TRABALHO FINAL DE GRADUAÇÃO

Técnicas de Separação Cega de Fontes em Matlab: Mistura Instantânea e Convolutiva

Candidato: Mario Filiage Svetlic Filho RA: 11026107

Orientadora: Prof[a] Dra Aline de Oliveira Neves Panazio

Palavras-chave: Separação Cega de Fontes, Análise por componentes independentes, misturas convolutivas

1. INTRODUÇÃO

A necessidade de extração ou restauração da informação contida em um sinal a partir de uma versão corrompida do mesmo é uma necessidade recorrente em processamento de sinais. Tanto o problema de equalização, que envolve a tentativa de inverter o efeito do canal, quanto o problema de separação cega de fontes - na qual sensores são utilizados para captação de um sinal sem conhecer, a priori, nenhuma das fontes - se encaixam neste contexto.[19]

A importância de se aprofundar o estudo em técnicas de separação cega de fontes, bem como algoritmos pertinentes que trabalham em cima da questão de misturas convolutivas, foi o assunto abordado nesse trabalho. Foram estudadas as técnicas clássicas existentes para separação de fontes, no caso convolutivo, na qual existe uma dependência temporal. As técnicas foram testadas no domínio do tempo [1] e, no domínio da frequência [6,7], com o intuito de compararmos suas eficácias.

Existem múltiplas aplicações envolvendo separação cega em misturas convolutivas. Em acústica, diferentes fontes de som são gravadas simultaneamente com vários microfones. Essas fontes podem ser de voz ou fontes de música. Também temos os sinais gravados de sonares ultramarinos. Nas comunicações de rádio, arranjo de antenas recebem misturas compostas por diferentes sinais. A separação de fontes é aplicada na área da astronomia e na comunicação via satélite para recompor imagens que estão deterioradas devido ruídos provenientes do meio. Por último, os modelos convolutivos têm sido utilizados para interpretar dados funcionais de imagens cerebrais e de tensões de sinais biomédicos.

Para alcançar esse objetivo, faremos uso do Matlab em diferentes cenários e problemas, como no exemplo da separação de sinais de voz e áudio.

2. FUNDAMENTAÇÃO TEÓRICA

A leitura e compreensão do cenário a ser exposto foram realizadas primeiramente em uma etapa anterior de iniciação cientifica, abordando naquela primeira fase somente misturas instantâneas. Tal etapa foi de suma importância para inteligibilidade dos problemas nos quais se aplicam as técnicas de separação cega de fontes, servindo como base para dar continuidade nos estudos que envolvem o mote dessa pesquisa.

Nesse trabalho enfatizamos o caráter de separação convolutiva e muitos dos conceitos já vistos se aplicam plenamente para esse novo caso.

O sistema completo (transmissão, mistura e separação) foi implementado em Matlab, incluindo uma interface gráfica amigável que torna a simulação mais simples e o acesso mais fácil. Com tal simulador, foi possível testar o desempenho dos diversos algoritmos quando aplicados à separação de diferentes misturas e os gráficos serão mostrados em janela apropriada a fim de compararmos os resultados obtidos.

Será imprescindível a compressão de alguns conceitos que envolvem as técnicas de separação que serão estudadas.

Esperança

A esperança de uma variável aleatória é por definição a média de sua distribuição de probabilidades. Sendo definida pelo símbolo E(x). Caso x seja uma variável aleatória discreta assumindo valores $\{x_1,x_2,x_3,...\}$ com probabilidade $\{p_1,p_2,p_3, ...\}$, respectivamente, então sua esperança é dada pela fórmula:

$$\mathrm{E}(\mathrm{x}) = \sum_{\mathrm{i}=1}^{\infty} \mathrm{x_i p(x_i)} \quad (\mathbf{1})$$

Correlação Cruzada

Em processamento de sinais, a correlação cruzada pode ser definida, como sendo a medida de similaridade de duas formas de onda e pode ser definida como:

$$\mathrm{E[xy]} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathrm{xy\ f(x,y)dx\,dy} \quad (\mathbf{2})$$

onde f(x,y) é definida como a função densidade de probabilidade das variáveis x e y.

Para o caso em que as variáveis são discretas, temos g e h como funções discretas, sendo f* o complexo conjugado, então a correlação entre elas será:

$$\mathrm{E[xy]} = \sum_{m=-\infty}^{\infty} f^*[m]g[m+n]$$

O coeficiente de correlação, quando normalizado, varia no intervalo entre -1 e +1. Quando E[xy] é nulo, os sinais são ditos descorrelacionados e sua covariância também é nula.

A autocorrelação mede a semelhança entre amostras de um mesmo sinal obtidas em instantes de tempo diferentes. Ela permite que se analise o grau de irregularidade de um sinal. Podemos dizer que é a correlação cruzada de um sinal com ele próprio.

A autocorrelação informa o quanto o valor de uma realização de uma variável aleatória é capaz de influenciar seus vizinhos. Seu valor pode ser nulo, indicante que os sinais não possuem correlação entre si.

Para um processo estacionário, a função autocorrelação depende apenas da diferença entre os instantes t_1 e t_2 . Assim, neste caso, definindo-se $\tau = t_2 - t_1$ pode-se reescrever a equação 2 como:

$$Rxx(\tau) = \mathrm{E[x(t)\ x(t+\tau)]} \quad (\mathbf{3})$$

2.1 DEFINIÇÃO DO MODELO:

Um dos problemas típicos investigados por técnicas de separação cega de fontes é o chamado *cocktail party*. Em uma festa ou reunião, vários tipos de sinais (fontes) estão presentes e perturbam a compreensão e identificação de uma determinada fonte em especial. Graças à grande capacidade de processamento do cérebro humano, não sentimos dificuldade em realizar tal tarefa e a fazemos facilmente no nosso dia-a-dia. Entretanto, se utilizarmos sensores, é necessário um esforço considerável para extrair, a partir de um sinal captado, a informação de uma fonte sem interferência de outras e sem conhecer, a priori, nenhuma das fontes.

A mistura empregada no sistema completo é desconhecida, entretanto, leva-se em consideração a hipótese de que os sinais das fontes são estatisticamente independentes entre si. Essa hipótese leva à conhecida Análise por Componentes Independentes (ICA - Independent Component Analysis) e foi considerada no desenvolvimento de técnicas empregadas no contexto de separação cega de fontes.

Em muitas situações, deseja-se recuperar todas as fontes de misturas gravadas ou, ao menos, separar algumas delas. No entanto, pode ser útil identificar o processo de mistura por si só, já que esse pode revelar informações sobre a natureza do sistema de mistura em si.

Para simplificar o problema abordado, iremos citar rapidamente um caso mais simples de misturas que considera uma aproximação grosseira do meio: as misturas instantâneas. Logo em seguida adentraremos com o exemplo de caso de mistura convolutivas.

2.2 Misturas instantâneas

Considerando uma aproximação grosseira para melhor entendimento do problema, denotaremos o exemplo a seguir considerando uma estimativa grosseira do que ocorre na prática, sem levar em conta o fator de atraso dos sinais tampouco o caminho através do meio.

Imagine que você está em uma sala onde duas pessoas estão falando simultaneamente. Você tem dois microfones fixados em dois locais distintos. Esse dois microfones fornecem duas gravações de sinais temporais, os quais podemos denotar por $X_1(n)$ e $X_2(n)$ no instante n.

A figura 1 ilustra o problema citado para o caso de M misturas. Na figura 1 temos N fontes independentes sendo também N o total de sinais recuperados. Denotamos a matriz de mistura por A e a matriz de separação por W.

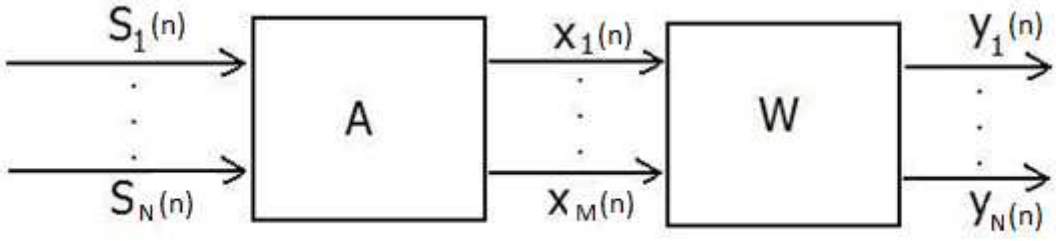


Figura 1: Modelagem de Separação Cega de Fontes

Cada uma das gravações temporais fornecidas por cada microfone é uma soma ponderada dos sinais emitidos. Podemos expressar o vetor **X(n)** que contém as observações (amostras) de um total de M misturas (geralmente M≥N) pela equação:

$$\mathbf{X(n)} = A \,.\, \mathbf{S(n)} \quad (4)$$

Onde **S(n)** = [$S_1(n)$, $S_2(n)$, $S_3(n)$, ... , $S_N(n)$] é um conjunto com N fontes independentes. De maneira geral, no caso de misturas instantâneas, podemos representar cada elemento do vetor de observações **X(n)** como sendo :

$$X_1 = A_{11}S_1 + A_{12}S_2 + ...A_{1N}S_N$$

$$X_2 = A_{21}S_1 + A_{22}S_2 + ...A_{2N}S_N \quad (5)$$

$$\vdots$$

$$X_M = A_{M1}S_1 + A_{M2}S_2 + ...A_{MN}S_N$$

onde A_{ij} são constantes.

O objetivo é então estimarmos uma matriz W que seja a inversa da matriz A, sem conhecer previamente as fontes originais e a matriz mistura A. Na prática, nós não sabemos como determinar W exatamente, pois não temos conhecimento da matriz A, mas podemos encontrar uma estimativa que nos dá uma boa aproximação. Basicamente, para isso, consideramos a hipótese de que as fontes são independentes e buscamos sinais $\mathbf{Y_i(n)}$ com { i=1, ... ,N } também independentes. Dessa forma precisaremos medir a independência entre os sinais recuperados e comentaremos a respeito logo a seguir.

Novamente considerando que a matriz de mistura é inversível, pode-se recuperar os sinais das fontes através de uma matriz W tal que:

$$\mathbf{Y(n)} = W \,.\, \mathbf{X(n)} \quad (6)$$

Idealmente, se temos a equação $W = A^{-1}$, relacionando ambas matrizes, podemos concluir que teremos

$$W.\, A\, \mathbf{S} = \mathbf{S} \qquad (7)$$

sabemos que $A.\, A^{-1} = I$ e que qualquer matriz T multiplicada pela identidade I resulta na própria matriz, ou seja, $T.\, I = T$, então realizando as substituições necessárias, chegamos na equação 7.

A chave para a estimativa do modelo ICA é a busca da independência entre os sinais. Visto que as fontes são independentes e, se os sinais **Y(n)** também forem independentes, garantimos que estes sinais serão iguais a S a menos de um ganho e de uma ambiguidade de ordem. As restrições impostas para isso são:

- O número de misturas observadas M deve ser maior ou igual ao número de componentes independentes N, assim M≥N;
- As fontes devem ser estatisticamente independentes entre si. Podemos definir essa independência estatística de misturas em termos das probabilidades das mesmas, ou seja, a densidade conjunta pode ser fatorada resultando no produto das densidades marginais. A maneira de representar a independência das fontes envolvidas deve obedecer a equação a seguir:

$$P(S) = \prod_{i=1}^{N} P[S_i(n)]$$

- As componentes independentes devem possuir distribuições de probabilidades não-gaussianas. Na verdade, apenas uma componente gaussiana é permitida, uma vez que combinações lineares de uma variável gaussiana também apresenta distribuição gaussiana. Isso deve-se ao fato de que se somente fontes gaussianas fossem empregas, a separação não iria além do branqueamento do vetor de misturas e a independência não seria alcançada. Vide mais detalhes na próxima seção 2.2.1.

Retomando a equação (6), para obtermos a independência estatísticas das fontes, temos que $\mathbf{Y(n)} = \mathrm{W} \cdot \mathbf{X(n)}$, adotando W = C e **X(n) = S**, teremos

$$\mathbf{Y} = \mathrm{C}\,\mathbf{S} \qquad (8)$$

onde

$$\mathrm{C} = \mathrm{DP} \qquad (9)$$

sendo D uma matriz diagonal de ganho e P uma matriz de permutação. Assim, as fontes originais serão recuperadas a menos de uma ambiguidade de ganho e ordem. Em seguida, discutiremos os principais conceitos e critérios que permitem medir a independência entre sinais.

2.2.1 Entropia

É definida como a medida de incerteza de uma variável aleatória. Entropia mede a quantidade média de informação que um sinal carrega.

Simplificadamente, entropia é um número, maior ou igual a zero, que representa a quantidade média de informação gerada pela fonte, em cada símbolo.

Matematicamente ela pode ser definida como:

$$\mathrm{H(x)} = -\sum P_x(\mathrm{x} = x_i)\ log_2\ P_x(\mathrm{x} = x_i)$$

com x sendo uma variável aleatória e $P_x(\mathrm{x})$ sua função densidade de probabilidade. Essa entropia é definida para o caso em que temos o sinal discreto.

2.2.2 Informação Mútua

Informação mútua é uma medida natural de dependência entre variáveis aleatórias. Trata-se da quantidade de informação que uma variável aleatória carrega sobre outra variável aleatória.

$$\mathrm{I\ (x,y)} = \mathrm{H(x)} - \mathrm{H(x/y)} \quad (\mathbf{10})$$

Onde x e y são duas variáveis aleatórias, H(x) é a medida da entropia de x e H(x**/**y) é a entropia condicional, ou seja, é a incerteza restante em uma distribuição após se conhecer a outra.

Uma outra maneira de definir a informação mútua I(x**,**y) é empregando a probabilidade de cada fonte e suas probabilidades conjuntas:

$$\mathrm{I\ (x,y)} = \mathrm{I(y,x)} = \sum\sum \mathrm{p(x,y)log_2}\frac{\mathrm{p\ \ (x,y)}}{\mathrm{[p(x)p(y)]}} \quad (\mathbf{11})$$

Quando duas variáveis possuem baixo grau de correlação consequentemente, sua informação mútua será baixa, e na situação contrária isso também é válido. Essa

grandeza é sempre não-negativa e vale zero se, e somente se, vigora uma condição de independência. Assim, I(x,y) = 0 somente se x e y forem independentes.

Assim, tal medida se torna interessante para ser usada como critério para separação de fontes. De fato, a minimização da informação mútua é equivalente a maximização da soma das não-gaussianidades das estimativas, quando as estimativas são forçadas a serem descorrelacionadas [18]. A restrição de descorrelação não é de fato necessária, mas simplifica consideravelmente o método computacional empregado.

Uma estratégia útil de pré-processamento em ICA é primeiro branquear as variáveis observadas. Isso quer dizer que antes da aplicação do algoritmo ICA, e depois da centralização; que trata-se essencialmente da subtração de cada um dos valores de **X(n)** de sua média temporal; transformamos linearmente o vetor **X(n)** observado para obtermos um novo vetor $\mathbf{X_b(n)}$ que é branco, i.e. suas componentes são descorrelacionadas e suas variâncias unitárias. Em outras palavras, a matriz de covariância de $\mathbf{X_b(n)}$ é igual a matriz identidade:

$$E\{\mathbf{X_b(n).\ X_b(n)}^T\} = I \qquad (12)$$

O branqueamento é sempre possível. Um método popular para branqueamento é usar a decomposição dos auto-valores (EVD do inglês Eigenvalue Decomposition) da matriz de covariância $E\{\mathbf{X_b(n).\ X_b(n)}^T\} = EDE^T$, onde E é a matriz ortogonal dos auto-vetores de $E\{\mathbf{X_b(n).\ X_b(n)}^T\}$ e D é a matriz diagonal dos seus auto-valores, $D = diag(d_1,...,d_k)$ [11]. Note que $E\{\mathbf{X_b(n).\ X_b(n)}^T\}$ pode ser estimada na forma padrão das amostras disponíveis $\mathbf{X_j(n)},...,\mathbf{X_M(n)}$, com j={1,...,M}. O branqueamento pode ser feito por:

$$\mathbf{X_b(n)} = ED^{-½}E^T\mathbf{X} \qquad (13)$$

onde a matriz $D^{-½}$ é computada por uma simples componente de operação como $\mathbf{D}^{-½} = diag\ (d_1^{-½},...\ d_k)^{-½}$. Agora é fácil de verificar que $E\{\mathbf{X_b(n).\ X_b(n)}^T\} = I$

O branqueamento transforma a matriz mistura em uma nova matriz, Ã. Conforme abaixo:

$$\mathbf{X_b} = ED^{-½}E^TA\mathbf{S} = \tilde{A}\mathbf{S} \qquad (14)$$

A utilidade do branqueamento reside no fato de que a nova matriz de mistura Ã é ortogonal. Isso pode ser visto dado que:

$$E\{\mathbf{X_b(n).\ X_b(n)}^T\} = \tilde{A}E\{\mathbf{S\ S}^T\}\tilde{A}^T = \tilde{A}\tilde{A}^T = I \qquad (15)$$

Aqui podemos ver que o branqueamento reduz o número de parâmetros a serem estimados. Ao invés de ter n^2 parâmetros que são os elementos da matriz original A, precisamos somente estimar a nova matriz ortogonal de mistura Ã. Uma matriz

ortogonal contém n(n-1)/2 graus de liberdade. Por exemplo, em duas dimensões, uma transformação ortogonal é determinada por um parâmetro angular único. Em grandes dimensões, uma matriz ortogonal contém somente cerca de metade do número de parâmetros de uma matriz arbitrária. Assim pode-se dizer que o branqueamento resolve metade do problema de ICA [11]. Porque o branqueamento é um procedimento padrão simples e muito mais simples que outros algoritmos em ICA, é uma excelente forma de reduzir a complexidade do problema. Alguns algoritmos exigem que os dados sejam pré-branqueados para que a convergência ocorra.

Para facilitar a visualização dessas etapas, a figura 2 ilustra duas fontes independentes com distribuições uniformes no intervalo entre [-1,1]. Os sinais X, obtidos da mistura destas duas fontes, pode ser visualizado na figura 3. Com a operação de pré-branqueamento, obtemos a figura 4. É possível notar que, com esta operação, conseguimos uma correção da escala, mas continua havendo uma rotação que será corrigida com a aplicação do algoritmo em questão.

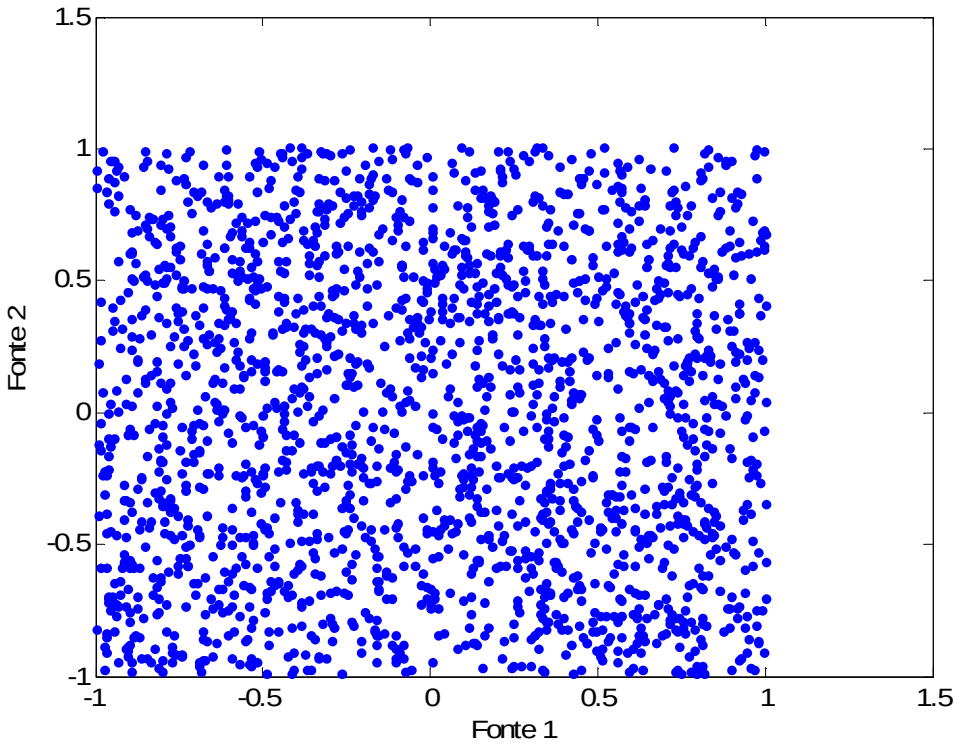


Fig. 2– Distribuição conjunta das componentes independentes S_1 e S_2

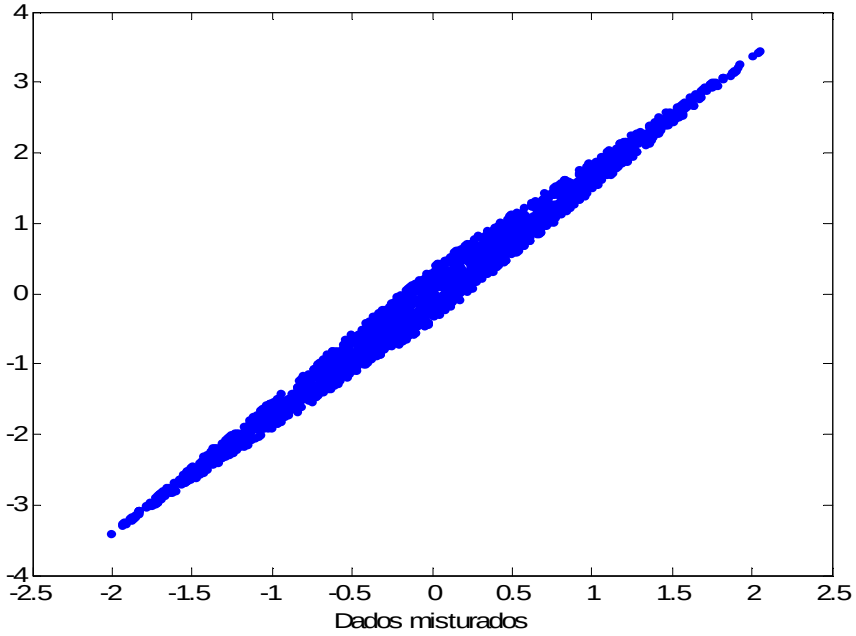


Fig. 3 – A distribuição conjunta das misturas X_1 e X_2

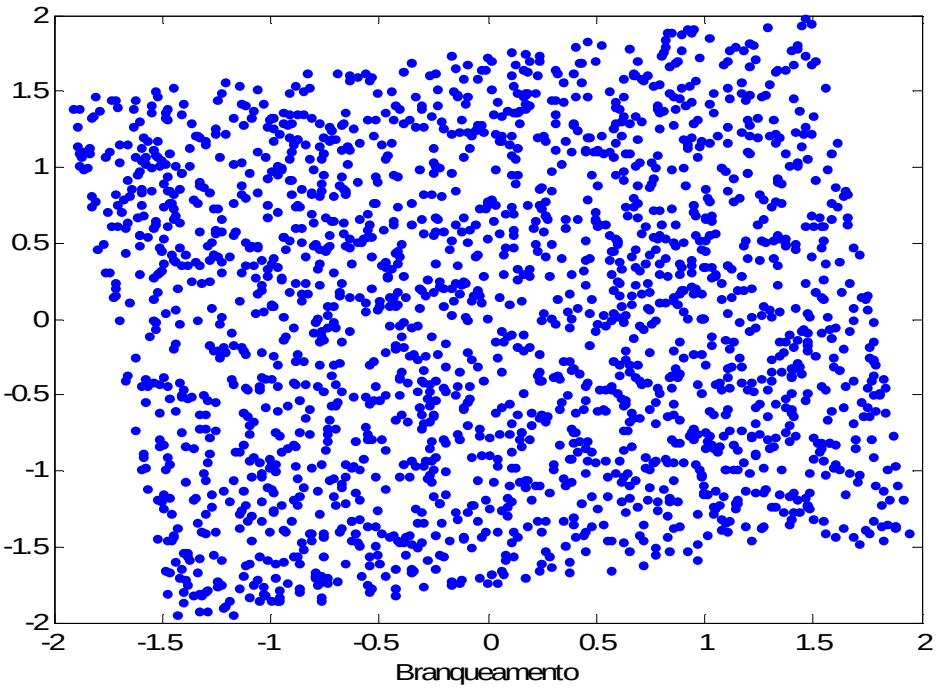


Fig. 4 – Branqueamento ou Descorrelação dos dados

2.2.3 Infomax

Um algoritmo bastante conhecido pela comunidade científica e estudado foi o Infomax [16]. Ele é baseado no conceito de maximização da entropia dos sinais de saída Y.

Uma vez que o pré-branqueamento não é uma etapa necessária para esse algoritmo, não houve necessidade, portanto, de branquear o sinal de mistura observado, uma vez que a convergência ocorreu sem grandes esforços adicionais. Mesmo aplicando o pré-branqueamento dos sinais, não notou-se diferença, tanto no tempo computacional quanto na convergência do algoritmo. Isso será explicado melhor, como veremos a seguir.

Para essa versão do Infomax, temos a atualização da matriz W dada pela equação abaixo e Y definido como em [6]:

$$W \leftarrow W + \mu.\{ [W^T]^{-1} + E[(1 - 2\mathbf{Y}) \mathbf{X_b}^T]\} \quad (\mathbf{16})$$

onde μ corresponde ao passo de adaptação arbitrário adotado no início da simulação.

2.2.4 Infomax Estendido

Outro algoritmo de grande importância foi proposto inicialmente por Te-Won Lee et al. [12], que desenvolveram uma versão generalizada ou estendida do INFOMAX, denominado Extendend-Infomax (ou também E-Infomax). O algoritmo proposto estima os momentos das fontes originais e chaveia o algoritmo de acordo com o tipo de fonte super ou sub-gaussiana.

Na sua forma original, o algoritmo Infomax considera uma não-linearidade fixa, de maneira que os cenários nos quais o algoritmo é capaz de separar os sinais ficam restritos a alguns tipos particulares de fontes. A influência da escolha destas não-linearidades é assunto de alguns artigos encontrados na literatura [13],[14]. No entanto, argumenta-se que, em geral, é possível obter bons resultados selecionando-se apenas algumas funções.

Esta idéia foi ponto de partida para uma das modificações do algoritmo Infomax, apresentada em [12]. No trabalho, introduziu- se um parâmetro adicional para determinar qual o tipo de não-linearidade que deve ser utilizada. Empregando apenas duas não-linearidades: $g(y) = y - \tanh(y)$, para fontes sub-gaussianas, e $g(y) = y + \tanh(y)$, para fontes super-gaussianas, é possível obter o algoritmo denominado Infomax Estendido, capaz de trabalhar em cenários com fontes de diferentes classes.

Com isso a equação de atualização da matriz W é dada como:

$$W \leftarrow W + \mu \{ I + E[D \tanh(\mathbf{Y})\mathbf{Y}^T - \mathbf{Y}\mathbf{Y}^T] \} \quad (17)$$

onde D = diag(m_1, . . . , m_k) é uma matriz diagonal tal que, idealmente, $m_i = 1$ para fontes super-gaussianas e $m_i = -1$ para fontes sub-gaussianas. A adaptação dos parâmetros μ é feita através da seguinte equação:

$$\mu = \text{sign}(E\{\text{sech}^2(\mathbf{Y}_i)\}E\{\mathbf{Y}^2_i\} - E\{\tanh(\mathbf{Y}_i)\mathbf{Y}_i\}) \quad (18)$$

2.2.5 Gradiente Natural

Diferentemente dos outros algoritmos analisados, o Gradiente Natural não requer inversão de matrizes, sendo esta uma grande vantagem com relação ao custo computacional requerido e conseqüentemente na velocidade de convergência [15].

Tal como o Infomax, em suas duas versões analisadas anteriormente, o Gradiente Natural exige que as fontes originais empregadas possuam independência estatísticas, além do que, o pré-braqueamento é uma etapa necessária antes da execução do algoritmo.

O algoritmo desenvolvido por Amari et al. [15] mede a dependência entre as saídas através da informação mútua e utiliza a técnica de gradiente natural para minimizá-la. Assim, a dependência entre as componentes é minimizada.

As funções g(.) , são funções não lineares escolhidas de acordo com os sinais de saída; em geral utiliza-se para g(.) os seguintes valores:

$\tanh(y) \rightarrow$ componentes super-gaussianas

$y^3 \rightarrow$ componentes sub-gaussianas

Assim tem-se a equação de atualização de W abaixo:

$$W = W + \mu (I + E[g(\mathbf{Y})\mathbf{Y}^T]) W$$

2.2.6 Não-Gaussianidade

Outra forma de separarmos sinais misturados é explorando o Teorema Central do Limite, um resultado clássico na teoria da probabilidade, que mostra como a distribuição de uma soma de variáveis aleatórias independentes, que não sejam gaussianas, tende em direção a distribuição gaussiana, sob certas condições. Assim, a soma de duas variáveis independentes normalmente tem uma distribuição que se aproxima mais de uma distribuição gaussiana do que as duas variáveis aleatórias originais[17].

Vamos assumir que um vetor de dados **X(n)** é uma mistura de componentes independentes, como mostrado na equação (4). Para estimar uma das componentes independentes, nós consideramos uma combinaçao linear de S . Se W fosse a inversa de A, (6) nos daria uma das fontes originais. A questão agora é: Como poderemos usar o Teorema Central do Limite para determinar W sendo que no caso ideal, ele deveria ser igual à inversa de A?

Como quanto mais sinais misturamos, mais gaussiano será o sinal de mistura, precisamos encontrar sinais Y que sejam o mais não-gaussianos possíveis. Portanto, precisamos estimar W de forma a maximizar a não gaussianidade de W^T **X**. Podemos, então, colocar a questão: Como medir a gaussianidade de um sinal? Algumas medidas nos permitem chegar lá. Como discutiremos na sequência:

2.2.7 Curtose

É definida como uma medida robusta de dispersão que caracteriza o "achatamento" da curva da função de distribuição da variável aleatória em relação a uma curva padrão ou curva normal. Pode ser positiva ou negativa. Variáveis aleatórias que tem curtose negativa são chamadas de sub-gaussianas, e aquelas com curtose positiva são chamadas super-gaussianas.[20]

A curtose de y é classicamente definida como:

$$\mathrm{k(y)} = \mathrm{E}(y^4) - 3(\mathrm{E}\{y^2\})^2 \quad (\mathbf{19})$$

Se x_1 e x_2 são duas variáveis aleatórias independentes, teremos:

$$\mathrm{k}(x_1 + x_2) = \mathrm{k}(x_1) + \mathrm{k}(x_2) \quad (\mathbf{20})$$

e

$$\mathrm{k}(\alpha x_1) = \alpha^4 \mathrm{k}(x_1) \quad (\mathbf{21})$$

onde α é um escalar.

Além disso, um sinal gaussiano possui curtose nula. Sendo assim, podemos usá-la para medir a não gaussianidade de um sinal definindo o seguinte critério [20]:

$$\text{máx} \mid \mathrm{k}(\mathrm{y})\mid \quad (\mathbf{22})$$

Uma outra medida possível é a chamada negentropia, descrita na seção a seguir.

2.2.8 Negentropia

A negentropia se baseia no fato de que a entropia de uma variável gaussiana é maior do que a entropia de todas as outras variáveis aleatórias de mesma variância. De fato, isso mostra que a distribuição gaussiana é a "mais aleatória" ou a menos "estruturada" de todas as distribuições [11]. Entropia é pequena para distribuições que são claramente concentradas em certos valores. Assim, a negentropia pode ser definida como [21]:

$$\mathrm{J}(\mathrm{y}) = \mathrm{H}(\mathrm{y}_{\mathrm{Gauss}}) - \mathrm{H}(\mathrm{y}) \ (\mathbf{23})$$

Onde "$\mathrm{y}_{\mathrm{Gauss}}$" é a variável aleatória gaussiana de mesma matriz de covariância que y. Devido às propriedades salientadas, a negentropia é sempre não negativa, e nula se e somente se y for gaussiana. Visto que a estimação da entropia de uma variável é uma tarefa computacionalmente difícil, é necessário se utilizar aproximações de (23). Como exemplo, uma aproximação possível é usar momentos de ordem superior:

$$\mathrm{J}\,(\mathrm{y}) \approx \frac{1}{12}\mathrm{E}(\mathrm{y}^3)^2 + \frac{1}{48}\mathrm{k}(\mathrm{y})^2 \quad (\mathbf{24})$$

Assim podemos definir, como busca da não-gaussianidade, um critério baseado na maximização da negentropia de y.

Esse conceitos formam a base para o desenvolvimento de algoritmos de separação cega de fontes, dentre eles o conhecido Fast-ICA.

2.2.9 Algoritmo Fast-ICA

O algoritmo FastICA é um algoritmo de ponto fixo que busca justamente maximizar a não-gaussianidade de Y, separando assim as fontes misturadas. É um algoritmo de convergência bastante rápida e cuja implementação é simples, o que faz dele um algoritmo muito usado dentro do contexto de separação de fontes.

Considere uma matriz A quadrada e compatível com (4), com coeficientes aleatórios uniformemente distribuídos. Considere também que X(n) foi previamente pré-branqueado como discutido na seção 2.2.1.

Assim, a equação de adaptação do algoritmo Fast-ICA é dada por [2]:

$$W \leftarrow E\{\mathbf{X_b}.g'(W^T.\mathbf{X_b})\} - E\{g''(W^T.\mathbf{X_b})\}W \quad (25)$$

Denotamos por g' e g'' a derivada de primeira e segunda ordem, respectivamente da função g não-linear usada na equação 25. Alguns exemplos de funções bastante usadas são:

$$g_1(u) = \tanh(a_1.u)$$

$$g_2(u) = u.\exp(-u^2/2)$$

onde $1 \leq a_1 \leq 2$ é uma constante adequada, muitas vezes tomada como $a_1 = 1$ [8].

Na seção 3 de simulações que será mostrada a seguir, foi adotado g'(u)=tanh(u) e g''(u)=sech^2(u).

Em seguida, veremos como as técnicas apresentadas aqui podem ser estendidas e usadas no contexto de misturas convolutivas.

2.3 Misturas Convolutivas

Agora vejamos o modelo de mistura linear convolutiva, que diferentemente daquele associado a misturas instantâneas, considera o efeito da propagação do sinal através do meio, seus multipercursos e atrasos.

Em um modelo simples, a mistura consiste em uma soma de fontes de sinais com diferentes ponderações. No entanto, em muitas aplicações do mundo real, como no exemplo de acústica, o processo de mistura é um pouco mais complexo. Neste caso, cada fonte contribui para a soma de múltiplos atrasos correspondentes aos múltiplos trajetos pelo qual um sinal acústico se propaga até um microfone. Essa soma de diferentes sinais é o que chamamos de mistura convolutiva.

A função de transferência entre cada par fonte - sensor, pode ser escrita como:

$$X_j(n) = \sum_{i=1}^{N} [\sum_{l=0}^{P-1} A_{ji}(l)\, S_i(n-l)] \qquad \textbf{(26)}$$

para j=1,...,M, de maneira que o sinal recebido é uma composição de sinais filtrados. A_{ji} é um filtro FIR com ordem P-1. Cada escalar $A_{ji}(l)$ determina o quanto da fonte i está presente na mistura j. Denotando de outra forma, onde o símbolo * representa convolução:

$$X_j(n) = \sum_{i=1}^{N} A_{ji} * S_i(n) \qquad \textbf{(27)}$$

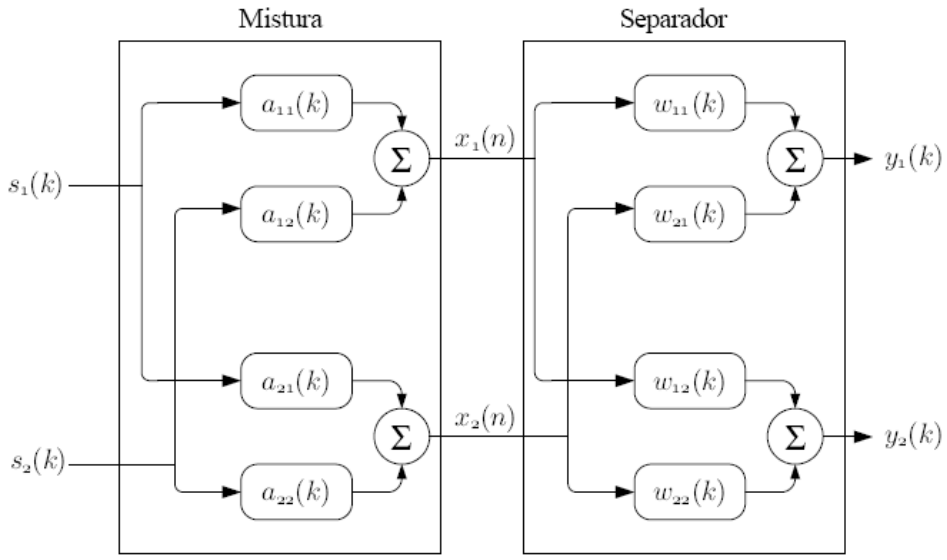
onde A_{ji} denota a resposta ao impulso do filtro correspondente à propagação entre a fonte i e o sensor j e o símbolo * representa a operação de convolução.

A matriz de mistura A em (27) é definida de forma que cada elemento A_{ji} = [$A_{ji}(0)$, $A_{ji}(1)$, ... , $A_{ji}(P-1)$] seja a resposta ao impulso de um filtro FIR (resposta ao impulso finita) de comprimento P.

Já a matriz separadora (ou matriz inversa) W é similar a matriz A, porém contém filtros W_{ij} = [$W_{ij}(0)$, $W_{ij}(1)$, ... , $W_{ij}(Q-1)$] separadores, de comprimento Q e tem dimensão NxM. Depois de estimada a matriz, cada fonte recuperada $\mathbf{Y_i(n)}$ é encontrada da seguinte forma:

$$\boldsymbol{Y_i}(n) = \sum_{i=1}^{M} [\sum_{l=0}^{Q-1} W_{ij}(l) X_j(n-l)] \qquad \textbf{(28)}$$

A separação, neste caso, pode ser obtida através de um conjunto de filtros W_{ij} de maneira que a estimativa das fontes também resulta de um processo de filtragem.



Extraído de: Suyama,2007 [8]

Figura 4: Mistura convolutiva com 2 fontes e 2 sensores

Note pela figura 4 que o modelo de mistura, assim como o sistema separador, são basicamente filtros com múltiplas entradas e múltiplas saídas (MIMO - Multiple Input – Multiple Output). Pode-se utilizar a notação matricial para representação do sistema, assim como nas equações 4 e 6, simplesmente alterando-se as matrizes A e W.

Existem vários métodos propostos na literatura que permitem estimar a matriz W e/ou extrair as fontes conhecendo-se somente as misturas **X(n)** . Uma questão de extrema relevância diz respeito às condições sob as quais é possível inverter perfeitamente o processo de mistura por meio de uma estrutura linear. Matematicamente, queremos determinar sob quais condições é possível obter um filtro W[z] tal que

$$W[z].A[z] = I \qquad \textbf{(29)}$$

Esta condição é conhecida no contexto de comunicações como condição zero-forcing, e W[z] é denominado equalizador zero-forcing [8].

No contexto de filtros MIMO, a condição de inversibilidade do sistema é definida através do posto da matriz A[z] [8].

Teorema 1: Seja A[z] uma matriz polinomial representada através da sua transformada z $A[z] = A(0) + A(0)z^{-1} + ... + A(P-1)z^{-P+1}$. Existirá uma matriz polinomial W[z] tal que

W[z]A[z] = I se somente se o posto de A[z] for completo, para qualquer valor de z na circunferência de raio unitário, |z|=1.

O teorema assegura a existência de uma matriz linear capaz de inverter a mistura. No entanto, não garante que a estrutura possua resposta ao impulso finita, ou FIR, tampouco que será causal.

Em alguns casos é possível inverter um sistema MIMO-FIR com outra estrutura de mesma natureza. De fato, isso ocorre para os casos enunciados no seguinte teorema, que corresponde à Identidade de Bezout [8].

Teorema 2: Seja A[z] uma matriz polinomial retangular, tal que o posto de A[z] seja completo para qualquer z incluindo ∞. Então existe uma matriz polinomial

$$W[z] = \sum_{l=0}^{Q-1} W(l) z^{-l} \qquad \textbf{(30)}$$

com Q finito, tal que W[z].A[z] = I.

Um fato observado em matrizes polinomiais com mais linhas do que colunas, o que representaria uma mistura com mais sensores do que fontes, é que, a menos de alguns casos restritos, a condição expressa pelo teorema 2 é quase sempre satisfeita [8]. O mesmo, no entanto, não ocorre para o caso em que as matrizes são quadradas.

Neste trabalho, estudamos algumas técnicas para separação de misturas convolutivas no domínio temporal e também no domínio da frequência. Tais métodos serão descritos a seguir.

2.3.1 Fast-ICA Convolutivo

O estudo foi iniciado pela implementação de técnicas no domínio temporal. Assim sendo, consideramos o artigo [1] no qual os autores estendem a técnica do Fast-ICA, já descrito para misturas instantâneas em 2.2.8, para o caso das misturas convolutivas.

O Fast-ICA Convolutivo é uma técnica de Separação Cega de Fontes (em inglês, Blind Source Separation ou BSS) uma vez que tem-se apenas disponíveis os sinais dos sensores ou misturas observadas (vetores observações **X(n)**). Nada se sabe sobre as fontes originais ou sobre o ambiente no qual elas estão inseridas.

O algoritmo aqui apresentado trabalha no domínio temporal, utilizando a não-gaussianidade como base para definição do critério que permitirá recuperar as fontes

misturadas conforme discutido na seção 2.2.8. De fato, é um algoritmo de ponto-fixo que utiliza a curtose ou a negentropia para maximizar a não-gaussianidade dos sinais observados.

O primeiro passo com o objetivo de estender o Fast-ICA para o caso de misturas convolutivas é abordar o processo de pré-branqueamento dos dados. Para isto, [1] propõe escrever o vetor de misturas da seguinte forma:

$$\widetilde{\mathbf{X}}(\mathbf{n}) = [\mathrm{X}_1(\mathrm{n}-\mathrm{R}), \dots, \mathrm{X}_1(\mathrm{n}+\mathrm{R}), \dots, \mathrm{X}_\mathrm{M}(\mathrm{n}-\mathrm{R}), \dots, \mathrm{X}_\mathrm{M}(\mathrm{n}+\mathrm{R})]^\mathrm{T} \quad \textbf{(31)}$$

onde R é um parâmetro que definirá o tamanho dos filtros da matriz W em um passo posterior. Note que $\widetilde{\mathbf{X}}(\mathbf{n})$ tem um tamanho que pode ser definido por V=(2R + 1)*M. Assim, o vetor $\grave{\mathbf{X}}(\mathbf{n}) = [\grave{\mathbf{X}}_\mathbf{1}(\mathrm{n}), \dots, \grave{\mathbf{X}}_\mathbf{V}(\mathrm{n})]^\mathrm{T}$ pode ser definido como:

$$\grave{\mathbf{X}}(\mathbf{n}) = \hat{B}\,\widetilde{\mathbf{X}}(\mathbf{n}) \qquad \textbf{(32)}$$

Onde $\hat{B}$ é uma matriz V x V escolhida tal que:

$$E\left[\grave{\boldsymbol{X}}_{\boldsymbol{i}}(n)\grave{\boldsymbol{X}}_{\boldsymbol{j}}(n)\right] = \delta_{ij}\,, \forall_{i,j} \in [1,\dots,V] \qquad \textbf{(33)}$$

Com relação às etapas citadas acima, a operação (32) pode ser considerada como branqueamento convencional, o que consiste da análise de componentes principais e normalização. Agora, com respeito ao vetor observação original **X(n)**, podemos interpretar diferentemente. As equações (31) e (32) mostram sem dúvida que os sinais $\grave{\mathbf{X}}(\mathbf{n})$ são misturas convolutivas de **X(n)** . Já a equação (33) demonstra que os sinais $\grave{\mathbf{X}}(\mathbf{n})$ são criados para terem variâncias unitárias e para terem descorrelação mútua, o que pode ser visto como um branqueamento espaço temporal e normalização das observações $\grave{\mathbf{X}}(\mathbf{n})$.

O algoritmo Fast-ICA, ou algoritmo de ponto fixo, é uma versão rápida, como o próprio nome já diz, para se encontrar o gradiente da negentropia (ou curtose) e busca-se maximizar o valor absoluto dessa ao realizar a derivação com respeito a matriz W. Especificamente, [1] começa propondo uma extensão de [2], ou seja, do algoritmo de ICA curtótico convolutivo de ponto-fixo rápido baseado na matriz W. Ele é capaz de encontrar com maior eficiência o ponto de maximização dos valores da função aplicada para a estimativa do algoritmo.

Assim, considerando primeiramente o critério baseado na curtose, a equação de adaptação do vetor W é dada por:

$$1)\ \mathrm{W} = \mathrm{E}\left[\dot{\mathbf{X}}\left(\mathrm{W}^{\mathrm{T}}\dot{\mathbf{X}}\right)^{3}\right] - 3\mathrm{W} \qquad \textbf{(34)}$$

$$2)\ \mathrm{W} = \frac{\mathrm{W}}{||\mathrm{W}||} \qquad \textbf{(35)}$$

onde os filtros que compõe os coeficientes da matriz W são iniciados aleatoriamente. Em seguida, repete-se os passos 1) e 2) acima quantas vezes forem necessárias a fim de atualizar os pesos até a convergência. Em geral, poucas iterações já são suficientes.

Outra função contraste que também permite avaliar a não-gaussianidade é a negentropia foi discutida em 2.2.7. Tal função tem mostrado render melhor robustez e variância mais baixa que a curtose aproximada [1]. Em particular, o critério negentrópico é mais robusto para valores extremos do que o critério curtótico, o qual envolve momentos de quarta ordem, cuja estimativa é mais sensível a discrepâncias (outliers).

A equação de adaptação que define o caso negentrópico é:

$$\mathrm{W} = \mathrm{E}[\dot{\mathbf{X}}\, \mathrm{g}\left(\mathrm{W}^{\mathrm{T}}\dot{\mathbf{X}}\right)] - \mathrm{E}[\dot{\mathbf{X}}\, \mathrm{g}'\left(\mathrm{W}^{\mathrm{T}}\dot{\mathbf{X}}\right)]\mathrm{W} \qquad \textbf{(36)}$$

A utilização de diferentes funções e específicas para cada algoritmo é de primordial valia uma vez que cada algoritmo explora a não-linearidade dessas funções a fim de obter o resultado desejado. A função g, não-linear, empregada na etapa descrita e que rendeu melhor desempenho foi $g(u) = \exp(-x^2/2)$ [1].

Tendo estimado W através de (34) e (35) ou (36) permite-se obter uma estimativa de uma das fontes, a menos de um fator de escala e um fator de atraso. O vetor $\mathbf{Y_i(n)}$ é extraído através da equação (37).

$$Y_i(\mathrm{n}) = \mathrm{W}^{\mathrm{T}}\dot{\mathbf{X}}(n) = \sum_{m=1}^{V} W_m\, \dot{\mathbf{X}}_{\boldsymbol{m}}(n) \qquad \textbf{(37)}$$

onde W é uma matriz composta por V entradas de coeficientes W_m que, juntos com (32), resulta em uma combinação convolutiva $\mathbf{Y_i(n)}$ das observações.

Realizando tais procedimentos e considerando todas as suposições iniciais já comentadas em tópicos anteriores desse trabalho, temos que rememorar que o objetivo principal da técnica BSS, no caso convolutivo, é tipicamente estimar a contribuição de todas as fontes em cada observação. Esse método chamado de Método Baseado em Deflação, consiste em subtrair cada fonte recuperada das

misturas **X(n)** e rodar o algoritmo novamente, de forma a recuperar uma outra fonte, diferente da primeira já identificada. Para isto, após a recuperação da fonte, precisamos identificar qual a contribuição do sinal $\mathbf{Y_j(n)}$ em cada mistura $\mathbf{X_i\ (n)}$. Os filtros que permitem executar tal tarefa são chamados de filtros de coloração e denotaremos por $\mathbf{C_{ij}(n)}$ o filtro que representa a contribuição da i-ésima fonte na j-ésima mistura.

A minimização do erro médio quadrático resulta nos conhecidos filtros de Wiener [10].

$$\mathrm{C_{ij}} = \mathrm{R}_{Y_j}^{-1}.\,\mathrm{r_{Y_jX_i}} \qquad \textbf{(38)}$$

onde R_{Y_j} é a matriz de autocorrelação do sinal E[$\mathbf{Y_j(n)}$.$\mathbf{Y_j(n)}$] e, $\mathbf{r_{Y_jX_i}}$ é o vetor de correlação cruzada entre E[$\mathbf{Y_j(n)}$.$\mathbf{X_i(n)}$].

Os filtros de coloração são obtidos buscando-se minimizar o erro médio quadrático entre $\mathbf{X_i(n)}$ e $\mathbf{C_{ij}(n)}$*$\mathbf{Y_j(n)}$**.** Esse filtro não-causal é calculado através da equação 38 mostrada logo a seguir**.** A condição de não-causalidade é empregada devido ao fato que o sinal **Y(n)** pode estar atrasado com relação as fontes **S(n)** .

Ainda aplicando o método de deflação, a próxima etapa consiste em subtrair os valores obtidos das contribuições $\mathbf{C_{ij}(n)}$*$\mathbf{Y_j(n)}$ de todas as M-ésimas observações, obtendo outra configuração de mistura com N-1 fontes.

Assim sendo, caso N ǂ 1, existirá mais fontes a serem recuperadas, então as etapas descritas nas equações de 31 à 35 devem ser repetidas. Para facilitar, podemos enumerar os passos a serem seguidos:

1) *Identificar os vetores $\boldsymbol{Y_i(n)}$ de uma fonte $\boldsymbol{S_i(n)}$ a partir das observações **X(n).***
2) *Identifica-se os M filtros de coloração e aplica-os ao vetor $\boldsymbol{Y_j(n)}$ a fim de recuperar as contribuições de $\boldsymbol{S_i(n)}$ em cada observação, isto é, $\boldsymbol{X_j(n)}$.*
3) *Subtrai-se as contribuições de todas as observações.*
4) *Seleciona-se N ← N – 1. Caso N ǂ 1, retorna-se para o passo 1) a fim de extrair a outra fonte.*

2.4 Domínio da frequência

Em se tratando de métodos para separação cega de fontes podemos também citar a separação no domínio da frequência.

Em um ambiente real, como já salientamos, os sinais de áudio são convoluídos com a resposta ao impulso do filtro empregado, de tal forma que isso representa o caminho entre a fonte e os microfones. Assim sendo, cada elemento da matriz de separação compõe um filtro FIR. Sabemos que uma operação de convolução no domínio do tempo representa uma multiplicação no domínio da frequência, obtida através da aplicação da Transformada de Fourier. Tendo uma simples multiplicação entre a matriz de mistura e as fontes, é possível contornar o problema da mistura convolutiva no domínio do tempo passando para uma mistura instantânea no domínio da frequência. Assim, a técnica de separação de misturas instantâneas baseada em ICA mostradas na seção 2.2, podem ser aplicadas no domínio da frequência, e posteriormente retorna-se ao domínio do tempo através da operação da Transformada Inversa de Fourier. A vantagem de tal procedimento é a redução considerável do custo computacional para execução do algoritmo.

Um ponto bastante importante a ser considerado é o fato de que as ambiguidades de permutação e escala inerentes à solução de BSS, no domínio da frequência precisam ser resolvidas para recuperação plena das fontes. Assim sendo, veremos que cada algoritmo resolve essa questão propondo uma resolução diferente.

Cada frequência gera componentes independentes a serem agrupadas para uma mesma fonte antes de aplicarmos a Transformada Inversa de Fourier. Assim, é de extrema importância corrigir a escala e garantir a ordem de recuperação das fontes, sendo essencial para que o sinal seja obtido corretamente após a operação da Transformada Inversa.

Uma visão abrangente do algoritmo de separação de fontes do domínio da frequência pode ser vista conforme mostra a figura 5 abaixo.

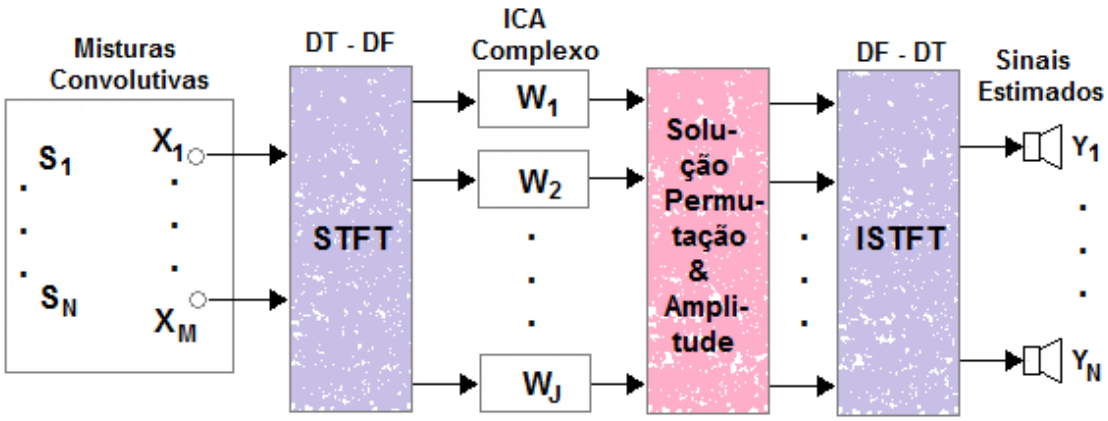


Figura 5: Diagrama do domínio da frequência BSS

O primeiro passo é transformar cada um dos sinais X_j (n), j=1,...,M para o domínio da frequência X_j (ω,t), com ω = 0,..., ω -1 (onde w é o índice em frequência e t é o índice temporal de cada quadro), utilizando para tal a Transformada de Fourier de Tempo-Curto (STFT – Short Time Fourier Transform).

O fluxo BSS no domínio da frequência é mostrado conforme denota a figura 5. Usando a Transformada de Fourier de Tempo-Curto, os sinais observados no domínio do tempo são transformados em sinais no domínio da frequência através da operação abaixo:

$$X_i(\omega, t) = \sum_{K=0}^{K-1} X_j(t+k)\, w(k) e^{-j\omega k/K} \qquad \textbf{(39)}$$

Onde w(k) denota uma função janela.

Após esse passo é realizado o pré-processamento, o que efetivamente, consiste no branqueamento dos sinais, gerando sinais branqueados X_{bj} (ω,t).

Uma desvantagem da utilização da Transformada de Fourier, é que na transformação para o domínio da frequência a informação do tempo é perdida, exceto no caso do sinal ser estacionário (mesma componente em frequência ao longo do tempo). Pelo fato dos sinais possuírem inúmeras características transitórias, ainda mais se tratando de sinais de áudio por exemplo, a Transformada de Fourier de Tempo-Curto foi adotada como forma de contornar essa situação. Desse modo, a STFT utiliza uma pequena porção do sinal de cada vez. É o que chamamos de "janela deslizante sobre o sinal". Essa técnica trabalha com o sinal numa função de duas dimensões de base tempo e frequência. No entanto, essa informação obtida somente terá boa precisão caso o tamanho e tipo de janela escolhidas estiverem de acordo e a janela deve ser fixa para todas as frequências trabalhadas.

A separação no domínio da frequência supõe que os sinais são independentes para cada frequência empregada. As técnicas de BSS no domínio da frequência estimam a matriz B(ω) para cada componente em frequência ω, e o vetor com as saídas separadas $Y(\omega,t) = [Y_1(\omega,t),..., Y_N(\omega,t)]^T$, ambos permutados e escalados.

A equação que define como o sinal estimado Y (ω,t) é encontrado encontra-se a seguir:

$$Y(\omega,t) = B(\omega).\ X(\omega,t) \qquad \textbf{(40)}$$

onde B é a matriz de separação na frequência ω e $Y_i(\omega,t)$, i=1,...,N.

Após a solução do problema para todas as frequências ω, são então resolvidos os problemas de permutação e escala.

Na última etapa, os sinais de tempo-frequência são reconstruídos usando a Transformada Inversa de Fourier no Tempo-Curto. Ela é utilizada para converter os sinais estimados das fontes do domínio da frequência em sinais no domínio do tempo novamente.

$$y_i = \frac{1}{K}\sum_{K=0}^{K-1} Y_i(\omega, t)e^{j\omega k/K} \qquad \textbf{(41)}$$

Visto que podemos aplicar vários algoritmos utilizados inicialmente no contexto de misturas instantâneas (seção 2.2) no domínio da frequência, descrevemos na sequência como se dá a transição do contexto instantâneo para o tratado aqui.

2.4.1 Fast-ICA revisitado

Para cada frequência ω os sinais misturados em tempo-frequência são separados de forma independente, portanto, as premissas de análises de componentes independentes, ou ICA, são utilizados amplamente para esse problema. Assim sendo, sabe-se sobretudo que o ICA explora o fato da independência estatística entre os sinais das fontes originais, a fim de separá-los a partir dois sinais de misturas, tentando fazer os sinais mais independentes possíveis um do outro. Essa consideração já é conhecida para o nosso estudo de caso e, por isso mesmo, quanto mais as fontes de sinais forem não-gaussianas e, mutualmente independentes, melhor dar-se-á a separação.

A extensão do Fast-ICA para aplicação no contexto do domínio em frequência estudado aqui foi proposto em [1] e tem como principal diferença com relação ao algoritmo apresentado em 2.2.8, o fato de agora temos que lidar com dados complexos. A equação de adaptação segue abaixo:

$$\widetilde{\mathbf{B}}_{\mathrm{i}}(\omega) = \mathrm{E}\{\mathrm{X}(\mathbf{B_i}(\omega)\mathrm{X})^{*}\mathrm{g}(|\mathbf{B_i}(\omega)\mathrm{X}|^2)\} - \mathrm{E}\{\mathrm{g}(|\mathbf{B}_{\mathrm{i}}(\omega)\mathrm{X}|^2) + |\mathbf{B_i}(\omega)\mathrm{X}|^2\mathrm{g}'(|\mathbf{B_i}(\omega)\mathrm{X}|^2)\}\mathbf{B_i}(\omega)$$

$$\mathbf{B_i}(\omega) = (\widetilde{\mathbf{B}}_{\mathbf{i}}(\omega)/\widetilde{\mathbf{B}}_{\mathrm{i}}(\omega)) \quad \textbf{(42)}$$

onde $\mathbf{B_i(\omega)}$ é um vetor que compõe a matriz de separação B(ω). A função não-linear g(.) empregada pelo artigo foi g(u) = log(0.1+u) e g'(.) denota seu diferencial.

A fim de recuperar as fontes de forma satisfatória no domínio temporal, todas indeterminações, tanto de permutação quanto de escala, devem ser resolvidas antes de aplicar a Transformada Inversa de Fourier em Tempo-Curto.

Mesmo realizando a operação de atualização referente à matriz de separação B(ω) (42), restará a ambiguidade.

A fim de sanar o problema de permutação, antes de mais nada deve-se considerar alguns fatores. Quando aplica-se a Transformada de Fourier de Tempo-Curto, os espectros do vetor de observação das fontes mudam ao longo da frequência, o que quer dizer que se as frequências ω são muito estreitas, os espectros vizinhos dessas frequências terão alto grau de correlação. Além disso, pode-se esperar que a matriz de separação obtida pelo Fast-ICA nas frequências adjacentes não serão muito diferentes e portanto a ordem de recuperação também não.

Devido a isso, emprega-se para o cálculo da matriz de separação para calcular a frequência atual. Inicia-se a matriz B(ω) com valores aleatórios uniformemente distribuídos, começando da frequência mais baixa ω_o e, para se calcular $B(\omega_1)$, inicializa-se a matriz com $B(\omega_o)$, conforme ilustrado na figura 6 . Veja essa esquematização na figura 6:

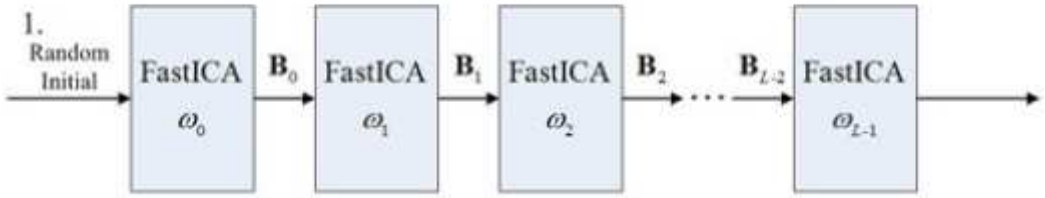


Figura 6: Fluxo de interação nas frequências dadas

Já para corrigir o problema de escala, considerando o número de fontes igual ao número de sensores (N=M), para cada frequência ω_i , obtém-se a inversa de $B(\omega_i)$.

$$C(\omega) = B^{-1}(\omega)$$

$$= \begin{bmatrix} c_{11}(\omega) & \cdots & c_{1M}(\omega) \\ \vdots & \ddots & \vdots \\ c_{M1}(\omega) & \cdots & c_{MM}(\omega) \end{bmatrix}$$

Tomando-se a matriz R como sendo composta por elementos diagonais de C, temos: R(ω) = diag {C(ω)} . Então aplica-se: Y(ω,t) = R(ω).B(ω).X(ω,t) e com isso uma boa reescala pode ser alcançada.[1]

2.4.2 Infomax revisitado

Em [6], os autores mantêm o mesmo esquema mostrado na figura 6 mas ao invés de utilizarem o Fast-ICA como técnica de separação no domínio da frequência, eles utilizam o algoritmo Infomax e Infomax estendido já discutidos nas seções 2.2.2 e 2.2.3, respectivamente.

Mais uma vez é necessário se alterar a equação de adaptação da matriz de separação, aqui também chamada de B(ω), uma vez que os dados são complexos.

Estendendo (16) para o caso tratado aqui chega-se à seguinte relação:

$$\Delta \mathrm{B}(\omega) \propto \mu[\mathrm{B}(\omega)^{-1}]^{\mathrm{H}} - 2 * \mathrm{g}\big(\mathrm{Y}(\omega)\big) * \mathrm{X}(\omega)^{\mathrm{H}}$$

O mesmo vale para o Infomax estendido:

$$\Delta \mathrm{B}(\omega) \propto \mu\big[\, \mathrm{I} - \mathrm{g}\big(\mathrm{Y}(\omega)\big) * \mathrm{Y}(\omega)^{\mathrm{H}}\big] * \mathrm{B}(\omega)$$

Um ponto importante a considerar é a função não-linear que concretiza a convergência para os valores complexos admitidos por B. A função que mostrou-se mais apropriada é g(z) = tanh(Re{z}) + tanh(Im{z})i. [6]

Entretanto não podemos deixar de citar novamente o fato que, ao trabalhar-se no domínio da frequência invariâncias de escalas e permutações devem ser consideradas para obtenção da estimativa das fontes. O artigo propõe remediar esse problema através da seguinte equação:

$$\mathrm{B}_{\omega}^{norm} = \mathrm{B}_{\omega}^{orig} * |B_{\omega}^{orig}|^{-1/N}$$

onde B_{ω}^{norm} e B_{ω}^{orig} são respectivamente a matriz de separação normalizada e original em uma dada frequência ω. Isso irá manter o envelope espectral inalterado enquanto preserva a separação solucionando o problema da invariância de escala.

Já a invariância de permutação é um pouco mais complexa e requer maior atenção. Para uma dada matriz de mistura A com poucos elementos de atraso, não teremos grandes problemas, entretanto, quando temos um cenário com filtros mais complexos, deveremos considerar uma solução para a invariância de permutação. O artigo propõe usar um "fator de influência" k para atualizar a matriz de separação B, a cada frequência utilizada, assim:

$$\Delta_{\mathrm{a}}\mathrm{B}_{\omega} = \Delta_{\mathrm{e}}\mathrm{B}_{\omega} + \mathrm{k} * \Delta_{\mathrm{e}}\mathrm{B}_{\omega-1}$$

Onde $\Delta_{\mathrm{a}}\mathrm{B}_{\omega+1}$ e $\Delta_{\mathrm{e}}\mathrm{B}_{\omega+1}$ são respectivamente a matriz de separação B efetivamente aplicada e a estimada para cada frequência ω, estando o fator de influência no intervalo $0 < k < 1$.

3 Resultados de Simulação

Consideramos aqui o caso de misturas convolutivas e instantâneas obtidas a partir de duas fontes e dois sensores. Consideramos dois casos: fontes aleatórias uniformemente distribuídas e independentes (caso instantâneo e posteriormente para envolvendo o cenário convolutivo) e, fontes dadas por sinais de voz (somente para o caso convolutivo).

3.1 Fontes Uniformemente Distribuídas

3.1.2 Misturas Instantâneas

Como um estudo inicial do algoritmo FastICA para misturas instantâneas, simulamos um caso simples envolvendo misturas contendo duas fontes independentes, com distribuição uniforme entre [-1, +1]. A matriz de mistura A é dada por uma matriz 2x2, escolhida aleatoriamente. Cada uma das fontes foi gerada com um certo número de amostras. Para a obtenção dos valores dependentes do operador esperança, usou-se uma média temporal calculada utilizando-se essa dada quantidade de amostras dos sinais envolvidos.

É importante ressaltar a existência de uma ambigüidade de ganho e de ordem nos resultados obtidos. Assim, os sinais foram comparados graficamente para se resolver a questão da ordem e normalizados para corrigir a amplitude.

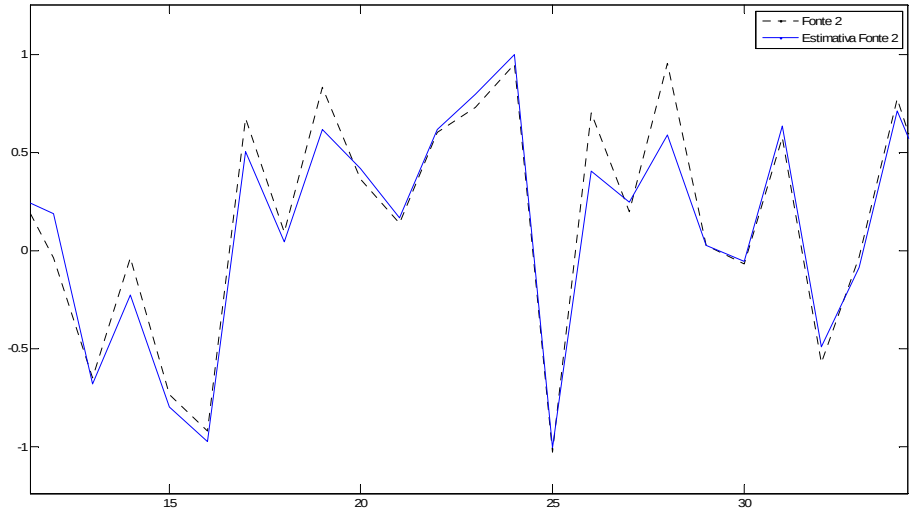


Fig. 7 – Representação contínua da Fonte 2 original (preto) e da estimativa de Fonte 2 sobreposta (azul) após 1 iteração com 80 amostras

Na fig.7 que foi adotado um número de amostras de 80. É perceptível que a fonte 2, representada pela cor preta tracejada e, a estimativa da mesma, representada pela cor azul, não estão totalmente coincidentes, uma vez que existem alguns pontos superiores bem visíveis nos quais apenas pode-se visualizar uma única cor em preto. Para esse caso, foi adotado uma única iteração, a fim de analisar-se inicialmente a robustez do FastICA.

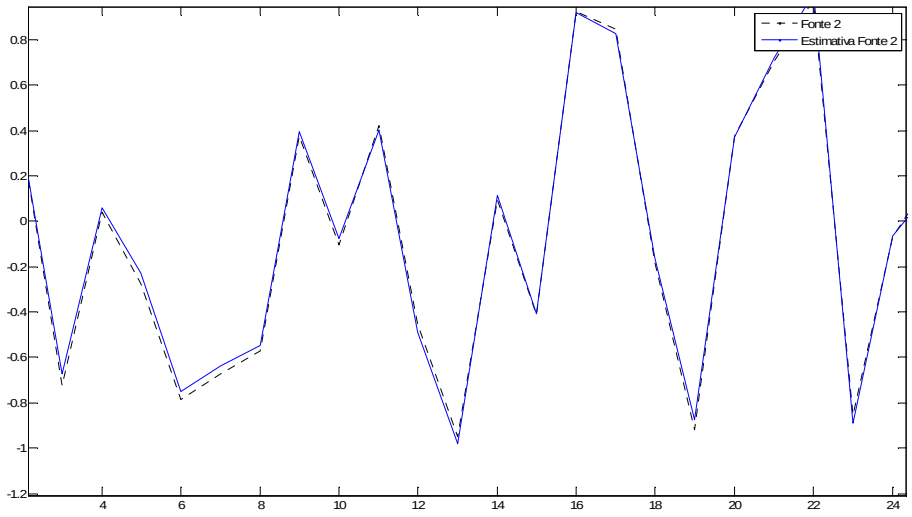


Fig. 9 – Representação tracejada da Fonte 2 original (preto) e da estimativa de Fonte 2 sobreposta (azul) após 6 iterações com 80 amostras

Agora, tomemos a figura 9 com execução das mesmas 80 amostras. Aumenta-se para três ou mais o número de iterações e dessa vez, a convergência ocorre satisfatoriamente.

Ao utilizamos 300 amostras, observou-se que em apenas uma iteração, a convergência não ocorre em certas situações. Uma vez aumentando-se o número de iterações para duas, a convergência ocorre de forma mais satisfatória, porém ainda não totalmente aceitável para um algoritmo como o FastIca (fig. 11). Finalmente aumentado-se para três ou mais iterações, esta dá-se de maneira bastante satisfatória como pode ser visto na figura 12.

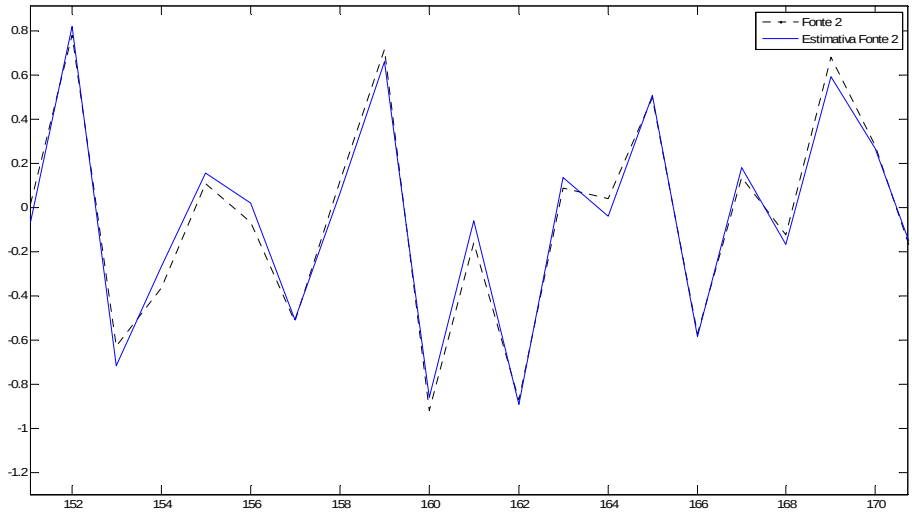


Fig. 11 – Representação tracejada da Fonte 2 original (preto) e da estimativa de Fonte 2 sobreposta (azul) após 2 iterações para 300 amostras

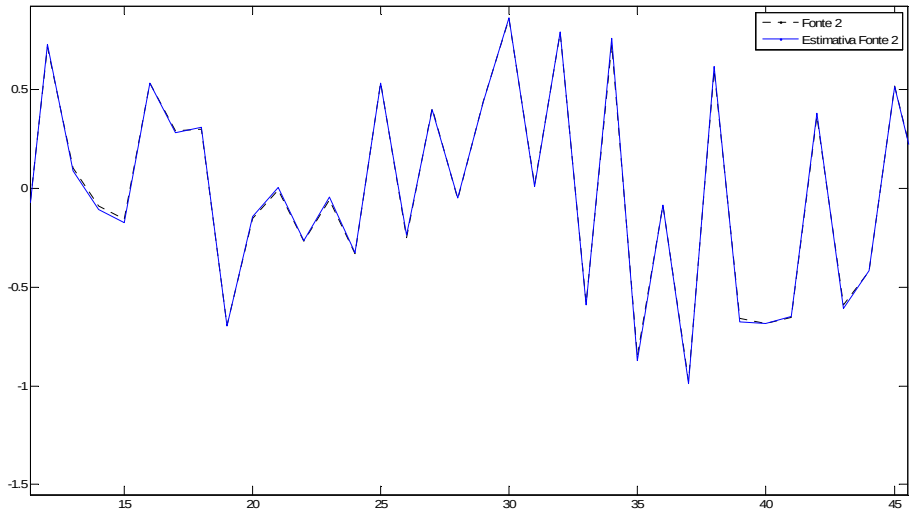


Fig. 12 – Representação tracejada da Fonte 2 original (preto) e da estimativa de Fonte 2 sobreposta (azul) após 3 iterações para 300 amostras

Por fim, a figura 13 mostra o resultado obtido utilizando-se 2000 amostras e uma única iteração. É notável que o resultado também não foi satisfatório apesar de ter sido melhor comparativamente às simulações anteriores em duas iterações. Agora, se houver duas ou mais iterações, sua convergência ocorre de maneira bastante eficaz.

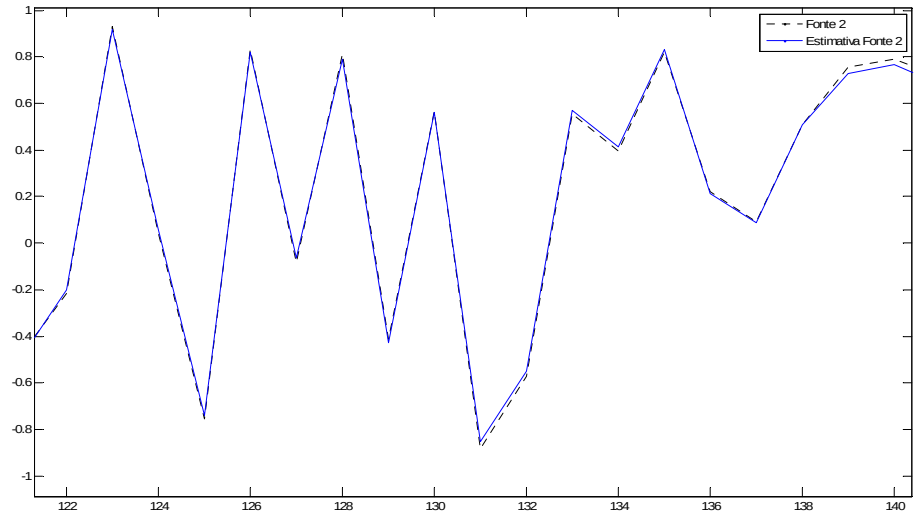


Fig. 13 – Representação tracejada da Fonte 2 original (preto) e da estimativa de Fonte 2 sobreposta (azul) após 1 iteração para 2000 amostras

Três variáveis são de suma importância para inicialmente verificarmos o comportamento do Infomax e mais adiante veremos que essas mesmas três variáveis serão também utilizadas e tão quão importantes no desempenho do Infomax em sua versão Estendida. Consideremos um número total "n" que utilizaremos para definir a quantidade de amostras das fontes. Teremos considerado um tamanho de bloco "p" para o cálculo da estimativa da esperança quando já estamos executando a principal parte do Infomax, que é a equação que rege a atualização da matriz inversa W. Por fim, teremos um valor μ, geralmente pequeno, que define o passo de adaptação para ser utilizado também na execução da equação de atualização de W.

Basicamente a relação entre o número de amostras consideradas, n, e o tamanho do bloco p, ocorre com mesmo grau de inter-relação. Valores de n muito baixos, e valores de p muito menores que o valor de n, influenciarão diretamente no desempenho do algoritmo, como veremos.

O valor ótimo para o passo de adaptação μ difere de uma versão para outra do algoritmo Infomax e logo a seguir realizaremos algumas simulações a fim de observamos a influência das alterações de μ, n e p no desempenho do algoritmo.

É importante ser dito que em alguns casos, quando deseja-se simular o algoritmo com valores já escolhidos e fixados de p, n e μ, o mesmo apresenta diferentes resultados gráficos e conseqüente alteração de desempenho em simulações consecutivas. Para ambas as versões do Infomax isso foi observado. Esse fato deve-se a existência de ambigüidade de ordem no algoritmo e também porque em cada simulação realizada, geram-se distintos valores aleatórios dos sinais das fontes.

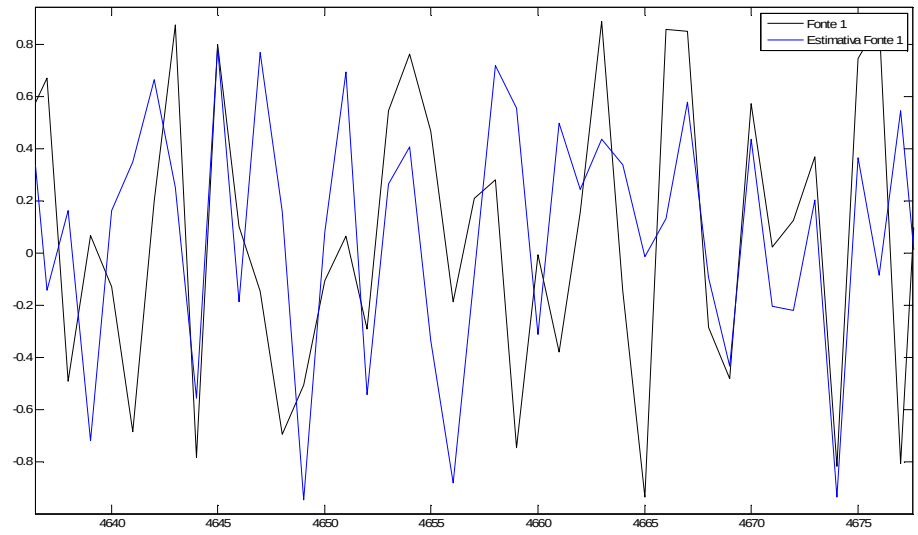


Fig. 14 – Fonte 1 em cor preta e sua estimativa em cor azul (n=20000 e p = 11000)

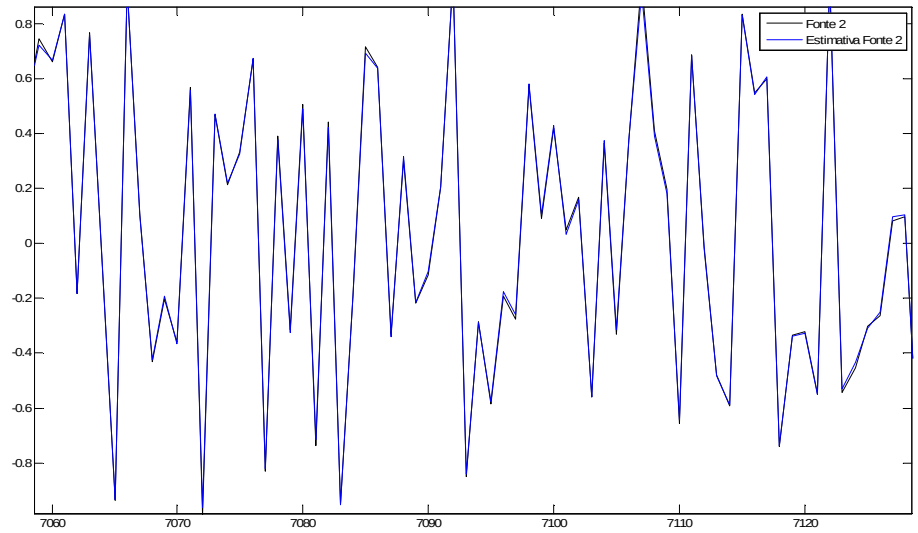


Fig. 15 – Fonte 2 em cor preta e sua estimativa em cor azul (n=20000 e p = 11000)

Através das simulações e resultados obtidos com as simulações anteriores, pode-se obter uma melhor estimativa dos valores iniciais para as três variáveis que influenciam diretamente o desempenho do algoritmo. São elas, p, n e μ. Dessa forma pode-se chegar em uma convergência razoável e satisfatória para ambas as fontes.

Inicialmente utilizamos um valor alto para o número de amostras, ou seja, n = 20.000 e um tamanho de bloco proporcional sendo p = 11.000 para podermos testar os valores de μ que melhor se adaptam a essa primeira simulação. Começamos com um valor de μ = 0.9. Estes valores resultaram em uma convergência ótima para a estimativa da Fonte 2, como pode-se visualizar através da figura 15. Em contrapartida podemos afirmar categoricamente que Fonte 1 não pode ser recuperada.

Como era de se esperar e tal como havíamos observado também nas simulações anteriores, diminuindo o valor do bloco p e não alterando-se as demais variáveis, obtemos como resultado uma convergência insatisfatória para ambas fontes e suas estimativas. Isto foi constatado quando mudou-se p para 2.000 e manteve-se inalterado os valores de μ e n.

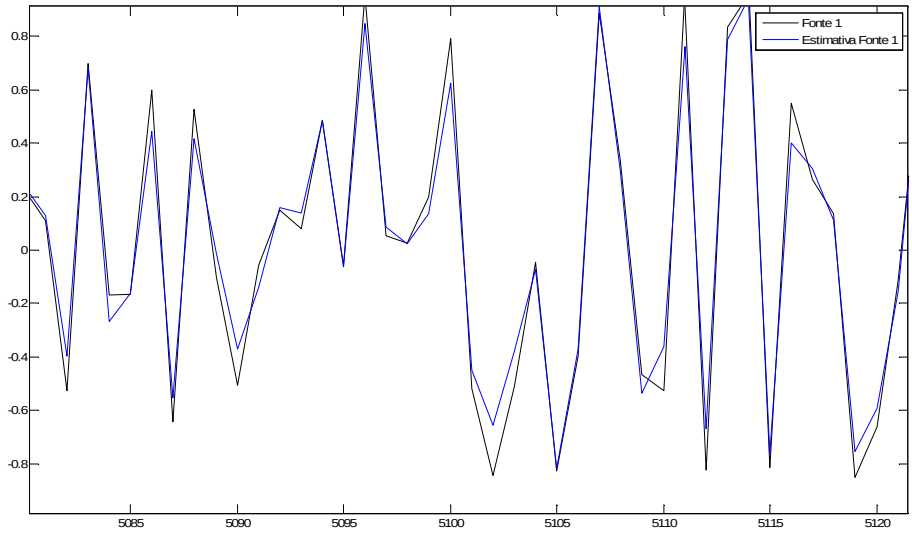


Fig. 16 – Fonte 1 em cor preta e sua estimativa em cor azul (n=22000 e p = 20000)

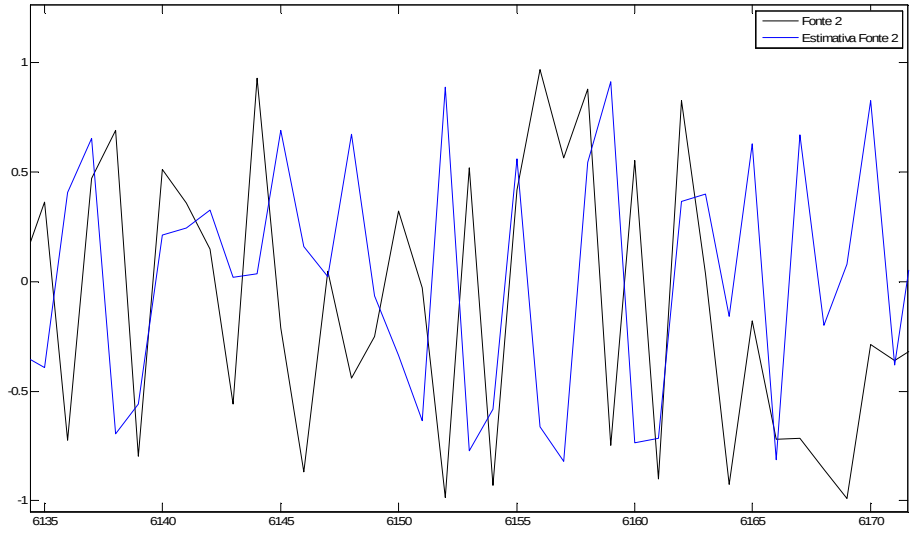


Fig. 17 – Fonte 2 em cor preta e sua estimativa em cor azul (n=22000 e p = 20000)

De agora em diante, centralizamos esforços em encontrar valores das três variáveis que satisfazem a convergência esperada da estimativa de Fonte 1. Adotou-se um número de amostras e tamanho de bloco (n=22.000 e p = 20.000**)** um pouco maior que outrora. Variando-se os valores para μ, decrescendo-o e logo em seguida aumentando-o, encontrou-se um valor aceitável quando notou-se que na maior parte das simulações a estimativa das duas fontes foram satisfatórias. O melhor valor de passo encontrado foi de 0.02. Pode-se observar pela figura 16 que a convergência da estimativa da Fonte 1 foi tida como satisfatória, ao contrário da figura 17 que nos mostra a estimativa da Fonte 2 em azul continuamente descompassada com a Fonte 2 original em preto.

Na descrição do algoritmo Infomax Estendido seremos mais enxutos, pois diversos comentários e observações já feitas anteriormente, também são válidos para este caso.

Basicamente há de se considerar que o Infomax Estendido possui um desempenho inferior comparativamente ao algoritmo FastIca, no sentido de exigir mais do componente de hardware do computador no qual esteja-se simulando, sendo sua convergência relativamente mais lenta e, isso mostrar-se-á na medida em que foram feitas as análises particulares desse algoritmo em sua versão estendida. Seguindo esse raciocínio, é importante também salientar que essa versão estendida do Infomax possui uma execução mais lenta comparativamente à versão inicial Infomax. Em todas as situações, notou-se que os resultados apresentados se deram de maneira efetivamente mais lento, em termos de menor tempo de execução do algoritmo.

As três variáveis consideradas outrora, p, n e μ são utilizadas mais uma vez, porém com valores distintos e isso constataremos mais adiante conforme os resultados de simulação obtidos. Essas três componentes n, p e μ influenciarão no desempenho do Infomax na medida em que alterando-se individualmente seus valores numéricos, teremos como conseqüência uma divergência, convergência razoavelmente satisfatória ou convergência satisfatória do algoritmo e, veremos logo a seguir de que forma isso se manifesta graficamente.

Inicialmente iremos adotar um valor n mediano para o total de amostras e, um valor inferior p para o tamanho do bloco. Não é interessante que p seja do mesmo tamanho do número de amostras devido à necessidade do algoritmo construir blocos consecutivamente menores que o valor atual e trabalhar de forma sobreposta ao bloco anterior quando deseja-se estimar a esperança de maneira mais eficiente.

O passo de adaptação μ será estimado inicialmente com um número pequeno qualquer pois, não sabemos de fato qual será inicialmente seu valor e isso teremos que descobrir testando conforme o algoritmo convirja ou não. Dessa forma, adotamos o parâmetro μ como sendo 5.10^{-4} e, logo em seguida simulamos o algoritmo no software Matlab.

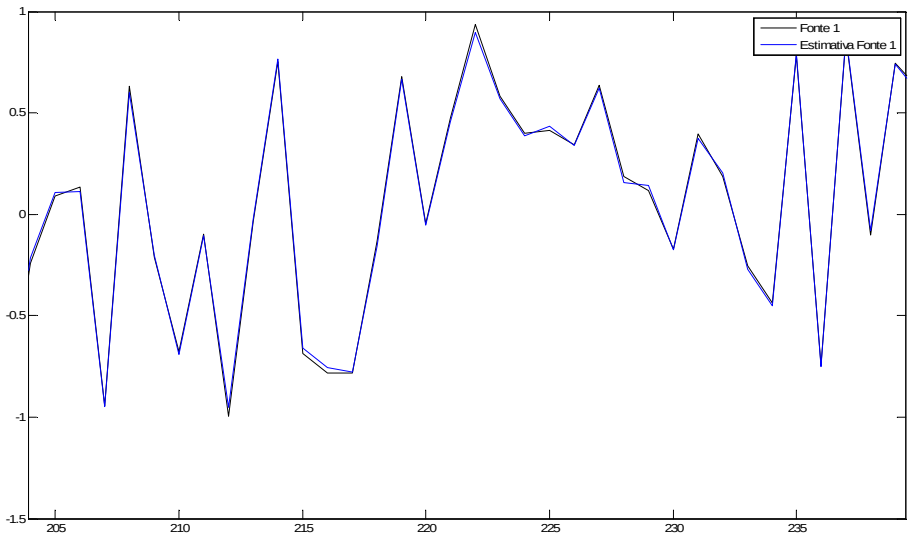


Fig. 18 – Fonte 1 em cor preta e sua estimativa em cor azul (n=2000 e p = 1800)

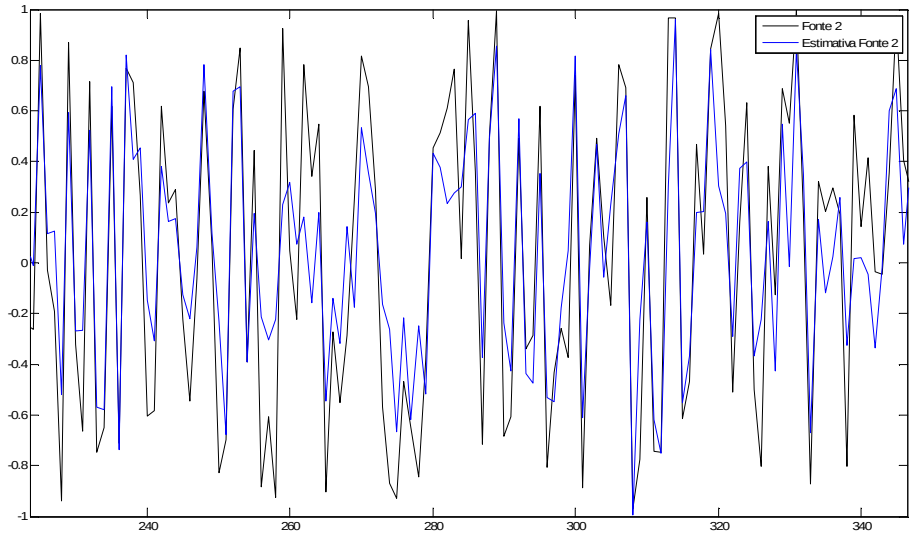


Fig. 19 – Fonte 2 em cor preta e sua estimativa em cor azul (n=2000 e p = 1800)

Iniciamos com n = 2000, p = 1800 e $\mu = 5.10^{-4}$ como já fora dito anteriormente. Ao observarmos a figura 18, nota-se claramente que a convergência deu-se de forma satisfatória uma vez que as linhas preta e azul se sobrepuseram em quase todo o intervalo considerado, exceto em alguns trechos bem pequenos como, por exemplo, entre as amostras 215 a 220 e também entre as amostras 227 a 230. Já quando visualizamos a figura 19, é notório que houve convergência insatisfatória, pois quase todo o intervalo considerado as linhas preta e azul não estão coincidentes, salvo pequenos trechos contínuos. No entanto, verifica-se uma tendência de convergência.

Agora vejamos o que acontece quando fixamos o valor de μ e o valor de n, e alteramos o valor do bloco p.

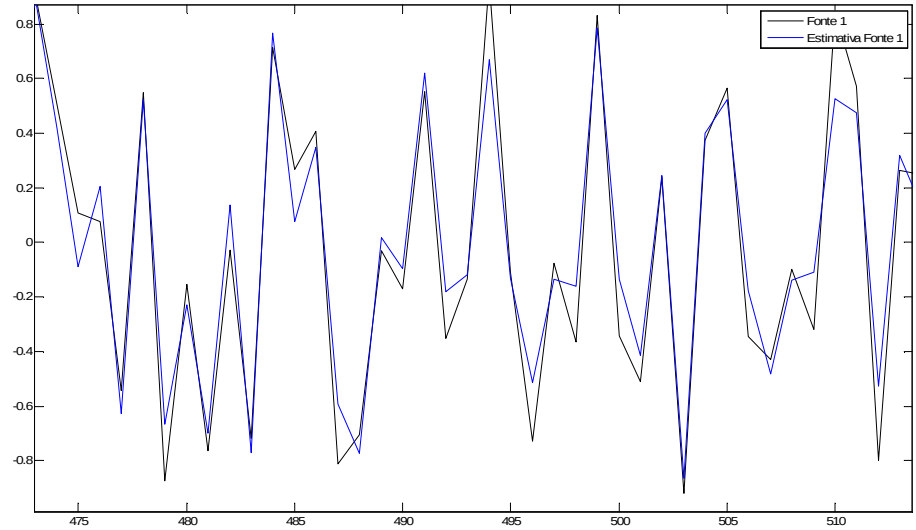


Fig. 20 – Fonte 1 em cor preta e sua estimativa em cor azul (n=2000 e p = 1400)

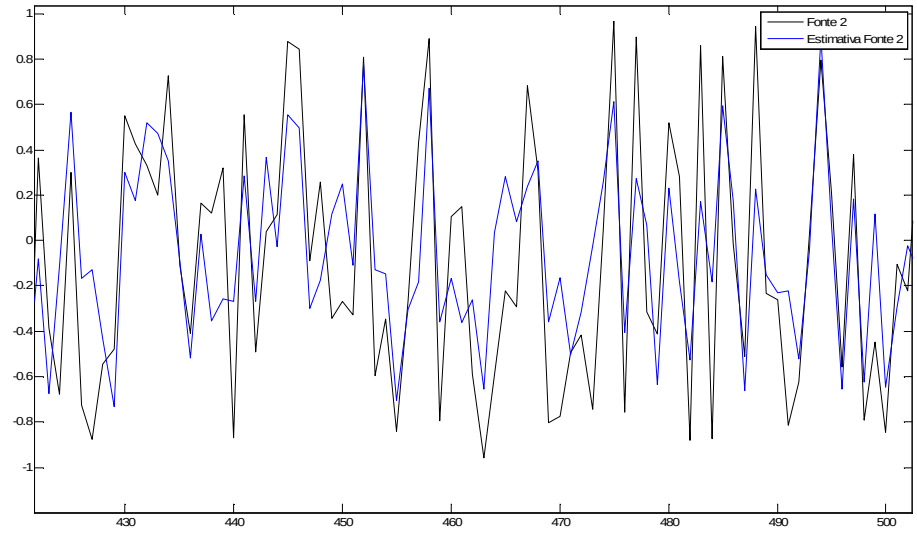


Fig. 21 – Fonte 2 em cor preta e sua estimativa em cor azul (n=2000 e p = 1400)

Alterando apenas o valor de p para 1400 e deixando n = 2000 e $\mu = 5.10^{-4}$, temos um comportamento gráfico distinto para ambas as fontes e suas estimativas.

Na figura 20 temos a Fonte 1, representada pela cor preta e sua estimativa, representada pela cor azul, com alguns trechos inferiores e superiores do gráfico em que elas não são coincidentes, apesar de ter havido convergência, essa deu-se de maneira um tanto insatisfatória. Diferentemente da figura anterior, a figura 21 nos

mostra que a Fonte 2 em preto e sua estimativa em azul, não tiveram convergência alguma e isso está bem claro no gráfico dessa figura.

Se continuarmos mantendo os valores fixados de n e μ e decrescendo o valor de p, essa tendência de convergência insatisfatória para ambas as fontes se acentua cada vez mais. Veja exemplos das figuras 22 e 23, onde utilizamos um valor de p = 900 e mantendo os valores de n = 2000 e $\mu = 5.10^{-4}$.

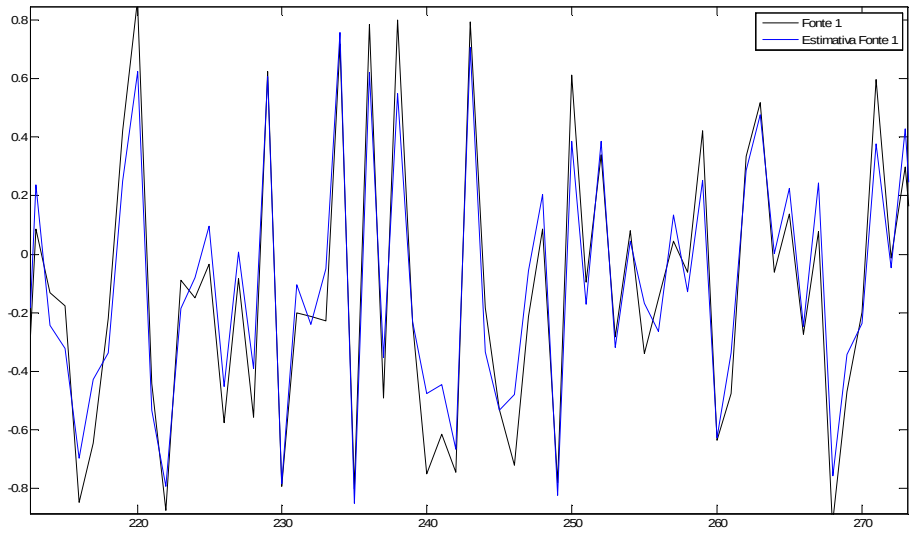


Fig. 22 – Fonte 1 em cor preta e sua estimativa em cor azul (n=2000 e p = 900)

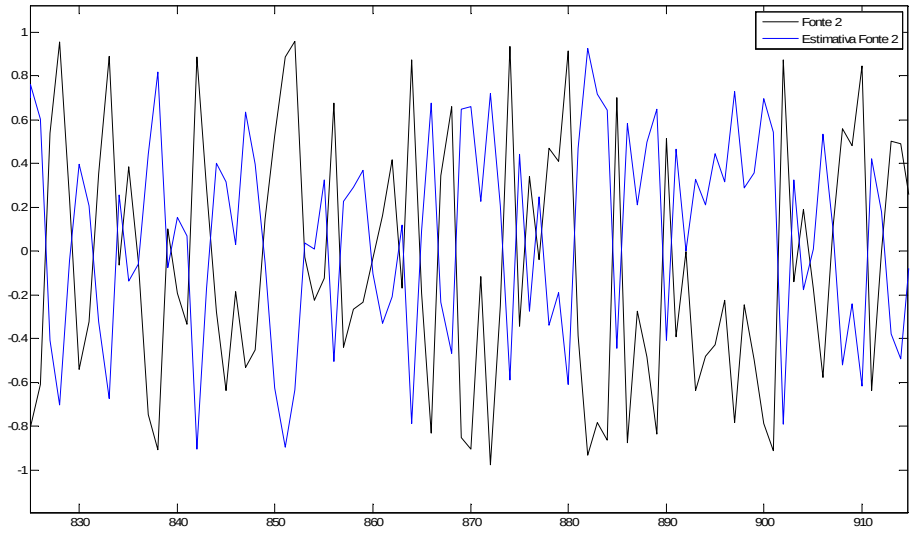


Fig. 23 – Fonte 2 em cor preta e sua estimativa em cor azul (n=2000 e p = 900)

Agora já podemos concluir que o tamanho do bloco p é crucial para o desempenho do algoritmo. Isso foi observado também quando aumentamos o valor do número de amostras n para 10.000 e mais tarde para 15.000. Mantendo-se ainda um valor de μ em $5.10^{-4,}$ alteramos o valor de p, começando com valores altos e diminuindo gradativamente tal como fora feito na análise das figuras acima. Para valores altos de p, próximos do número total de amostras, temos uma convergência tida como razoável e, conforme vamos decrescendo o valor de p, obtêm-se convergência cada vez menos aceitável.

O que faltou analisarmos foi a influência do passo de adaptação μ no algoritmo e também o porquê da Fonte 2 não ter convergido em nenhuma das análises feitas. Isso foi o motivo para testarmos alguns valores μ aleatoriamente e ver o que teríamos como resultado através da observação e análise gráfica. Alguns valores de μ fizeram com que o algoritmo não convergisse, tanto para a Fonte 1 quanto para a Fonte 2. Após sucessivas tentativas, foi encontrado um valor de μ = 0.9 , em que para quaisquer valores de amostragem n e valores de bloco p, houve convergência satisfatória. Obviamente o número de amostras e o tamanho do bloco não podem ser pequenos demais, pois dessa forma não teremos dados suficientes para a execução satisfatória do Infomax Estendido, mas ademais, o valor de μ encontrado foi tido como ótimo para todas as simulações variando-se n e p, além de resolver a questão da não convergência da Fonte 2.

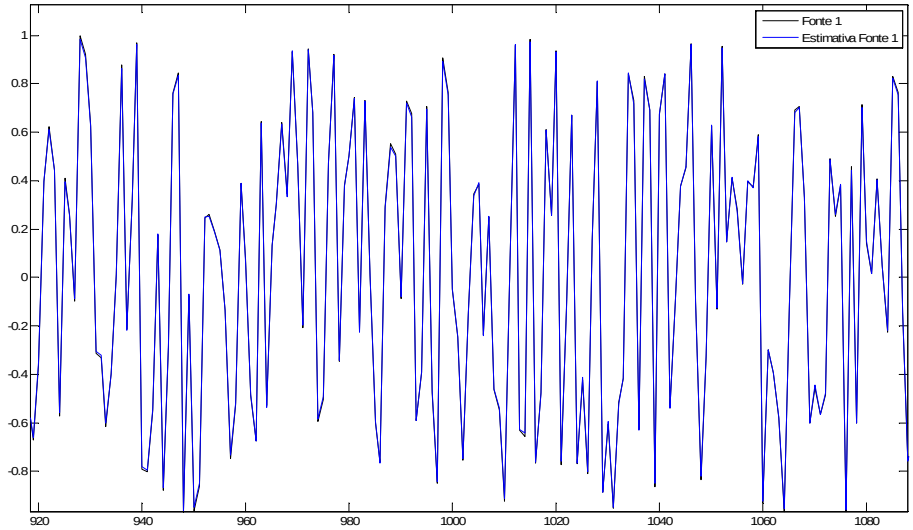


Fig. 24 – Fonte 1 em cor preta e sua estimativa em cor azul (n=12000 e p = 1000)

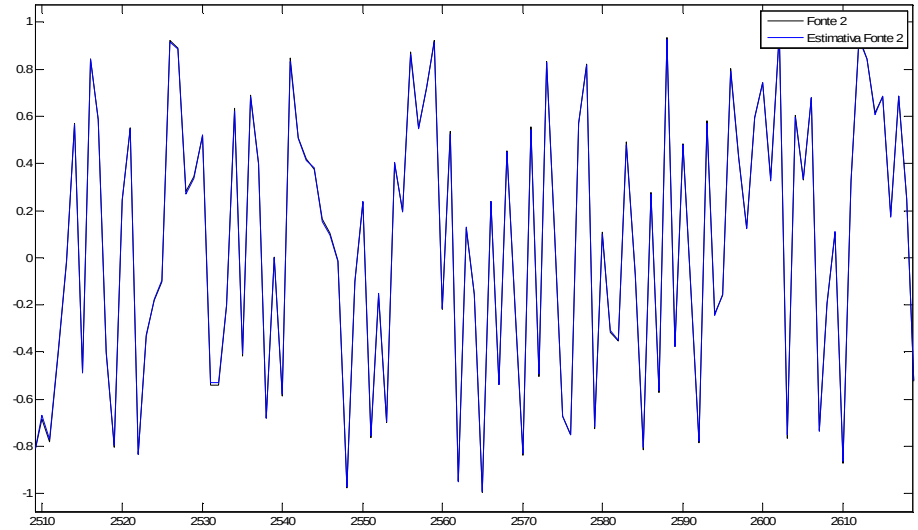


Fig. 25 – Fonte 2 em cor preta e sua estimativa em cor azul (n=12000 e p = 1000)

Na figura 24 e 25 temos um total de 12.000 amostras com tamanho de bloco de 1.000 e μ = 0.9. Tanto para a Fonte 2 quanto para a Fonte 1, as estimativas estão sobrepostas em todo o intervalo considerado, não podendo-se distinguir qual é qual. É importante salientar que as simulações efetuadas com esse total de 12.000 amostras e decrescendo o valor do bloco gradativamente tal como feito anteriormente e fixado o valor de μ em 0,9, temos a representação gráfica da Fonte 1 e Fonte 2 convergindo tal como mostrados nas figuras 24 e 25.

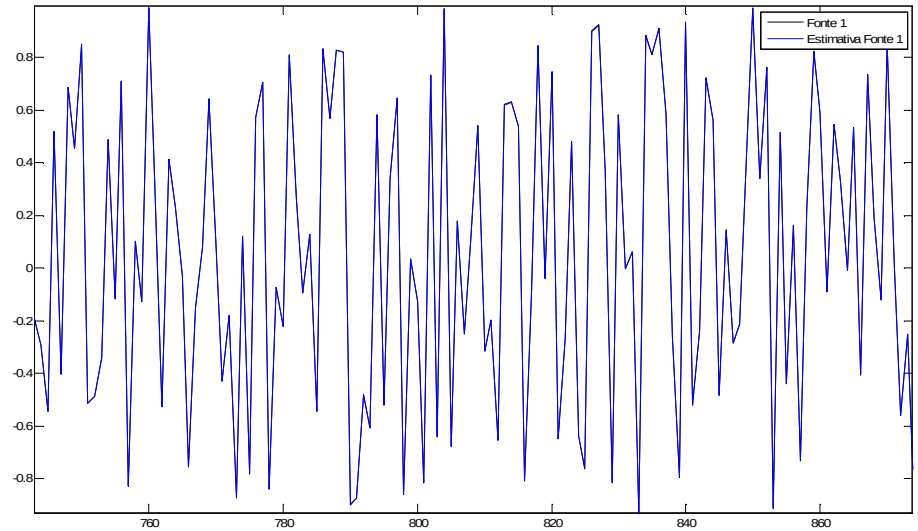


Fig. 26 – Fonte 1 em cor preta e sua estimativa em cor azul (n=4000 e p = 2000)

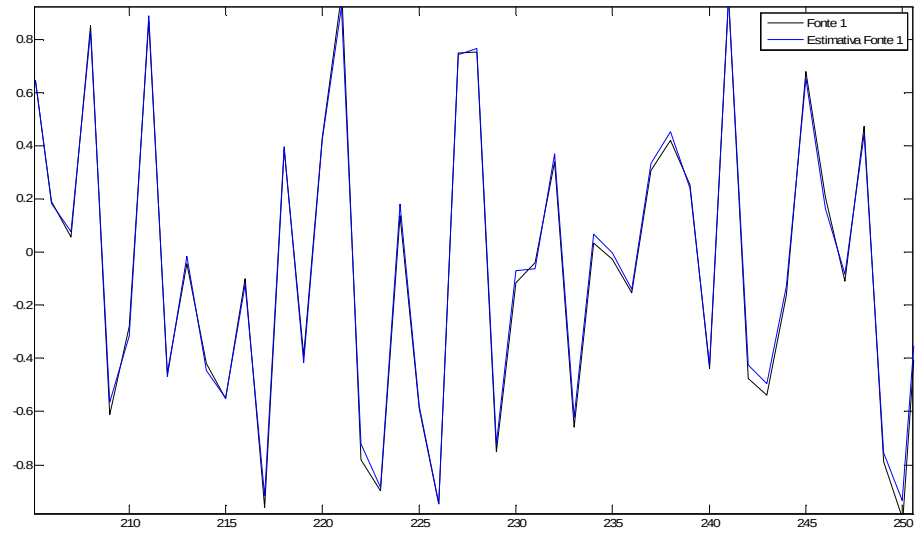


Fig. 27 – Fonte 1 em cor preta e sua estimativa em cor azul (n=1000 e p = 400)

Para finalizar mostramos alguns gráficos tendo μ como o valor encontrado de 0.9. Apresentamos as figuras 26 e 27 como exemplo de que mesmo diminuindo a amostragem e valor equivalente de bloco, temos convergência satisfatória. Interessante pautar também, que quando olhamos para a figura 27, nota-se que essa convergência já torna-se menos satisfatória comparativamente a figura 26, justamente porque o valor considerado de amostras n = 1.000, já está se aproxima de um limiar tido como baixo para execução do Infomax Estendido.

Finalmente para tratar do algoritmo Gradiente Natural, devemos também informar um valor para o número de amostras n, o passo de adaptação μ e, por fim o tamanho do bloco p. As devidas precauções que havíamos tomado anteriormente para configurar o valor inicial para cada uma dessas variáveis, aqui não é diferente. Por exemplo, não iremos iniciar com um valor para n muito baixo, nem alto demais, muito menos configurar um valor de p acima do valor já utilizado para n pois não faria sentido termos que estimar uma esperança de forma eficiente com um tamanho de bloco que ultrapassasse o total do número de amostras.

Através de algumas simulações, pudemos estimar um valor tido como ideal para o passo de adaptação. Obtivemos μ ao compararmos graficamente o resultado da fonte e sua respectiva estimativa. Considerando todas as fontes admitidas, quanto mais sobrepostas estivessem a fonte e sua estimativa, para um dado μ, melhor. Esse valor foi μ = 0.005.

Começamos com duas fontes (k=2), com um número de amostras n de 2.222 e um tamanho de bloco p de 220. Como já foi dito, o passo de adaptação foi fixado em μ = 0.005, valor esse encontrado como ótimo.

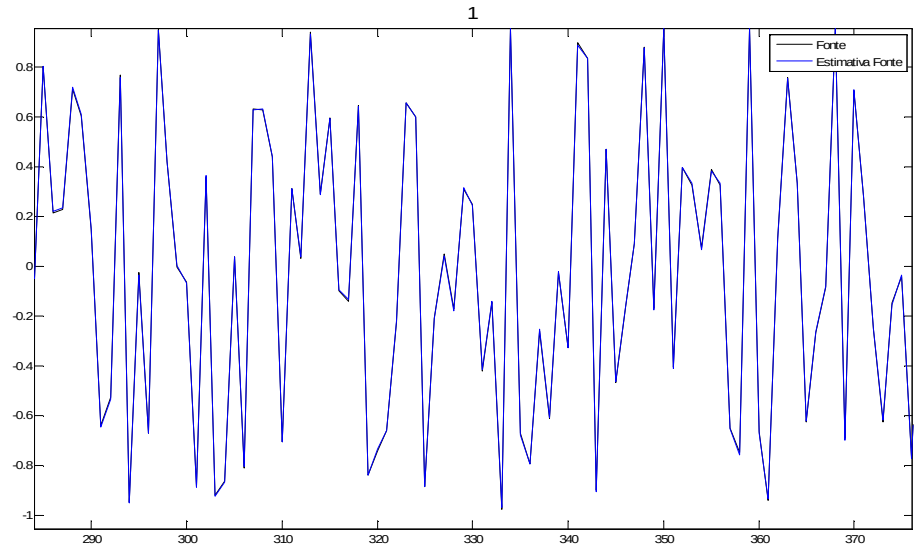


Fig. 28 – Fonte 1 em cor preta e sua estimativa em cor azul (n=2222 e p = 220)

Nessa nossa primeira simulação obtivemos um excelente desempenho, tanto para a Fonte 1 quanto para a Fonte 2. A figura 28 nos mostra a interseção total da Fonte 1 em preto com sua estimativa em azul. Não é possível distinguir pontos onde não houve sobreposição de ambas.

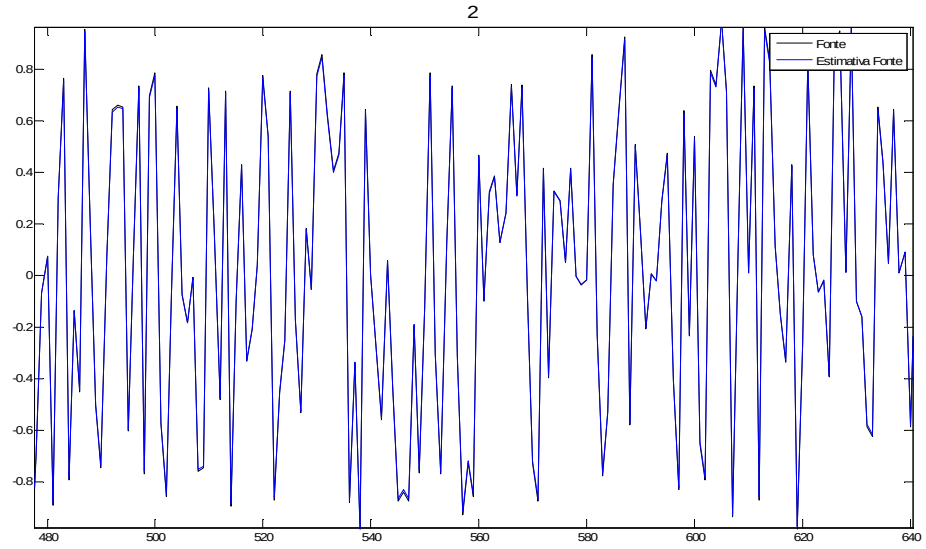


Fig. 29 – Fonte 2 em cor preta e sua estimativa em cor azul (n=2222 e p = 220)

A figura 29 acima, tal como ocorreu com a figura 28, mostra uma ótima estimativa da fonte 2. O Gradiente Natural mostrou-se um algoritmo extremamente eficiente e de rápida execução.

Seguindo com as simulações, vejamos o que acontece quando aumentamos o número de fontes k para 3 e tendo como número de amostras n igual a 4.444 e tamanho de bloco p igual a 440. A tendência de rápida execução do Gradiente Natural também foi verificada neste caso, mesmo com um número de fontes e amostras maiores que a simulação anterior.

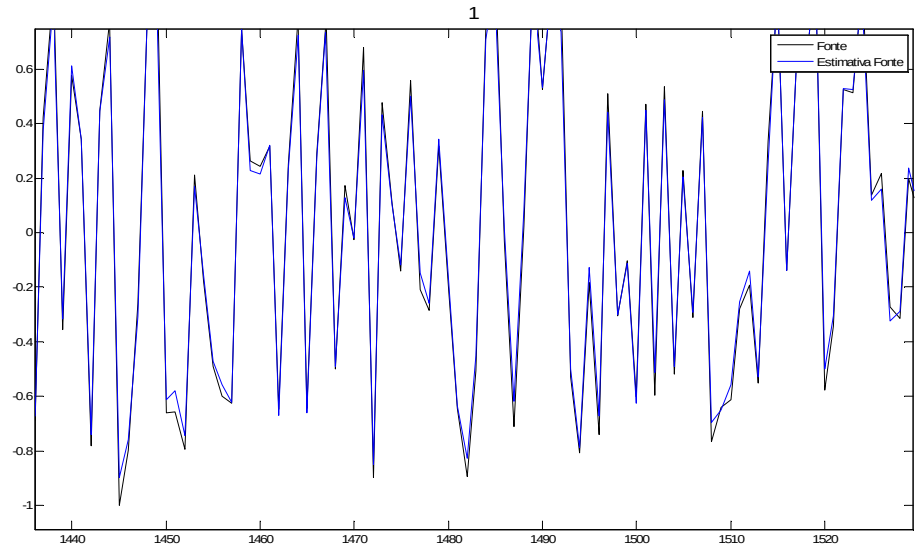


Fig. 30 – Fonte 1 em cor preta e sua estimativa em cor azul (n=4444 e p = 440)

Na figura 31 acima temos uma boa sobreposição da fonte 1 em preto e de sua estimativa em azul. Nota-se alguns pequenos pontos de incongruência entre as amostras 1450 e 1460, porém isso é irrelevante frente a totalidade do gráfico que nos mostra uma ótima recuperação.

A fonte 2 e sua estimativa são apresentadas na figura 31. Em todo o trecho considerado tivemos uma sobreposição da fonte com sua estimativa, verificando-se assim uma recuperação satisfatória.

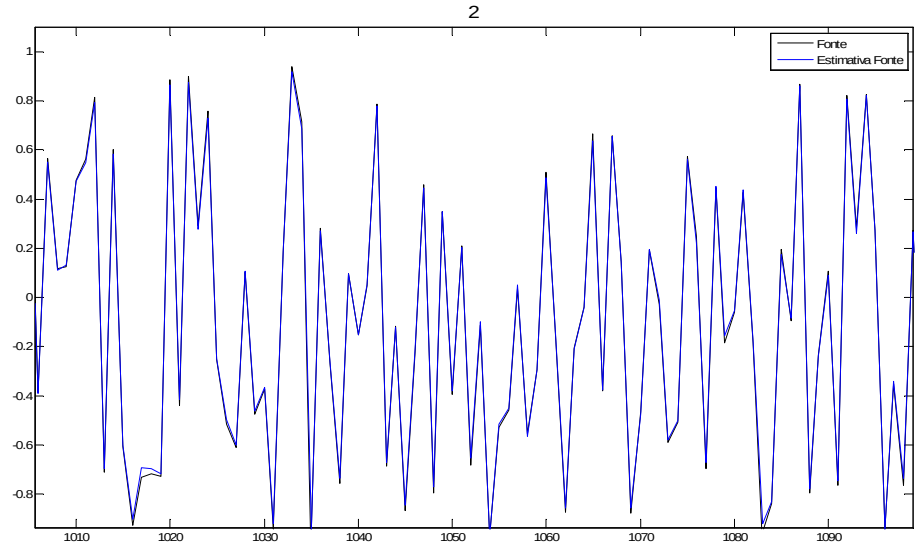


Fig. 32 – Fonte 2 em cor preta e sua estimativa em cor azul (n=4444 e p = 440)

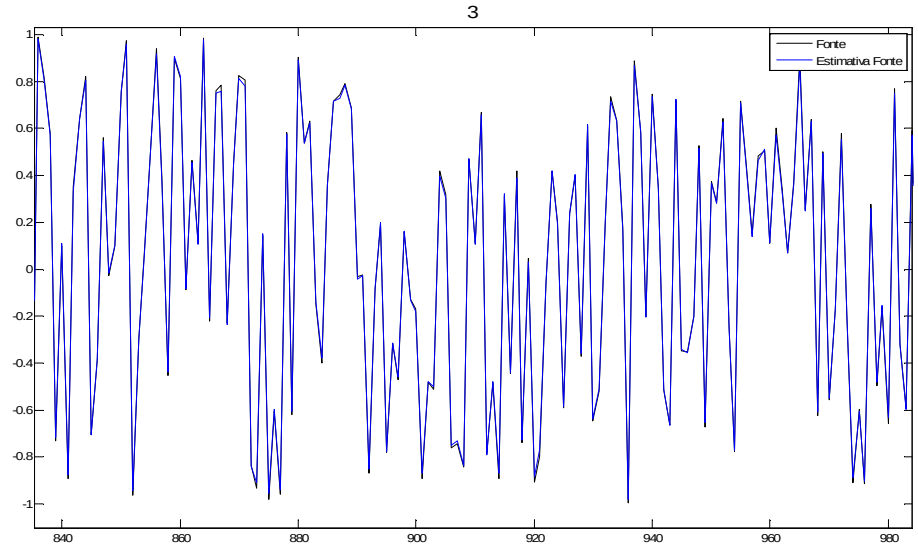


Fig. 33 – Fonte 3 em cor preta e sua estimativa em cor azul (n=4444 e p = 440)

A tendência positiva de recuperação da fonte e sua estimava se confirmou com a figura 32, que nos mostra, por fim, a fonte 3 e sua excelente estimativa no trecho considerado.

Através dos comentários tecidos até o momento, podemos concluir que não existe a necessidade de aumentarmos o número de amostras uma vez que já constatamos uma recuperação favorável para as fontes consideradas em uma

simulação anterior. Há sim, a preocupação de não utilizar-se de um valor de n muito baixo, pois dessa maneira o algoritmo Gradiente Natural não terá informações de amostras para executar satisfatoriamente.

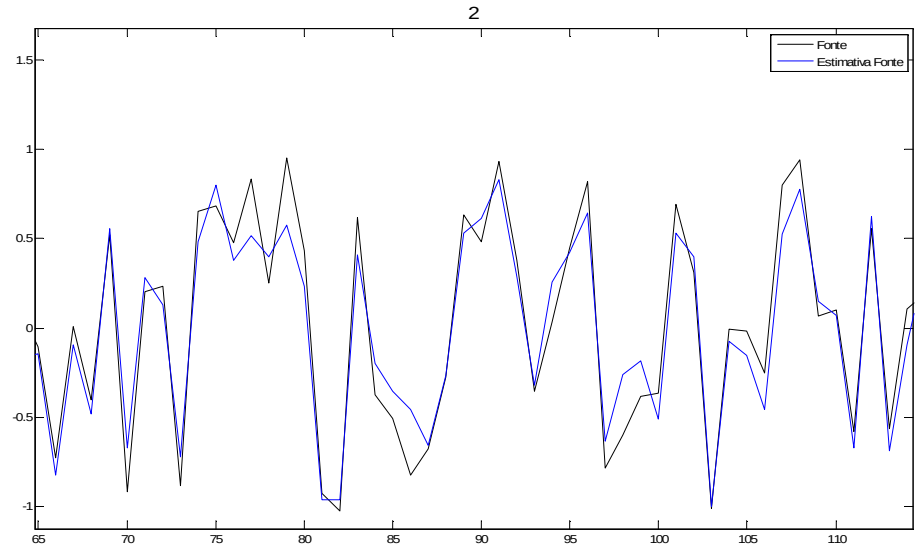


Fig. 33 – Fonte 2 em cor preta e sua estimativa em cor azul (n=220 e p = 110)

Comprovando o que acabou de ser dito, executou-se uma simulação com um total de 2 fontes, um número de amostras de 220 e um tamanho de bloco de 110. Pela figura 33, nota-se claramente que a convergência da fonte 2 e de sua estimativa ocorreu de maneira insatisfatória justamente porque o valor de n e, consequentemente o valor de p, são tido como o aquém do considerado necessário para uma ótima convergência.

3.1.3 Misturas Convolutivas

Selecionando como parâmetros iniciais duas fontes uniformemente distribuídas no intervalor de -1 a +1 e, sendo elas independentes, simulamos o algoritmo para verificar as estimativas de recuperação de ambas fontes.

Primeiramente utilizamos a matriz de mistura A_{2x2} :

A_{11}=[1 3];
A_{12}=[0 -2 4];
A_{21}=[1 3];
A_{22}=[1 2];

Considerando inicialmente o Fast-ICA Convolutivo, um parâmetro não menos importante e que deve ser considerado para a correta execução do algoritmo é o valor de R que define o tamanho dos filtros da matriz de separação W. Para a referida simulação escolheu-se R igual a 7, o que resultou em um primeiro filtro W de tamanho igual a 30. Valores inferiores de R não levaram à convergência do algoritmo e, em contrapartida, valores superiores apenas aumentavam o tempo computacional, sem grandes melhorias na convergência.

O sistema completo (transmissão, mistura e separação) foi implementado em Matlab, incluindo uma interface gráfica amigável que tornou a simulação mais simples e o acesso mais fácil. De posse do simulador, foi possível testar o desempenho do Fast-ICA Convolutivo aplicado à separação de diferentes misturas, valores de R e tamanhos de filtros. A figura 34 mostra a janela gráfica de interação com o usuário.

Os valores dos parâmetros necessários são inseridos em seus devidos campos e, após isso ter sido realizado, deve-se clicar no botão "Calcular" para obtermos a resposta gráfica como resultado da execução do algoritmo selecionado. A próxima interface também permite escolher qual o algoritmo a ser simulado considerando-se as técnicas descritas nas seções 2.3 e 2.4.

Nas próximas simulações mostraremos apenas o gráfico obtido após clicar-se em "Calcular", com intuito de focarmos no resultado de interesse e chegarmos às conclusões pertinentes.

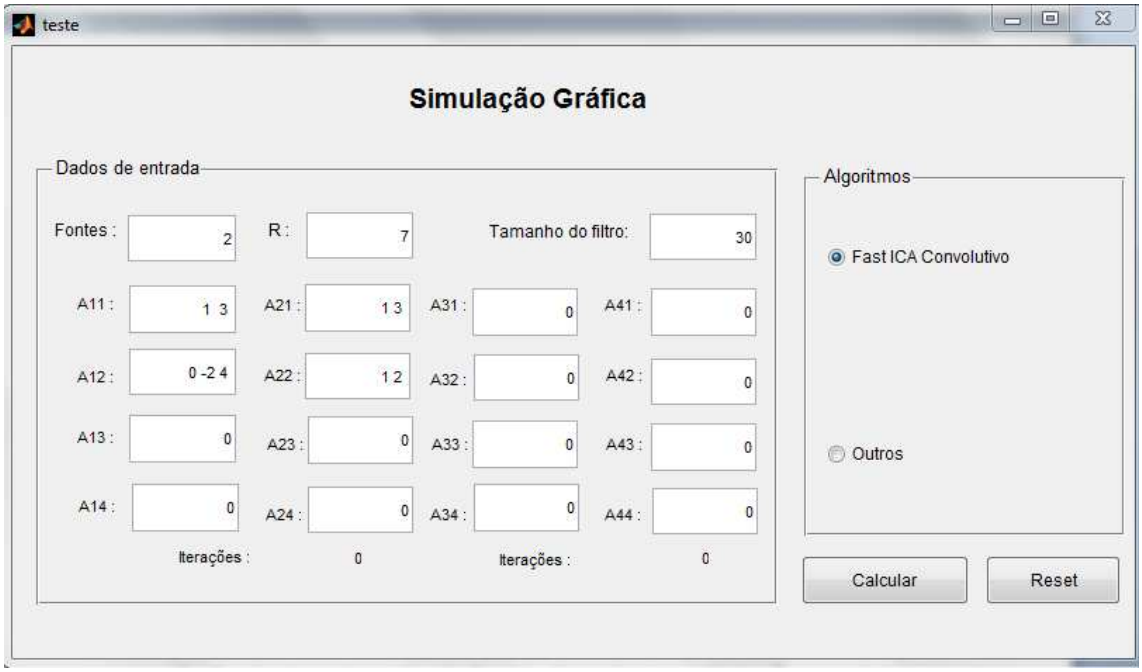


Figura 34: Interface gráfica elaborada em Matlab

A figura 35 mostra que a recuperação da primeira fonte ocorreu de forma satisfatória exceto por um fator de amplitude e atraso. A seguir explicaremos como chegamos a essa conclusão tendo a figura 36 como corroborativa desse fato.

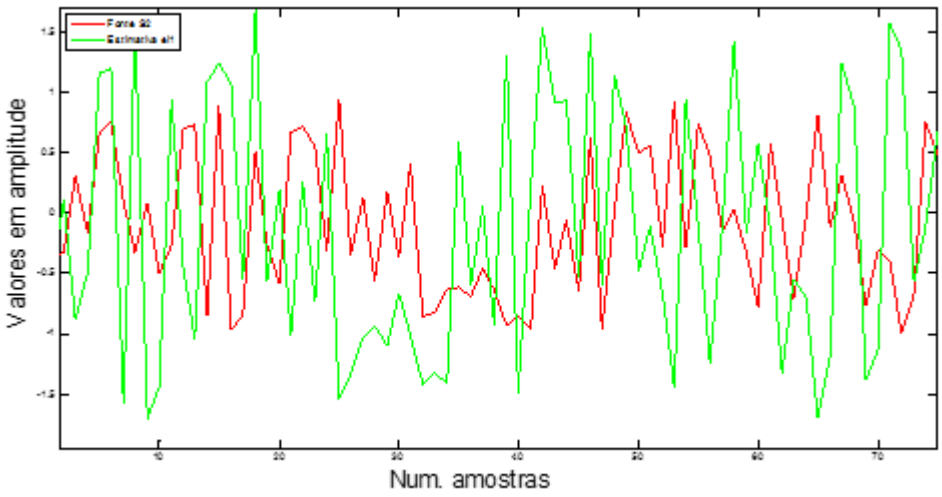


Figura 35: Fonte original “S2” e sua estimativa “Y1”

Corrigindo o fator de atraso através da identificação de quantas amostras encontram-se atrasadas da estimativa de fonte e, dividindo todas amostras pelo valor máximo de amplitude para correção dos fatores de escala, pode-se fazer uma nova plotagem da fonte S2 (cor vermelha) e sua estimativa Y1 (cor verde). Temos agora uma boa visualização de que, para os parâmetros selecionados inicialmente, a recuperação ocorre de maneira bastante satisfatória. Veja o resultado na figura 36 abaixo.

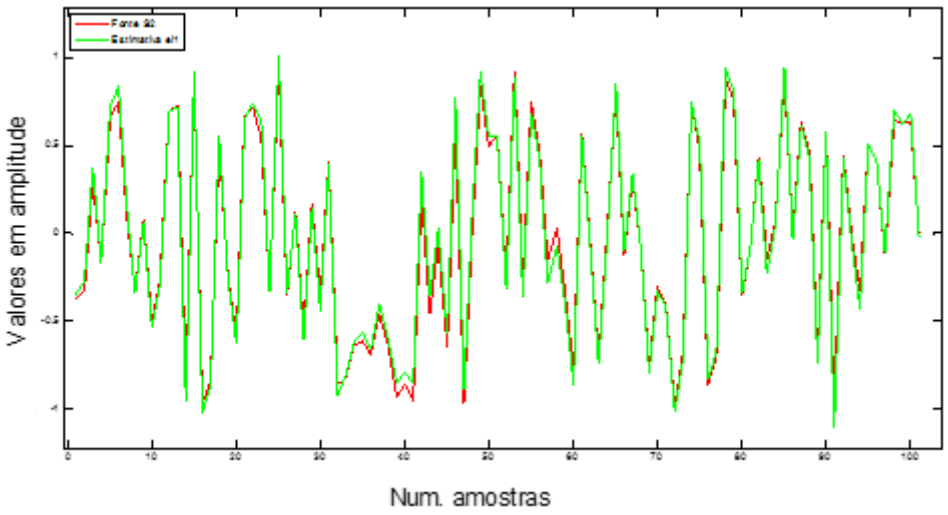


Figura 36: Fonte original “S2” e sua estimativa “Y1” (atraso e amplitude ajustados)

O mesmo podemos dizer com relação à segunda fonte, especificamente a fonte S1 e sua estimativa de recuperação. Realizando os ajustes de parâmetros necessários, temos como resultado o que podemos observar na figura 37 a seguir:

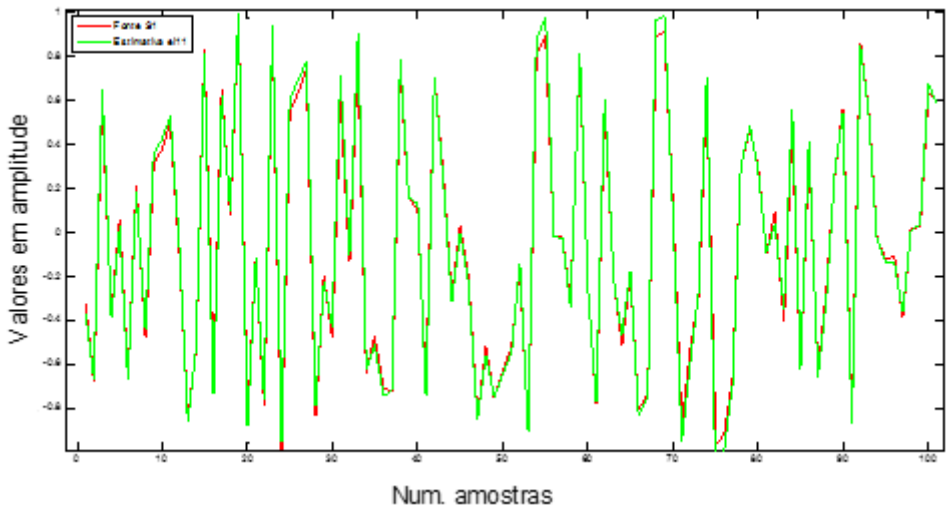


Figura 37: Fonte original "S1" e sua estimativa "Y2" (atraso e amplitude ajustados)

Como análise da medida de desempenho a fim de mensurarmos e compararmos os resultados obtidos, adotaremos a relação sinal interferência (SIR) que é definida como:

$$SIR = \frac{E[(S_i(n))^2]}{E[(S_i(n) - \widehat{S}_i(n))^2]}$$

Calculando a SIR para a primeira estimativa de fonte que obtivemos com relação à fonte S2 e sua estimativa Y1, tivemos um resultado igual a 0.3035, ao passo que para a fonte S1 e, sua estimativa Y2, a relação sinal interferência foi sutilmente menor: 0.2485. Vejamos para as próximas simulações como essa medida de desempenho contribuirá para nossas análises.

Vejamos agora de que forma os fatores ajustáveis para execução do algoritmo afetam a estimativa de recuperação das fontes desejadas. Mantivemos o mesmo valor de R igual a 7 e, alteramos a matriz de mistura A para:

A_{11}= [1 +0.5 +0.2];
A_{21}=[-0.7 0.3 0.2 0.2];
A_{12}=[0.3 .01 0.2 0.1];
A_{22}=[0.8 0.2 0.2 0.1];

Para facilitar a análise do resultado obtido, calculamos a correlação cruzada entre as fontes originais e a fonte de recuperação ao final do algoritmo.

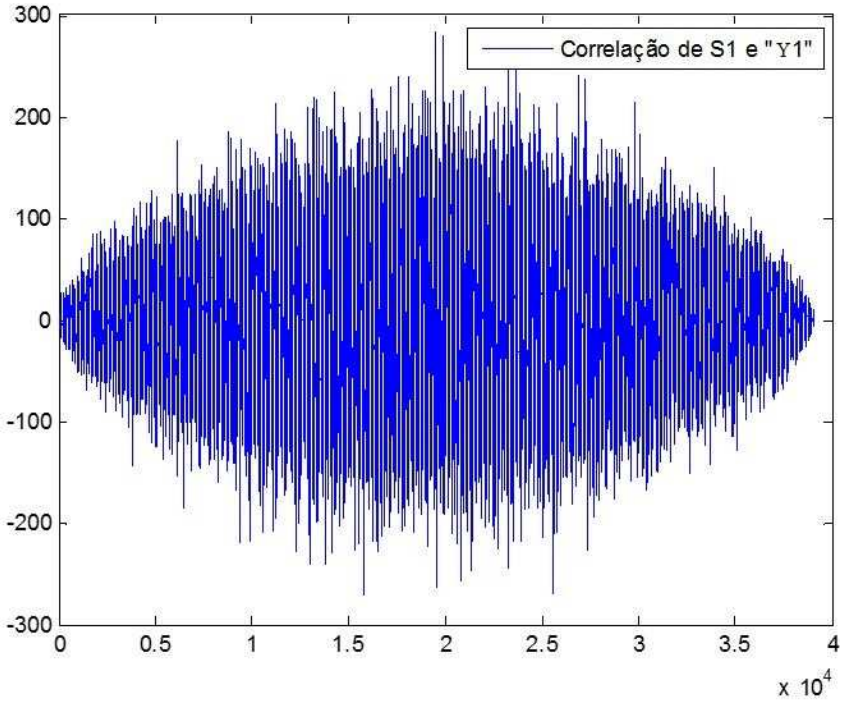


Figura 38: Correlação cruzada entre a Fonte "S1" e sua estimativa "Y1"

A figura 11 ilustra o resultado da correlação cruzada entre a fonte $\mathbf{S_1}$ e a fonte recuperada $\mathbf{Y_1}$. Isso denota que há um baixo grau de correlação entre ambas. Por outro lado, se realizarmos também a correlação entre a fonte $\mathbf{S_2}$ e o mesmo vetor $\mathbf{Y_1}$, observamos o resultado na figura 39.

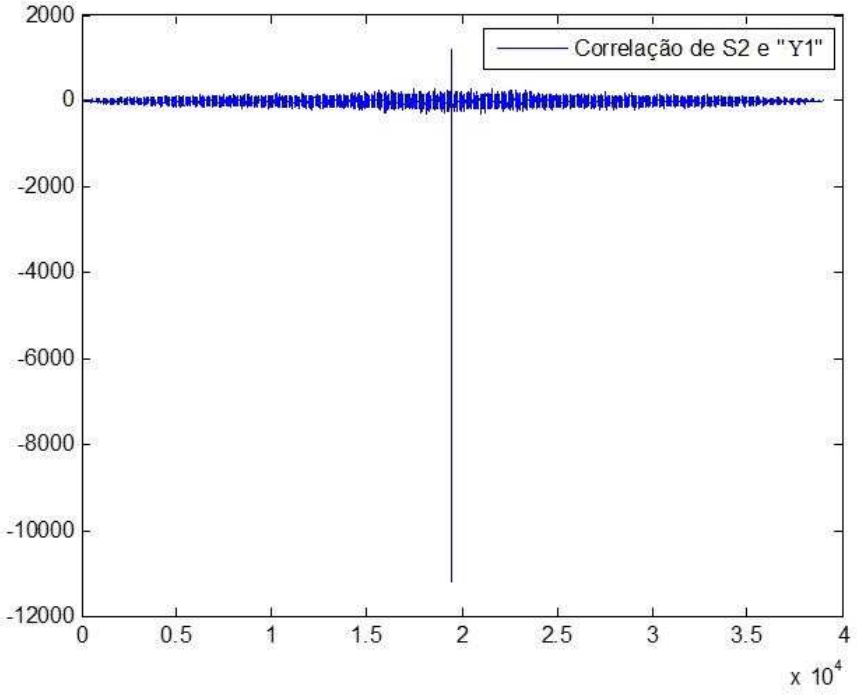


Figura 39: Correlação cruzada entre a Fonte “S2” e sua estimativa “Y1”

O que inferimos é que o vetor $\mathbf{Y_1}$ é a fonte recuperada de $\mathbf{S_2}$, uma vez que a figura 39 deixa claro a existência de uma forte correlação entre ambos.

Rodando o algoritmo mais uma vez para recuperar a segunda fonte, obtemos a figura 40 que mostra uma forte correlação entre $\mathbf{S_1}$ e, o próximo vetor $\mathbf{Y_2}$, representado na figura 40 pela denominação “Y11”.

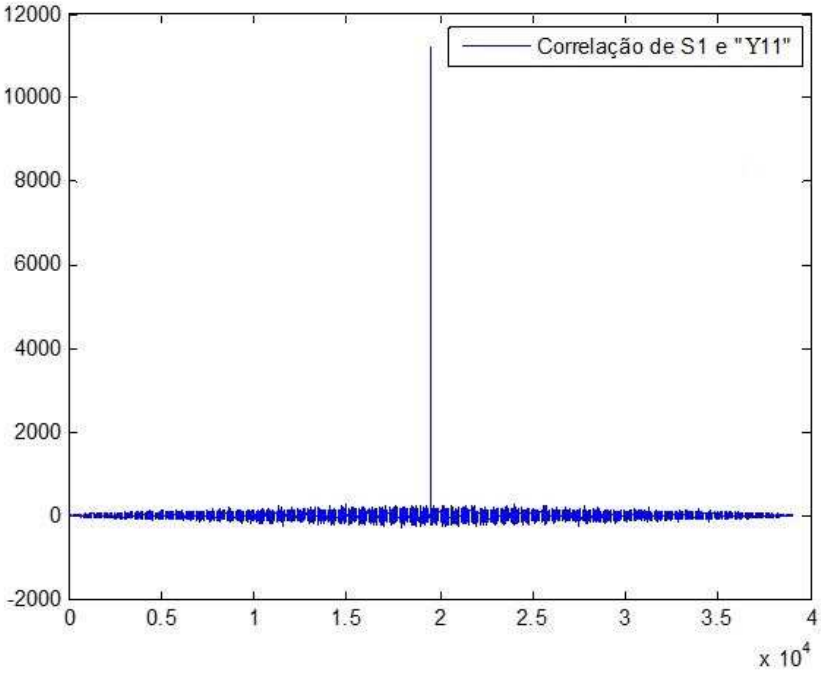


Figura 40: Correlação cruzada entre a Fonte “S1” e sua estimativa “Y_2”

Por fim, foi feita a última correlação, entre $\mathbf{S_2}$ e $\mathbf{Y_2}$, a fim de certificarmos que a recuperação de $\mathbf{S_1}$ ocorreu de forma satisfatória. A figura 41 abaixo, de fato, exibe um padrão gráfico que denota baixo grau de correlação entre ambos.

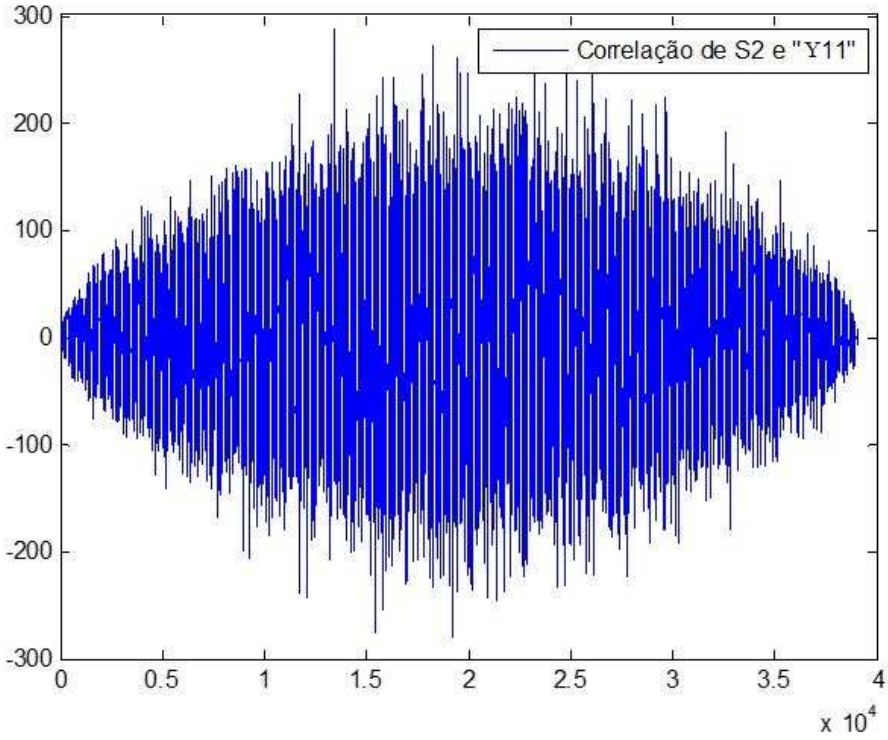


Figura 41: Correlação cruzada entre a Fonte "S2" e sua estimativa "Y_2"

Considerando agora os valores de SIR para essa simulação, tivemos como resultado para a fonte S2 e sua estimativa Y1 um valor de 0.2395, ao passo que para a fonte S1 e sua respectiva estimativa Y2, um valor bem próximo e igual a 0.2456. Se realizado um comparativo com os valores de SIR da primeira simulação feita, na qual a matriz de mistura era mais simples, não houveram grandes mudanças nos valores quantitativamente de SIR obtidos.

Um fato importante a ser salientado para as simulações desse algoritmo é que quando utilizamos fontes de áudio, em nenhuma das simulações foi possível obter uma convergência satisfatória. Diversos tipos de fontes de áudio, com padrões de amostragem e misturas distintos foram utilizados e todos os procedimentos de [1] foram seguidos à risca, ainda assim não foi possível determinar que fator ou fatores possivelmente estariam influenciado para que fosse possível realizar a simulação e obtenção da recuperação das fontes em sua totalidade.

Na realidade apenas a primeira fonte era recuperada, ao passo que a próxima fonte ainda mantinha-se misturada, o que possivelmente denota alguma inconsistência no processo de coloração.

3.2 SINAIS DE VOZ

Primeiramente, nessa etapa da simulação iremos utilizar duas fontes de áudio, uma voz masculina e outra feminina, com frequência de amostragem de 8khz e, uma média de 180.000 amostras para cada fonte. Após a etapa de centralização das fontes, adota-se para matriz de mistura A os valores a seguir:

$$A_{11}=[\ 1\ 3\ ];$$
$$A_{12}=[\ 0\ -2\ 4\ ]; \qquad \textbf{(43)}$$
$$A_{21}=[\ 1\ 3\ ];$$
$$A_{22}=[\ 1\ 2\ ];$$

Primeiramente, simulamos o Fast-ICA frequencial descrito na seção 2.4.1.

A aplicação da Transformada de Fourier de Tempo-Curto requer a escolha do tamanho da janela deslizante a ser utilizada sobre o sinal. A janela aqui adotada foi do tipo hanning de tamanho igual a 1024 com 50% de sobreposição. Isso nos fornece 512 frequências e 350 amostras temporalmente diferentes. Em seguida, de posse das matrizes com as informações na base tempo-frequência, branqueia-se os sinais de mistura a fim de tornar os sinais descorrelacionados e obter covariância nula.

A matriz de separação B(ω) é inicializada com valores distribuídos uniformemente e independentes, ao passo que a função não-linear adotada para atualização da matriz foi $g(u) = \log(0.1 + u)$.

O resultado gráfico da simulação é retratado pelas figuras 42 e 43 a seguir.

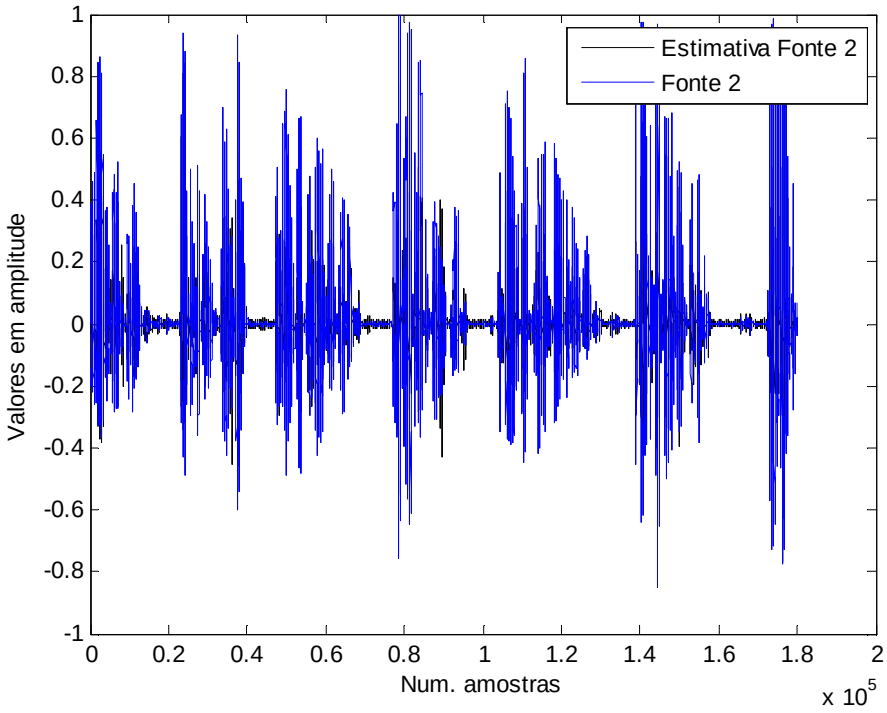


Figura 42: Fonte de áudio 2 (azul) e sua estimativa (preto)

Apesar de em ambos os gráficos a fonte original (cor azul) não coincidir exatamente em todos os pontos com a estimativa de sua respectiva fonte (cor preta), houve uma boa recuperação pois, ao executarmos o comando “sound” para analisar o resultado sonoro obtido da estimativa da fonte, foi perfeitamente possível distinguir uma única fonte e notar que o algoritmo atuou de forma satisfatória.

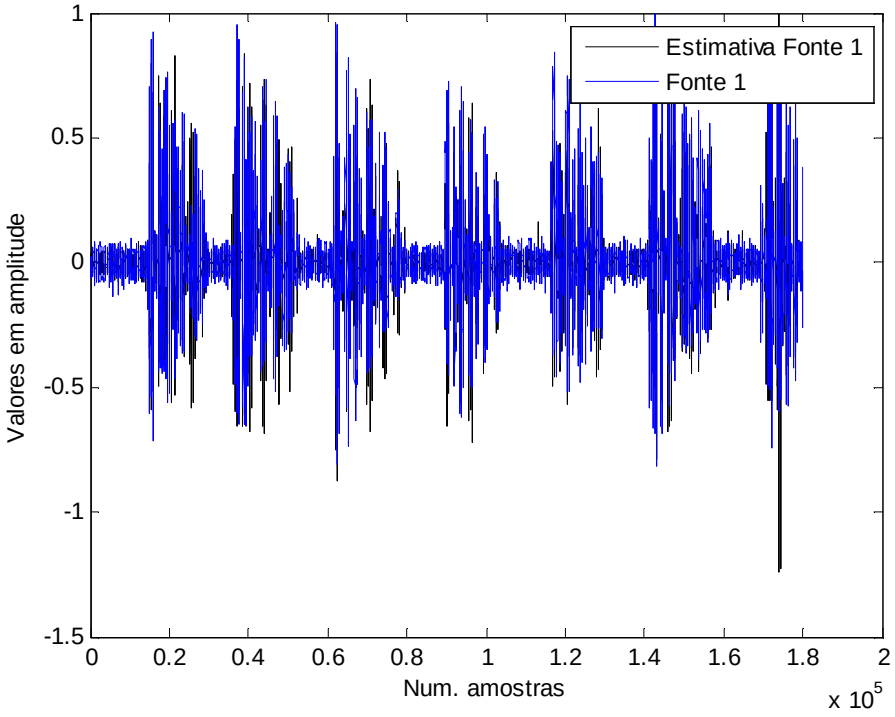


Figura 43: Fonte de áudio 1 (azul) e sua estimativa (preto)

Para a simulação realizada retratada pelas figuras 42 e 43 tivemos como resultado a SIR1 = 1.5708 e SIR2 = 1.6442, respectivamente, o que denota uma relação de taxa sinal interferente relativamente maior para a fonte 2. Quanto menor for o valor apresentado pela SIR, isso quer dizer que pior será a estimativa que foi obtida da fonte em questão.

Faremos as mesmas simulações para o algoritmo Infomax frequencial, apresentado na seção 2.4.2.

As fontes foram mantidas as mesmas, assim como a matriz de mistura A dada pela equação 43. Os resultados obtidos são mostrados nas figuras 44 e 45.

A fonte 1 pôde ser estimada de forma satisfatória ainda que a figura 44 abaixo mostre que os dois sinais, original e recuperado, não se sobrepõe perfeitamente. A reprodução sonora da fonte permite observar que o sinal foi separado com sucesso.

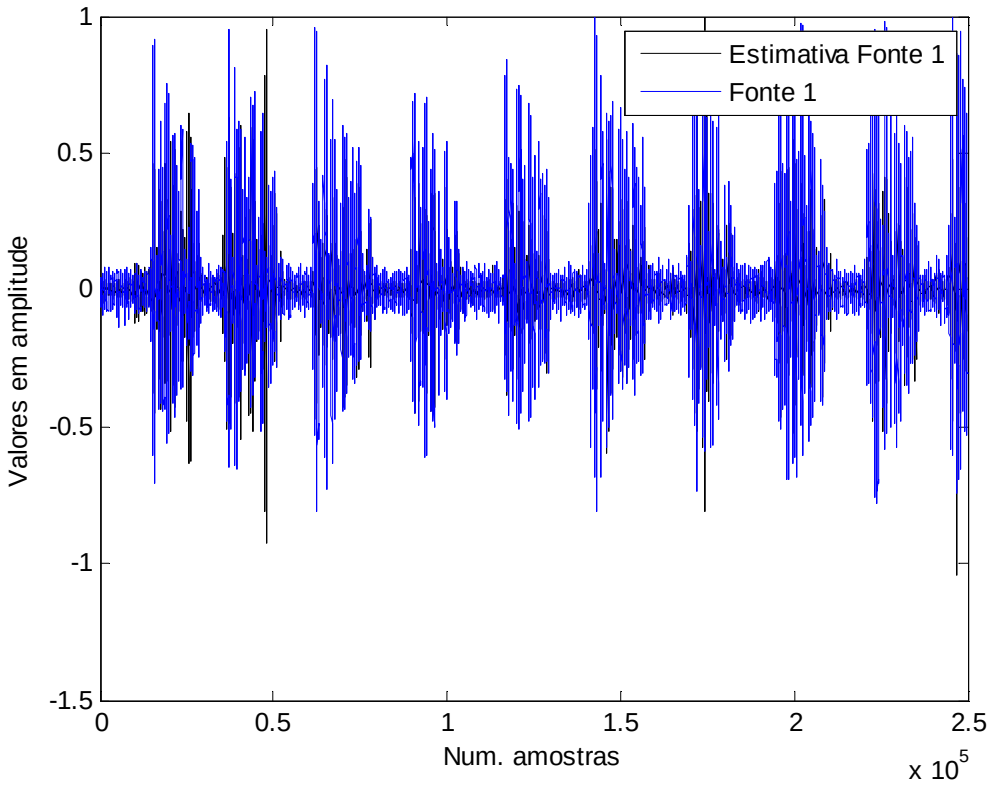


Figura 44: Fonte de áudio 1 (azul) e sua estimativa (preto)

Em seguida o mesmo procedimento foi aplicado para a fonte 2 e o resultado obtido foi a voz masculina reproduzida de forma clara e audível. A figura 45 demonstra o resultado gráfico obtido a partir da plotagem da fonte original 2 e sua estimativa.

Um ponto a se considerar para a simulação desse algoritmo é que em uma média de cinco simulações idênticas realizadas consecutivamente, apenas três delas, ou seja, em 60% obtivemos sucesso no que diz respeito a uma boa estimativa das fontes utilizadas. Nas outras duas simulações não obtínhamos qualquer sucesso e o resultado apresentado eram fontes ainda misturadas. Isso é um ponto a ser melhor avaliado em um futuro aprimoramento de estudos ou continuidade desse trabalho pois tal comportamento não era esperado.

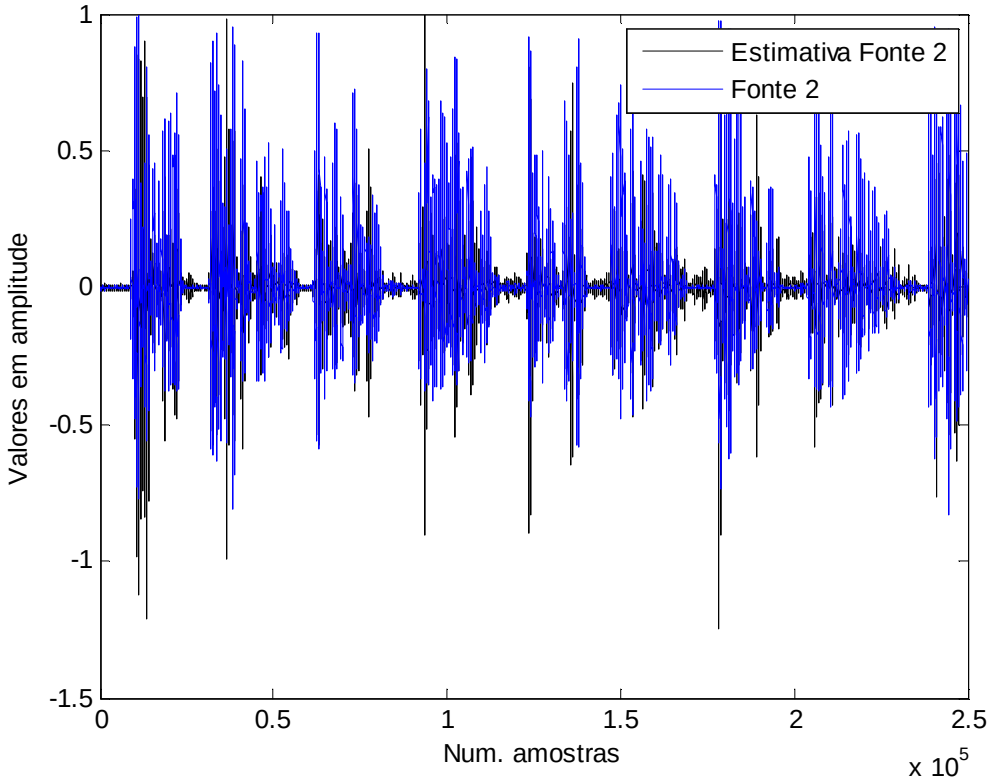


Figura 45: Fonte de áudio 2 (azul) e sua estimativa (preto)

Agora com relação as medições de desempenho, tivemos respectivamente a SIR = 0.6344 para a estimativa da fonte 1 e SIR = 0.5910 para a estimativa da fonte 2. Se fizermos um comparativo com as simulações do algoritmo anterior, esse resultado apresentado ficou consideravelmente abaixo. Graficamente isso é notório pois comparando-se as figuras 44 e 45 com as obtidas aplicando o Fast-ICA frequencial mostrados em 42 e 43, tal diferença de desempenho também é perceptível na reprodução sonora das estimativas. Embora tenha sido possível distinguir as fontes originais no sinal recuperado, o resultado obtido com o Infomax ficou com qualidade inferior ao do obtido com o Fast-ICA.

Vejamos agora como a influência da alteração da matriz de mistura afeta o resultado final com relação à estimativa desejada das fontes e valores de SIR para ambos algoritmos em frequência. Adotaremos dessa vez uma matriz A com maior quantidade de elementos de atraso e com maiores pesos:

$$A_{11}=[1\ 0.2\ 0.1];$$
$$A_{12}=[0.6\ 0.4\ 0.1];$$
$$A_{21}=[0.5\ 0.3\ 0.1];$$
$$A_{22}=[0.9\ 0.3\ 0.1];$$

(44)

As figuras 46 e 47, 48 e 49 mostram os resultados obtidos com a aplicação do Fast-ICA e Infomax frequenciais respectivamente.

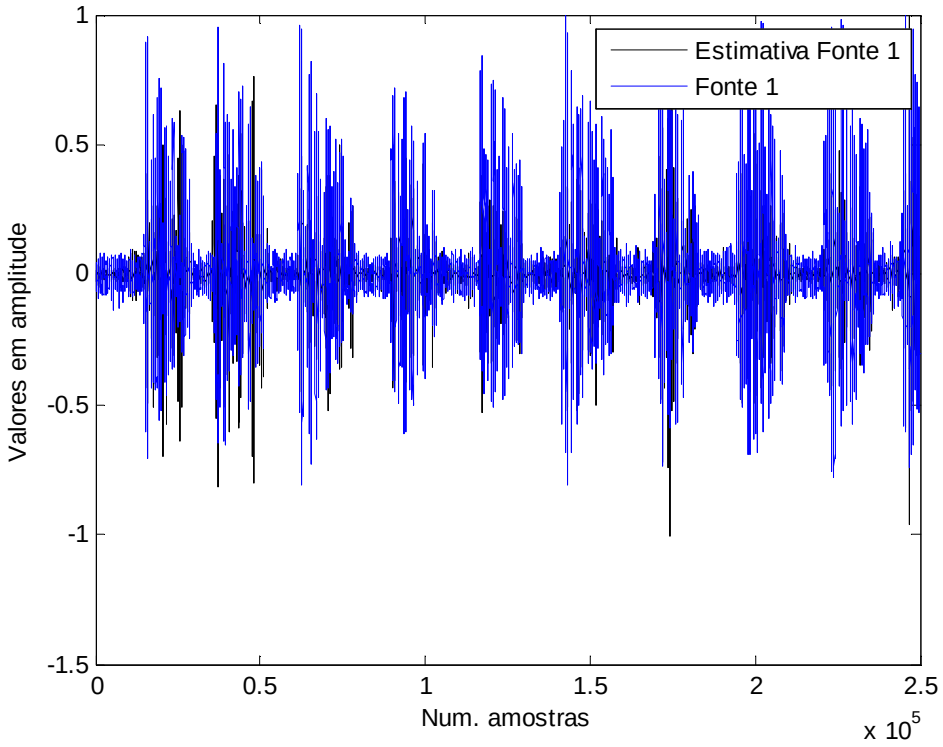


Figura 46: Fonte de áudio 1 (azul) e sua estimativa (preto)

Podemos salientar que, ao menos graficamente, não houve mudanças significativas com relação à simulação anteriormente realizada na qual a matriz A era mais simples. Tampouco houveram mudanças quando o comando "sound" é aplicado para analisarmos a qualidade sonora das estimativas: ambas puderam ser ouvidas claramente.

Com relação à SIR, usando o Fast-ICA, para recuperação da fonte 1 o resultado foi de SIR = 1.4079 e para a fonte 2 foi de SIR = 0.6351. Já para o Infomax, para a fonte 1 obtivemos SIR = 1.3217 e, para a fonte 2 foi de SIR = 0.6588. A Tabela 1 resume os resultados obtidos até aqui. Para o Fast-ICA, a recuperação da fonte 2 teve uma ligeira queda de desempenho e a fonte 1 foi a que apresentou melhor resultado . Já para o Infomax, observamos uma queda no valor obtido apenas para a fonte 1. No entanto, devemos frisar que o algoritmo nem sempre converge, o que levanta dúvidas com relação aos resultados obtidos. Na tabela 1 também é possível medir o percentual de quanto aumentou ou diminuiu a SIR para cada uma das duas simulações, com ambos algoritmos.

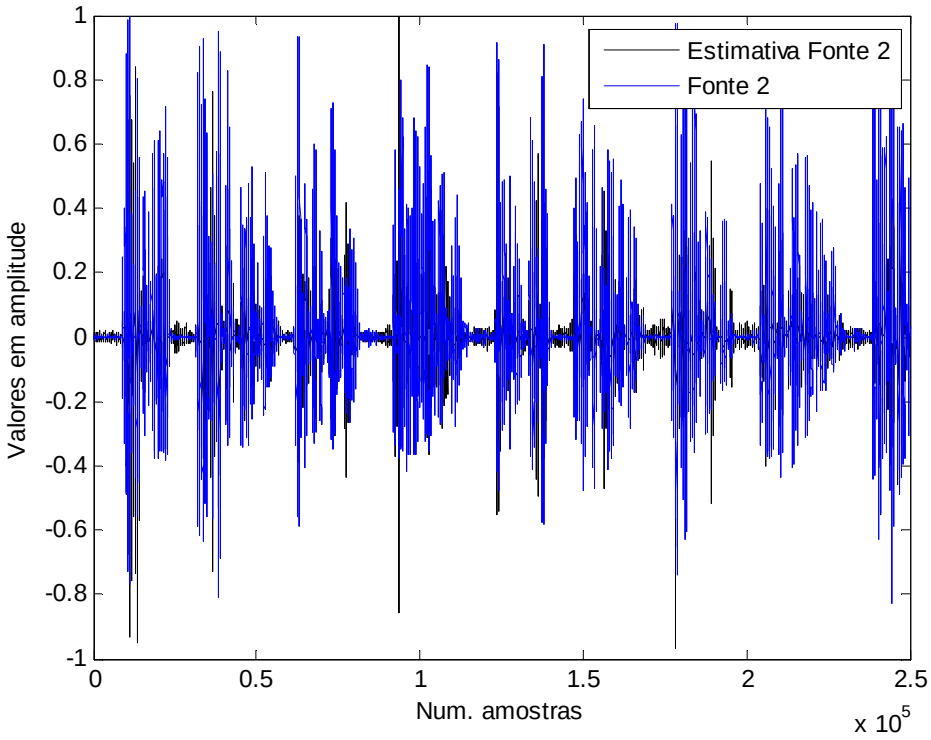


Figura 47: Fonte de áudio 2 (azul) e sua estimativa (preto)

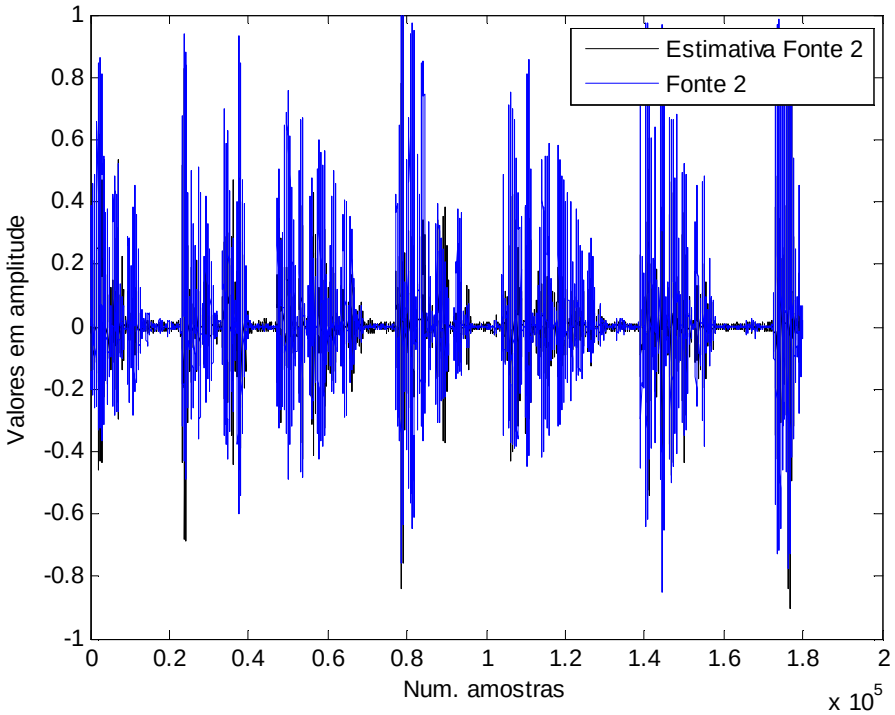


Figura 48: Fonte de áudio 2 (azul) e sua estimativa (preto)

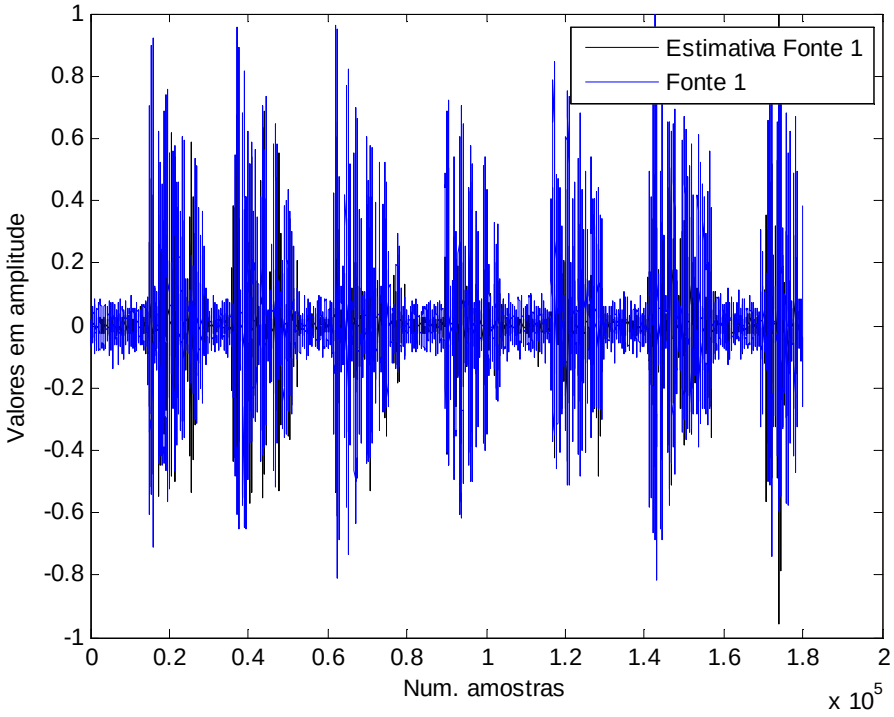


Figura 49: Fonte de áudio 1 (azul) e sua estimativa (preto)

Para finalizar essa etapa de simulações utilizaremos uma matriz de mistura A com um número ainda maior de elementos de atraso:

$$A_{11}= [\ 1\ +0.5\ +0.2\ ];$$
$$A_{21}=[\ -0.7\ 0.3\ 0.2\ 0.2\ ];$$
$$A_{12}=[\ 0.3\ .01\ 0.2\ 0.1\ ];$$
$$A_{22}=[\ 0.8\ 0.2\ 0.2\ 0.1\ ];$$

(45)

O primeiro gráfico retratado à esquerda da figura 50 apresenta a estimativa da fonte 1 para a simulação com o Fast-ICA utilizando a matriz de mistura (45). Comparando-se ao seu correspondente (à direita da mesma figura 50), referente à simulação com o Infomax, não é possível notar significativos padrões de mudanças . Pode-se dizer que pequenas variações são perceptíveis a olho nu, ao passo que, se levarmos em consideração a medição SIR, temos também diferença considerável. A SIR para a simulação atual considerando apenas a recuperação da primeira fonte foi SIR = 0.5981 para o Fast-ICA e, para a simulação com o Infomax foi de SIR = 1.4214. Percebe-se que são valores muito dispares. Veja ainda na tabela 1 que o percentual de variação foi de 138%.

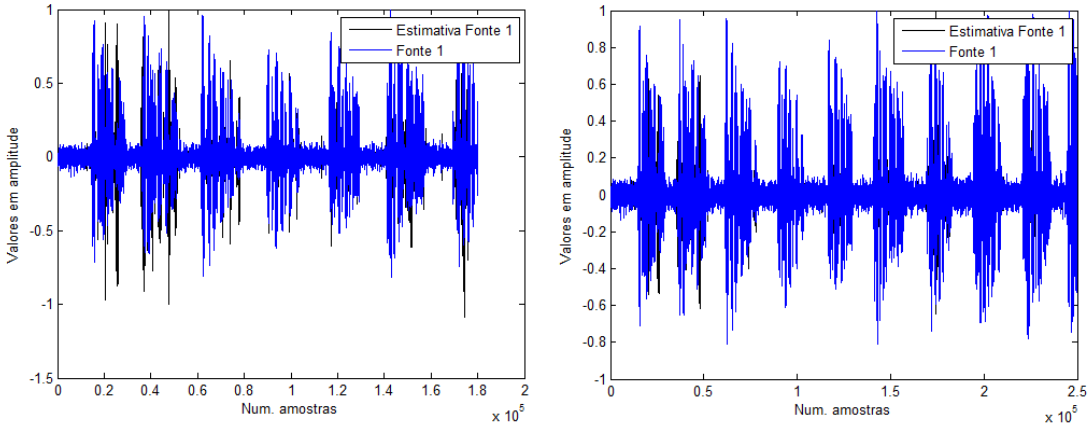


Figura 50: Fonte de áudio 1 (azul) e sua estimativa (preto)

O padrão de observação anterior não repete-se quando compara-se a estimativa da fonte 2 para o algoritmo Fast-ICA, retratada à esquerda da figura 51. Ela possui algumas diferenças com relação a estimativa da fonte 2 para a simulação com o Infomax (à direita da figura 51). Isso é perceptível pois as nuances no gráfico são notoriamente distintas comparando ambas estimativas, muito embora haja uma boa recuperação de ambas estimativas pois, na reprodução sonora foi perfeitamente possível distinguir a voz masculina, favorecendo uma melhor qualidade para a estimativa feita pelo Infomax.

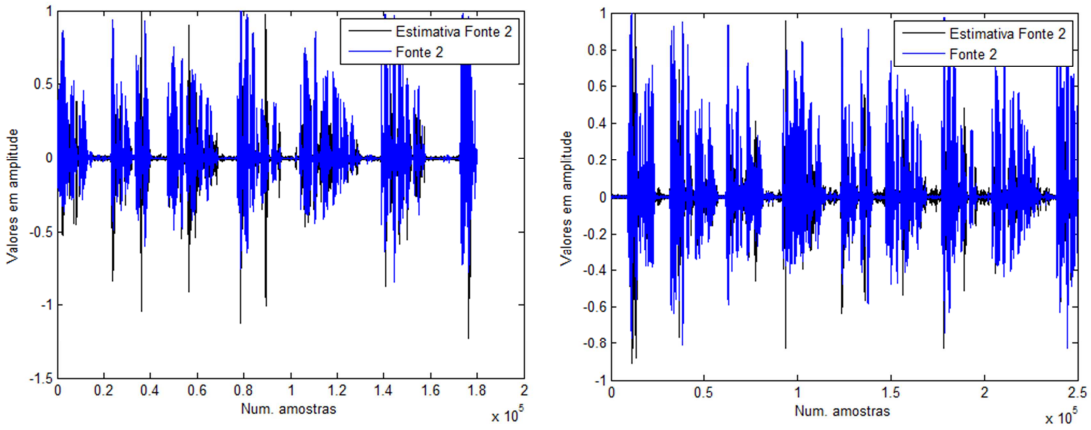


Figura 51: Fonte de áudio 2 (azul) e sua estimativa (preto)

Analisando agora as medidas de desempenho SIRs, percebe-se uma proximidade dos valores obtidos. A medição de desempenho SIR obtida para a estimativa da fonte 2 foi de SIR = 1.1243 na situação com o Fast-ICA e, esse valor foi de SIR = 1.2129 para o

Infomax. Essa pequena diferença, ainda que sutil, foi mais acentuada do que as SIRs para a estimativa de fonte 1. Vide tabela 1.

Com intuito de melhor mensurar os resultados obtidos com relação às SIRs obtidas para o Fast-ICA e para o Infomax além dos percentuais de variações com relação a uma mesma matriz de mistura, a tabela 1 foi construída.

Tabela 1: Valores correspondentes às SIRs dos algoritmos em frequência

	SIR1	SIR2	SIR1	SIR2	Fast-ICA vs Infomax	Fast-ICA vs Infomax
Equ. (43)	1.5708	1.6442	0.6344	0.5910	↑60%	↑64%
Equ. (44)	1.4079	0.6351	1.3217	0.6588	↑6,5%	↓4,5%
Equ. (45)	0.5981	1.1243	1.4214	1.2129	↑138%	↓7%

Com os valores dispostos na tabela 1 fica mais fácil visualizar que algoritmo predomina em uma dada simulação. Na verdade, como variou-se apenas os padrões da matriz de mistura, teve-se ora melhor desempenho do Fast-ICA, ora melhor desempenho do Infomax. As colunas em verde referem-se as simulações realizadas para a fonte 1 com o Fast-ICA e Infomax, respectivamente. As colunas que estão em azul, referem-se as simulações para a fonte 2 também para ambos algoritmos. Um comparativo de desempenho desses algoritmos pode ser observado nas colunas em cor amarela.

Preponderantemente, se levarmos em conta as três simulações realizadas, o Fast-ICA teve desempenho superior em 67% dos casos. Considerando também a velocidade nas estimativas das fontes 1 e 2 aqui trabalhadas e na qualidade sonora dessas estimativas, o Fast-ICA também sobressai com pequena vantagem.

4. CONCLUSÃO

Neste trabalho, implementamos algumas técnicas de separação de fontes para misturas convolutivas tanto no domínio temporal como no domínio da frequência, aplicando-se a Transformada de Fourier de Tempo Curto.

No que tange o Fast-ICA , no domínio temporal, não foi possível conseguir a recuperação das duas fontes quando consideramos sinais de voz. Uma das fontes foi recuperada corretamente, mas a segunda não. Acreditamos que ainda possa haver algum problema na etapa de recoloração, o qual não foi possível de se resolver no período deste estudo. Já considerando-se fontes aleatórias e independentes, o desempenho do algoritmo foi bastante satisfatório.

Com relação aos algoritmos no domínio da frequência, Infomax e Fast-ICA revisitados, tiveram melhor desempenho que o algoritmo testado no domínio temporal. Como resultado das simulações realizadas com sinais de voz, vimos que é difícil dizer qual dos dois possui um melhor desempenho, já que a situação se inverteu dependendo da matriz de mistura utilizada. Por outro lado, é importante observar que o algoritmo Infomax não convergiu para todas as simulações, conseguindo recuperar as fontes corretamente em cerca de 60% das simulações realizadas.

5. REFERÊNCIAS BIBLIOGRÁFICAS

[1] Johan Thomas, Y. Deville, and S. Hosseini. “Time-domain fast fixed-point algorithms for convolutive ICA”, IEEE Signal Process. Pp. 228-231 ,2006.

[2] Hyvarinen, A. and Oja, E., “A fast fixed-point algorithm for independent component analysis,” Neural Comput., vol. 9, pp. 1483–1492, 1997.

[3] Hyvarinen, A., and Karhunem, J., and Oja, E. “Independent component analysis.” John Wiley & Sons, Inc. Bell A.J. and Sejnowski T.J. “An information-maximization approach to blind separation and blind deconvolution. Neural Computation”, vol. 7, pp. 1129-1159, 2001.

[4] Haykin, S. and Chen, Z. “The cocktail party problem. Neural Computation”, vol. 17: pp. 1875-1902, 2005.

[5] Duarte, L. T. “Um estudo sobre separação cega de fontes e contribuições ao caso de misturas não-lineares”. Master’s tesis, Universidade Estadual de Campinas (UNICAMP), 2006.

[6] Smaragdis, Paris. “Blind Separation of Convolved Mixtures in the Frequency Domain”. Neurocomputing; vol. 22: pp. 21–34, 1998.

[7] Xie, Peng and Grant, Steven L. “A Fast and Efficient Frequency-Domain Method for Convolutive Blind Source Separation”. Dept. of Electrical and Computer Engineering - Missouri University of Science and Technology - Rolla, MO, USA, april, 2008.

[8] Suyama, Ricardo. Tese (Doutorado) - “Proposta de métodos de separação cega de fontes para misturas convolutivas e não-lineares”. Departamento de Comunicações – Universidade Estadual de Campinas – SP – 24th august 2007.

[9] A. Hyvarinen, “Fast and robust fixed-point algorithms for independent component analysis,” IEEE Trans. Neural Netw., vol. 10, no. 3, pp. 626–634, May 1999.

[10] S. V. Vaseghi, “Advanced Digital Signal Processing and Noise Reduction.” Chichester, U.K.: Wiley, 2000.

[11] Oliveira da Silva, Alan Paulo. “Uma implementação da Análise de Componentes Independentes em Plataforma de Hardware Reconfigurável”. Tese (Pós-graduação) Departamento de Computação e Automação – Universidade Federal do Rio Grande do Norte – RN – june 2010.

[12] Te-Won Lee, Mark Girolami, Terrence J. Sejnowski: “Independent Component Analysis Using an Extended Infomax Algorithm for Mixed Sub-Gaussian and Super-Gaussian Sources.” Neural Computation vol. 11 no.2: pp. 417-441 ,1999.

[13] T.-P. Chen, and A. Cichocki. “Stability analysis of adaptive blind source separation.” Neural Networks, vol. 10 no.8: pp. 1345–1351, 1997.

[14] Jean-François Cardoso, “Blind signal separation: statistical principles”, Proceedings of the IEEE, vol. 90, no. 8, pp. 2009-2026, Oct., 1998

[15] Amauri, S. “Natural gradiente Works efficiently in learning”, Neural Computation, vol. 10, pp. 251-276, 1998.

[16] A. J. Bell and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. Neural Computation, 7 no.6: pp.1129–1159, 1995.

[17] Santana C., E. Eder. Tese (Doutorado) – “Um modelo para Redes Neuroniais biologicamente inspirado baseado em minimização de Divergência Local”, Departamento Microeletrônica/Processamento da Informação – Universidade Federal de Campina Grande – PB – pp. 31 – november – 2009.

[18] Pinto, Daniel Francisco. Tese (Mestrado) – “Uso de Banco de Filtros em Separação Cega de Fontes” – COOPE – Universidade Federal do Rio de Janeiro – RJ – pp. 9 – october 2012.

[19] Leite, Valéria C. M. Nascimento. Tese (Mestrado) – “Separação Cega de Sinais: Análise Comparativa entre Algoritmos” – Universidade Federal de Itajubá – MG – pp. 3 – june 2014.

[20] Vidal, Francisco J. Targino. Tese (Doutorado) – “Calibração Cega de Receptores Cinco Portas Baseada em Separação Cega de Fontes” – Universidade Federal do Rio Grande do Norte – RN – pp. 46 – may 2013.

[21] Silva, Alan P. Oliveira da. Tese (Mestrado) – “Uma Implementação da Análise de Componentes Independentes em Plataforma de Hardware Configurável” – Universidade Federal do Rio Grande do Norte – RN – pp. 28 – june 2010.