e_{ij}^\text{artif} (t) \right]^2}. \quad (1.32)$$

Fortunately, SiSEC has also published simple functions to calculate these metrics. All results described in this work use these functions to estimate performance.
2. SPARSITY

As stated in section 1.6.1, the difficulty of separating a blind mixture is dependent more on the environment than the sources themselves. In almost all BSS literature, the environment for any simulations or measurements are described with at least its $T_{60}$. In some instances, the SNR of the mixture is given as well, though many cases assume zero noise.

In experiments with multiple simulations, it was frequently seen that environments differing in geometry, but similar in other aspects, would lead to very different results. The problem is in the descriptor for reverberation time. $T_{60}$ describes how long it takes for the energy to decay 60 dB below its original strength. However, this energy may arrive as a few strong echoes, or groups of weaker echoes—with the latter producing a more detrimental effect on performance. This spacing of echoes is best quantified by a measure of the impulse response’s sparsity.

To quantify the differences in RIR sparsity, Hoyer’s Sparseness Parameter [53] of an $L$-length vector $h$ is employed:

$$S(h) = \frac{\sqrt{L - \|h\|_2} / \|h\|_2}{\sqrt{L - 1}}$$  \hspace{1cm} (2.1)

where $\|\cdot\|_1$ and $\|\cdot\|_2$ represent the $\ell_1$ and $\ell_2$ norms, respectively. This parameter ranges from 0 if the input vector is all of the same magnitude, to 1 if there is only one non-zero element to the vector.

Intended for use in sparse coding of data, this parameter is applied here to better classify the environment of the separation. The sparseness of a vector is not affected by the absolute value of the largest element, but merely its magnitude relative to the rest of the elements. With the assumption, then, that the direct path from source to sensor has the highest magnitude, the sparsity of an impulse response is determined by the strength and abundance of reflections. Weaker reflections lead to a more sparse result. Likewise, reflections that are more spread out in time create a more sparse impulse response.

The sparseness parameter for a single RIR is determined by applying (2.1) to the 100-millisecond-long window that starts with the direct path. The average of these
among the impulse responses in each simulation is used as the *sparsity measure* for the room.

Equation (2.1) is applied only to a 100 ms window of the impulse response to improve the dynamic range of the measure; it is assumed that after this window, the reflections contribute less to the measure than the raw length of the impulse response. As a phenomenon seen at the microphones, and not attributable to any one source, isotropic noise does not contribute to the sparseness of an environment.

This sparseness measure is a more useful metric than $T_{60}$ in predicting the difficulty of separation. Indeed it is possible to have a large $T_{60}$, with a high sparseness, that is not much more difficult than purely additive combinations.

The addition of a sparsity measure creates four numerical parameters of an environment that might affect BSS performance: noise, reverberation, microphone spacing, and sparsity. To determine the influence of these on performance, five algorithms were tested in the two-source two-microphone (2x2) setup, under various simulated conditions. The tests were performed using simulated environments. Room impulse responses were generated using the image model, recorded samples of human speech were used for sources, and Gaussian white noise was added to the pickups of each simulated microphone. The microphone array and sources were grouped at the center of each simulated room. Each source was 1 m away from the center of the microphone array, and 90° away from each other (equivalent to $\theta_1 = 45°, \theta_2 = 135°$ in Figure 1.7).

Performance of the algorithms were quantified by examining the signal-to-distortion ratio (SDR) of the outputs, and by comparing the difference between the signal-to-interference ratio (SIR) at the output to that at the input to get the SIR Improvement (SIRI).

### 2.1. ALGORITHMS UNDER TEST

The algorithms examined are Cascaded ICA with Intervention Alignment (CICAIA) (from section 1.4); Trinicon [14]; the algorithms of Pham, Servière, and Boumaraf [22]; Rahbar and Reilly [25]; and Parra and Spence [23].

The algorithm of Parra and Spence is the earliest of those considered. The authors demonstrate that demixing filters with permutations between frequencies are longer in the time domain, and thus less smooth in frequency. A projection operator
calculated from the time domain is applied to enforce the smoothness of the frequency response. This projection operator, alternating with the gradient, is applied to the demixing matrix during separation. The result is a separation stage that converges with a consistent permutation of outputs.

The algorithm of Pham, Servière, and Boumaraf relies on the inter-frequency correlation of signals. This algorithm uses “profiles,” which it defines as the logarithm of the power spectrum at each frequency bin, for one output, over time. Two profiles that come from the same source should be similar. Rather than compare frequencies directly, however, the profiles of the output are averaged across frequency to form a target, and then each frequency bin is compared to the resulting mean. After one pass, the global average is computed again, and the individual frequencies are again compared. This is iterated until the output converges. A more detailed description can be found in Appendix A.

Rahbar and Reilly’s algorithm is similar in appearance to that of Pham, Servière, and Boumaraf. The Spectrum Modulation—the power spectral density of each output at each frequency over time—is used to determine the relation between outputs at different frequencies. The permutations for the entire output are aligned by dyadically consolidating frequency bins: adjacent frequencies are paired, and then grouped into sets of four bins, and so on, until there is only one set. At each grouping, the spectrum modulations are compared directly by calculating their correlation, the outputs are reordered, and the spectrum modulations for each output of the new group are composed of the average across the frequencies in the set.

Trinicon is the name given to a family of time-domain algorithms which separate mixtures by attempting to undo the cross-correlations of the inputs. Since the signals are never converted to the time-frequency domain, the family is immune to the permutation ambiguity. The specific implementation used in this paper is created by Aichner, Buchner, and Kellermann. It is an online algorithm, meant to process real-time inputs. As a tradeoff, the first few seconds of output are not separated, and the algorithm requires time to converge before noticeable separation is achieved. To compare this Trinicon implementation with the other algorithms, the system was allowed to converge over the entire length of the input, the output was discarded, then the final mixing matrices were
used to separate the original mixture, and the separation performance was computed on the result.

CICAIA is described in more detail in section 1.4. In addition, a “Nonblind” algorithm is shown. This process uses the ICA stage of CICAIA, but then uses nonblind information (specifically, the algorithm’s performance on the same configuration when only one source at a time is active) to create a “perfect” permutation alignment. This is a rough estimate of CICAIA’s theoretical maximum performance. Other algorithms may exceed this value due to differences in their implementation of ICA.

The parameters for each of these algorithms were set equal where applicable (e.g. filter lengths, % overlap), or if not possible, set to the algorithm’s default. The full list of parameters is included in Table 2.1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>CICAIA</th>
<th>Trinicon</th>
<th>Pham</th>
<th>R&amp;R</th>
<th>P&amp;S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference</td>
<td>§1.4</td>
<td>[14]</td>
<td>[22]</td>
<td>[25]</td>
<td>[23]</td>
</tr>
<tr>
<td><strong>W</strong> Filter Length</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>Time Overlap</td>
<td>75%</td>
<td>75%</td>
<td>75%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FFT Size</td>
<td>8192</td>
<td></td>
<td></td>
<td></td>
<td>4096</td>
</tr>
<tr>
<td>Sampling freq.</td>
<td>8 kHz</td>
<td>8 kHz</td>
<td>8 kHz</td>
<td>8 kHz</td>
<td>8 kHz</td>
</tr>
<tr>
<td>Input Duration</td>
<td>8 sec</td>
<td>8 sec</td>
<td>8 sec</td>
<td>8 sec</td>
<td>8 sec</td>
</tr>
<tr>
<td>Block Length</td>
<td>31</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iterations/block</td>
<td>300</td>
<td></td>
<td></td>
<td></td>
<td>1000</td>
</tr>
<tr>
<td>Step Size</td>
<td>0.01</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># of Cross-Spectral Density Matrices</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Samples per epoch</td>
<td>2000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Matrices to diagonalize</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Intervals used to estimate cross-power</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>
2.2. INPUT NOISE

Multiple aspects of the algorithms in the comparison were tested using simulated mixtures. All source signals used were eight-second long samples of speech, sampled at 8 kHz. All tests used two sources and two microphones. The impulse response from each source location to the microphone array was generated using the image model, using different room dimensions and reflectivity constants. No mixing filter exceeded 3000 samples in length. Additional white Gaussian noise was applied to the pickup of each microphone when the mixed signals were generated.

The algorithms were tested in a simulated room $6 \times 5 \times 3$ meters, with a reverberation time of $T_{60} = 260$ ms. The microphones were spaced 16 cm apart (parameter $d$ in Figure 2.1). The isotropic noise added to the microphones was increased with each simulation, starting at 50 dB below the mixture strength.

Figure 2.1 shows the results of this trail, and the two conducted in section 2.3. As shown in part (a), each algorithm’s separation performance suffered once noise reached a certain level, but the robustness of each varied. At low noise levels, Pham’s algorithm performed best, but by 30 dB SNR, it began to decline. Rahbar & Reilly also performed well, but with slightly less separation than Pham. It began to fall off earlier (~35 dB SNR), but kept pace with Pham otherwise. Though initially providing less separation, CICAIA continued to perform at the same strength with increasing noise until 5 dB SNR. Indeed, at very low SNR, CICAIA performed almost at its theoretical maximum.

Figure 2.1 (b) shows the signal-to-distortion ratio of the various algorithms under the same conditions as Figure 2.1 (a). CICAIA and Pham have the highest initial SDR. Of the two, the latter starts to decline at around 30 dB SNR, while the former lasts until 20 dB SNR.

2.3. MICROPHONE SPACING

The previous experiment was repeated twice more, once with a microphone spacing ($d$) of only 8 cm, and once with a spacing of only 4 cm. The results are displayed in parts (c)-(f) of Figure 2.1.
As the microphone spacing decreases, almost all algorithms become more susceptible to noise. Pham’s algorithm in particular loses significant noise immunity: at $d = 16$ cm, it maintains 15 dB of SIRI until 20 dB of SNR, but by $d = 4$ cm, it drops below this level by 40 dB of SNR. Above 40 dB SNR, the algorithm of Rahbar & Reilly seems to have the same SIRI regardless of microphone spacing, but at any higher noise, its separation performance drops with decreasing microphone distance. At odds with the
rest of the algorithms, Trinicon’s separation changes very little with microphone spacing. CICAIA also loses some noise immunity, but actually gains maximum separation with decreasing microphone spacing. As a result, at the smallest spacing (4 cm), CICAIA produces the best separation of any of these algorithms at any SNR.

As the microphones were placed closer together, the maximum quality of the output (in terms of SDR) of most algorithms also suffered. However, their susceptibility to noise did not change significantly. With the microphones placed closer together, Rahbar & Reilly lost roughly 4 dB of maximum SDR, and Pham lost roughly 7. CICAIA, however, gained 2 dB of SDR, gaining an impressive lead over the rest of the algorithms. Again, Trinicon did not change significantly with microphone spacing.

2.4. SPARSITY

The five algorithms were again tested in the two-source scenario with two microphones spaced 16 cm apart. The isotropic noise added to the microphones was kept 20 dB below the level of the signal. A series of five rooms were simulated, descending in size, but with reflection coefficients scaled to maintain $T_{60}$ at roughly 250 ms. The details of the rooms have been included as Table 2.2. For comparison, room 1 is about the size of a large conference room; room 4 is about the size of a typical office. The end result was a series of room impulse responses with the same reverberation time, but with decreasing sparseness.

<table>
<thead>
<tr>
<th>Room</th>
<th>Dimensions (m)</th>
<th>$\beta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8 x 8 x 4</td>
<td>0.65</td>
</tr>
<tr>
<td>2</td>
<td>8 x 6 x 4</td>
<td>0.695</td>
</tr>
<tr>
<td>3</td>
<td>6 x 5 x 3</td>
<td>0.769</td>
</tr>
<tr>
<td>4</td>
<td>4 x 3 x 3</td>
<td>0.853</td>
</tr>
<tr>
<td>5</td>
<td>2.6 x 2.6 x 2.5</td>
<td>0.89</td>
</tr>
</tbody>
</table>

The value of $\beta$ represents the reflection coefficient of all walls and ceiling; the floor was given a reflection coefficient of 0.5.
Figure 2.2 shows the performance of the algorithms in the five rooms, and their measured sparsity. For comparison, the sparseness values of the first experiment (Figure 2.1.A) were between 0.514 and 0.534.

Figure 2.2.A and C imply a good correlation between the sparsity of a room and an algorithm’s separation performance. In terms of SIR improvement, Pham’s algorithm performed the best regardless of the sparsity. In the sparsest case, Trinicon actually did better than any algorithm except Pham; however, as the sparsity of the room decreased, its separation performance diminished rapidly, until it had the second-worst performance.

Figure 2.2.C show there is a similar impact on the quality of the separated speech. As before, the best algorithm in terms of SDR is CICAIA. Interestingly, Pham also performs well in this regard, maintaining the second-highest quality throughout.

Figure 2.2. Performance vs. RIR Sparsity

(a) SIRI vs. room #  (b) SDR vs. room #  (c) Sparsity and $T_{60}$ vs. room number
2.5. REVERBERATION

The five algorithms were tested in the presence of two sources, with two microphones spaced 16 cm apart. The isotropic noise added to the microphones was kept 20 dB below the level of the signal. A series of eleven rooms were simulated with increasing reflection coefficients, and slightly increasing size. The resulting rooms all had impulse response sparsity close to 0.7, but with $T_{60}$ beginning at 80 milliseconds and increasing exponentially up to 960 ms.

Figure 2.3 shows the performance of the algorithm as compared to the RIR reverberation time. Parts (a) and (b) show the SIRI and SDR of each room, respectively, while part (c) shows the reverberation coefficient for the corresponding rooms.

Interestingly, despite the fact that the reverberation time increases by an order of magnitude, no algorithm loses more than 10 dB of separation performance, and most
algorithms lose only about 5. Here, Pham proves to be the best algorithm in terms of
degree of separation, starting with 22 dB of SIRI and leading by 3 ~ 5 dB across the
board. CICAIA starts at the second-highest with about 17 dB of separation, which only
diminishes slightly with increasing T_{60}, but it drops below the performance of Rahbar &
Reilly when the reverberation time increases beyond 130 ms.

The rise in reverberation time also had minimal effect on the quality of output on
the algorithms. Most algorithms only lost about 5 dB of SDR over the course of the
trials. Consistent with previous experiments, CICAIA shows the best quality of output,
with 14 dB of SDR at the output in the first room. Pham consistently maintains the
second-highest SDR, approximately 1 dB below CICAIA under all T_{60} conditions.

These experiments confirm that sparsity is a more significant measure of the
difficulty of separation performance than is T_{60}.

2.6. REAL ROOM EXPERIMENT

To confirm the accuracy of the simulations, the five algorithms were tested using
real recordings made with varying levels of isotropic noise and varying microphone
spacing. Mixtures were recorded using the Immersive Audio Environment [54] in Rolla,
Missouri. This chamber uses amplitude panning over 60 loudspeakers to recreate audio
sources coming from arbitrary directions.

To record the mixture, two (additional) speakers and two microphones were
placed in the center of the chamber in the same configuration as previous simulations:
microphones 16 or 8 centimeters apart, and speakers 90° apart and one meter away from
the microphones, (again equivalent to $\theta_1 = 45^\circ, \theta_2 = 135^\circ$ in Figure 1.7). The remainder
of the speakers in the chamber played white noise, to effect isotropic noise at the
microphones. To change the SNR, the speech was held constant, and the strength of the
white noise was varied.

A technique using autocorrelation of Golay codes calculated the impulse response
between the primary speakers and the microphones. From the estimated impulse
response, the reverberation constant was found to be approximately 550 ms, and the
rooms sparsity measure was found to be roughly 0.77. The testing room is much larger
than any of the previous simulated environments, thus slightly different behavior than in
previous simulations is expected. Therefore, the algorithms were tested in a new simulation. Due to the shape of the actual environment, it could not be perfectly digitally reproduced, but the simulation is close in dimension and resulting parameters. The simulated room measures 18 by 15 by 5 meters, has a reverberation time of 530 ms, and a sparsity of 0.630. The results of the simulation are included as Figure 2.4 and Figure 2.5 for the microphone spacings of 16 and 8 centimeters, respectively.

Parts A and B of the figures show the performance of the algorithms in the simulated environment. In general, the algorithms performed as expected. Under good (i.e. low noise) conditions, Pham produced the best SIR improvement. As noise increased, the performance of all algorithms declined. With strong isotropic noise, CICAIA performed the best. In terms of SDR, all algorithms performed about equally well, with CICAIA and Pham being the two best results.

By contrast, in parts C and D of Figure 2.4, most of the algorithms (CICAIA, Trinicon, Rahbar & Reilly) have a maximum performance that is lower in practice than it is in theory. In particular, Rahbar and Reilly, showed the second-best low-noise separation in the simulation, but in actual trials it tied with Parra & Spence for the worst. CICAIA did worse than expected during low-noise conditions, but actually outperformed the simulation in high-noise environments. The observed maximum SDR of all algorithms was 5 dB lower than expected, with the exception of Trinicon, which was 10 dB lower than expected. This appears only to affect the maximum SDR, however. For each algorithm, the observations match the simulated results below the SNR where the performance starts to degrade.

In the 8 cm microphone spacing, shown in Figure 2.5, there are slightly more visible differences, but the behavior of the simulation is still an accurate model for the observed results. In the simulation, Pham is the best algorithm for noise levels below 20 dB of SNR; the actual results show that Pham outperforms CICAIA until closer to 15 dB of SNR, but the difference in SIRI is less than 2 dB.

According to the simulations, Trinicon should not suffer with decreased microphone spacing, however Figure 2.4 and Figure 2.5 suggest the algorithm is more susceptible than simulations indicate. The observed maximum SIRI is only 7 dB, far lower than the 17 dB seen in the simulations. As with the previous trial, the observed
maximum SDR of all algorithms is 5 dB lower than the simulated performance or 10 dB lower in the case of Trinicon.

2.7. DISCUSSION

Section 2.2 shows all algorithms degrade with increased noise, but to different extents. The change in performance of Rahbar and Reilly, and of Pham, stems from the inter-frequency correlation used to ensure permutation alignment. With low noise, outputs are distinct, and comparing correlation between frequencies provides sufficient information to align the permutations. However, increasing noise degrades this correlation, lowering the efficacy of this technique. By contrast, increased noise affects the CICAIA algorithm by triggering the intervention algorithm more frequently. However, the TDOA technique used in intervention permutation alignment is dependent
on the geometry of the scenario, and is less affected by noise. Thus, at low SNR, the algorithm continues to perform well.

It is generally considered that increased microphone spacing can improve the performance of a BSS algorithm, since the wider-spaced microphones share less mutual information. By contrapositive, BSS algorithms perform worse on microphone arrays with narrower spacing. Section 2.3 shows this is the case for most of the considered algorithms. The algorithms of Pham, Rahbar and Reilly, and Parra and Spence all decrease in separation performance and output quality as the microphone spacing decreases. Two of the algorithms show exception to this, however. Trinicon does not significantly change with microphone spacing, maintaining roughly 15 dB of SIR Improvement and 8 dB of SDR under low noise conditions. CICAIA, on the other hand, actually improves with diminished spacing. The algorithm suffers from spatial aliasing at higher frequencies in a wider-spaced microphone array [18]. This reduces the
effectiveness of the TDOA intervention step, lowering the algorithm’s performance. CICAIA’s best performance in the tests shown is with a microphone spacing of 4 cm. This is roughly the same spacing as the width of a smartphone (in portrait orientation), which makes CICAIA perfect for such applications.

CICAIA prevents permutations during the separation stage by cascading initialization, and attempts to prevent any errors from propagating by utilizing an intervention step for permutation alignment. It has decent separation in the best conditions, the best separation in most of the bad circumstances, and the best quality of outputs under all scenarios. It shows remarkable noise immunity in small microphone arrays, making it useful for practical applications such as handheld platforms in non-ideal settings. In addition, the algorithm’s technique of applying alignment steps only when necessary make it very efficient, and as such it ran faster than any other algorithm on test under low- to moderate-noise conditions.

Pham’s algorithm generates energy profiles from separated outputs, and uses the similarity between these to re-order outputs in each bin, without directly computing cross-correlation matrices between each frequency. This creates excellent low-noise separation, decent quality of output, and decent noise immunity. The algorithm relies only on inter-frequency correlation, so there is no restriction on the geometry of the environment. This makes it ideal for distributed or widely-spaced microphone setups.

It is generally accepted that reverberation time of an environment strongly affects the performance of separation algorithms. However, despite its prevalence, describing a mixture’s environment using only the $T_{60}$ is insufficient. Section 2.4 shows variations in sparsity may lead to dramatic changes in performance, while section 2.5 shows that an order-of-magnitude change in reverberation affects performance to a much lesser extent. To properly describe a mixing environment, the sparsity measure must be included along with $T_{60}$, or else the description is incomplete.
3. CICADI

In tests comparing it against established benchmarks, CICAIA showed exceptional performance in “bad” conditions—high noise, high reverberation, and low sparsity—but performed poorly in “good” conditions. It was determined during further testing, that while intervention alignment using beamforming could successfully order the outputs, it was not always triggering on output order changes from bin to bin. A new method of detecting permutation changes was therefore necessary.

The new technique uses a different component of the demixing matrix to detect possible permutations. It is named Cascading ICA with Demixing-matrix Intervention (CICADI).

3.1. EVOLUTION OF TRIGGER

The evolution from CICAIA to CICADI arose from exploiting the nuances in the mechanism of separation. Like its progenitor, CICADI utilizes the fixed-point ICA algorithm proposed by Hyvärinen [37], described in section 1.3.

In realistic environments of blind mixtures, there are physical constraints to the change of $A_k$ over frequency. As a direct result, $W_k$, which when properly aligned is proportional to $A_k^{-1}$, is similarly restricted in change over frequency. For this reason, CICAIA used its gradient as a distance metric to flag permutations. However $W_k$ is the product of the separation matrix and the whitening matrix, $\tilde{W}_k^H Q_k$, and since $Q_k$ is dependent on source data, its norm is not strictly regulated. For source data such as speech, without uniform spectral excitation, the average $\|W_k\| = \|\tilde{W}_k^H Q_k\|$ will drift over frequency. This makes a direct threshold comparison of $W_k - W_{k-1}$ impossible, hence the normalization step in (1.24).

We can eliminate this need if we consider the factorization more closely. The Banach Inequality states that the norm of the product of two matrices is at most the product of their norms. If one or both of the matrices is unitary, the bound becomes equality. $Q_k$ is computed from the source data; its norm is not constrained. In its
construction, $\tilde{W}_k$ is orthonormalized after each iteration. It will always have unit norm, and so we can consider

$$\|W_k\| = \|\tilde{W}_k^H Q_k\| = \|Q_k\|$$  \hspace{1cm} (3.1)

Equation 1.8 then becomes

$$d_{\text{CICAIA}}(k) = \frac{1}{N_i N_s} \sum_{i,j} \frac{w_{i,j,k} - w_{i,j,k-1}}{\|Q_k\|_2} - \frac{\|Q_{k-1}\|_2}{\|Q_{k-1}\|_2}$$ \hspace{1cm} (3.2)

$Q_k$ is also deterministic, and not subject to permutations. Therefore, it should not be included when detecting permutation changes. What remains is comparing only the unwhitened demixing matrices, which do not require normalization. As a result, the triggering metric for the new algorithm becomes

$$d_{\text{CICADI}}(k) = \|\tilde{W}_k - \tilde{W}_{k-1}\|_F$$ \hspace{1cm} (3.3)

The change from CICAIA to CICADI also involves the use of the Frobenius norm ($\|\|_F$), instead of a straight average of elements. If we assume sufficient frequency oversampling, then the $\tilde{W}_k$ matrices of adjacent frequencies with the same output order should be almost equal. The task of comparing a demixing matrix to that of the previous bin is then approximately equivalent to comparing the demixing matrix to a column-permuted version of itself.

$$\tilde{W}_k - \tilde{W}_{k-1} \approx \tilde{W}_k - \tilde{W}_k \Pi$$ \hspace{1cm} (3.4)

Substituting (3.4) into (3.3) and simplifies the latter into:

$$\|\tilde{W}_k - \tilde{W}_{k-1}\|_F \approx \|\tilde{W}_k - \tilde{W}_k \Pi\|_F = \|\tilde{W}_k\|_F \|\Pi\|_F$$ \hspace{1cm} (3.5)

The matrix $(\mathbf{I} - \Pi)$ can only adopt certain discrete values: each element is either 1, -1, or 0, and each row sums to zero. Here, the benefit of the Frobenius norm becomes clear: the right hand side of (3.5) becomes

$$\|\mathbf{I} - \Pi\|_F = \sqrt{2P}$$ \hspace{1cm} (3.6)

where $P$ is the number of rows that are permuted (0, 2, 3, etc.).
In practical implementation of CICADI, however, some differences between the correctly-ordered $\tilde{W}_k$ and $\tilde{W}_{k-1}$ will occur. Therefore, the strict threshold of $\sqrt{4}$ that follows from (3.6) is not used for $\lambda$ in (3.3); the more relaxed threshold of 1.0 is adopted instead. In nonblind simulations with a frequency overlap of $K=8$, this threshold proved to accurately detect all injected permutations, while falsely triggering on less than 1% of the remaining bins.

3.2. PERFORMANCE

The performance of the new algorithm was tested in many of the same conditions as its predecessor (in section 2). As before, most of these tests were simulations.

First, the performance of CICADI against noise and microphone spacing was tested. The results are included in Figure 3.1. (For clarity, some of the benchmark algorithms were omitted. For their performance, see Figure 2.1.)

Figure 3.1. Performance of CICADI vs. Noise
CICAIA’s primary failing proved to be wide-spaced microphone arrays and low noise levels. Figure 3.1 shows that the new algorithm makes up for these deficiencies. With wide microphone spacing and at low noise levels (≥ 35 dB SNR), the proposed algorithm and Pham both produce the best separation, at roughly 17 dB of SIRI. At noises slightly higher than this, CICADI performance remains constant, while that of Pham degrades. At the highest noise, CICADI performs the best, with CICAIA second-best.

With an array spacing of only four centimeters, CICADI shows improvement over Pham and CICAIA alike. In low noise, the gain is 3 dB over other algorithms. This advantage grows to 10 dB over Trinicon and 15 dB over Pham in moderate levels of noise. In extremely high noise (≤ 5 dB SNR), all algorithms perform equally poorly.

Regardless of configuration, CICADI produces the best quality sound, by a slight gain in SDR over CICAIA, and a marked improvement over Trinicon. In wide spacing and low noise, it only provides a slight advantage over Pham and CICAIA—roughly 1 dB. However, in narrow microphone spacing, its separation quality is consistently about 7 dB better than either Pham or Trinicon.

3.3. SPARSITY

Next, the new algorithm’s robustness to sparsity was examined. The five rooms of section 2.4 were used with the new algorithm, to produce Figure 3.2.

In the effects of sparsity, CICADI is an incremental advancement on CICAIA. With high microphone spacing, it shows 3 dB more SIRI than CICAIA at high sparsity, and roughly the same resistance to its decline. At narrower microphone spacing (not shown), the performance is equal in rooms 1 and 2, but CICADI shows improved resistance to decreasing sparsity, ending in 4 dB additional separation in the least sparse rooms. In terms of SDR, CICADI and CICAIA are again of almost identical quality. Using a widely-spaced microphone array, they have only a couple dB more SDR than Pham; in the narrowly-spaced array, they show 7 dB of advantage. In both cases, they showed roughly 7 dB of SDR gain over Trinicon.
Finally, CICADI was tested on the mixtures generated in the real room measurement of section 2.6. Only the microphone spacing of 8 cm was processed. The results are in Figure 3.3.

Again, the measurements show a better high-noise performance than the simulations suggest, and the observed performance of Trinicon was lower than what was predicted. Other than that, the simulations remain a good model for the algorithm performance.

3.5. DISCUSSION

In many ways, Cascading ICA with Demixing-matrix Intervention (CICADI) is an incremental improvement over CICAIA. Noise immunity tests (Figure 3.1) show that the new algorithm overcomes its predecessor’s faults in wide microphone spacing and in
low noise, while maintaining the overall noise resilience and the good performance at low microphone spacing. Other tests (Figure 3.2 and Figure 3.3) show equal performance or incremental gains in similar conditions. These improvements usually stem from additional calls to the intervention alignment routine, though the cost in processing time is slight. This puts it on or slightly-above par with conventional techniques in terms of SIR, and much better in terms of SDR.

Figure 3.3. Simulation versus Observation on CICADI
4. THREE SOURCES

The tests in sections 2 and 3 show decent separation performance from several algorithms, in various conditions of noise, microphone spacing, reverberation, and microphone sparsity. In most of these, Pham and CICADI prove to be the two best algorithms. However, all of these experiments share one condition: all had only two sources.

This, inherently, is not uncommon. The literature documenting all of the included benchmark algorithms described exclusively the two-source condition. Until recently, BSS literature focused primarily on the two-source scenario, leaving the generalization of the existing algorithm to additional sources to the reader. This is not always a trivial task.

The benchmarks presented earlier were tested again in the three-source, three microphone (3x3) scenario. The simulated room measured 8 x 6 x 4 meters, with a reverberation time of $T_{60} = 260$ ms and sparsity of 0.590). The three simulated microphones were placed in a line, 8 cm apart. The three sources were placed 1 m away from the center of the array, 45° away from each other (equivalent to $\theta_1 = 45^\circ$, $\theta_2 = 90^\circ$, $\theta_3 = 135^\circ$ in Figure 1.7). The results of a noise sweep created Figure 4.1. Note that Trinicon could not be compared; its code was not open-source, and was restricted to only two sources. Pham’s code had similar limitations, but was open-source, and the documentation was sufficiently clear. It was modified to allow for additional sources.

Figure 4.1. Performance vs. Noise—Three Sources
As Figure 4.1 shows, none of the algorithms performed well in this condition. Additional experiments confirmed this trend. According to the nonblind approach, in low noise conditions, 15 dB of separation should have been possible. The closest performance was that of Pham’s algorithm, which only produced 8 dB—a significant reduction from the 17 dB produced with only two sources.

Further investigation in this scenario revealed that several frequency bins—far more than the two-source scenario—failed to separate. In fact, due to the smoothness of real impulse responses in the frequency domain, one “bad bin” usually accompanied several others consecutively. The achievement of the nonblind approach suggested that the fault in the CICA family stems not from these failures, which were properly detected, but from alignment of the permutation errors that followed. The problem stems from beamforming’s approach in ordering outputs, and its inability to scale with sources. With only two sources, outputs can be ordered based only on which null precedes the other. With more sources, proper alignment requires some estimation of the sources’ DOA based on additional frequency bins—which is incompatible with the intervention alignment paradigm.

Pham’s algorithm performed better than any of the other benchmarks, but far worse than similar performance with only two sources. Close examination suggested that its alignment mechanism would have been better able to sort the additional outputs, had the global average not been corroded by abundant bad bins. Unfortunately, the defects in the average profiles resulted in several permutation misalignments going unchecked.
5. OVERDETERMINED METHODS

In the previous section, the algorithms under test proved incapable of handling more than two sources. In particular, CICADI proved unable to align permutations with greater than three sources present. It would be possible to modify the beamforming technique to allow for separation of more than two sources. Doing so, however, would probably require constructing the beampattern of additional frequencies, and would significantly increase the algorithm’s complexity. The easiest solution is to utilize the permutation alignment method of Pham; however, this places additional emphasis on solving the bad bin phenomena. Great effort must be made to ensure frequency bins properly separate. This is where the overdetermined setup proves useful.

Consider again the frequency-domain BSS solution:
\[ y_k(t) = W_k x_k(t) \] \hspace{1cm} (1.4)

In the (classic) critically-determined paradigm, when \( A_k \) is ill-conditioned, separation at that frequency will fail, and overall performance will suffer. In the overdetermined case, there is additional information present in the observed mixture that may be used to improve the separation. Two techniques were developed to utilize this setup to address the bad bin problem. The first method, discussed in section 5.1, uses microphone diversity and simply switches to a different set of sensors when the current set encounters a bad bin. It differs from conventional uses of microphone diversity by more efficiently detecting the occurrence of the bad bin and also by using a new, efficient method to find a microphone set with good separation properties. The second technique utilizes principal extraction, but mitigates its complexity by exploiting frequency domain smoothness, recognizing that the required unitary matrix for PCA does not always need to be recalculated for consecutive bins. Instead, we detect when a new unitary matrix is required and only calculate it then. Interestingly, this also improves its performance.

5.1. ICMD

As stated in section 1.5.1, microphone diversity chooses a subset of microphones on which to perform separation for the given frequency. The most pressing challenge in
this is to determine which subset of microphones will produce the best output. Many algorithms handle this by trying all subsets.

The first overdetermined algorithm, Inter-frequency Correlation with Microphone Diversity (ICMD), uses cascaded ICA initialization to reduce output permutation changes [26], aligns permutations using energy profiles [22], and supplements this with a new direct cross-correlation method to ensure the current bin has successfully separated. If the latter is suspect using the current set of microphones, microphone diversity is exploited to find another set that will separate better. That is, as the frequency bins are sequentially processed, the current set of microphones is held constant, until, in a particular frequency bin, it is necessary to change it. This is in direct contrast to [45] and [46]. Furthermore, when a new microphone set must be chosen, it is sufficient to consider only a small number of random selections. Also, a low-complexity selection criterion that avoids performing ICA on all those considered is used to find the best set. The result is a significant decrease in computational complexity, with low processing time, good separation performance, and improved output quality.

5.1.1. Triggering Microphone Diversity. As with all frequency domain algorithms, the inputs from all microphones are first transformed into the frequency domain using a windowed STFT. Assuming that the number of sources $M$ is known, we take as many microphones to be a set, and treat them as though handling the determined case.

Mazur and Mertin’s $\alpha$-algorithm [28] is “based on the observation, that false alignments usually happen at positions where separation performance is poor.” In these circumstances, the outputs at one frequency bin will resemble more than one output of the previous bin—there will be no simple one-to-one correlation and thus no mapping can be made. Their solution is to monitor the angle between the demixing vectors, and mark suspect bins where this angle is too small. In a similar vein, Zhang and Chambers [46] select which microphone set to use based on the independence of each pair of output vectors (quantified by their cross correlation). These approaches work for two sources, but do not scale well with more because the number of checks increases with the square of the number of sources, and the method to combine multiple decision variables is unspecified.
Let us define $R_{yy}(k-1,k)$ as a normalized sample cross-correlation matrix of the magnitude of the de-mixed signals in consecutive bins. First consider the vector of output signals, acquired from equation (1.4),

$$y_k(t) = \begin{bmatrix} y_k(1,t) & \cdots & y_k(N_\tau,t) \end{bmatrix}^T$$

where $y_k(m,t)$ is the $m^{th}$ output in frequency $k$ at time sample $t$. The estimated standard deviation of each signal $y_k(m,t)$ is

$$\hat{\sigma}_{y_k}(m) = \sqrt{\frac{1}{T} \sum_{t=1}^{T} |y_k(m,t)|^2}.$$ (5.2)

where $T$ is the number of samples in each frequency bin. The normalized output signal is

$$v_k(t) = y_k(t) / \hat{\sigma}_{y_k}(m).$$ (5.3)

Then,

$$R_{yy}(k-1,k) = \sum_{t=1}^{T} |v_{k-1}(t)||v_k^T(t)|.$$ (5.4)

When the output signals in both bins are well separated (statistically independent) and the same permutation exists in both bins, then $R_{yy}(k-1,k)$ will be strongly diagonally dominant and, because of the normalization in (5.3), also well-conditioned. If, on the other hand, there is a permutation change in the outputs between bin $k-1$ and bin $k$, $R_{yy}(k-1,k)$ will still be well-conditioned, but permuted in the same way that $y_k(t)$ is permuted from $y_{k-1}(t)$. Furthermore, if $y_k(t)$ is not well separated then two or more rows (and columns) of $R_{yy}(k-1,k)$ will be linearly dependent, giving it a high condition number. The proposed algorithm monitors $\kappa(R_{yy}(k-1,k))$, the condition number of $R_{yy}(k-1,k)$, and uses it as a watchdog for when a reliable permutation mapping between bins cannot be made. For the experiments below, the threshold was chosen to be 10. When $\kappa(R_{yy}(k-1,k))$ exceeds this threshold, a new set of microphones is selected. That set is then used in the subsequent frequency bins until the threshold is again exceeded.
5.1.2. Finding a New Microphone Set. It would be possible to select the new set of microphones based on the best (lowest) value of $\kappa(R_{yy}(k-1,k))$. In creating the output for a frequency bin, the calculation of $R_{yy}(k-1,k)$ and its condition number is straightforward. However, when selecting a new set of microphones, using this as a metric would involve running ICA on every potential collection of microphones. Even with the performance gains shown by cascade initialization [26], this can be quite computationally expensive.

As stated, a sufficient condition for ICA to fail at a particular frequency bin is an ill-conditioned mixing matrix. Consider equation (1.3). If $A_k$ is ill-conditioned, the demixing matrix $W_k$ will be unable to separate the signals properly, and so permutation alignment will fail.

The task, then, is to find a set of microphones to use for the current frequency with the best-conditioned mixing matrix. To test all potential collections of microphones, as per [46] and [45], would produce the optimal solution; however, the computational complexity of this approach increases as a polynomial of the number of available microphones. Fortunately, it is usually sufficient to choose the best of a random sample of microphone sets.

To demonstrate, the impulse responses from eleven sources to nine microphones were simulated using the image model. The resulting condition number at each frequency for each set $\kappa(A_k)$ was calculated. At each bin, of the 330 potential sets, ten were randomly selected, and the best condition number from the sample was selected. Figure 5.1 shows the range of condition numbers of all possible sets in black, and the condition number of the best set of the sample plotted in red on top. Only frequencies processed by ICMD are displayed, for clarity. The optimal set is the bottom of the black curve. The best of the randomly selected sets is usually not the optimal set, but it is quite close.

The state of $A_k$ for a given microphone set would be the ideal metric to determine the set’s usefulness. Obviously, the algorithm cannot directly inspect $A_k$ due to the blind nature of the system. Fortunately, any arbitrary matrix has the same
condition value as its inverse—assuming the latter exists. Were \( W_k \) a perfect inverse of \( A_k \), it would make a perfect substitute to monitor. Unfortunately, the two ambiguities of ICA result in (1.6). Ignoring isotropic noise for a moment, this admits the following substitution:

\[
y_k(t) = W_k A_k s_k(t) = \Pi_k D_k s_k(t).
\]  

(5.5)

Recall that ICA makes all outputs independent, and the Hyvärinen implementation in particular scales them to have unit variance. That is,

\[
E\{y_k y_k^\text{H}\} = I.
\]  

(5.6)

Combining (5.5) and (5.6) reveals the relationship between the sources and the scaling ambiguity:
\[ E\{y_k y_k^H\} = I = E\{\Pi_k D_k s_k s_k^H D_k^H \Pi_k^H\} \]
\[ = \Pi_k D_k R_{ss} (k) D_k^H \Pi_k^H \]
\[ \Pi_k^H \Pi_k = D_k R_{ss} (k) D_k^H \]
\[ I = D_k R_{ss} (k) D_k^H \]

(5.7)

As in all cases of BSS, the sources are assumed to be statistically independent—i.e. \( R_{ss} (k) \) is diagonal. So,

\[ D_k = R_{ss}^{1/2} (k) \]

(5.8)

When this is substituted into the ambiguity equation (1.6),

\[ W_k = \Pi_k D_k A_k^{-1} \]
\[ \kappa \{W_k\} = \kappa \{D_k A_k^{-1}\} \]
\[ = \kappa \{R_{ss}^{1/2} (k) A_k^{-1}\} \]

(5.9)

the scaling ambiguity prevents \( W_k \) from being a perfect estimate of \( A_k^{-1} \); however, it is determinate, and affected by the sources, not the mixing environment. \( R_{ss} (k) \) changes with each frequency, but does not change with selection of microphones. Therefore, \( \kappa (W_k) \) can be monitored as a proxy for \( \kappa (A_k) \), to select the best set of microphones for a given frequency bin. Unfortunately, this would again require running ICA on each potential collection.

Section 1.3 shows the breakdown of how \( W_k \) is made. As the norm of \( W_k \) is largely dictated by \( Q_k \), so too is its condition number. As a result, \( \kappa (Q_k) \) is used as the metric for fitness of a potential microphone set, with the advantage that there is no need to run ICA for every candidate mixture.

Figure 5.2 parts (a) and (b) show condition numbers for \( Q_k \) and \( R_{yy} (k-1,k) \), respectively, for an example mixture of speech. The vertical scale of (b) has been limited to 140, even though the maximum value actually extends to almost 2000.

Derived directly from the output of ICA, the condition of \( R_{yy} (k-1,k) \) indicates when separation has failed at that bin. The condition of the whitening matrix, on the other hand, is dependent on the effective mixing matrix as well as the relative excitation.
of sources at that frequency. Due to the non-uniform excitation of signals such as speech, and due to the shape of $\kappa(A_k)$ (seen in Figure 5.1), there will be a frequency-dependent scaling factor whose shape cannot be easily predicted. Therefore, while $Q_k$ can provide an estimate of the best microphone set to select, it should not be used as a triggering
metric. Any threshold applicable to higher frequencies would trigger the diversity routine on every bin below a certain frequency. Furthermore, the shape of the trend cannot be predicted in a blind setting, so it cannot be easily corrected.

The final diversity combining algorithm used in ICMD: monitors $\kappa(\mathbf{R}_{yy}(k-1,k))$ to determine when a new set of microphones are necessary, uses random sampling to reduce the number of candidate sets considered, and uses $\kappa(\mathbf{Q}_k)$ as a quickly-computed statistic to decide which of the candidates to use. This approach substantially reduces the computational load compared to algorithms that process all possible combinations of microphones at all frequencies, such as [45] and [46]. In addition, it requires no knowledge of the geometry of the system.

5.1.3. Profiles. Energy profiles, defined in [22], are based on the concept that frequency components of wideband sources (such as speech) will have amplitudes over time similar to the envelope of the wideband signal itself, but with a frequency-dependent scaling factor. Separated outputs in each frequency bin can be properly ordered by comparing their energy profile to target profiles.

In [22], the target profiles of each output are constructed by continually updating the average profile of that output over all frequencies. This average is improved as alignment is performed over the entire frequency range in successive iterations. This is described in more detail in Appendix A. In ICMD, since additional steps are taken to ensure separation in all bins, it is sufficient to create the target profiles $\mathbf{e}^j(t)$ from a moving average of the previous $B$ bins. For a typical system described in the simulations, $B \approx 100$ is sufficient.

It would be possible to eliminate the energy profiles altogether and align permutations based solely on $\mathbf{R}_{yy}(k-1,k)$ in a manner somewhat similar to [28]. Doing so, however, would make each frequency bin dependent only on the immediately previous one. This would require sufficient excitation in all frequencies in all sources. If two sources had a common null frequency, where microphone diversity could not repair the error, the system would be irrecoverable. Using profiles as described above allows for a basis of comparison that is more robust to local perturbations.
To sum up, the proposed ICMD algorithm performs the following steps at each frequency bin,

1) Extract \( y_k(t) \) via Cascade-Initialized ICA

2) Calculate the cross-correlation matrix \( R_{yy}(k-1,k) \)

3) If \( \kappa(R_{yy}(k-1,k)) \) exceeds a threshold:
   a. Select a random sample of microphone sets
   b. Choose the set with the smallest \( \kappa(Q_k) \)

4) Extract \( y_k(t) \) from the new set

5) Align the permutation of bin \( k \) according to the energy profiles of the past \( B \) bins

5.2. ITP

As stated in section 1.5.2, principal extraction reduces \( N_i \) inputs down to \( N_s \) signals by means of a linear transform, which results in sufficient information for separation with less noise. This again assumes \( N_s \) is known \textit{a priori}. The general approach is to apply PCA to the \( N_i \) signals in each frequency bin to create the \( N_s \times N_i \) linear transform matrix \( V_k \). That is, singular value decomposition is applied to the input autocorrelation matrix to produce the left and right singular values (\( U \) and \( \bar{V} \), respectively) and the singular values (\( \Sigma \)):

\[
R_{xx,k} = \frac{1}{T} \sum_{t=1}^{T} x_k(t)x_k^H(t),
\]

\[
R_{xx,k} = U \Sigma \bar{V}^H \quad (5.10)
\]

\( V \) is composed of the first \( N_s \) columns of \( \bar{V}^H \). \( V \) is then used to create the \( N_s \) principal signals by

\[
\hat{x}_k(t) = V_k^H x_k(t) .
\]

(5.11)

To improve performance and decrease the computational complexity somewhat, techniques used in ICMD can be applied to this approach as well. The resulting
algorithm is called ICA with Triggered PCA (ITP). Specifically, the optimum transform \( V_k \) is not recalculated for every frequency bin. Because of the smoothness in the frequency domain, most adjacent frequency bins may use the same \( V \) without ill-effect. In fact, section 5.3 shows that if \( V \) is changed only when necessary, both separation and overall quality are improved. This is due to the fact that adjacent frequency bins have slightly different optimal transforms and using those transforms produces slightly different mixtures in the principal signals of each bin. This slightly affects the scaling in each bin which in turn affects the performance of the minimum distortion algorithm [17] used to resolve the scaling ambiguity. Clearly, keeping the same transform from bin to bin is not optimal in the PCA sense, but the resulting decrease in “mixture jitter” appears to be more important.

In ITP, an optimal linear transform is calculated in the first frequency bin; this transform is then fixed for subsequent frequencies, until it requires updating. To detect when the transform needs to be recalculated again utilizes the condition number of \( R_{xy}(k-1,k) \). This indicates when the output signals of bin \( k \) are insufficiently separated. The calculation of \( \kappa(R_{xy}(k-1,k)) \) for every bin is considerably less complex than calculating \( V_k \), as the latter requires the eigen-analysis of the \( N \times N \) autocorrelation of the \( N \) inputs. Therefore, the ITP algorithm shows improvement in computational complexity, as demonstrated in Section 5.4; however, as performance evaluations will show (Section 5.3), its real benefit comes from the improvement in output quality.

It is worth noting that ITP uses the same cascade-initialized ICA and permutation alignment as in ICMD. Its primary difference is in the mechanism of converting overdetermined mixtures to determined ones.

In summary, the steps taken at each subsequent frequency bin \( k \) are:

1) Create \( \hat{x}_k(t) = V_k^T x_k(t) \)

2) Extract \( y_k(t) \) from the \( N \) signals of \( \hat{x}_k \) via cascade-initialized ICA

3) Calculate the cross-correlation matrix \( R_{yy}(k-1,k) \)

4) If \( R_{yy}(k-1,k) \) is ill-conditioned, re-calculate \( V \) and re-run steps 1) and 2)
5) Align the permutation according to the energy profiles

5.3. PERFORMANCE OF ICMD AND ITP

Several tests were conducted to examine the performance of the proposed overdetermined algorithms. The first two were simulations using the image derived model [55] and the last used a recording in a real office. In addition to the proposed algorithms, two others are included as benchmarks. Given its performance in earlier trials [26], the algorithm of Pham, Servière, and Boumaraf [22] is included. In ITP’s precursor, [47], $V_k$ is calculated in every frequency bin. This approach is tested here in an algorithm called PCA-ICA, though the permutation alignment technique used is the same as ITP and ICMD.

5.3.1. Simulations. Impulse responses from a simulated 6 m by 9 m by 2.5 m room (roughly the size of a classroom), with a reflectivity constant of 0.810, were constructed using the image model. The resulting room impulse responses (RIRs) had a reverberation factor of $T_{60} = 440$ ms, and a sparseness factor of 0.450 [56]. Three sources were scattered throughout the room, and nine microphones were clustered near the center. Eight-second-long recordings were made using a sampling frequency of 8000 Hz. Isotropic noise was added to the microphone pickups during simulation, varying the input SNR. Figure 5.3 shows the position of sources and microphones in the simulation.

The algorithms were first tested in simple conditions with only two sources and low background noise. Under these conditions, all four algorithms performed well, showing 16 to 23 dB of SIRI.

When the third source was added, the performance of all algorithms dropped, to some extent. The results are included as Figure 5.4. The initial performance of Pham dropped from 18 dB to only 7 dB—in agreement with section 4. The third source did not excessively affect Pham’s noise immunity; however, as performance did not degrade until the input noise reached 10 dB below the signal. In good conditions, ICMD produced the best separation at a SIRI of 14 dB. However, the PCA-based algorithms produced the best separation overall. ITP showed a consistent 1~2 dB gain over PCA-
ICA. For this configuration, ITP also seemed quite robust to noise, and as such still produced 11 dB SIRI when the noise was as strong as the sources.

In terms of separation quality, ICMD outperformed all the others in high SNR conditions. It maintained roughly 9.5 dB of SDR for most of the trial, compared to

Figure 5.3. Position (in meters) of sources (•) and 9 microphones (○) in simulated room

Figure 5.4. Performance with 3 sources, 9 microphones
Pham’s 4.2 dB. Again, ITP was consistently better than PCA-ICA by several dB on average. Its noise immunity resulted in output quality better than ICMD when the SNR dropped below 10 dB.

The experiment was repeated with five sources, to view the flexibility of the algorithm. The results are shown in Figure 5.5. Of the four algorithms, the ITP trend changed the most. Both PCA-ICA and ITP showed a greater deterioration in performance with increasing background noise—at 0 dB SNR, no algorithm produced greater than 5 dB of SIRI. In low noise, ICMD still had the best separation, which ITP only exceeded when the SNR dropped below 20 dB. The SDR result shows that ICMD and ITP both displayed the best separation quality, but the gap between the two diminished compared to the three-source tests.

![Figure 5.5. Performance with 5 sources, 9 microphones](image)

**5.3.2. Real Room Measurement.** Using multi-channel audio hardware from the Immersive Audio Environment [54], a hardware setup for BSS was constructed in a classroom-like setting. The room measured 6.7 by 9.1 by 2.6 meters. A ring of desks outlined the room, with associated office chairs. Sixteen loudspeakers were scattered about the room; five of which played a recorded sample of speech, while the other 11 were used to inject white noise. Ten microphones were placed near the center of the room, with no set configuration. In lieu of individual suspensions, microphones were
supported on acoustically-absorbent foam pads. Impulse response estimates showed the room to have a reverberation time of $T_{60} = 390$ ms, and a sparsity of 0.410.

The results from the measurements are shown in Figure 5.6. As with the simulations, ICMD again showed the best performance at low noise. Moreover, as the five-source simulation suggests, the results show performance loss from all algorithms as noise increases. Unlike the simulations, however, ICMD and ITP perform almost identically when the SNR dropped below 20 dB, and both shared a consistent improvement over PCA-ICA and Pham.

5.3.3. Discussion. In low noise, ICMD actually produces the highest SIRI and SDR. At these high SNRs, the primary gain of the PCA front-end—namely, noise reduction—is unnecessary, and ultimately leads to lower effective separation performance and quality. At higher noise, however, the PCA front-end greatly helped in the simulations. The benefits are less impressive in the real room experimental results, however. This stems from the deviation from the ideal isotropic noise model, as the noise sources are not truly isotropic, but are instead white noise coupled to the microphones via impulse responses.

ITP consistently performed better than PCA-ICA by at least a few dB of both SIRI and SDR in just about every test. Recall that the only difference between these
algorithms is how often \( V_k \) is recalculated. The performance, then, is due to ITP’s consistency over consecutive bins in the frequency domain. That is, by fixing the transformation matrix, we preserve smoothness in the frequency domain, which results in less distortion in the output. This is the same reason why ICMD shows good output quality.

All four tested algorithms utilize Pham’s energy profiles to order the outputs. The gains of the other three algorithms, therefore, stem from their effort to ensure a well-conditioned mixture at each frequency bin. The algorithms have also been tested with as many as seven sources, with good results from ICMD, ITP, and PCA-ICA; however, for this number of sources, processing time was extensive. In general, ICMD and ITP are able to separate any arbitrary number of sources, given enough microphones, and enough time.

5.4. COMPLEXITY

The computational complexity of an algorithm is the number of operations necessary to compute the result. This is typically simplified to counting the number of necessary multiplications. Recall that \( N_s \) is the number of sources, \( N_i \) is the number of inputs, and let \( T \) be the number of samples in each bin. It is assumed that \( N_s \ll N_i \ll T \).

All the algorithms compared here (except for Pham’s) use Hyvärinen’s [8] FastICA implementation. The complexity of this is roughly

\[
O_{ICA} = 3N_i^2T + N_i^3 + K_{ICA} \left[ 3N_i^2T + 10N_iT + 5N_s^3 \right]
\]

(5.12) multiplications per frequency bin, where \( K_{ICA} \) is the number of iterations of ICA. In the implementation here, this is upper-bounded at 20, but simulations show that a value of \( K_{ICA} = 10 \) is more appropriate.

The PCA-ICA algorithm requires constructing a cross-correlation matrix from a \( N_i \times T \) matrix, performing singular value decomposition on it, and applying the resulting \( N_i \times N_i \) transformation matrix to the input for that frequency. The number of multiplications per frequency is thus

\[
O_{PCA-ICA} = O_{PCA} + O_{ICA}
\]

(5.13)
where

\[ O_{PCA} = N_s^2 T + N_s N_s T + N_s^3. \]  \hspace{1cm} (5.14)

Instead of recalculating the transformation matrix at every frequency, ITP checks the outputs to ensure separation. This reduces the number of calculations per bin, but runs the risk of having to calculate a new transformation afterward. This increases the number of calculations above that of PCA-ICA; however, such an event is infrequent unless the SNR is low. The average number of multiplications per frequency bin is then

\[ O_{ITP} = N_s N_s T + N_s^2 T + 2T + 2N_s^3 + \rho O_{PCA} + O_{ICA}, \]  \hspace{1cm} (5.15)

where \( \rho \in [0,1] \) is the percentage of bad bins. This varies depending on environment and noise, but a reasonable value is 10%.

While ICMD performs the same check as ITP and ICMD every frequency for bad bins, it has an advantage in that diversity combining does not require a transformation to obtain the principal signals. On the other hand, when a new microphone set needs to be found, ICMD must check multiple sets for fitness; however, on the whole, those tests are less complex than PCA-ICA or ITP. The average number of multiplications per frequency is

\[ O_{ICMD} = N_s^2 T + 2T + N_s^3 + \rho K_{ICMD} \left[ N_s^2 T + N_s^3 \right] + O_{ICA} \]  \hspace{1cm} (5.16)

where \( \rho \) is again the percentage of bad bins, and \( K_{ICMD} \) is the number of tested microphone sets. For this comparison, \( K_{ICMD} \) has been set to \( N_s + 1 \). Experiments such as the one in section 5.1.2 show that this is reasonable.

The complexities of all three algorithms scale with increasing number of sources at roughly the same rate (dominated largely by ICA); however, they react differently when the number of inputs is varied. shows the relative computational complexity of ICMD, PCA-ICA, and ITP for a fixed number of sources (three), and a variable number of microphones. For this configuration, the complexity of ICA is fixed at \( O_{ICA} = 0.7 \times 10^5 \) multiplications.

Since ITP does not recalculate the transformation matrix at every frequency, it requires fewer multiplications than PCA-ICA, and the gains increase with more system microphones. Since it does not compute a linear transformation at all, ICMD has a computational complexity far less than either PCA approaches, with gains that increase

\[ \]
dramatically with rising $N_j$. Note that if it were necessary to compute separation for every candidate set of microphones (as suggested in [45] or [46]); even with a bad bin percentage of only $\rho = 10\%$, ICMD’s complexity would surpass PCA-ICA by almost an order of magnitude.
6. ESTIMATING THE NUMBER OF SOURCES

The algorithms of Inter-frequency Correlation with Microphone Diversity (ICMD) and of ICA with Triggered PCA (ITP) are capable of extracting an arbitrary number of sources from an arbitrary number of mixtures, so long as the latter outnumbers the former. This separation is performed efficiently and effectively, with no prior knowledge of microphone, source, or room geometry. Unfortunately, prior knowledge of the number of sources, \( N_s \), is required. The versatility of these algorithms would be greatly improved if this parameter could be estimated in a blind setting, before the primary separation algorithm is applied.

In an instantaneous mixture with sufficient SNR, estimating the number of independent sources is straightforward. As described in Appendix C, the number of sources present in an instantaneous overdetermined mixture is usually visible in the singular values. In the same fashion, the eigenvalues can provide this information. Since ICA begins with a prewhitening stage that reveals the eigenvalues, i.e. equation (1.11), such a thresholding operation is not only straightforward, but already included in many Fast ICA implementations.

Consider the following simulated mixtures of \( N_s \) sources observed by six microphones. Figure 6.1 shows the eigenvalues—the sorted diagonal elements of \( \Lambda \) of (1.11)—for a mixture of (a) two sources and (b) four sources. Visual inspection clearly shows two dominant eigenvalues in the left graph, and four in the rightmost. These correspond correctly to the number of sources present in each mixture. With a good threshold, chosen by careful analysis of the background SNR of the mixing environment, \( N_s \) estimation can be very reliable in an instantaneous mixture. Even with a poorly-chosen threshold, an estimate of the number of sources will not be too far off.

Unfortunately, estimating the number of sources in a convolutive mixture is not nearly so simple. If the reverberation of the mixing time is long (high \( T_{60} \)) or strong (low sparsity), the transformation created by the STFT is insufficient, and equation (1.3) does not hold [57]. In this case, the mixing channels at each frequency retain some memory—that is, \( A_k \) of (1.3) remains a matrix of FIR filters, albeit ones with fewer taps than the
time-domain mixing matrix. Then, the number of significant eigenvalues in the input mixture may exceed the actual number of sources. Otherwise, some frequencies may show fewer dominant eigenvalues than sources. This can happen because speech sources do not have consistent excitation at all frequencies, or because the mixing matrix has component vectors that are too similar (leading to subsets of $A_x$ that are ill-conditioned).

Many of the techniques for estimating the number of sources would not apply well to the framework of ICMD/ITP. Asaei et al. [34] entwine source recovery with source enumeration (as a subset of geometry estimation) by casting the entire problem as spatially-sparse sources, and using sparse recovery techniques. Istemic [58] examines a statistic called Activity Index to determine the number of sources. This shows promise, and has been tested on up to five sources. Unfortunately, the tested sources were only pulse trains, with short mixing filter lengths ($L \leq 10$)—both inappropriate for the cocktail party model. Olsoon [59], uses a statistical analysis to create the Bayes Information Criterion (BIC), with which it can estimate not only how many sources are present, but when each one is active. Unfortunately, this algorithm has only been tested with two sources, and is meant to be run concurrent with time-domain separation. Additional investigation would be necessary to determine if it would be effective as a pre-processor for a frequency-domain algorithm.
Fortunately, the algorithm of Sawada et al. [57] can create an estimate of $N_s$ at each frequency bin. The authors’ intent was to estimate the number of sources after processing every bin, but experiments suggest that testing a random sample of frequency bins may be sufficient. This approach may work well as a preprocessing stage for ICMD or ITP, as experiments will show.

The algorithm begins by temporarily assuming the determined case and performing separation on the $N_i$ inputs, producing $N_i$ outputs. Scaling is applied to each output by the following formula:

$$Y_k = \text{diag}\left\{ \left( W_k W_k^H \right)^{-1} \right\}^{1/2}.$$  
\( (6.1) \)

$$y_k(t) = Y_k W_k x_k(t)$$

As before, $\text{diag}\{A\}$ returns the matrix $A$ with all off-diagonal elements set to zero. Equation (6.1) restores the power of each output to how it was measured at the input. It is similar in construction and concept to the minimal distortion principle. From the scaled outputs, two metrics are defined. The normalized power for each output $m$ $NP(m)$ is calculated such that

$$NP(m) = \frac{\mathbb{E}\left[ |y_k(m,t)|^2 \right]}{\sum_{m=1}^{N_s} \mathbb{E}\left[ |y_k(m,t)|^2 \right]}.$$  
\( (6.2) \)

Secondly, the envelope of each output, found as

$$v_k(m,t) = |y_k(m,t)| - \mathbb{E}\left[ |y_k(m,t)| \right],$$  
\( (6.3) \)

can be correlated against the other output envelopes to form the cross-correlation coefficient $\rho_{ij}$ by the formula:

$$\rho_{ij} = \frac{\mathbb{E}\{v_k(i,t)v_k(j,t-\tau)\}}{\sqrt{\mathbb{E}\{v_k^2(i,t)\}} \sqrt{\mathbb{E}\{v_k^2(j,t-\tau)\}}}.$$  
\( (6.4) \)

Once the two metrics are defined, the outputs can be classified. Sawada labels each output as either a) a component of an independent source, b) a reverberation of such a component, or c) a fragment of background noise. If the normalized power $NP(m)$ is smaller than a certain threshold (such as 0.01), then the output signal $y_k(m,t)$ is
considered noise. If the normalized power is above a higher threshold (such as 0.2), the output signal is labeled a source component. If the normalized power is between these two thresholds, then it is either a source or a reverberation; if the correlation between it and other components is above 0.5, then it is a reverberation, otherwise it is another source. The thresholds provided match the examples given in [59], but better values may be calculated based on estimates of the mixing environment.

For an example of the classification algorithm’s steps, consider the following figure. Figure 6.2 shows the two metrics for bin 492, corresponding to roughly 480 Hz, of a mixture of four sources, recorded on nine microphones. The inputs to the system were simulated mixtures, created in a simulated room with $T_{60} = 430 \text{ms}$ and a sparseness of 0.430, and with an SNR of 30 dB. Part (a) shows the normalized power of the outputs, while part (b) shows the cross-correlation coefficients between outputs. Note that for display purposes, $\rho_{ij}$ for all $i, j$ have been calculated, whereas in practice the coefficients of only some pairings will be necessary.

(a) $NP(i)$ and thresholds
(b) Cross-correlation coefficient $\rho_{ij}$

Figure 6.2. Metrics for a $N_s$ estimation (true $N_s = 4$)

The algorithm first classifies outputs 1 and 2 as sources, due to sufficient normalized power. It then has to test outputs 3 through 6 to determine their nature.
Output 3 does not correlate to either prior output—therefore it is a source. Outputs 4 and 5 correlate too strongly to 3 and 2, respectively. Output 6 is again its own source. The final three outputs have too low of power, and are discarded as noise. This correctly identifies four separate sources at this frequency bin.

For reasons described earlier, not every source will be present at each bin. As such, it is necessary to test multiple frequency bins and use the plurality of $N_s$ estimations. The number of frequency bins to test is a tradeoff between processing time and estimate accuracy, but preliminary experiments show that a random sample of thirty bins may be sufficiently accurate. The test produced on the data set above produced the result in Figure 6.3. Of the thirty bins tested, the algorithm correctly identified 19 of them as having four sources, with a majority of the remainder having only three. This, actually, is typical in behavior: at high levels of SNR, there is usually a clear majority, but the second-highest estimate is for $N_s - 1$ sources.

![Figure 6.3. Estimated $N_s$ from thirty random frequency bins](image)
7. CONCLUSION AND FUTURE WORK

A half-century has passed since Professor Cherry tasked engineers to create a machine to separate mixtures of human speech, and while the problem is not yet solved, the community has risen to the challenge. This dissertation has described three new algorithms for the separation of blind mixtures, and additional insights as to how environments should be characterized.

In classic two-source BSS, the existing algorithm of CICAIA, and the new CICADI, represent a successful paradigm of expending computational effort only where it is necessary. The latter algorithm in particular shows performance on par with conventional benchmarks in good conditions, and excellent performance in poor conditions of noise, reverberation, or room sparsity. Typical performance was 17 dB of SIR Improvement, with 13 dB of output SDR. It, however, suffered from the common restriction of allowing no more than two sources. Still, if it can be guaranteed that there are only two sources, CICADI’s speed and output quality make it an exceptional algorithm to apply.

The targeted effort paradigm continued with new algorithms designed specifically to handle additional sources. The proposed algorithms, Inter-frequency Correlation with Microphone Diversity (ICMD), and ICA with Triggered Principal component analysis (ITP), are improvements to recent work in the field of overdetermined blind source separation. Both have the capability to extract an arbitrary number of sources in moderate noise, with 10 dB of SIR improvement on a mixture of five sources in good conditions. They have good performance with increasing numbers of sources and with no notable restrictions on source or microphone placement.

ICMD’s innovation is in its handling of microphone diversity. By using a random sample of microphone sets, and by using a prediction metric to determine which replacement set is the best without performing separation, the set switching procedure is made very efficient. Running this procedure only on the frequency bins where it is needed benefits both the algorithm’s complexity and its output quality.
ITP enjoys the same benefits as ICMD from its consistency across sequential frequency bins. This gives it advantages over the traditional PCA approach in terms of separation performance, output quality, and computational complexity.

Simulations suggest that ITP is capable of better separation in high noise, but actual room experimental results show that the difference is less pronounced in application. ICMD shows a small (1~2 dB) advantage in terms of both SIRI and SDR, but a very large benefit in terms of computational complexity. As a result, ICMD is the recommended algorithm for scenarios with many microphones and moderate noise.

The microphone diversity front-end of ICMD shows great potential for additional uses. It requires no information about the hardware geometry, yet is capable of selecting the best sensors to use. This makes it useful for front-ends to other BSS algorithms, or to multi-input multi-output systems in general. In particular, in the case of distributed sources and sensors, ICMD provides a conceptually and computationally simple algorithm to select microphones close to the sources themselves.

Currently, ICMD’s and ITP’s largest restriction is the necessity of knowing $N_s$ prior to processing. Unfortunately, most existing algorithms to determine $N_s$ involve running separation on all frequencies, and estimating the source count from the statistics, which is incompatible with the framework of ICMD or ITP. Fortunately, the preprocessor based on the work of Sawada et al. may be capable of producing reliable estimates of the number of sources present. Additional testing is required to properly characterize its behavior, but initial tests show that it can correctly identify the number of sources present after processing as few as thirty frequency bins. Once $N_s$ can be reliably estimated from processing a subset of frequencies, the algorithm will be capable of deployment in a truly blind setting, on any microphone array in any configuration, with any arrangement of sources.
APPENDIX A

ALGORITHM OF PHAM, SERVIÈRE, AND BOUMARAF
The algorithm of Pham, Servière, and Boumaraf [22] is a frequency domain BSS algorithm that relies on inter-frequency correlation to align permutations. The algorithm utilizes “profiles” as a useful and efficient mechanism to exploit this correlation.

Profiles are designed to exploit the nonstationarity of wideband sources such as speech. Frequency components of such sources will have amplitudes over time that mirror the envelope of the original signal. Due to uneven excitation of such sources, however, individual frequency components will have a frequency-dependent scaling factor.

The energy profile of output vector $y_k(t)$ for frequency bin $k$ is calculated as

$$e_k(t) = \log(||y_k(t)||).$$  

(A.1)

The logarithm operation turns the frequency-dependent scaling factor into a mean. Removing this, results in the corresponding centered energy profile:

$$e'_k(t) = e_k(t) - \bar{e}_k$$  

(A.2)

where $\bar{e}_k$ is the sample mean of $e_k(t)$.

Visually, this can be represented in Figure A.1. The left side is the standard spectrogram, with frequency on the vertical axis, and time on the horizontal. The amplitudes at each bin are displayed on a log scale. On the right, the energy profile at one frequency (3000 Hz) is displayed. Profiles in other frequencies are similar in shape.

![Figure A.1. Formation of a Profile](image-url)
Given a vector of target profiles $e'_t(t)$, which is presumed to be an accurate representation of the profile of the sources, the most obvious method to determine the proper output order in each frequency bin would be via cross-correlation. This operation, however, would involve vector multiplications. Considering that the frequency-dependent scaling factor is removed by centering the profile, all profiles should be at the same relative scale. The alignment can thus be found more simply, by comparing the differences.

In [22], the proper order of outputs in bin $k$ is achieved by finding the best permutation matrix $\Pi_k$ that minimizes the sum of the norms of the difference between the existing profiles and their targets; that is,

$$\Pi_k = \arg \min_n \sum_m \|e'_k(t) - \Pi_k e'_t(t)\|_2.$$  \hspace{1cm} (A.3)

where $\mathbf{e}'_t = \begin{bmatrix} e'_t(1, t) & \cdots & e'_t(m, t) \end{bmatrix}$ for both existing and target profiles, and the $\ell_2$ norm is calculated with respect to time. This operation is less computationally complex than processing via cross-correlation.

Ideally, the exact profiles of the sources make the best values for $e'_t(t)$, but these are not available due to the blind nature of the system. After separation, however, the average energy profile (across all frequencies) of each output will more closely resemble the source more prevalent at that output. These can be used as an initial estimate of the profiles of sources.

In [22], separation is performed on all frequencies in one sweep, and then the target profile for each output is constructed as the average over all frequencies. Alignment is then performed over the entire frequency range. Afterward, a new average is taken, which is used in another sweep of permutation alignment. This process is repeated until convergence, with an upper bound of twenty iterations.
APPENDIX B

CONDITION NUMBER
The condition number of a matrix $\mathbf{A}$, abbreviated $\kappa(\mathbf{A})$, is defined as the ratio of its largest to its smallest singular value, but it is better known as a measure of the matrix’s invertibility. The value is a number in the range of $[1, \infty)$, with higher values corresponding to a matrix that is more difficult to invert. Unitary matrices such as the identity matrix have a condition number of 1, while singular matrices have a condition number of infinity. Linear systems with a matrix with too high of a condition number are considered *ill-conditioned*. Specifically, the value of $\log_n \kappa(\mathbf{A})$ gives the number of digits of precision in base $n$ that are lost when computing $\mathbf{A}^{-1}$ [60].

The condition number of a matrix is equal to the condition number of its inverse (assuming the latter exists). This follows from one of the definitions of the condition number as the product of two induced norms:

$$\kappa(\mathbf{A}) = \|\mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2$$  \hspace{1cm} (B.1)

The row- or column-permutation of a matrix does not change its condition number.
APPENDIX C

PRINCIPAL COMPONENT ANALYSIS
Principal Component Analysis (PCA) is a powerful statistical technique for analyzing data sets, especially those of high dimensionality. It is useful for pattern recognition [61], feature extraction, and data compression [62]. It performs a linear transformation on the data to discard redundancy and de-correlate variables. This is similar to the aims of ICA; however, in PCA the correlation between inputs is all that matters, while in ICA the much richer concept of independence is used [62]. As a result, most ICA algorithms begin with the prewhitening stage, which is accomplished via PCA and multiplication of a scalar. These principal component outputs are then converted to independent components by an additional linear operation, which is calculated using the higher-order statistics of the signals.

PCA begins with the observation of a vector $\mathbf{x}(t) = [x_1(t) \cdots x_N(t)]^T$ of dimension $N$, $N > 1$. The first step is to remove the mean from the data:

$$\mathbf{x}(t) = \mathbf{x}(t) - \mathbb{E}\{\mathbf{x}\}. \quad (C.1)$$

There are a few methods of performing PCA which diverge here, but the typical method is by variance maximization. With this approach, the next step is calculation of the covariance matrix $\mathbf{R}_x$:

$$\mathbf{R}_x = \mathbb{E}\{\mathbf{x}\mathbf{x}^H\}. \quad (C.2)$$

This is then factored using eigendecomposition:

$$\mathbf{R}_x = \mathbf{E}\Lambda\mathbf{E}^H. \quad (C.3)$$

where $\mathbf{E} = [\mathbf{e}_1 \cdots \mathbf{e}_N]$ is a matrix of unit length eigenvectors, and $\Lambda$ is a diagonal matrix of the corresponding eigenvalues $\lambda_1, \ldots, \lambda_N$, arranged such that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$. The output produced by the linear transformation

$$y_1(t) = \mathbf{e}_1^H\mathbf{x}(t) \quad (C.4)$$

has the largest variance for any combination

$$y_1(t) = \mathbf{w}^H\mathbf{x}(t) \quad \text{s.t.} \|\mathbf{w}\| = 1, \quad (C.5)$$

and as such is known as the principal component. The output from PCA, is then given by

$$\mathbf{y}(t) = \mathbf{E}^H\mathbf{x}(t). \quad (C.6)$$
This output contains the components of the data, now uncorrelated, and arranged in descending order of magnitude.

PCA also frequently coincides with dimensionality reduction in a data set. By including only the first \( M \) eigenvectors of \( \mathbf{E} \), \( M < N \), it is possible to shed redundancy from the data. The output dimensionality is usually chosen by threshold: any eigenvalues below a set limit are discarded, along with their corresponding principal components. For example, in the case of an instantaneous mixture of \( M \) sources observed on \( N \) dimensions, the last \( N - M \) eigenvalues will be far lower than the first \( M \). These can be safely discarded; the remaining principal components can then yield the original sources via ICA.

For an example of the PCA’s behavior, consider the following data set:

![Figure C.1. Data set on which to perform PCA](image)

Without knowing the method of generation, visual inspection of Figure C.1 reveals that the data is in fact positively correlated. The data also appears to have a Gaussian density that is constant over an ellipse. Figure C.2, below, shows this ellipse, and its two radii.
After principal component analysis, the data has been rotated to reveal the independent components, which puts the new coordinate system on the radii of the ellipse. The principal component, with a stronger variance, is aligned to the first output (the x axis).
BIBLIOGRAPHY


87


Christopher Thomas Paul Osterwise was born in St. Louis, Missouri, on September 10th, 1984. He completed both a Bachelor of Science in Electrical Engineering and one in Computer Engineering in May 2002 from the University of Missouri-Rolla, where he was accepted directly into the Electrical Engineering Ph.D. program, with an emphasis in signal processing. During his undergraduate, and in the early years of his PhD program, he was employed to rebuild the power lab, replacing twenty-year-old control circuitry with newer products, and designing an entirely new control interface. During this time, he was also in charge of the Student Projects Lab for the Electrical and Computer Engineering honor society, Eta Kappa Nu, where he managed inventory, and prepared project kits to give undergraduate students experience with circuit construction. In his last year as a graduate student, he worked with L-3 Mission Integration Division in Greenville, Texas, where he will return after graduation in May 2013. His research interests include signal processing for blind source separation and echo cancellation.