ALGORITMOS DE SEPARAÇÃO CEGA DE SINAIS DE ÁUDIO NO DOMÍNIO DA FREQUÊNCIA EM AMBIENTES REVERBERANTES: ESTUDO E COMPARAÇÕES

Luiz Victorio de Menezes Laporte

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientadora: Mariane Rembold Petraglia

Rio de Janeiro
Outubro de 2010

ALGORITMOS DE SEPARAÇÃO CEGA DE SINAIS DE ÁUDIO NO DOMÍNIO DA FREQUÊNCIA EM AMBIENTES REVERBERANTES: ESTUDO E COMPARAÇÕES

Luiz Victorio de Menezes Laporte

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Mariane Rembold Petraglia, Ph.D.

Prof. José Gabriel Rodriguez Carneiro Gomes, Ph.D.

Prof. Tadeu Nagashima Ferreira, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
OUTUBRO DE 2010

Laporte, Luiz Victorio de Menezes

Algoritmos de Separação Cega de Sinais de Áudio no Domínio da Frequência em Ambientes Reverberantes: Estudo e Comparações/Luiz Victorio de Menezes Laporte. – Rio de Janeiro: UFRJ/COPPE, 2010.

XV, 128 p.: il.; 29, 7cm.

Orientadora: Mariane Rembold Petraglia

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2010.

Referências Bibliográficas: p. 111 – 119.

1. separação cega de fontes. 2. reverberação. 3. análise de componentes independentes. 4. direção de chegada. 5. algoritmos não-supervisionados. I. Petraglia, Mariane Rembold. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

A Jesus Cristo, meu Senhor e Salvador, que me deu muito mais do que eu merecia.

Agradecimentos

Seria injusto dizer que esta dissertação é minha, quando tantas pessoas contribuíram para que ela terminasse, e não estou falando dos autores cujo nome está citado na bibliografia, mas sim dos amigos, familiares e professores. Tenho muito a agradecer à minha esposa Aislan, pela paciência nos momentos em que tive que abdicar de sua companhia para me dedicar à dissertação, e por seu encorajamento quando pensei em desistir. Sem essa ajuda, seria impossível continuar. Prometo que compensarei o tempo perdido.

Agradeço ao meu amigo Diego Haddad, pelas inúmeras dicas dadas e pelo árduo trabalho de revisão desta dissertação, além de ter sido o responsável por eu estar aqui. Ao meu colega Daniel Mendes, pelos e-mails me lembrando da inscrição em disciplinas do PEE. Meus chefes também foram de imensa valia, e sem sua permissão não conseguiria cursar as matérias ou escrever a dissertação. Por isso, agradeço ao Bruno Jouan, da época em que cursava as disciplinas, e Luciano Diniz, pelo tempo concedido nas últimas semanas para terminar a escrita da dissertação.

Da minha orientadora, Mariane Petraglia, posso dizer que é a melhor orientadora que este mundo já viu. Sua compreensão é infinita, e ela foi um suporte em todas as fases deste projeto. Ela prefere orientar e incentivar, a cobrar. Este é um atributo raro. Adicionalmente, seu conhecimento tecnológico é indiscutível.

Porém, em primeiro lugar, agradeço a Deus, por ter realizado alguns pequenos milagres para que esta dissertação pudesse ser concluída.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

ALGORITMOS DE SEPARAÇÃO CEGA DE SINAIS DE ÁUDIO NO DOMÍNIO DA FREQUÊNCIA EM AMBIENTES REVERBERANTES: ESTUDO E COMPARAÇÕES

Luiz Victorio de Menezes Laporte

Outubro/2010

Orientadora: Mariane Rembold Petraglia

Programa: Engenharia Elétrica

Recentemente, temos visto um interesse crescente em Separação Cega de Fontes, especialmente no caso reverberante, que está longe de ter uma solução completa, mas tem evoluído num passo incrivelmente rápido. Nesta dissertação, apresentamos os algoritmos no domínio da frequência do "estado da arte" para resolver este problema, e fazemos uma comparação entre eles. Modificamos vários parâmetros de todos os diferentes algoritmos para separação e alinhamento da permutação a fim de compararmos seus desempenhos. Também propomos algumas modificações nos mesmos, como mudar o tipo de janela na transformação de frequência e o passo de pós-processamento, ou trocar o algoritmo de clusterização em algumas das propostas de alinhamento da permutação, este último com ganhos significativos de desempenho. Conduzimos os testes em um ambiente reverberante simulado, o que nos deu mais liberdade para modificar os parâmetros do ambiente e analisar o desempenho dos algoritmos frente a essas mudanças.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

FREQUENCY DOMAIN BLIND AUDIO SEPARATION IN REVERBERANT ENVIRONMENTS: STUDY AND COMPARISON

Luiz Victorio de Menezes Laporte

October/2010

Advisor: Mariane Rembold Petraglia

Department: Electrical Engineering

Recently, we have seen an increasing interest in Blind Source Separation, especially in the reverberant case, which is far from a complete solution, but has evolved in an amazingly fast pace. In this dissertation, we present the state-of-the-art frequency domain algorithms for solving this problem, and make a comparison among them. We change various parameters of all the different algorithms for separation and permutation alignment and compare their performances. We also propose some modifications to them, like changing the window type in the frequency transformation and the postprocessing step, or changing the clustering algorithm in some of the permutation alignment proposals, the last case with significant performance gains. We conducted the tests in a simulated reverberant environment, which gave us more freedom in changing the parameters of the environment and analysing the performance of the algorithms with these changes.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Recentemente, interfaces automáticas de conversação [1] para máquinas inteligentes como robôs e computadores receberam muita atenção da comunidade científica, porque elas facilitam o controle dos usuários, bem como permitem um diálogo natural e simples, independentemente da sofisticação intrínseca ao sistema. O interesse pelo desenvolvimento de tais sistemas, que podem ouvir, entender e falar em uma linguagem natural, não é nova [2]. Entretanto, nas últimas duas décadas a pesquisa se intensificou. De fato, os avanços nas técnicas de reconhecimento de voz, reconhecimento de locutor e desenvolvimento de *softwares* eficientes de reconhecimento de fala automática, aceleraram a demanda por sistemas ativados por voz.

Embora as técnicas de reconhecimento automático de fala estejam bem avançadas, muitas restrições dificultam sua aplicação em ambientes reais. O principal problema é a qualidade do sinal de voz que o sistema processa. Se o sinal estiver limpo e livre de distorções, o sistema funciona razoavelmente bem, mas à medida que a qualidade do sinal piora, o desempenho de um sistema de reconhecimento de voz cai dramaticamente. O caminho do sinal de voz até o sensor de recepção é complexo, sendo comum sua contaminação por ruídos de fundo, vozes de outras pessoas, música, ou até mesmo por versões atrasadas do próprio sinal de voz devido a reflexões em paredes ou móveis (reverberação). Tudo isso depende de onde o sistema foi instalado, e de qual o seu propósito. Porém, mesmo que o sistema seja instalado em um ambiente livre de ruído e outras contaminações de áudio, a reverberação é quase inevitável, a não ser que a sala seja transformada em um ambiente anecóico[1], o que em geral não é prático. Porém, de uma forma geral, não é possível evitar a contaminação do sinal de voz com outros sinais de áudio, que também sofrem com

[1]Um ambiente anecóico é um ambiente livre de reverberação, onde, por definição, a absorção do som é completa. Em geral, estúdios de gravação, ou laboratórios de teste possuem um ambiente quase anecóico, através de tratamento das paredes, piso e teto com materiais com um alto coeficiente de absorção do som. Num ambiente desse tipo, o único caminho que o sinal de som percorre é o caminho direto entre a fonte de som e o sensor.

a reverberação, e há necessidade de algoritmos que possam separar o sinal desejado de voz de outros sinais indesejados, antes de realizar o resto do processamento.

A idéia de equipar uma máquina com um sistema de reconhecimento de voz é baseada no sistema natural humano. O problema da separação de sinais de voz é bem comum em nossa vida diária e fazemos isso frequentemente sem nos darmos conta. Os humanos são capazes de manter sua atenção em um locutor particular, mesmo na presença de inúmeras outras fontes de áudio e ruído. Esta habilidade é bem conhecida na literatura como o *Cocktail Party Effect* [3], traduzido ao pé da letra como efeito (festa de) coquetel. Infelizmente, muito pouco é conhecido sobre nosso processamento cerebral de sinais de voz. De um ponto de vista de engenharia, esse problema está mostrado na Figura 1.1, onde há várias fontes de sinais acústicos e os sinais de todas elas se misturam nos sensores (os microfones). O sinal do microfone gravado nas condições da figura é inútil para um sistema de reconhecimento de voz, e necessita de processamento adicional para separar as fontes umas das outras.

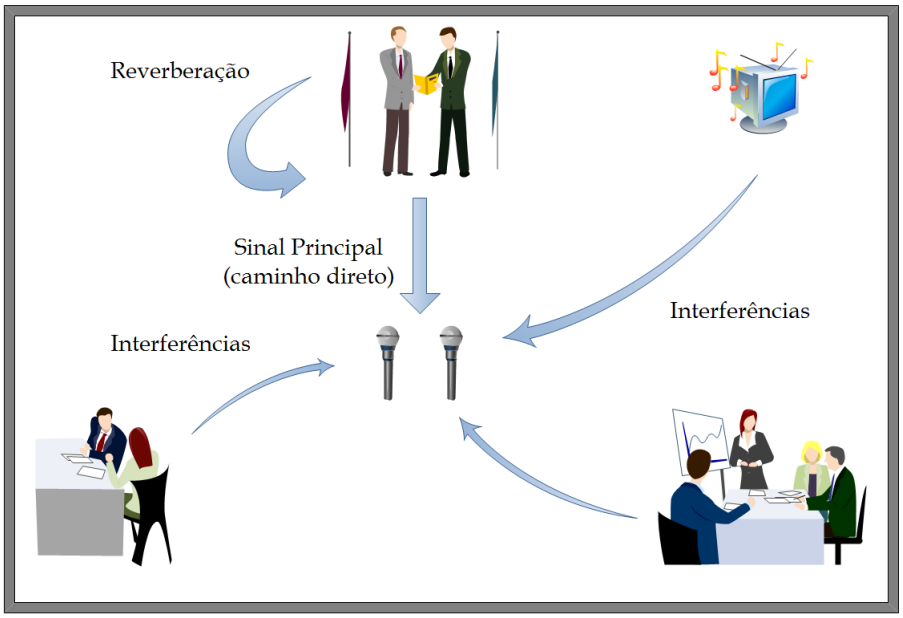

Figura 1.1: Ilustração do *Cocktail Party Effect*. O interesse é captar o sinal dos locutores, mas muitas outras interferências são capturadas. O cérebro humano não encontra problemas em focar sua atenção em apenas uma fonte de som, mas um algoritmo de reconhecimento de fala não funciona na presença de interferências.

Hoje, a maior limitação dos sistemas de reconhecimento de voz é a separação destas fontes, quando estão disponíveis as misturas dos sons, captadas pelos microfones. Com este problema resolvido, a implementação de uma máquina que entenda humanos se torna plausível. Recentemente, muitos artigos têm sido publicados sobre este assunto e muitas técnicas diferentes para resolver tal problema foram apontadas. A separação de tais fontes, tendo disponíveis apenas os sinais dos sensores, é chamada

de BSS (Separação Cega de Fontes, do inglês *Blind Source Separation*), sendo denominada cega devido à ausência de informações prévias acerca tanto das fontes, como do ambiente, que é responsável pelas funções de transferência dos modelos acústicos envolvidas no processo de gravação.

Em ambientes livres de reverberação e com atrasos desprezíveis de propagação entre sensores, tal problema já está resolvido. A idéia é tentar separar as componentes independentes de um sinal, supondo que cada uma das fontes é estatisticamente independente das outras; este algoritmo é chamado de ICA (Análise de Componentes Independentes, do inglês *Independent Component Analysis*). Na prática, a reverberação costuma estar presente, o que torna o problema muito mais complicado e ainda sem solução definitiva. A grande maioria das abordagens utiliza esta suposição de que os sinais são independentes, pois, sem isso, não há forma de separar as fontes com ICA. O problema é que, em ambientes reverberantes, o ICA se torna muito complexo, porque cada sinal é filtrado pela resposta de frequência do ambiente (sala). Essa alta complexidade se traduz em maior tempo de processamento. Uma forma de aliviar esta complexidade é tratar os sinais no domínio da frequência, o que diminui drasticamente o tempo de processamento. Essa abordagem é conhecida como FDBSS (Separação Cega de Fontes no Domínio da Frequência, do inglês *Frequency-Domain Blind Source Separation*) [4]. Infelizmente, esse tratamento traz outros problemas, como o chamado *problema da permutação*, o qual passa a não ser mais trivial. Formas de resolver este problema têm sido propostas, ainda que sejam necessárias muitas contribuições para o refino destas técnicas.

O objetivo desta dissertação é estudar os algoritmos FDBSS propostos na literatura, e destacar as vantagens e desvantagens de cada um, além de comparar seus desempenhos. Todos os testes foram realizados com sinais reais de voz, gravados em ambiente anecóico (estúdio), aplicados à simulação de uma sala reverberante. A opção pela simulação da sala é justificada pela flexibilidade de se modificar os parâmetros do ambiente, sem necessitar de intervenções físicas. Além disso, os algoritmos de simulação de resposta de frequência de um ambiente emulam razoavelmente diversas características de respostas reais de um ambiente acústico [5, 6].

1.1 Organização da Dissertação

No Capítulo 2, definiremos o problema da separação cega de fontes de uma forma matemática, tanto na forma instantânea (livre de reverberação), quanto na forma convolutiva (ambientes reverberantes), e mostraremos as ambiguidades inerentes ao problema. Também apresentaremos a Análise de Componentes Independentes. Adicionalmente, discutiremos as estatísticas amostrais de números complexos, que serão muito úteis, por causa da implementação no domínio da frequência. Por fim,

definiremos as medidas de avaliação de desempenho dos algoritmos utilizada ao longo da dissertação.

No Capítulo 3, apresentamos a solução para o problema de separação cega de fontes em ambientes reverberantes, utilizando a transformada de Fourier. O ICA é estendido para o domínio da frequência, e são descritos os problemas adicionais decorrentes da transformação de domínio, e as formas de tentar resolvê-los. Uma visão geral de um sistema de separação cega de fontes no domínio da frequência é mostrada, e cada um dos passos é detalhado a seguir.

No Capítulo 4, focamos no problema da permutação, que será inicialmente apresentado no Capítulo 3, e é inerente a implementações no domínio da frequência. Hoje, este é o problema mais difícil de se resolver nestas implementações, para ambientes reverberantes. As propostas encontradas na literatura são descritas e comparadas, destacando suas limitações.

Por fim, o Capítulo 5 apresenta algumas conclusões e perspectivas de trabalho futuro.

O Apêndice A descreve nosso ambiente de testes, e contém mais informações sobre os sinais de voz utilizados e a simulação da resposta de frequência da sala. O Apêndice B mostra a maneira que utilizamos para testar a convergência do ICA, e, dessa forma, diminuir o tempo de processamento do sistema de separação cega de fontes como um todo. O Apêndice C apresenta um método supervisionado para resolver o problema da permutação, para que possamos estimar o quão eficiente um algoritmo seria se não houvesse permutação.

Capítulo 2

Introdução à Separacão Cega de Fontes

Este capítulo introduz o problema de BSS, apresentando primeiramente as definições de misturas lineares e instantâneas para então abordar as configurações de misturas convolutivas, cuja separação constitui o principal objetivo desta dissertação. Na Seção 2.4, são descritas estatísticas amostrais de números complexos, um tema pouco abordado na literatura, mas de suma importância para a aplicação prática de algoritmos de BSS no domínio da frequência. A seguir, descrevemos o algoritmo mais utilizado para solução de problemas de BSS e a forma utilizada na dissertação para avaliar o desempenho dos algoritmos.

2.1 Mistura Linear e Instantânea

Sejam N fontes $s_i(n), i = 1, \cdots, N$. O vetor $\mathbf{s}(n)$ compreende as fontes s_i no instante n:

$$\mathbf{s}(n) = \begin{bmatrix} s_1(n) \\ s_2(n) \\ \vdots \\ s_N(n) \end{bmatrix} \tag{2.1}$$

Seja $\mathbf{H}$ uma matriz (de dimensões $M \times N$) denominada matriz de mistura. Considerando a mistura linear, instantânea e não-ruidosa, podemos expressar o vetor $\mathbf{x}(n)$ que contém as n-ésimas observações (ou amostras[1]) das M misturas pela equação:

$$\mathbf{x}(n) = \mathbf{H}\mathbf{s}(n) \tag{2.2}$$

[1] A nomenclatura "amostra" será evitada neste capítulo para evitar confusão, devido às discussões sobre estatística amostral, que considera que uma amostra é um conjunto de observações. Nos próximos capítulos, retomaremos esta nomenclatura.

A matriz de mistura $\mathbf{H}$ é definida por:

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1N} \\ h_{21} & h_{22} & \cdots & h_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ h_{M1} & h_{M2} & \cdots & h_{MN} \end{bmatrix} \tag{2.3}$$

onde cada escalar h_{ji} determina o quanto da fonte i está presente na mistura j. Podemos concatenar os vetores $\mathbf{x}(n)$ e obter a matriz das misturas $\mathbf{X}$ (não confundir com a matriz *de mistura* $\mathbf{H}$), segundo mostrado em (2.4), onde N_{amost} é o número total de observações de cada fonte.

$$\mathbf{X} = \begin{bmatrix} x_1(1) & x_1(2) & \cdots & x_1(N_{\text{amost}}) \\ x_2(1) & x_2(2) & \cdots & x_2(N_{\text{amost}}) \\ \vdots & \vdots & \vdots & \vdots \\ x_M(1) & x_M(2) & \cdots & x_M(N_{\text{amost}}) \end{bmatrix} \tag{2.4}$$

O mesmo pode ser feito com o vetor $\mathbf{s}(n)$ e obter a matriz das fontes $\mathbf{S}$. Essas representações só são válidas para processamento em bloco, i.e, quando todas as observações de todas as fontes estão disponíveis.

$$\mathbf{S} = \begin{bmatrix} s_1(1) & s_1(2) & \cdots & s_1(N_{\text{amost}}) \\ s_2(1) & s_2(2) & \cdots & s_2(N_{\text{amost}}) \\ \vdots & \vdots & \vdots & \vdots \\ s_N(1) & s_N(2) & \cdots & s_N(N_{\text{amost}}) \end{bmatrix} \tag{2.5}$$

No caso MLI (Mistura Linear e Instantânea), cada mistura (linhas da matriz das misturas $\mathbf{X}$) é uma combinação linear das fontes (linhas da matriz das fontes $\mathbf{S}$), ou seja:

$$x_j(n) = \sum_i h_{ji} s_i(n),\ i = 1, \cdots, N,\ j = 1, \cdots, M \tag{2.6}$$

O objetivo é recuperar as fontes $s_i(n)$ a partir das misturas $x_j(n)$. Para isso, definimos a matriz separadora $\mathbf{W}$, e o sinal $y_i(n)$, que é a estimativa do sinal $s_i(n)$. Dessa forma, temos:

$$\begin{bmatrix} y_1(n) \\ y_2(n) \\ \vdots \\ y_N(n) \end{bmatrix} = \mathbf{W} \begin{bmatrix} x_1(n) \\ x_2(n) \\ \vdots \\ x_M(n) \end{bmatrix} \tag{2.7}$$

A matriz $\mathbf{W}$, de dimensões $N \times M$, é definida por:

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1M} \\ w_{21} & w_{22} & \cdots & w_{2M} \\ \vdots & \vdots & \vdots & \vdots \\ w_{N1} & w_{N2} & \cdots & w_{NM} \end{bmatrix} \tag{2.8}$$

Concatenando todas as observações das fontes estimadas $y_i(n)$, como feito em (2.4) e (2.5), chegamos à matriz $\mathbf{Y}$ de fontes estimadas :

$$\mathbf{Y} = \begin{bmatrix} y_1(1) & y_1(2) & \cdots & y_1(N_{\text{amost}}) \\ y_2(1) & y_2(2) & \cdots & y_2(N_{\text{amost}}) \\ \vdots & \vdots & \vdots & \vdots \\ y_N(1) & y_N(2) & \cdots & y_N(N_{\text{amost}}) \end{bmatrix} \tag{2.9}$$

Se a matriz $\mathbf{H}$ fosse conhecida, a solução para o problema seria simplesmente $\mathbf{W} = \mathbf{H}^{-1}$, se $N = M$, ou $\mathbf{W} = \mathbf{H}^{\dagger}$, se $M > N$, onde o operador † simboliza a pseudo-inversa [7]. Porém, em BSS só conhecemos os sinais x_j das misturas, portanto, a matriz separadora deve ser estimada de outra forma. O método mais utilizado para este fim é a Análise de Componentes Independentes, que será vista na Seção 2.5.

2.2 Caso Convolutivo

O modelo MLI não é suficiente para modelar um cenário acústico. Nesse tipo de cenário, as misturas são convolutivas, por causa dos atrasos que resultam da propagação do som através do espaço e do fenômeno de múltiplos percursos (*multipath*) gerado por reflexões do som em diferentes objetos (reverberação). Como resultado disto, cada uma das M misturas é filtrada por um sistema multicanal:

$$x_j(n) = \sum_{i=1}^{N} \left[\sum_{l=-\infty}^{\infty} h_{ji}(l) s_i(n-l) \right], \tag{2.10}$$

onde o filtro $h_{ji}(l)$ tem comprimento P, i.e, apenas P coeficientes não-nulos. Representando de outra forma, onde $*$ denota convolução:

$$x_j(n) = \sum_{i=1}^{N} (h_{ji} * s_i)(n) \tag{2.11}$$

A matriz $\mathbf{H}$ em (2.3) é redefinida em (2.12), onde cada elemento $\mathbf{h}_{ji} = [h_{ji}(0), h_{ji}(1), \cdots, h_{ji}(P-1)]$ é um filtro FIR (Resposta ao Impulso Finita,

do inglês *Finite Impulse Response*) de comprimento P. A notação $\underline{\mathbf{V}}$ foi utilizada para manter consistência com trabalhos anteriores, em especial as definições de Álgebra Linear FIR em [8], onde uma matriz ou vetor sublinhado simboliza uma matriz ou vetor em que cada elemento é um filtro FIR. Estenderemos esta definição para indicar não somente um filtro FIR, mas qualquer sinal finito, como o vetor $\mathbf{s}_i$ que contém todas as N_{amost} observações da fonte i.

$$\underline{\mathbf{H}} = \begin{bmatrix} \mathbf{h}_{11} & \mathbf{h}_{12} & \cdots & \mathbf{h}_{1N} \\ \mathbf{h}_{21} & \mathbf{h}_{22} & \cdots & \mathbf{h}_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{h}_{M1} & \mathbf{h}_{M2} & \cdots & \mathbf{h}_{MN} \end{bmatrix} \tag{2.12}$$

A Equação (2.2) é substituída por (2.13), onde o operador "$\cdot$" funciona como a multiplicação de matrizes, porém as multiplicações escalares são substituídas por convoluções entre vetores, novamente seguindo as definições de Álgebra Linear FIR de [8]. Cumpre notar nesta equação a concatenação das misturas, como em (2.4), assim como das fontes (ver Equação (2.5)). O vetor $\underline{\mathbf{s}} = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_N]^T$ segue a notação estabelecida acima, onde cada vetor-linha $\mathbf{s}_i$ contém as N_{amost} observações da fonte s_i ($\mathbf{s}_i = [s_i(1), s_i(2), \cdots, s_i(N_{\text{amost}})]$).

$$\mathbf{X} = \underline{\mathbf{H}} \cdot \underline{\mathbf{s}} \tag{2.13}$$

$$\mathbf{X} = \begin{bmatrix} \mathbf{h}_{11} * \mathbf{s}_1 & \mathbf{h}_{12} * \mathbf{s}_2 & \cdots & \mathbf{h}_{1N} * \mathbf{s}_N \\ \mathbf{h}_{21} * \mathbf{s}_1 & \mathbf{h}_{22} * \mathbf{s}_2 & \cdots & \mathbf{h}_{2N} * \mathbf{s}_N \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{h}_{M1} * \mathbf{s}_1 & \mathbf{h}_{M2} * \mathbf{s}_2 & \cdots & \mathbf{h}_{MN} * \mathbf{s}_N \end{bmatrix}$$

A matriz separadora $\underline{\mathbf{W}}$ é similar a $\underline{\mathbf{H}}$, porém contém os filtros $\mathbf{w}_{ij} = [w_{ij}(0), w_{ij}(1), \cdots, w_{ij}(Q-1)]$ separadores, de comprimento Q e tem dimensão $N \times M$:

$$\underline{\mathbf{W}} = \begin{bmatrix} \mathbf{w}_{11} & \mathbf{w}_{12} & \cdots & \mathbf{w}_{1M} \\ \mathbf{w}_{21} & \mathbf{w}_{22} & \cdots & \mathbf{w}_{2M} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{w}_{N1} & \mathbf{w}_{N2} & \cdots & \mathbf{w}_{NM} \end{bmatrix} \tag{2.14}$$

Depois de estimada a matriz, cada sinal estimado y_i é encontrado da seguinte forma:

$$y_i(n) = \sum_{j=1}^{M} \left[\sum_{l=-\infty}^{\infty} w_{ij}(l) x_j(n-l) \right] \tag{2.15}$$

ou na forma matricial, da mesma forma que feito em (2.13), onde

$\underline{\mathbf{x}} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M]^T$, e $\mathbf{x}_j = [x_j(1), x_j(2), \cdots, x_j(N_{\text{amost}})]$:

$$\mathbf{Y} = \underline{\mathbf{W}} \cdot \underline{\mathbf{x}} \tag{2.16}$$

$$\mathbf{Y} = \begin{bmatrix} \mathbf{w}_{11} * \mathbf{x}_1 & \mathbf{w}_{12} * \mathbf{x}_2 & \cdots & \mathbf{w}_{1M} * \mathbf{x}_M \\ \mathbf{w}_{21} * \mathbf{x}_1 & \mathbf{w}_{22} * \mathbf{x}_2 & \cdots & \mathbf{w}_{2M} * \mathbf{x}_M \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{w}_{N1} * \mathbf{x}_1 & \mathbf{w}_{N2} * \mathbf{x}_2 & \cdots & \mathbf{w}_{NM} * \mathbf{x}_M \end{bmatrix}$$

Encontrar a matriz $\underline{\mathbf{W}}$ é um problema muito mais difícil do que encontrar a matriz $\mathbf{W}$ do caso MLI. O método de Análise de Componentes Independentes, utilizado com sucesso no caso MLI, deve ser alterado para tratar o caso convolutivo. No Capítulo 3 este caso será tratado com mais detalhes, e suas especificidades serão abordadas ao longo da dissertação.

2.3 Ambiguidades de BSS

Mesmo que a separação seja bem-sucedida, algumas ambiguidades são inerentes à solução:

Ambiguidade do Escalamento As variâncias (energias) das componentes independentes não podem ser encontradas.

Isto acontece porque, como não conhecemos nem a fonte s_i nem o componente h_{ji} da matriz de mistura, qualquer fator k multiplicado a s_i poderia ser cancelado multiplicado-se h_{ji} por $\frac{1}{k}$, e a mistura x_j (que é a única informação disponível) seria a mesma, e torna-se impossível recuperar este valor de k. Isto também leva à *ambiguidade do sinal*. Um tratamento desta ambiguidade será dado na Seção 3.6.

Ambiguidade da Permutação Não é possível determinar a ordem das componentes independentes.

Isto também acontece porque não conhecemos nem o vetor das fontes $\mathbf{s}$ nem a matriz de mistura $\mathbf{H}$ em (2.2). A propriedade de comutação da soma de diversos termos torna irrelevante para esta operação a ordem destes. Isto implica a arbitrariedade do ordenamento, o qual, portanto, não é passível de recuperação na ausência de conhecimentos *a priori* acerca da matriz de mistura ou das fontes. Esta ambiguidade é discutida brevemente na Seção 3.5, e uma discussão detalhada aparece no Capítulo 4.

A Figura 2.1 ilustra melhor este problema. Note que a saída y_2 apresentou a ambiguidade do sinal, i.e, sua fase foi alterada de 180°.

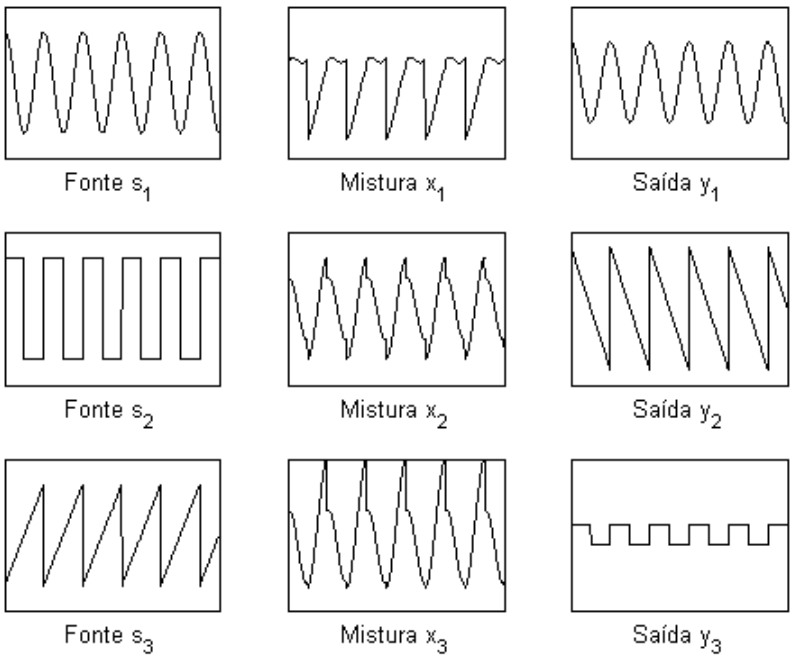

Figura 2.1: Ilustração da ambiguidade da solução BSS. As saídas do algoritmo de separação apresentam escalamentos aleatórios e estão desordenadas em relação às fontes.

Quando trabalhamos em cenários com transmissão de ondas acústicas no domínio do tempo, tais ambiguidades não constituem um problema grave, pois a ordem (permutação) das fontes não é uma informação importante, e o "volume" (escalamento) pode ser facilmente alterado. Quando trabalhamos no domínio da frequência, entretanto, como será visto no Capítulo 3, torna-se crítica a solução destas ambiguidades.

A ambiguidade de escalamento pode ser modelada multiplicando-se a matriz das fontes estimadas por uma matriz diagonal de ganhos $\mathbf{\Lambda}$:

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \lambda_N \end{bmatrix} \tag{2.17}$$

Da mesma forma, a ambiguidade da permutação pode ser modelada multiplicando-se a matriz $\mathbf{Y}$ por uma matriz de permutação $\mathbf{P}$, que consiste em uma matriz de zeros e uns, onde apenas um elemento de cada linha é 1. Um exemplo é mostrado em (2.18), para $N = 3$. Algumas vezes, utilizaremos a notação à direita em (2.18), que simboliza que a primeira e segunda linha da matriz devem ser

permutadas.

$$\mathbf{P}_{3\times3} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \triangleq \begin{Bmatrix} 2 \\ 1 \\ 3 \end{Bmatrix} \tag{2.18}$$

O exemplo (2.19) esclarece melhor estes conceitos. A operação $\mathbf{PV}$ permuta as linhas de $\mathbf{V}$, como mostrado no exemplo, e a operação $\mathbf{VP}^T$ permuta as colunas de $\mathbf{V}$.

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{y}_2 \\ \mathbf{y}_1 \\ \mathbf{y}_3 \end{bmatrix} \tag{2.19}$$

Resumindo, se a separação foi bem sucedida, podemos afirmar que existem $\mathbf{\Lambda}$ e $\mathbf{P}$ tais que:

$$\mathbf{S} = \mathbf{\Lambda P Y} \tag{2.20}$$

2.4 Estatísticas amostrais de números complexos

Em todas as aplicações práticas de processamento de sinais, incluindo BSS, não é possível obter estatísticas exatas sobre as variáveis. As médias, variâncias, curtose, e outras estatísticas de maior ordem devem ser estimadas utilizando-se as observações disponíveis. Estas estimativas devem ser o mais próximo possível dos valores reais para que o processamento seja efetivo. Nesta seção serão mostradas algumas definições utilizadas ao longo da dissertação envolvendo estimativas de estatísticas através das observações disponíveis, i.e, estatísticas amostrais. É importante rever alguns conceitos e introduzir outros que estão implícitos na maior parte da literatura de BSS disponível e são essenciais para o bom entendimento dos algoritmos e, principalmente, para sua implementação na prática. Adicionalmente, falaremos de estatísticas de números complexos, tema que normalmente não é abordado, por causa de sua aplicação limitada.

Para calcular o valor esperado $E\{.\}$ de uma variável, necessitamos de conhecimento acerca de sua densidade de probabilidade. Na prática, aproximamos o valor esperado utilizando as observações disponíveis. A média $\mu = E\{x(n)\}$, por exemplo, é calculada através de sua média amostral $\bar{x}$, onde N_{amost} é o número total de observações:

$$\bar{x} \triangleq \frac{1}{N_{\text{amost}}} \sum_{n=1}^{N_{\text{amost}}} x(n) \tag{2.21}$$

Se N_{amost} tende a infinito e as amostras disponíveis são iid (independente e identicamente distribuídas), então $\bar{x} = \mu_x$. No nosso caso, isto é impossível e o conceito de população se torna puramente teórico. O que temos é uma amostra da população, que supomos ser representativa. A média amostral complexa é calculada

da mesma forma que em (2.21).

A média amostral é uma medida *não-polarizada*. Isto significa que o valor esperado da média amostral é igual à média real. Segundo [9] mostra em seu Capítulo 5 (modificando a notação para ficar consistente com a nossa):

$$\text{Se } \bar{x} = (x_1 + x_2 + \cdots + x_N)/N \text{ com } E\{x_i\} = \mu_x \text{ para } i = 1, 2, \cdots, N,$$

$$E\{\bar{x}\} = \mu_x \tag{2.22}$$

$$\text{Se } x_1, x_2, \cdots x_N \text{ também forem independentes, com variância } \sigma_x^2,$$

$$\sigma_{\bar{x}} = \frac{\sigma_x}{N} \tag{2.23}$$

O resultado acima diz que, se considerarmos que cada observação tem média real μ_x, então o valor esperado da média amostral é idêntico à média real. Consideramos que cada um dos valores obtidos pelo sensor do microfone é uma variável aleatória de uma observação só, com média e variância específicas. Adicionalmente, se considerarmos que as observações são independentes umas das outras, podemos derivar a variância da média amostral. Embora este resultado tenha sido derivado para números reais, como a operação de adição de números reais é igual à de números complexos, este resultado pode ser estendido para números complexos. Na verdade, a expressão (2.21) pode ser utilizada para calcular o valor esperado de qualquer função da seguinte forma:

$$E\{f(x)\} = \frac{1}{N_{\text{amost}}} \sum_{n=1}^{N_{\text{amost}}} f(x(n)) \tag{2.24}$$

A variância de uma variável é dada por

$$\sigma_x^2 \triangleq E\{(x - E\{x\})^2\} = E\{x^2\} - \mu_x^2 \tag{2.25}$$

ou, no caso de números complexos, por

$$\sigma_x^2 \triangleq E\{(x - E\{x\})(x - E\{x\})^*\} = E\{|x|^2\} - |\mu_x|^2 \tag{2.26}$$

onde * denota complexo conjugado. Utilizando a mesma abordagem, a variância pode ser estimada atráves da variância amostral s_x^2 (considerando-se números complexos), dada por:

$$\mathsf{s}_x^2 \triangleq \frac{1}{N_{\text{amost}}} \sum_{n=1}^{N_{\text{amost}}} (x(n) - \bar{x})(x(n) - \bar{x})^* \tag{2.27}$$

onde $\bar{x}$ é a média amostral de x, dada por (2.21), que é *diferente da média estatística* μ_x (embora o valor esperado de $\bar{x}$ seja, segundo (2.22)). Como na prática só temos

acesso a uma amostra da população, a medida acima é considerada *polarizada*. Para entender porque isto acontece, vejamos o valor esperado da variância amostral, segundo mostrado em (2.27):

$$\begin{aligned} E\{\mathsf{s}_x^2\} &= E\left\{\frac{1}{N_{\text{amost}}}\sum_{n=1}^{N_{\text{amost}}}(x(n)-\bar{x})(x(n)-\bar{x})^*\right\} \\ &= \frac{1}{N_{\text{amost}}}\left[E\left\{\sum_{n=1}^{N_{\text{amost}}}x(n)x^*(n)\right\}+E\left\{\sum_{n=1}^{N_{\text{amost}}}\bar{x}\bar{x}^*\right\}\right. \\ &\quad \left. -E\left\{\sum_{n=1}^{N_{\text{amost}}}x(n)\bar{x}^*\right\}-E\left\{\sum_{n=1}^{N_{\text{amost}}}x^*(n)\bar{x}\right\}\right] \end{aligned}$$

De (2.21), e como $\bar{x}$ é constante:

$$E\{\mathsf{s}_x^2\} = \frac{1}{N_{\text{amost}}}\left[\sum_{n=1}^{N_{\text{amost}}}E\{x(n)x^*(n)\}-N_{\text{amost}}E\{\bar{x}\bar{x}^*\}\right]$$

Considerando (2.26), i.e, $E\{|x(n)|^2\} = E\{x(n)x^*(n)\} = \sigma_x^2 + |\mu_x|^2$, e (2.23) aplicado a números complexos, i.e, $E\{|\bar{x}|^2\} = \frac{\sigma_x^2}{N_{\text{amost}}} + |\mu_x|^2$, temos:

$$\begin{aligned} E\{\mathsf{s}_x^2\} &= \frac{1}{N_{\text{amost}}}\left[\sum_{n=1}^{N_{\text{amost}}}(\sigma_x^2+|\mu_x|^2)-N_{\text{amost}}\left(\frac{\sigma_x^2}{N_{\text{amost}}}+|\mu_x|^2\right)\right] \\ &= \tfrac{N_{\text{amost}}-1}{N_{\text{amost}}}\sigma_x^2 \end{aligned}$$

Portanto a Equação (2.27) representa uma medida polarizada da variância real σ_x^2. À medida que N_{amost} tende ao infinito, entretanto, a equação tende à variância real. Porém, uma medida não-polarizada da variância pode ser escrita da seguinte forma:

$$\mathsf{s}_x^2 = \frac{1}{N_{\text{amost}}-1}\sum_{n=1}^{N_{\text{amost}}}(x(n)-\bar{x})(x(n)-\bar{x})^* \tag{2.28}$$

a qual é bem parecida com a anterior, com uma diferença sutil. Na prática, em BSS, nos cálculos em que se utiliza variância, essa diferença é irrelevante, segundo os testes feitos, portanto, pode-se escolher qualquer uma das duas definições. Optamos pela definição não-polarizada (2.28).

A variância calculada segundo (2.28) é sempre real, e isto pode ser visto expandindo-se essa expressão. Observando a Equação (2.26), percebemos que o valor da variância segundo esta equação de uma variável complexa $\mathbf{x}(n)$ é similar à variância calculada segundo (2.25) a partir do módulo $|\mathbf{x}|$ desta variável, o que garante que a variância seja sempre real. Colocando de outra forma, se substituirmos o valor interno da soma em (2.28) por $|x(n)-\bar{x}|^2$, o resultado será o mesmo. Há outra forma de calcular a variância de números complexos, a qual [10] chama de

pseudovariância. Basta retirar o complexo conjugado da expressão (2.28), e o valor interno da soma em (2.28) fica $(x(n) - \bar{x})^2$, o que não garante que a variância seja sempre real.

O cálculo de estatísticas de maior ordem, como obliquidade (*skewness*, em inglês), a qual é uma estatística de terceira ordem, e curtose, também sofre do problema de medidas polarizadas, e no caso destas, é mais difícil chegar a medidas não-polarizadas. A medida "não-polarizada" da curtose, por exemplo, em geral é polarizada [11]. A definição de obliquidade que utilizaremos aqui (denotada por γ) é dada por (2.29), no caso de variáveis reais, e sua versão amostral (denotada por skew, para diferenciar) é dada por (2.30).

$$\gamma_y \triangleq \frac{E\{(y - \mu_y)^3\}}{\sigma^3} \tag{2.29}$$

$$\text{skew}(x) \triangleq \frac{\dfrac{1}{N_{\text{amost}}} \displaystyle\sum_{n=1}^{N_{\text{amost}}} (x(n) - \bar{x})^3}{\left[\dfrac{1}{N_{\text{amost}}} \displaystyle\sum_{n=1}^{N_{\text{amost}}} (x(n) - \bar{x})^2\right]^{\frac{3}{2}}} \tag{2.30}$$

Esta medida amostral é polarizada[2], entretanto optamos por utilizá-la. Assim como no caso da variância, e no da curtose, mostrado a seguir, a diferença é irrelevante.

A definição de obliquidade para o caso complexo é mais complicada. A definição (2.29) considera que a obliquidade é o momento central de terceira ordem dividido pelo cubo do desvio padrão. O problema com números complexos é que há ceil($\frac{p+1}{2}$) formas de se calcular o momento central de ordem p, onde ceil(r) arredonda o número real r para o maior número natural que não supera o argumento. Lembremos que a variância (que é igual ao momento central de segunda ordem) pode ser calculada de 2 formas (onde a segunda é a pseudovariância). O momento central de terceira ordem pode ser calculado como $E\{(y-\mu_y)^3\}$ ou $E\{(y-\mu_y)^2(y-\mu_y)^*\}$, portanto, há mais de uma forma de se calcular a obliquidade. Em [12], o autor define o que ele chama de *momento central absoluto de ordem p*, que é dado por $E\{|y-\mu_y|^p\}$, e esta será a definição que utilizaremos. Note que o momento central absoluto de segunda ordem fornece um resultado idêntico à variância que utilizamos, segundo discutido

[2] A medida não-polarizada é multiplicada por $\frac{\sqrt{N_{\text{amost}}(N_{\text{amost}}-1)}}{N_{\text{amost}}-2}$.

acima. De acordo com estas definições, a obliquidade para números complexos é:

$$\text{skew}(x) \triangleq \frac{\dfrac{1}{N_{\text{amost}}} \displaystyle\sum_{n=1}^{N_{\text{amost}}} |x(n) - \bar{x}|^3}{\left[\dfrac{1}{N_{\text{amost}}} \displaystyle\sum_{n=1}^{N_{\text{amost}}} |x(n) - \bar{x}|^2\right]^{\frac{3}{2}}} \tag{2.31}$$

A definição de curtose (denotada por curt) utilizada aqui, é dada pelo momento central de quarta ordem dividido pelo quadrado da variância, segundo a Equação (2.32), e sua versão amostral (denotada por $\mathsf{curt_{am}}$) é dada por (2.33).

$$\text{curt}(x) \triangleq \frac{E\{|y - \mu_y|^4\}}{\sigma^4} \tag{2.32}$$

$$\mathsf{curt_{am}}(x) \triangleq \frac{\dfrac{1}{N_{\text{amost}}} \displaystyle\sum_{n=1}^{N_{\text{amost}}} |x(n) - \bar{x}|^4}{\left[\dfrac{1}{N_{\text{amost}}} \displaystyle\sum_{n=1}^{N_{\text{amost}}} |x(n) - \bar{x}|^2\right]^{2}} \tag{2.33}$$

A medida de curtose mostrada também é polarizada. A versão mostrada em (2.32) é diferente da comumente usada na literatura estatística, onde se diminui o valor de 3, para que a curtose de uma distribuição gaussiana seja 0, e esta curtose é chamada de *curtose em excesso*. Não utilizaremos esta definição, portanto, a curtose de uma distribuição gaussiana será 3. A curtose é muito útil na diferenciação do tipo de distribuição de uma variável aleatória. Na literatura de BSS, é comum fazer distinção entre variáveis com distribuição gaussiana, variáveis com distribuição *subgaussiana* (a curtose neste caso é *menor* do que a curtose de uma variável com distribuição gaussiana) e variáveis com distribuição *supergaussiana* (a curtose neste caso é *maior* do que curtose de uma variável com distribuição gaussiana). Os sinais de voz, em geral, tem distribuição supergaussiana, i.e, sua curtose, segundo nossa definição, é maior do que 3. A Figura 2.2 ilustra melhor esta diferença. Na figura, todas as distribuições representadas tem variância 1.

Na literatura estatística, os nomes subgaussiana, gaussiana e supergaussiana são substituídos por platicúrtica, mesocúrtica e leptocúrtica, respectivamente. As distribuições supergaussianas são mais concentradas em torno da média, enquanto que as subgaussianas são mais "espalhadas". Um exemplo clássico de distribuição supergaussiana é a distribuição de Laplace (curt = 6) e um exemplo de distribuição subgaussiana é a distribuição uniforme (curt = 1, 8). O valor mínimo[3] para a curtose é 1, e o valor máximo é ∞.

[3]Este valor é atingido com uma distribuição discreta de dois pontos, de média zero, e simétrica em relação ao zero.

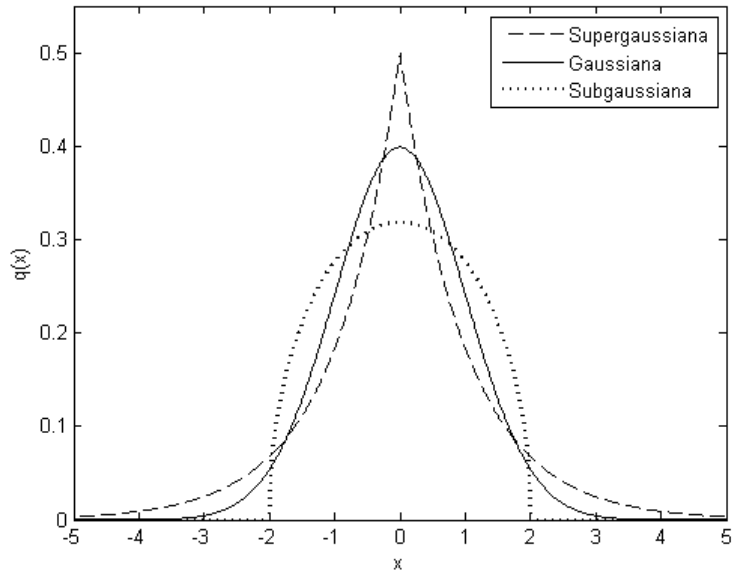

Figura 2.2: Comparação entre distribuições subgaussianas e supergaussianas.

A covariância, que é uma generalização da variância, entre duas variáveis x e y é dada por:

$$\mathrm{cov}_{xy} \triangleq E\{(x-\mu_x)(y-\mu_y)^*\} \tag{2.34}$$

A covariância de números complexos também pode ser calculada de outra forma, chamada de *pseudocovariância* [10], e é similar à pseudovariância citada anteriormente. O cálculo da covariância em sua forma amostral (denotada por $\overline{\mathrm{cov}}$) utilizado aqui é feito da seguinte forma:

$$\begin{aligned} \overline{\mathrm{cov}}_{xy} &\triangleq \frac{1}{N_{\mathrm{amost}}-1}\sum_{n=1}^{N_{\mathrm{amost}}}(x(n)-\bar{x})(y(n)-\bar{y})^* \\ &= \frac{1}{N_{\mathrm{amost}}-1}\left[\sum_{n=1}^{N_{\mathrm{amost}}}x(n)y^*(n)\right] - \frac{N_{\mathrm{amost}}}{N_{\mathrm{amost}}-1}\bar{x}\bar{y}^* \end{aligned} \tag{2.35}$$

que é uma medida não-polarizada. A matriz de covariância entre dois vetores-coluna aleatórios $\mathbf{x} = [x_1, x_2, \cdots, x_N]^T$ e $\mathbf{y} = [y_1, y_2, \cdots, y_M]^T$ é a matriz em que o elemento ij contém a covariância $\mathrm{cov}_{x_i y_j}$.

$$\mathbf{\Sigma}_{xy} \triangleq \mathbf{x}\mathbf{y}^H - \mu_{\mathbf{x}}{\mu_{\mathbf{y}}}^H = \begin{bmatrix} \mathrm{cov}_{x_1y_1} & \mathrm{cov}_{x_1y_2} & \cdots & \mathrm{cov}_{x_1y_M} \\ \mathrm{cov}_{x_2y_1} & \mathrm{cov}_{x_2y_2} & \cdots & \mathrm{cov}_{x_2y_M} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{cov}_{x_Ny_1} & \mathrm{cov}_{x_Ny_2} & \cdots & \mathrm{cov}_{x_Ny_M} \end{bmatrix} \tag{2.36}$$

onde $\mu_{\mathbf{x}} = [\mu_{x_1}, \mu_{x_2}, \cdots, \mu_{x_N}]$ e $\mu_{\mathbf{y}} = [\mu_{y_1}, \mu_{y_2}, \cdots, \mu_{y_M}]$. A matriz de covariância

amostral é similar, mas o operador cov é substituído por sua versão amostral $\overline{\text{cov}}$:

$$\overline{\mathbf{\Sigma}}_{xy} \triangleq \begin{bmatrix} \overline{\text{cov}}_{x_1y_1} & \overline{\text{cov}}_{x_1y_2} & \cdots & \overline{\text{cov}}_{x_1y_M} \\ \overline{\text{cov}}_{x_2y_1} & \overline{\text{cov}}_{x_2y_2} & \cdots & \overline{\text{cov}}_{x_2y_M} \\ \vdots & \vdots & \ddots & \vdots \\ \overline{\text{cov}}_{x_Ny_1} & \overline{\text{cov}}_{x_Ny_2} & \cdots & \overline{\text{cov}}_{x_Ny_M} \end{bmatrix} \tag{2.37}$$

Em processamento de sinais, em geral, a covariância é substituída pelo coeficiente de correlação (sendo estatisticamente rigoroso, do produto-momento Pearson), que chamaremos simplesmente de correlação (assim como é chamado na literatura de BSS). A correlação entre duas variáveis x e y é dada por:

$$r_{xy} \triangleq \frac{E\{xy\} - E\{x\}E\{y\}}{\sqrt{E\{(x - E\{x\})^2\}E\{(y - E\{y\})^2\}}} = \frac{\text{cov}_{xy}}{\sqrt{\sigma_x\sigma_y}} \tag{2.38}$$

Por definição, r_{ij} varia entre -1 e 1. A sua versão amostral é dada por:

$$\rho_{xy} \triangleq \frac{\overline{\text{cov}}_{xy}}{\sqrt{\mathsf{s}_x\mathsf{s}_y}} \tag{2.39}$$

A matriz de correlação entre dois vetores aleatórios coluna $\mathbf{x}$ e $\mathbf{y}$ é a matriz com as correlações entre as variáveis aleatórias dos vetores, ou seja:

$$\mathbf{R}_{xy} \triangleq \mathbf{\Sigma}_{xy} \circ \widehat{\sigma_{\mathbf{x}}}\widehat{\sigma_{\mathbf{y}}}^T = \begin{bmatrix} r_{x_1y_1} & r_{x_1y_2} & \cdots & r_{x_1y_M} \\ r_{x_2y_1} & r_{x_2y_2} & \cdots & r_{x_2y_M} \\ \vdots & \vdots & \ddots & \vdots \\ r_{x_Ny_1} & r_{x_Ny_2} & \cdots & r_{x_Ny_M} \end{bmatrix} \tag{2.40}$$

onde $\sigma_{\mathbf{x}} = [\sqrt{\sigma^2_{x_1}}, \sqrt{\sigma^2_{x_2}}, \cdots, \sqrt{\sigma^2_{x_N}}]^T$ é o vetor-coluna com os desvios padrões do vetor aleatório $\mathbf{x}$, e $\sigma_{\mathbf{y}}$ é o similar de $\mathbf{y}$. O operador $\circ$ implementa o produto Hadamard entre duas matrizes. O produto Hadamard realiza a multiplicação elemento a elemento entre duas matrizes de mesmas dimensões, i.e, $(\mathbf{A} \circ \mathbf{B})_{ij} = (\mathbf{A})_{ij} \cdot (\mathbf{B})_{ij}$ [13]. O operador $\widehat{\mathbf{A}}$ implementa a inversa de Hadamard, que é a inversa elemento-a-elemento da matriz $\mathbf{A}$, i.e, $(\widehat{\mathbf{A}})_{ij} = \frac{1}{(\mathbf{A})_{ij}}$. A operação conjunta $\mathbf{A} \circ \widehat{\mathbf{B}}$ é uma divisão elemento-a-elemento da matriz $\mathbf{A}$ sobre a matriz $\mathbf{B}$ de mesmas dimensões.

A versão amostral da matriz de correlação é similar a (2.40), com as correlações r_{ij} substituídas por suas versões amostrais ρ_{ij}:

$$\hat{\mathbf{R}}_{xy} \triangleq \overline{\mathbf{\Sigma}}_{xy} \circ \widehat{\mathsf{s}_{\mathbf{x}}}\widehat{\mathsf{s}_{\mathbf{y}}}^T = \begin{bmatrix} \rho_{x_1y_1} & \rho_{x_1y_2} & \cdots & \rho_{x_1y_M} \\ \rho_{x_2y_1} & \rho_{x_2y_2} & \cdots & \rho_{x_2y_M} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{x_Ny_1} & \rho_{x_Ny_2} & \cdots & \rho_{x_Ny_M} \end{bmatrix} \tag{2.41}$$

2.5 Análise de Componentes Independentes

Análise de Componentes Independentes, ou ICA, é um método para encontrar componentes ou fatores (no nosso caso, as fontes) de dados estatísticos multidimensionais (no nosso caso, as misturas), através de uma busca por componentes estatisticamente independentes e não-gaussianos. O método tenta encontrar a matriz separadora $\mathbf{W}$ de forma que as fontes estimadas y sejam estatisticamente independentes.

Para que os componentes possam ser separadas por ICA, eles devem atender a três condições [14]:

1. As componentes a serem encontradas (as fontes) devem ser *estatisticamente* independentes;

2. As componentes devem ser *não-gaussianas* [4];

3. A matriz separadora deve ser quadrada, i.e, o número de fontes e o de misturas devem ser iguais ($N = M$) [5].

Os algoritmos ICA podem ser derivados através da maximização da não-gaussianidade [16–18] ou da estimativa da ML (máxima verossimilhança) [19–21]. Embora também se possa utilizar InfoMax [22], o algoritmo resultante é similar ao obtido por estimativa da máxima verossimilhança (segundo apontado em [23]). A maioria dos algoritmos ICA são desenvolvidos para trabalhar com números reais, entretanto eles podem ser estendidos para trabalhar com números complexos, bastando para isso escolher apropriadamente a função G (no caso da maximização da não-gaussianidade) ou a função *score* (no caso da estimativa de ML), as quais serão abordadas a seguir.

Nas próximas seções, mostraremos os principais conceitos dos algoritmos ICA, suas limitações e algumas derivações. Derivações algébricas mais detalhadas de cada algoritmo ICA podem ser encontradas em [14, 17, 18, 24, 25].

2.5.1 Conceitos básicos

O grande desafio do ICA é medir a independência entre os componentes e maximizá-la. O primeiro desafio é descobrir quando dois componentes, no nosso caso variáveis aleatórias, são independentes.

[4] No máximo, uma das componentes pode ser gaussiana. Se isto não for verdade, a separação não vai além do branqueamento do vetor de misturas (ver Seção 2.5.1).

[5] Se $M > N$ (mais misturas que fontes), pode ser aplicada uma redução dimensional antes de se aplicar o ICA. O caso em que $N > M$ é muito mais difícil, e requer algoritmos específicos, alguns citados em [15], e está fora do escopo desta dissertação.

Podemos relacionar independência entre variáveis aleatórias (as nossas fontes estimadas y_i) com alguns conceitos, para entender melhor o problema. A descorrelação entre as variáveis (doravante chamadas de fontes) é um destes conceitos, e está relacionada com independência da seguinte forma: se N fontes $y_i(n)$, $i = 1, \cdots, N$ são independentes, então elas são descorrelacionadas, i.e, a covariância entre qualquer par delas é zero. Isto não significa que fontes descorrelacionadas são necessariamente independentes. Outro conceito (mais forte, como veremos) relacionado com independência é o de vetor aleatório branco. Seja $\mathbf{y} = [y_1(n), \cdots, y_N(n)]^T$ o vetor das saídas (fontes estimadas). Este vetor é branco quando seu vetor de médias $\mu_y = [\mu_{y_1}, \mu_{y_2}, \cdots, \mu_{y_N}]^T = [E\{y_1(n)\}, E\{y_2(n)\}, \cdots, E\{y_N(n)\}]^T$ é um vetor de zeros e a matriz de covariância $\mathbf{\Sigma}_{yy}$ (que contém a covariância $\text{cov}_{y_i y_j}$ entre todos os pares de saídas e as variâncias $\sigma^2_{y_i}$ das saídas na diagonal principal) é a matriz identidade, segundo mostrado em (2.42). Isto significa que a covariância cruzada entre as fontes estimadas é nula, e a variância de cada fonte é 1.

$$\begin{aligned} &\mathbf{\Sigma}_{yy} = E\{\mathbf{y}(n)\mathbf{y}^H(n)\} - \mu_y, \text{ mas } \mu_y = \mathbf{0} \quad \therefore \\ &\mathbf{\Sigma}_{yy} = \begin{bmatrix} E\{y_1^2(n)\} & E\{y_1(n)y_2(n)\} & \cdots & E\{y_1(n)y_N(n)\} \\ E\{y_2(n)y_1(n)\} & E\{y_2^2(n)\} & \cdots & E\{y_2(n)y_N(n)\} \\ \vdots & \vdots & \ddots & \vdots \\ E\{y_N(n)y_1(n)\} & E\{y_N(n)y_2(n)\} & \cdots & E\{y_N^2(n)\} \end{bmatrix} = \mathbf{I} \end{aligned} \tag{2.42}$$

Perceba que a propriedade de "branqueza" é mais forte do que a de descorrelação, pois se um vetor de fontes é branco, então todas as fontes são descorrelacionadas entre si. Na verdade, esta propriedade impõe $\frac{N(N+1)}{2}$ restrições [24], porém a matriz $\mathbf{W}$ possui N^2 parâmetros a serem encontrados, então sobram $\frac{N(N-1)}{2}$ parâmetros a serem determinados por outra informação estatistíca. Branqueando o vetor das fontes, fazemos aproximadamente *metade* do trabalho e só utilizamos estatísticas de no máximo segunda ordem.

Para esclarecer melhor o poder do branqueamento, considere a Figura 2.3. Cada ponto é uma observação do vetor-coluna de fontes $\mathbf{s}(n) = [s_1(n), s_2(n)]^T$, onde cada fonte s_i é um sinal de voz de 2 segundos de duração amostrado com uma frequência de amostragem $f_s = 16\,\text{kHz}$. A Figura 2.4 mostra as observações do vetor de misturas $\mathbf{x}(n)$, após as fontes serem misturadas com uma matriz $\mathbf{H}$ de dimensões 2×2, caracterizando uma mistura linear instantânea, i.e, o caso MLI.

A Figura 2.5 mostra o vetor $\mathbf{z}(n)$, que consiste nas misturas x_j da Figura 2.4 após passar por um branqueamento. Observe que basta uma rotação para que as fontes sejam separadas, ou seja, o branqueamento já fez metade do trabalho, como dito anteriormente. A Figura 2.6 mostra as saídas do ICA (as fontes estimadas).

Isso pode ser matematicamente mostrado da seguinte forma [24]: suponha, por

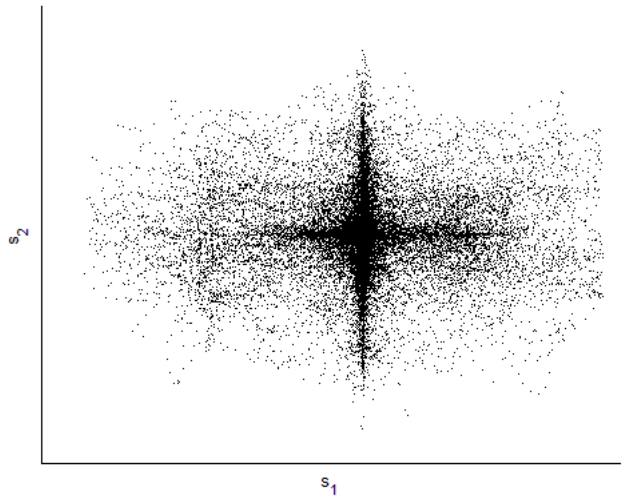

Figura 2.3: Observações do vetor de fontes $\mathbf{s}(n)$. Cada fonte s_i é um sinal de voz de 2 segundos de duração.

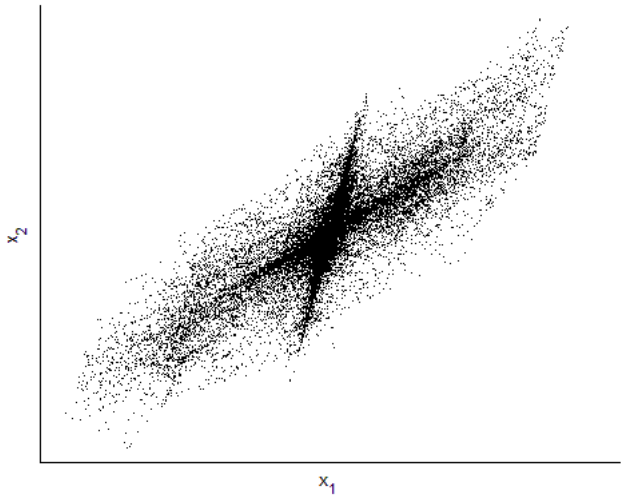

Figura 2.4: Observações do vetor de misturas $\mathbf{x}(n)$ instantâneas.

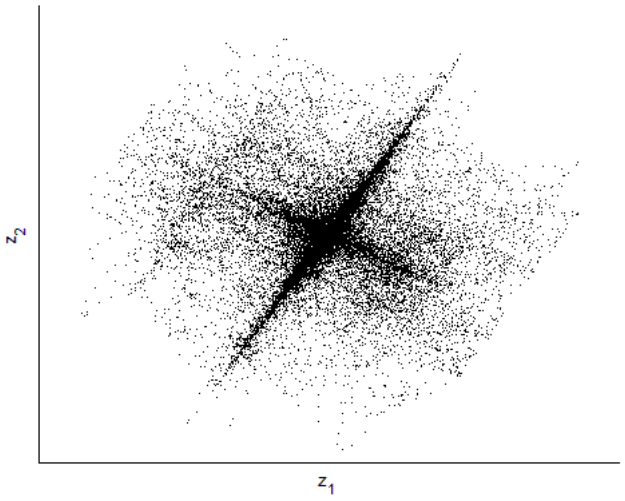

Figura 2.5: Observações do vetor branqueado $\mathbf{z}(n)$, que foi gerado através do branqueamento do vetor $\mathbf{x}(n)$.

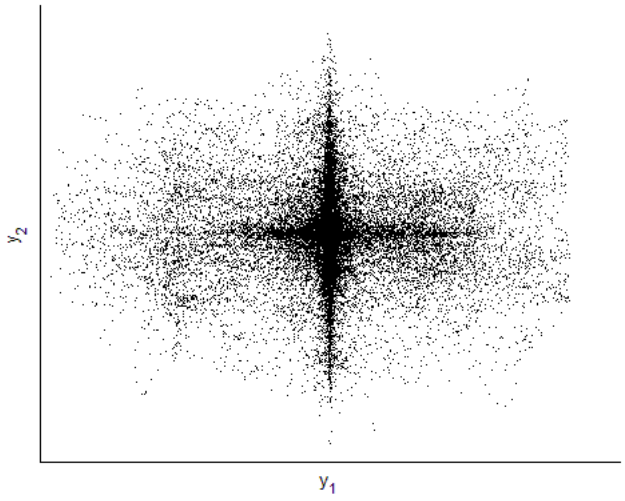

Figura 2.6: Observações do vetor de saída $\mathbf{y}(n)$.

simplicidade, que o vetor de fontes $\mathbf{s}$ é branco, i.e, sua matriz de covariância é a matriz identidade [6]. Suponha também que o branqueamento é realizado através de uma matriz branqueadora $\mathbf{V}$ tal que $\mathbf{z}(n) = \mathbf{V}\mathbf{x}(n)$. A matriz composta $\mathbf{VH}$ é uma rotação, pois $\mathbf{z}(n) = \mathbf{VHs}(n)$, ou seja, ela relaciona dois vetores brancos $\mathbf{s}(n)$ e $\mathbf{z}(n)$ (como o módulo deles é unitário, apenas o ângulo no plano i-dimensional muda, i.e, eles são rotacionados). A matriz separadora $\mathbf{W} = (\mathbf{VH})^{-1}$, tal que $\mathbf{y}(n) = \mathbf{W}\mathbf{z}(n)$, também é uma rotação. Isso mostra que aproximadamente metade do trabalho já foi feito pela matriz de branqueamento $\mathbf{V}$, e o ICA somente precisa encontrar uma matriz de rotação.

É importante ressaltar que não é estatisticamente eficiente *manter a restrição* de que o vetor de fontes estimadas seja branco durante a adaptação, pois isto pode implicar uma limitação no desempenho do algoritmo (ver Seção VI-B de [24] e [26]), i.e, a solução final não será a ótima. Por outro lado, branquear o vetor que contém as misturas antes de realizar a separação em si pode acelerar muito a convergência do algoritmo, pois, como dito acima, aproximadamente metade do trabalho já foi feito, e o algoritmo de branqueamento é mais rápido (computacionalmente falando) do que um algoritmo ICA de separação. Se as fontes s_i forem gaussianas, não há mais nada que se possa fazer além do branqueamento (ver Seção 7.5 de [14]), pois variáveis gaussianas descorrelacionadas já são independentes, e portanto o conceito de independência não pode ser utilizado para separá-las.

Para separar variáveis não-gaussianas, após o branqueamento, é necessário utilizar informações estatísticas de maior ordem, o que será explicado nas próximas seções. Se considerarmos outras suposições além da independência, podem-se utilizar informações estatísticas de segunda ordem para separar as variáveis (ver Seção 3.4.1).

2.5.2 Utilizando maximização da não-gaussianidade

Uma forma de medir independência entre duas estimativas de fontes é recorrer à sua não-gaussianidade, pois o Teorema Central do Limite em teoria da probabilidade diz (traduzido de [27]):

> Dadas n variáveis aleatórias independentes x_i, formamos sua soma $x = x_1 + \cdots + x_n$. Esta é uma variável aleatória com média $\eta = \eta_1 + \cdots + \eta_n$ e variância $\sigma^2 = \sigma_1^2 + \cdots + \sigma_n^2$. O Teorema Central do Limite diz que, dentro de certas condições gerais, a distribuição $F(x)$ de x se aproxima de uma distribuição normal com a mesma média e variância à medida que n cresce.

[6] Mesmo que o vetor $\mathbf{s}(n)$ não seja branco, há uma matriz $\mathbf{A}$ tal que $\mathbf{s}(n) = \mathbf{A}\mathbf{s}'(n)$, onde $\mathbf{s}'(n)$ é o vetor de fontes branco (segundo nossa suposição). Se considerarmos que $\mathbf{x}(n) = \mathbf{HA}\mathbf{s}'(n)$, a suposição continua sendo verdade, e nossa matriz de mistura é $\mathbf{HA}$, e o vetor de fontes é $\mathbf{s}'(n)$.

Portanto, quanto mais misturadas estiverem as fontes em determinada mistura, mais gaussiana ela estará. Duas formas são comumente utilizadas para medir a não-gaussianidade: uma é a curtose e a outra é a negentropia.

Curtose

A definição de curtose que utilizaremos nesta seção é (2.43), que chamaremos de curt_{ICA}[7].

$$\text{curt}_{ICA}(y) \triangleq E\{(y-\mu_y)^4\} - 3\sigma_y^4, \text{ onde } \mu_y = E\{y\} \text{ e } \sigma_y^2 = E\{(y-\mu_y)^2\} \quad (2.43)$$

Segundo a definição (2.43), se a curtose for nula, a variável tem distribuição gaussiana, se a curtose for positiva, a distribuição da variável é chamada de super-gaussiana, e se a curtose for negativa, a distribuição é subgaussiana. A diferença entre essas distribuições é ilustrada na Figura 2.2.

A definição de curtose mostrada aqui é diferente da Seção 2.4, que é a definição que utilizaremos no trabalho, por isso modificamos a notação. A definição mostrada aqui foi elaborada para que uma variável com distribuição gaussiana tenha curtose zero. Baseado nisso, devemos maximizar o módulo do valor da curtose de uma das fontes estimadas para maximizar a não-gaussianidade desta. Para comprovar que a medida em (2.43) é uma boa medida de não-gaussianidade, a Tabela 2.1 mostra a medida para alguns sinais e para a mistura destes. A medida foi normalizada apenas para que os números sejam mais tratáveis, e a Figura 2.7 mostra as distribuições das fontes em comparação com a distribuição gaussiana.

Tabela 2.1: Curtose de alguns sinais e da mistura destes

Sinal de voz de 6 segundos	$\frac{\text{curt}_{ICA}}{\sigma^4}$
Fonte 1 (voz feminina)	$4,007$
Fonte 2 (voz feminina)	$2,305$
Fonte 3 (voz masculina)	$17,553$
Fonte 4 (voz masculina)	$2,634$
Fonte 1 + Fonte 2	$1,584$
Fonte 1 + Fonte 3	$4,006$
Fonte 1 + Fonte2 + Fonte 3	$1,691$
Fonte 1 + Fonte2 + Fonte 3 + Fonte 4	$1,143$

[7]Essa definição diz que a curtose é o quarto cumulante κ_4 de y. A definição mais comum é o quarto cumulante dividido pelo quadrado do segundo cumulante (que é igual à variância) $\frac{\kappa_4}{\kappa_2^2} = \frac{\kappa_4}{\sigma^4}$, chegando na expressão $\text{curt} = \frac{E\{(y-\mu_y)^4\}}{\sigma^2} - 3$, que é a mais utilizada na literatura estatística.

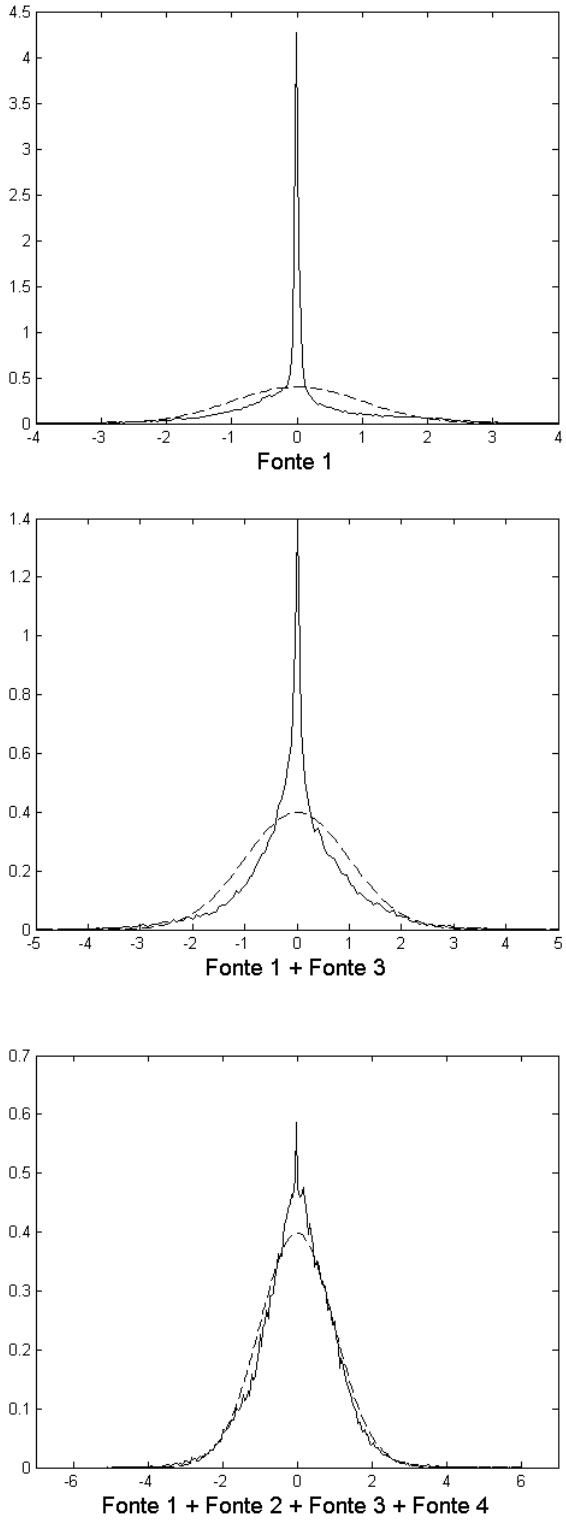

Figura 2.7: Distribuições das fontes da Tabela 2.1. A distribuição gaussiana é a tracejada, para comparação.

Primeiramente apresentaremos algoritmos de *uma unidade*, i.e, algoritmos que apenas estimam uma das fontes. Por isso, a matriz separadora será substituída pelo *vetor separador*, que consiste em apenas uma linha da matriz separadora:

$$\mathbf{w}_i = \begin{bmatrix} w_{i1} & w_{i2} & \cdots & w_{iM} \end{bmatrix} \tag{2.44}$$

Após a derivação dos algoritmos de uma unidade, serão apresentadas extensões destes que estimam todas as fontes desconhecidas.

Branquear as misturas antes de maximizar a curtose e aplicar a restrição de que o vetor separador deve ser unitário diminui o espaço de busca de soluções, o que faz com que o algoritmo seja mais rápido (embora a solução final possa não ser a ótima, como visto na Seção 2.5.1), além de facilitar a derivação do algoritmo. Uma explicação gráfica da necessidade do branqueamento e restrição no vetor separador pode ser vista na Seção 8.2.1 de [14]. Utilizar o branqueamento como pré-processamento é um pré-requisito do algoritmo de maximização do módulo da curtose.

Para derivar o algoritmo de maximização da curtose, segundo definida em (2.43), basta lembrar que nosso objetivo é descobrir a alteração no vetor $\mathbf{w}_i$ que aponta para a direção onde o módulo da curtose de $y_i(n)$ cresce mais, i.e, descobrir o gradiente do módulo da curtose em função da matriz separadora, considerando a restrição anterior. Seja $\mathbf{z}(n)$ o vetor das misturas $\mathbf{x}(n)$ após passar por um branqueamento. Sabemos que $\mu_{y_i} = 0$, pois $y_i(n) = \mathbf{w}_i\mathbf{z}(n)$, ou seja, é uma combinação linear (onde os pesos são os coeficientes do vetor $\mathbf{w}_i$) das misturas branqueadas $z_1(n), z_2(n), \cdots, z_M(n)$, todas com média zero ($\mu_{z_j} = 0, j = 1, 2, \cdots, M$). Ora, $\mu_{y_i} = E\{y_i(n)\} = w_{i1}E\{z_1(n)\} + w_{i2}E\{z_2(n)\} + \cdots + w_{iM}E\{z_M(n)\} = 0$. A partir disso, $\sigma^2_{y_i} = E\{(\mathbf{w}_i\mathbf{z}(n))^2\} = \|\mathbf{w}_i\|^2$, pois $E\{\mathbf{z}(n)\} = \mathbf{1}_{M\times 1}$ para sinais branqueados. Daí, o gradiente é

$$\frac{\partial|\text{curt}_{ICA}(\mathbf{w}_i\mathbf{z}(n))|}{\partial\mathbf{w}_i} = 4\,\text{sign}(\text{curt}_{ICA}(\mathbf{w}_i\mathbf{z}(n)))[E\{\mathbf{z}^T(n)(\mathbf{w}_i\mathbf{z}(n))^3\} - 3\mathbf{w}_i\|\mathbf{w}_i\|^2] \tag{2.45}$$

O algoritmo baseado no gradiente consiste em, a cada iteração, somar o gradiente da curtose em (2.45) ao vetor de separação $\mathbf{w}_i$. Note que, se expandirmos o gradiente, o último termo $(-12\,\text{sign}(\text{curt}_{ICA}(\mathbf{w}_i\mathbf{z}(n)))\mathbf{w}_i\|\mathbf{w}_i\|^2)$, quando somado ao vetor $\mathbf{w}_i$, somente altera o valor de sua norma, e não de sua direção, e por isso tal termo pode ser omitido, afinal o algoritmo deve projetar $\mathbf{w}_i$ na esfera unitária a cada iteração (o valor da norma de $\mathbf{w}_i$, $\|\mathbf{w}_i\|$, deve ser 1). Assim, encontramos o algoritmo de adaptação da matriz separadora baseado na maximização da curtose, que é dado pela Equação (2.46), onde η é o passo de adaptação e a segunda equação do algoritmo

serve para manter a restrição $\|\mathbf{w}_i\| = 1$.

$$\mathbf{w}_i \leftarrow \mathbf{w}_i + \eta \,\text{sign}(\text{curt}_{ICA}(\mathbf{w}_i\mathbf{z}(n))E\{\mathbf{z}^T(n)(\mathbf{w}_i\mathbf{z}(n))^3\} \tag{2.46}$$

$$\mathbf{w}_i \leftarrow \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|} \tag{2.47}$$

Deste ponto em diante, nesta seção, para simplificar a notação, (n) será omitido.

Como dito na Seção 2.5.1, a restrição de manter a matriz separadora unitária pode limitar o desempenho do algoritmo. Isto é uma limitação de todos os algoritmos que se utilizam de maximização da não-gaussianidade. A utilização destes algoritmos é justificada, porém, porque existem versões de ponto fixo de rápida convergência disponíveis, chamadas de FastICA.

O problema do algoritmo baseado no gradiente mostrado em (2.46) é que o passo de adaptação deve ser sabiamente escolhido, ou o algoritmo pode sofrer de convergência lenta (se for muito pequeno), ou chegar a um resultado muito diferente do ótimo (se for muito grande), ou ainda acabar divergindo. Esses problemas podem ser resolvidos utilizando-se versões de iteração para ponto fixo. Descreveremos um algoritmo de iteração para ponto fixo.

Seja f uma função definida para todos os números reais. A partir de um ponto inicial x_0, a *iteração para ponto fixo* é dada por:

$$x_{n+1} = f(x_n), n = 0, 1, 2, \cdots \tag{2.48}$$

que gera uma sequência $x_0, x_1, x_2, \cdots$, que deve convergir para um ponto fixo da função. Um ponto fixo na função f é um ponto onde

$$f(x_{FP}) = x_{FP} \tag{2.49}$$

Para que o algoritmo baseado no gradiente da curtose possa convergir para um ponto estável (o ponto fixo), o gradiente deve apontar na direção de $\mathbf{w}_i$. Apenas neste caso, quando adicionarmos o gradiente a $\mathbf{w}_i$ em (2.46), o vetor $\mathbf{w}_i$ manterá sua direção, embora sua norma $\|\mathbf{w}_i\|$ seja modificada. Como em cada iteração, $\mathbf{w}_i$ é dividido pela sua norma, quando o gradiente apontar na mesma direção de $\mathbf{w}_i$, o algoritmo convergiu para um ponto fixo.

Da equação do gradiente (2.45), chegamos à seguinte expressão:

$$\mathbf{w}_i \propto E\{\mathbf{z}^T(\mathbf{w}_i\mathbf{z})^3\} - 3\mathbf{w}_i\|\mathbf{w}\|^2 \tag{2.50}$$

A expressão (2.50) é a condição de convergência do algoritmo, como discutido acima. O termo $4\,\text{sign}(\text{curt}_{ICA}(\mathbf{w}_i\mathbf{z}(n)))$ é irrelevante, pois não altera a direção de $\mathbf{w}_i$, apenas sua norma. O lado direito da expressão $(E\{\mathbf{z}^T(\mathbf{w}_i\mathbf{z})^3\} - 3\mathbf{w}_i)$ é a nossa

função f em (2.48), e quando o algoritmo convergir, $f(\mathbf{w}_i) = \alpha\mathbf{w}_i$, onde α é um escalar que somente altera a norma de $\mathbf{w}_i$. Como aplicamos a restrição $\|\mathbf{w}_i\| = 1$ em cada iteração, α é eliminado depois de dividirmos $\mathbf{w}_i$ pela sua norma.

O algoritmo pode ser ainda mais simplificado, considerando que $\|\mathbf{w}_i\|^2 = 1$, chegando na forma final do algoritmo FastICA [16]:

$$\mathbf{w}_i \leftarrow E\{\mathbf{z}^T(\mathbf{w}_i\mathbf{z})^3\} - 3\mathbf{w}_i \tag{2.51}$$

$$\mathbf{w}_i \leftarrow \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|} \tag{2.52}$$

É sabido que o algoritmo que utiliza curtose possui alguns problemas na prática, quando seu valor deve ser estimado a partir das observações disponíveis, e os valores esperados em (2.43) devem ser substituídos por suas estimativas, segundo (2.24). O principal problema destas estimativas é sua falta de robustez com relação a *outliers*, que são observações da amostra que se desviam completamente de outras observações da mesma amostra (por exemplo, um 10 em uma amostra de 1000 observações onde todas as outras observações são menores do que 1). Um *outlier* altera completamente o valor da medida de curtose. Por este motivo, o algoritmo que utiliza curtose não é muito utilizado.

Negentropia

A negentropia é uma medida mais robusta do que a curtose, porém computacionalmente mais intensiva, embora haja aproximações mais simples que obtêm resultados satisfatórios.

A entropia é uma medida advinda da Teoria da Informação relacionada ao grau de incerteza do resultado de uma variável aleatória. Quanto maior o seu valor, mais aleatória (ou seja, imprevisível e sem estrutura determinada) é a variável. Um resultado fundamental de Teoria da Informação é que *uma variável contínua com distribuição gaussiana tem a maior entropia diferencial* entre todas as variáveis de igual variância. A entropia diferencial $H(\mathbf{y})$ de um vetor de variáveis aleatórias $\mathbf{y}$ com densidade de probabilidade $q(\mathbf{y})$ é dada por:

$$H(\mathbf{y}) \triangleq -\int q(\mathbf{y}) \log(q(\mathbf{y}))d\mathbf{y} \tag{2.53}$$

Um problema da entropia diferencial é que seu valor é alterado quando a variável aleatória é multiplicada por uma constante. Para resolver este problema, introduzimos a negentropia. A negentropia $J(\mathbf{y})$ é dada por:

$$J(\mathbf{y}) \triangleq H(\mathbf{y}_{gauss}) - H(\mathbf{y}) \tag{2.54}$$

onde $\mathbf{y}_{gauss}$ é um vetor com distribuição gaussiana e de mesma matriz de covariância $\mathbf{\Sigma}_{yy}$ que $\mathbf{y}$. A entropia $H(\mathbf{y}_{gauss})$ é:

$$H(\mathbf{y}_{gauss}) = \frac{1}{2}\log(|\det(\mathbf{\Sigma}_{yy})|) + \frac{N}{2}[1 + \log(2\pi)] \tag{2.55}$$

onde N é a dimensão de $\mathbf{y}$. Adicionalmente, a negentropia é invariante a qualquer transformação linear invertível. Isto pode ser provado facilmente, lembrando que $\mu_y = \mathbf{0}$, daí $\mathbf{\Sigma}_{(My)(My)} = \mathbf{M}\mathbf{\Sigma}_{yy}\mathbf{M}^H$, onde $\mathbf{M}$ é uma matriz (transformação linear) quadrada. A negentropia de $\mathbf{My}$ é:

$$\begin{aligned}
J(\mathbf{My}) &= \frac{1}{2}\log|\det(\mathbf{M}\mathbf{\Sigma}_{yy}\mathbf{M}^H)| + \frac{N}{2}[1 + \log(2\pi)] - (H(\mathbf{y}) + \log(|\det(\mathbf{M})|)) \\
&= \frac{1}{2}\log|\det(\mathbf{\Sigma}_{yy})| + 2\frac{1}{2}\log(|\det(\mathbf{M})|) + \frac{N}{2}[1 + \log(2\pi)] - H(\mathbf{y}) - \log(|\det(\mathbf{M})|) \\
&= \frac{1}{2}\log|\det(\mathbf{\Sigma}_{yy})| + \frac{N}{2}[1 + \log(2\pi)] - H(\mathbf{y}) \\
&= H(\mathbf{y}_{gauss}) - H(\mathbf{y}) = J(\mathbf{y})
\end{aligned}$$

o que prova a afirmação feita. Como a distribuição gaussiana $\mathbf{y}_{gauss}$ tem uma entropia H maior do que $\mathbf{y}$, segundo apontado anteriormente, a negentropia é sempre positiva, e é nula quando a distribuição de $\mathbf{y}$ é gaussiana. As vantagens citadas tornam esta medida preferida em detrimento da entropia diferencial.

Diferentemente da entropia, quanto maior a negentropia de uma variável aleatória, mais previsível é a variável e mais distante é a sua distribuição da distribuição gaussiana. Nosso objetivo, então, é maximizar a negentropia.

Calcular a negentropia diretamente pela definição é computacionalmente muito difícil. Na prática, só é necessária uma aproximação unidimensional, que chegue próximo do valor real. A aproximação utilizada na literatura é a (2.56), onde G é uma função não-quadrática tal que $E\{G\}$ é uma aproximação da entropia em (2.53).

$$J(y) \propto [E\{G(y_{gauss})\} - E\{G(y)\}]^2 \tag{2.56}$$

Com uma boa escolha de G, esta aproximação provou ser bastante útil. O algoritmo de adaptação da matriz separadora utilizando a maximização da negentropia é derivado tomando-se o gradiente de (2.56) em função da matriz separadora, assim como foi feito no caso da curtose. Este algoritmo possui as mesmas restrições do algoritmo que utiliza a curtose, ou seja, o vetor das misturas deve ser branqueado e

mantém-se $\|\mathbf{w}_i\| = 1$ a cada iteração:

$$\mathbf{w}_i \leftarrow \mathbf{w}_i + \eta[E\{G(y_{gauss})\} - E\{G(\mathbf{w}_i\mathbf{z})\}]E\{\mathbf{z}^T g(\mathbf{w}_i\mathbf{z})\} \tag{2.57}$$

$$\mathbf{w}_i \leftarrow \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|} \tag{2.58}$$

onde $g = G'$ é a derivada da função não-quadrática G, e y_{gauss} é uma variável com distribuição gaussiana e de mesma variância que $\mathbf{w}_i\mathbf{z}$.

Da mesma forma que feito com a curtose, pode-se derivar um algoritmo de ponto fixo para maximização da negentropia. De acordo com a discussão feita anteriormente, o algoritmo converge quando o gradiente apontar para a mesma direção que $\mathbf{w}_i$. Na expressão (2.57), o termo $E\{G(y_{gauss})\} - E\{G(\mathbf{w}_i\mathbf{z})\}$ é um escalar, e não altera a direção do gradiente, e uma primeira iteração para ponto fixo seria:

$$\mathbf{w}_i \leftarrow E\{\mathbf{z}^T g(\mathbf{w}_i\mathbf{z})\} \tag{2.59}$$

seguido da normalização de $\mathbf{w}_i$.

O problema de uma iteração como (2.59) é que a não-linearidade g não garante que o algoritmo venha a convergir rápido como no caso do FastICA usando curtose. Portanto, ele deve ser modificado. Somar $\alpha\mathbf{w}_i$ a ambos os lados da equação não altera o ponto fixo, daí:

$$\mathbf{w}_i \leftarrow \frac{1}{1+\alpha}E\{\mathbf{z}^T g(\mathbf{w}_i\mathbf{z})\} + \frac{\alpha}{1+\alpha}\mathbf{w}_i \tag{2.60}$$

Com uma boa escolha de α, o algoritmo pode ter boas propriedades de convergência. Esse parâmetro α pode ser encontrado partindo-se de outro ponto, segundo [16]. Basta lembrarmos que nosso objetivo é maximizar a negentropia, ou minimizar a entropia, que foi aproximada por $E\{G(\mathbf{w}_i\mathbf{z})\}$. Com a restrição de que $\|\mathbf{w}_i\| = 1$, podemos minimizar esta função objetivo (a estimativa da entropia) utilizando multiplicadores de Lagrange. A função de Lagrange a minimizar é:

$$\mathcal{L}(\mathbf{w}_i, \lambda) = E\{G(\mathbf{w}_i\mathbf{z})\} - \lambda(\|\mathbf{w}_i\|^2 - 1) \tag{2.61}$$

Podemos utilizar o método de Newton[8] para minimizar a função (2.61). As derivadas são:

$$\mathcal{L}'(\mathbf{w}_i, \lambda) = E\{\mathbf{z}^T g(\mathbf{w}_i\mathbf{z})\} - \lambda\mathbf{w}_i \tag{2.62}$$

[8]O método de Newton é um método iterativo onde o mínimo de uma função é encontrado através da iteração (aplicando ao nosso caso) $\mathbf{w}_i = \mathbf{w}_i - \frac{f'(\mathbf{w}_i)}{f''(\mathbf{w}_i)}$, sendo que $f(\mathbf{w}_i)$ é a função objetivo a minimizar.

$$\begin{aligned}\mathcal{L}''(\mathbf{w}_i, \lambda) &= E\{\mathbf{z}\mathbf{z}^T g'(\mathbf{w}_i\mathbf{z})\} - \lambda\mathbf{I} \\ &\approx E\{\mathbf{z}\mathbf{z}^T\}E\{g'(\mathbf{w}_i\mathbf{z})\} - \lambda\mathbf{I} \\ &= E\{g'(\mathbf{w}_i\mathbf{z})\}\mathbf{I} - \lambda\mathbf{I} \\ &= [E\{g'(\mathbf{w}_i\mathbf{z})\} - \lambda]\mathbf{I}\end{aligned} \tag{2.63}$$

A aproximação feita em (2.63) serve para tornar mais simples a inversão da matriz resultante de $\mathcal{L}''(\mathbf{w}_i, \lambda)$. Utilizando o método de Newton, temos:

$$\mathbf{w}_i \leftarrow \mathbf{w}_i - \frac{E\{\mathbf{z}^T g(\mathbf{w}_i\mathbf{z})\} - \lambda\mathbf{w}_i}{E\{g'(\mathbf{w}_i\mathbf{z})\} - \lambda} \tag{2.64}$$

Multiplicando ambos os lados por $E\{g'(\mathbf{w}_i\mathbf{z})\} - \lambda$:

$$\begin{aligned}[E\{g'(\mathbf{w}_i\mathbf{z})\} - \lambda]\mathbf{w}_i &\leftarrow E\{g'(\mathbf{w}_i\mathbf{z})\}\mathbf{w}_i - \lambda\mathbf{w}_i - E\{\mathbf{z}^T g(\mathbf{w}_i\mathbf{z})\} + \lambda\mathbf{w}_i \\ [E\{g'(\mathbf{w}_i\mathbf{z})\} - \lambda]\mathbf{w}_i &\leftarrow E\{g'(\mathbf{w}_i\mathbf{z})\}\mathbf{w}_i - E\{\mathbf{z}^T g(\mathbf{w}_i\mathbf{z})\}\end{aligned} \tag{2.65}$$

Como o termo à esquerda em (2.65) é um escalar (não altera a direção de $\mathbf{w}_i$), e o vetor $\mathbf{w}_i$ é normalizado a cada iteração, ele pode ser eliminado. Perceba que chegamos a uma versão parecida com (2.60), por um caminho diferente. Enfim, chegamos à versão rápida de ponto fixo do algoritmo (2.57), que é o FastICA:

$$\mathbf{w}_i \leftarrow E\{g'(y_i)\}\mathbf{w}_i - E\{\mathbf{z}^T g(y_i)\} \tag{2.66}$$

$$\mathbf{w}_i \leftarrow \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|} \tag{2.67}$$

onde $y_i = \mathbf{w}_i\mathbf{z}$ é a fonte i estimada, e g' é a derivada de g.

Este algoritmo só encontra uma das fontes independentes. Para encontrar todas as fontes, nos utilizamos do fato que cada vetor $\mathbf{w}_i$ que corresponde à separação da i-ésima fonte independente é ortogonal a todos os outros. Isto ocorre porque, para que as fontes y_i e $y_{i'}$ sejam independentes, elas devem ser não correlacionadas, ou seja, $E\{y_i y_{i'}\} = E\{\mathbf{w}_i\mathbf{z}\mathbf{w}_{i'}\mathbf{z}\} = 0$. Como o vetor $\mathbf{z}$ é branco, i.e, $E\{z_j z_{j'\neq j}\} = 0$, e $E\{z_j z_j\} = 1$, concluímos que $E\{y_i y_{i'}\} = E\{\mathbf{w}_i\mathbf{w}_{i'}^T\} = 0$, i.e, os vetores $\mathbf{w}_i$ e $\mathbf{w}_{i'}$ são ortogonais.

Assim sendo, podemos utilizar o algoritmo (2.66) para encontrar todas as fontes paralelamente, e, a cada iteração, nos certificarmos que a matriz $\mathbf{W}$ é ortogonal. A forma usual utilizada para ortogonalizar uma matriz [28] é:

$$\mathbf{W} \leftarrow (\mathbf{W}\mathbf{W}^H)^{-\frac{1}{2}}\mathbf{W}, \tag{2.68}$$

operação que deve ser feita a cada iteração. Isto nos leva ao algoritmo FastICA em

forma matricial:

$$\mathbf{W} \leftarrow E\{g(\mathbf{y})\mathbf{z}^H\} - \text{diag}(E\{g'(\mathbf{y})\})\mathbf{W} \tag{2.69}$$

$$\mathbf{W} \leftarrow (\mathbf{W}\mathbf{W}^H)^{-\frac{1}{2}}\mathbf{W} \tag{2.70}$$

onde diag($\mathbf{v}$) simboliza uma matriz diagonal (todos os elementos fora da diagonal principal são nulos) cuja diagonal principal é dada pelo vetor $\mathbf{v}$.

Resta agora escolher uma função G apropriada que tenha boas propriedades de convergência e ao mesmo tempo seja uma boa aproximação da entropia. A escolha mais natural, observando (2.53), seria $G(s_i) = -\log(q(s_i))$, segundo [17], onde $q(s_i)$ é a densidade de probabilidade estimada da fonte s_i. Este resultado se parece com o resultado do Natural ICA [19], visto adiante, do algoritmo InfoMax [22], que é similar a ele, e com o resultado independente de [21] (onde é chamado de *Relative Gradient*). O problema agora se torna estimar a densidade de probabilidade das fontes. Devido à similaridade com o Natural ICA, algumas densidades, juntamente com a função g resultante, são mostradas na Tabela 2.3, onde a função g é chamada de função *score*. Como o ICA trabalha somente com números reais, estas funções não são indicadas para trabalhar com números complexos. A Seção 3.4 mostrará como estender essas funções para trabalhar com números complexos.

Em [25], o autor propõe uma aproximação "*bottom-up*" para descobrir a função G, ou seja, estima funções não-lineares arbitrárias, que sejam de fácil cálculo e possuam boas propriedades de convergência, e depois prova que seus extremos coincidem com as fontes independentes. Algumas funções utilizadas na literatura são mostradas na Tabela 2.2, onde o simbolo * denota complexo conjugado. Estas funções foram escolhidas para trabalhar com números complexos (pode-se observar que todas se baseiam no módulo de y, menos a última), que é o nosso principal foco. Existem outras funções para trabalhar com números reais utilizadas na literatura, que não serão citadas.

2.5.3 Utilizando a estimativa por ML

Os algoritmos que utilizam estimativa por ML não colocam nenhuma restrição na matriz separadora, portanto teoricamente chegam a resultados mais precisos. Eles são baseados na maximização da verossimilhança da matriz $\mathbf{W}$ dadas as misturas observadas $\mathbf{x}(n)$. Lembrando que a verossimilhança de um conjunto de parâmetros de um modelo estatístico dadas as observações ($\mathcal{L}(\theta \mid x)$) tem relação com a densidade de probabilidade destas observações dados os parâmetros ($q(x \mid \theta)$). No nosso caso, os parâmetros são os elementos da matriz separadora e as observações são as misturas. A verossimilhança da matriz separadora $\mathbf{W}$ dadas N_{amost} observações de

Tabela 2.2: Funções comuns utilizadas no FastICA

$G(y)$	$g(y)$
$\sqrt{\lvert y\rvert^2 + \alpha}$	$\dfrac{1}{2\sqrt{\lvert y\rvert^2 + \alpha}}$
$\log(\lvert y\rvert^2 + \alpha)$	$\dfrac{1}{\lvert y\rvert^2 + \alpha}$
$\dfrac{1}{\alpha}\log(\cosh(\alpha\lvert y\rvert^2))$	$\tanh(\alpha\lvert y\rvert^2)$
$-\exp\left(\dfrac{-\lvert y\rvert^2}{2}\right)$	$y^* \exp\left(\dfrac{-\lvert y\rvert^2}{2}\right)$

um vetor de misturas $\mathbf{x}(n)$ é:

$$L(\mathbf{W} \mid \mathbf{x}(n)) = \prod_{n=1}^{N_{\text{amost}}} q(\mathbf{x}(n) \mid \mathbf{W}) \tag{2.71}$$

A expressão (2.71) assume que as observações $\mathbf{x}(n)$ são independentes entre si. Mesmo que na prática estas observações não sejam independentes, a técnica ainda obtém resultados razoavelmente precisos. Podemos entender que ela opta por não utilizar as dependências entre observações para o propósito de separação [29]. Lembrando do modelo ICA em (2.2), a probabilidade $q(\mathbf{x}(n))$ pode ser dada por (assumindo que $\mathbf{W} = \mathbf{H}^{-1}$):

$$q(\mathbf{x}(n) \mid \mathbf{W}) = |\det(\mathbf{W})| q(\mathbf{s}(n)) = |\det(\mathbf{W})| \prod_{i=1}^{N} q(s_i(n)) = |\det(\mathbf{W})| \prod_{i=1}^{N} q(\mathbf{w}_i \mathbf{x}(n)) \tag{2.72}$$

Substituindo em (2.71), chegamos à expressão de verossimilhança da matriz $\mathbf{W}$:

$$L(\mathbf{W} \mid \mathbf{x}(n)) = L(\mathbf{W}) = \prod_{n=1}^{N_{\text{amost}}} \prod_{i=1}^{N} q(\mathbf{w}_i \mathbf{x}(n)) |\det(\mathbf{W})| \tag{2.73}$$

Normalmente é utilizado o logaritmo da verossimilhança, pois o máximo é encontrado no mesmo ponto, e ele é algebricamente mais simples, pois o produto se transforma em soma. Para tornar a notação consistente com o que foi feito anteriormente, a expressão (2.74) teve a soma $\sum_{n=1}^{N_{\text{amost}}}$ substituída pelo operador $E\{\}$, que simboliza (somente neste caso) a estimativa do valor esperado, ou valor esperado amostral. Na prática, todos os valores esperados devem ser estimados, então essa substituição não tem consequências graves.

$$\log(L(\mathbf{W})) = E\{\sum_{i=1}^{N} \log(q(\mathbf{w}_i \mathbf{x}(n))) + N_{\text{amost}} \log(|\det(\mathbf{W})|)\} \tag{2.74}$$

O máximo da verossimilhança é encontrado iterativamente, utilizando-se o gradiente estocástico da verossimilhança. Dois algoritmos que se utilizam do princípio da estimativa da ML são o Natural ICA [19] e o algoritmo Bell-Sejnowski [20].

A derivação do algoritmo Bell-Sejnowski parte direto do gradiente de (2.73), que é dado por:

$$\frac{1}{N_{\text{amost}}}\frac{\partial \log(L(\mathbf{W}))}{\partial \mathbf{W}} = [\mathbf{W}^H]^{-1} - E\{\Phi(\mathbf{W}\mathbf{x}(n))\mathbf{x}^H(n)\} \tag{2.75}$$

onde a função Φ é dada por (2.81), explicada melhor adiante. A matriz separadora é adaptada por (2.76), onde $\mathbf{y} = \mathbf{W}\mathbf{X}$ simboliza as fontes estimadas.

$$\mathbf{W} \leftarrow \mathbf{W} + \eta((\mathbf{W}^H)^{-1} - E\{\Phi(\mathbf{y})\mathbf{x}\}) \tag{2.76}$$

Este algoritmo converge muito lentamente, por causa da inversão da matriz $\mathbf{W}$, que é uma operação computacionalmente intensiva, e deve ocorrer a cada iteração. Se os sinais de entrada forem branqueados antes de aplicar o algoritmo, a convergência melhora [14], mas o Natural ICA possui uma convergência melhor sem necessitar de nenhum pré-processamento (embora, em geral, os sinais sejam branqueados, de uma forma que será explicada abaixo), fato que tornou o algoritmo Bell-Sejnowski obsoleto.

O gradiente segundo a definição matemática usual aponta para a direção de maior inclinação em um espaço Euclidiano. Porém, o espaço de busca de parâmetros no ICA não é sempre Euclidiano, mas tem uma estrutura métrica Riemaniana [30], também chamada de geometria elíptica. Neste caso, deve ser utilizado o chamado *gradiente natural*, que aplicado à verossimilhança em (2.73), dá origem ao algoritmo Natural ICA. O gradiente natural de uma função objetivo $\mathcal{J}$ (no nosso caso (2.74)) em função do parâmetro que se deseja obter, que no nosso caso é a matriz separadora $\mathbf{W}$, é dado por (2.77). Ele difere do gradiente usual, mostrado em (2.78) para comparação, apenas pela multiplicação por $\mathbf{W}^H\mathbf{W}$, que é o quadrado do parâmetro $\mathbf{W}$.

$$\nabla_N \mathcal{J} = \frac{\partial \mathcal{J}(\mathbf{W})}{\partial \mathbf{W}}\mathbf{W}^H\mathbf{W} \tag{2.77}$$

$$\nabla \mathcal{J} = \frac{\partial \mathcal{J}(\mathbf{W})}{\partial \mathbf{W}} \tag{2.78}$$

Aplicando o gradiente natural à (2.74), chegamos a um resultado parecido com (2.75), mas sem a inversão de matriz, o que torna o algoritmo resultante computacionalmente muito mais simples:

$$\frac{1}{N_{\text{amost}}}\frac{\partial \log(L(\mathbf{W}))}{\partial \mathbf{W}}\mathbf{W} = [\mathbf{I} - E\{g(\mathbf{W}\mathbf{x}(n))\mathbf{W}\mathbf{x}^H(n)\}]\mathbf{W} \tag{2.79}$$

Este algoritmo é o mais utilizado na solução de problemas convolutivos, que serão tratados com mais detalhes no Capítulo 3. Uma derivação mais completa dele é matematicamente complexa, e pode ser vista com detalhes em [19] para o caso real, e uma derivação para o caso complexo (que chega no mesmo resultado) pode ser vista em [31].

A atualização da matriz separadora é dada por (2.80). Este algoritmo possui uma importante vantagem: enquanto que o desempenho da maioria dos algoritmos do tipo ICA depende bastante da matriz de mistura $\mathbf{H}$ [32], o Natural ICA se comporta bem mesmo quando $\mathbf{H}$ é mal condicionada. Isto ocorre porque (2.80) não contém nenhuma restrição (como a restrição de manter a matriz $\mathbf{W}$ unitária) e depende apenas dos sinais estimados $\mathbf{y}$ das fontes. Na Seção VI-C de [24], o autor prova matematicamente a afirmação anterior, com a condição de que o ruído possa ser negligenciado[9], o que em geral é verdade, afinal, em casos onde o ruído poderia ser relevante (por possuir uma variância alta), o problema de BSS já é difícil demais por si só, e provavelmente nenhum algoritmo trará resultados satisfatórios.

$$\mathbf{W} \leftarrow \mathbf{W} + \eta(I - E\{\Phi(\mathbf{y})\mathbf{y}^H\})\mathbf{W} \tag{2.80}$$

A função Φ é chamada de função *score*, e é calculada baseada na densidade de probabilidade das fontes. A Equação (2.81) mostra a relação entre a função *score* e a densidade de probabilidade estimada $q(y_i)$ das fontes estimadas y_i, considerando que y_i é um sinal real. A derivação da função $\Phi(y_i)$ para y_i complexo será tratada na Seção 3.4.

$$\Phi(y_i) = -\frac{\partial}{\partial y_i}\log(q(y_i)) \tag{2.81}$$

A densidade de probabilidade das fontes deve ser estimada, e deve ser não-gaussiana. Estas densidades não precisam ser exatas, porém há um limite para o quão erradas elas podem estar. Uma análise quantitativa deste erro é dada em [24], considerando o algoritmo Bell-Sejnowski, e para passos de adaptação η pequenos. As condições de estabilidade discutidas pelo autor não serão repetidas aqui, visto que o algoritmo Bell-Sejnowski não será utilizado.

A suposição de que a distribuição das fontes é conhecida é uma desvantagem dos algoritmos baseados em ML em relação aos algoritmos que utilizam maximização da não-gaussianidade. Se a distribuição estimada for muito diferente da distribuição real, os algoritmos baseados em ML terão desempenho inferior aos algoritmos anteriores, mesmo sem a restrição de que a matriz separadora seja unitária.

A Tabela 2.3 relaciona algumas das densidades de probabilidade para fontes reais utilizadas no algoritmo Natural ICA e suas respectivas funções *score*, onde σ é

[9] A matriz de covariância de $\mathbf{s}$ deve ser bem maior do que a matriz de covariância de $\mathbf{H}^{-1}\mathbf{n}$, onde cada linha de $\mathbf{s}$ é uma fonte, $\mathbf{n}$ é o ruído, e $\mathbf{H}$ é a matriz de mistura.

o desvio padrão estimado da densidade de probabilidade. Encontrar a função *score* para fontes complexas será discutido na Seção 3.4. Os nomes dados às funções *score* são apenas para futuras referências nesta dissertação.

Tabela 2.3: Funções *score* para diferentes densidades de probabilidade de fontes reais

Nome da distribuição	Densidade de probabilidade $q(y)$	Nome da função *score*	Função *score* $\Phi(y)$
Laplace	$\frac{1}{2\sigma}\exp\left(\frac{-\lvert y\rvert}{\sigma}\right)$	sign	$\frac{\text{sign}(y)}{\sigma}$
Laplace generalizada	$\frac{1}{B}\exp\left(\frac{-\sqrt{\lvert y\rvert^2+\alpha}}{\sigma}\right)$	genLaplace	$\frac{y}{2\sigma\sqrt{\lvert y\rvert^2+\alpha}}$
Cosseno hiperbólico	$\frac{1}{\pi\cosh\left(\frac{y}{\sigma^2}\right)}$	tanh	$\tanh\left(\frac{y}{\sigma^2}\right)$
Unimodal	$\frac{\exp\left(\frac{-2s}{\sigma^2}\right)}{\left(1+\exp\left(\frac{-2s}{\sigma^2}\right)\right)^2}$	tanh	$\tanh\left(\frac{y}{\sigma^2}\right)$
Gaussiana generalizada	$\frac{r}{2\sigma\Gamma\left(\frac{1}{r}\right)}\exp\left(-\frac{1}{r}\left\lvert\frac{y}{\sigma}\right\rvert^r\right)$	genGaussian	$\frac{\lvert y\rvert^{r-1}}{\sigma^r}\text{sign}(y)$

Na maioria dos algoritmos ICA, os sinais a serem separados são branqueados e o algoritmo é aplicado aos sinais branqueados (no caso do FastICA) ou a matriz branqueadora é utilizada como matriz inicial (no caso do Natural ICA) (a Seção 3.3 resume as vantagens do branqueamento). O algoritmo Natural ICA pode ser modificado para incluir o branqueamento simultaneamente à separação [33]. Tal algoritmo é mostrado em (2.82). Os testes iniciais realizados com este algoritmo mostraram que ele não tem um desempenho superior ao branqueamento seguido de Natural ICA, portanto ele não foi utilizado.

$$\mathbf{W} \leftarrow \mathbf{W} + \eta(I - E\{\mathbf{Y}\mathbf{Y}^H\} - E\{\Phi(\mathbf{Y})\mathbf{Y}^H\} + E\{\mathbf{Y}[\Phi(\mathbf{Y})]^H\})\mathbf{W} \tag{2.82}$$

Como o algoritmo Natural ICA é baseado em um gradiente, se a potência da fonte variar muito de um instante de tempo para outro, o algoritmo pode ter problemas de convergência (se for implementado na forma *on-line*). Para resolver esse tipo de problema, em [34], o autor deriva o Natural ICA de uma forma ligeiramente diferente, com restrições que ele chama de não-holonômicas, e chega à expressão (2.83), onde o operador maindg($\mathbf{V}$) extrai a diagonal principal da matriz $\mathbf{V}$.

$$\mathbf{W} \leftarrow \mathbf{W} + \eta(\text{diag}(\text{maindg}(E\{\Phi(\mathbf{y})\mathbf{y}^H\})) - E\{\Phi(\mathbf{y})\mathbf{y}^H\})\mathbf{W} \tag{2.83}$$

No nosso caso, sempre branqueamos o sinal antes de aplicar o algoritmo e a atualização é realizada com um algoritmo em batelada, i.e, que utiliza todas as

amostras de uma vez só, o que elimina esse tipo de preocupação. O Natural ICA não-holonômico pode vir a ser útil, no entanto, se for necessária uma aplicação online (lembrando que o fato de nosso algoritmo funcionar em batelada não impede que este seja em tempo real, contanto que ele consiga convergir rápido o suficiente).

2.6 Avaliação de Desempenho

Utilizamos ao longo desta dissertação a avaliação de desempenho proposta por [35]. Neste artigo, os autores decompõem a saída das fontes estimadas $y_i(n)$ como:

$$y_i(n) = [s_{target}]_i(n) + [e_{interf}]_i(n) + [e_{noise}]_i(n) + [e_{artif}]_i(n) \tag{2.84}$$

onde s_{target} é a fonte real, com alguma distorção aceitável, e_{interf} é a parcela do erro proveniente de interferências de outras fontes $i' \neq i$ na estimativa y_i da fonte, e_{noise} é a parcela do erro proveniente de ruído dos sensores, e e_{artif} é a parcela do erro que não provém nem de interferências nem de ruído.

O autor propõe 4 medidas de avaliação, das quais utilizaremos 3[10]:

1. SIR (Razão Sinal-Interferência) - mede a razão entre o sinal da fonte desejada e a interferência de outras fontes;

2. SDR (Razão Sinal-Distorção) - mede a razão entre o sinal desejado e as distorções provenientes de ruído, janelamento, transformações não-lineares, e inclusive interferência de outras fontes;

3. SAR (Razão Sinal-Artefatos) - mede a razão entre o sinal de saída (com interferências e ruído) e os artefatos, que são todas as distorções do sinal excluídas as interferências de outras fontes e o ruído dos sensores.

Essas medidas, considerando (2.84), são encontradas da seguinte forma:

$$\mathrm{SIR}_i \triangleq 10\log_{10}\frac{\|[\mathbf{s}_{target}]_i\|^2}{\|[\mathbf{e}_{interf}]_i\|^2} \tag{2.85}$$

$$\mathrm{SDR}_i \triangleq 10\log_{10}\frac{\|[\mathbf{s}_{target}]_i\|^2}{\|[\mathbf{e}_{interf}]_i + [\mathbf{e}_{noise}]_i + [\mathbf{e}_{artif}]_i\|^2} \tag{2.86}$$

$$\mathrm{SAR}_i \triangleq 10\log_{10}\frac{\|[\mathbf{s}_{target}]_i + [\mathbf{e}_{interf}]_i + [\mathbf{e}_{noise}]_i\|^2}{\|[\mathbf{e}_{artif}]_i\|^2} \tag{2.87}$$

onde todas as definições utilizam os vetores coluna $\mathbf{s}_x$ e $\mathbf{e}_x$ que contêm todas as N_{amost} observações dos sinais. Resta agora encontrar os valores de $\mathbf{s}_{target}$, $\mathbf{e}_{interf}$, $\mathbf{e}_{noise}$ e $\mathbf{e}_{artif}$. O autor propõe projeções ortogonais sobre os vetores das fontes estimadas

[10] O autor também define o SNR, que trata apenas dos ruídos dos sensores, desconsiderados por nós.

$\mathbf{y}_i$, que são os vetores coluna N_{amost} dimensionais que contém todas as observações de uma determinada fonte. São definidos um subespaço de vetores, que contém todos os vetores $\mathbf{y}_i$ das fontes estimadas, e três projeções ortogonais. Uma projeção ortogonal pode ser entendida de forma análoga a uma projeção de um vetor $\mathbf{v}$ sobre um outro vetor $\mathbf{u}$ num espaço bidimensional. No caso bidimensional, a projeção de $\mathbf{v}$ sobre $\mathbf{u}$ é $\mathbf{v}_{proj(u)} = \mathbf{v}\cos(\theta)$, onde θ é o ângulo entre $\mathbf{v}$ e $\mathbf{u}$. No caso N-dimensional, as projeções são um pouco mais complicadas, e suas deduções fogem ao escopo desta dissertação. O que precisamos saber é o resultado que provém dessas projeções. O autor define os valores desejados em função dessas projeções e depois as expande, chegando aos seguintes resultados:

$$[\mathbf{s}_{target}]_i = (\mathbf{y}_i^T \mathbf{s}_i)\frac{\mathbf{s}_i}{\|\mathbf{s}_i\|^2} \tag{2.88}$$

$$[\mathbf{e}_{interf}]_i = \sum_{i' \neq i} \left\{ (\mathbf{y}_i^T \mathbf{s}_{i'})\frac{\mathbf{s}_{i'}}{\|\mathbf{s}_{i'}\|^2} \right\} \tag{2.89}$$

$$[\mathbf{e}_{noise}]_i = (\mathbf{y}_i^T \mathbf{n}_j)\frac{\mathbf{n}_j}{\|\mathbf{n}_j\|^2} \tag{2.90}$$

$$[\mathbf{e}_{artif}]_i = \mathbf{y}_i - [\mathbf{s}_{target}]_i - [\mathbf{e}_{interf}]_i - [\mathbf{e}_{noise}]_i \tag{2.91}$$

onde $\mathbf{n}_j$ é o sinal com o ruído dos sensores, que desconsideramos nesta dissertação, fazendo $\mathbf{e}_{noise} = \mathbf{0}$. Estas equações definem o método de avaliação utilizado ao longo da dissertação.

Capítulo 3

Métodos de Separação Cega de Fontes no Domínio da Frequência

Em um ambiente real, os sinais de áudio são convoluídos com a resposta ao impulso de um filtro, que representa o caminho entre a fonte e os sensores. Ou seja, cada elemento da matriz separadora é um filtro (assim como mostrado na Seção 2.2). Sabendo que podemos aproximar uma convolução no domínio do tempo por uma multiplicação no domínio da frequência, uma forma de resolver este problema é aplicar a transformada de Fourier ao sinal, e resolver múltiplos problemas de misturas instantâneas. Em seguida, aplica-se a transformada inversa de Fourier para retornar ao domínio do tempo. Neste caso, qualquer algoritmo ICA que trabalhe com números complexos e misturas instantâneas pode ser utilizado, e o tempo de computação do algoritmo é reduzido consideravelmente. Entretanto, as ambiguidades de permutação e escalamento inerentes à solução de BSS (Seção 2.3) passam a ser relevantes, e precisam ser resolvidas, notadamente a da permutação. Embora ICA gere componentes independentes em cada raia de frequência, as componentes de frequência de uma mesma fonte devem ser agrupadas consistentemente antes que se aplique a transformada inversa de Fourier. Esta ambiguidade é bem conhecida na literatura como o problema da permutação de BSS no domínio da frequência (ou FDBSS). Resolver este problema é essencial para que a solução seja aceitável.

Outro problema relevante é a circularidade da representação da Transformada Discreta de Fourier (DFT). A multiplicação no domínio da frequência é equivalente à convolução circular no domínio do tempo, ou seja, o filtro no domínio do tempo deve ser periódico, o que não corresponde à realidade. Para que a multiplicação da resposta de um filtro no domínio da frequência por um trecho de sinal seja equivalente à convolução linear no domínio do tempo, a representação no domínio da frequência deve ter um número de raias maior ou igual ao tamanho do filtro somado ao tamanho do trecho do sinal, e o sinal inteiro deve ser reconstruído através da técnica *Overlap-Add* [36–39]. Esta técnica é chamada de *FFT Filtering*, utilizada para

fazer a convolução rápida de um filtro com um sinal muito longo, e é detalhada em [39]. Se estes critérios não forem obedecidos, o sinal resultante da multiplicação no domínio da frequência seguida de IDFT (Transformada Inversa Discreta de Fourier, do inglês *Inverse Discrete Fourier Transform*) será uma versão distorcida do sinal obtido se a convolução fosse feita no domínio do tempo.

Neste capítulo, descreveremos todas as etapas do método FDBSS, discorrendo sobre as abordagens encontradas na literatura para resolver cada um dos problemas descritos acima. Testes comparam a eficácia dos métodos, e modificações são propostas em algumas etapas para tentar melhorar os resultados.

3.1 Visão Geral

Uma visão geral do algoritmo de separação de fontes no domínio da frequência pode ser vista na Figura 3.1. O primeiro passo é transformar cada um dos sinais $x_j(n), j = 1, \cdots, M$ em suas representações $x_j(m,k), k = 0, \cdots, K-1$ (onde m é o índice do *frame* e K o número de raias) no domínio da frequência, utilizando a Transformada de Fourier de Curto Termo (STFT, do inglês *Short Time Fourier Transform*) para este propósito. Após este passo, é realizado um pré-processamento, que consiste no branqueamento dos sinais, gerando os sinais branqueados $z_j(m,k)$ e a matriz branqueadora $\mathbf{V}(k)$, seguido da separação propriamente dita. Esta separação supõe que os sinais $s_i(k), i = 1, \cdots, N$ no domínio da frequência são independentes para cada raia de frequência. Após a separação, é gerada uma matriz separadora $\mathbf{W}(k)$ em cada raia de frequência k, e o vetor com as saídas separadas $\mathbf{y}(m,k) = [y_1(m,k), \cdots, y_N(m,k)]^T$, ambos permutados e escalados. São então resolvidos os problemas de permutação e escalamento, através das matrizes $\mathbf{P}(k)$ e $\mathbf{\Lambda}(k)$ e é feito um pós-processamento opcional, que consiste na suavização da matriz separadora $\mathbf{W}(k)$. No fim, é utilizada a Transformada Inversa de Fourier de Curto Termo (ISTFT) para converter os sinais estimados das fontes no domínio da frequência em sinais no domínio do tempo. Estes passos serão detalhados nas próximas seções.

3.2 Transformação Tempo-Frequência

A STFT de um sinal de uma mistura x (rigorosamente, x_j, porém o índice j será omitido nesta seção) é dada por (3.1), onde cada $k = 0, \cdots, K-1$ representa uma frequência discreta $f_k \in \left\{0, \left(\frac{1}{K}\right) f_s, \cdots, \left(\frac{K-1}{K}\right) f_s\right\}$, K é o número de raias de frequência da DFT, L é o tamanho da janela, J é o salto (deslocamento da janela), e f_s é a frequência de amostragem. O símbolo $\hat{\jmath}$ representa a *unidade imaginária*, i.e, $\hat{\jmath} = \sqrt{-1}$. A janela de análise $win_a(n)$ é definida como sendo não-nula apenas

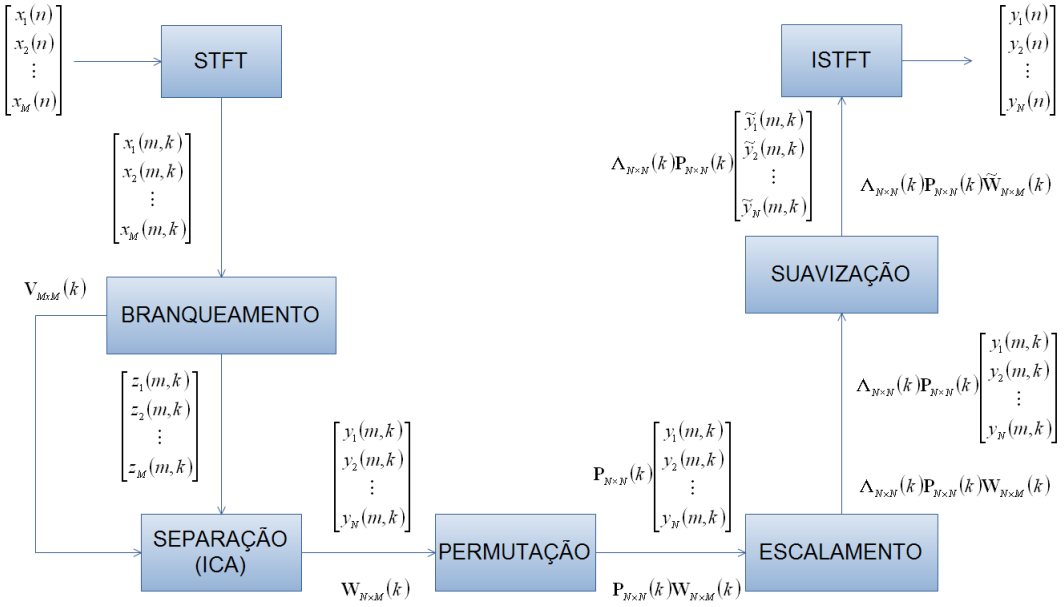

Figura 3.1: Diagrama geral do algoritmo completo de Separação de Fontes no Domínio da Frequência.

no intervalo $[0, (L-1)]$. O salto J é obviamente menor ou igual a L, ou haverá perda de observações (doravante chamadas de amostras). Este salto deve ser bem escolhido para que não haja distorção na síntese dos sinais.

$$X(m,k) = \sum_n x(n) win_a(n - mJ) \exp\left(-\hat{\jmath}\frac{2\pi kn}{K}\right), k = 0, \cdots, K-1 \qquad (3.1)$$

Observando a a Equação (3.1), vemos que $K = L$, mas há casos em que $K > L$, que são chamados de superamostrados (*oversampled*). Nestes casos, o sinal deve ser preenchido com zeros antes de passar para o domínio da frequência [40], o que é chamado na literatura de *zero-padding*.

Na prática, a STFT é calculada como em (3.2), onde DFT($\mathbf{v}$) simboliza a Transformada Discreta de Fourier do vetor $\mathbf{v}$, que pode ser realizada de forma rápida através da FFT [36] (Transformada Rápida de Fourier, do inglês *Fast Fourier Transform*). O vetor $\mathbf{X}(m) = [X(f_0, m), \cdots, X(f_{K-1}, m)]^T$, de comprimento K é o vetor com a representação em frequência do quadro (ou *frame*) m, e o vetor $\mathbf{x}_{frame}(m) = [x(mJ), \cdots, x(mJ + L - 1)]^T$, de comprimento L, representa o *frame* m no domínio do tempo. O vetor $\mathbf{win_a}$ contém os elementos não nulos da janela $win_a(n)$ mostrada em (3.1), ou seja, tem comprimento L. O produto diag($\mathbf{x}(mJ)$)$\mathbf{win_a}$ gera um vetor de comprimento L, e portanto, se $K > L$, são inseridas $K - L$ amostras nulas no final do vetor antes de ser aplicada a DFT, i.e,

é aplicado o *zero-padding* no sinal.

$$\mathbf{X}(m) = \text{DFT}(\text{diag}(\mathbf{x}_{frame}(m))\mathbf{win_a}) \tag{3.2}$$

A STFT inversa, ou ISTFT, é dada por (3.3), onde o índice i da fonte foi omitido. Os *frames* superpostos são adicionados para formar o sinal completo [40]. Esta técnica é chamada de *Overlap-Add*, doravante denominada OLA. A ISTFT também pode ser realizada segundo a Equação (3.4), onde é utilizada uma janela de síntese $win_s(n)$, que é não-nula apenas no intervalo $[0, (L-1)]$, assim como a janela de análise. Ambas são aplicadas no domínio do tempo. Esta técnica é chamada de WOLA (*Weighted Overlap-Add*) [39], e na verdade é uma generalização da OLA[1].

$$y(n) = \sum_m \left\{ \sum_k Y(m,k) \exp\left(\hat{\jmath}\frac{2\pi kn}{K}\right) \right\} \tag{3.3}$$

$$y(n) = \sum_m \left\{ win_s(n - mJ) \sum_k Y(m,k) \exp\left(\hat{\jmath}\frac{2\pi kn}{K}\right) \right\} \tag{3.4}$$

A ISTFT é calculada na prática por (3.5) e (3.6), onde o vetor $\mathbf{y}_{frame}(m) = [y_{frame}(0,m), \cdots, y_{frame}(K-1,m)]$ tem comprimento K, assim como o vetor $\mathbf{Y}(m)$. IDFT($\mathbf{v}$) simboliza a DFT inversa do vetor $\mathbf{v}$. A janela $\mathbf{win_s}$ contém os L elementos não-nulos de $win_s(n)$, e mais $K - L$ valores adicionados ao final do vetor, para que a janela tenha comprimento K [39], no caso de $K > L$ (superamostragem).

A operação WOLA é realizada em (3.6). O sinal $\mathbf{y}(n)$ tem comprimento igual ao número total N_{amost} de amostras do sinal, e a operação SHIFT($\mathbf{v}, a, c$) desloca o vetor $\mathbf{v}$ de a amostras e aumenta seu comprimento para c, de forma que o vetor resultante tenha apenas len_v elementos não-nulos, onde len_v é o tamanho do vetor v. Ou seja, em (3.6), o vetor resultante de SHIFT($\mathbf{y}_{frame}(m), mJ, N_{\text{amost}}$) tem apenas K elementos não-nulos, que são os únicos elementos nos quais a soma é realizada na prática.

$$\mathbf{y}_{frame}(m) = \text{diag}(\mathbf{win_s})\text{IDFT}(\mathbf{Y}(m)) \tag{3.5}$$

$$\mathbf{y}(n) = \sum_m \text{SHIFT}(\mathbf{y}_{frame}(m), mJ, N_{\text{amost}}) \tag{3.6}$$

Se o sinal não sofrer nenhuma modificação no domínio da frequência, o vetor IDFT($\mathbf{Y}(m)$) (a DFT inversa do *frame*) terá $K - L$ amostras nulas, portanto os valores adicionados à janela são irrelevantes, e podem ser nulos. Entretanto, se o sinal sofrer alguma modificação, as $K - L$ amostras podem não ser nulas. Em geral,

[1] A OLA nada mais é que uma WOLA com a janela de síntese retangular, ou seja, pesos iguais.

há dois tipos de modificações feitas no domínio da frequência:

- *FFT Filtering* - é uma transformação linear, que consiste em fazer uma convolução no domínio do tempo entre um filtro e um sinal longo através de multiplicações no domínio da frequência entre a resposta na frequência do filtro e trechos do sinal (que posteriormente são somados utilizando OLA). Algumas considerações devem ser satisfeitas (ver (3.7) e (3.8)).

- Transformação Não-Linear - a não-linearidade se refere ao domínio do tempo, i.e, transformações lineares nos trechos de sinal no domínio da frequência podem resultar em transformações não-lineares, se alguns cuidados não forem tomados.

No caso de *FFT Filtering*, a janela win_s deve ser uma janela retangular de comprimento K, i.e, deve ser aplicado OLA ao invés de WOLA, pois os *frames* de entrada de comprimento L foram expandidos para um comprimento K, devido à oscilação do filtro, e uma janela diferente da retangular descartaria amostras importantes ou distorceria o sinal, não gerando o resultado desejado [39]. Para que o *FFT Filtering* gere o resultado esperado, as condições (3.7) e (3.8) devem ser satisfeitas, onde Q é o comprimento do filtro. A condição (3.8) é chamada de condição COLA (*Constant Overlap-Add*). Se não houver transformação ou a transformação for não-linear, a condição é a (3.9), onde a janela de síntese é levada em consideração. Na prática, estas condições não são satisfeitas nas bordas do sinal, como visto na Figura 3.2.

$$K \geq L + Q - 1 \tag{3.7}$$

$$\sum_m win_a(n - mJ) = c,\ c \text{ constante},\ \forall n \in \mathbb{Z} \tag{3.8}$$

$$\sum_m win_a(n - mJ)win_s(n - mJ) = c,\ c \text{ constante},\ \forall n \in \mathbb{Z} \tag{3.9}$$

Em geral, na literatura, utiliza-se a janela de Hanning como janela de análise, definida por (3.10), que atende à COLA quando J é igual a $\frac{L}{2}$ ou $\frac{L}{4}$. A janela de síntese é normalmente a retangular, pelos motivos descritos acima. Na literatura, é comum encontrarmos a nomenclatura 25% Overlap quando $J = \frac{3L}{4}$, 50% Overlap quando $J = \frac{L}{2}$ e 75% Overlap quando $J = \frac{L}{4}$, que indicam o quanto uma janela em um determinado instante de tempo n se sobrepõe à janela do instante de tempo adjacente $n + mJ$ para determinado salto J. A Figura 3.2 mostra a janela de Hanning com um salto $J = \frac{L}{4}$, e, para comparação, a Figura 3.3 mostra a janela de Kaiser,

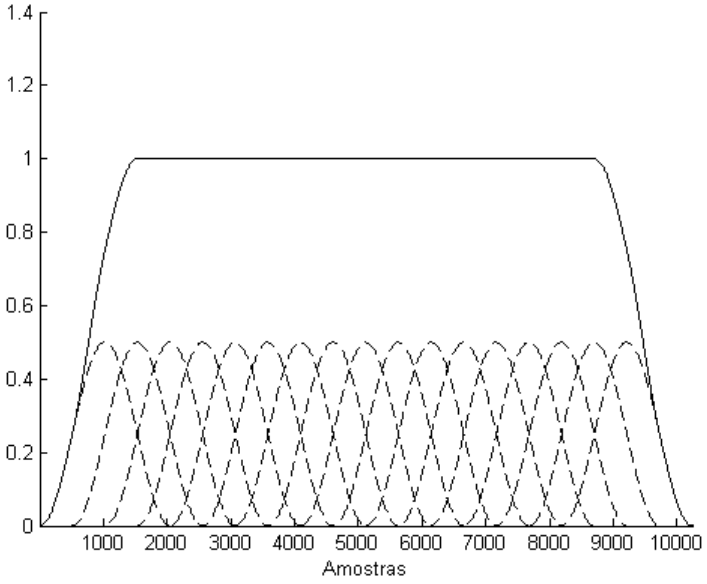

Figura 3.2: Ilustração do OLA com a janela de Hanning, que atende à COLA para $J = \frac{L}{4}$. A janela tem tamanho de 2048 amostras, e está indicada pela linha tracejada, e o Overlap-Add com salto de 512 amostras está indicado pela linha contínua.

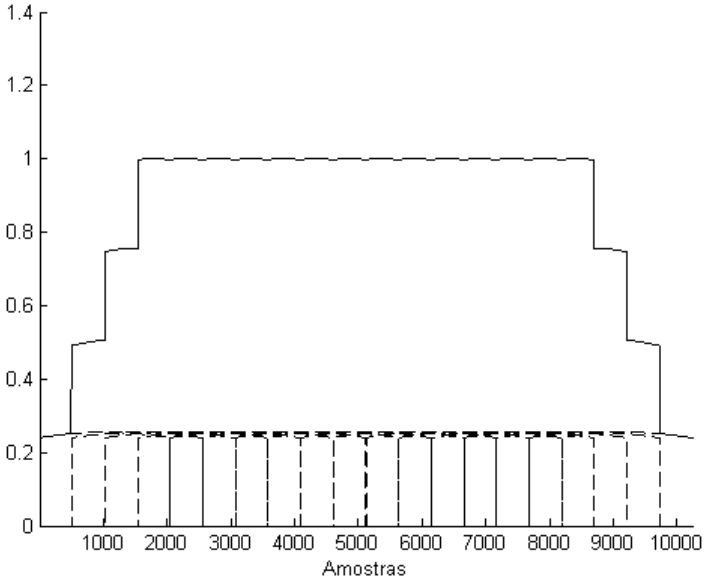

Figura 3.3: Ilustração do OLA com a janela de Kaiser ($\beta = 0.5$), que não atende à COLA. A janela tem tamanho 2048 amostras, e está indicada pela linha tracejada, e o Overlap-Add com salto de 512 amostras está indicado pela linha contínua.

que não atende à COLA para nenhum valor de J [2]. Observe que no caso da janela de Hanning, a constante não é 1 quando $J = \frac{L}{4}$, mas na prática, escalamos as janelas para que esta constante seja 1 (no caso da janela de Hanning, basta multiplicar (3.10) por $0,5$, como mostrado em (3.11)).

$$w_{hanning}(n) = 0,5\left(1 - \cos\frac{2\pi n}{L}\right) \tag{3.10}$$

$$[w_{hanning}(n)]_{scale} = 0,5 w_{hanning}(n) = 0,25\left(1 - \cos\frac{2\pi n}{L}\right) \tag{3.11}$$

Qualquer janela que obedeça à COLA para um determinado salto J pode ser utilizada como janela de análise. Alguns exemplos são mostrados na Tabela 3.1. Para ilustrar o efeito de uma janela que não obedeça esta condição no desempenho do algoritmo, foi realizado um teste com a janela Blackman-Harris com salto de $J = \frac{L}{4}$ (atende à COLA) e $J = \frac{L}{2}$ (não atende à COLA) e o resultado é mostrado na Tabela 3.2. Foi simulada a resposta de frequência de uma sala segundo a Figura A.1. Os resultados são a média de 10 simulações. Maiores detalhes sobre o ambiente de teste podem ser vistos no Apêndice A.

Tabela 3.1: Janelas que obedecem à COLA.

Janela	Equação	Pulo J para atender à COLA
Hanning	$0,5\left(1 - \cos\left(\frac{2\pi n}{L}\right)\right)$	$\frac{L}{2}$ ou $\frac{L}{4}$
Chebyshev	$\text{IDFT}\left(\frac{\cos\left(L\cos\left(\beta\cos\left(\frac{\pi n}{L}\right)\right)^{-1}\right)}{\cosh(L\cosh(L\cosh(\beta)^{-1})^{-1})}\right)$, $\beta = \cosh\frac{1}{L}\cosh 10^{\alpha-1}$, $\alpha = 5$, no nosso caso	$\frac{L}{4}$
Blackman-Harris mínima de 4 termos [41]	$0,35875 - 0,48829\cos\left(2\pi\frac{n}{L}\right) + 0,14128\cos\left(2\pi\frac{2n}{L}\right) + 0,01168\cos\left(2\pi\frac{3n}{L}\right)$	$\frac{L}{4}$
Nuttall mínima de 4 termos [42]	$0,3635819 - 0,4891775\cos\left(2\pi\frac{n}{L}\right) + 0,1365995\cos\left(2\pi\frac{2n}{L}\right) + 0,0106411\cos\left(2\pi\frac{3n}{L}\right)$	$\frac{L}{4}$

Embora a janela de Hanning seja a mais utilizada, não significa que outras não sejam empregadas. Para avaliar o desempenho de cada janela, foram realizados testes utilizando janelas diferentes e os resultados estão mostrados na Tabela 3.2.

[2] A maior aplicação da janela de Kaiser é no projeto de filtros FIR, onde a condição COLA não é importante.

Foram escolhidas aleatoriamente 10 combinações 2 a 2 dentre 8 locutores disponíveis, ou seja, foram feitas 10 realizações e foi tirada a média para cada janela. Foram realizados mais testes modificando-se o método para resolver a permutação, tamanho da janela, número de raias da FFT e número de realizações que chegaram a resultados similares[3], e portanto não serão mostrados aqui. O método para resolver a permutação é explicado no Capítulo 4, e é indiferente para a análise realizada aqui. Ele foi colocado simplesmente para futura referência. A janela de Hanning obteve resultados melhores. Segue-se uma possível explicação para isso.

Tabela 3.2: Comparação do desempenho em BSS quando a janela win_a da STFT é modificada.

Número de fontes e misturas - $N = M = 2$
Tempo de reverberação - $T_{60} = 130\,\text{ms}$
Número de raias da FFT - $K = 4096$
Tamanho da janela - $L = 2048$
Separação - Natural ICA com sign ($\eta = 0,1$)
Método para resolver a permutação - DOA + HarmCorr
Número de realizações - 10
Disposição dos microfones e fontes - Figura A.1

Janela	Pulo J	Atende à COLA?	SIR médio	SDR médio	SAR médio
Retangular	2048	Sim	$17,2\,\text{dB}$	$9,8\,\text{dB}$	$10,9\,\text{dB}$
Hanning	1024	Sim	$17,6\,\text{dB}$	$10,4\,\text{dB}$	$11,7\,\text{dB}$
Hanning	512	Sim	$18,7\,\text{dB}$	$11,4\,\text{dB}$	$12,9\,\text{dB}$
Chebyshev	512	Sim	$18,0\,\text{dB}$	$11,0\,\text{dB}$	$12,5\,\text{dB}$
Blackman-Harris	512	Sim	$17,9\,\text{dB}$	$10,9\,\text{dB}$	$12,5\,\text{dB}$
Blackman-Harris	1024	Não	$17,4\,\text{dB}$	$7,6\,\text{dB}$	$8,3\,\text{dB}$
Nuttall	512	Sim	$17,8\,\text{dB}$	$10,9\,\text{dB}$	$12,4\,\text{dB}$

Como é sabido, multiplicar uma janela por um trecho de sinal é equivalente a filtrar a resposta em frequência desse trecho com a resposta em frequência da janela, que é o mesmo que dizer que uma multiplicação no domínio do tempo equivale a uma convolução no domínio da frequência. Pensando assim, o ideal seria encontrar uma janela que tenha como resposta em frequência um impulso, pois, desta forma, a resposta em frequência do nosso sinal (que é a informação desejada) ficaria intacta. É claro que uma janela desse tipo é irrealizável, pois a transformada inversa de Fourier de um impulso é um sinal igual a 1 para todo tempo n. Isso significa que uma janela "perfeita" trata o sinal todo de uma vez só, e isso é exatamente o que não queremos fazer, pois não teríamos nenhuma resolução temporal: cada raia de

[3]Por resultados similares, quero dizer que as conclusões tiradas não são diferentes, i.e, embora a SIR seja diferente, as mesmas janelas obtiveram melhores resultados.

frequência conteria a informação de todo o sinal, e seria inútil do ponto de vista estatístico. Logo, é necessário sacrificar a resolução na frequência para conseguir alguma resolução temporal [43], e o compromisso entre as duas depende da aplicação específica.

A resposta de frequência normalizada da janela de Hanning é mostrada na Figura 3.4, onde a escala do eixo das ordenadas é logarítmica para facilitar a visualização dos lóbulos. A escala utilizada é a de decibéis ($10\log(|win_a(f)|^2)$, no domínio da frequência). É comum normalizar a resposta para que o pico esteja em 0 dB. O eixo das abcissas mostra uma medida adimensional, de frequência sobre frequência ($\frac{f}{f_s/L}$), onde f_s é a frequência de amostragem da representação no *domínio do tempo* da janela e L o comprimento da janela em número de amostras[4]. Esta normalização foi cuidadosamente escolhida para que o valor L no eixo corresponda à frequência de amostragem f_s, pois dessa forma, a escala do eixo está em raias de frequência, e isso facilita muito a visualização e entendimento do gráfico. Existem três parâmetros importantes que definem uma janela, baseado na resposta de frequência desta: a largura do lóbulo principal, a amplitude no primeiro lóbulo lateral e o decaimento com a frequência. Os três estão representados na figura. A largura do lóbulo principal é multiplicada por 2 porque a resposta de frequência também se estende para as frequências negativas.

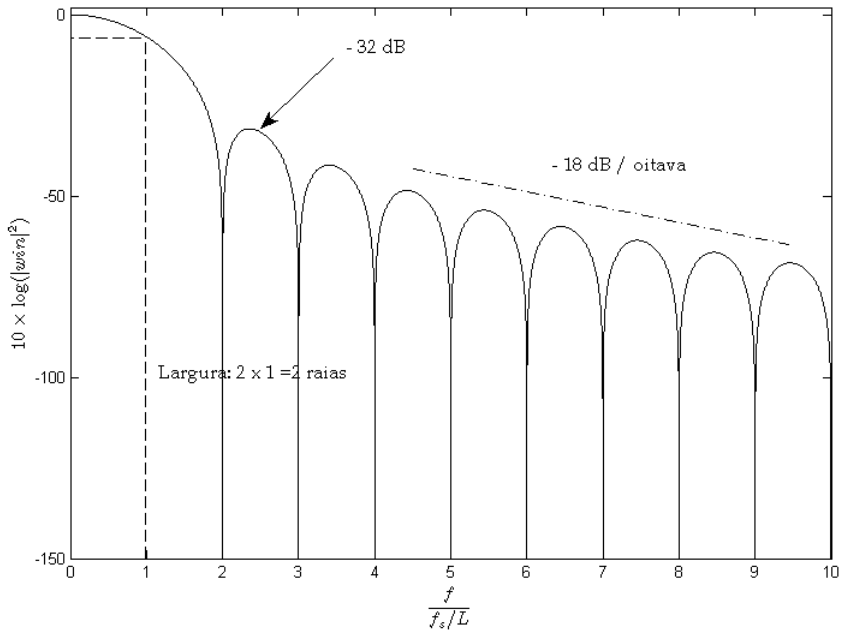

Figura 3.4: Resposta em frequência da janela de Hanning.

[4]Sendo mais criterioso, como esta resposta de frequência está no domínio da frequência *contínua*, a normalização deveria ser substituída por $f \cdot T_0$, onde T_0 é o tempo (em segundos) da janela. Se a janela for amostrada no tempo com frequência f_s, então $T_0 = \frac{L}{f_s}$, e chegamos ao mesmo resultado. Na literatura é comum encontrar a notação T_0, mas achamos mais claro utilizar f_s/L.

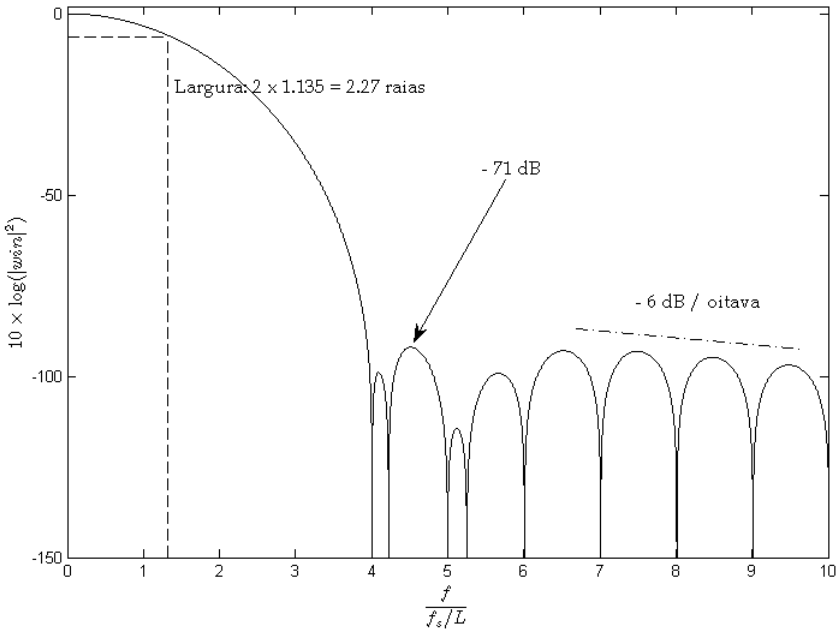

Figura 3.5: Resposta em frequência da janela de Blackman-Harris.

Existem diferentes tipos de janelas para várias aplicações, e estes três parâmetros dão uma idéia geral de aplicação de uma determinada janela. Em aplicações onde há interferências (ruído) em frequências distantes da frequência de interesse, um decaimento rápido é preferido. Em aplicações onde as interferências estão em frequências mais próximas, a altura do primeiro lóbulo lateral é importante. Um lóbulo principal largo melhora a precisão da medida de amplitude, mas, em contrapartida, diminui a resolução de frequência. Se um sinal contém componentes de frequência muito próximas umas das outras, deve ser escolhida uma janela com um lóbulo principal estreito. A Figura 3.5 mostra a janela Blackman-Harris, cuja altura do primeiro lóbulo lateral é bem mais baixa ($-71\,\mathrm{dB}$) do que a da janela de Hanning ($-32\,\mathrm{dB}$). A Figura 3.6 mostra a janela retangular. E por fim, se for necessário uma boa resolução temporal, a janela deve ser estreita no domínio do tempo (em geral isso significa um lóbulo principal mais largo no domínio da frequência), e a Figura 3.7 mostra a resposta no tempo de algumas janelas utilizadas, onde se vê que a janela retangular é a que tem a pior resolução temporal dentre todas, mas em compensação, é a que tem a melhor resolução de frequência (lóbulo lateral mais estreito).

Na Tabela 3.2, observa-se que quanto mais estreito o lóbulo principal, melhor foi o desempenho da janela. Com uma exceção: a janela retangular obteve um desempenho inferior ao da janela de Hanning. Como o salto utilizado foi $J = L$, a resolução temporal foi prejudicada, e talvez por isso o desempenho tenha sido pior. Entretanto, a janela retangular possui uma propriedade interessante: ela atende à

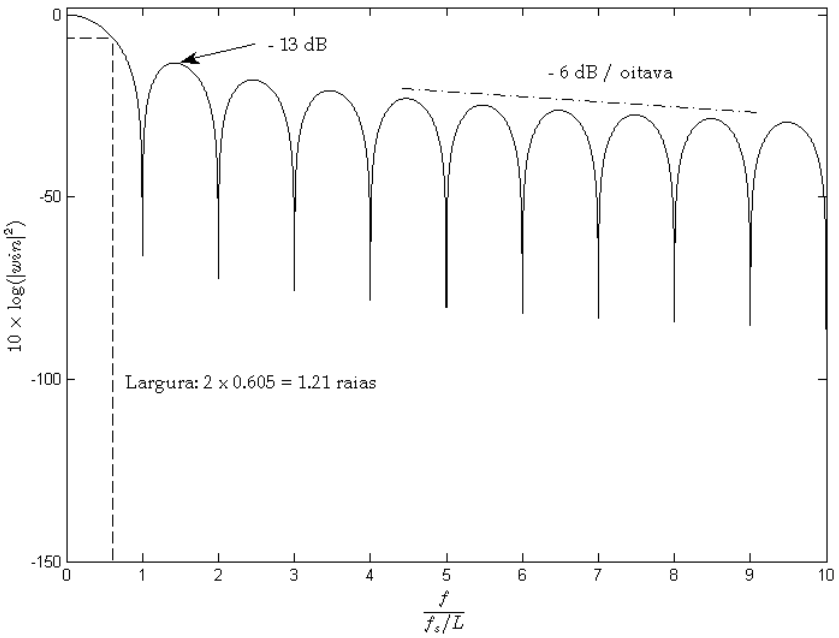

Figura 3.6: Resposta em frequência da janela retangular.

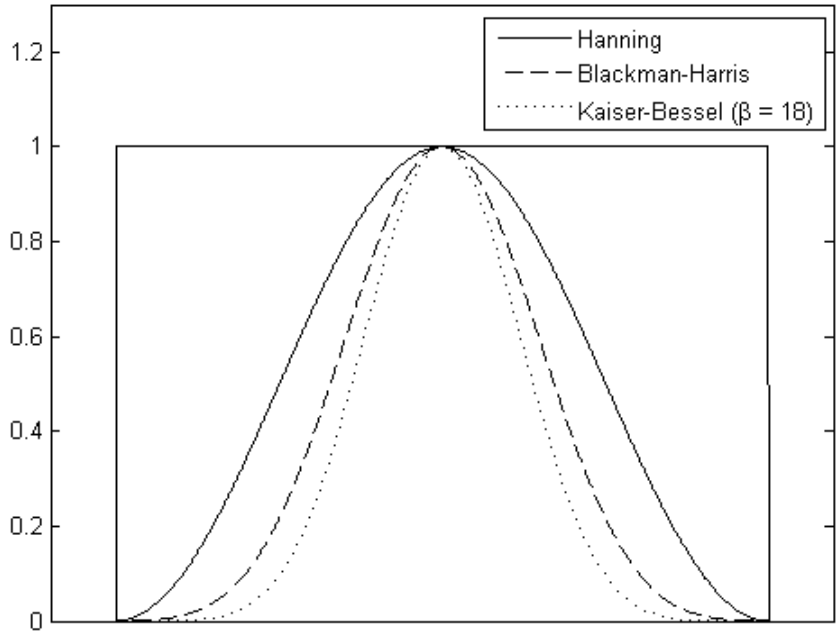

Figura 3.7: Comparação de algumas janelas no tempo. A retangular está representada para comparação, e é a que tem a melhor resolução na frequência e pior resolução temporal. Quanto mais estreita a janela, melhor sua resolução temporal e pior sua resolução na frequência.

COLA para qualquer salto J (com a condição de que $J < L$, obviamente). Com isso em mente, foram realizados testes com três valores de saltos diferentes para verificar se a janela retangular obtém desempenho melhor, e o resultado está na Tabela 3.3. Como era de se esperar, a janela retangular com salto $J = \frac{L}{4}$ obteve o melhor desempenho dentre todas.

Tabela 3.3: Desempenho em BSS utilizando a janela retangular como janela win_a da STFT, para diferentes saltos J.

Número de fontes e misturas - $N = M = 2$			
Tempo de reverberação - $T_{60} = 130\,\text{ms}$			
Número de raias da FFT - $K = 4096$			
Tamanho da janela - $L = 2048$			
Separação - Natural ICA com sign ($\eta = 0,1$)			
Método para resolver a permutação - DOA + HarmCorr			
Número de realizações - 10			
Disposição dos microfones e fontes - Figura A.1			
Janela win_a - Retangular			
Salto J	SIR médio	SDR médio	SAR médio
2048	$17,2\,\text{dB}$	$9,8\,\text{dB}$	$10,9\,\text{dB}$
1024	$18,6\,\text{dB}$	$11,3\,\text{dB}$	$12,9\,\text{dB}$
512	$19,2\,\text{dB}$	$12,3\,\text{dB}$	$14,0\,\text{dB}$

Resta saber se este resultado é válido para qualquer método para resolver a permutação (abordados no Capítulo 4) ou somente para o método utilizado ("DOA + HarmCorr"). Na Seção 4.4 serão realizados mais testes para chegarmos a uma conclusão.

3.3 Branqueamento

Branquear o sinal antes de aplicar o algoritmo de separação traz algumas vantagens à separação, além de ser pré-requisito em alguns casos. De uma forma geral, branquear o vetor de misturas é uma boa idéia, pois faz uma parte significativa do trabalho de separação (a descorrelação). No caso dos algoritmos rápidos que se utilizam de maximização da não-gaussianidade (FastICA), o branqueamento é um pré-processamento obrigatório. Já no caso dos algoritmos que utilizam estimativa da ML (como o Natural ICA), o branqueamento apenas torna a variância das misturas (energia) unitária. Isto torna a convergência do ICA no domínio da frequência rápida e robusta. Sem a normalização do vetor de misturas, a convergência não seria uniforme de frequência para frequência, pois os sinais de áudio em geral são muito coloridos, i.e, a energia varia muito de uma frequência para outra. Isto significa

que, para um passo de adaptação fixo, a convergência seria muito mais rápida em algumas frequências do que em outras.

Resumindo, o branqueamento é necessário porque:

1. Faz aproximadamente *metade* do trabalho de separação, com um menor custo computacional;

2. É pré-requisito dos algoritmos FastICA;

3. Faz com que a convergência seja uniforme em cada raia de frequência, para um passo de adaptação fixo.

O branqueamento é realizado no domínio da frequência, portanto as misturas agora são representadas da seguinte forma: $\mathbf{x}_k(m)$, onde k é o índice da frequência e m é o índice do *frame*. O primeiro passo para branquear o vetor de misturas $\mathbf{x}_k(m)$ é tornar sua média zero. Isso pode ser feito fazendo com que a média de cada uma das misturas $x_{jk}(m)$ seja zero. Ou seja:

$$\begin{bmatrix} x_{1k}(m) \\ x_{2k}(m) \\ \vdots \\ x_{Mk}(m) \end{bmatrix} \leftarrow \begin{bmatrix} x_{1k}(m) \\ x_{2k}(m) \\ \vdots \\ x_{Mk}(m) \end{bmatrix} - \begin{bmatrix} E\{x_{1k}\} \\ E\{x_{2k}\} \\ \vdots \\ E\{x_{Mk}\} \end{bmatrix} \tag{3.12}$$

Após centralizar o vetor de misturas, precisamos tornar sua matriz de covariância a matriz identidade. Como $E\{\mathbf{x}\} = \mathbf{0}$, a matriz de covariância do vetor de misturas pode ser dada por:

$$\mathbf{\Sigma}_{x_k x_k} = E\{\mathbf{x}_k \mathbf{x}_k^H\} \tag{3.13}$$

O branqueamento é feito em cada frequência por uma matriz $\mathbf{V}$, ou seja:

$$\mathbf{z}_k(m) = \mathbf{V}_k \mathbf{x}_k(m) \tag{3.14}$$

de forma que $\mathbf{\Sigma}_{z_k z_k} = \mathbf{I}$. Como $\mathbf{\Sigma}_{z_k z_k} = \mathbf{V}_k \mathbf{\Sigma}_{x_k x_k} \mathbf{V}_k^H$, se decompormos $\mathbf{\Sigma}_{x_k x_k}$ de forma que $\mathbf{\Sigma}_{x_k x_k} = \mathbf{E}\mathbf{D}\mathbf{E}^H$, temos que:

$$\mathbf{\Sigma}_{z_k z_k} = \mathbf{V}_k \mathbf{\Sigma}_{z_k z_k} \mathbf{V}_k^H \Rightarrow \mathbf{V}_k \mathbf{E}\mathbf{D}\mathbf{E}^H \mathbf{V}_k^H = \mathbf{I} \Rightarrow \mathbf{V}_k = \mathbf{D}^{-\frac{1}{2}} \mathbf{E}^H \tag{3.15}$$

A matriz $\mathbf{E}$ é a matriz de autovetores de $\mathbf{R}_{x_k x_k}$, onde cada coluna é um autovetor, e $\mathbf{D}$ é uma matriz diagonal que contém os autovalores de $\mathbf{\Sigma}_{x_k x_k}$. Uma propriedade importante é que se permutarmos $\mathbf{D}$ e $\mathbf{E}$ com a mesma permutação, o produto $\mathbf{D}^{-\frac{1}{2}} \mathbf{E}^H$ se mantém. Seja $\mathbf{P}_k$ uma matriz de permutação qualquer, segundo definido

em (2.18). Temos:

$$\begin{aligned}\mathbf{D}_P^{-\frac{1}{2}}\mathbf{E}_P^H &= (\mathbf{D}\mathbf{P}_k^T)^{-\frac{1}{2}}(\mathbf{E}\mathbf{P}_k^T)^H) \\ &= \mathbf{D}^{-\frac{1}{2}}\mathbf{P}_k^T\mathbf{P}_k\mathbf{E}^H \\ \mathbf{P}_k^T\mathbf{P}_k &= \mathbf{I} \qquad \therefore\end{aligned}$$

$$\mathbf{D}^{-\frac{1}{2}}\mathbf{E}^H = (\mathbf{D}\mathbf{P}_k)^{-\frac{1}{2}}(\mathbf{E}\mathbf{P}_k^T)^H) = \mathbf{D}_P^{-\frac{1}{2}}\mathbf{E}_P^H \tag{3.16}$$

A Equação (3.16) nos diz que podemos alterar a ordem dos autovalores à vontade, contanto que alteremos a ordem dos autovetores correspondentes, e a matriz $\mathbf{V}_k$ continuará sendo uma matriz branqueadora. Isso significa que podemos ordenar os autovalores da matriz $\mathbf{D} = \text{diag}([d_1, d_2, \cdots, d_M])$ de forma que $d_1 > d_2 > \cdots > d_M$, e ordenar os autovetores correspondentes da matriz $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_M]$, de forma que o autovetor $\mathbf{e}_j$ corresponda ao autovalor d_j. Isto é útil para realizar redução dimensional, pois as componentes principais de $\mathbf{x}_k(m)$ corresponderão às primeiras linhas de $\mathbf{z}_k(m)$, então a redução dimensional é realizada mantendo-se as N primeiras linhas de $\mathbf{z}_k(m) = [z_{1k}(m), \cdots, z_{Nk}(m), \cdots, z_{Mk}(m)]$, de acordo com o valor do autovalor d_j correspondente. Este método de redução dimensional é conhecido como Análise de Componente Principais (PCA, do inglês *Principal Component Analysis*).

3.4 Separação

A separação dos sinais é realizada em cada raia de frequência, utilizando um algoritmo ICA que trabalhe com números complexos. O algoritmo Natural ICA é preferido, porque não possui restrições com relação à matriz separadora (ver Seção 2.5.2), embora o FastICA também seja utilizado, por causa de sua simplicidade e baixo custo computacional. Outra vantagem do FastICA é a independência do seu desempenho com relação à distribuição das fontes. Se as distribuições das fontes forem difíceis de estimar corretamente, o FastICA em geral apresenta um desempenho superior ao Natural ICA (ver Seção 2.5.3).

No caso do FastICA, o algoritmo de separação é aplicado nos sinais branqueados $\mathbf{z}_k(m)$, em forma matricial, segundo mostrado em (2.69), e, após a adaptação, é gerada uma matriz separadora unitária $\mathbf{U}_k$. Podemos encontrar então a matriz separadora $\mathbf{W}_k$ da seguinte forma:

$$\mathbf{W}_k = \mathbf{U}_k\mathbf{V}_k \tag{3.17}$$

onde $\mathbf{V}_k$ é a matriz branqueadora de (3.14). Esta matriz $\mathbf{W}_k$ é utilizada para encontrar as fontes estimadas independentes $\mathbf{y}_k(m)$ em cada raia de frequência,

utilizando a Equação (2.7).

No algoritmo Natural ICA, após o branqueamento, a matriz $\mathbf{V}_k$ é utilizada como solução inicial $\mathbf{W}_k = \mathbf{V}_k$ e a matriz $\mathbf{W}_k$ é atualizada iterativamente utilizando o algoritmo (2.80). As funções *score* mostradas na Tabela 2.3 trabalham somente com números reais. Em [44, 45], o autor propõe que a função seja aplicada às partes real e imaginária separadamente:

$$\Phi(y_{ik}) = \Phi(\Re(y_{ik})) + \hat{\jmath}\Phi(\Im(y_{ik})) \tag{3.18}$$

que é conhecido como função *score* de coordenadas cartesianas. A função Φ é derivada considerando-se que os sinais são reais, segundo a Equação (2.81). Em [31, 46], os autores derivam o algoritmo que utiliza ML (o Natural ICA) diretamente no domínio complexo, definindo a derivada no domínio complexo e encontrando o gradiente. Em [46], a derivação matemática é mais simples, mas ambos chegam ao mesmo resultado: utilizar o Natural ICA como definido anteriormente em (2.80) com números complexos diretamente e aplicar a função *score* às partes real e imaginária separadamente.

Se analisarmos o algoritmo de adaptação do Natural ICA na Equação (2.80), perceberemos que ele converge para um ponto que satisfaz a condição $E\{\Phi(y_{ik})y_{ik}^*\} = 1$. Analisando apenas a parte imaginária desta equação (que deve ser nula), temos a condição (3.19), que indica uma restrição adicional: que as partes real e imaginária de y_{ik} devem ser independentes.

$$E\{\Phi(\Im(y_{ik}))\Re(y_{ik}) - \Phi(\Re(y_{ik}))\Im(y_{ik})\} = 0 \tag{3.19}$$

O uso do Natural ICA não-holonômico (Equação (2.83)) alivia essa restrição, mas em [47], o autor propõe outra forma de aplicar a função *score* à fonte y_{ik}. Se a distribuição $q(y_{ik})$ for independente da fase, i.e, se $q(y_{ik}) = \alpha q(|y_{ik}|)$, então a função *score* pode ser aplicada da seguinte forma:

$$\Phi(y_{ik}) = \Phi(|y_{ik}|)e^{\hat{\jmath}\theta(y_{ik})} \tag{3.20}$$

onde a função Φ é dada por:

$$\Phi(y_i) = -\frac{\partial}{\partial|y_i|}\log(q(|y_i|)) \tag{3.21}$$

que é conhecido como função *score* de coordenadas polares. A consideração feita parece bastante natural para sinais de áudio no domínio da frequência, onde a fase depende da posição das janelas e pode ser modificada arbitrariamente. Sendo matematicamente mais rigoroso, a independência da fase significa que a variável complexa

y_{ik} é circular, ou seja, para qualquer valor real α, y_{ik} e $\exp(\hat{\jmath}\alpha)y_{ik}$ têm a mesma distribuição. Foram realizados testes utilizando ambos os métodos, onde a permutação foi resolvida de maneira supervisionada para que não houvesse influência da permutação nos resultados, e o resultado está na Tabela 3.4. Foram testados os algoritmos usual (2.80) e o não-holonômico (2.83). Os nomes das funções *score* foram definidos na Tabela 2.3.

Tabela 3.4: Comparação entre as funções *score* de coordenadas cartesianas e polares, utilizando o Natural ICA usual e o não-holonômico.

Número de fontes e misturas - $N = M = 2$
Tempo de reverberação - $T_{60} = 130\,\text{ms}$
Número de raias da FFT - $K = 2048$
Tamanho da janela - $L = 1024$
Método para resolver a permutação - Supervisionado
Número de realizações - 10
Disposição dos microfones e fontes - Figura A.1
Janela win_a - Hanning

Natural ICA	função *score*	Coordenadas	SIR médio	SDR médio	SAR médio
Usual	sign	Cartesianas	$24,1\,\text{dB}$	$20,4\,\text{dB}$	$22,9\,\text{dB}$
Usual	sign	Polares	$23,4\,\text{dB}$	$20,1\,\text{dB}$	$22,8\,\text{dB}$
Usual	tanh	Cartesianas	$21,9\,\text{dB}$	$18,6\,\text{dB}$	$21,4\,\text{dB}$
Usual	tanh	Polares	$22,3\,\text{dB}$	$18,9\,\text{dB}$	$21,8\,\text{dB}$
Usual	genLaplace ($\alpha = 0,1$)	Cartesianas	$22,8\,\text{dB}$	$19,5\,\text{dB}$	$22,3\,\text{dB}$
Usual	genLaplace ($\alpha = 0,1$)	Polares	$22,9\,\text{dB}$	$19,7\,\text{dB}$	$22,6\,\text{dB}$
Não-holonômico	sign	Cartesianas	$24,8\,\text{dB}$	$20,9\,\text{dB}$	$23,2\,\text{dB}$
Não-holonômico	sign	Polares	$24,5\,\text{dB}$	$20,8\,\text{dB}$	$23,2\,\text{dB}$
Não-holonômico	tanh	Cartesianas	$21,5\,\text{dB}$	$18,2\,\text{dB}$	$20,9\,\text{dB}$
Não-holonômico	tanh	Polares	$21,9\,\text{dB}$	$18,5\,\text{dB}$	$21,3\,\text{dB}$
Não-holonômico	genLaplace ($\alpha = 0,1$)	Cartesianas	$22,7\,\text{dB}$	$19,2\,\text{dB}$	$21,9\,\text{dB}$
Não-holonômico	genLaplace ($\alpha = 0,1$)	Polares	$23,1\,\text{dB}$	$19,5\,\text{dB}$	$22,0\,\text{dB}$

O que primeiro se percebe nesta tabela é que a escolha entre coordenadas polares ou cartesianas depende da função *score* utilizada. Por exemplo, as coordenadas polares obtiveram melhores resultados para as funções tanh e genLaplace, mas obtiveram resultados piores para a função sign, porém, a diferença não passa de 1 dB em nenhum dos casos. Portanto, para tomar uma decisão, é importante comparar as abordagens de outra maneira, como por exemplo, a convergência. A Tabela 3.5 mostra o número de iterações necessário para convergência, com as mesmas condições da tabela anterior, e utilizando o Natural ICA usual. Para detalhes sobre como descobrir o número de iterações para convergência, ver Apêndice B.

Claramente, utilizar coordenadas polares nos dá uma convergência mais rápida, o que acaba sendo preferido. Com relação ao custo computacional, tudo vai depender

Tabela 3.5: Comparação entre as funções *score* de coordenadas cartesianas e polares em número de iterações para convergir em cada raia de frequência.

Função *score*	Coordenadas	Iterações médias por frequência
sign	Cartesianas	113
sign	Polares	77
tanh	Cartesianas	156
tanh	Polares	70
genLaplace	Cartesianas	129
genLaplace	Polares	76

do custo da função *score* contra o custo do operador $|.|$ acrescido de duas multiplicações. Por exemplo, a função sign tem menor custo computacional no modelo cartesiano e a função tanh apresenta menor complexidade quando usado o modelo polar.

Outro motivo para utilizar as coordenadas polares é sua convergência mais regular. As Figuras 3.8 e 3.9 mostram a convergência típica do Natural ICA em um caso de duas fontes e duas misturas. A convergência é mostrada apenas para uma raia de frequência, escolhida aleatoriamente (o resultado é o mesmo para todas as raias de frequência), e a função *score* utilizada foi a sign. Apenas o valor do módulo é mostrado, pois, no caso cartesiano, tanto a parte real como a imaginária gerariam um gráfico similar, e no caso polar, a parte imaginária é irrelevante. Percebe-se a convergência mais suave do modelo polar, o que favorece sua utilização. Por estes motivos, utilizaremos sempre o modelo polar com o Natural ICA.

A função *score* pode também ser diferente em cada raia de frequência. A função genGaussian da Tabela 2.3, baseada na gaussiana generalizada, possui um parâmetro r ajustável, que modifica a função, e é repetida aqui, com variância normalizada para 1:

$$q(y) = \frac{r}{2\Gamma\left(\frac{1}{r}\right)} \exp\left(-\frac{1}{r}|y|^r\right) \tag{3.22}$$

onde $\Gamma(.)$ é a função Gamma, que é definida como:

$$\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx \tag{3.23}$$

Como exemplos, quando $z = 1$, $\Gamma(z) = 1$, e quando $z = 4$, $\Gamma(z) = 6$. A função *score* correspondente, obtida através de $-\frac{\partial}{\partial y_i} \log(q(y_i))$, é:

$$\Phi(y) = |y|^{r-1} \operatorname{sign}(y) \tag{3.24}$$

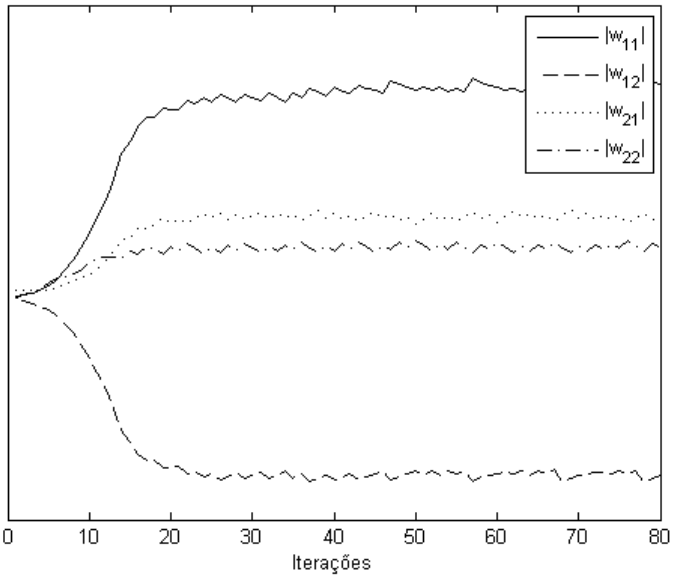

Figura 3.8: Convergência típica do Natural ICA utilizando funções *score* calculadas através do modelo *cartesiano*.

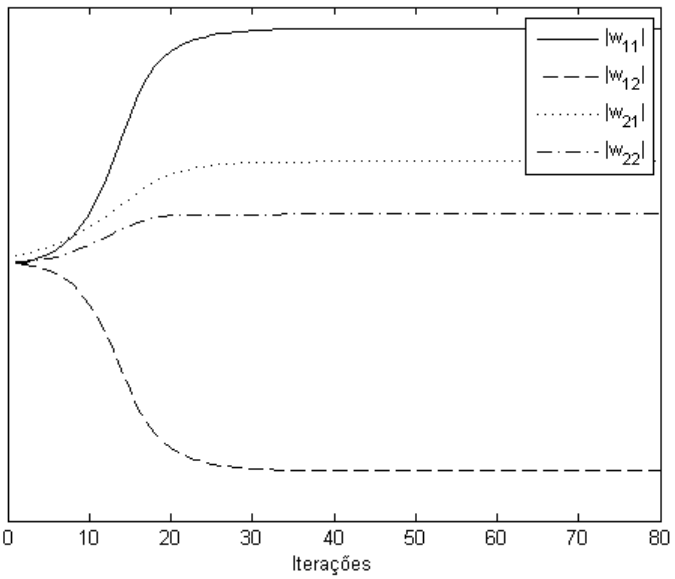

Figura 3.9: Convergência típica do Natural ICA utilizando funções *score* calculadas através do modelo *polar*.

No caso especial em que $r = 1$, a distribuição (3.22) se reduz à laplaciana, e a função (3.24), à função sign. Da mesma forma, quando $r = 1$, a distribuição se torna a cúbica, que é eficaz em BSS quando os sinais são subgaussianos [33]. As Figuras 3.10, 3.11 e 3.12 mostram a distribuição gaussiana generalizada de números complexos para três valores de r diferentes, onde a variância foi normalizada para 1 ($\sigma = 1$).

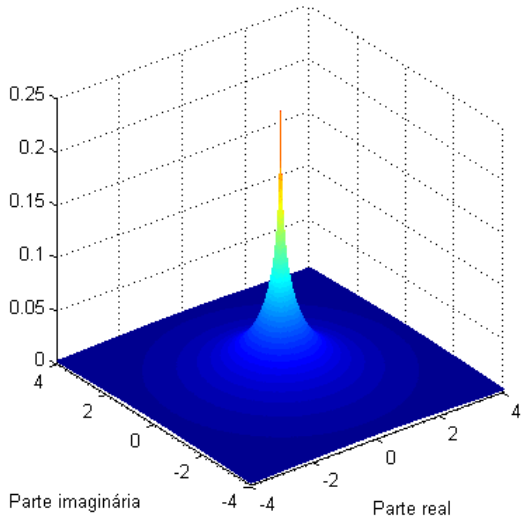

Figura 3.10: Gaussiana generalizada complexa para $r = 0.5$.

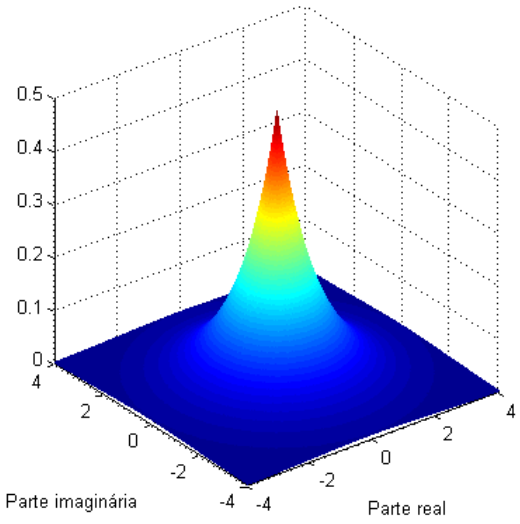

Figura 3.11: Gaussiana generalizada complexa para $r = 1$.

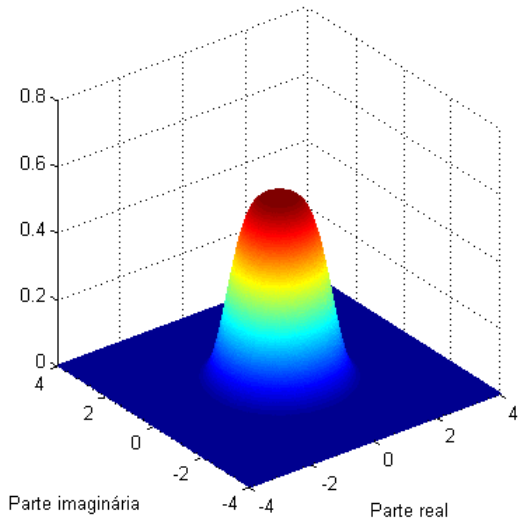

Figura 3.12: Gaussiana generalizada complexa para $r = 4$.

A idéia é utilizar funções diferentes dependendo da distribuição estimada das fontes ou pelo menos, diferenciar as funções *score* quando a distribuição é subgaussiana

ou supergaussiana. Podemos medir a gaussianidade de uma distribuição, como visto na Seção 2.5.2, através de sua curtose, calculada de (2.32). Para encontrar a curtose da distribuição gaussiana generalizada, primeiro definamos a seguinte fórmula integral, derivada em [48]:

$$\int_0^\infty y^{v-1} e^{-\sigma y^a} dy = \frac{1}{a}\sigma - \frac{1}{v}\Gamma\left(\frac{v}{a}\right) \tag{3.25}$$

Considerando sinais de média zero, a variância (2.26) da distribuição gaussiana generalizada é (considerando a definição de valor esperado [27]):

$$\begin{aligned} \sigma_y^2 &= \int_{-\infty}^{\infty} |y|^2 q(y,r) dy \\ &= 2\int_{-\infty}^{\infty} |y|^2 \frac{r}{2\Gamma\left(\frac{1}{r}\right)} \exp(-|y|^r) dy \end{aligned} \tag{3.26}$$

Como o valor de y em (3.26) só pode ser positivo, retiramos os módulos das integrais, para facilitar:

$$\sigma_y^2 = 2\frac{r}{2\Gamma(\frac{1}{r})}\int_0^\infty y^2 \exp(-y^r) dy \tag{3.27}$$

Utilizando a fórmula (3.25),com $v = 3$, $\sigma = 1$ e $a = r$:

$$\sigma_y^2 = \frac{\Gamma\left(\frac{3}{r}\right)}{\Gamma\left(\frac{1}{r}\right)} \tag{3.28}$$

Utilizando a mesma derivação, chegamos ao quarto momento central $E\{(y-\mu_y)^4\}$ para o caso de $\mu_y = 0$:

$$E\{(y-\mu_y)^4\} = \frac{\Gamma\left(\frac{5}{r}\right)}{\Gamma\left(\frac{1}{r}\right)} \tag{3.29}$$

De (3.28) e (3.29), podemos chegar à curtose (2.32):

$$\text{curt}(y) = \frac{\Gamma\left(\frac{5}{r}\right)\Gamma\left(\frac{1}{r}\right)}{\Gamma^2\left(\frac{3}{r}\right)} \tag{3.30}$$

A curtose da distribuição gaussiana generalizada em função do parâmetro r pode ser vista nas Figuras 3.13 e 3.14.

Decidimos escolher 3 valores de r diferentes, dependendo da curtose da distribuição estimada das fontes. Para distribuições subgaussianas, é suficiente escolher um valor só. Se $\text{curt}(y) < 3$ (distribuição subgaussiana), escolhemos $r = 4$. Se $\text{curt}(y) < 10$, fizemos $r = 1$ (Laplace), e, finalmente, se a distribuição tiver uma

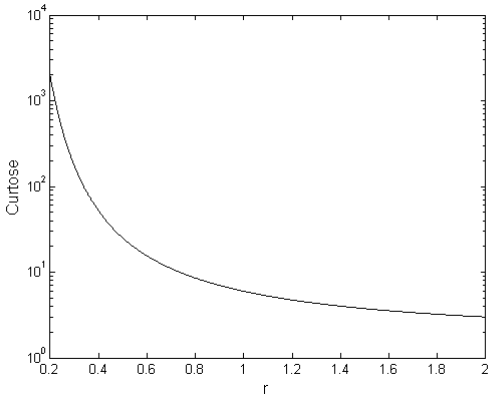

Figura 3.13: Curtose da distribuição gaussiana generalizada em função de r, para distribuições supergaussianas.

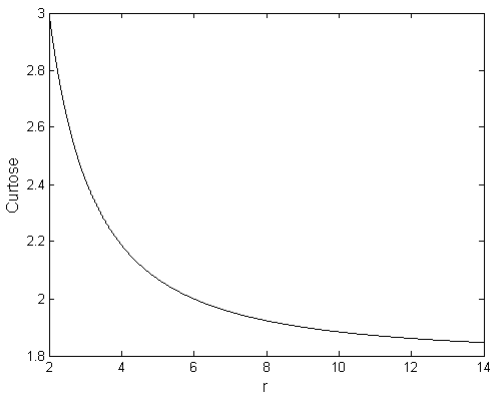

Figura 3.14: Curtose da distribuição gaussiana generalizada em função de r, para distribuições subgaussianas.

curtose muito alta, $\text{curt}(y) > 10$, escolhemos $r = 0.5$. Como não temos os valores da curtose, decidimos utilizar os valores das misturas branqueadas $\mathbf{z}(m)$, que obviamente tem uma curtose mais próxima da gaussiana do que as fontes separadas $\mathbf{y}(m)$, e isso limita o desempenho do algoritmo. Pré-separar as fontes com outro método de separação para obter os valores "reais" das curtoses e depois aplicar o algoritmo acima não mostrou muita melhora, até por causa do erro permitido na estimativa da distribuição das fontes, segundo apontado na Seção 2.5.3. Portanto, decidimos utilizar a curtose dos sinais branqueados.

Também se pode aliar a velocidade do FastICA com a precisão do Natural ICA. Em [4], o autor utiliza o FastICA com uma distribuição de Laplace generalizada e depois utiliza o Natural ICA. A distribuição de Laplace generalizada é a segunda da Tabela 2.3, e é repetida aqui:

$$q(y_i) = \frac{1}{B} \exp\left(-\frac{\sqrt{|y_i|^2 + \alpha}}{\sigma^2}\right) \tag{3.31}$$

onde σ^2 é a variância da distribuição, e o parâmetro B existe somente para que a integral da distribuição seja a unidade, segundo a definição de densidades de probabilidade. Essa normalização é irrelevante no ICA. A variância também não fará muita diferença, pois apenas afeta a escala da fonte estimada y_i, que será ajustada no estágio de escalamento. Dessa forma, fazemos $B = 1$ e $\sigma^2 = 1$. A função $G(y_i)$ é encontrada fazendo-se $G(y_i) = -\log(q(y_i))$, segundo apontado na Seção 2.5.2, e é a primeira da Tabela 2.2, juntamente com sua derivada $g(y_i)$.

Primeiramente são branqueados os sinais, obtendo-se uma matriz $\mathbf{V}_k$, segundo (3.14). Depois aplica-se o FastICA ao sinal branqueado, como explicado anteriormente, e obtém-se a matriz separadora unitária $\mathbf{U}_k$. Através da Equação (3.17),

é obtida a matriz separadora $\mathbf{W}_k$, que será utilizada como solução inicial do algoritmo Natural ICA, que não tem a restrição de que a matriz separadora seja unitária, melhorando o desempenho.

A Tabela 3.6 mostra uma sequência de testes realizados com várias abordagens diferentes, e comparando os valores de SIR, SDR, SAR e iterações necessárias para convergência. Foi utilizada uma sala segundo a Figura A.2. Maiores detalhes podem ser encontrados no Apêndice A. O método Natural ICA utilizado é o usual, para várias funções *score* diferentes. O FastICA + Natural ICA é a abordagem que une os dois métodos, e utiliza a função genLaplace no FastICA com $\alpha = 0,1$ e funções variadas no Natural ICA. O método Natural ICA adaptável é o que utiliza genGaussian, o qual apresentamos anteriormente.

Tabela 3.6: Comparação entre várias abordagens de separação, tanto Natural ICA como o método conjugando FastICA e Natural ICA.

Número de fontes e misturas - $N = M = 3$
Tempo de reverberação - $T_{60} = 130\,\text{ms}$
Número de raias da FFT - $K = 2048$
Tamanho da janela - $L = 1024$
Método para resolver a permutação - Supervisionado
Número de realizações - 10
Disposição dos microfones e fontes - Figura A.3
Janela win_a - Hanning

Método	Função *score*	SIR médio	SDR médio	SAR médio
Natural ICA	sign	$22,9\,\text{dB}$	$17,3\,\text{dB}$	$21,0\,\text{dB}$
Natural ICA	tanh	$21,3\,\text{dB}$	$15,8\,\text{dB}$	$20,1\,\text{dB}$
Natural ICA	genLaplace	$22,3\,\text{dB}$	$16,9\,\text{dB}$	$20,9\,\text{dB}$
FastICA + Natural ICA	sign	$21,9\,\text{dB}$	$16,5\,\text{dB}$	$20,7\,\text{dB}$
FastICA + Natural ICA	tanh	$21,3\,\text{dB}$	$15,8\,\text{dB}$	$20,1\,\text{dB}$
FastICA + Natural ICA	genLaplace	$21,7\,\text{dB}$	$16,3\,\text{dB}$	$20,6\,\text{dB}$
Natural ICA adaptável	genGaussian	$23,5\,\text{dB}$	$17,9\,\text{dB}$	$21,8\,\text{dB}$

Como pode ser visto na tabela, a função *genGaussian* apresentou resultados marginalmente superiores às outras, porém a complexidade computacional extra provavelmente não vale o ganho em desempenho. Se mesmo com a complexidade extra, o algoritmo conseguir convergir em tempo real, então vale a pena utilizá-lo em detrimento dos outros. A conjugação do FastICA com o Natural ICA obteve resultados inferiores ao Natural ICA em todos os casos, porém, o algoritmo conjugado converge muito mais rápido, o que é um ponto a favor dele. A escolha do algoritmo de separação depende, então, da aplicação específica e do *hardware* disponível. Se não houver muito poder computacional disponível, a melhor opção é o ICA conju-

gado utilizando sign. Se ele for muito grande, o Natural ICA adaptável é preferido. Um meio termo seria utilizar o Natural ICA com a função sign.

Nota-se na tabela que a função sign obteve melhores resultados do que as outras. Isso indica que ela é uma boa estimativa média de um sinal de voz. Realmente, o parâmetro r no Natural ICA adaptável foi 1 para quase todas as frequências, o que indica a utilização da função sign.

3.4.1 Outros Algoritmos de Separação

Além do ICA, existem outros algoritmos de separação que utilizam outras abordagens, além da não-gaussianidade (o caso do ICA), para definir a independência das fontes. De uma forma geral, em [49], o autor separa as abordagens encontradas na literatura para separação de fontes independentes em três categorias:

1. Espectro colorido. Supõe que as fontes não são sinais brancos, e tenta diagonalizar a matriz de correlação das saídas, com diferentes atrasos de tempo (*lags*);

2. Não-estacionaridade. Supõe que as fontes são sinais quase-estacionários, e tenta diagonalizar a matriz de correlação das saídas em diferentes janelas de tempo;

3. Não-gaussianidade. Supõe que os sinais são não-gaussianos, e utiliza estatísticas de maior ordem (ICA).

Percebe-se que as duas primeiras abordagens apenas utilizam informações estatísticas até segunda ordem. A primeira abordagem, proposta por [50, 51], consiste primeiro em diagonalizar matrizes de correlação $\hat{\mathbf{R}}_{y'_i y'_i}$ $N \times N$, calculadas segundo (2.41), com $\mathbf{y}'_i(n) = [y_i(n), y_i(n - lag_1), \cdots, y_i(n - lag_D)]$, onde $lag_d, d = 1, \cdots, D$, são os D atrasos considerados. Se as fontes não forem sinais brancos, então, a diagonal principal de $\hat{\mathbf{R}}_{y'_i y'_i}$ deve conter apenas 1s e todos os outros elementos devem ser zero, pois $\rho_{y(n)y(n-lag)} = 0$.

A segunda abordagem é utilizada em [52–55]. Para trechos de sinais de voz menores do que 10 ms, os sinais são considerados estacionários, mas para trechos maiores (perto de 100 ms), os sinais são quase-estacionários, e a suposição de não-estacionaridade vale. As matrizes de correlação $\hat{\mathbf{R}}_{yy}$ a serem diagonalizadas são calculadas segundo (2.41), mas considerando apenas N_{bloco} amostras para computar as correlações amostrais ρ_{yy}. Existem, então, $\frac{N_{\text{amost}}}{N_{\text{bloco}}}$ matrizes a serem diagonalizadas, uma para cada bloco. Diagonalizar cada matriz (zerar os elementos fora da diagonal principal) é similar a realizar um branqueamento em cada bloco (ver a Equação (2.42)), porém a diferença é que, neste caso, deve ser encontrada uma *mesma* matriz

$\mathbf{W}$ tal que todas as $\frac{N_{\text{amost}}}{N_{\text{bloco}}}$ matrizes estejam diagonalizadas, portanto não pode se utilizar a mesma abordagem que utilizamos na Seção 3.3. Esta abordagem também só utiliza informações de segunda ordem.

A limitação de que utilizar estatísticas de maior ordem torna o algoritmo ICA muito sensível a *outliers*, dita em [53], não acontece com os algoritmos ICA que utilizam negentropia como medida de não-gaussianidade, segundo dito na Seção 2.5.2, que possui maior robustez, nem nos algoritmos ICA baseados em ML, que utilizam uma suposição a mais: que a distribuição das fontes é conhecida. A maior vantagem dos algoritmos que só utilizam informações estatísticas de segunda ordem é o menor custo computacional.

Em [56], o autor utiliza uma abordagem que une as três suposições acima (espectro colorido, não-estacionaridade e não-gaussianidade), num algoritmo que ele chama de TRINICON [57] (TRiple INdependent component analysis for CONvolutive mixtures). Este algoritmo funciona no domínio do tempo, e portanto, não sofre do problema da permutação. Uma transformação das equações do TRINICON para o domínio da frequência de uma forma rigorosa é mostrada em [49], que teoricamente não sofre com os problemas da permutação, pois leva em consideração informações sobre todas as frequências de uma vez só (é um algoritmo de banda larga). A desvantagem é o maior custo computacional, que é limitante em uma aplicação em tempo real. Com o poder de processamento dos computadores aumentando consideravelmente, num futuro próximo talvez essa abordagem possa ser utilizada em tempo real, com a vantagem de evitar o problema da permutação.

3.5 Permutação

O estágio da permutação é o mais crítico em qualquer algoritmo FDBSS. Ele consiste em encontrar a matriz de permutação $\mathbf{P}_k$, segundo definida na Seção 2.3, onde o índice k representa a raia de frequência, de forma que o as fontes fiquem ordenadas em cada frequência, e a ISTFT gere resultados consistentes (ou seja, que todas as raias de frequência pertençam à mesma fonte). O Capítulo 4 é dedicado a este assunto.

Este problema pode ser abordado de diferentes formas, que podem ser classificadas em três abordagens. A primeira consiste em utilizar informações sobre o sistema de mistura, testando a consistência dos coeficientes dos filtros separadores, ou seja, a matriz separadora $\mathbf{W}_k, k = 0, \cdots, K-1$; a segunda abordagem utiliza informações sobre o espectro na frequência dos sinais recuperados, i.e, as fontes estimadas $y_{ik}(m)$; e a terceira utiliza informações do domínio tempo-frequência, por exemplo, empregando um ICA multidimensional, que atue em todas as frequências e *frames* ao mesmo tempo. Outra opção é unir duas ou mais destas abordagens em uma abor-

dagem conjugada, que alie as vantagens de todas elas. Todas as abordagens tentam obter informações que classifiquem as fontes para que elas possam ser identificadas em cada frequência, ou seja, se tratam de algoritmos de reconhecimento de padrões não-supervisionados, o qual é um problema bem difícil.

Com relação à primeira abordagem (utilizar a matriz separadora), uma primeira proposta é tornar os filtros $\mathbf{W}$ suaves no domínio da frequência durante a adaptação do algoritmo. Em [44], o autor propõe utilizar um *fator de influência* durante a adaptação, de forma que o $\Delta\mathbf{W}_k$ (a atualização da matriz separadora da frequência k) seja uma combinação linear entre $\Delta\mathbf{W}_k$ e $\Delta\mathbf{W}_{k+1}$ (a atualização da frequência adjacente). O problema é que isso modifica a matriz separadora, e acaba degradando o desempenho do algoritmo ICA em cada raia de frequência. Em [54], o autor utiliza a suposição de que $K \gg Q$, i.e, o número de raias utilizado na FFT é muito maior que o comprimento Q do filtro separador no domínio do tempo. Por exemplo, se $K = 10Q$, há 10 raias de frequência para cada coeficiente $w_{ij}(l)$ do filtro no domínio do tempo. Utilizando essa suposição, ele aplica uma projeção sobre os filtros da matriz separadora a cada iteração. Obviamente, essa projeção é aplicada a todas as frequências de uma só vez, e torna os filtros $w_{ij}(k)$ suaves no domínio da frequência. Mais uma vez, este procedimento altera o resultado da separação, e muitas vezes a suposição $K \gg Q$ não é verdadeira, causando degradação no desempenho da separação (que no caso de [54], não é feita através de ICA, mas sim utilizando informações estatísticas de segunda ordem).

Outra proposta relacionada com a matriz separadora é encontrar o DOA (direção de chegada), que consiste no ângulo de chegada das fontes. Por limitações do DOA, entretanto, é possível que duas fontes tenham o mesmo DOA. Isso acontece porque a estimativa de DOA utilizada funciona num plano 2D, e fontes cujos ângulos de chegada tenham o mesmo valor de cosseno estão, para o algoritmo, na mesma posição, o que nos deixa com uma limitação de ângulos entre 0 e 180°. Para resolver este problema, pode-se tentar estimar a posição num espaço tridimensional das fontes (DOA 3D), mas é um problema computacionalmente mais difícil.

Uma proposta mais robusta que o DOA é tentar encontrar a diferença entre tempos de chegada (TDOA) de cada fonte a dois sensores. Ela não sofre da limitação de ângulo que o DOA sofre, e não é computacionalmente muito intensiva. Através do TDOA de cada fonte entre todos os pares de sensores, pode-se identificá-las em cada raia de frequência. Essa proposta é abordada em [4].

Com relação à segunda abordagem (utilizar as fontes estimadas), uma proposta é utilizar a AM (modulação de amplitude) dos sinais das fontes, proposto primeiramente por [58]. A idéia é calcular a correlação entre duas raias de frequência da mesma fonte. Um coeficiente de valor alto indica que as fontes não são descorrelacionadas, ou seja, não são independentes. O autor também utiliza este coeficiente

para realizar a separação, ou seja, o problema da permutação e da separação são resolvidos simultaneamente. Em [59], o autor utiliza outro método para separação e resolve o problema da permutação separadamente, utilizando a correlação entre frequências. Ele calcula a correlação para todas as permutações possíveis, entre frequências adjacentes. A permutação que obtiver o maior valor de correlação indica a permutação correta.

A terceira abordagem consiste em modificar o estágio de separação, e, neste caso, o estágio de permutação deixa de existir, pois resolver o problema da permutação já está implícito no algoritmo de separação. Em geral, abordagens desse tipo consistem em derivar uma função custo que leve em consideração todas as raias de frequência ao mesmo tempo, o que aumenta o tempo computacional, mas ainda o mantém menor do que o tempo computacional de um algoritmo ICA no domínio do tempo, por exemplo. Essa abordagem pode ser encontrada em [60–62], e está fora do escopo desta dissertação.

Mesmo que consigamos resolver com sucesso o problema da permutação em cada raia de frequência, de forma que o sinal $y_{ik}(m)$ esteja na mesma posição em todas as raias de frequência k, ainda teremos o problema da *permutação global*. Isto significa que descobrimos as fontes independentes e podemos até descobrir a posição delas (ver Seção 4.1), mas não saberemos qual fonte pertence a qual posição originalmente. Entretanto, para descobrir a posição de cada fonte, podemos tentar utilizar a esparsidade destas, da seguinte forma: supondo que as fontes são vozes (conforme o foco desta dissertação), no momento que apenas uma das pessoas estiver falando, o algoritmo BSS adapta e tenta descobrir a posição de todas as fontes, que no caso é uma só, e assim, não há problema de ambiguidade. Quando rodarmos o algoritmo novamente com as duas fontes, a posição de uma delas já é conhecida. Se as pessoas estiverem se movimentando, entretanto, a situação é mais complicada. Descobrir posição das fontes foge ao escopo deste trabalho, que se preocupa somente em separar as fontes. A posição só nos é interessante a partir do ponto em que ajuda na separação, como veremos na Seção 4.1.

3.6 Escalamento

O estágio do escalamento tenta encontrar a matriz diagonal $\mathbf{\Lambda}_k$ em (2.20), em cada raia de frequência k, de forma que o sinal de áudio da fonte estimada fique consistente no domínio do tempo. Para isso é utilizado o MDP (Princípio da Mínima Distorção) [63], mostrado em (3.32), onde $\mathbf{T}$ é uma matriz escolhida arbitrariamente que indica a distorção aceitável, e $\mathbf{H}_k$ é a resposta de frequência da matriz de mistura

(definida em (2.12)) na raia de frequência k.

$$\mathbf{\Lambda}_k = \text{diag}(\mathbf{T}\mathbf{H}_k) \tag{3.32}$$

Normalmente, é utilizada $\mathbf{T} = \mathbf{I}$, mas $\mathbf{H}_k$ não é conhecida. Assumindo que o ICA foi bem sucedido e a inversa da matriz separadora corresponde à matriz de mistura a menos de um escalamento, i.e, $\mathbf{W}_{Pk}^{-1}\mathbf{D}_k = \mathbf{H}_k$, onde $\mathbf{D}_k$ é uma matriz diagonal, e o índice $_P$ simboliza que o problema da permutação foi resolvido, podemos aproximar $\mathbf{H}_k$ por $\mathbf{W}_{Pk}^{-1}$, e a Equação (3.32) é modificada para que ela possa ser calculada na prática:

$$\mathbf{\Lambda}_k = \text{diag}(\mathbf{W}_{Pk}^{-1}) \tag{3.33}$$

Substituindo a Equação (3.33) nos modelos instântaneos (2.2) e (2.7) aplicados a uma raia de frequência do caso convolutivo, e lembrando que $\mathbf{W}_{Pk}^{-1}\mathbf{D}_k = \mathbf{H}_k$ e $\mathbf{W}_{Pk} = \mathbf{P}_k\mathbf{W}_k$, temos:

$$\begin{aligned}
\mathbf{y}_k(m) &= \mathbf{\Lambda}_k\mathbf{P}_k\mathbf{W}_k\mathbf{H}_k\mathbf{s}_k(m) \\
&= \text{diag}(\mathbf{W}_{Pk}^{-1})\mathbf{W}_{Pk}\mathbf{W}_{Pk}^{-1}\mathbf{D}_k\mathbf{s}_k(m) \\
&= \text{diag}(\mathbf{W}_{Pk}^{-1}\mathbf{D}_k)\mathbf{s}_k(m) \\
&= \text{diag}(\mathbf{H}_k)\mathbf{s}_k(m)
\end{aligned} \tag{3.34}$$

Ou seja, em cada raia de frequência, $y_{ik} = h_{ii}s_{ik}$. Isto significa que o sinal da fonte estimada no tempo $y_i(n)$ é equivalente à fonte $s_i(n)$ vista pelo sensor i, uma versão filtrada da fonte real. Esta é uma suposição razoável, e não influencia na separação entre as fontes.

Também poderíamos escolher outro valor para a matriz $\mathbf{T}$ em (3.32), como por exemplo, $\mathbf{T} = \mathbf{1}$, onde $\mathbf{1}$ é uma matriz onde todos os elementos são 1. Seguindo os passos anteriores para descobrir a distorção aceita nas fontes estimadas para esta matriz $\mathbf{T}$, chegamos ao resultado $y_{ik} = \sum_{j=1}^{M} h_{ji}s_{ik}$. Nesse caso, no domínio do tempo, a fonte estimada é a soma das contribuições da fonte $s_i(n)$ em cada um dos sensores j.

Por mais que se modifique $\mathbf{T}$, pelo MDP só é possível recuperar versões filtradas das fontes. Outro ponto importante é que embora possamos resolver o problema do escalamento em cada raia de frequência, para que a fonte estimada $y_i(m)$ fique com o mesmo escalamento em todas as raias de frequência, não podemos resolver o *escalamento global*, i.e, as fontes $y_i(n)$ no domínio do tempo ainda terão escalamentos arbitrários. Por exemplo, suponhamos que a fonte s_1 esteja longe dos sensores, e, portanto, com um volume baixo em relação a s_2, que está muito próxima destes. Após o BSS, a saída y_1 (supondo que ela corresponde a s_1, ver problema de permutação global na Seção 3.5) pode estar com um volume muito mais alto que y_2 (que

corresponde a s_2), por causa do problema do escalamento global.

3.7 Suavização

A última etapa consiste na suavização (ou não) do sinal. A suavização tenta resolver o problema da circularidade descrito no início do capítulo. Uma forma de mitigar este problema é fazer com que $K > L$, segundo visto na Seção 3.2. O problema é que não conhecemos o tamanho real P dos filtros da matriz de mistura. Para que não haja distorção nenhuma, como o trecho da mistura tem L amostras, o filtro deveria ter tamanho $P = K - L - 1$ amostras. Em geral, isso não é verdade, e os filtros têm comprimento infinito. Isso gera distorção no domínio do tempo, e, para mitigar este efeito, [64] propõe que o filtro seja janelado no tempo, de forma que seus coeficientes próximos das bordas sejam pequenos. O objetivo é controlar a resposta de frequência destes filtros, de forma que ele tenha tamanho finito no domínio do tempo e valores pequenos nas bordas, aplicando a janela no domínio da frequência.

O autor escolhe a janela de Hanning no domínio do tempo. Lembrando que uma multiplicação no domínio do tempo equivale a uma convolução no domínio da frequência, precisamos descobrir a resposta de frequência da janela de Hanning e filtrar o trecho do sinal utilizando esta resposta de frequência. Ora, a resposta de frequência da janela de Hanning possui somente três coeficientes (os outros são teoricamente nulos, e na prática, muito próximos de zero), e é dada por

$$w_{hanning}(k) = 0,25\delta(k-1) + 0,5\delta(k) + 0,25\delta(k+1) \tag{3.35}$$

no domínio da frequência discreta, onde $\delta(.)$ é a função delta de Dirac. Observe que este filtro não é causal, o que não é problema para nós, pois temos todas as amostras de frequência ao mesmo tempo. Podemos representar (3.35) em forma vetorial como $[0.25, 0.5, 0.25]$, e utilizando este filtro, recalculamos cada elemento da matriz separadora como:

$$W_{ij}(k) \leftarrow 0,25W_{ij}(k-1) + 0,5W_{ij}(k) + 0,25W_{ij}(k+1) \tag{3.36}$$

ou, alternativamente:

$$W_{ij}(k) \leftarrow W_{ij}(k) * w_{hanning}(k) \tag{3.37}$$

onde $*$ denota convolução. Perceba que este método tem um problema: ele altera o valor da solução ICA, e isso pode piorar o desempenho ainda mais. Se a circularidade tiver uma influência muito grande na solução, então talvez valha a pena perder de

um lado (a matriz separadora será diferente da solução ótima do ICA) para ganhar de outro (mitigar o efeito da circularidade). Embora o autor utilize a janela de Hanning, nada impede que outras janelas sejam utilizadas. A Tabela 3.7 mostra os coeficientes da resposta de frequência de algumas janelas, na forma vetorial. As respostas mostradas não são analíticas, pois algumas das janelas possuem respostas em frequência analíticas muito complicadas, então truncamos estas respostas no domínio da frequência. Os coeficientes estão centralizados, i.e, o coeficente 0 é sempre o do meio, pois nenhuma das respostas de frequência é causal.

Tabela 3.7: Coeficientes da resposta de frequência truncada de algumas janelas. O coeficiente 0 é sempre o coeficiente do meio.

Janela	Coeficientes
Hanning	$[0,25 \quad 0,5 \quad 0,25]$
Chebyshev	$[0,003 \quad 0,0602 \quad 0,2516 \quad 0,3902 \quad 0,2516 \quad 0,0602 \quad 0,003]$
Blackman	$[0,01 \quad 0,0817 \quad 0,24 \quad 0,3363 \quad 0,24 \quad 0,0817 \quad 0,01]$
Nuttall	$[0,0092 \quad 0,0795 \quad 0,2407 \quad 0,3409 \quad 0,2407 \quad 0,0795 \quad 0,0092]$
Kaiser	$[0,0014 \quad 0,0032 \quad 0,0129 \quad 0,9787 \quad 0,0129 \quad 0,0032 \quad 0,0014]$

A Tabela 3.8 mostra os resultados de utilizar suavização após o algoritmo convergir. Há um ganho de desempenho marginal utilizando-se suavização, o que mostra que a circularidade não é o problema maior que enfrentamos, sendo este a permutação. O maior ganho está na SAR, porque os artefatos decorrentes do efeito da circularidade diminuem, mas como a suavização altera a solução ICA, o ganho que esta poderia dar na SIR acaba sendo perdido por causa da alteração da solução. O tipo de janela utilizado, entretanto, não altera muito o desempenho, porém, a janela de Hanning não é a ótima, e tem um desempenho inferior ao de outras janelas. Optamos por utilizar a janela de Chebyshev, embora a diferença de desempenho entre ela e as janelas da família Blackman seja mínima.

Aplicar a suavização é importante por causa dos ganhos marginais que ela proporciona ao SAR, ou seja, não perdemos nada e ganhamos algum desempenho, com um custo computacional desprezível.

Tabela 3.8: Comparação entre várias abordagens de separação, tanto Natural ICA como o método conjugando FastICA e Natural ICA.

Número de fontes e misturas - $N = M = 3$ Tempo de reverberação - $T_{60} = 150\,\text{ms}$ Tamanho da janela - $L = 2048$ Número de raias da FFT - $K = 2048$ Método para resolver a permutação - DOA + ConjCorr Número de realizações - 5 Disposição dos microfones e fontes - Figura A.3 Janela win_a - Hanning				
Janela	Suavização?	SIR médio	SDR médio	SAR médio
—	Não	$14,9\,\text{dB}$	$10,3\,\text{dB}$	$16,9\,\text{dB}$
Hanning	Sim	$15,2\,\text{dB}$	$12,3\,\text{dB}$	$18,3\,\text{dB}$
Chebyshev	Sim	$15,1\,\text{dB}$	$12,4\,\text{dB}$	$19,1\,\text{dB}$
Blackman	Sim	$14,9\,\text{dB}$	$12,3\,\text{dB}$	$18,9\,\text{dB}$
Nuttall	Sim	$14,9\,\text{dB}$	$12,2\,\text{dB}$	$19,0\,\text{dB}$
Kaiser	Sim	$15,0\,\text{dB}$	$11,2\,\text{dB}$	$17,1\,\text{dB}$

Capítulo 4

Métodos para Resolver o Problema da Permutação

O principal desafio do ICA no domínio da frequência é, sem dúvida, o problema da permutação. Várias técnicas foram e ainda estão sendo desenvolvidas para resolvê-lo. O foco deste capítulo é estudar e comparar os métodos mais recentes, que estão divididos em dois grupos. Seguindo o que foi introduzido na Seção 3.5, o primeiro grupo tenta estimar a localização das fontes, utilizando informações sobre o sistema de mistura, e classificar as fontes em cada raia de frequência baseado na sua localização. O segundo grupo utiliza as informações do espectro de frequência dos sinais estimados, mais especificamente, estudando a correlação entre raias de frequência deste, de forma a classificar as fontes em cada raia de frequência baseado nesta correlação. Dentro do primeiro grupo, se destacam os métodos que tentam descobrir a direção de chegada (DOA) ou a diferença entre tempos de chegada (TDOA) das fontes em cada sensor. No segundo grupo, os métodos se diferenciam no grupo de frequências a ser utilizado e na forma de medir a correlação. Tudo isto será abordado neste capítulo.

É importante agora introduzir os vetores-base, para facilitar a notação, que nada mais são do que as colunas da inversa da matriz separadora em cada raia de frequência. Consideraremos que eles são vetores linha para ficar consistente com nossa notação de filtros utilizada ao longo do trabalho. Os vetores-base são formados pelas colunas da matriz dada na Equação (4.1), com a condição de que $M \geq N$. Se $N = M$, a pseudoinversa se reduz à inversa.

$$\mathbf{A}_{M\times N}(k) = \begin{bmatrix} \mathbf{a}_1^T(k) & \mathbf{a}_2^T(k) & \cdots & \mathbf{a}_N^T(k) \end{bmatrix} = \mathbf{W}_{N\times M}^{\dagger}(k) \tag{4.1}$$

Se a BSS foi bem sucedida, os vetores-base são uma boa estimativa (a não ser

de uma permutação e escalamento) dos elementos da matriz de mistura, ou seja:

$$\mathbf{a}_i(k) \approx \alpha_{ik} \begin{bmatrix} h_{1i_P}(k) & h_{2i_P}(k) & \cdots & hMi_P(k) \end{bmatrix} \tag{4.2}$$

onde i_P indica que falta resolver o problema da permutação, e α_{ik} é o escalamento da fonte i na raia de frequência k.

4.1 Localização das Fontes

Algumas abordagens (padrões de diretividade, DOA, TDOA) utilizam a informação da localização do locutor em cada raia de frequência para resolver o problema da permutação. Elas se baseiam nos modelos de campo próximo e campo distante. O primeiro não tem nenhuma restrição, mas o segundo assume que as fontes estão distantes dos sensores. Ambos assumem que não há reverberação (*direct-path*) na sala, entretanto isso não impede que eles sejam aplicados a ambientes reverberantes. Aqui iremos mostrar primeiro o modelo mais geral e depois o mais específico, e no caso dos métodos, começaremos pelo mais simples e seguiremos mostrando os mais complexos. Seguindo esta idéia, primeiro consideraremos o modelo de campo próximo. A Figura 4.1 mostra o modelo de campo próximo. O vetor $\mathbf{q}_i$ especifica a posição da fonte i e o vetor $\mathbf{p}_j$ especifica a posição do microfone j. A distância entre o sensor j e a fonte i é dada por $\|\mathbf{q}_i - \mathbf{p}_j\|$, a distância da fonte à origem $\mathbf{o}$ é dada por $\|\mathbf{q}_i\|$, e a distância do sensor à origem é $\|\mathbf{p}_j\|$. A origem $\mathbf{o} = [0, 0, 0]^T$ pode ser arbitrada, como por exemplo, o centro do arranjo de microfones. Uma observação importante sobre esta figura é que deveríamos ter utilizado o vetor $\mathbf{p}_j - \mathbf{q}_i$ ao invés do vetor $\mathbf{q}_i - \mathbf{p}_j$, pois o som se origina das fontes em direção aos sensores, e na frente, como utilizaremos o conceito de velocidade de propagação, o sentido deste vetor é importante. Mesmo assim, na literatura se costuma utilizar este sentido, principalmente por causa do modelo de campo distante, onde o ângulo de direção de chegada (DOA) das fontes é mais facilmente deduzido e visualizado desta forma. Outra forma de ver este modelo, considerando os atrasos τ_{ji} entre o sensor j e a fonte i, é mostrado na Figura 4.2. Baseado nesta figura, podemos definir o TDOA (diferença entre tempos de chegada) de uma fonte i a dois sensores j e j' como:

$$\zeta_{jj'}(i) = \tau_{ji} - \tau_{j'i} \tag{4.3}$$

A restrição anterior de não-reverberação implica que o canal de propagação da onda de som só introduz uma atenuação e um atraso, e portanto, no domínio do tempo contínuo, o sinal no microfone j originado pela fonte i pode ser dado por [65]:

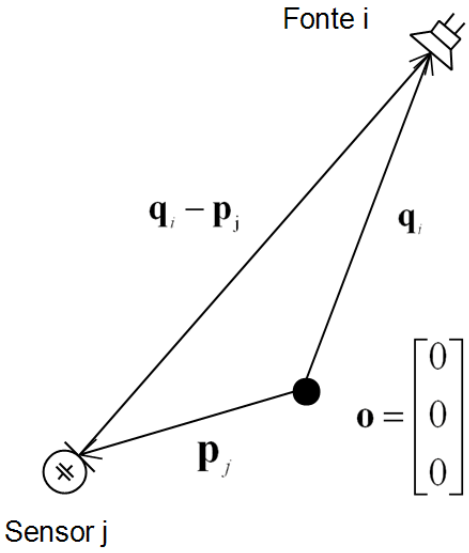

Figura 4.1: Modelo de campo próximo (ignorando reverberação).

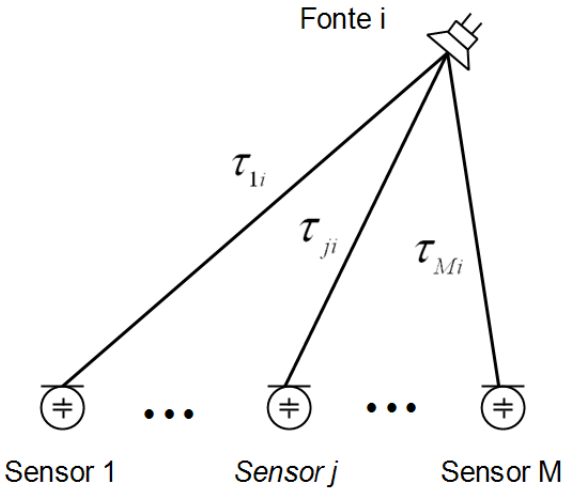

Figura 4.2: Modelo de campo próximo visualizado através dos atrasos entre os sensores e a fonte.

$$x_j(t) \approx \alpha_{ji} s_i(t - \tau_{ji}) \tag{4.4}$$

onde α_{ji} é a atenuação do caminho entre a fonte i e o sensor j e τ_{ji} é o atraso mostrado na Figura 4.2. Utilizamos o operador $\approx$ para indicar que o modelo é uma aproximação do caso real, onde há reverberação. Passando (4.4) para o domínio da frequência, temos:

$$x_j(f) \approx \alpha_{ji} \exp(-\hat{\jmath} 2\pi f \tau_{ji}) s_i(f) = h_{ji}(f) s_i(f) \tag{4.5}$$

onde $h_{ji}(f)$ é a resposta de frequência do caminho entre a fonte i e o sensor j. Ora, se chamarmos de c a velocidade de propagação da onda no meio, temos que:

$$\tau_{ji} = -c^{-1} \|\mathbf{q}_i - \mathbf{p}_j\| \tag{4.6}$$

onde o sinal de $-$ foi colocado para ficar consistente com a definição feita anteriormente[1]. Dessa forma, obtemos a resposta em frequência $h_{ji}(f)$ [66]:

$$h_{ji}(f) \approx \frac{1}{\|\mathbf{q}_i - \mathbf{p}_j\|} \exp(\hat{\jmath} 2\pi f c^{-1} (\|\mathbf{q}_i - \mathbf{p}_j\|)) \tag{4.7}$$

Observe que nesta expressão o sinal $-$ de (4.5) foi substituído por um sinal $+$, por causa do sentido que definimos para o vetor que liga a fonte ao sensor. Essa resposta não considera a reverberação, e o sinal é atenuado por um fator que é dependente da distância entre a fonte e o sensor. Para que a Equação (4.7) seja útil, precisamos da informação de mais de um sensor, e utilizamos a informação destes múltiplos sensores em conjunto. Fazemos isso porque, mesmo que estimarmos corretamente $h_{ji}(f)$, ainda temos o problema do escalamento. No entanto, o escalamento altera igualmente os elementos de uma mesma coluna da matriz de mistura, i.e, a razão $\frac{h_{ji}(f)}{h_{j'i}(f)}$, mostrada em (4.8) não sofre com este problema, onde j e j' indicam dois sensores diferentes.

$$\frac{h_{ji}(k)}{h_{j'i}(k)} = \frac{(\mathbf{W}_k^{-1} \mathbf{\Lambda}_k^{-1})_{ji}}{(\mathbf{W}_k^{-1} \mathbf{\Lambda}_k^{-1})_{j'i}} = \frac{(\mathbf{W}_k^{-1})_{ji}}{(\mathbf{W}_k^{-1})_{j'i}} = \frac{(\mathbf{a}_i)_j}{(\mathbf{a}_i)_{j'}} \tag{4.8}$$

O modelo mostrado lida com a frequência contínua f, porém, no nosso caso, utilizaremos a frequência discreta k, como nos capítulos anteriores. Aplicando (4.7)

[1] Na verdade c é um vetor que aponta para o sentido de propagação, e como alteramos o sentido do vetor entre as fontes e sensores, c deve ser negativo.

em (4.8):

$$\begin{aligned}\frac{h_{ji}(k)}{h_{j'i}(k)} &= \frac{\|\mathbf{q}_i - \mathbf{p}_{j'}\|}{\|\mathbf{q}_i - \mathbf{p}_j\|} \exp(\hat{\jmath} 2\pi f_k c^{-1}(\|\mathbf{q}_i - \mathbf{p}_j\| - \|\mathbf{q}_i - \mathbf{p}_{j'}\|)) \\ &= \frac{\|\mathbf{q}_i - \mathbf{p}_{j'}\|}{\|\mathbf{q}_i - \mathbf{p}_j\|} \exp(\hat{\jmath} 2\pi f_k \zeta_{jj'}(i))\end{aligned} \tag{4.9}$$

onde $\zeta_{jj'}(i)$ é o TDOA da fonte i aos sensores j e j'. Esse modelo sofre de *aliasing espacial*, que é similar ao *aliasing* temporal, mas aplicado ao espaço. Relembrando o *aliasing* temporal, segundo Nyquist, um sinal amostrado no tempo com período T_s deve ser limitado em banda por $\frac{1}{2T_s}$. Colocando de outra forma, no domínio da transformada de Fourier, este sinal não pode possuir componentes acima de $\frac{f_s}{2}$, pois isso gera ambiguidade. Por exemplo, uma senóide de frequência 3 kHz amostrada a $f_s = 8$ kHz e uma senóide de frequência 5kHz amostrada a $f_s = 8$ kHz geram sequências iguais, daí a ambiguidade. Da mesma forma, um sinal no espaço amostrado por microfones a uma distância d deve ser limitado em banda. A análise desse limite de banda no modelo de campo próximo é complexa, e em [67], o autor mostra que não há como evitar *aliasing* completamente neste modelo. Entretanto, na prática, é comum utilizar neste modelo a mesma restrição que no modelo de campo distante, onde o limite de banda em função da distância d é mais simples de se encontrar, como mostraremos a seguir.

O modelo de campo distante consiste em simplificar a expressão (4.9) considerando que a fonte está distante dos sensores. A Figura 4.3 ilustra esse caso. Se a fonte estiver suficientemente distante, os vetores $\mathbf{q}_i - \mathbf{p}_j$ e $\mathbf{q}_i - \mathbf{p}_{j'}$ são quase paralelos. A condição para que este modelo sejá válido é que $\|\mathbf{q}_i - \mathbf{p}_j\| \gg \|\mathbf{p}_j - \mathbf{p}_{j'}\|$, i.e, a distância entre os sensores seja muito menor do que a distância entre o arranjo de sensores e a fonte i.

Na Figura 4.3, vemos que o TDOA (de acordo com (4.3) e (4.6)) $\zeta_{jj'}(i) = \tau_{ji} - \tau j' i = -c^{-1}(\|\mathbf{q}_i - \mathbf{p}_j\| - \|\mathbf{q}_i - \mathbf{p}_{j'}\|)$ pode ser dado por $\zeta_{jj'}(i) = -c^{-1}(\mathbf{p}_{j'} - \mathbf{p}_j)^T \mathbf{u}_{ji}$, onde $\mathbf{u}_{ji}$ é um vetor unitário que aponta do sensor j na direção da fonte i, i.e, $\mathbf{u}_{ji} = \frac{\mathbf{q}_i - \mathbf{p}_j}{\|\mathbf{q}_i - \mathbf{p}_j\|}$. Daí, a expressão (4.9) pode ser reescrita como [66]:

$$\frac{h_{ji}(k)}{h_{j'i}(k)} = \exp(\hat{\jmath} 2\pi f_k c^{-1}(\mathbf{p}_{j'} - \mathbf{p}_j)^T \mathbf{u}_{ji}) \tag{4.10}$$

onde $\frac{\|\mathbf{q}_i - \mathbf{p}_{j'}\|}{\|\mathbf{q}_i - \mathbf{p}_j\|}$ foi omitido pois, nesse modelo, $\|\mathbf{q}_i - \mathbf{p}_{j'}\| \approx \|\mathbf{q}_i - \mathbf{p}_j\|$. O modelo resumido pela Equação (4.10) pode ser utilizado entre quaisquer dois pares de sensores. Se estivermos interessados apenas no ângulo $\theta_{jj'}(i)$, podemos simplificar este

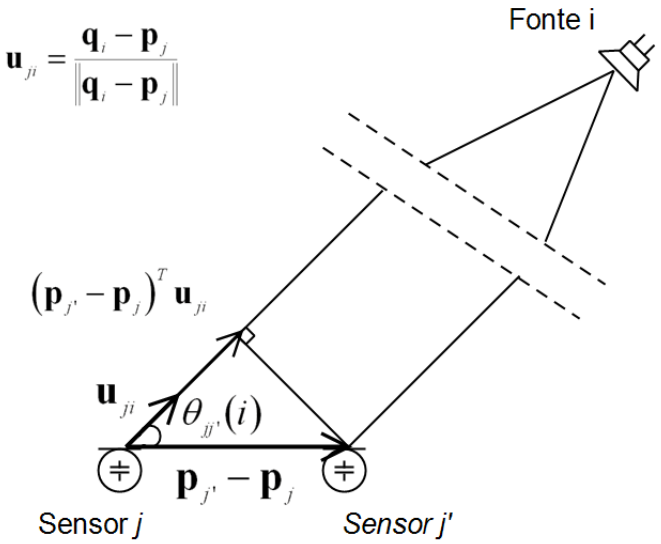

Figura 4.3: Modelo de campo distante (ignorando reverberação).

modelo:

$$\begin{aligned} \cos(\theta_{jj'}(i)) &= \frac{(\mathbf{p}_{j'} - \mathbf{p}_j)^T \mathbf{u}_{ji}}{\|\mathbf{p}_{j'} - \mathbf{p}_j\|} \quad \therefore \\ \frac{h_{ji}(k)}{h_{j'i}(k)} &= \exp(\hat{\jmath} 2\pi f_k c^{-1} (\|\mathbf{p}_{j'} - \mathbf{p}_j\|) \cos(\theta_{jj'}(i))) \\ &= \exp(\hat{\jmath} 2\pi f_k c^{-1} d_{jj'} \cos(\theta_{jj'}(i))) \end{aligned} \tag{4.11}$$

onde $d_{jj'}$ é a distância entre os sensores j' e j. Algumas literaturas trocam o cos(.) pelo sen(.) na expressão (4.11), principalmente as relacionadas a *beamforming*, como [68]. Isso muda a definição de $\theta_{jj'}(i)$, que passa a ser 0° quando a fonte está perpendicular ao eixo do arranjo de microfones, em vez de ser 90°, como no nosso caso (ver Figura 4.3). Há uma versão da Equação (4.7) para o modelo de campo distante muito utilizada, onde se desconsidera a atenuação, pois esta informação só nos diz a distância entre a fonte e o arranjo de sensores (no modelo de campo distante, a fonte está tão distante dos sensores que a distância entre ela e qualquer um dos sensores é praticamente a mesma), e idealmente, no modelo de campo distante, essa distância tende a ∞. Também é arbitrada uma origem $\mathbf{o}$ onde a fase é zero, e desta forma, obtemos a fase relativa a esta origem, para que não precisemos da distância entre a fonte e os sensores. Estas considerações são ilustradas na Figura 4.4.

Com essas considerações, a resposta em frequência $h_{ji}(f)$ no modelo de campo distante fica:

$$h_{ji}(f) \approx \exp(\hat{\jmath} 2\pi f c^{-1} \|\mathbf{p}_j\| \cos(\theta_j(i))) \tag{4.12}$$

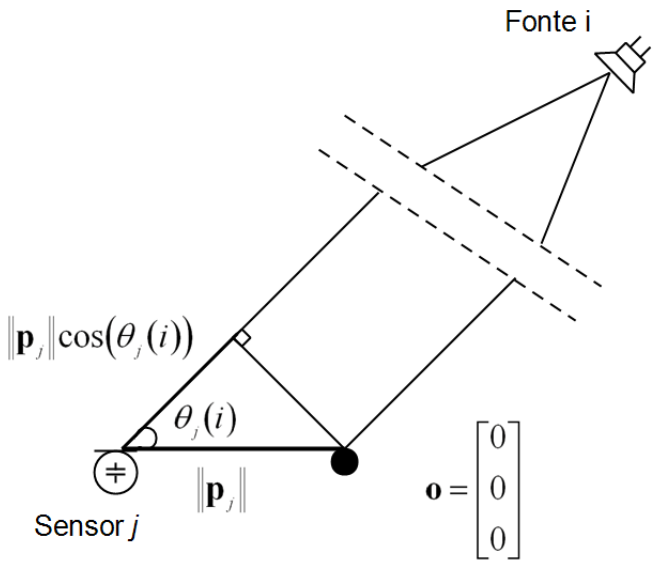

Figura 4.4: Modelo de campo distante (ignorando reverberação).

Utilizando este modelo, podemos derivar o limite de banda em função da distância dos microfones. Definamos um eixo que passa exatamente pela linha dos microfones, ou seja, pelo vetor $\mathbf{p}_{j'} - \mathbf{p}_j$. Seja l o número real que define a posição neste eixo. Seja $x_i(l)$ o sinal da fonte i assim como é visto no ponto l desse eixo, que é dado por:

$$x_i(l) \approx \alpha_{li} \exp(\hat{j}2\pi f c^{-1} l \cos(\theta_l(i)))s_i(t) \tag{4.13}$$

onde α_{li} representa a atenuação do sinal vindo da fonte i que chega no ponto l e $\cos(\theta_l(i))$ é o cosseno do ângulo DOA da fonte i no ponto l, similar ao ângulo $\cos(\theta_j(i))$ em (4.12). A transformada de Fourier de (4.13) em função de l é dada por (4.14), onde α_{ji} é uma constante, ξ é a "frequência espacial" (relacionada à dimensão l) e $\delta(.)$ é a função delta de Dirac.

$$X_i(\xi) = \alpha_{ji}\delta(\xi - f c^{-1} \cos(\theta_l(i))) \tag{4.14}$$

Utilizando Nyquist, para não haver *aliasing* espacial, a máxima frequência ξ que pode ser representada é $\frac{1}{2d_s}$, onde d_s é a distância de amostragem, no nosso caso, a distância d entre os microfones. Isto é similar a dizer que, para não haver *aliasing* temporal, a frequência f do sinal não pode ser maior que $\frac{1}{2T_s} = \frac{f_s}{2}$. De (4.14), para que isso aconteça,

$$f c^{-1} \cos(\theta_l(i)) < \frac{1}{2d} \Rightarrow f < \frac{c}{2d\cos(\theta_l(i))} \Rightarrow f < \frac{c}{2d} \tag{4.15}$$

onde a última simplificação foi possível porque o pior caso ocorre no máximo valor de $\cos(\theta_l(i))$, ou seja, quando ele é 1. A condição (4.15) é bem conhecida na literatura, e também é utilizada no caso de campo próximo, como dito acima, pois mesmo que essa condição não seja suficiente para evitar *aliasing* no caso de campo próximo, a condição diminui bastante seu efeito. Para que esta análise de banda estreita seja válida, consideramos implicitamente que podemos descrever a resposta de um arranjo de microfones a uma onda plana de banda larga linearmente e utilizando a transformada de Fourier. Para uma análise matemática mais precisa sobre o assunto, ver [69]. Neste trabalho, o autor realiza a análise acima em banda larga, e chega à conclusão de que o *aliasing* espacial depende também da resposta do sinal, de seus harmônicos, estacionaridade e outros fatores.

4.1.1 Padrões de Diretividade

Para derivar o algoritmo de padrões de diretividade, assumimos que os sensores (microfones) estão montados em linha, como na Figura 4.5, e equiespaçados, i.e, $d_j - d_{j'}$, para dois sensores adjacentes, é igual. Também assumimos, como dito anteriormente, que não há reverberação na sala. Mesmo com essa restrição, em [70], o autor comparou este método com os métodos de correlação e chegou à conclusão de que se obtém resultados melhores em ambientes reverberantes com o método de padrões de diretividade.

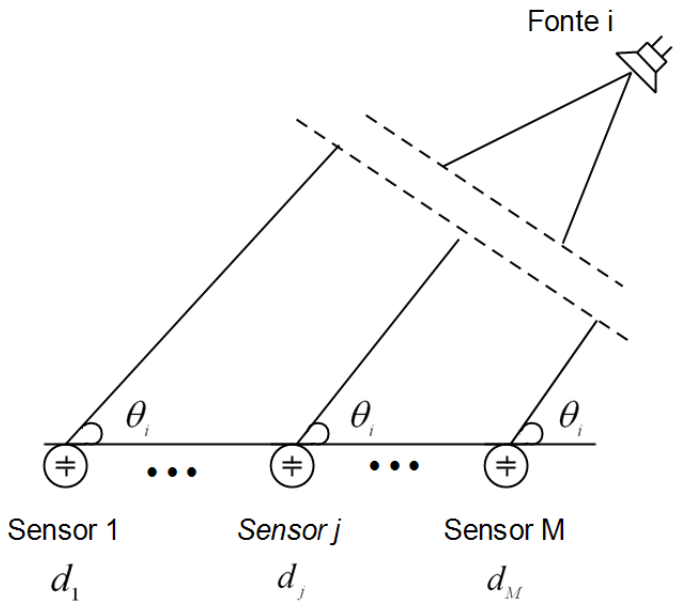

Figura 4.5: Montagem em linha de microfones, no modelo de campo distante. Assume-se que os ângulos de chegada de uma mesma fonte são os mesmos para todos os sensores.

Segundo mostrado na figura, assumimos o modelo de campo distante, e daí os ângulos de chegada DOA de uma mesma fonte são os mesmos para todos os sensores j. Os escalares d_j representam a posição dos sensores em cima do eixo de montagem destes. A origem do eixo pode ser arbitrada, como por exemplo, o centro da montagem ou um dos microfones.

Após o BSS, cada saída $y_i(k)$ pode ser dada por (onde o índice k foi omitido, para simplificar):

$$\begin{aligned} y_i &= w_{11}x_1 + w_{12}x_2 + \cdots + w_{1M}x_M \\ &= (w_{11}h_{11}s_1 + w_{11}h_{12}s_2 + \cdots + w_{11}h_{1N}s_N) + \\ &\quad + (w_{12}h_{21}s_1 + w_{12}h_{22}s_2 + \cdots + w_{12}h_{2N}s_N) + \cdots + \\ &\quad + (w_{1M}h_{M1}s_1 + w_{1M}h_{M2}s_2 + \cdots + w_{1M}h_{MN}s_N) \\ &= (w_{11}h_{11} + \cdots + w_{1M}hM1)s_1 + (w_{11}h_{12} + \cdots + w_{1M}hM2)s_2 + \cdots + \\ &\quad + (w_{11}h_{1N} + \cdots + w_{1M}h_{MN})s_N \end{aligned} \tag{4.16}$$

De acordo com o modelo utilizado, cada $h_{ji}(k)$ é dado por (4.12), onde $\|\mathbf{p}_j\| = d_j$. Então, a única diferença entre $h_{1i}, h_{2i}, \cdots, h_{Mi}$ é a distância d_j, pois, como dito acima, o ângulo (DOA) de uma fonte i é o mesmo para todos os sensores j. Ora, definamos então o padrão de diretividade F_i como:

$$F_i(k,\theta) = \sum_{j=1}^{M} w_{1j}(k)h_{ji}(k) = \sum_{j=1}^{M} w_{ij}(k)\exp(\hat{\jmath}2\pi f c^{-1} d_j \cos(\theta)) \tag{4.17}$$

onde o ângulo θ é uma variável. Substituindo (4.17) em (4.16), temos:

$$y_i = F_i(\theta(1))s_1 + F_i(\theta(2))s_2 + \cdots + F_i(\theta(N))s_N \tag{4.18}$$

Fica claro em (4.18) que a função $F_i(\theta(i))$ referente à fonte i deve ser máxima e, idealmente, todas as outras $F_i(\theta(k)), \forall k \neq i$ devem ser nulas. Ou seja, se traçarmos um gráfico de F_i em função de θ, os ângulos correspondentes às outras fontes devem ser nulos (mínimos da função, na prática). A Figura 4.6 mostra o padrão de diretividade F para 2 fontes, após o BSS ter sido realizado com sucesso e com o problema da permutação já resolvido. O mínimo do padrão de diretividade da fonte 1 representa o DOA da fonte 2, pois $F_1(\theta(2))$ na expressão (4.18) deve ser nulo e, de forma similar, o mínimo do padrão de diretividade da fonte 2 representa o DOA da fonte 1. A simulação foi feita num ambiente reverberante, então o modelo de campo distante não representa com exatidão o modelo da sala. Por isto, os padrões de diretividade de algumas frequências mais baixas, ou altas demais, começam a sofrer com a inexatidão do modelo, e fica difícil estimar os mínimos, como pode ser visto na figura.

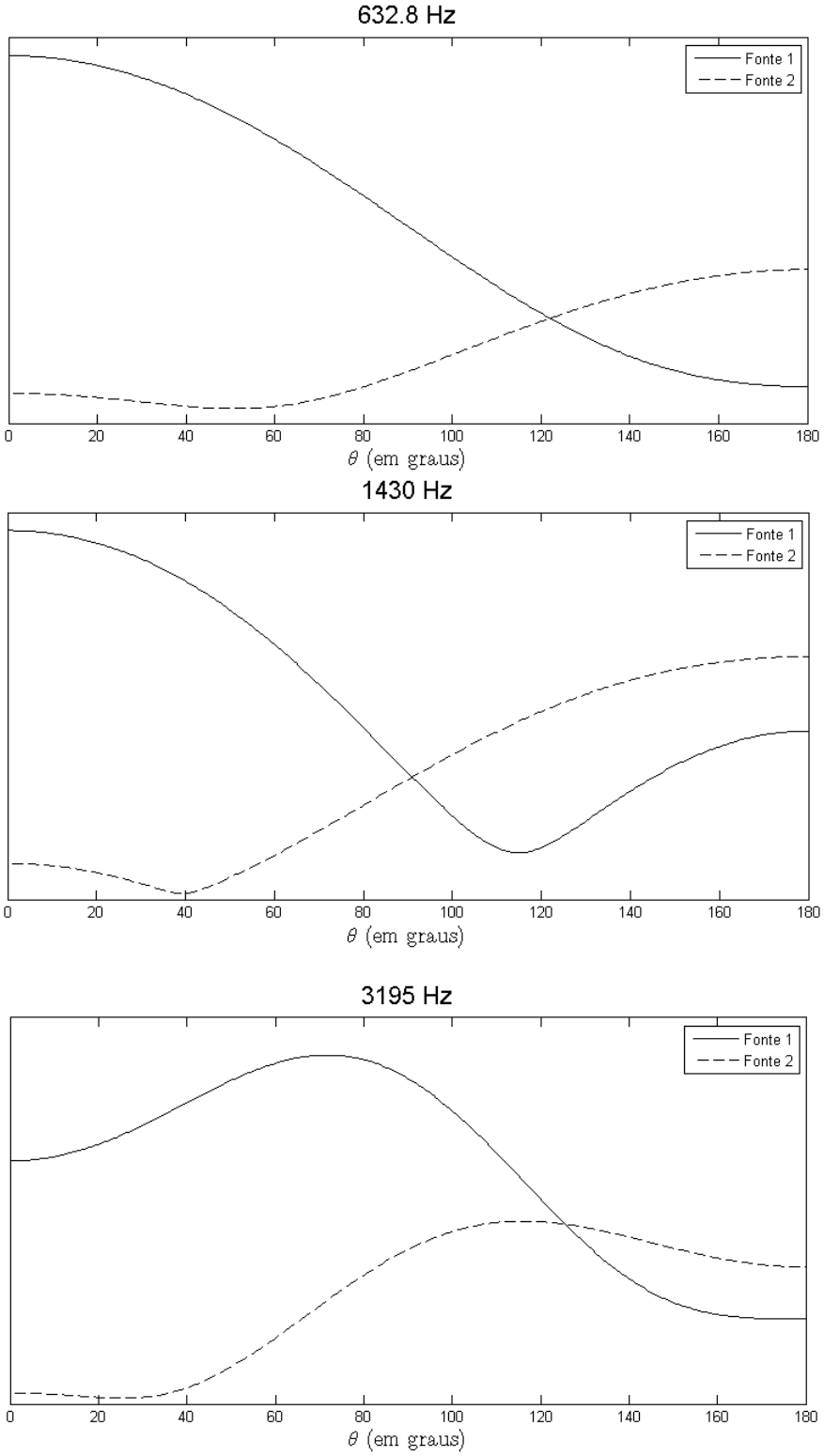

Figura 4.6: Padrões de diretividade F_i de dois sinais de voz i para 3 frequências diferentes em um ambiente com $T_{60} = 130\,\text{ms}$. Os padrões foram gerados após o BSS ter sido realizado com sucesso e com o problema da permutação resolvido, utilizando a expressão (4.17) com os $w_{ij}(k)$ encontrados. Para frequências baixas ou altas demais, fica difícil encontrar o mínimo, pois a reverberação começa a fazer diferença no modelo. Ambas as fontes estavam a 1 metro da montagem de microfones. O DOA real da fonte 1 era 40° e o da fonte 2, 135°.

Encontrar o mínimo de uma função não é uma tarefa computacionalmente simples, e a situação piora quando há mais de duas fontes, pois, nesse caso, existem mínimos locais. A Figura 4.7 mostra o padrão de diretividade num caso deste tipo, para uma dada frequência. Para que a abordagem funcione neste caso, é necessário estimar dois mínimos para cada fonte em cada raia de frequência k. Esta figura também ilustra outro problema, inerente ao modelo de campo distante. Acontece que o ângulo $\theta_j(i)$ em (4.12) deve estar no intervalo $[0, \pi]$, e se não estiver, isto gera uma ambiguidade, por causa da simetria do cosseno. Se o $\cos(\theta_j(i))$ de duas fontes for o mesmo, no modelo de campo distante, as fontes estão na mesma posição e possuem o mesmo DOA. O modelo de campo próximo também possui suas limitações, como por exemplo, duas fontes com o mesmo TDOA $\zeta_{jj'}(i)$. No modelo de campo próximo, essas duas fontes estão na mesma posição. Porém, no caso do campo próximo, isso pode ser remediado colocando-se mais sensores, daí mesmo que os TDOAs de duas fontes 1 e 2 sejam iguais, i.e, $\zeta_{jj'}(1) = \zeta_{jj'}(2)$, o TDOA utilizando sensores diferentes pode ser diferente, $\zeta_{jj''}(1) \neq \zeta_{jj''}(2)$, resolvendo-se a ambiguidade.

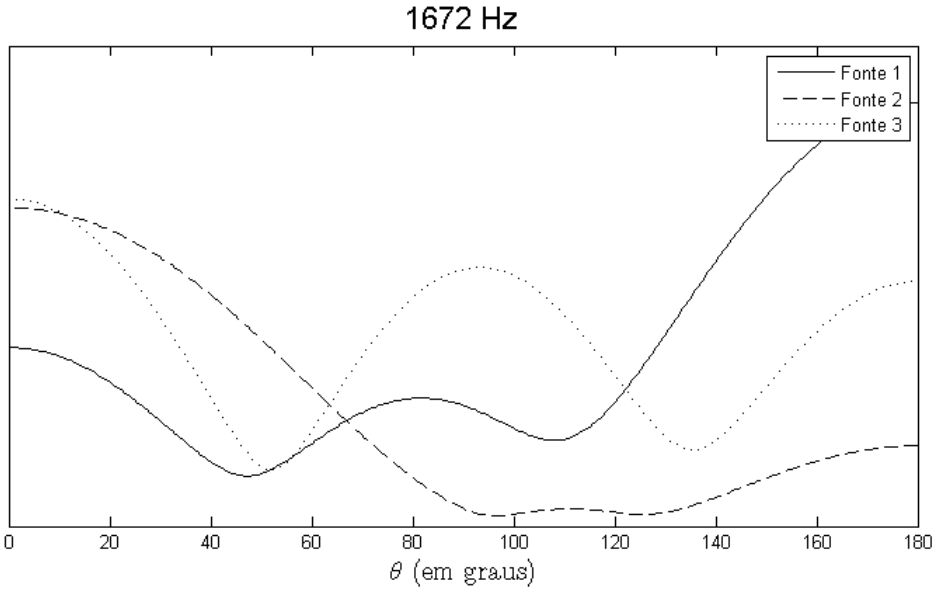

Figura 4.7: Padrões de diretividade quando há 3 fontes presentes, o que gera mínimos locais no padrão de diretividade. O DOA real da fonte 1 é 135°, da fonte 2 é 40° e da fonte 3 é 280° (80° na realidade, por causa da ambiguidade do modelo de campo distante).

O procedimento para resolver o problema da permutação utilizando padrões de diretividade consiste em encontrar os valores de θ que minimizam a função $F_i(k, \theta)$ para todas as fontes i e raias de frequência k, e compará-los para decidir se houve permutação ou não. Primeiramente, deve-se calcular o DOA médio para todas as frequências k, que deve ser bem próximo do DOA real, mesmo que em algumas frequências o mínimo de $F_i(k, \theta)$ esteja obscuro. Um exemplo da média entre os padrões de diretividade de todas as frequências é mostrado na Figura 4.8, que mostra

o mesmo caso da Figura 4.7. Observe que os mínimos são bem próximos dos valores de DOA reais.

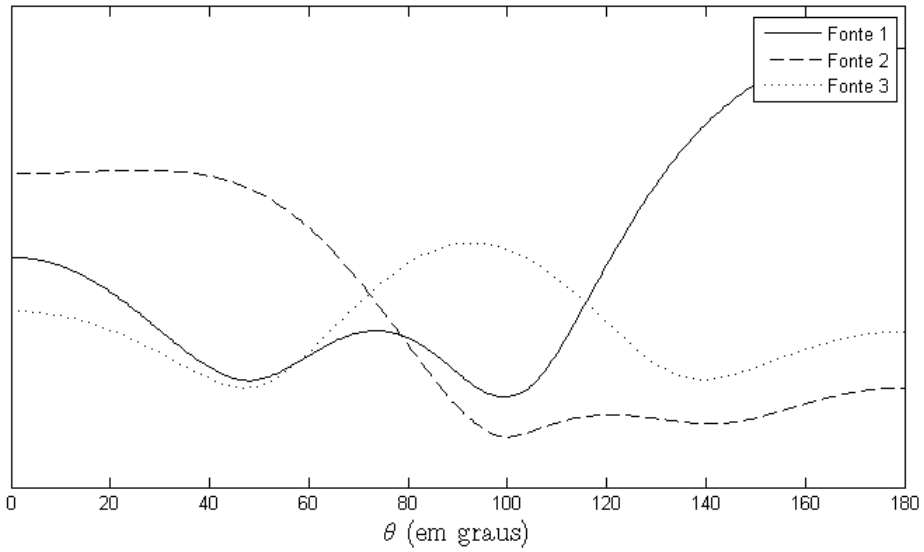

Figura 4.8: Média dos padrões de diretividade $F_i(k, \theta)$ para todas as frequências k. O DOA real da fonte 1 é 135°, da fonte 2 é 40° e da fonte 3 é 280° (80° na realidade, por causa da ambiguidade do modelo de campo distante).

Em posse do DOA médio, pode-se decidir, em cada raia de frequência, se houve permutação ou não, comparando-se os valores de DOA de cada frequência com os valores "reais" encontrados anteriormente. Este método possui algumas desvantagens que inibiram sua utilização, em favor do método mostrado na próxima seção, que utiliza DOA calculado de outra forma. As duas desvantagens básicas deste modelo são:

1. DOAs não podem ser bem estimados em algumas frequências, principalmente as frequências baixas, onde a diferença de fase, causada pelo espaçamento do sensor, é muito pequena[2];

2. Calcular os padrões de diretividade é computacionalmente intensivo;

3. Estimar os DOAs através de padrões de diretividade, quando há mais que duas fontes, é difícil (por causa dos mínimos locais).

A primeira desvantagem é inerente ao DOA, não interessando de que forma foi encontrado (inclusive a mostrada na Seção 4.1.2, a seguir). As outras duas ocorrem porque estamos encontrando o DOA utilizando padrões de diretividade, e a próxima

[2]Em frequências mais baixas, o comprimento de onda $\lambda = \frac{c}{f}$ é grande, por exemplo, em $f = 150Hz$ ele é $2,28m$, o que significa que para um espaçamento de sensor de $4cm$, a fase só varia de $6,3°$. Como calculamos o DOA baseado nesta diferença de fase (ver (4.11)), pequenos erros númericos ou de estimativa da matriz de mistura levam a valores errados de DOA.

abordagem corrige estes problemas. Embora essa abordagem não seja utilizada, entendê-la é essencial para compreender e analisar outras abordagens.

4.1.2 Direção de Chegada (DOA)

O método DOA tenta encontrar a direção de chegada das fontes, mais precisamente, o ângulo $\theta_j(i)$ em (4.12). As suposições são as mesmas do método de padrões de diretividade, isto é, considera-se um ambiente sem reverberação, os sensores estão montados em linha, e as fontes estão suficientemente distantes destes para que possamos considerar que $\theta_j(i) = \theta_{j'}(i), \forall j, j' = 1, \cdots, M$, e só precisamos encontrar o ângulo de cada fonte $\theta(i)$, que é o mesmo para qualquer sensor.

O método de padrões de diretividade, como citado acima, é computacionalmente intensivo, e fica complicado se houver mais que duas fontes, pois começam a aparecer mínimos locais nas funções. Em [71], o autor propôs uma forma mais simples de encontrar cada DOA. Ele propôs uma abordagem algébrica, que por não envolver algoritmos de minimização, é muito mais rápida. O DOA é encontrado através da Equação (4.11), que pode ser simplificada pelas condições citadas acima:

$$\begin{aligned}
\frac{h_{ji}(k)}{h_{j'i}(k)} &= \exp(\hat{\jmath} 2\pi f_k c^{-1}(d_j - d_{j'}) \cos(\theta_k(i))) \\
\arg\left(\frac{h_{ji}(k)}{h_{j'i}(k)}\right) &= 2\pi f_k c^{-1}(d_j - d_{j'}) \cos(\theta_k(i)) \\
\cos(\theta_k(i)) &= \frac{\arg\left(\dfrac{h_{ji}(k)}{h_{j'i}(k)}\right)}{2\pi f_k c^{-1}(d_j - d_{j'})}
\end{aligned}$$

De (4.8):

$$\theta_k(i) = \arccos\left(\frac{\arg\left(\dfrac{(\mathbf{a}_i(k))_j}{(\mathbf{a}_i(k))_{j'}}\right)}{2\pi f_k c^{-1}(d_j - d_{j'})}\right) \tag{4.19}$$

onde o operador $\arg(c)$ retorna a fase de um número complexo c, e o sub-índice k adicionado ao ângulo θ não significa que o ângulo muda com a frequência. Teoricamente, o ângulo $\theta_k(i)$ deve ser o mesmo para toda frequência k. A Equação (4.19) representa uma forma analítica de se encontrar o DOA de uma fonte, utilizando os vetores-base $\mathbf{a}_i$. Há um problema, no entanto. O valor de arccos(.) não é definido para valores fora do intervalo $[-1, 1]$, então, se o termo dentro de arccos(.) em (4.19) estiver fora deste intervalo, o DOA não pode ser encontrado. Isso acontece para valores pequenos de f_k, por exemplo. Obviamente, a expressão é indefinida quando $f_k = 0$. Se o valor de arccos(.) for indefinido, o que se faz na prática é escolher outro valor para j', se houver mais do que 2 sensores. Todas as combinações $[j, j']$

são testadas, até que se encontre um valor para o qual o DOA possa ser calculado. Em [71], o autor demonstrou que quando o número de fontes é igual a 2, utilizar esta expressão obtém os mesmos resultados do que utilizar o método de padrões de diretividade.

As Figuras 4.9 e 4.10 mostram os DOAs encontrados para 2 e 3 fontes, respectivamente. O DOA estimado foi bem parecido com o DOA real. Nos dois casos, o problema da permutação foi resolvido de forma supervisionada e depois foi encontrado o DOA de cada fonte. As linhas nos gráficos são a mediana de cada fonte. Na literatura, é comum utilizar-se a média. Porém, como pode ser observado nos gráficos, a medida de DOA possui muitos *outliers*, e, portanto, a mediana se torna uma medida muito mais robusta. Na prática, utilizar a mediana resulta em valores de DOA mais próximos dos valores reais, como foi observado em alguns testes. De acordo com alguns testes, em cerca de 2% a 5% das raias de frequência, a medida de DOA não pode ser encontrada, pois o valor de arccos(.) em (4.19) é indefinido.

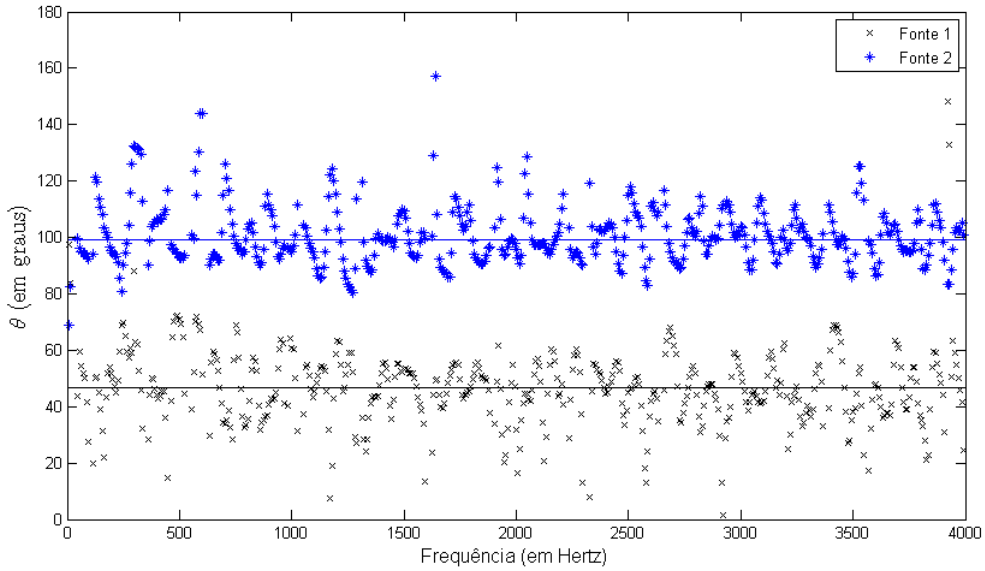

Figura 4.9: DOA encontrados em função da frequência para o caso de 2 fontes. O DOA real da fonte 1 é 45° e o da fonte 2 é 100°.

Para utilizar a informação do DOA e resolver o problema da permutação, primeiro se encontra o DOA de cada uma das fontes em cada raia de frequência k segundo (4.19). Depois eles são ordenados em cada raia de frequência, e essa ordenação determina a permutação. Utilizar simplesmente o DOA para resolver o problema da permutação não produz resultados satisfatórios. O ideal é que se utilize outro método em conjunto com ele, como por exemplo, a correlação espectral (Seção 4.2).

Em [72], o autor argumenta que utilizar espaçamentos de sensores maiores melhora a resolução do DOA, pois aumenta a quantidade de posições discretas onde

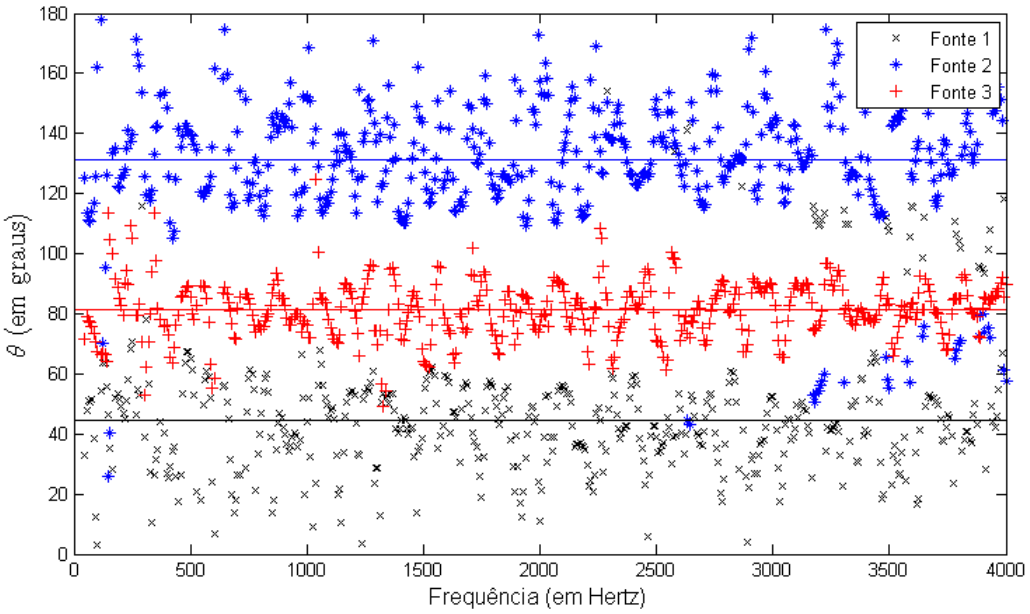

Figura 4.10: DOA encontrados em função da frequência para o caso de 3 fontes, relativamente mais difícil do que o caso de 2 fontes. O DOA real da fonte 1 é 40°, da fonte 2 é 80° e da fonte 3 é 135°

podem haver fontes (esse número é discreto por causa da amostragem). Contudo, no mesmo artigo, o autor depois conclui que não obtém resultados melhores aumentando muito a distância d entre os sensores pois esbarra na condição (4.15), o que limita as frequências onde se pode estimar o DOA, e, portanto, polarizando demais as estimativas. Em suma, métodos que se utilizam de localização de uma forma geral não ganham muito com espaçamentos grandes entre sensores.

4.1.3 Diferença entre Tempos de Chegada (TDOA)

Outra abordagem que se utiliza da localização das fontes estima o TDOA das fontes em cada par de sensores e depois "clusteriza" os resultados, utilizando um algoritmo qualquer de clusterização. O TDOA é dado por (4.3), e utiliza o modelo de campo próximo. Na Figura 4.2, fica claro que para M sensores, temos $\frac{1}{2}M(M-1)$ combinações diferentes entre os atrasos, e consequentemente, o mesmo número de TDOAs diferentes. Entretanto, eles são redundantes, por exemplo $\tau_{1i} - \tau_{2i} = (\tau_{3i} - \tau_{2i}) + (\tau_{1i} - \tau_{3i})$, i.e, $\zeta_{12}(i) = \zeta_{32}(i) + \zeta_{13}(i)$ (um TDOA é uma combinação linear de outros dois, e, portanto, não traz nenhuma informação a mais). Para M sensores, existem apenas $M-1$ TDOAs únicos, portanto, vamos definir os TDOAs como:

$$\zeta_j(i) = \tau_{ji} - \tau_{Ji} \tag{4.20}$$

onde J é um microfone de referência. Omitiremos a dependência de J em ζ para simplificar.

O TDOA é encontrado diretamente da Equação (4.9):

$$\frac{h_{ji}(k)}{h_{Ji}(k)} = \frac{\|\mathbf{q}_i - \mathbf{p}_J\|}{\|\mathbf{q}_i - \mathbf{p}_j\|} \exp(\hat{\jmath} 2\pi f_k \zeta_j(i))$$

$$\arg\left(\frac{h_{ji}(k)}{h_{Ji}(k)}\right) = 2\pi f_k \zeta_j(i)$$

$$\zeta_j(i) = \frac{\arg\left(\frac{h_{ji}(k)}{h_{Ji}(k)}\right)}{2\pi f_k}$$

De (4.8):

$$\zeta_j(i) = \frac{\arg\left(\frac{(\mathbf{a}_i(k))_j}{(\mathbf{a}_i(k))_J}\right)}{2\pi f_k} \tag{4.21}$$

onde vemos a dependência de ζ com a frequência. Obviamente, se f_k for zero, não há estimativa de TDOA. Da mesma forma, para frequências pequenas, a estimativa não é muito estável [4]. A faixa de frequências para a qual o método é aplicável ainda é (4.15), com uma pequena modificação:

$$f < \frac{c}{2d_{max}} \tag{4.22}$$

onde d foi substituído por d_{max}, que é a distância máxima entre os sensores, pois o TDOA é calculado entre pares de sensores que não necessariamente precisam estar adjacentes. A distância d_{max} representa, então, o pior caso. Deste resultado, já podemos pensar em montagens de sensores mais eficientes para este método. Uma montagem em linha certamente não é eficiente, pois $d_{max} = Md$, o que limita muito a faixa de frequências que podemos trabalhar. Uma montagem em "cluster" 2D, como desenhar um polígono em um plano com os sensores, é uma solução melhor, pois, para $M = 4$, por exemplo, $d_{max} = \sqrt{2}d$, e para $M = 6$, $d_{max} = (1 + \sqrt{3})d$, o que aumenta muito a faixa de frequências que podemos trabalhar. Uma solução ainda melhor é utilizar um polígono 3D para isso. Observe também que independentemente do método de montagem, à medida que se aumenta o número M, mais díficil fica nossa estimativa dos TDOAs. Em [73], o autor propõe uma alternativa para melhorar o desempenho quando o espaçamento entre sensores é grande. No nosso caso, utilizaremos espaçamentos pequenos, e não avaliaremos esta proposta, entretanto, é importante citá-la.

Definamos um vetor-linha de tamanho $M - 1$ que contenha todos os TDOAs de

uma determinada fonte em uma determinada frequência:

$$\zeta_i(k) = \left[\zeta_1(i,k) \quad \zeta_2(i,k) \quad \cdots \quad \zeta_{M-1}(i,k)\right] |_{i=\mathbf{P}_k(i)} \tag{4.23}$$

onde "$|_{i=\mathbf{P}_k(i)}$" simboliza que o problema da permutação ainda não foi resolvido. Ora, existem N vetores $\mathbf{c}_\zeta(i)$ (centróides) que definem a posição real das fontes. Se o ICA funcionou bem, nossas estimativas em cada frequência devem formar N *clusters* de vetores ao redor destes N centróides $\mathbf{c}_\zeta(i)$. Não conhecemos a posição real das fontes, porém, podemos estimá-las através dos vetores $\zeta_i(k)$ encontrados, utilizando algum algoritmo de clusterização. Um algoritmo de clusterização encontra os centróides de um conjunto de vetores, no nosso caso, o conjunto $\zeta_i(k), \forall i = 1, \cdots, N \quad , k = 1, \cdots, K_{lim}$, formado por $N \times K_{lim}$ amostras de vetores. O número de raias de frequência é K_{lim} ao invés de K por causa da limitação (4.22). O número de centróides que o algoritmo precisa encontrar é uma entrada deste algoritmo, e no nosso caso, é N. A Figura 4.11 mostra um exemplo de estimativas de TDOA e o resultado de sua clusterização. Claramente se observam os 3 clusters, um para cada fonte. Nesta situação o tempo de reverberação T_{60} foi de 100 ms, mas a clusterização não é tão simples assim para salas mais reverberantes. A Figura 4.12 mostra os valores de TDOA para uma sala com $T_{60} = 250$ ms. Observe que é muito mais difícil clusterizar os TDOAs, porque o modelo considera apenas o caminho direto do som, e é invalidado à medida que a reverberação aumenta.

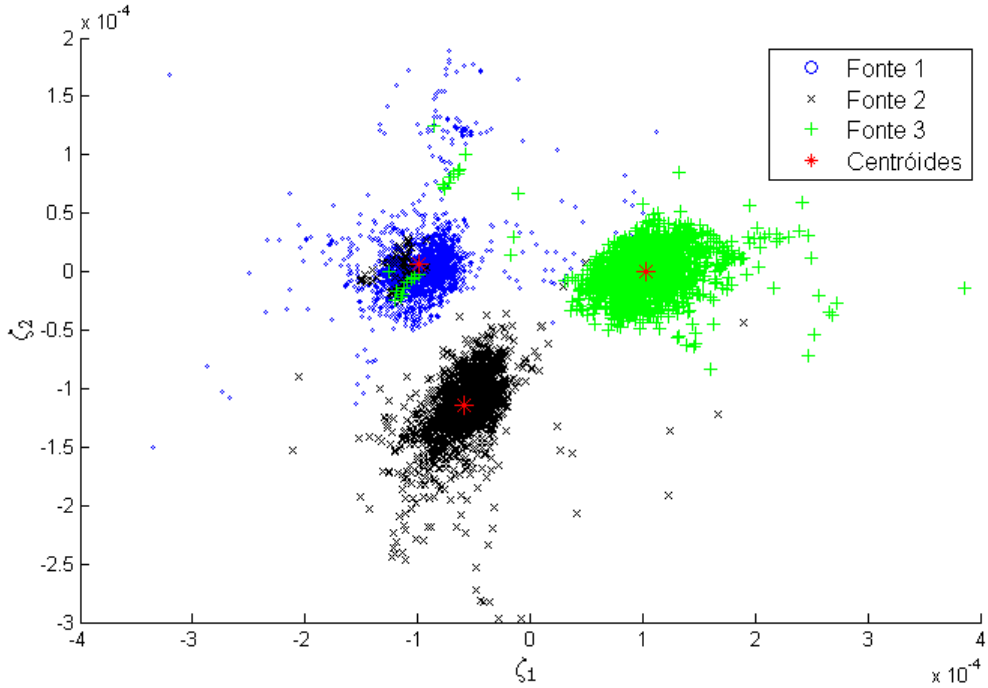

Figura 4.11: Resultado da clusterização dos TDOAs de 3 fontes em uma sala com $T_{60} = 100$ ms utilizando *K-means*.

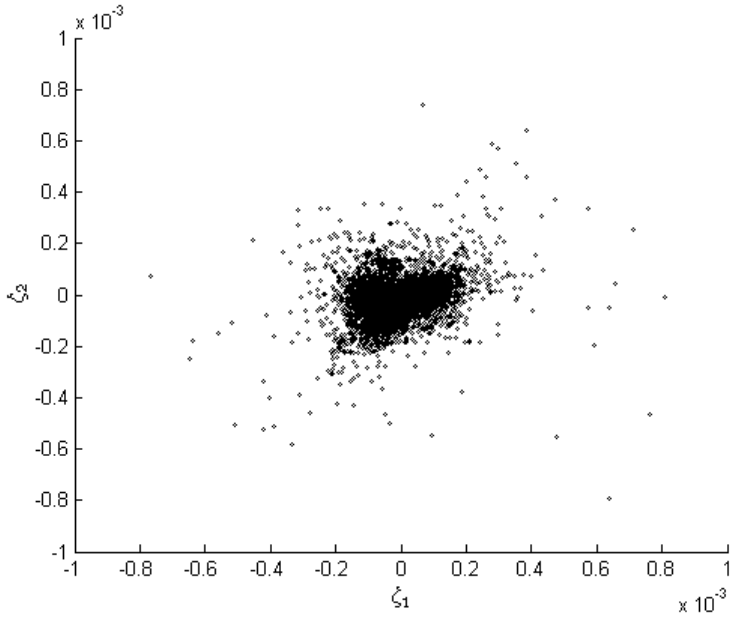

Figura 4.12: TDOAs de 3 fontes em uma sala com $T_{60} = 250\,\text{ms}$. A clusterização não produz resultados bons neste caso.

Os algoritmos de clusterização constituem uma parte de um grupo mais geral de técnicas de aprendizagem não-supervisionada, e qualquer um deles pode ser utilizado para encontrar os agrupamentos em torno dos TDOAs reais. Aqui optamos por utilizar o *K-means* [74], que é um dos mais simples métodos de clusterização. Os N centróides são inicializados amostrando-se arbitrariamente N vetores do conjunto, e depois computa-se a distância entre estes centróides e cada um dos vetores do conjunto. Essa distância em geral é o quadrado da distância Euclidiana, isto é, $\|\zeta_i(k) - \mathbf{c}_\zeta(i)\|^2$. Cada vetor é classificado de acordo com essa distância, e enquadrado no grupo (*cluster*) i cujo centróide é $\mathbf{c}_\zeta(i)$. Após todos os vetores serem classificados, os centróides $\mathbf{c}_\zeta(i)$ são calculados novamente, através da média de todos os vetores $\zeta_i(k)$ pertencentes àquele *cluster* i. A seguir, o processo recomeça, calculando-se as distâncias novamente para classificar mais uma vez os vetores, agora baseando-se nos novos centróides encontrados. A clusterização termina quando não houver modificações na classificação dos vetores quando os centróides forem recalculados.

Após aplicar o *K-means*, obtemos os N centróides $\mathbf{c}_\zeta(i)$ de dimensão $M-1$, e, similarmente, o quadrado das distâncias Euclidianas entre cada estimativa $\zeta_i(k)$ e o respectivo centróide $\mathbf{c}_\zeta(i)$, que é independente da frequência. Utilizar outras medidas que não a Euclidiana não melhoraram o desempenho do algoritmo, por isso utilizamos a distância Euclidiana. Agora resta resolver a permutação. Ora, após a convergência do algoritmo *K-means*, obtemos uma matriz $\mathcal{D}$ formada por NK_{lim} vetores linha $\mathfrak{d}_{ik}$, cada um com comprimento N. Cada um destes vetores contém a

distância do TDOA na frequência k da fonte i estimada a cada um dos centróides $\mathbf{c}_\zeta(i)$:

$$\mathfrak{d}_{ik} = \begin{bmatrix} \|\zeta_i(k) - \mathbf{c}_\zeta(1)\|^2 & \|\zeta_i(k) - \mathbf{c}_\zeta(2)\|^2 & \cdots & \|\zeta_i(k) - \mathbf{c}_\zeta(N)\|^2 \end{bmatrix} \tag{4.24}$$

Essa matriz pode ser separada em K matrizes $\mathcal{D}_k$ de comprimento $N \times N$, cujas linhas são as fontes permutadas e as colunas são as fontes reais (os centróides):

$$\mathcal{D}_k = \begin{bmatrix} \|\zeta_1(k) - \mathbf{c}_\zeta(1)\|^2 & \|\zeta_1(k) - \mathbf{c}_\zeta(2)\|^2 & \cdots & \|\zeta_1(k) - \mathbf{c}_\zeta(N)\|^2 \\ \|\zeta_2(k) - \mathbf{c}_\zeta(1)\|^2 & \|\zeta_2(k) - \mathbf{c}_\zeta(2)\|^2 & \cdots & \|\zeta_2(k) - \mathbf{c}_\zeta(N)\|^2 \\ \vdots & \vdots & \ddots & \vdots \\ \|\zeta_N(k) - \mathbf{c}_\zeta(1)\|^2 & \|\zeta_N(k) - \mathbf{c}_\zeta(2)\|^2 & \cdots & \|\zeta_N(k) - \mathbf{c}_\zeta(N)\|^2 \end{bmatrix} \tag{4.25}$$

Cada elemento contém a distância entre uma das fontes permutadas e as fontes reais, então agora é preciso descobrir a correlação entre estas frequências. A tabela 4.1 mostra dois exemplos, para duas frequências diferentes, de distâncias encontradas, no caso $N = 3$ e $M = 3$. Em [4], o autor propõe uma heurística para escolher estas fontes. Ele primeiro escolhe a menor distância da matriz $\mathcal{D}_k$ e atribui esta linha à fonte real da coluna correspondente. Na Tabela 4.1 à esquerda, na frequência $922Hz$, por exemplo, a distância $0,019$ é a menor, portanto, atribuímos a linha 1 da respectiva raia de frequência à fonte 3. Depois, ele elimina a respectiva linha e coluna da escolha, nesse caso, a primeira linha e terceira coluna. O próximo passo é escolher a próxima menor distância, com as linhas e colunas que sobraram. A próxima menor é $0,035$, e, da mesma forma, a segunda linha é atribuída à segunda fonte. Procede-se desta forma até que todas as permutações tenham sido resolvidas. Nesse primeiro caso, a matriz de permutação obtida é $\mathbf{P} = \{3 \quad 2 \quad 1\}^T$.

Tabela 4.1: Exemplo das distâncias $\|\zeta_i(k) - \mathbf{c}_\zeta(i)\|^2$ entre centróides e vetores com estimativas dos TDOAs. Os números em negrito representam os valores escolhidos pela heurística apresentada no texto.

	$\mathbf{c}_\zeta(1)$	$\mathbf{c}_\zeta(2)$	$\mathbf{c}_\zeta(3)$
$\zeta_1(f_k = 922Hz)$	$0,135$	$0,237$	$\mathbf{0,019}$
$\zeta_2(f_k = 922Hz)$	$0,620$	$\mathbf{0,035}$	$0,119$
$\zeta_3(f_k = 922Hz)$	$\mathbf{0,063}$	$0,760$	$0,397$

	$\mathbf{c}_\zeta(1)$	$\mathbf{c}_\zeta(2)$	$\mathbf{c}_\zeta(3)$
$\zeta_1(f_k = 445Hz)$	$0,218$	$\mathbf{0,014}$	$0,035$
$\zeta_2(f_k = 445Hz)$	$\mathbf{0,013}$	$0,312$	$0,123$
$\zeta_3(f_k = 445Hz)$	$1,005$	$0,249$	$\mathbf{0,516}$

Nem sempre se obtém o melhor resultado com esta heurística. Um exemplo está na mesma Tabela 4.1 à direita, para a raia de frequência de $445Hz$. O resultado obtido está em negrito, ou seja, a matriz de permutação é $\mathbf{P} = \{2 \quad 1 \quad 3\}^T$. O problema é que o valor da soma das distâncias neste caso é $0,543$, o qual não é a solução ótima. A melhor solução seria $\mathbf{P} = \{3 \quad 1 \quad 2\}^T$, pois a soma das distâncias

é $0,297$.

Uma forma alternativa de realizar a clusterização é utilizar um algoritmo similar ao *K-means*, mas modificado para atender às nossas necessidades. Embora o *K-means* encontre o número de clusters certo, ele não considera uma restrição importante: que o número de TDOAs por cluster deve ser igual, e que em cada raia de frequência, só pode haver uma amostra de TDOA para cada cluster. Como ele não leva isso em consideração, acaba encontrando clusters errados algumas vezes, como vimos quando realizamos alguns testes. Apresentaremos então, uma clusterização que considere as restrições do nosso problema específico.

Primeiramente vamos definir os centróides $\mathbf{c}_\zeta(i)$ como:

$$\mathbf{c}_\zeta(i) = \frac{1}{K_{lim}} \sum_{k=1}^{K_{lim}} \mathfrak{d}_{ik}|_{i=\mathbf{P}_k(i)} \tag{4.26}$$

onde, novamente, o $|_{i=\mathbf{P}_k(i)}$ indica que a permutação não foi resolvida, e, portanto, o centróide não está corretamente posicionado por causa das permutações. O primeiro passo é calcular os N centróides iniciais segundo (4.26). Depois, em cada raia de frequência k, calculamos o respectivo vetor $\mathfrak{d}_{ik}$, várias vezes, uma para cada permutação possível. Se $N = 3$, por exemplo, há 3 permutações possíveis, então calculamos 3 vezes cada vetor, e depois escolhemos a permutação que obteve a menor soma. Ou seja:

$$\mathbf{P}_k = \text{argmin}_{P_k} \{|\mathfrak{d}_{ik}|_1 \mathbf{P}_k\} \tag{4.27}$$

onde $\text{argmin}_P f$ simboliza que estamos encontrando a função f para todos os valores de P e depois escolhendo o P para o qual o valor de f foi o *menor* possível. A matriz $\mathbf{P}$ é a matriz de permutação. O operador $|.|_1$ simboliza a norma-1 de um vetor, que é simplesmente a soma dos módulos dos elementos deste. Após realizar a operação (4.27) para cada raia de frequência, calculamos novamente os centróides $\mathbf{c}_\zeta(i)$, segundo (4.26), de acordo com as novas permutações, e repetimos esses dois passos até a convergência, que acontece quando não houver mais mudanças nas matrizes de permutação $\mathbf{P}_k$. Para referências futuras, chamaremos este algoritmo de *TDOAclust*, e o método que utiliza o *K-means* e a heurística de *TDOAKmeans*. Comparamos os dois e obtemos os resultados mostrados na Tabela 4.2, onde fica clara a vantagem do TDOAclust sobre o TDOAKmeans.

4.2 Correlação Espectral

Uma outra abordagem para tentar resolver o problema da permutação é supor que um mesmo sinal de voz possui alguma correlação entre raias de frequência. A Figura 4.13 mostra um espectrograma de um sinal de voz com duração de 6 segundos.

Tabela 4.2: Comparação entre os métodos de otimização TDOAclust e TDOAKmeans, para 3 fontes e 3 misturas, com tempo de reverberação 150 ms. Foram utilizados $K = 4096$ e $L = 2048$. O resultado é a média de 10 realizações.

Algoritmo	SIR médio	SDR médio	SAR médio
TDOAclust	$19,9$ dB	$14,5$ dB	$17,1$ dB
TDOAKmeans	$11,8$ dB	$8,9$ dB	$11,0$ dB

Um espectrograma é uma representação espectral (no domínio da frequência) de um sinal que mostra como a densidade de potência deste varia com o tempo. O eixo das abcissas mostra o tempo, e o eixo das ordenadas mostra a frequência, e a cor mostra a densidade de potência (quanto mais claro, maior a densidade de potência). Ele é obtido encontrando-se a densidade de potência de cada elemento de $\mathbf{X}(m)$ (que é um vetor de números complexos), obtido por (3.2), e formando uma coluna do espectrograma. Fazendo-se isso para todo m, encontramos sua variação ao longo do tempo. Obviamente, uma STFT de um espectrograma deve ter $J = L$, i.e, o salto tem o mesmo comprimento que a janela da STFT, para que não haja sobreposição, e a figura do espectrograma seja consistente. A informação da fase do sinal (o ângulo dos sinais complexos) é perdida no espectrograma, mas a fase em sinais de áudio não é muito importante, e portanto um espectrograma de "fase" não contém nenhuma informação importante.

Percebe-se nesta figura, que existem similaridades entre frequências, mesmo entre frequências distantes, e esta informação pode ser utilizada para tentar corrigir o problema da permutação. As similaridades nítidas são entre frequências adjacentes e entre harmônicos. Utilizou-se a densidade de potência em cada frequência, calculada em cada *frame* como $\mathbf{X}(m) \circ \mathbf{X}^*(m)$, onde $\circ$ é o produto Hadamard, assim como explicado na Seção 2.4. Nada impede de se utilizarem outras medidas, como o módulo de cada elemento de $\mathbf{X}(m)$, como veremos a seguir.

Em [58], o autor utiliza a modulação de amplitude (AM) dos sinais na frequência, que nada mais é do que o envelope de cada raia da frequência, i.e, cada linha do espectrograma mostrado na Figura 4.13. Neste caso, o envelope é a magnitude dos sinais complexos no domínio da frequência. O autor chama este envelope de envelope AM porque cada raia pode ser considerada como uma frequência portadora e a variação de amplitude em cima desta frequência é o envelope AM, assim como em transmissão AM. Aqui, chamaremos simplesmente de envelope, assim como em [75, 76]. Para comparação, mostramos o envelope de algumas frequências específicas na Figura 4.14. É notória a grande similaridade entre frequências adjacentes ($429,7$ Hz e $437,5$ Hz), e uma similaridade mais sutil entre frequências harmônicas ($632,8$ Hz e 1266 Hz).

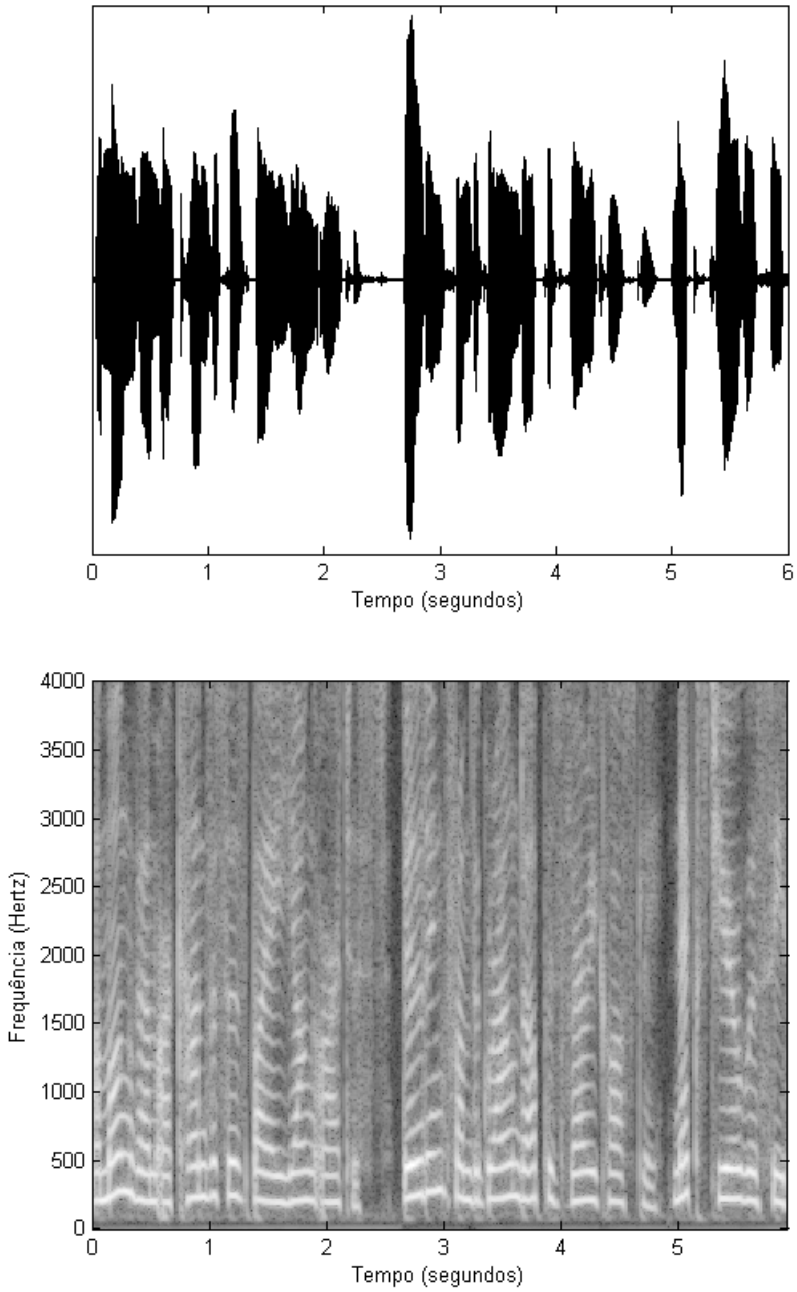

Figura 4.13: Espectrograma de um sinal de voz de 6 segundos, em comparação com sua representação no domínio do tempo. O espectrograma está numa escala logarítmica e foi escalado, para melhor visualização. Foram utilizados $K = 1024$, $L = 512$ e $J = 128$ com uma janela de Hanning.

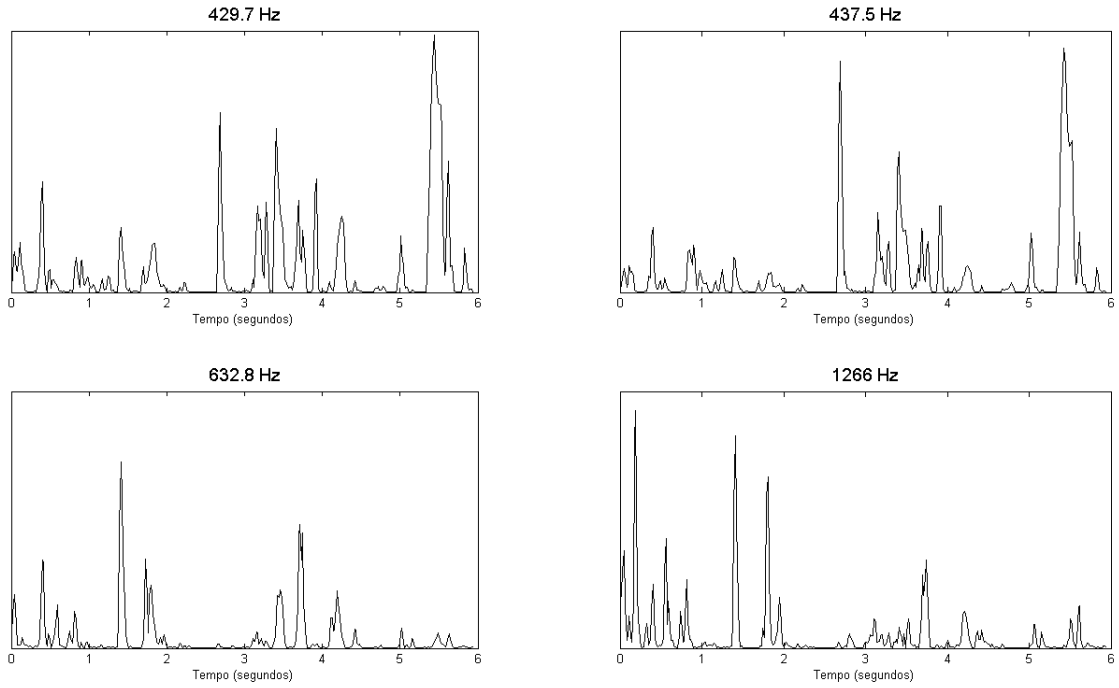

Figura 4.14: Envelope de um sinal de voz de 6 segundos, nas frequências adjacentes $429,7\,\text{Hz}$ e $437,5\,\text{Hz}$, na frequência $632,8\,\text{Hz}$ e sua harmônica $1266Hz$. Foram utilizados $K = 1024$, $L = 512$ e $J = 128$ com uma janela de Hanning.

O primeiro passo é decidir qual a medida de correlação que será utilizada. No artigo [58], citado anteriormente, o autor utiliza a covariância entre os envelopes das duas raias de frequência, segundo a expressão (2.34), considerando que cada uma das raias é uma variável aleatória, e cada um dos instantes m representa uma observação, e na prática, é calculada por (2.35). O problema de utilizar a covariância é que ela é dependente da potência dos sinais, o que pode gerar resultados falsos. Por exemplo, suponhamos o caso de duas fontes s_1 e s_2, onde $\sigma_{s_1}^2 \gg \sigma_{s_2}^2$. Ora, se a variância de s_1 é muito maior, então, $\text{cov}_{s_1 s_2} > \text{cov}_{s_2 s_2}$, e isto é exatamente o contrário do que deveria ocorrer. Utilizando covariância, corremos o risco de escolher erradamente as permutações. Claro que normalizar os sinais em cada raia de frequência resolve este problema, mas é mais robusto utilizar a medida de correlação, que não adiciona praticamente nenhum custo computacional, segundo a expressão (2.38). Por definição, como dito na Seção 2.4, este valor varia entre -1 e 1, sendo que 1 significa que a similaridade entre os sinais é máxima. Esta medida é utilizada em [55, 59, 71, 77, 78].

A Figura 4.15 mostra um gráfico da correlação entre pares de frequência de um mesmo locutor. Obviamente, a correlação é máxima (1) na diagonal principal da figura, que corresponde à mesma frequência. Mas ao redor da diagonal principal, há uma faixa onde a correlação ainda é alta, como tínhamos observado anteriormente. A matriz (4.28) apresenta alguns valores retirados da Figura 4.15, onde a diagonal

principal está em negrito. Como foram utilizados $K = 4096$ pontos na FFT, a resolução de frequência é $\Delta_{min} f = \frac{f_s}{K} = 1,95\,\text{Hz}$. Para uma diferença de $\Delta f = \pm 6\,\text{Hz}$, a correlação ainda fica acima de $0,5$. Também pode-se observar na figura algumas "retas" de correlação mais alta com inclinação diferente da diagonal principal. Estas "retas" representam as correlações harmônicas, inerentes a um sinal de voz, que também podem ser utilizadas para diferenciar locutores.

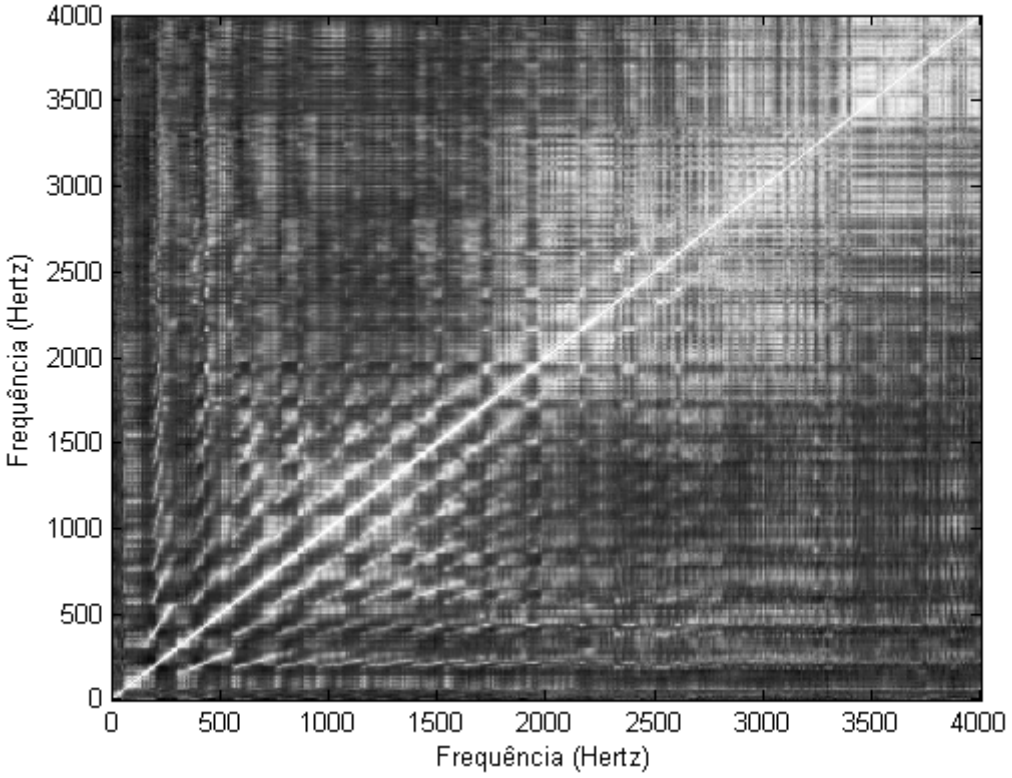

Figura 4.15: Correlação entre frequências de um *mesmo* locutor. A correlação foi escalada de forma que o branco correspondesse a 1 e o preto a -0.4. Foram utilizados $K = 4096$, $L = 2048$ e $J = 512$ com uma janela de Hanning.

$$\begin{bmatrix} \mathbf{1,0000} & 0,9656 & 0,8704 & 0,6742 & 0,4376 & 0,3186 & 0,2176 \\ 0,9656 & \mathbf{1,0000} & 0,9536 & 0,7825 & 0,5527 & 0,4269 & 0,3083 \\ 0,8704 & 0,9536 & \mathbf{1,0000} & 0,9230 & 0,7401 & 0,6168 & 0,4819 \\ 0,6742 & 0,7825 & 0,9230 & \mathbf{1,0000} & 0,9343 & 0,8273 & 0,6758 \\ 0,4376 & 0,5527 & 0,7401 & 0,9343 & \mathbf{1,0000} & 0,9461 & 0,8126 \\ 0,3186 & 0,4269 & 0,6168 & 0,8273 & 0,9461 & \mathbf{1,0000} & 0,9499 \\ 0,2176 & 0,3083 & 0,4819 & 0,6758 & 0,8126 & 0,9499 & \mathbf{1,0000} \end{bmatrix} \tag{4.28}$$

Para comparação, na Figura 4.16 temos a correlação entre pares de frequência de locutores diferentes. Agora, diferente do caso anterior, a correlação é baixa na maioria das frequências, e não se vê uma diagonal principal ou "retas" com diferentes inclinações como visto na Figura 4.15. A matriz (4.29) apresenta alguns valores

retirados da Figura 4.15, onde a diagonal principal está em negrito. Os valores são, na sua maioria, negativos ou próximos de zero, o que qualifica a correlação entre frequências como uma forma de diferenciar locutores diferentes.

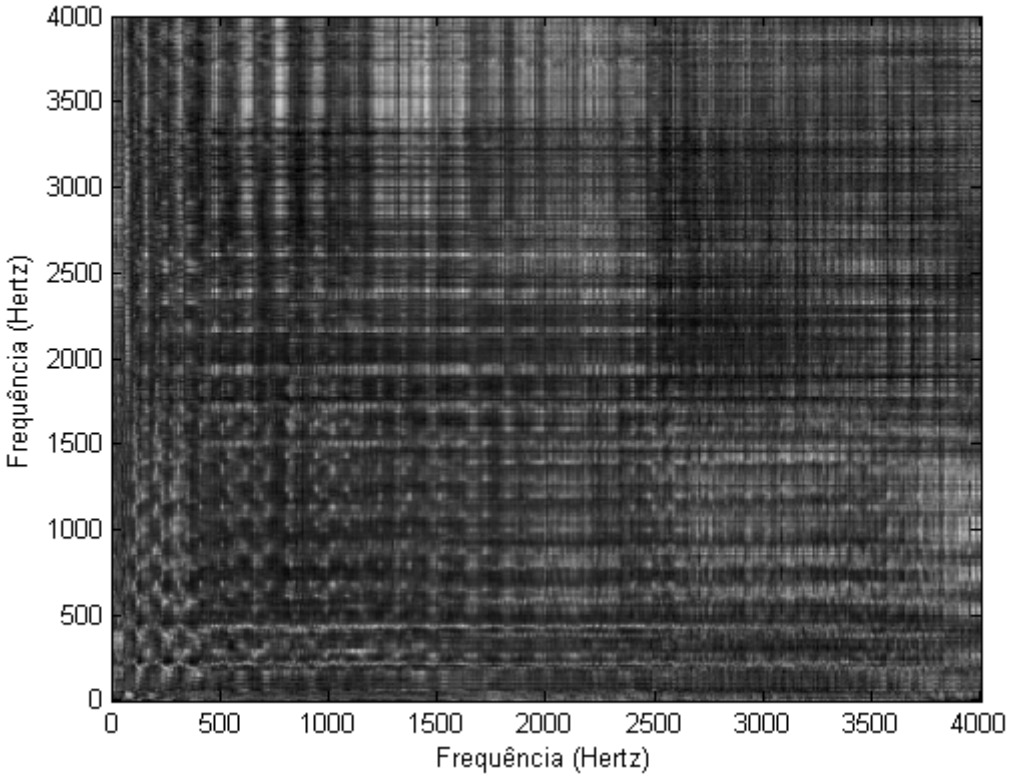

Figura 4.16: Correlação entre frequências de locutores *diferentes*. A correlação foi escalada de forma que o branco correspondesse a 1 e o preto a -0.4. Foram utilizados $K = 4096$, $L = 2048$ e $J = 512$ com uma janela de Hanning.

$$\begin{bmatrix} \mathbf{-0,2421} & -0,2366 & -0,1967 & -0,1752 & -0,1897 & -0,1834 & -0,1572 \\ -0,2530 & \mathbf{-0,2432} & -0,2013 & -0,1792 & -0,1917 & -0,1814 & -0,1540 \\ -0,2829 & -0,2737 & \mathbf{-0,2327} & -0,2070 & -0,2131 & -0,1988 & -0,1707 \\ -0,2789 & -0,2718 & -0,2404 & \mathbf{-0,2202} & -0,2234 & -0,2093 & -0,1853 \\ -0,2401 & -0,2319 & -0,2119 & -0,2081 & \mathbf{-0,2182} & -0,2064 & -0,1882 \\ -0,2361 & -0,2233 & -0,1998 & -0,2035 & -0,2255 & \mathbf{-0,2162} & -0,1978 \\ -0,1916 & -0,1731 & -0,1444 & -0,1495 & -0,1772 & -0,1684 & \mathbf{-0,1513} \end{bmatrix} \tag{4.29}$$

Não é obrigatório utilizar envelopes de frequência com a magnitude dos sinais. De uma forma geral, a correlação pode ser aplicada a um envelope qualquer dos sinais, ou seja:

$$r(\upsilon_x, \upsilon_y) = \frac{\text{cov}_{\upsilon_x \upsilon_y}}{\sqrt{\sigma_{\upsilon_x} \sigma_{\upsilon_y}}} \tag{4.30}$$

onde, no caso da magnitude:

$$\upsilon_{y_i}(m) = |y_i(m)| \tag{4.31}$$

Em [79], o autor propõe uma medida diferente, a qual ele chama de *powRatio*, utilizada ao invés da magnitude, e dada por (4.32), que é aplicada a cada raia de frequência k. Esta medida, por definição, está contida no intervalo $[0, 1]$, e é próxima de 1 se o i-ésimo termo $\mathbf{a}_i y_i(m)$ for dominante em relação aos outros termos $\mathbf{a}_{i'} y_{i'}(m), \forall i' \neq i$, e é zero onde os outros termos são dominantes.

$$\upsilon_{y_i}(m) = powRatio_i(m) = \frac{\|\mathbf{a}_i y_i(m)\|^2}{\sum_{i'=1}^{N} \|\mathbf{a}_{i'} y_{i'}(m)\|^2} \tag{4.32}$$

É importante notar que, na expressão (4.32) o vetor $\mathbf{a}_i$ simboliza a resposta de frequência do caminho entre a fonte i e *todos* os sensores j, onde cada elemento corresponde a um sensor. Encontrar $\|\mathbf{a}_i y_i(m)\|^2$, portanto, significa encontrar a soma das energias da fonte y_i como vista em cada um dos M sensores. Sua vantagem em relação à medida simples de magnitude é que ela aproveita melhor a esparsidade dos sinais, o que é comum em sinais de voz misturados. Em geral, a componente de um sinal de voz em determinado instante de tempo é muito maior do que a dos outros sinais, como pode ser visto na Figura 4.17, que mostra o envelope em (4.32) aplicado a duas fontes já separadas, onde as permutações foram resolvidas de um modo supervisionado (ver Apêndice C). Duas importantes características podem ser verificadas nesta figura. A primeira, é que a medida é limitada, e os sinais ativos são representados como valores próximos de 1, mesmo que a potência deles seja baixa. Um sinal ativo é o sinal de voz do locutor que está falando no instante considerado. A segunda, é que os valores das fontes são exclusivos, i.e, se $powRatio_1(m)$ é próximo de 1, então com certeza $powRatio_2(m)$ é próximo de 0, para o caso de duas fontes. Isto pode ser facilmente estendido para N fontes.

Para verificar se esta medida pode melhorar o desempenho do algoritmo, plotamos, na Figura 4.18, a correlação entre pares de frequência do mesmo locutor, e na Figura 4.19, a correlação entre pares de frequência de locutores diferentes, utilizando o envelope *powRatio*. Comparando estas figuras com as Figuras 4.15 e 4.16, é notável a diferença que, para uma mesma fonte, a correlação entre frequências é maior de uma forma geral e, entre fontes diferentes, a correlação é bem menor. Uma observação curiosa é que, utilizando o *powRatio* não se vêem mais as retas que correspondem às correlações harmônicas, como utilizando a magnitude. Isto significa que, dependendo do envelope utilizado, a gama de frequências nas quais a correlação é uma boa medida de similaridade muda.

Após escolher qual medida de correlação será utilizada, resta escolher o algoritmo

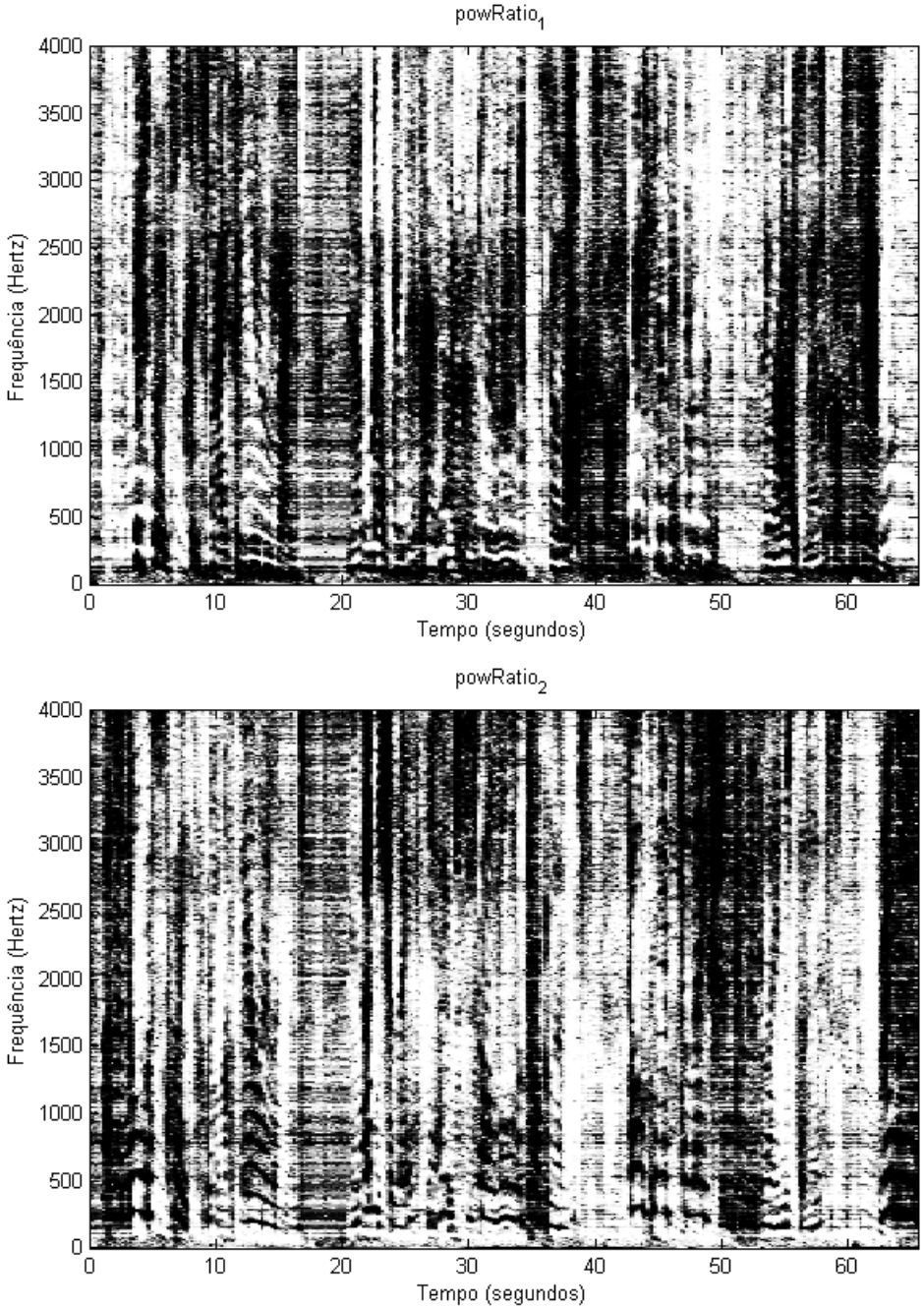

Figura 4.17: Espectro de frequência do envelope *powRatio* de duas fontes, após a separação.

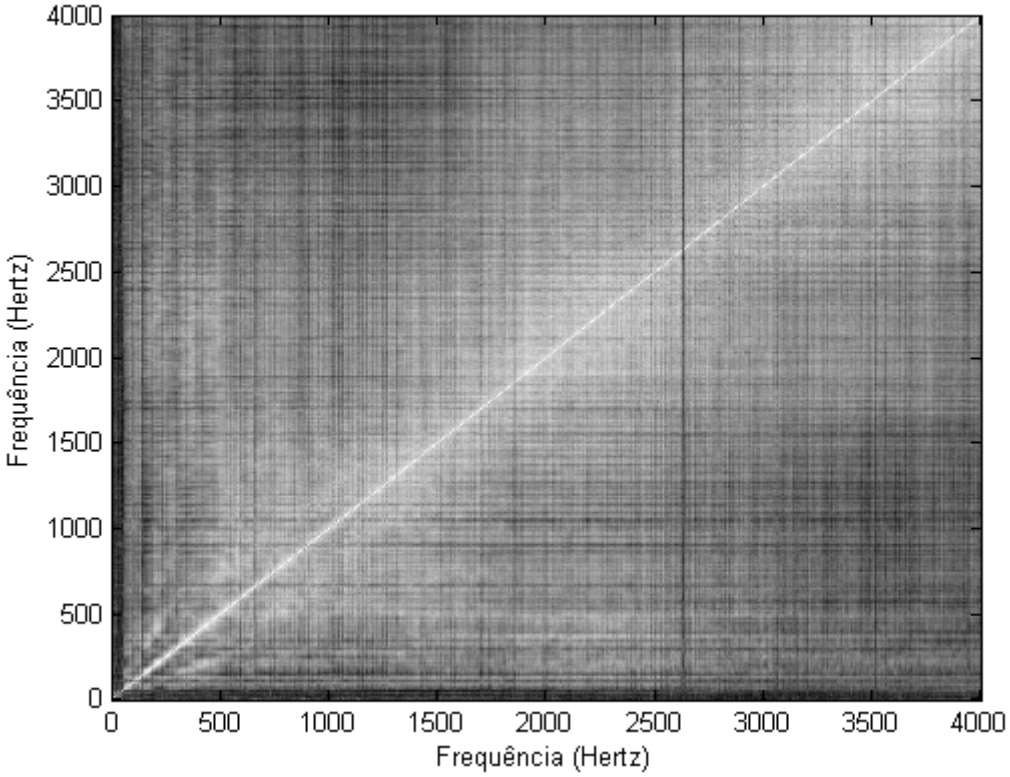

Figura 4.18: Correlação entre envelopes *powRatio* de frequências de um *mesmo* locutor. A correlação foi escalada de forma que o branco correspondesse a 1 e o preto a -0.4. Foram utilizados $K = 4096$, $L = 2048$ e $J = 512$ com uma janela de Hanning.

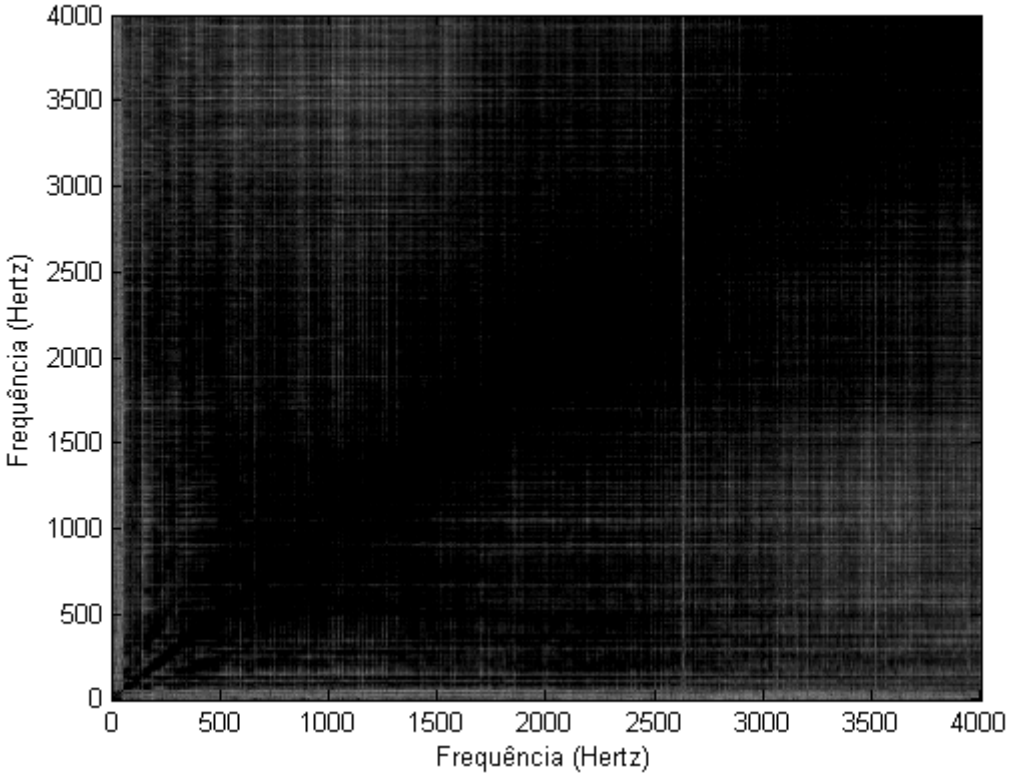

Figura 4.19: Correlação entre entre envelopes *powRatio* de frequências de locutores *diferentes*. A correlação foi escalada de forma que o branco correspondesse a 1 e o preto a -0.4. Foram utilizados $K = 4096$, $L = 2048$ e $J = 512$ com uma janela de Hanning.

de otimização. A primeira abordagem consiste em realizar uma otimização global baseada nas correlações. Isto significa calcular um envelope global para cada fonte e comparar a fonte permutada de cada frequência com esse envelope global para decidir se há permutação na frequência específica ou não. Utilizaremos um algoritmo de clusterização similar ao utilizado na Seção 4.1.3, com a diferença que as correlações serão a medida de distância. Primeiro obtemos N envelopes globais (centróides) de todas as frequências de uma determinada fonte:

$$\mathbf{c}_{\upsilon_i} = \frac{1}{K}\sum_{k=1}^{K} r(\boldsymbol{\upsilon}_{ik}) \tag{4.33}$$

onde $\mathbf{c}_{\upsilon_i}$ é um vetor linha com todas as N_{fsamp} amostras do centróide, e $\boldsymbol{\upsilon}_{ik}$ é um vetor linha com todas as N_{fsamp} amostras do envelope da fonte i na frequência k. Depois, escolhe-se a permutação em cada frequência através da correlação entre o envelope dessa frequência de determinada fonte i_P (onde o problema da permutação ainda não foi resolvido) e o envelope global, encontrado anteriormente. Isto é,

$$\mathbf{P}_k = \text{argmax}_{P_k}\left\{\sum_{i=1}^{N} r(\boldsymbol{\upsilon}_{\mathbf{P}(i)k}, \mathbf{c}_{\upsilon_i})\right\} \tag{4.34}$$

onde $\text{argmax}_P\{f\}$ simboliza que estamos encontrando a função f para todos os valores de P e depois escolhendo o P para o qual o valor de f foi o *maior* possível, e $\mathbf{P}(i)$ simboliza a fonte i permutada pela matriz $\mathbf{P}$. Em cada frequência, o envelope que obtiver maior correlação com um centróide de determinada fonte i deve pertencer a esta fonte i. Depois de decididas as permutações $\mathbf{P}_k$ em cada raia de frequência, o envelope global em (4.33) é calculado novamente, e estas duas operações são repetidas até a convergência. Esse algoritmo será chamado de *GlobalCorr*.

Outro algoritmo de otimização que pode ser utilizado consiste em maximizar a soma das correlações entre um grupo selecionado de frequências, ou seja:

$$\mathbf{P}_k = \text{argmax}_{P_k}\left\{\sum_{k'\in\mathcal{G}(f)}\sum_{i=1}^{N} r(\boldsymbol{\upsilon}_{\mathbf{P}(i)k}, \boldsymbol{\upsilon}_{\mathbf{P}(i)k'})\right\} \tag{4.35}$$

onde o grupo $\mathcal{G}(f)$ deve consistir de frequências que tenham uma correlação alta entre si, para uma mesma fonte. Como vimos anteriormente, as frequências adjacentes e harmônicas possuem uma boa correlação entre si. Se chamarmos o grupo de frequências adjacentes $\mathcal{A}(f)$ e o grupo de frequências harmônicas $\mathcal{H}(f)$, então $\mathcal{G}(f) = \mathcal{A}(f) \cup \mathcal{H}(f)$. Utilizamos:

$$\mathcal{A}(f) = \{f - 3\Delta f, f - 2\Delta f, f - \Delta f, f + \Delta f, f + 2\Delta f, f + 3\Delta f\} \tag{4.36}$$

$$\mathcal{H}(f) = \{\text{ceil}(f/2 - \Delta f), \text{ceil}(f/2), \text{ceil}(f/2 + \Delta f), 2f - \Delta f, 2f, 2f + \Delta f\} \tag{4.37}$$

onde $\Delta f = \frac{f_s}{K}$. Os valores acima estão em Hertz, e é necessário indexá-los, referenciando cada frequência f a um índice k. A maximização de (4.35) acontece com uma frequência k de cada vez. Torna-se importante, então, definir a ordem em que as frequências serão atualizadas. Definiremos aqui três formas diferentes de atualizar as frequências.

A primeira atualiza começando[3] com $k = 0$ até $k = K - 1$. Alinha-se a frequência k e depois a $k + 1$, notando-se que a alteração da permutação $\mathbf{P}_k$ altera a soma em (4.35) para a raia de frequência $k + 1$. Após varrer todas as frequências, inicia-se a segunda iteração, e todas as frequências são varridas novamente. Continua-se até que não haja alteração em nenhuma $\mathbf{P}_k, k = 0, \cdots, K - 1$ para uma dada iteração em relação a iteração anterior. Esse algoritmo será chamado de *LocalCorr*.

A segunda forma introduz uma nova equação. Adicionalmente a encontrar a matriz $\mathbf{P}_k$, também encontramos o $r_{max}(k)$ para cada frequência, obtido da seguinte forma:

$$r_{max}(k) = \sum_{k' \in \mathcal{G}(f)} \sum_{i=1}^{N} r(\upsilon_{\mathbf{P}(i)k}, \upsilon_{\mathbf{P}(i)k'})|_{\mathbf{P}=\mathbf{P}_k} \tag{4.38}$$

após permutar as fontes i de acordo com a matriz $\mathbf{P}_k$ encontrada. Primeiro encontramos a matriz $\mathbf{P}_k$ para todas as frequências sem, no entanto, alterar *nenhuma* permutação, como feito no caso anterior. Ou seja, alinha-se a frequência k e depois a $k + 1$, sem considerar a permutação $\mathbf{P}_k$ encontrada. Depois comparamos os valores de $r_{max}(k)$ e alinhamos *somente* a permutação da frequência k onde $r_{max}(k)$ foi máximo. Nesta frequência a permutação é classificada como *confiável*, e a maximização (4.35) não é realizada nesta frequência. Na segunda iteração, portanto, encontramos a matriz $\mathbf{P}_k$ para as $K - 1$ frequências restantes. O processo se repete até que todas as frequências sejam classificadas como confiáveis. Representando matematicamente, existe um conjunto $\mathcal{F}_{conf}$ que contém todas as raias de frequência k consideradas como confiáveis. Esse algoritmo será chamado de *ConjCorr*.

A terceira forma é muito parecida com a segunda (ConjCorr), com a diferença que é dividido em três etapas. A primeira etapa considera $\mathcal{G}(f) = \mathcal{A}(f)$, i.e, somente a correlação entre frequências adjacentes é calculada, e estabelece um limite th_{adj} para $r_{max}(k)$. Se o maior valor de $r_{max}(k)$ encontrado em uma determinada iteração for menor que esse limite, então a primeira etapa termina. Em geral grande parte das permutações já estarão alinhadas (claro, com uma boa escolha de th_{adj}), e foram classificadas como confiáveis, e partimos para a segunda etapa. Como agora $\mathcal{G}(f) = \mathcal{A}(f)$, então somente a correlação entre frequências harmônicas é calculada, e outro limite th_{harm} é estabelecido. Esta segunda etapa é mais parecida com o al-

[3]Na verdade, mesmo que a permutação em $k = 0$ não esteja alinhada, isso pouco altera os resultados, porque modificar a componente DC de um sinal de áudio não têm influência nem sobre a forma como ouvimos nem sobre o desempenho de algoritmos de reconhecimento de fala, por exemplo. Então, pode-se começar de $k = 1$ e deixar a componente DC desalinhada.

goritmo LocalCorr. As matrizes $\mathbf{P}_k$ são encontradas para todas as frequências ainda não confiáveis, mas somente nas frequências onde $r_{max}(k) > th_{harm}$ as permutações são alinhadas, e estas frequências são classificadas como confiáveis. Após varrer todas as frequências somente uma vez, segue-se a última etapa. A terceira e última etapa é quase idêntica à primeira, com a única diferença de que $th_{adj} = 0$, ou seja, não há limite estabelecido. Este algoritmo será chamado de *HarmCorr*. A justificativa para o HarmCorr, segundo [71], é que utilizar as frequências harmônicas para calcular as correlações só funciona se a maioria das permutações já estiver resolvida.

A Tabela 4.3 mostra os resultados de testes realizados com diferentes métodos de correlação para alinhamento das permutações, e com o método supervisionado. Nota-se que os métodos ConjCorr, HarmCorr e LocalCorr não apresentam resultados muito bons. Isso acontece por causa da falta de robustez dos algoritmos que utilizam correlação. Uma permutação desalinhada em uma frequência acaba impactando várias outras frequências. Por este motivo, normalmente se utiliza o DOA antes da correlação, como será visto na Seção 4.3. Outra forma de pré-alinhar as permutações para evitar o problema citado é utilizar o GlobalCorr, que alinha todas as frequências de uma vez só. Nota-se na tabela que este método obteve resultados satisfatórios sem precisar do DOA. Pode-se também integrar os métodos GlobalCorr e LocalCorr. Inicialmente as permutações são pré-alinhadas com o algoritmo GlobalCorr, obtendo-se uma matriz de permutação para cada frequência. Em seguida, utilizando esta matriz obtida como matriz inicial, aplica-se o LocalCorr, e percebe-se uma melhora significativa. Na verdade, o desempenho foi similar ao do método supervisionado, o que é impressionante. Com relação ao tipo de envelope utilizado, nota-se a superioridade do envelope powRatio em relação ao módulo, como já foi discutido anteriormente.

À primeira vista, pode-se pensar que outros métodos de correlação, além do LocalCorr, podem também ser utilizados após o GlobalCorr, mas isto não é verdade. Os métodos ConjCorr e HarmCorr são diferentes do LocalCorr, no sentido de que a cada iteração o LocalCorr realinha todas as permutações de todas as frequências, enquanto os dois primeiros só realinham uma permutação de uma frequência a cada iteração, que não é alinhada novamente depois (a frequência passa a pertencer ao conjunto $\mathcal{F}_{conf}$). Ou seja, para utilizar o pré-alinhamento do GlobalCorr de forma eficiente nestes dois métodos, algumas frequências devem ser classificadas como confiáveis, e colocadas no conjunto $\mathcal{F}_{conf}$, e estas frequências não serão mais realinhadas. Podemos classificar estas frequências baseado na correlação entre elas e o respectivo centróide, porém os resultados obtidos não foram encorajadores. Analisando do ponto de vista de otimização, se considerarmos que cada um dos métodos consiste na minimização de uma função objetivo cujos parâmetros são as matrizes de permutação em cada frequência, o método LocalCorr converge para o

Tabela 4.3: Comparação dos diferentes métodos de correlação para alinhamento das permutações.

Número de fontes e misturas - $N = M = 3$ Tempo de reverberação - $T_{60} = 200$ms Número de raias da FFT - $K = 4096$ Tamanho da janela - $L = 2048$ Separação - Natural ICA com sign ($\eta = 0,1$) Número de realizações - 10 Disposição dos microfones e fontes - Figura A.3 Janela win_a - Retangular		
Método utilizado	Envelope	SIR médio
Supervisionado	N.A.	$24,1$ dB
ConjCorr	Módulo	$3,8$ dB
ConjCorr	powRatio	$3,7$ dB
HarmCorr	Módulo	$8,0$ dB
HarmCorr	powRatio	$8,5$ dB
GlobalCorr	Módulo	$12,2$ dB
GlobalCorr	powRatio	$16,0$ dB
LocalCorr	Módulo	$4,1$ dB
LocalCorr	powRatio	$3,6$ dB
GlobalCorr + LocalCorr	Módulo	$15,8$ dB
GlobalCorr + LocalCorr	powRatio	$24,0$ dB

mínimo, porém, se não for bem inicializado, ele acaba convergindo para um mínimo local. Já os métodos ConjCorr e HarmCorr, por causa da heurística inerente, não convergem da forma usual, como um algoritmo que utiliza gradiente, que atualiza gradativamente seus parâmetros na direção do mínimo mais próximo do ponto atual. Eles provavelmente "saltam" na função objetivo, não necessariamente na direção do mínimo mais próximo (talvez na direção do mínimo global, mas isso carece de uma análise mais profunda), assim como alguns métodos estatísticos. Isso permite que eles tenham um desempenho melhor, mas não conseguem convergir para o mínimo global, independentemente da inicialização. Esta foi apenas uma análise superficial para esclarecer a diferença entre os métodos; não é nosso foco realizar uma análise matemática mais profunda sobre a convergência destes algoritmos, o que é bem complexo, por sinal, por causa da não-linearidade dos parâmetros da função objetivo (as matrizes de permutação, que alteram a ordem das fontes), e da dificuldade de se analisar heurísticas.

4.3 Unindo Abordagens

É possível conjugar os métodos mostrados anteriormente, unindo suas vantagens. Os algoritmos baseados em localização, em geral, não são muito precisos, principalmente em ambientes reverberantes. Em compensação, são métodos robustos, no sentido de que, se errarmos o valor do DOA para uma frequência, isso não afetará frequências adjacentes. No caso do TDOA, onde calculamos os centróides, um valor calculado errado pode alterar o valor dos centróides, principalmente quando se utiliza a clusterização *K-means*. Entretanto, quando se utiliza a clusterização TDOAclust, se o número de valores errados não for muito grande, a classificação por frequência não será muito alterada.

Os algoritmos baseados em correlação entre frequências são algoritmos mais precisos, se as frequências forem bem escolhidas. Em compensação, uma escolha errada de permutação em uma frequência afeta todas as frequências adjacentes. São algoritmos pouco robustos, pois um alinhamento errado em uma frequência causa desalinhamentos consecutivos. Em [71], o autor propõe uma forma de unir as duas abordagens.

A idéia é primeiro utilizar o DOA para alinhar as permutações, porém, deve-se utilizar um critério para decidir se a permutação decidida pelo algoritmo é confiável ou não. Após decidir quais permutações alinhadas pelo DOA são confiáveis, as restantes são alinhadas por outro algoritmo. O autor indica o HarmCorr, explicado na Seção 4.2, mas outros podem ser utilizados. Há três critérios utilizados para decidir se a permutação decidida pelo algoritmo DOA é confiável ou não. O DOA de uma determinada frequência *não é* confiável se:

1. Não foi possível encontrá-lo segundo a expressão (4.19) para nenhuma combinação $[j, j']$;

2. Seu valor é muito diferente da média (ou mediana[4]) dos DOA encontrados para determinada fonte, i.e, $|\theta_k(i) - \bar{\theta}(i)| > th_{DOA}(i)$;

3. Os ângulos $\theta_k(i' \neq i)$ encontrados não são os mínimos do padrão de diretividade da fonte i, ou seja, para toda fonte i, $|\boldsymbol{F}_i(k, \theta_k(i))|^2 < \sum_{i' \neq i} |\boldsymbol{F}_i(k, \theta_k(i'))|^2$. Na prática, utilizamos a condição $\sum_{i=1}^{N} \left\{ 10 \log_{10}(|\boldsymbol{F}_i(k, \theta_k(i))|^2) - 10 \log_{10}(\sum_{i' \neq i} |\boldsymbol{F}_i(k, \theta_k(i'))|^2) \right\} < th_F$.

Se o DOA não se encaixar em nenhuma destas restrições, ele é considerado como confiável e a permutação é alinhada de acordo com a ordenação do DOA. As frequências não confiáveis são alinhadas, então, utilizando algum algoritmo de correlação citado anteriormente.

[4]Nos nossos testes, optamos por utilizar a mediana, por razões explicadas na Seção 4.1.2.

4.4 Simulações

Nesta seção realizaremos alguns testes finais, comparando todos os métodos para alinhamento das permutações. Resumimos aqui todos os algoritmos testados para alinhamento das permutações:

- DOA + GlobalCorr
- DOA + GlobalCorr + LocalCorr
- DOA + ConjCorr
- DOA + HarmCorr
- TDOAclust
- GlobalCorr + LocalCorr
- ConjCorr

Para os limites, foram utilizados os seguintes valores:

$$th_{DOA}(i) = 1,5 \times \mathsf{s}_{\theta(i)} \tag{4.39}$$

$$th_{F} = 0\,\text{dB} \tag{4.40}$$

$$th_{adj} = 30\% \times \max(r_{max}(f))|_{\mathcal{G}(f)=\mathcal{A}(f)} = 0,3 \times 6 \times N \tag{4.41}$$

$$th_{harm} = 10\% \times \max(r_{max}(f))|_{\mathcal{G}(f)=\mathcal{H}(f)} = 0,1 \times 6 \times N \tag{4.42}$$

onde $\max(r_{max}(f))$ são os valores máximos possíveis para (4.38), considerando os conjuntos $\mathcal{A}(f)$ e $\mathcal{H}(f)$. Estes valores podem ser facilmente calculados, pois sabemos que o valor máximo da correlação r é 1, e sabemos o número de frequências de cada conjunto. O desvio padrão $\mathsf{s}_{\theta(i)}$ é calculado variando-se a frequência k.

Primeiramente vamos confirmar os resultados da Seção 3.2 com relação à janela utilizada na STFT. A Tabela 4.4 mostra a comparação entre utilizar a janela de Hanning e utilizar a janela retangular. Podemos concluir que os métodos de alinhamento de permutação funcionam melhor com a janela retangular (resolução de frequência maior). Entretanto, a *separação* em si não é ótima utilizando a janela retangular (no método supervisionado, a janela de Hanning teve um melhor desempenho).

Os métodos de alinhamento da permutação foram realizados para vários tempos de reverberação diferentes, e as condições de teste estão na Tabela 4.5. As próximas figuras mostram a variação da SIR de vários métodos em função do tempo de reverberação T_{60} da sala. Quando $T_{60} = 0\,\text{ms}$, a sala é anecóica, ou seja, não há reverberação (embora ainda haja o atraso e atenuação do sinal). A Figura 4.20 compara os métodos DOA + ConjCorr, DOA + HarmCorr, e DOA + GlobalCorr +

Tabela 4.4: Comparação da SIR utilizando a janela de Hanning ou a retangular na transformação para o domínio da frequência. O método de resolver a permutação foi variado. Foi utilizado um salto $J = \frac{L}{4}$ para ambas as janelas.

Número de fontes e misturas - $N = M = 3$ Tempo de reverberação - $T_{60} = 150\,\text{ms}$ Número de raias da FFT - $K = 4096$ Tamanho da janela - $L = 2048$ Separação - Natural ICA com sign ($\eta = 0,1$) Número de realizações - 10 Disposição dos microfones e fontes - Figura A.3		
Método para resolver a permutação	Hanning	Retangular
Supervisionado	$28,2\,\text{dB}$	$25,7\,\text{dB}$
TDOAclust	$19,9\,\text{dB}$	$20,4\,\text{dB}$
DOA + ConjCorr	$16,8\,\text{dB}$	$18,5\,\text{dB}$
DOA + HarmCorr	$16,6\,\text{dB}$	$16,2\,\text{dB}$
DOA + GlobalCorr	$16,2\,\text{dB}$	$17,6\,\text{dB}$
DOA + GlobalCorr + LocalCorr	$16,9\,\text{dB}$	$18,5\,\text{dB}$
ConjCorr	$3,2\,\text{dB}$	$4,0\,\text{dB}$
GlobalCorr + LocalCorr	$20,6\,\text{dB}$	$24,5\,\text{dB}$

LocalCorr, todos baseados em localização seguida de correlação (o método robusto e preciso). Percebe-se que o desempenho deles é similar, com uma ligeira vantagem do método DOA + GlobalCorr + LocalCorr, principalmente quando o tempo de reverberação é maior. Em tempos de reverberação pequenos, o DOA + ConjCorr apresenta um desempenho melhor. Os resultados de utilizar somente a correlação, como era de se esperar, são muito ruins, e não variam muito com o tempo de reverberação. Eles são mostrados na Figura 4.21. A Figura 4.22 compara o método DOA + GlobalCorr + LocalCorr quando utilizado um arranjo em *cluster* e um arranjo em linha. Como esperado, o arranjo em linha obtém um desempenho melhor, pois a teoria do DOA foi formulada tendo como suposição um arranjo deste tipo. Finalmente, a Figura 4.23 compara os métodos TDOA (o melhor dentre os métodos de localização das fontes), GlobalCorr + LocalCorr (melhor dentre os métodos de correlação espectral) e DOA + GlobalCorr + LocalCorr (o melhor dentre os métodos conjugados).

Observando o desempenho do método TDOA, fica clara a limitação em métodos que utilizam localização das fontes em ambientes muito reverberantes. Já o desempenho de métodos baseados em correlação espectral não varia muito com o aumento da reverberação. Com isso, em ambientes pouco reverberantes, o TDOA

Tabela 4.5: Condições dos testes dos métodos de alinhamento de permutação.

Número de fontes e misturas - $N = M = 3$
Tempo de reverberação - T_{60} = variável
Número de raias da FFT - $K = 4096$
Tamanho da janela - $L = 2048$
Separação - Natural ICA com sign ($\eta = 0, 1$)
Número de realizações - 10
Arranjo de microfones - Figuras A.2, A.3 e A.3 com o arranjo de microfones modificado para um arranjo em linha
Janela win_a - Retangular

teve o melhor desempenho dentre todos, mas em ambientes mais reverberantes, ele não produz resultados confiáveis, e o método GlobalCorr + LocalCorr superou todos os outros. O método DOA + GlobalCorr + LocalCorr, quando a reverberação é pequena, apresentou resultados piores do que o TDOA, mas, à medida que a reverberação aumenta, a parte precisa (correlação GlobalCorr + LocalCorr) do método começa a sobressair, e para tempos de reverberação altos, ele apresenta resultados melhores do que o TDOA. Nota-se que, à medida que aumenta a reverberação, o resultado ótimo (caso supervisionado) também piora, porque os filtros que representam o caminho entre a fonte e os sensores ficam maiores.

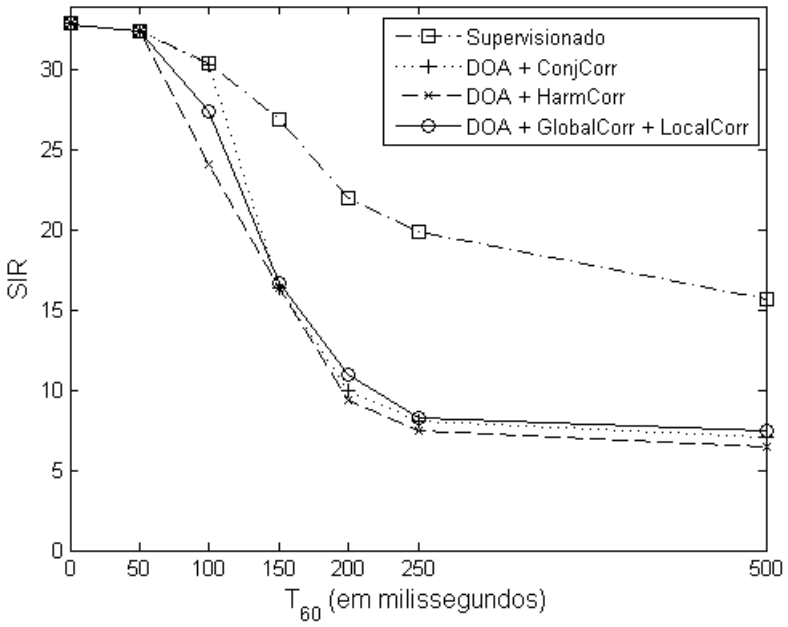

Figura 4.20: Comparação entre os métodos DOA + ConjCorr, DOA + HarmCorr, e DOA + GlobalCorr + LocalCorr, utilizando a disposição da Figura A.2.

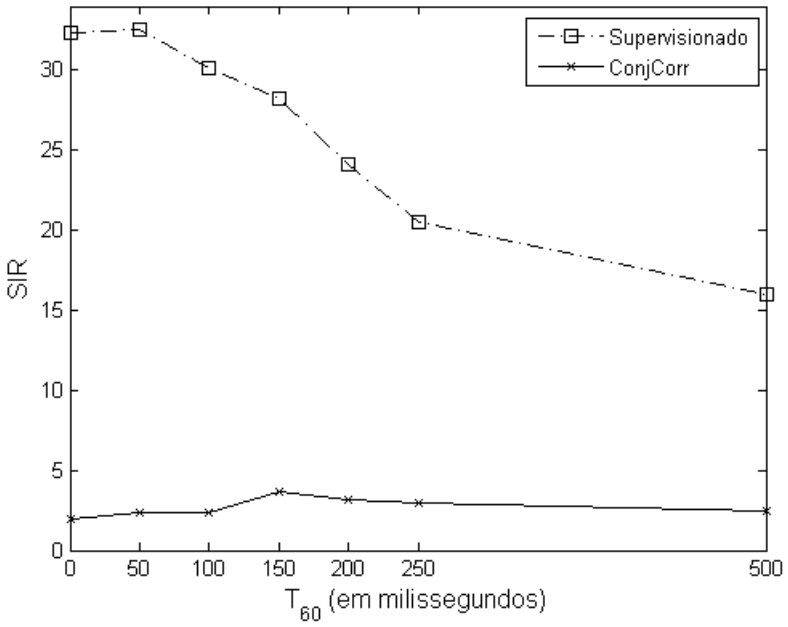

Figura 4.21: Desempenho do método ConjCorr, utilizando a disposição da Figura A.3.

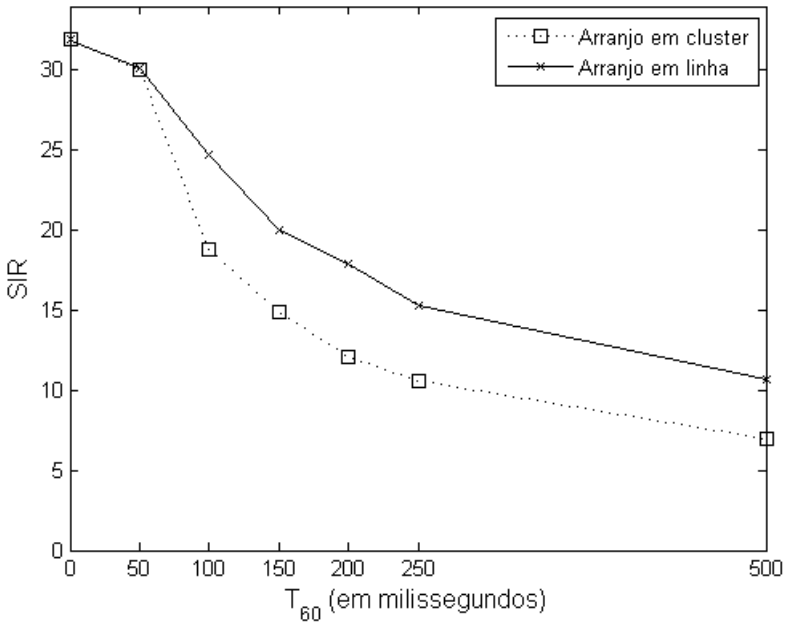

Figura 4.22: Desempenho do método DOA + GlobalCorr + LocalCorr, utilizando a disposição da Figura A.3, com o arranjo em *cluster* e com o arranjo (modificado) em linha.

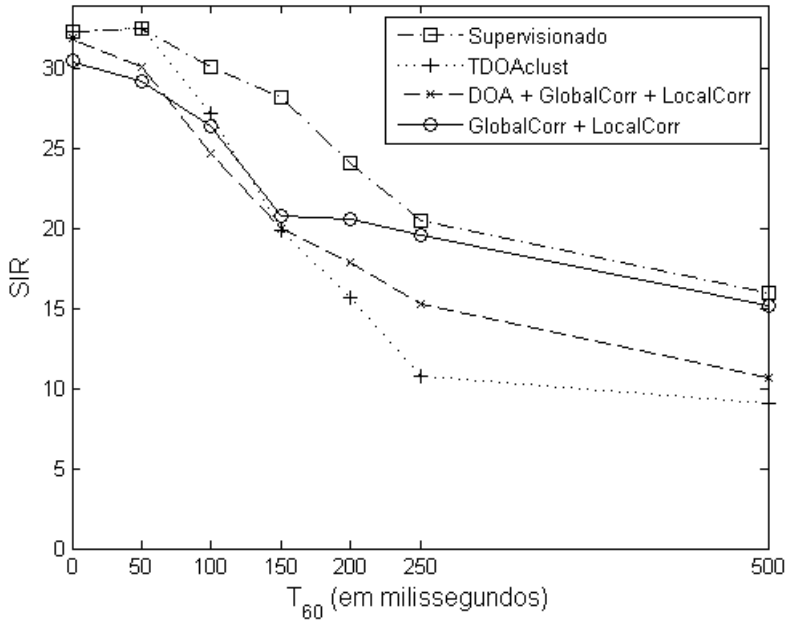

Figura 4.23: Comparação entre os métodos TDOA, DOA + GlobalCorr + LocalCorr, e GlobalCorr + LocalCorr, utilizando o arranjo da Figura A.3 (no caso do DOA + GlobalCorr + LocalCorr, o arranjo de microfones foi modificado para um arranjo em linha).

Capítulo 5

Conclusões

Essa dissertação abordou o tema de separação de fontes cegas no domínio da frequência em ambientes reverberantes. As configurações das misturas pertencem, obviamente, ao caso convolutivo, e os testes foram feitos em uma simulação de sala reverberante.

No Capítulo 2, foram contemplados os casos instantâneo e convolutivo de BSS. Os problemas de ambiguidade, inerentes à BSS, foram apresentados, e viu-se que não há como recuperar a ordem das fontes nem sua amplitude, a não ser que se utilize alguma informação "a priori" sobre as fontes. Ainda neste capítulo, algumas propriedade estatísticas importantes foram estendidas para trabalhar com números complexos, o que normalmente não é encontrado na literatura. Foi visto que, para estatísticas de ordem maior de que 1, como variância e obliquidade, há mais de uma forma de calculá-las, mais especificamente, há ceil($\frac{p+1}{2}$) formas de se calcular o momento central de ordem p. Em nossas aplicações, a medida mais consistente de se calcular o momento é utilizar o momento central absoluto de ordem p, que utilizamos para estatísticas de ordem maior do que 1. Também mostramos os cálculos amostrais destas estatísticas, que é um conceito indispensável em aplicações práticas. Existem medidas amostrais polarizadas ou não-polarizadas, porém, para um número grande de amostras, elas convergem para o mesmo valor, e podemos optar por qualquer uma das duas medidas. Uma estatística importante, e muito utilizada em BSS, é a correlação, mas não há consenso sobre sua definição na literatura de BSS. O conceito estatístico é o mais correto, mas muitas vezes se considera implicitamente que a média dos sinais é zero e sua variância é 1, e a correlação fica idêntica à covariância de dois sinais de média zero.

Os algoritmos ICA trabalham com o conceito de independência, que está intimamente relacionado com a não-gaussianidade, que deve ser maximizada para que os sinais sejam independentes. Foram derivados os algoritmos que utilizam curtose e negentropia para maximização da não-gaussianidade, chegando nos mesmos resultados. Um algoritmo de ponto fixo pode ser obtido, que converge rapidamente e

com um baixo custo computacional. Este algoritmo é conhecido como FastICA. Paralelamente, o ICA pode ser derivado através da maximização da verossimilhança, obtendo-se o algoritmo conhecido como Natural ICA. Este último necessita da estimativa da distribuição das fontes, que vimos não ser um fator muito crucial. Mesmo com uma estimativa grosseira, o algoritmo obtém bons resultados. Ele também possui uma importante propriedade: é relativamente independente do valor da matriz de mistura $\mathbf{H}$, no sentido de que converge bem mesmo para uma matriz de mistura mal condicionada. Estes dois algoritmos formam a base da separação no domínio da frequência. O FastICA é mais rápido e não necessita de passo de adaptação, enquanto que o Natural ICA obtém resultados melhores. Ainda neste capítulo, foi mostrada a forma de avaliação de desempenho utilizada, através do SIR, SDR e SAR.

O Capítulo 3 tratou da Separação Cega de Fontes no Domínio da Frequência, que transforma os sinais para o domínio da frequência, transformando o ICA convolutivo em K ICAs instantâneos, onde K é o número de raias da FFT. Isso torna as ambiguidades de escalamento e permutação um grande problema a ser resolvido. A ambiguidade do escalamento pode ser facilmente resolvida através do MDP (Princípio da Mínima Distorção), segundo detalhado na Seção 3.6, o que nos deixa com o problema da permutação a resolver. Foi mostrado com detalhes como a transformação tempo-frequência é realizada, inclusive na prática. É comum utilizar-se uma janela de análise, que deve atender à COLA para que não haja distorção, o que afeta o desempenho do BSS, como foi visto na Seção 3.2. Citamos várias janelas que atendem à COLA para determinado salto J, e foi feita uma comparação entre elas, para verificar qual obtém o melhor desempenho. Foi observado que a janela retangular com $J = \frac{L}{4}$ obteve o melhor desempenho. Foram feitos testes adicionais no Capítulo 4, que mostraram que a janela retangular não é a ótima para a separação, mas funciona melhor para todos os métodos de resolver a permutação, o que incentiva o seu uso.

As janelas de análise podem ser aplicadas em qualquer situação, mas as janelas de síntese não fazem sentido se o objetivo for a implementação de convoluções lineares, que é o caso de BSS. Neste caso, a restrição de que o número de raias K deve ter aproximadamente o tamanho da janela somado ao tamanho do filtro deve ser seguido, para que a reconstrução seja perfeita. Vimos, no entanto, que não conhecemos o tamanho do filtro (que é a resposta de frequência da sala), e, portanto, essa condição não será satisfeita, gerando o efeito da circularidade. Uma forma de mitigar esta distorção é usar um número de raias maior do que a janela, através de *zero-padding*. Outra forma é suavizar os filtros da matriz separadora $\mathbf{W}$, segundo visto na Seção 3.7. Esta suavização se mostrou eficaz, aumentando o SDR e SAR das saídas. Testamos a suavização com várias janelas diferentes para verificar qual

obtém um melhor desempenho, e a janela de Hanning, comumente utilizada na literatura nesta etapa, não se mostrou a melhor opção. Janelas com maior resolução temporal se mostraram melhores, como a de Chebyshev e as da família Blackman.

Vimos a importância do branqueamento antes de realizar a separação dos sinais propriamente dita, que é essencial no caso do FastICA e um passo muito útil se utilizado o Natural ICA. Como o Natural ICA é baseado em gradiente, ele acaba sendo muito influenciado pelo passo de adaptação. Se a potência do sinal for muito diferente de uma raia pra outra, é necessário ajustar o passo de adaptação por raia para que a convergência seja uniforme. Branqueando os sinais, esta potência é normalizada, e não precisamos lidar com este problema. Adicionalmente, o branqueamento faz aproximadamente metade do trabalho de separação, como visto na Seção 2.5.1, por um custo computacional muito menor do que o do ICA. Uma outra vantagem do branqueamento é que pode ser realizado PCA para redução dimensional (no caso de mais sensores que fontes $M > N$) sem nenhum custo adicional, pois os autovetores e autovalores já foram calculados.

Com relação à separação dos sinais propriamente dita, foi apresentado um algoritmo que une a velocidade do FastICA à precisão do Natural ICA. Primeiramente é aplicado o FastICA, que rapidamente converge, e a matriz separadora resultante é utilizada como matriz inicial no algoritmo Natural ICA, que aumenta a precisão do FastICA ajustando a matriz separadora. Comparamos a aplicação da função *score* do Natural ICA na forma cartesiana e na forma polar, chegando à conclusão de que a forma polar é a melhor opção. Verificamos isto através de testes e análise da convergência dos algoritmos. Também apresentamos um algoritmo do tipo Natural ICA adaptável, cuja função *score* pode ser modificada de acordo com a curtose da distribuição dos sinais a serem separados. Verificamos que ele obteve um resultado superior a outros algoritmos de separação, porém com um custo computacional mais elevado, o que restringe sua aplicação.

No Capítulo 4, focamos no problema da permutação, o principal problema do ICA no domínio da frequência. As soluções para este problema se classificam em dois grupos (excluindo as abordagens que modificam a etapa de separação para incluir todas as frequências de uma vez só na adaptação): as baseadas em localização das fontes e as baseadas em correlação de envelope.

Os dois modelos utilizados para localização de fontes são o de campo distante e campo próximo, mas nas soluções, só utilizamos o modelo de campo distante. Os modelos de localização de fontes possuem uma limitação com relação à distância entre microfones. Embora o desempenho com uma distância maior entre eles deveria ser melhor, ele acaba piorando devido ao *aliasing* espacial. Na prática utilizamos microfones com espaçamento de 4 cm entre eles. A primeira abordagem derivada da localização de fontes utiliza os padrões de diretividade de cada fonte, cujos mínimos

indicam o DOA das outras fontes. O problema é que encontrar mínimos de uma função não é fácil, e para N fontes, há $N-1$ mínimos em cada padrão de diretividade, i.e, para $N > 2$, aparecem mínimos locais, o que dificulta ainda mais a localização dos mínimos globais. Felizmente, existe uma forma analítica de encontrar esses ângulos mínimos, que torna a aplicação muito mais simples. Deve-se tomar cuidado com a ambiguidade do DOA, entretanto, pois se o cosseno de dois ângulos DOA de duas fontes diferentes for o mesmo, não conseguimos distinguir entre elas. Esta ambiguidade não existe no método TDOA, que também utiliza localização das fontes, mas em vez de utilizar o ângulo de chegada, utiliza a diferença entre tempos de chegada do sinal da fonte a dois microfones. Esse método necessita de um algoritmo de clusterização, e foram apresentados dois. O primeiro utiliza o *K-means*, mas não é o preferido, pois não considera restrições inerentes ao nosso problema. O segundo considera estas restrições, e apresentou resultados superiores. Infelizmente, à medida que a reverberação da sala aumenta, as estimativas de TDOA passam a não ser suficientes para classificar as fontes.

A correlação, no entanto, não sofre deste problema, e se comporta bem à medida que a reverberação da sala aumenta. A correlação de envelope é feita entre frequências adjacentes ou harmônicas, pois foi visto que uma mesma fonte possui correlação alta entre estas frequências. Pode ser utilizado o envelope AM, i.e, o módulo dos valores complexos de uma determinada raia em função do *frame*, ou então uma medida que foi chamada de *powRatio*, que apresentou resultados melhores de acordo com os testes realizados. Há mais de um algoritmo de otimização disponíveis em função das medidas de correlação interfrequências. Apresentamos quatro: HarmCorr, LocalCorr, GlobalCorr e ConjCorr, e verificamos seu desempenho. Todos eles, com exceção do GlobalCorr, só apresentam resultados satisfatórios se algumas raias de frequência já estiverem com permutação alinhada. Isso nos leva a utilizar o DOA antes de algum dos outros três métodos, ou ainda o próprio GlobalCorr (no caso de ser seguido pelo LocalCorr). Fica destacada então a falta de robustez dos métodos de correlação, pois uma permutação desalinhada provoca um efeito cascata, desalinhando todas as frequências adjacentes a ela.

Os dois grupos de soluções para resolver o problema da permutação podem ser unidos em um algoritmo que se beneficie da precisão da correlação e da robustez da localização das fontes. Primeiramente o DOA é encontrado, mas somente nas frequências onde a medida é confiável a permutação é alinhada. Nas outras frequências, as permutações são alinhadas através de algum dos métodos de correlação. Vimos que os métodos que combinam as abordagens funcionam muito bem, embora seu desempenho degrade bastante à medida que a reverberação aumenta. Este não é um problema do método conjugado, mas sim da localização de fontes. O método que utiliza a combinação de GlobalCorr com LocalCorr foi o que obteve o melhor desem-

penho para salas reverberantes, chegando muito perto do desempenho do método supervisionado.

5.1 Trabalhos Futuros

Separação Cega de Sinais de Áudio é um trabalho relativamente recente, e por isso a gama de trabalhos futuros é muito extensa. Esporadicamente, é realizada uma campanha de avaliação de desempenho de algoritmos de separação de fontes do mundo todo. Inclusive, os sinais de áudio utilizados nos testes nesta dissertação foram obtidos no *site* da campanha de 2007 [80]. Recentemente, em 2010, foi realizada uma outra campanha deste tipo [81], que mostrou que ainda há muito que avançar em ambientes reverberantes. Essa campanha focou em ambientes com bastante reverberação. O resultado dos algoritmos que resolvem o problema da permutação utilizando GlobalCorr + LocalCorr superou o dos outros algoritmos, o que era de se esperar, visto sua robustez com relação à reverberação da sala. Melhorias em métodos baseados em correlação parecem ser um bom caminho para trabalhos futuros. Na nossa opinião, a localização das fontes provavelmente se tornará um fim, ao invés de um meio. Utilizar a localização das fontes para resolver problemas de separação será substituído por utilizar a separação das fontes para resolver problemas de localização destas. Pensando assim, as técnicas de localização de fontes apresentadas aqui continuam sendo importantes, mesmo com esta mudança de foco. Em [57], o autor propõe utilizar o algoritmo TRINICON, introduzido na Seção 3.4.1, para encontrar os TDOAs e descobrir a localização da fonte.

Explorar casos onde há mais misturas que fontes também é um caminho promissor, e merece ser explorado. Se a informação extra puder ser utilizada para melhorar o desempenho da separação, ao invés de ser descartada, aumentar o número de microfones ajudaria a resolver problemas mais difíceis. Unificar os algoritmos de separação com algoritmos de reconhecimento de fala, e realizar testes de desempenho, pode ajudar a avaliar os algoritmos de separação, de um ponto de vista mais prático.

Existem, além dos citados, muitos outros trabalhos que podem ser explorados, bastando apenas uma mente inspirada de um pesquisador astuto. Finalizando, esse tópico têm um futuro brilhante pela frente, e aguardamos o dia em que poderemos conversar com máquinas assim como conversamos com pessoas, em um cenário digno de um filme de James Cameron.

Referências Bibliográficas

[1] ZUE, V., GLASS, W., JAMES, R. "Conversational Interfaces: Advances and Challenges", *Proc. IEEE*, v. 88, n. 8, pp. 1166–1180, 2000.

[2] PRASAD, R. K., SARUWATARI, H., SHIKANO, K. "Robots That Can Hear, Understand and Talk", *Advanced Robotics*, v. 18, n. 5, pp. 533–564, 2004.

[3] BRONKHORST, A. W. "The Cocktail Party Phenomenon: A Review on Speech Inteligibility in Multiple-Talker Conditions", *Acta Acustica united with Acustica*, v. 86, pp. 117–128, 2000.

[4] MAKINO, S., ARAKI, S., SAWADA, H. "Frequency-Domain Blind Source Separation". In: S. Makino, T. Lee, H. S. (Ed.), *Blind Speech Separation*, Springer, cap. 2, pp. 47–78, 2007.

[5] LEHMANN, E. A., JOHANSSON, A. M. "Prediction of Energy Decay in Room Impulse Responses Simulated with an Image-Source Model", *The Journal of the Acoustic Society of America*, v. 124, n. 1, pp. 269–277, Julho 2008.

[6] LEHMANN, E. A., JOHANSSON, A. M., NORDHOLM, S. "Reverberation-Time Prediction Method for Room Impulse Responses Simulated with the Image-Source Model". In: *2007 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, 2007.

[7] GROSSMAN, S. I. *Elementary Linear Algebra*. Wadsworth, 1984.

[8] LAMBERT, R. H. *Multichannel Blind Deconvolution: FIR Matrix ALgebra and Separation of Multipath Mixtures*. Ph.D. dissertation, University of Southern California, Maio 1996.

[9] MONTGOMERY, D. C., RUNGER, G. C. *Applied Statistics and Probability for Engineers*. John Wiley & Sons, 2002.

[10] ERIKSSON, J. O., KOIVUNEN, V. E. "Complex Random Vectors and ICA Models: Identifiability, Uniqueness, and Separability", *IEEE Transactions on Information Theory*, v. 52, n. 3, pp. 1017–1029, Março 2006.

[11] JÖRESKOG, K. G. “Formulas for Skewness and Kurtosis”, .

[12] ERIKSSON, J. O., KOIVUNEN, V. E. “Statistics for Complex Random Variables Revisited”. In: *IEEE Int. Conf. Acoustics, Speech, Signal Proc. (ICASSP 2009)*, pp. 3565–3568, Taipei, Abril 2009.

[13] MATHIAS, R. “Matrix Completions, Norms, and Hadamard Products”, *Proceedings of the American Mathematical Society*, v. 117, n. 4, pp. 905–918, Abril 1993.

[14] HYVÄRINEN, A., KARHUNEN, J., OJA, E. *Independent Component Analysis*. John Wiley & Sons, 2001.

[15] PEDERSEN, M. S., LARSEN, J., KJEMS, U., et al. “A Survey of Convolutive Blind Source Separation Methods”. In: J. Benesty, Y. Huang, M. S. (Ed.), *Springer Handbook of Speech Processing*, Springer Press, pp. 1–34, Novembro 2007.

[16] HYVÄRINEN, A., OJA, E. “A Fast Fixed-Point Algorithm for Independent Component Analysis”, *Neural Computation*, v. 9, n. 7, pp. 1483–1492, 1997.

[17] HYVÄRINEN, A. “Fast and Robust Fixed-Point Algorithms for Independent Component Analysis”, *IEEE Transactions on Neural Networks*, v. 10, n. 3, pp. 626–634, 1999.

[18] DOUGLAS, S. C., GUPTA, M., SAWADA, H., et al. “Spatio-Temporal FastICA Algorithms for the Blind Separation of Convolutive Mixtures”, *IEEE Trans. Acoustics, Speech, Signal Proc.*, v. 15, n. 5, pp. 1511–1520, Julho 2007.

[19] AMARI, S., CICHOCKI, A., YANG, H. H. “A New Learning Algorithm for Blind Signal Separation”. In: *Advances in Neural Information Processing Systems*, v. 8, pp. 757–763, 1996.

[20] BELL, A. J., SEJNOWSKI, T. J. “An Information-Maximization Approach to Blind Separation and Blind Deconvolution”, *Neural Computation*, v. 7, pp. 1129–1159, 1995.

[21] CARDOSO, J. F., LAHELD, B. H. “Equivariant Adaptive Source Separation”, *IEEE Trans. Signal Proc.*, v. 44, n. 12, pp. 3017–3030, Dezembro 1996.

[22] BELL, A. J., SEJNOWSKI, T. J. “An Information-Maximization Approach to Blind Separation and Blind Deconvolution”, *Neural Computation*, v. 7, pp. 1129–1159, 1995.

[23] CARDOSO, J. F. "Infomax and Maximum Likelihood for Blind Source Separation", *IEEE Signal Proc. Letters*, v. 4, n. 4, pp. 112–114, Abril 1997.

[24] CARDOSO, J. F. "Blind Signal Separation: Statistical Principles", *Proc. IEEE*, v. 86, n. 10, pp. 2009–2025, Outubro 1998.

[25] BINGHAM, E., HYVÄRINEN, A. "A Fast Fixed-Point Algorithm for Independent Component Analysis of Complex Valued Signals", *International Journal of Neural Systems*, v. 10, n. 1, pp. 1–8, Fevereiro 2000.

[26] CARDOSO, J. F. "On the Performance of Orthogonal Source Separation Algorithms", *Proc. EUSIPCO*, pp. 776–779, Setembro 1994.

[27] PAPOULIS, A. *Probability, Random Variables and Stochastic Processes*. McGraw Hill, 1991.

[28] COVER, T. M. *Elements of Information Theory*. John Wiley & Sons, 1991.

[29] FÉVOTTE, C., GODSILL, S. J. "A Bayesian Approach for Blind Separation of Sparse Sources", *IEEE Transactions on Audio and Speech Processing*, v. 14, n. 6, pp. 2174–2188, 2006.

[30] AMARI, S. "Natural Gradient Works Efficiently in Learning", *Neural Computation*, v. 10, n. 2, pp. 251–276, 1998.

[31] LI, H., ADALI, T. "Complex-Valued Adaptive Signal Processing using Nonlinear Functions", *EURASIP Journal on Advances in Signal Processing*, v. 2008, 2008.

[32] BUCHNER, H., AICHNER, R., KELLERMAN, W. "A Generalization of Blind Source Separation Algorithms for Convolutive Mixtures Based on Second-Order Statistics", *IEEE Trans. Speech Audio Proc.*, v. 13, n. 1, pp. 120–134, Janeiro 2005.

[33] CICHOCKI, A., AMARI, S. "Adaptive Blind Signal and Image Processing - Learning Algorithms and Applications". cap. 6, pp. 231–272, John Wiley & Sons, 2002.

[34] AMARI, S., CHEN, T. P., CICHOCKI, A. "Nonholonomic Orthogonal Learning Algorithm for Blind Source Separation", *Neural Computation*, v. 12, n. 6, pp. 1463–1484, 2000.

[35] VINCENT, E., GRIBONVAL, R., FÉVOTTE, C. "Performance Measurement in Blind Audio Source Separation", *IEEE Trans. Audio, Speech, Language Proc.*, v. 14, n. 4, pp. 1462–1469, Julho 2006.

[36] OPPENHEIM, A. V., SCHAFER, R. W. *Discrete-Time Signal Processing*. Prentice Hall, 1999.

[37] ALLEN, J. B. "Short-term Spectral Analysis, Synthesis and Modification by Discrete Fourier Transform", *IEEE Trans. Acoustics, Speech, Signal Proc.*, v. ASSP-25, n. 3, pp. 235–238, Junho 1977.

[38] ALLEN, J. B. "Applications of the Short-Time Fourier Transform to Speech Processing and Spectral Analysis". In: *IEEE Int. Conf. on Acoustics, Speech and Signal Proc, ICASSP'82*, pp. 1012–1015, Maio 1982.

[39] SMITH, J. O. *Spectral Audio Signal Processing, October 2008 Draft*. `http://ccrma.stanford.edu/~jos/sasp/`, acessado em julho de 2010.

[40] ALLEN, J. B., RABINER, L. R. "A Unified Approach to Short-Time Fourier Analysis and Synthesis", *Proc. IEEE*, v. 65, n. 11, pp. 1558–1564, Novembro 1977.

[41] HARRIS, F. J. "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform", *Proc. IEEE*, v. 66, pp. 51–84, Janeiro 1978.

[42] ALLEN, J. B., RABINER, L. R. "Some Windows with Very Good Sidelobe Behavior", *IEEE Trans. Acoustics, Speech, Signal Proc.*, v. ASSP-29, n. 1, pp. 84–91, Fevereiro 1981.

[43] RABINER, L. R., SCHAFER, R. W. *Digital Processing of Speech Signals*. Prentice Hall, 1978.

[44] SMARAGDIS, P. "BLind Separation of Convolved Mixtures in the Frequency Domain", *Neurocomputing*, v. 22, pp. 21–34, 1998.

[45] SMARAGDIS, P. "Efficient Blind Separation of Convolved Sound Mixtures", *Applications Signal Proc. Audio Acoustics*, Outubro 1997.

[46] ADALI, T., LI, H. "A Practical Formulation for Computation of Complex Gradients and Its Application to Maximum Likelihood ICA". In: *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Proc. (ICASSP'07)*, v. 2, pp. 633–636, Honolulu, Hawaii, USA, 2007.

[47] SAWADA, H., MUKAI, R., ARAKI, S., et al. "Polar Coordinate based Nonlinear Function for Frequency Domain Blind Separation", *IEICE Trans. Fund.*, v. E86-A, n. 3, pp. 590–596, Março 2003.

[48] CHOI, S., CICHOCKI, A., AMARI, S. "Flexible Independent Component Analysis", *Journal of VLSI Signal Processing*, v. 26, pp. 25–38, 2000.

[49] BUCHNER, H., AICHNER, R., KELLERMAN, W. "Blind Source Separation for Convolutive Mixtures: A Unified Treatment". In: Y. Huang, J. B. (Ed.), *Audio Signal Processing for Next-Generation Multimedia Communication Systems*, Kluwer Academic Publishers, cap. 10, pp. 255–293, Fevereiro 2004.

[50] MOLGEDEY, L., SCHUSTER, H. G. "Separation of a Mixture of Independent Signals using Time Delayed Correlations", *Physical Review Letters*, v. 72, pp. 3634–3636, 1994.

[51] TONG, L., LIU, R. W., SOON, V. C., et al. "Indeterminacy and Identifiability of Blind Identification", *IEEE Trans. on Circuits and Systems*, v. 38, pp. 499–509, 1991.

[52] KAWAMOTO, M., MATSUOKA, K., OHNISHI, N. "A Method of Blind Separation for Convolved Non-Stationary Signals", *Neurocomputing*, v. 22, pp. 157–171, 1998.

[53] IKEDA, S., MURATA, N. "An Approach to Blind Source Separation of Speech Signals". In: *Proc. Int. Symposium on Nonlinear Theory and its Applications*, Crans-Montana, Switzerland, 1998.

[54] PARRA, L., SPENCE, C. "Convolutive Blind Separation of Non-Stationary Sources", *IEEE Trans. Speech Audio Proc.*, v. 8, n. 3, pp. 320–327, Maio 2000.

[55] SCHOBBEN, D. W. E., SOMMEN, P. C. W. "A Frequency Domain Blind Signal Separation Method Based on Decorrelation", *IEEE Trans. Signal Proc.*, v. 50, n. 8, pp. 1855–1865, Agosto 2002.

[56] BUCHNER, H., AICHNER, R., KELLERMAN, W. "Blind Source Separation for Convolutive Mixtures Exploiting Nongaussianity, Nonwhiteness, and Nonstationarity". In: *Proc. Int. Workshop on Acoustic Echo and Noise Control (IWAENC)*, Kyoto, Japan, Setembro 2003.

[57] BUCHNER, H., AICHNER, R., KELLERMANN, W. "TRINICON-based Blind System Identification with Application to Multiple-Source Localization and Separation". In: S. Makino, T. Lee, H. S. (Ed.), *Blind Speech Separation*, Springer, cap. 4, pp. 101–147, 2007.

[58] ANEMÜLLER, J., KOLLMEIER, B. "Amplitude Modulation Decorrelation for Convolutive Blind Source Separation". In: *Proc. ICA 2000*, pp. 215–220, Junho 2000.

[59] RAHBAR, K., REILLY, J. P. "A Frequency Domain Method for Blind Source Separation of Convolutive Audio Mixtures", *IEEE Trans. Speech Audio Proc.*, v. 13, n. 5, pp. 832–844, Setembro 2005.

[60] LEE, I., KIM, T., LEE, T.-W. "Complex FastIVA: A Robust Maximum Likelihood Approach of MICA for Convolutive BSS", *ICA'06*, pp. 625–632, 2006.

[61] KIM, T., ATTIAS, H., LEE, S.-Y., et al. "Blind Source Separation Exploiting Higher-Order Frequency Dependencies", *IEEE Trans. Audio, Speech, Lang. Proc.*, v. 15, n. 1, Janeiro 2007.

[62] HIROE, A. "Solution of Permutation Problem in Frequency Domain ICA, using Multivariate Probability Density Functions", *ICA'06*, pp. 601–608, Abril 2006.

[63] MATSUOKA, K., NAKASHIMA, S. "Minimal Distortion Principle for Blind Source Separation", *Proc. ICA*, pp. 722–727, Dezembro 2001.

[64] SAWADA, H., MUKAI, R., DE LA K. DE RYHOVE, S., et al. "Spectral Smoothing for Frequency-Domain Blind Source Separation". In: *International Workshop on Acoustic Echo and Noise Control (IWAENC'03)*, pp. 311–314, Kyoto, Japan, Setembro 2003.

[65] BENESTY, J., CHEN, J., HUANG, Y. "Microphone Array Signal Processing". cap. 3, pp. 39–66, Springer, 2008.

[66] MUKAI, R., SAWADA, H., ARAKI, S., et al. "Frequency-Domain Blind Source Separation of Many Speech Signals Using Near-Field and Far-Field Models", *EURASIP Journal on Applied Signal Processing*, 2006. Article ID 83683.

[67] ABHAYAPALA, T. D., KENNEDY, R. A., WILLIAMSON, R. C. "Spatial Aliasing for Near-Field Sensor Arrays", *Eletronics Letters*, v. 35, n. 10, pp. 764–765, Maio 1999.

[68] VEEN, B. D. V., BUCKLEY, K. M. "Beamforming: a Versatile Approach to Spatial Filtering", *IEEE ASSP Magazine*, pp. 2–24, Abril 1988.

[69] DMOCHOWSKI, J., BENESTY, J., AFFÈS, S. "On Spatial Aliasing in Microphone Arrays", *IEEE Trans. Signal Proc.*, v. 57, n. 4, pp. 1383–1395, Abril 2009.

[70] KURITA, S., SARUWATARI, H., KAJITA, S., et al. "Evaluation of Blind Signal Separation Method using Directivity Pattern under Reverberant Conditions". In: *Proc. of IEEE Int. Conf. Acoustics, Speech, Signal Proc., 2000 (ICASSP'00)*, v. 5, pp. 3140–3143, Istanbul, Turquia, Agosto 2000.

[71] SAWADA, H., MUKAI, R., ARAKI, S., et al. "A Robust and Precise Method for Solving the Permutation Problem of Frequency-Domain Blind Source Separation", *IEEE Trans. Speech, Audio Proc.*, v. 12, n. 5, pp. 530–538, Setembro 2004.

[72] IKRAM, M. Z., MORGAN, D. R. "A Beamforming Approach to Permutation Alignment for Multichannel Frequency-Domain Blind Speech Separation". In: *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Proc. 2002 (ICASSP'02)*, v. 1, pp. I–881–I–884, Orlando, Florida, USA, Maio 2002.

[73] MUKAI, R., SAWADA, H., ARAKI, S., et al. "Solving the Permutation Problem of Frequency-Domain BSS when Spatial Aliasing Occurs with Wide Sensor Spacing", *IEEE Int. Conf. Acoustics, Speech, Signal Proc. (ICASSP'06)*, v. 5, pp. V–77–V–80, Maio 2006.

[74] DUDA, R. O., HART, P. E., STORK, D. G. *Pattern Classification*. Wiley Interscience, 2000.

[75] MURATA, N., IKEDA, S. "An On-line Algorithm for Blind Source Separation on Speech Signals", *Proceedings of 1998 Int. Symposium on Nonlinear Theory and Its Applications (NOLTA'98)*, v. 3, pp. 923–926, Setembro 1998.

[76] MURATA, N., IKEDA, S. "A Method of ICA in Time-Frequency Domain", *Proc. Int. Workshop Independent Comp. Analysis and Blind Signal Separation (ICA'99)*, pp. 365–371, Janeiro 1999.

[77] SERVIÈRE, C., PHAM, D. T. "Permutation Correction in the Frequency Domain in Blind Separation of Speech Mixtures", *EURASIP Journal on Applied Signal Processing*, 2006. Article ID 75206.

[78] SAWADA, H., ARAKI, S., MAKINO, S. "Measuring Dependence of Bin-wise Separated Signals for Permutation Alignment in Frequency-Domain BSS". In: *IEEE Int. Symp. Circuits Systems (ISCAS'07)*, pp. 3247–3250, Maio 2007.

[79] SAWADA, H., ARAKI, S., MAKINO, S. "Measuring Dependence of Bin-Wise Separated SIgnals for Permutation Alignment in Frequency-Domain BSS".

In: *IEEE Int. Symp. Circuits and Systems, 2007 (ISCAS'07)*, pp. 3247–3250, New Orleans, LA, Maio 2007.

[80] VINCENT, E., SAWADA, H., BOFILL, P., et al. "First Stereo Audio Source Separation Evaluation Campaign: Data, Algorithms and Results". In: *Proc. Int. Conf. on Independent Component Analysis and Signal Separation*, 2007.

[81] ARAKI, S., OZEROV, A., GOWREENSUNKER, V., et al. "The 2010 Signal Separation Evaluation Campaign (SiSEC2010): Audio Source Separation". In: *Lectures Notes in Computer Science, 2010 (LNCS'10)*, v. 6365, pp. 114–122, 2010.

[82] HADDAD, D. B. *Propostas para Separação Cega e Supervisionada de Fontes*. Dissertação de mestrado, COPPE - Universidade Federal do Rio de Janeiro, Junho 2008.

[83] COMON, P. "Independent Component Analysis, a new concept?" *Signal Processing*, v. 36, pp. 287–314, 1994.

[84] BOASHASH, B. *Time Frequency Signal Analysis and Processing: A Comprehensive Reference*. Elsevier, 2003.

[85] VAIDYANATHAN, P. P. *Multirate Systems and Filter Banks*. Prentice Hall, 1993.

[86] MUKAI, R., SAWADA, H., ARAKI, S., et al. "Frequency Domain Blind Source Separation Using Small and Large Sensor Pairs", *Proc. of the 2004 Int. Symp. on Circuits and Systems (ISCAS'04)*, v. 5, pp. V–1–V–4, Maio 2004.

[87] SAWADA, H., ARAKI, S., MUKAI, R., et al. "Blind Extraction of a Dominant Source Signal from Mixtures of Many Sources". In: *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Proc. 2005 (ICASSP'05)*, v. 3, pp. iii/61–iii/64, Março 2005.

[88] WEIHUA, W., FENGGANG, H. "Improved Method for Solving Permutation Problem of Frequency Domain Blind Source Separation". In: *6th IEEE Int. Conf. Industrial Informatics, 2008 (INDIN'08)*, pp. 703–706, Daejeon, Julho 2008.

[89] IKRAM, M. Z., MORGAN, D. R. "Exploring Permutation Inconsistency in Blind Separation of Speech Signals in a Reverberant Environment". In:

Proc. IEEE Int. Conf. Acoustics, Speech, Signal Proc. (ICASSP), v. 2, pp. 1041–1044, Istanbul, Turkey, Junho 2000.

Apêndice A

Ambiente de Teste

Simulamos a resposta de frequência de uma sala utilizando o algoritmo chamado de *Image-Source Model*, detalhado em [5, 6], ambos do mesmo autor. Detalhes de como funciona o algoritmo estão fora do escopo desta dissertação; o que nos interessa é o seu funcionamento.

A sala utilizada nas simulações tem dimensões $4,45\,\mathrm{m} \times 3,55\,\mathrm{m} \times 2,5\,\mathrm{m}$ (largura × comprimento × altura). Todos os lados da sala (paredes, teto e chão) não possuem portas nem janelas, para fins de simulação. Além disso, o coeficiente de absorção de todos os lados é o mesmo, como se todos fossem feitos do mesmo material. O material utilizado varia com o tempo de reverberação, e o algoritmo de simulação calcula o coeficiente de absorção de cada lado da sala dependendo do tempo de reverberação escolhido. O arranjo de microfones, em todas as simulações, foi montado em torno do ponto $Mic_c = [2 \quad 1,5 \quad 1,6]^T$. Independentemente do arranjo, o "centro de massa" do arranjo de microfones era sempre o mesmo. As fontes foram distribuídas em torno do centro do arranjo, com dois parâmetros para identificá-las: o DOA de cada uma, e a distância delas até o arranjo. O parâmetro mais importante da sala é o tempo de reverberação desta. O tempo utilizado nesta dissertação é o T_{60}, que é o tempo requerido para que as reflexões cheguem a 60 dB abaixo do nível do som direto.

As fontes utilizadas foram do SASSEC [80], e são trechos de sinal de voz de 10 segundos de duração cada, amostrados a 16 kHz. Decimamos os sinais para que a frequência de amostragem caísse para 8 kHz, portanto, a não ser que dito o contrário, em todos os testes os sinais de voz utilizados foram amostrados a 8 kHz. Em testes com duas fontes e dois microfones, o ambiente de teste está representado na Figura A.1. Em testes com três fontes e três microfones, dois ambientes de teste foram utilizados. O primeiro considera que os microfones foram montados em linha, segundo mostrado na Figura A.2 e o segundo considera que os microfones foram montados em "cluster", segundo mostrado na Figura A.3.

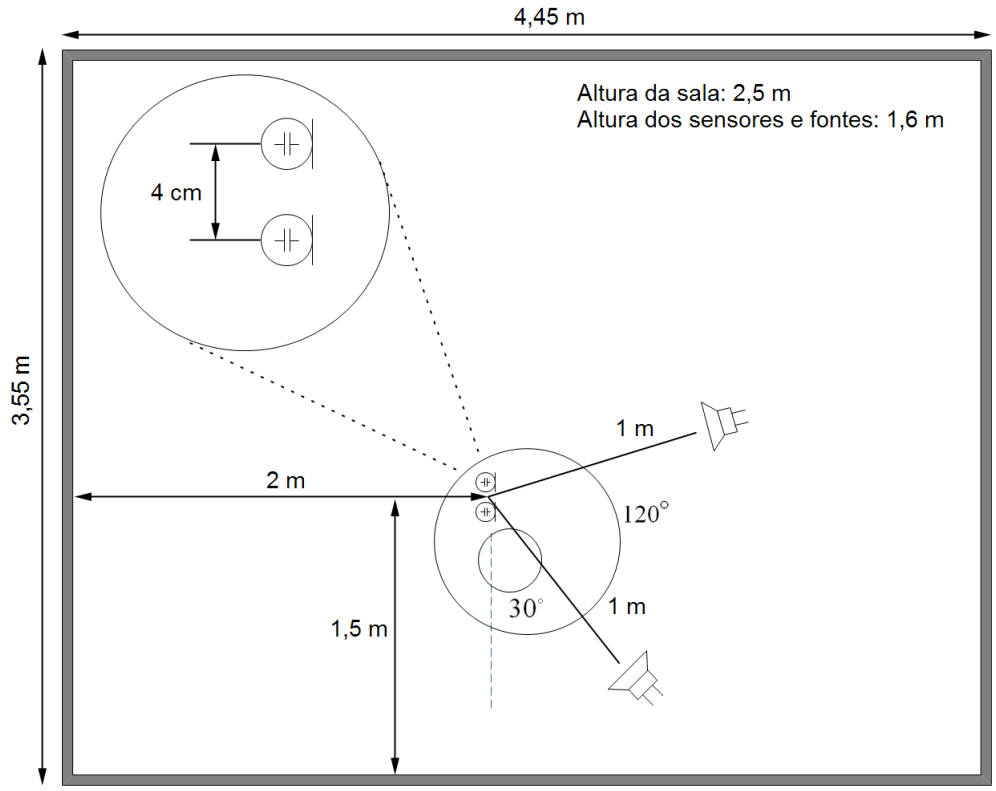

Figura A.1: Configuração da sala utilizada nos testes quando há dois microfones e duas fontes.

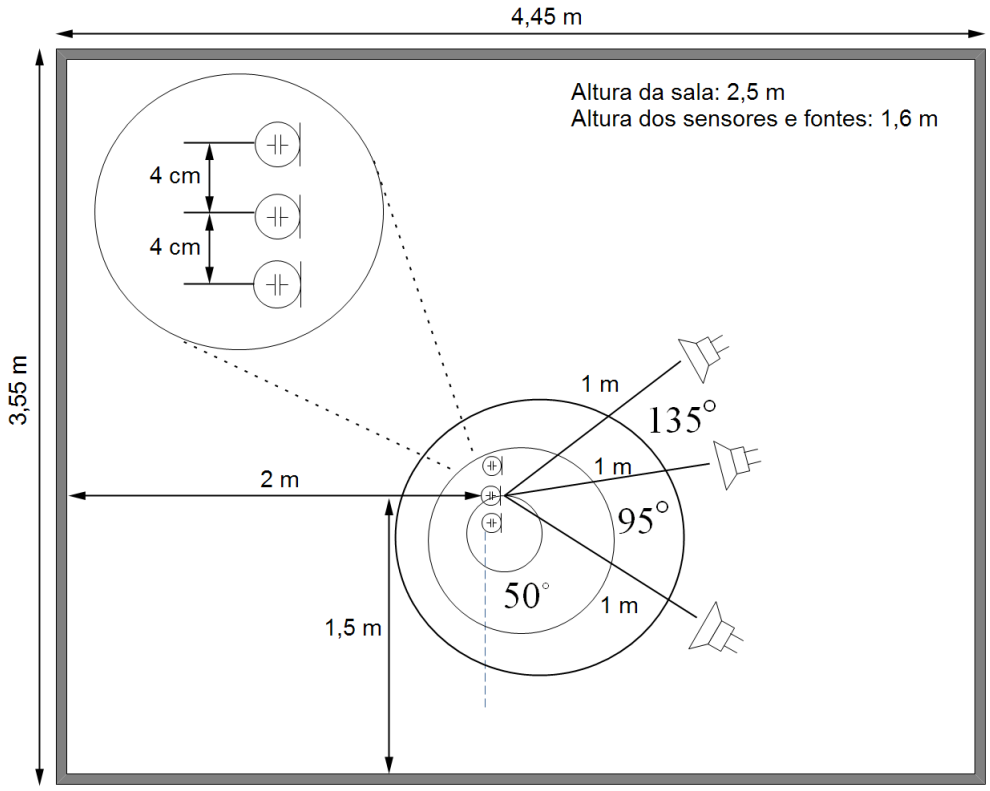

Figura A.2: Configuração da sala utilizada nos testes quando há três microfones e três fontes, e o arranjo de microfones é em linha.

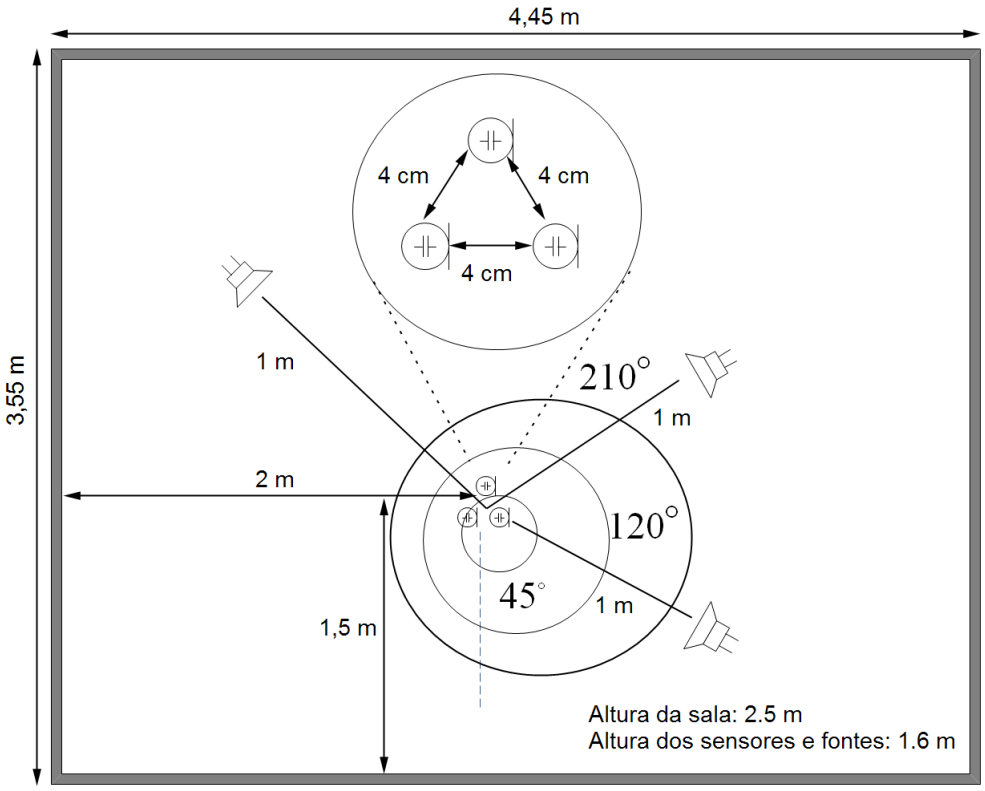

Figura A.3: Configuração da sala utilizada nos testes quando há três microfones e três fontes, e o arranjo de microfones é em cluster.

Apêndice B

Descobrindo Convergência dos Algoritmos ICA

Aqui descreveremos a forma que utilizamos para descobrir a convergência dos algoritmos FastICA e Natural ICA. É importante utilizar um critério de convergência em vez de um número fixo de iterações, pois isso diminui bastante o tempo de processamento do algoritmo, e não conseguimos encontrar um critério na literatura para o Natural ICA.

No caso do FastICA, o critério de convergência é mais simples. Segundo o desenvolvimento do algoritmo de ponto fixo feito na Seção 2.5.2, quando o vetor separador $\mathbf{w}_i$ apontar na mesma direção do gradiente, ou seja, quando ele não mudar mais de direção após uma iteração. Como o algoritmo FastICA (mostrado em (2.66)) restringe o vetor separador a ser unitário, a condição de parada é:

$$|\mathbf{w}_i(it)||\mathbf{w}_i^H(it-1)| \approx 1 \Rightarrow |\mathbf{w}_i(it)||\mathbf{w}_i^H(it-1)| = 1 - \epsilon \tag{B.1}$$

onde $\mathbf{w}_i(it)$ é o valor da iteração atual e $\mathbf{w}_i(it-1)$ o valor da iteração anterior, e ϵ é um valor muito pequeno. O $|.|$ aparece porque estamos trabalhando com números complexos. Como os vetores são unitários, seu produto interno não pode ser maior do que 1, e eles convergem para 1, que acontece quando sua direção é igual. O valor ϵ define a condição de parada. Nas simulações, utilizamos $\epsilon = 0$ sem maiores complicações. Um ponto importante a se notar é que tivemos que definir um número mínimo de iterações, pois em algumas raias de frequência, a adaptação inicial é muito lenta, e o vetor $\mathbf{w}_i$ mudava muito pouco de direção de uma iteração para outra, como pode ser observado na Figura B.1. Um valor mínimo de 8 iterações foi suficiente para que o algoritmo convergisse sem problemas.

No caso do Natural ICA, se a função *score* foi aplicada de forma cartesiana, as partes real e imaginária dos elementos de $\mathbf{w}_i$ convergem para um valor fixo, e se ela foi utilizada a forma polar, apenas o módulo dos elementos converge. Utilizamos o

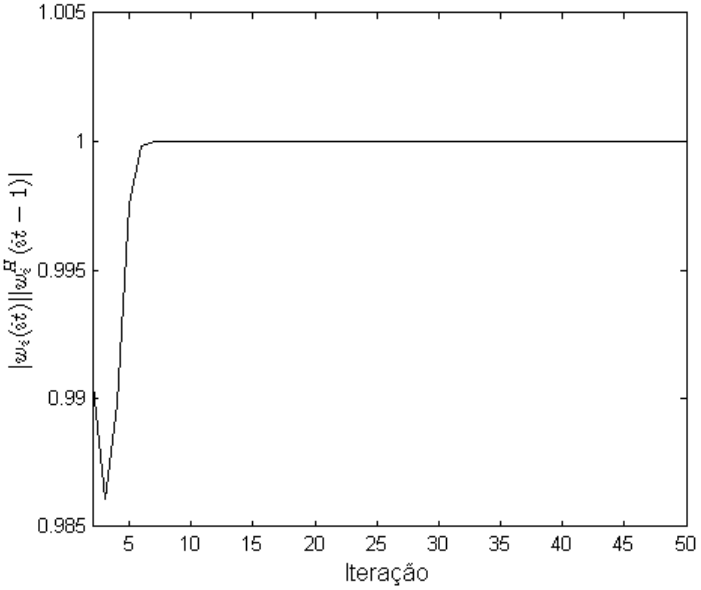

Figura B.1: Convergência típica do FastICA.

módulo para testar convergência, por dois motivos: o módulo converge em ambas as formas e utilizamos sempre a forma polar, como explicado na Seção 3.4.

É típico utilizar como condição de convergência um pequeno valor ϵ, e se a diferença entre o módulo do elemento do vetor separador da iteração anterior e o da atual for menor que ϵ, o algoritmo convergiu, como mostrado em (B.2).

$$|w_{ij}(it)| - |w_{ij}(it-1)| < \epsilon \tag{B.2}$$

Entretanto, há ocasiões em que, na convergência do Natural ICA, ele fica oscilando entre dois valores, como mostrado na Figura B.2. Dessa forma, não podemos escolher um valor de ϵ muito pequeno, ou a condição de convergência não vai funcionar. Porém, se escolhermos um valor de ϵ grande, isso pode afetar o desempenho de outras raias de frequência, onde esse problema não acontece, e o algoritmo pararia antes de realmente convergir.

Para resolver este problema, definimos uma janela $\mathbf{w}_{err}$ de tamanho E_w, par, que é preenchida com os valores da diferença entre os módulos dos elementos do vetor separador das últimas E_w iterações (erros). A diferença entre os módulos, obviamente, pode ser negativa. Então, definimos a condição de parada como sendo a soma dos erros das últimas iterações. Sabendo que $|.|_1$ simboliza a norma-1 de um

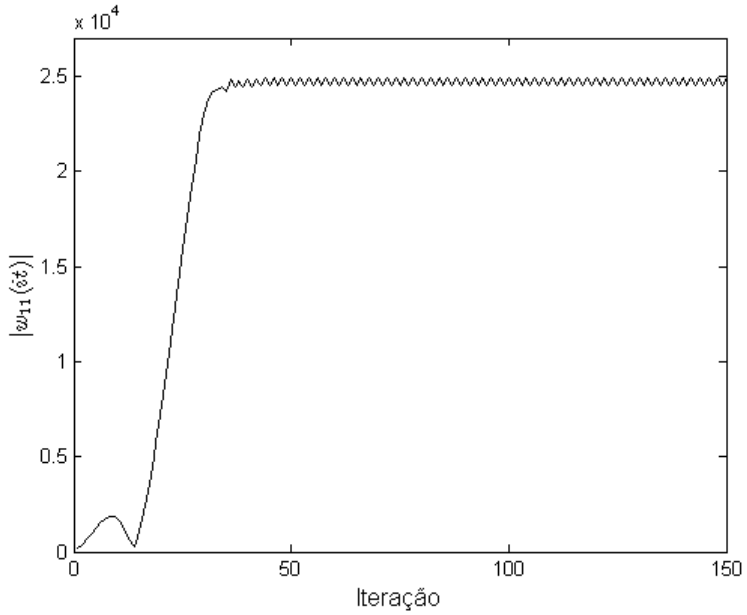

Figura B.2: Convergência do Natural ICA em algumas raias de frequência, onde o valor final fica oscilando.

vetor, ou seja, a soma dos módulos de seus elementos, então:

$$|\mathbf{w}_{err}|_1 < \epsilon_w \tag{B.3}$$

$$\mathbf{w}_{err} = \begin{bmatrix} |w_{ij}(it)| - |w_{ij}(it-1)| \\ |w_{ij}(it-1)| - |w_{ij}(it-2)| \\ \vdots \\ |w_{ij}(it-E_w+1)| - |w_{ij}(it-E_w)| \end{bmatrix} \tag{B.4}$$

Se a solução estiver oscilando, a soma entre os E_w valores de erros, onde E_w é par, dará 0, e o algoritmo convergiu. E se a solução não oscilar, e ir somente em uma direção, esta condição é similar a (B.2). O algoritmo terá um número mínimo de iterações igual a E_w, para encher a janela, e, se houver oscilações durante a convergência, deve-se garantir que o período dessas oscilações seja maior do que E_w, senão pode ser indicada falsa convergência. Na prática, escolhemos $E_w = 8$ e $\epsilon_w = 0,01$, para um passo de adaptação $\eta = 0,2$ (embora o valor de ϵ não seja muito dependente de η).

Apêndice C

Métodos Supervisionados para Resolver o Problema da Permutação

Aqui apresentaremos dois métodos supervisionados utilizados para resolver o problema da permutação. Eles foram utilizados quando se queria fazer uma comparação sem que o problema da permutação influenciasse ou quando se queria descobrir qual a máxima SIR obtido se o problema da permutação estivesse perfeitamente resolvido.

É bom ressaltar que este método visa resolver *apenas* o problema da permutação, então ele não influencia na separação das fontes. Ele não altera a matriz separadora de nenhuma forma a não ser permutar suas linhas. Ambos os métodos obtiveram os mesmos resultados, então, quando na dissertação estiver escrito que algum problema de permutação foi resolvido de forma supervisionada, pode ter sido utilizado qualquer um dos dois métodos.

O primeiro método é chamado de *MaxSIR*, e foi proposto por Makino, em [71]. Ele utiliza as observações da fonte em cada microfone:

$$q_{ji}(n) = \sum_{l=-\infty}^{\infty} h_{ji}(l) s_i(n-l) \tag{C.1}$$

O método consiste em maximizar a SIR em cada raia de frequência. Primeiro é aplicada a STFT a (C.1), segundo já explicado na Seção 3.2, utilizando qualquer janela, embora o salto J tenha que ser igual ao salto utilizado para passar as misturas para o domínio da frequência. Para manter a consistência, é utilizada a mesma janela que foi utilizada nas misturas:

$$q_{ji}(m,k) = \sum_{n} q_{ji}(n) win_a(n - mJ) \exp\left(-\hat{\jmath}\frac{2\pi kn}{K}\right) \tag{C.2}$$

onde $win_a(n)$ é a janela utilizada na STFT das misturas. Seja $\mathbf{Q}_k(m)$ a matriz $M \times N$ formada pelos elementos $Q_{ji}(m,k)$, isto é, cada linha corresponde a um sensor e cada coluna, a uma fonte. Então, a permutação na frequência k é dada por:

$$\mathbf{P}_k = \text{argmax}_P \left\{ \text{trace}(E\{\mathbf{P}_k \mathbf{W}_k \mathbf{Q}_k(m)\}) \right\} \tag{C.3}$$

onde, obviamente, o valor esperado $E\{.\}$ é calculado em sua forma amostral, segundo (2.24), e trace($\mathbf{V}$) calcula o traço da matriz $\mathbf{V}$, i.e, a soma dos elementos de sua diagonal principal. A diagonal principal de $\mathbf{W}_k\mathbf{Q}_k(m)$, se a permutação estiver correta, contém o sinal das fontes, e os outros elementos, em cada linha i, contém as interferências das fontes $i' \neq i$ na fonte i, para a matriz separadora $\mathbf{W}$. Então, a permutação que obtiver o maior traço é a permutação correta.

O outro método consiste em utilizar os sinais das fontes $s_i(n)$ diretamente, mais especificamente, sua representação $s_{ik}(m)$ no domínio da frequência. O método consiste em calcular a soma da correlação (segundo (2.38)) entre o sinal real da fonte $s_{ik}(m)$ e a saída $y_{ik}(m)$, para $i = 1, \cdots, N$, para todas as permutações possíveis. A maior soma representa a matriz de permutação correta.

$$\mathbf{P}_k = \text{argmax}_P \left\{ \sum_{i=1}^{N} r_{sy}(s_i k, y_{i_P k}) \right\} \tag{C.4}$$

onde o índice i_P simboliza a fonte i depois de ser permutada pela matriz $\mathbf{P}$. Podemos interpretar (C.4) (para ficar similar ao método (C.3)) como o cálculo do traço da matriz de correlação (ver (2.40)) entre o vetor coluna $\mathbf{s}_k$, que contém os valores de todas as fontes da raia de frequência k, e o vetor $\mathbf{P}\mathbf{y}_k$. O maior traço corresponde à permutação correta, pois a correlação entre s_{ik} e y_{ik} quando a permutação está correta é máxima.

$$\mathbf{P}_k = \text{argmax}_P \left\{ \text{trace}(R_{sy}(\mathbf{s}_k, \mathbf{P}\mathbf{y}_k)) \right\} \tag{C.5}$$

Poderíamos utilizar o sinal $q_{ji}(m,k)$ ao invés do sinal da fonte, pois, como visto na Seção 3.6, a saída $y_i(n)$ fica no máximo igual a $s_i(n)$ filtrada, entretanto, não faz diferença. O único requerimento para o algoritmo funcionar bem é que a correlação entre fontes iguais seja maior do que entre fontes diferentes, para todas as raias de frequência, i.e, $r_{s_{1k}y_{1k}} > r_{s_{1k}y_{2k}}$, o que é verdade, senão o algoritmo de BSS não funcionaria.