DEPARTMENT OF ELECTRICAL ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

CERTIFICATE

I, Subramanian S, Roll No. 2K12/C&I/022 student of M. Tech. (Control and Instrumentation), hereby declare that the dissertation/project titled "ONLINE BLIND SIGNAL SEPARATION OF SPEECH SIGNALS" under the supervision of Dr. Dheeraj Joshi of Electrical and Electronics Engineering Department, Delhi Technological University in partial fulfilment of the requirement for the award of the degree of Master of Technology has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi **SUBRAMANIAN S**

Date:

Dr. DHEERAJ JOSHI

SUPERVISOR

Associate Professor

Department of Electrical & Electronics Engineering

Delhi Technological University

ABSTRACT

Separation of sources consists of recovering a set of signals of which only linear instantaneous mixture is observed is of prime importance for audio quality enhancement and speech recognition. In this thesis, analysis of various time-domain BSS (Blind Signal Separation) algorithms are performed and their performance compared. AMUSE (Algorithm for Multiple Unknown Source Extraction) algorithm based online BSS is implemented and its performance studied. The permutation problem of online BSS is tackled with a novel approach. The proposed method's performance parameters has been optimized with the help of GA (Genetic Algorithm) performed on polynomial regression, neural network and ANFIS (Adaptive Neuro Fuzzy Inference System) model for the algorithm. The scope of the thesis is limited to instantaneous mixtures for over determined case (the number of sensors recording signals greater than or equal to the number of source signals) with no noise.

ACKNOWLEDGEMENT

I would like to express my gratitude towards all the people who have contributed their precious time and efforts to help me complete this project; without whom it would not have been possible for me to complete this project.

I would like to thank my project supervisor Dr. Dheeraj Joshi, Associate Professor, Department of Electrical and Electronics Engineering, Delhi Technological University for his guidance, support, motivation and encouragement from the start of this project to its completion. I would also like to thank Dr. Madhusudan Singh, Head of Department, Department of Electrical Engineering, Delhi Technological University for providing me the facilities and the opportunity that helped me in preparation and submission of my thesis.

Place: Delhi

Date:

Subramanian S

M.Tech (C&I)

Delhi Technological University

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS	MEANING
BSS	Blind Signal Separation
AMUSE	Algorithm for Multiple Unknown Source Extraction
GA	Genetic Algorithm
ANFIS	Adaptive Neuro Fuzzy Inference System
FIR	Finite Impulse Response
ICA	Independent Component Analysis
PCA	Principle Component Analysis
cICA	Contextual ICA
SOS	Second Order Statistics
HOS	Higher Order Statistics
PDF	Probability Density Function
JADE	Joint Approximate Diagonalization of Eigen-Matrices
fpICA	Fixed Point ICA
EFOBI	Extended Fourth Order Blind Identification
SOBI	Second Order Blind Identification
SOBI-RO	Second Order Blind Identification with Robust Orthogonalization
SVD	Singular Value Decomposition
EVD	Eigen Value Decomposition
SDR	Source to Distortion Ratio
SIR	Source to Interference Ratio
SNR	Source to Noise Ratio
SAR	Source to Artifact Ratio
ANN	Artificial Neural Networks
LM	Levenberg-Marquardt
MSE	Mean Square Error
DSP	Digital Signal Processing
FPGA	Field-Programmable Gate Array

LIST OF SYMBOLS

SYMBOL	MEANING
$x(k)$	Instantaneous signal mixture
A	Mixing matrix
$s(k)$	Source signals
$n(k)$	Noise signals
P	Permutation matrix
$\hat{s}(t)$	Source estimate
$\hat{A}$	Channel estimate
σ	Noise variance
R_x	Covariance matrix of $x(k)$
$\widehat{W}$	Whitening matrix
$\lambda_1,\ldots,\lambda_n$	Eigen values of R_x
$h_1,\ldots,h_n$	Eigen vectors of R_x
$z(t)$	Whitened signal
D_{total}	Total relative distortion
D_{interf}	Relative distortion due to interference
D_{noise}	Relative distortion due to additional noise
D_{artif}	Relative distortion due to algorithmic artifacts
$\hat{y}$	Model's estimates
n	Window size
p	Percentage overlap
T_d	Time delay for each step of Online BSS

CHAPTER 1
INTRODUCTION

1.1 BLIND SIGNAL SEPARATION

It is common to have microphone recordings as shown in Equation (1.1) in audio signal processing.

$$x(k) = A\ s(k) + n(k) \tag{1.1}$$

The situation occurs when there is more than one speaker in an anechoic room with more than one microphone recording the signal. The mixing matrix A is the attenuation introduced by the channel. Matrix A may be instantaneous when the channel introduces no delay (or same delay) from the source to the microphone array. If the channel delay is not same from the source to the microphone array, mixing matrix takes the form of time delayed convolutive matrix.

If no priori information on the mixing matrix A is available to the separating algorithm, then the recovery of sources is termed as “blind”. If some information on the mixing matrix (or the sources) is available to the algorithm, then the process of recovering signals is termed as ‘semi-blind’.

BSS (Blind Signal Separation) is the process of estimating A and/or retrieving the source signals $s(k)$ without any priori information about the mixing matrix A. This is achieved by exploiting the statistical properties of the signal recordings, of course with some basic assumptions on the source signals (and/or the recordings). In most practical

cases, these assumptions do not limit the functionality of the algorithm and hence, BSS algorithms hold valid for a wide range of signals.

BSS is not a simple problem owing to the fact that it is neither easy nor practical to estimate the channel parameters A. Also, it is nearly impossible to practically record the unattenuated signals at every source without possibility of interference from other source. BSS for convoluted mixture is more complex owing to the fact additional steps in the form of FIR filters are required to deconvolute the mixture before BSS algorithms can be applied.

When BSS is performed online, more complications arise due to the permutation problem inherent with BSS algorithms. This problem is solved in this thesis with the help of a novel method: method of overlapping samples. The scope of the thesis is limited to instantaneous mixtures for overdetermined case. No noise model is assumed here, as estimating noise using BSS is not a complex issue as discussed in the algorithms in Chapter 2. Moreover, noise can be further reduced by additional post processing steps.

1.1.1 Literature review

The problem of ICA (Independent Component Analysis) was actually first proposed by Herault and Jutten around 1986, and it was named so because of its similarities with PCA (Principal Component Analysis) [10]. Even though Herault's algorithm has huge limitations, it is commendable that they were able to provide an algorithmic solution to the BSS problem when there was no theoretical explanation available.

BSS algorithms using higher order statistics have been studied for separating signals [14]. Inouye proposed a method for the separation of two sources [19]. At the same

time, Comon proposed another method for more than two sources [20]. These three solutions were among the first direct (within polynomial time) solutions to the ICA problem [15].

Tong derived the AMUSE algorithm and assessed its performance with other known algorithms [11]. Tong formulated the mathematical structure for blind identification from which the identifiability conditions were derived [8].

Linsker was the first to develop learning algorithm for maximization of mutual information between two layers of neural network [24]. Linsker used a Hebbian learning rule to maximize the mutual information when the network receives both noise and signal and an anti-Hebbian rule to maximize information when the network receives only noise [16].

Roth and Baram proposed an estimation method of probability density function of a random vector by maximising the entropy of a feed-forward network of sigmoidal units and applied it in classification, estimation and forecasting [22]. Bell and Sejnowski were first to develop information maximization approach to ICA problem with the help of self-organizing learning algorithm which maximizes the information transferred in network of non-linear units [16].

ICA can be expressed in terms of entropy [16], mutual information [17], contrast function [15] or other methods of statistical independence of signals.

Pearlmutter and Parra developed the contextual ICA (cICA) which derives from the maximum likelihood density estimation. cICA can separate in a number of situations where standard ICA cannot like sources with low kurtosis, colored Gaussian sources and sources with Gaussian histograms [21].

Girolami and Fyfe introduced a generalized ICA method with the help of negentropy as the distance measure and introduced a simple adaptive non-linearity which was formed by on-line estimation of latent variable kurtosis and removes the stand ICA constraint of latent variable pdf modality uniformity [23].

Cardosa exploited the second-order statistics of the received signals by joint diagonalization of a set of covariance matrices [7].

Although much overlooked in the literature, the performance measures for evaluating the performance of BSS algorithms is not a completely trivial topic. Many robust performance estimates are discussed in Gribonval et al, 2003; Vincent et al, 2006. [11] [12].

1.1.2 Applications

The classical application of BSS is the cocktail party problem. Although the first works on Cocktail party problem traces back to Colin Cherry [28], much of the early work was focussed on human listener's recognition of audio signals. The term 'cocktail party problem' can be expressed as the ability to focus one's listening attention on a single talker among a cacophony of conversations and background noise [29]. Humans are adept at this task although much of the brain's computational model to recognise and focus on one audio signal of interest is not fully understood. Nevertheless, there are algorithms which can perform this task of source separation. These algorithms are collectively referred to as BSS which utilizes the statistical properties of the signals to separate the sources.

Another important application of BSS is to enhance the signal for speech recognition and authentication. Many times in automatic speech recognition, it is necessary to

remove the ambient noise, includes other speakers in the room, for better recognition rates. Ethers has successfully implemented BSS techniques to automatic speech recognition and achieved an error rate of 12% against the earlier 52% error rate without the use of BSS techniques [30].

Other applications of BSS include airport surveillance [31], brain tumour analysis [32], natural image processing [33], rotation machine vibration analysis [34], seismic signals analysis [35] and the list goes on.

1.2 BLIND IDENTIFIABILITY

Tong proved that the channel attenuation matrix A can be estimated only up to scaling and permutation ambiguity if and only if source signals have distinct autocorrelation functions, or equivalently, signal spectrums [8].

Tong showed that A can be best identified up to post-multiplication of diagonal matrices and permutation matrices. This indeterminacy stems from the very nature of 'blind' identification that all the three quantities in Equation (1.1): A, s(t) and n(t) are unknown.

Let Λ be any non-singular diagonal matrix and P be a permutation matrix. Let us omit noise $n(t)$ in Equation (1.1) without loss of any generality. We can rewrite Equation (1.1) as

$$x(t) = (A\,\Lambda^{-1}\,P^{-1})(P\Lambda\,s(t)) \tag{1.2}$$

The second term in Equation (1.2) is scaled and reordered signal $s(t)$. This shows that

1) The recovered signal cannot be ordered by BSS algorithm.

2) The signal can only be estimated to a scalar multiple.

3) The signal may be inverted in phase.

This shows that there is no unique solution to the BSS problem without forgoing the 'blind' assumption.

In batch processing, these indeterminacies are acceptable since they do not affect the information content of the signal. But in online processing, this may cause huge complications in the form of permutation problem as discussed in Chapter. 5.2.

1.3 ASSUMPTIONS

Almost all BSS algorithms require some common assumptions on the signals that they can separate. Some of the basic assumptions are enumerated below:

A. 1 Source signals $s_j(t)$ must be mutually independent.

A. 2 The mixing matrix should be a full column rank matrix: if $A \in R^{n \times m}$, then $rank(A) = m$.

A. 3 Source signals should not have Gaussian distribution (only for ICA algorithms).

A. 4 $m \geq n$

A. 1: ensures that the signals do not depend on each other. It is not sufficient for the source signals to be uncorrelated as uncorrelated signals do not necessarily mean they are independent [3].

A. 2: ensures that mixing matrix A makes the observations mutually independent on each other. If not, then the signals cannot be separated even if the sources are mutually independent.

A. 3: Gaussian variables' joint probability density is completely symmetric. Therefore it does not have information on the direction of the columns of the mixing matrix A making it impossible to estimate A by ICA methods. This assumption is only required by ICA algorithms.

A. 4: This assumption means that the number of sensor observations must be more than the number of source signals. It is referred to as over-determined case. If $n > m$, then more assumptions are needed to estimate the source signals. It is referred to as under-determined ICA. The algorithms discussed in this thesis cannot process under-determined case without additional modifications.

CHAPTER 2
BLIND SIGNAL SEPARATION ALGORITHMS

2. 1 INTRODUCTION

Although there are algorithms for BSS which use the geometric properties of the signal, they are very limited and not applicable to wide range of signals. Most of the ICA algorithms use the statistical properties of the signal to separate them. In this thesis, the scope is limited only to statistical approaches. These algorithms can be classified into the following three types: Second Order Statistics (SOS), Higher Order Statistics (HOS) and Probability Density Functions (PDF) based algorithms. Algorithms like JADE, fpICA, EFOBI fall under the HOS category. AMUSE, SOBI, SOBI-RO use the SOS properties to separate source signals. Algorithms based on the PDF of the signal are arcane and not widely used for BSS problems anymore.

2. 2 ALGORITHMS

2.2.1 Algorithm for Multiple Unknown Source Extraction

Even though some authors refer to SOS algorithms as ICA algorithms, these algorithms do not exploit implicitly or explicitly statistical independence. Also, they can separate Gaussian signals unlike ICA algorithms. Although, AMUSE algorithm have some similarity with standard PCA, they differ from PCA in that AMUSE employs PCA two times in two separate steps: in the first step, standard PCA can be applied for whitening

(Sphering) data and in the second step, SVD/PCA is applied for time delayed covariance matrix of the pre-whitened data [11]. The flowchart of the AMUSE algorithm is shown in Figure 2.1.

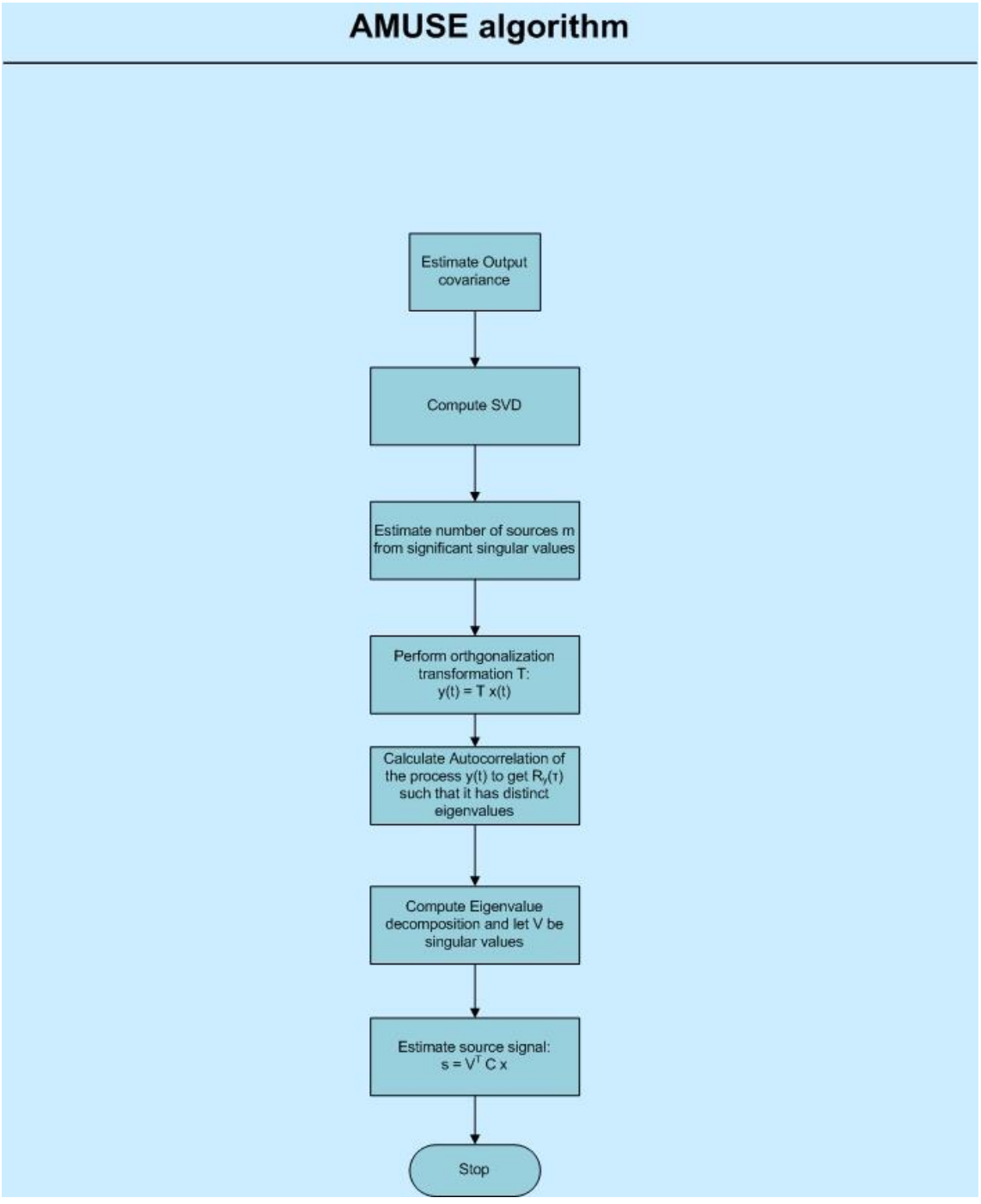

Figure 2.1 AMUSE algorithm

The AMUSE algorithm is explained as follows [8]:

1. Calculate output covariance

$$R_x = E\{x(t)\, x^T(t)\} \tag{2.1}$$

2. Calculate the SVD of R_x:

$$R_x = U\ diag(\lambda_1^2, \lambda_2^2, .., \lambda_n^2)\ U^T \tag{2.2}$$

3. Use the number of significant singular values to estimate the number of sources m, the noise variance σ^2 can be estimated from the insignificant singular values.

4. Perform Orthogonalization transformation:

$$d_i = \sqrt{\lambda_i^2 - \sigma^2}$$

$$U_s = [u_1, \dots, u_m]$$

$$T = diag\left(\frac{1}{d_1}, \frac{1}{d_2}, \dots, \frac{1}{d_n}\right) U_s^T$$

$$y(t) = T\, x(t) \tag{2.3}$$

5. Choose τ: $\left(R_y(\tau) + R_y(\tau)'\right)/2$ such that it has distinct eigenvalues, where $R_y(\tau) = E\{y(t)\, y(t-\tau)'\}$.

6. Perform Eigen-decomposition of $\left(R_y(\tau) + R_y(\tau)'\right)/2$ to the eigenmatrix V.

7. Channel estimation:

$$\hat{A} = T^{\#} V \tag{2.4}$$

where, $^{\#}$ denotes pseudo-inverse.

8. Signal estimation:

$$\hat{s}(t) = V^T y(t) \tag{2.5}$$

The main advantage of AMUSE algorithm is that it is possible to automatically order components based on their singular values of the time-delayed covariance matrix. All components estimated by AMUSE are uniquely defined and consistently ranked: the probability that two real-world signals having same singular value is very minimal.

The one disadvantage of AMUSE algorithm is that is relatively sensitive to additive noise since the algorithm exploits only one time delayed covariance matrix. This can be alleviated up to an extent by pre-processing and post-processing to remove the additive noise.

2.2.2 Second Order Blind Identification

SOBI technique separates source signals based on joint diagonalization of a set of covariance matrices which in comparison with HOS algorithms can separate Gaussian signals. The flowchart outlining the algorithm of SOBI is shown in Figure 2.

The algorithm for SOBI is explained as follows [7]:

1) From the data samples, calculate covariance $\hat{R}(0)$. Perform EVD: let $\lambda_1,\ldots,\lambda_n$ be the Eigen-values and $h_1,\ldots,h_n$ be the corresponding Eigen-vectors.
2) Estimate noise variance σ^2 from insignificant eigenvalues. The whitening matrix is given by Equation (2.6).

$$\widehat{W} = \left[(\lambda_1 - \sigma^2)^{-1/2} h_1, \ldots, (\lambda_n - \sigma^2)^{-1/2} h_n\right]^T \quad (2.6)$$

Then, the whitened signal is

$$z(t) = \widehat{W}\, x(t) \quad (2.7)$$

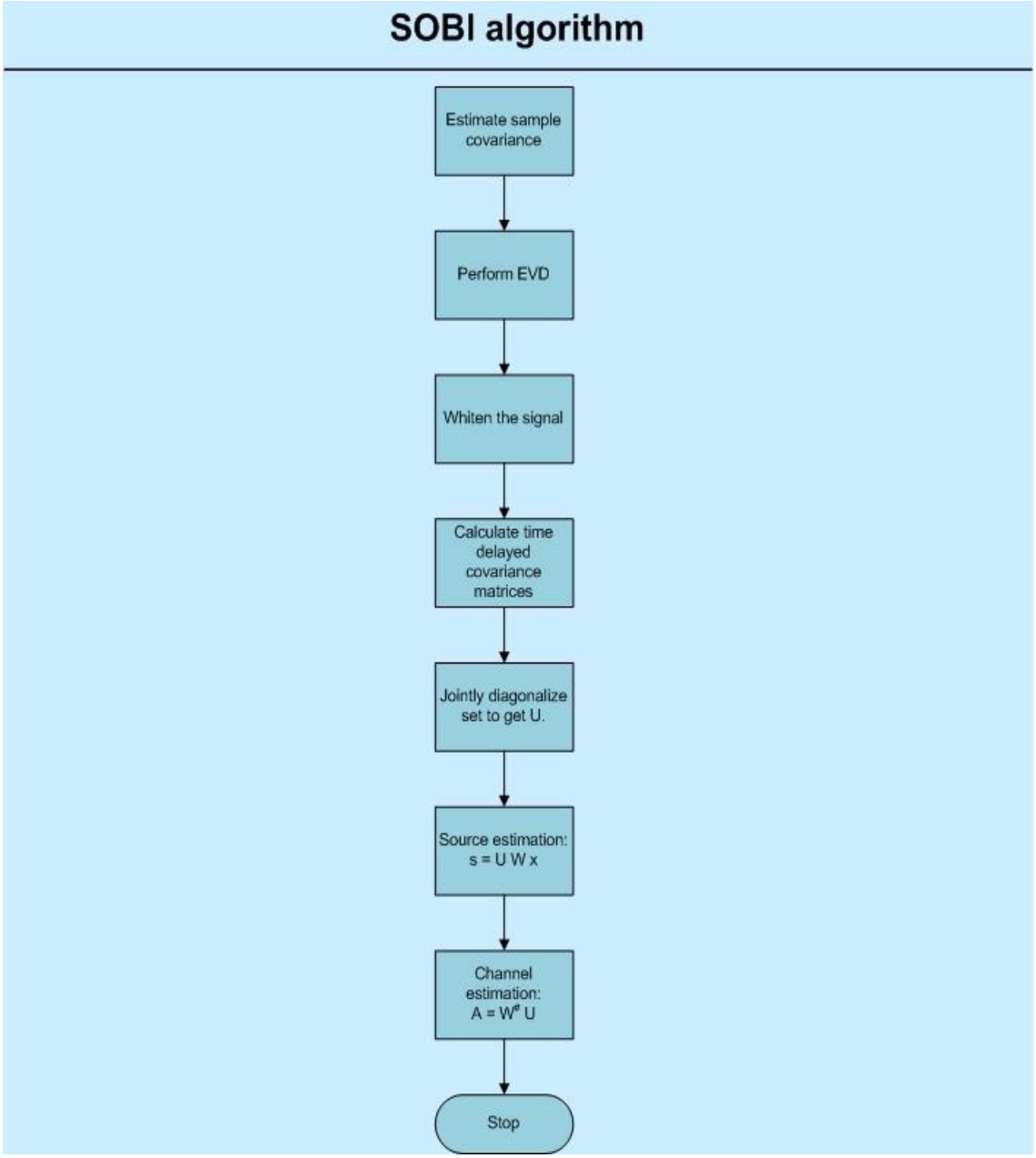

Figure 2.2 SOBI algorithm

3) Calculate time delayed covariance matrices $\hat{R}(\tau)$ of $z(t)$ for a fixed set of time lags $\tau = \{\tau_j | j = 1, \dots, K\}$.

4) Jointly diagonalize the set $\{\hat{R}(\tau_j) | j = 1, \dots, K\}$ to get U.

5) Source estimates:

$$\hat{s}(t) = \hat{U}\,\hat{W}\,x(t) \qquad (2.8)$$

And, channel estimates:

$$\hat{A} = \hat{W}^{\#}\,\hat{U} \tag{2.9}$$

The use of several covariance matrices makes the algorithm more robust [7].

2.2.3 Second Order Blind Identification with Robust Orthogonalization

In original SOBI algorithm, to make the correlation matrix positive definite, Cardosa used a lag close to zero. The problems associated with having correlation matrix at zero lag is that it becomes sensitive to noise and choosing such a lag requires very high sampling rate. SOBI-RO was proposed by Belouchrani and Cichocki to solve these problems. SOBI-RO estimates the whitening matrix from EVD of a positive definite matrix formed by linear combination of correlation matrices at non-zero lags [6].

The flowchart for SOBI-RO is outlined in Figure 2.3.

The main advantage of SOBI-RO algorithm compared to SOBI is its robustness to noise. The downside to calculating whitening matrix from time-delayed correlation matrix is that it increases the processing time, which is not advisable in real time applications.

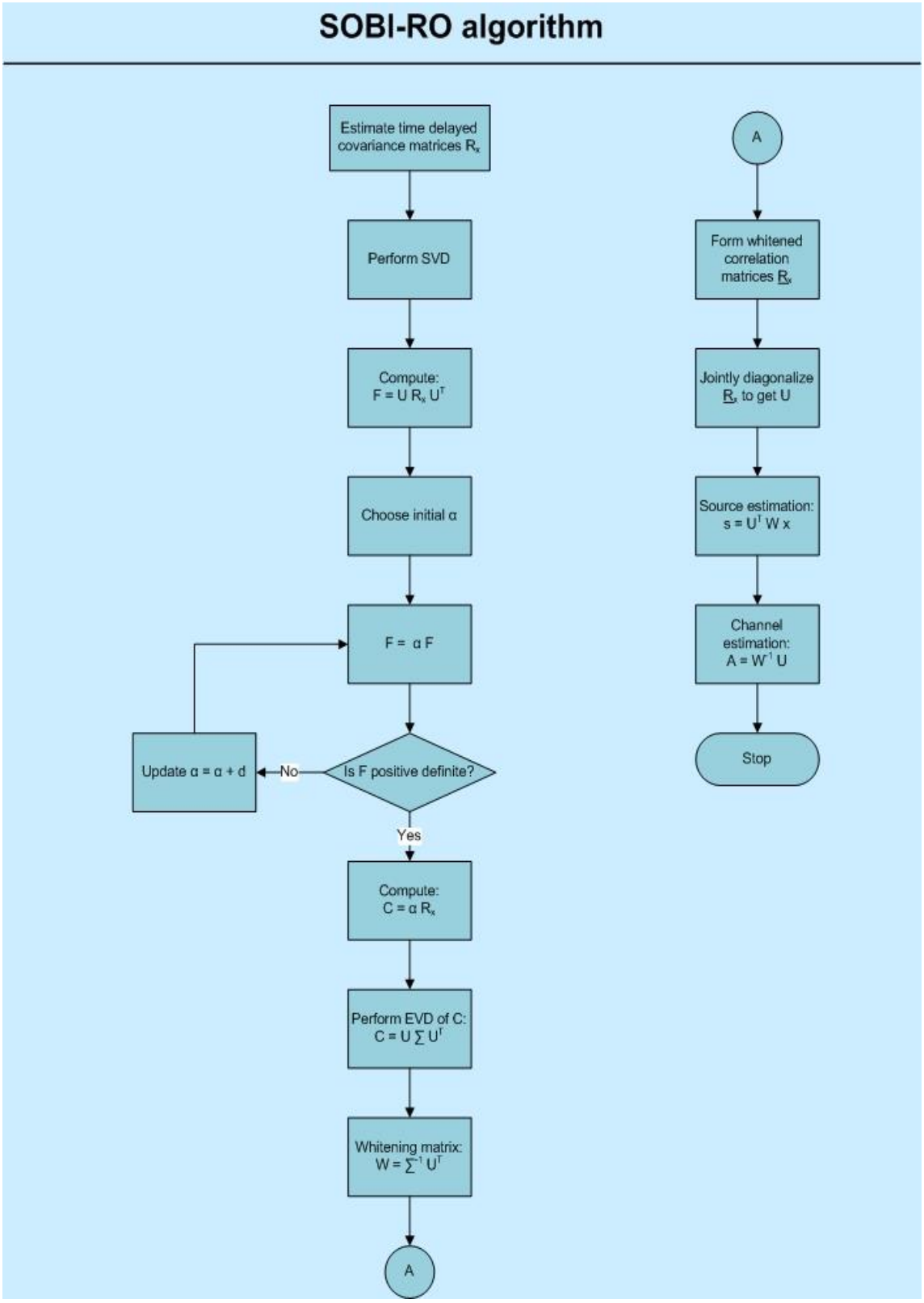

Figure 2.3 SOBI-RO algorithm

The algorithm for SOBI-RO is explained as follows:

1) Estimate the time-delayed correlation matrices and perform SVD of the matrix set $R = [\hat{R}_x(1) \quad \dots \quad \hat{R}_x(K)]$:

$$R = U_R \, \Sigma \, V^T \tag{2.10}$$

where $U_R \in \mathbb{R}^{n \times n}$ and $V \in \mathbb{R}^{nK \times nK}$ and Σ is a diagonal matrix.

2) For $i = 1, \dots, K$ compute:

$$F_i = U_R^T \, \hat{R}_x(i) \, U_R \tag{2.11}$$

3) Choose initial $\alpha \in \mathbb{R}^n$.

4) Compute:

$$F = \sum_{i=1}^{K} \alpha_i F_i \tag{2.12}$$

5) Test Schur decomposition of $F \in \mathbb{R}^{n \times n}$: if F is positive definite go to step (6). If not, chose an eigen-vector u corresponding to the smallest eigen-value of F and update α with α+d, where

$$d = \frac{[u^T F_1 u \quad \dots \quad u^T F_K u]^T}{\|[u^T F_1 u \quad \dots \quad u^T F_K u]\|} \tag{2.13}$$

and go to step (4).

6) Compute:

$$C = \sum_{i=1}^{K} \alpha_i \hat{R}_x(i) \tag{2.14}$$

7) Perform EVD of C:

$$C = U_C \, diag[\lambda_1^2 \quad \dots \quad \lambda_n^2] \, U_C^T \tag{2.15}$$

Then the whitening matrix is

$$W = diag[\lambda_1 \quad \dots \quad \lambda_n]^{-1} U_C^T \tag{2.16}$$

8) Form whitened correlation matrices:

$$\underline{\hat{R}}_x(i) = W\ \hat{R}_x(i)\ W^T, \forall\ i = 1, \dots, K \tag{2.17}$$

9) Jointly diagonalize the set $\{\underline{\hat{R}}_x(i) | i = 1, \dots, K\}$ to get a unitary matrix U.

10) Source estimation:

$$\hat{s}(t) = U^T\ W\ x(t) \tag{2.18}$$

And, the channel estimation:

$$\hat{A} = W^{-1}\ U \tag{2.19}$$

For proof of finite step based global convergence algorithm, refer to Tong, 1991 and, Tong, 1992 [5] [4].

2.2.4 Fixed Point Independent Component Analysis

Fixed point ICA algorithm is a modification of neural network learning rule into fixed point iteration. This algorithm finds independent components, irrespective of the probability distribution. Fixed point algorithm is based on minimization (or maximization) of Kurtosis [3].

The flowchart of the fixed point ICA algorithm is shown in Figure 2.4.

The basic form of the fixed point ICA algorithm is as follows:

1) Apply whitening transformation:

$$\tilde{x} = E\ D^{-1/2} E^T\ x \tag{2.20}$$

E and D are obtained from EVD of covariance matrix:

$$E\{xx^T\} = E D^{-1/2} E^T \tag{2.21}$$

2) Choose any random $w(0), such\ that\ \|w(0)\| = 1$. Let k=1.

3) Compute new weights $w(k)$ by

$$w(k) = E\{\tilde{x}(w(k-1)^T\ \tilde{x})^3\} - 3w(k-1) \quad (2.22)$$

4) Use one of the below mentioned decorrelation scheme:

a. Apply deflation scheme based on Gram-Schmidt like decorrelation:

$$w(k) = w(k) - \sum_{j=1}^{k-1}\left(w(k)^T\ w(j)\right)\ w(j) \quad (2.23)$$

b. Apply symmetric decorrelation:

$$W = (W\ W^T)^{-1/2}W \quad (2.24)$$

where $(W\ W^T)^{-1/2}$ is obtained from EVD of $W\ W^T = FDF^T$and $(W\ W^T)^{-1/2} = F\ D^{-1/2}F^T$.

c. Use iterative algorithm [1]:

i. $$W = W/\sqrt{\|W\ W^T\|} \quad (2.25)$$

ii. Repeat until convergence:

$$W = \frac{3}{2}\ W - \frac{1}{2}\ WW^TW \quad (2.26)$$

5) Perform: $w(k) = \frac{w(k)}{\|w(k)\|}$

6) Check for convergence: If $|w(k)^Tw(k-1)| \sim 1$, go to step 7; otherwise let $k = k + 1$ and then go back to step 3.

7) The source signals are estimated as

$$\hat{s}(t) = w\ \tilde{x}(t) \quad (2.27)$$

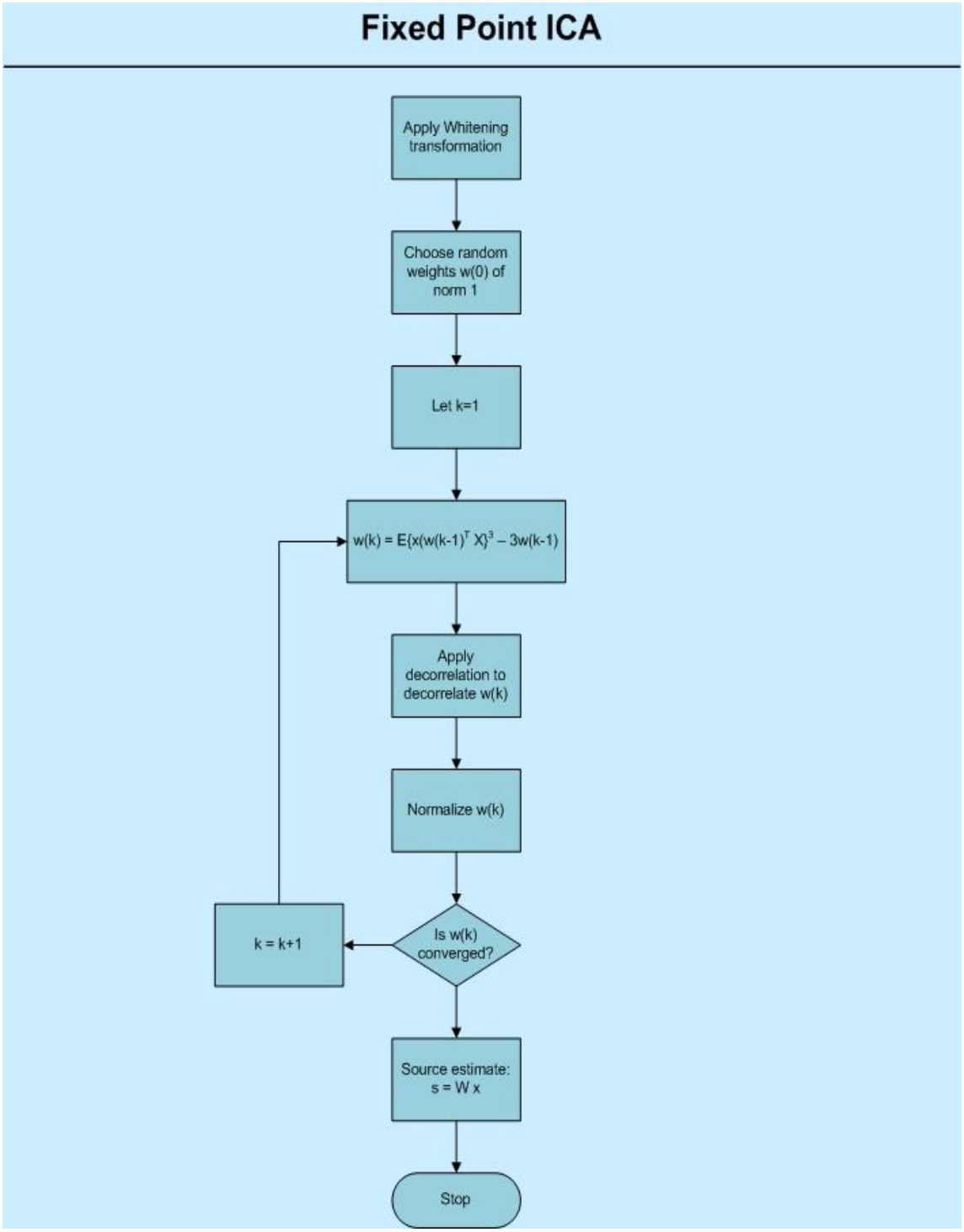

Figure 2.4 Fixed Point ICA algorithm

For proof for the convergence of the fixed point algorithm, refer to Hyvärinen & Oja, 1997 [2]. The fixed point algorithm has convergence that is cubic and has no step size parameter to choose like gradient based algorithm: fpICA algorithm does not have convergence problems. Fixed point ICA algorithm shares the advantages of neural

networks: parallel, distributed, computationally simple, and requires very less memory [3].

2.2.5 Joint Approximate Diagonalization of Eigen-Matrices

JADE algorithm is derived from the principle that optimizing cumulant approximations of the data performs the estimation of distribution of independent components. JADE algorithm uses fourth order cumulants to estimate the channel (and signal).

The flowchart depicting the steps of JADE algorithm is shown in Figure 2.5.

The JADE algorithm can be described as follows [13]:

1) Form the sample covariance $\hat{R}_x$ and compute Whitening matrix $\widehat{W}$. Under white noise assumption, noise variance $\hat{\sigma}$ is equal to the smallest $m-n$ eigenvalues of $\hat{R}_x$. Let $\mu_1, \mu_2, .., \mu_n$ be n largest eigenvalues and $h_1, h_2, ..., h_n$ be the corresponding eigenvectors of $\hat{R}_x$. The whitening matrix is then defined as

$$\widehat{W} = \left[(\mu_1 - \hat{\sigma})^{-1/2} h_1, (\mu_2 - \hat{\sigma})^{-1/2} h_2, ..., (\mu_n - \hat{\sigma})^{-1/2} h_n \right]^T \qquad (2.28)$$

2) Form the sample forth-order cumulants $\hat{D}_z$ of the whitened process $\hat{z}(t) = \widehat{W} x(t)$ and compute the n most significant Eigen-pairs $\{\hat{\lambda}_r, \widehat{M}_r | 1 \leq r \leq n\}$.

3) Jointly diagonalize the set $\widehat{N}^e = \{\hat{\lambda}_r, \widehat{M}_r | 1 \leq r \leq n\}$ by a unitary matrix $\widehat{U}$. For n=2, a unique Givens rotation achieves joint diagonalization and for n>2, use Jacobi iteration method to diagonalize matrix.

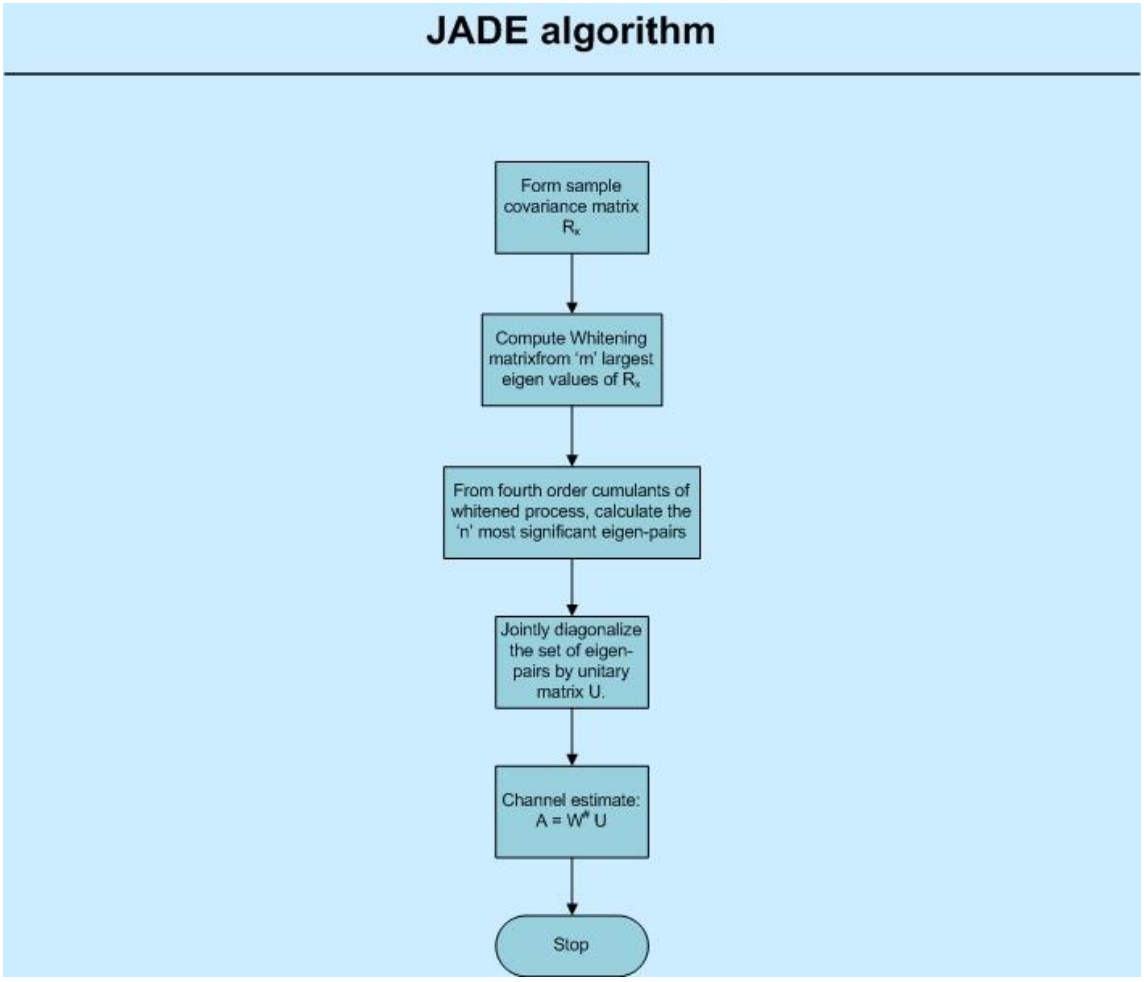

Figure 2.5 JADE algorithm

4) An estimate of A is $\hat{A} = \widehat{W}^{\#}\,\widehat{U}$. $\widehat{W}^{\#}$ is the pseudo-inverse of $\widehat{W}^{\#}$ calculated as:

$$\widehat{W}^{\#} = \left[(\mu_1 - \hat{\sigma})^{1/2} h_1, (\mu_2 - \hat{\sigma})^{1/2} h_2, \ldots, (\mu_n - \hat{\sigma})^{1/2} h_n \right] \tag{2.29}$$

The advantage of JADE algorithm is that it does not have convergence problem owing to the fact that it does not require gradient descent [25]. A drawback that JADE algorithm shares with AMUSE algorithm is that both don't have any parameters for performance tuning.

2.2.6 Extended Fourth Order Blind Identification

EFOBI is an Eigen structure based method for blind separation of multiple source signals in spatially correlated noise. The main assumption required for this algorithm to separate signals apart from mutual independence is that their fourth order moments should be different. Also, this algorithm does not work for underdetermined mixed signals.

The flowchart depicting the EFOBI algorithm is shown in Figure 2.6.

The EFOBI algorithm is as follows [18]:

1) Estimate the covariance matrix

$$R_x = E\{x(t)\, x^T(t)\} \tag{2.30}$$

2) Computer SVD:

$$R_x' = [U_s U_n]\, diag(\lambda_1^2, \lambda_2^2, .., \lambda_n^2)\, [U_s U_n]^T \tag{2.31}$$

3) Apply transformation:

$$C = diag\left[\frac{1}{\lambda_1}, \dots, \frac{1}{\lambda_n}\right] U_s^T \tag{2.32}$$

4) Estimate the weighted covariance matrix:

$$R_z = E\{|z|^2 z(t) z^T(t)\} \tag{2.33}$$

5) Compute: $R_z' = R_z\, \Delta R_z$, where

$$\Delta R_z = (m+4)\, C \Sigma_o C^T - C\, \emptyset\, C^T \tag{2.34}$$

Where, $\emptyset = \sum_{j=1}^{n} \sum_{k=1}^{n} \left(\sum_{i=1}^{m} (1/\lambda_i^2) u_{ji} u_{ki}\right) \psi_{jk}$, u_{ji}s are elements of U_s.

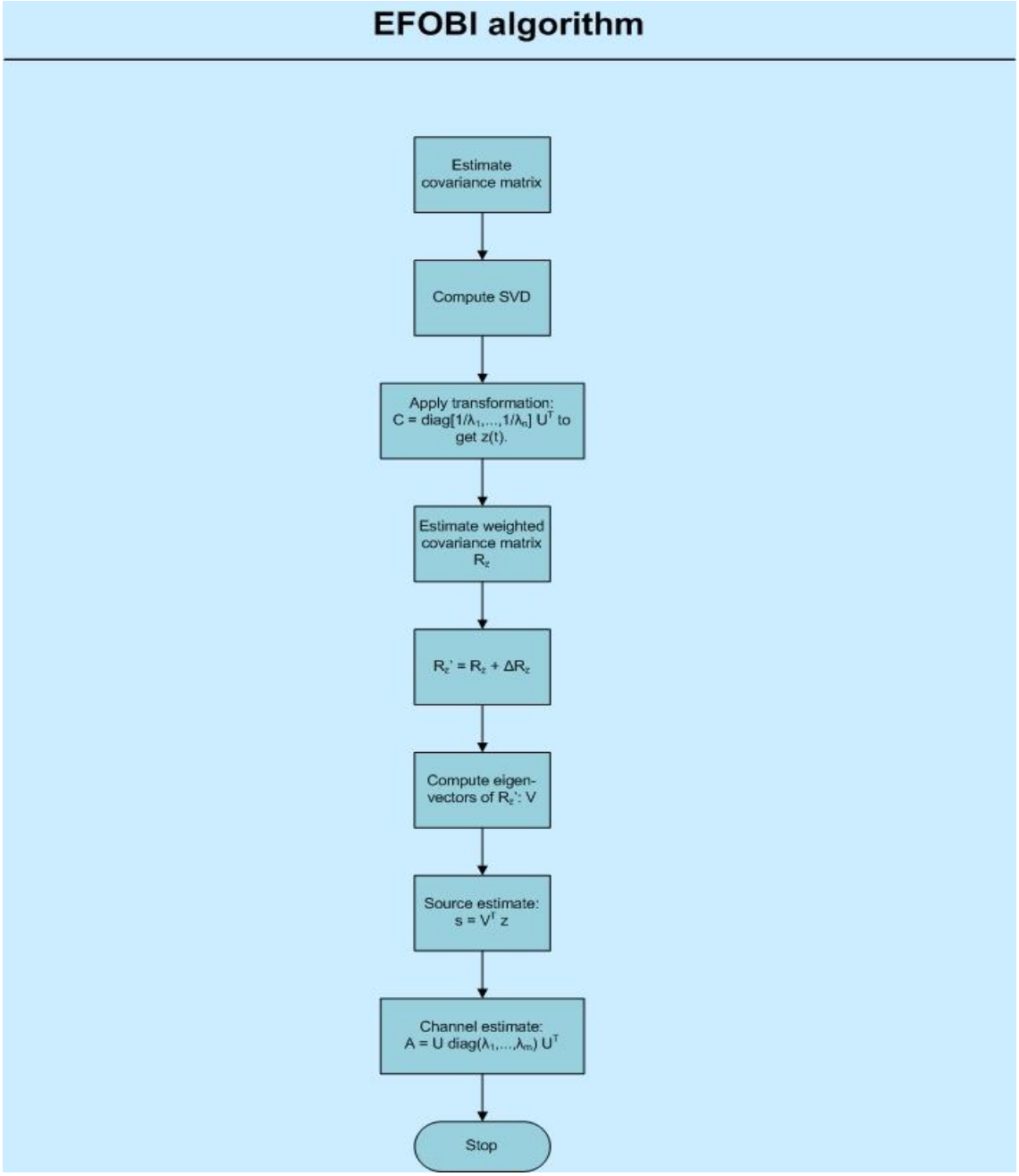

Figure 2.6 EFOBI algorithm

6) Compute eigenvectors of R_z' denoted by $V = [v_1 \quad \dots \quad v_m]$

7) Estimate source signals:

$$\hat{s}(t) = V^T z(t) \tag{2.35}$$

8) Estimate mixing matrix:

$$\hat{A} = U_s \, diag(\lambda_1, .., \lambda_m) \, V \tag{2.36}$$

CHAPTER 3
PERFORMANCE MEASURES

3.1 INTRODUCTION

Before proceeding to compare various BSS algorithms, it is prudent to study few performance measures. Because of the permutation and scaling (and sign) ambiguity, traditional comparative tools like least mean square error, Euclidean distance cannot be applied. Correlation will not measure the interference from the other sources and hence would not be a robust performance measure. Also, there would be no way to measure algorithmic 'artifacts' by all these methods. Hence, more robust performance measures needs to be applied for performance analysis of these algorithms.

3.2 PERFORMANCE MEASURES

Since BSS can only estimate sources up to permutation ambiguity and sign ambiguity as discussed in Chapter 1.2, the direct definition of relative distortion given by Equation (3.1) cannot be applied.

$$D_1 = \|\hat{s}_n - s_n\|^2 / \|\hat{s}_n\|^2 \tag{3.1}$$

Here, $s(t)$ is the original source signal and $\hat{s}(t)$ the estimated signal. The gain indeterminacy can be handled by comparing L_2 normalized versions of sources with their relative square distance as given by Equation (3.2) [12].

$$D_2 = \min_{\epsilon=\pm 1} \left\| \frac{\hat{s}_j}{\|\hat{s}_j\|} - \epsilon \frac{s_j}{\|s_j\|} \right\|^2 \quad (3.2)$$

The disadvantage of this method is when there is no contribution of true source to an estimate; we get an estimate $D_2 = 2$ instead of the expected $D_2 = \infty$. To solve this problem, total relative distortion given by Equation (3.3) is used.

$$D_{total} = \frac{\|\hat{s}_n\|^2 - |\langle \hat{s}_n, s_n \rangle|^2}{|\langle \hat{s}_n, s_n \rangle|^2} \quad (3.3)$$

When the estimated source is orthogonal to the true source, we get $D_{total} = \infty$; which makes D_{total} a more appropriate measure than D_2.

The definition of D_{total} contains two terms in the decomposition:

$$\hat{s}_n = \langle \hat{s}_n, s_n \rangle s_n + e_{total} \quad (3.4)$$

The error term e_{total} includes contribution of interferences from other sources, noise and artifacts of the separating algorithm. Thus, the estimated source has an orthogonal decomposition:

$$\hat{s}_n = \langle \hat{s}_n, s_n \rangle s_n + e_{interf} + e_{noise} + e_{artif} \quad (3.5)$$

where $e_{interf} = \sum_{l \neq n} \langle \hat{s}_n, s_n \rangle s_n$ is the error term due to interference due to other sources, $e_{noise} = \sum_{k=1}^{N} \langle \hat{s}_n, n_k \rangle n_k$ is the error term due to the additive noise and $e_{artif} = \hat{s}_n - \langle \hat{s}_n, s_n \rangle s_n - e_{interf} - e_{noise}$ is the error term due to the artifacts of the separating algorithm [11].

The relative distortion due to interferences is defined as

$$D_{interf} = \frac{\|e_{interf}\|^2}{|\langle \hat{s}_n, s_n \rangle|^2} \tag{3.6}$$

The relative distortion due to additive noise:

$$D_{noise} = \frac{\|e_{noise}\|^2}{\|\langle \hat{s}_n, s_n \rangle | s_n + e_{interf}\|^2} \tag{3.7}$$

The relative distortion due to algorithmic artifacts:

$$D_{artif} = \frac{\|e_{artif}\|^2}{\|\langle \hat{s}_n, s_n \rangle | s_n + e_{interf} + e_{noise}\|^2} \tag{3.8}$$

Now, the Source to Distortion Ratio (SDR) is defined as:

$$SDR = 10 \log_{10} D_{total}^{-1} \tag{3.9}$$

Source to Interference Ratio (SIR) is defined as:

$$SIR = 10 \log_{10} D_{interf}^{-1} \tag{3.10}$$

Source to Noise Ratio (SNR) is defined as

$$SNR = 10 \log_{10} D_{noise}^{-1} \tag{3.11}$$

and, Source to Artifacts Ratio (SAR) is defined as:

$$SAR = 10 \log_{10} D_{artif}^{-1} \tag{3.12}$$

In this thesis, evaluation of the aforementioned performance indices is done with the help of BSS EVAL toolbox [41].

3.3 COMPARISON OF BSS ALGORITHMS

To select an algorithm for implementation in online BSS, we analyse the performance of these algorithms. The testing of these algorithms is carried out with the help of ICALAB toolbox [26] [27].

3.3.1 Time

Time of execution of the algorithm should be less for online implementation. The table showing the time required for the execution of algorithm of various algorithms for four different signal mixtures is shown in Table 3.. Bar charts showing the execution time of various algorithms for these signals are shown in Figure 3.1.

Table 3.1 Comparison of processing time of various algorithms

Time (s)						
Signal	**AMUSE**	**SOBI**	**SOBI-RO**	**fpICA**	**JADE**	**EFOBI**
ACsin4D	0.01	0.01	0.03	0.03	0.01	0.27
Speech4D	0.02	0.02	0.05	0.03	0.01	0.01
Speech_Music2D	1.04	0.97	9.95	2.83	0.55	0.78
Speech2D	0.82	0.8	9.77	2.67	0.49	0.60

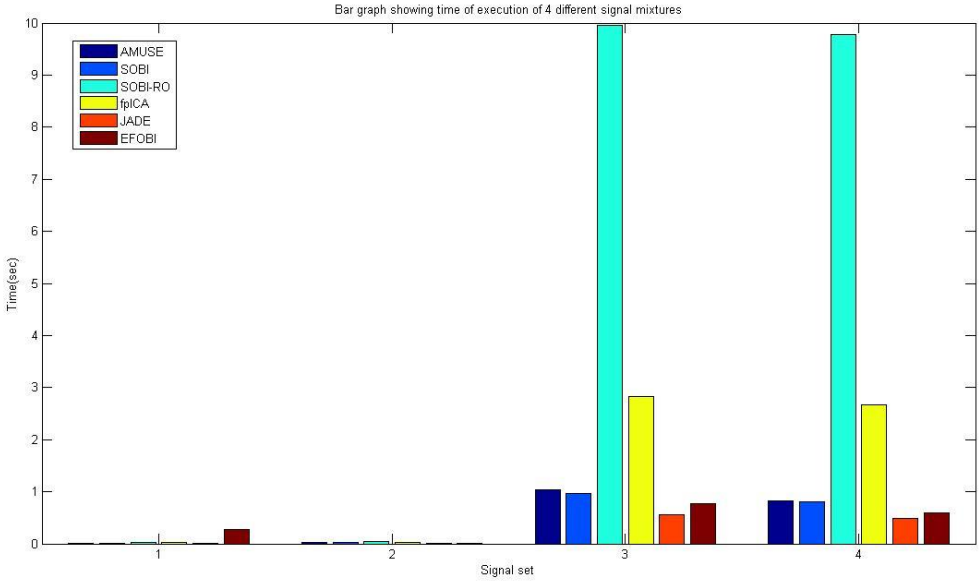

Figure 3.1 Bar chart comparing time taken for processing of various algorithms

3.3.2 Signal to Interference Ratio

A high SIR is desirable for any BSS algorithm. High SIR means the signal strength of the desired source is higher than the interference from other sources: exactly what is expected out of any BSS algorithm. SIRs of these algorithms on different signal mixtures is shown in Table 3.. Bar chart showing SIR for various algorithms are shown in Figure 3.2.

Table 3.2 Comparison of SIR of various algorithms

SIR (dB)						
Signal	**AMUSE**	**SOBI**	**SOBI-RO**	**fpICA**	**JADE**	**EFOBI**
ACsin4D	277.50	54.68	289.78	10.07	9.74	4.38
Speech4D	21.85	29.61	32.91	29.62	24.84	13.99
Speech_Music2D	6.01	6.01	6.01	6.01	12.27	6.09
Speech2D	6.04	6.04	6.03	6.03	12.60	5.92

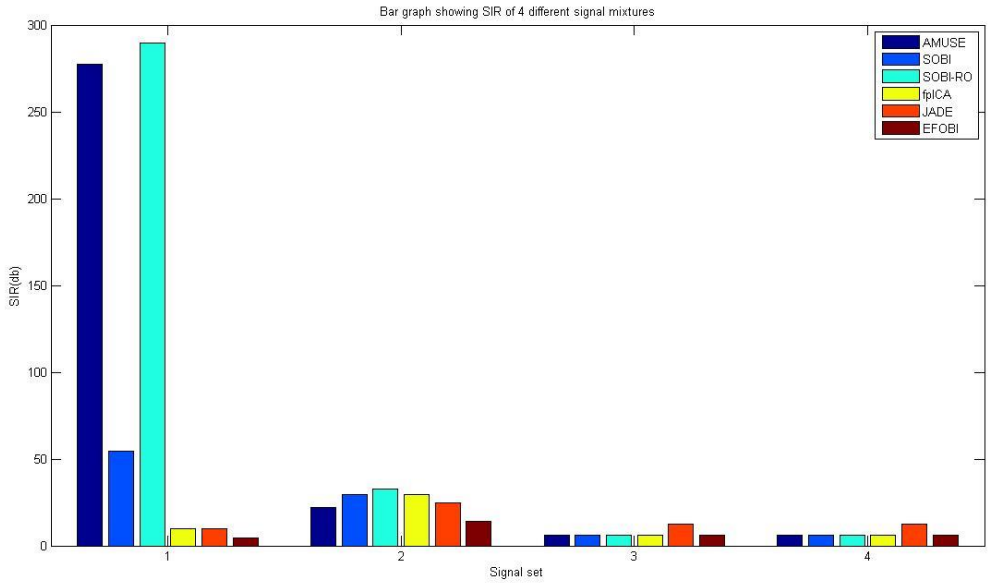

Figure 3.2 Bar chart comparing SIRs of various algorithms

3.3.3 Signal to Artifacts Ratio

In many speech signal processing applications, artifacts introduced by the algorithm may be more undesirable than the interference by other sources. So, a high SAR is desirable of any BSS algorithm. Table showing SARs of these algorithms on four different signal mixtures is shown in Table 3.. Figure 3.3 shows the bar chart of SAR of various algorithms for the signal mixtures.

Table 3.3 Comparison of SAR of various algorithms

SAR (dB)						
Signal	**AMUSE**	**SOBI**	**SOBI-RO**	**fpICA**	**JADE**	**EFOBI**
ACsin4D	302.44	300.95	298.96	311.77	299.44	313.56
Speech4D	276.53	285.22	282.71	286.06	282.06	285.99
Speech_Music2D	255.92	262.33	262.99	268.75	257.71	259.08
Speech2D	253.17	253.52	259.56	257.11	256.77	253.75

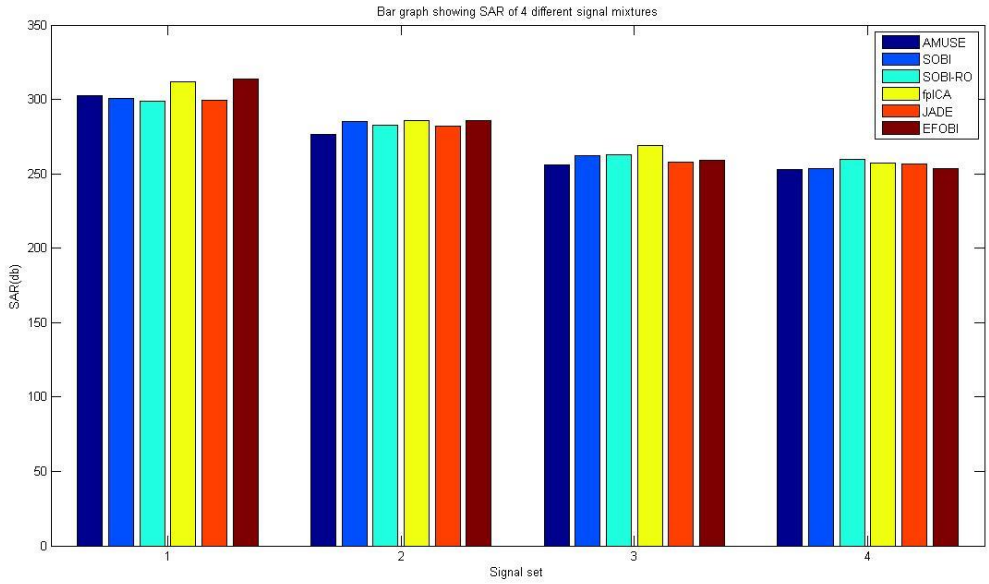

Figure 3.3 Bar chart comparing SARs of various algorithms

3.4 SELECTION OF ALGORITHM FOR ONLINE IMPLEMENTATION

It can be observed from Figure 3.1 that all algorithms except SOBI-RO and fpICA take very less time for execution even for large signals. SOBI-RO would not be a good fit for the online implementation because the time taken for execution is almost 10sec for 1 minute audio.

As it can be observed in Figure 3.2, the SIR of AMUSE and SOBI-RO are high compared to other algorithms in ACsin4D. For the other mixtures, performance of all the algorithms are relatively the same although JADE algorithm's SIR is higher in mixtures – Speech_Music2D and Speech2D. It can be seen from Figure 3.3 that SAR is relatively constant for all the algorithms.

Out of the algorithms that fared well in separation of signals, the ones that have high SIR consistently: AMUSE and SOBI-RO; SOBI-RO takes very high time to separate

the signals as compared to the other algorithms. So, AMUSE algorithm will be a very good selection for online implementation.

The advantages of AMUSE algorithm bolster the selection criteria: the automatic ordering of signals based on their singular values. Although this does not completely solve the permutation problem, it ensures that the separated signals are not identical. This makes AMUSE algorithm an ideal algorithm for online implementation.

CHAPTER 4
SOFT COMPUTING TECHNIQUES

4.1 INTRODUCTION

Soft computing is the field of computer science which deals with formulating inexact solutions to problems where conventional techniques cannot be applied easily. Soft computing techniques also yield low cost solutions to problems which can be quite intractable. Soft computing originated with machine learning techniques like neural networks. Recent trends tend to involve evolutionary and biological inspired techniques into soft computing techniques. Some of the soft computing techniques used exhaustively in engineering are: neural networks, support vector machines, fuzzy logic, and evolutionary algorithms.

4.2 GENETIC ALGORITHM

GA is an evolutionary algorithm which uses method of natural selection to solve the optimization problem. Unlike the traditional optimization algorithms which evaluate the fitness value at every point per iteration, GA evaluates the fitness value for a population per iteration.

These population of individuals constituting an iteration is called ‘parents’ and every iteration is called a ‘generation’. Based on the fitness values, few parents are selected which will reproduce to create the next generation. The process of reproduction to create next generation is called ‘crossover’. ‘Mutation’ is the process of applying some random changes to a parent to form a child. Crossover preserves the traits of the individuals who were the best in the current generation. The process of crossover will not explore the search space effectively. Mutation introduces changes to the individual

such that new points are explored, assisting the search for global optimum. The three processes: random selection of a population of parents, mutation and crossover makes GA a powerful tool in global optimization. Even though, there is no guarantee that GA will provide global optimum, smoothening of the optimization landscape helps convergence to global optimum.

4.3 MODELLING OF A SYSTEM

The model of the algorithm is important in three aspects:

1) The model can be approximated to fit the general tendency of the BSS algorithm rather than optimizing the data for a specific data set.
2) The model results in less time consuming optimization.
3) The model can be tuned to avoid 'overfitting' so that optimization landscape will be a smooth terrain instead of a jagged one.

In this thesis, modelling of BSS algorithm is done with the help of three techniques: Polynomial regression, neural networks and ANFIS.

4.3.1 Polynomial Regression

Polynomial regression is the statistical technique which models the statistical relationship between the input and the output as n^{th} order polynomial. Polynomial regression is essentially linear regression even though it fits a non-linear surface for the data. Consider an independent variable 'x' and a dependent variable $y = f(x)$. Then the polynomial regression model is

$$\hat{y} = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n \tag{4.1}$$

The polynomial fit is then computed by estimating the unknown parameters: $a_0, a_1, \dots, a_n$ so that the MSE between the estimate $\hat{y}$ and the target y is reduced i.e. the objective function here is

$$\underset{x}{minimize}\ \sqrt{(\hat{y} - y)^2} \tag{4.2}$$

With increase in the order of the polynomial, we face the problem of 'Overfitting': the regression model models the high frequency noise, measurement and random errors instead of modelling the underlying relationship between the input and output. The result is that the model will add random variations to the subsequent predictions [37]. Overfitting can be reduced to an extent by limiting the order of the polynomial terms.

Scaling of the input data is an essential pre-processing step in polynomial regression to ensure convergence. One way is to use scale the input so that they fall in the specified range, usually $[-1,1]$. The minimum and maximum of the data needs to be stored so that they can be applied to the testing and validation data in subsequent stages. Another way is to normalize the data based on the mean and the standard deviation. Here, the mean and the standard deviation need to be stored so that they can be applied for the testing and validation data.

4.3.2 Artificial Neural Networks

Artificial Neural Networks (ANN) is an extension of statistical regression techniques. ANN models the relationship between input and output automatically without the need of the user to compute the polynomial terms, pre-process them and feed them into the linear regression model. ANN achieves complex fit of data with the help of hidden layers. Hidden layers are units that are lie between the input and the output layer. Every unit in ANN, other than the input layer, receives its input by weighted linear addition of the activation signals of the previous layers. Once the input is received, the neuron is activated and it transmits an output signal to the next layer.

The weights of the interconnection between layers need to be trained so that the neural network will follow the target for the specific input pattern. The weights are trained so that MSE between the target and the output is reduced. There are several training algorithms like gradient descent, conjugate gradient methods, Gauss-Newton methods, Levenberg-Marquardt (LM) method, and resilient backpropagation method and so on. The convergence and MSE are the most important qualifiers in selecting the algorithm for neural network training.

Also, neural networks may overfit the data when there are many hidden layers and hidden neurons. ‘Overfitting’, although will reduce the MSE, will result in random variations in the subsequent predictions which are undesirable. ‘Overfitting’ can be avoided by careful selection of hidden layers and hidden units; and choosing a proper ‘Regularization parameter’.

4.3.3 Adaptive Neuro Fuzzy Inference System

ANFIS is a soft computing technique that is based on Sugeno fuzzy inference system. Since ANFIS combines both neural networks and fuzzy logic, it extracts the benefits of both them. Fuzzy model cannot be applied since fuzzy necessitates rule structure which is formulated by the user's understanding of the system. Also, changing the rules of the system can generate local/global optima thus rendering the search unreliable.

On contrary, ANFIS can be used for data modelling which can be further applied for optimization. ANFIS system generates fuzzy IF-THEN rules and then tunes the parameters of the membership function parameters with the help of a neural network trained by the input-output training data. ANFIS network can be tuned by changing the parameters: Number and shape of membership function of input. ANFIS architecture is functionally equivalent to Radial Basis Function Network [39].

ANFIS can model non-linear data successfully yielding successful results. Since its inception, ANFIS has been widely utilized in control systems and signal processing. We analyse the possibility of utilizing ANFIS model in optimizing online BSS.

CHAPTER 5
ONLINE BLIND SIGNAL SEPARATION

5.1. INTRODUCTION

Most of the literature on BSS concentrates on batch processing of audio signals. In online time-domain BSS, the incoming signal is processed in small batches. This will introduce a time delay equal to the batch size plus the processing time of the algorithm to separate the signal. Larger the batch size, better the performance of the algorithm will be (all algorithms exploit the statistical properties of the signal) but it increases the delay. Online BSS with small batch size can separate sources that are non-stationary but the quality of the separated signal may not be good. Careful selection of the batch size is necessary.

In this thesis, after analysing various algorithms, AMUSE algorithm is selected to implement online BSS as discussed in Chapter.3.4.

5.2. PERMUTATION PROBLEM

As discussed in Chapter 1.1.11.1.2, BSS algorithms can only estimate signals leaving sign and permutation ambiguity. The sign ambiguity is not a serious problem in audio signal processing. In online BSS, permutation problem may cause signals from different sources to be joined erroneously in the recovered signal causing serious degradation.

The main limitation of time domain BSS techniques that will adversely impact the performance of online implementation is the permutation problem.

None of the BSS techniques can order the source signals in specific order, unless "semi-blind" processing is done exploiting some properties of source signals. Since the source signals are absolutely random, no techniques based on their statistical properties can be employed to solve the permutation problem. Techniques like linear prediction fail miserably as they do not predict speech signals effectively. The permutation problem at hand necessitates a very robust technique.

5.3. ALGORITHM

In this thesis, we propose a novel method to deal with the permutation problem for online BSS without jeopardising the 'blind' assumption. This technique does effectively solve the problem at hand without increasing the processing time significantly. The newly recovered signal is matched with the previously processed samples' overlapping region. The permutation ambiguity is then solved by joining the newly recovered signal samples with the signal stream that has the highest SIR in the overlapping region. The comparison cannot be D_1, D_2 *or* D_{total} because these techniques do not provide a robust estimate of the signal with its source. SIR proves to be a robust measure in matching the signal as it is not affected by sign or scaling ambiguity.

The detailed algorithm implementation of the technique of overlapping batches with AMUSE algorithm for BSS is explained below:

1) Choose an appropriate Window size = 'n' and percentage overlap = 'p'. Let i=0.

2) Choose $s_j = \{s_j(t) | t_{start} \leq t \leq t_{end}\} \forall\, j = 1, .., m,$ where

$t_{start} = i * n * (1 - p) + 1$ and $t_{end} = t_{start} + n$.

3) Apply AMUSE algorithm:

i. Estimate the output covariance by Equation (2.1).

ii. Compute the SVD of R_x by Equation (2.2).

iii. Estimate the number of sources m from the number of significant singular values.

iv. Perform Orthogonalization transformation as explained in Equation (2.3).

v. Select a τ such that $(R_y(\tau) + R_y(\tau)')/2$ has distinct eigenvalues, where $R_y(\tau) = E\{y(t)\ y(t-\tau)'\}$.

vi. Let V be the Eigen matrix obtained from the Eigen-decomposition of $(R_y(\tau) + R_y(\tau)')/2$.

vii. Channel estimation is given in Equation (2.4) and Signal estimation by Equation (2.5).

4) If $i \neq 0$, estimate the SIR of the overlap part of the i^{th} estimates and $(i-1)^{th}$ estimates. SIR of matched signal will be higher than the other signals.

5) Order the signals based on their SIRs and append signal excluding the overlap part.

6) If end of the signal is not reached, $i = i + 1$ and go to step (2).

5.4. FLOWCHART

The flowchart of the technique discussed in previous section is explained in Figure 5.1.

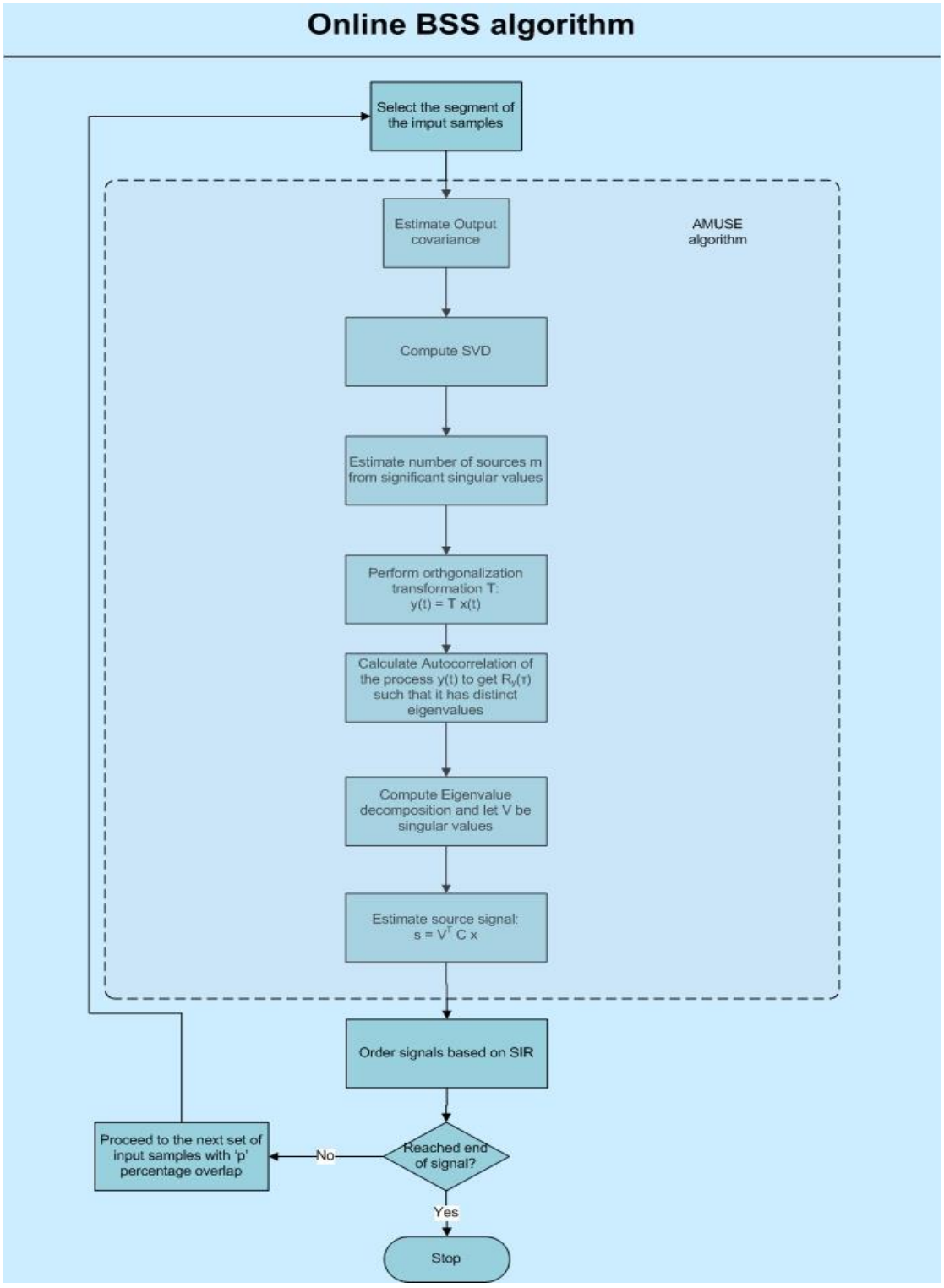

Figure 5.1 Online BSS with overlapping samples

CHAPTER 6
OPTIMIZATION

6.1 INTRODUCTION

By optimization, we mean the process of selection of the best set of input variables among all the possible sets of inputs to achieve the lowest possible objective function value. The objective function is the mathematical model that best depicts the parameter to be optimized in terms of the input parameters. Let us assume there are 'n' input variables in the optimization process. The domain of the input parameters, the optimization process searches to find the optimum, is called the 'search space'. In an unconstrained optimization, the search space is the whole Euclidean space $\mathbb{R}^n$. In a constrained optimization, the search space is limited by equalities and inequalities every search element has to satisfy.

The first step in any optimization process is the formulation of the objective function which the optimization process has to minimize (or maximize). If the optimization function is convex, the global minima can be found out by any convex optimization techniques. The convexity property ensures that there is only one global optimum making it easier to find it by one of the conventional procedures: Gradient descent, Newton's method, Interior Points methods and so on [36].

For most of the practical systems, mathematical model takes a highly non-linear form resulting in an objective function having many local optima. Some of the techniques used to model an unknown process are discussed in Chapter 4.3. The aforementioned

conventional procedures for these systems will converge into one of these local optima which need not be the global optimum. The optimization landscape for BSS processes are highly non-linear as can be observed from Figure and Figure . This necessitates the use of global optimization techniques that may possibly find out the global optima. In this thesis, GA based optimization technique is used for finding the global optimum.

With the technique discussed in 0 5, we have two parameters of the system: Segment length and Percentage overlap which needs careful selection. Segment length parameter if chosen large, can introduce huge delays. On contrary, the algorithm's performance can be hampered if chosen small. Similarly, choice of large Percentage overlap will cause additional delays but will solve the permutation problem more efficiently. Percentage Overlap if chosen too small may not help in solving the permutation problem. These two parameters require additional optimization to effectively solve the permutation problem keeping in check the time delay introduced. It is therefore necessary to optimize these parameters; with the optimization objective to increase the SAR and SIR at the same time reducing the time delay. The objective function takes the form of Equation (6.1).

$$Objective\ Function = f(SAR, SIR, T_d) \tag{6.1}$$

Time delay for processing is a very important consideration in real time. Since BSS techniques exploit the statistical properties of signal, time delay is inevitable. Let us denote the time delay introduced by the algorithm in separating the signals as $\mathbf{t}$ and the time delay encountered in collection of samples as $\mathbf{T}$.

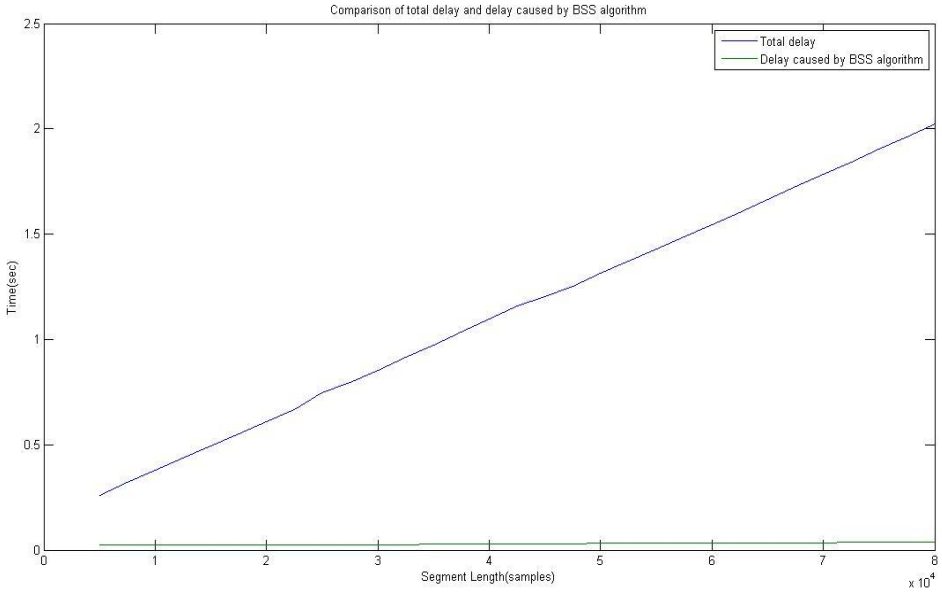

Figure 6.1 Comparison of T and t for a typical BSS run

Figure 6.1 is a typical BSS run comparing T with t: it can be observed that $T \gg t$. It is observed that even though t is relatively constant for the selected algorithm; T increases with increase in Segment Length, increasing time delay for data availability in real-time. The total time delay is given by Equation (6.2).

$$T_d = T + t \tag{6.2}$$

Based on the sample parameter inputs to the algorithm and their performance indices, the algorithm if first modelled before proceeding for optimization. The model of the algorithm is outlined in Figure 6.2 with the constraints given by Equation (6.3)

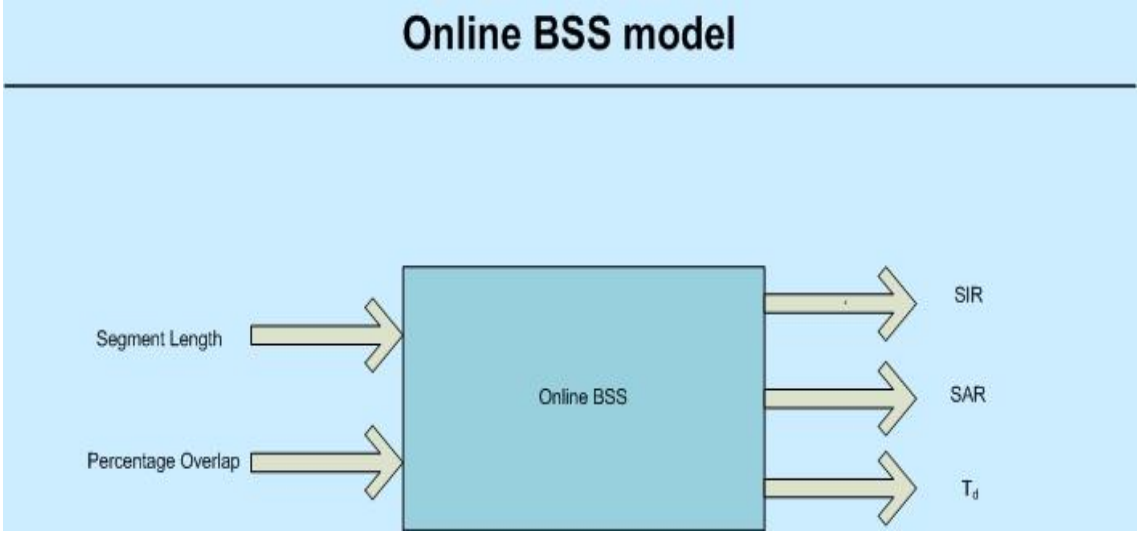

Figure 6.2 Optimization model for online BSS process

$$Segment\ Length \in [7500, 80000]$$

$$Percentage\ Overlap \in [0.2, 0.8] \quad (6.3)$$

6.2 FITNESS FUNCTION

The fundamental problem of optimization is constructing an appropriate fitness function for optimization. This fitness function is a measure of the performance of the process. The optimization objective would be to minimize (or maximize) the fitness value. Here, the optimization objective is to increase SAR and SIR at the same time reducing T_d. The fitness function formulated for this optimization objective is as follows:

$$Fitness\ function = K1 * (5 - SAR) + K2 * (100 - SIR) + K3 * Time \quad (6.4)$$

K1, K2 and K3 are scaling factors which determine the relative weightage of the components of SAR, SIR and Time respectively.

$$K1 = \frac{1}{3}, K2 = \frac{1}{30}, K3 = 2 \quad (6.5)$$

Since, quantities SAR and SIR have to be increased, the scaled negative quantities are taken so that the respective term can be minimized. SAR and SIR are deducted from constants 5 and 100 respectively in Equation (6.4) because the maximum observed values of SAR and SIR are nearly 101db and 7db respectively and the average values are nearly 2db and 70db respectively. If SAR = 2db, SIR = 70db and T_d = 1 sec; then each of these terms will add a value of one to the fitness function's value as per Equation (6.4). So, the optimization will be zero if SAR = 5db, SIR = 100db and T_d = 0 sec. Any values close to these values will give the ideal results. As per Equation (6.5), the relative weightage of SAR, SIR and T_d is 10:1:60. The values of Equation (6.5) are chosen based on trial and error to give Td the highest weightage and SAR the next.

6.3 POLYNOMIAL REGRESSION

6.3.1 Regression Model

Let us denote the input parameters: Segment Length and Percentage overlap by x_1 and x_2 respectively. Even though a higher order polynomial fit would give the least MSE, they tend to overfit the data. Also, when value of one of x1 or x2 is high, higher order terms tends to yield very high value as output. To avoid these issues, we limit the order of the polynomial to two. The Polynomial regression model with second order polynomial is given in Equation (6.6).

$$\hat{y} = a_0 + a_1x_1 + a_2x_2 + a_3x_1^2 + a_4x_2^2 + a_5x_1x_2 \quad (6.6)$$

Now, the problem descends to estimating the unknown parameters $a_0, \ldots, a_5$ so that the MSE between $\hat{y}$ and y is minimized with the constraints given by Equation (6.7).

$$7500 \leq x_1 \leq 80000 \; and \; 0.2 \leq x_2 \leq 0.8 \tag{6.7}$$

The training set used here is formed by executing the online BSS algorithm for 372 different input pairs recording the Segment Length, Percentage Overlap, SAR, SIR and T_d. Pre-processing of input and output data are performed to ensure convergence and the settings are saved so that these settings can be applied to subsequent validation, testing and optimization runs. Figure 6.3 shows that the Regression model converges with MSE ~ 0.5.

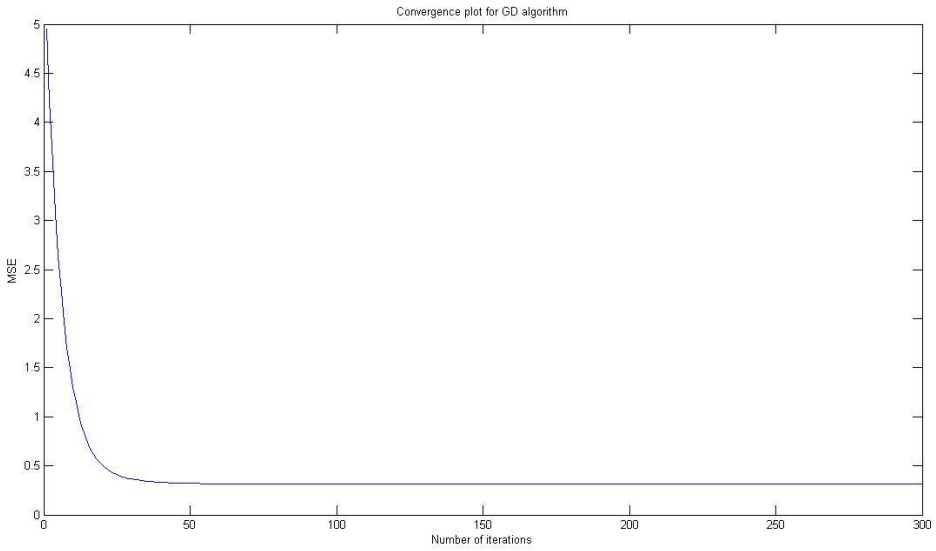

Figure 6.3 Convergence plot of Polynomial regression model

6.3.2 Optimization of polynomial regression based model

The optimization landscape for the polynomial regression model is given in Figure 6.4. The optimization is performed with different parameter settings in GA algorithm as shown in Table 6.1. These fitness values are compared with the optimized values of other parameter settings in GA and the best value is chosen.

The best optimized parameters from the Table 6.1 are:

$$Segment\ Length = \ 80000\ samples$$

$$Percentage\ Overlap = 80\ \% \tag{6.8}$$

Table 6.1 GA based optimization of Polynomial Regression based model for various parameters of the algorithm

Segment Length	% overlap	Theoretical				Simulation			
		SAR (db)	SIR (db)	Time (secs)	Fitness value	SAR (db)	SIR (db)	Time (secs)	Fitness value
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.8936	6.3949
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.8915	6.3908
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.9020	6.4117
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.8920	6.3917
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.8914	6.3906
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.8989	6.4054
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.8912	6.3902
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.9031	6.4138
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.8935	6.3947
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.8968	6.4013
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.8916	6.3910
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.8912	6.3902
80000	0.80	6.8383	70.3368	1.9563	4.2887	2.2398	49.3705	1.9008	6.4094
80000	0.80	6.8382	70.3367	1.9563	4.2887	2.2398	49.3705	1.9074	6.4224
80000	0.80	6.8382	70.3367	1.9563	4.2887	2.2398	49.3705	1.8908	6.3894
80000	0.80	6.8382	70.3367	1.9563	4.2887	2.2398	49.3705	1.8939	6.3956
80000	0.80	6.8382	70.3365	1.9563	4.2887	2.2398	49.3705	1.8915	6.3908
79999	0.80	6.8381	70.3363	1.9563	4.2887	2.2398	49.3705	1.8911	6.3899
79998	0.80	6.8380	70.3355	1.9563	4.2887	2.2398	49.3705	1.8933	6.3943
79997	0.80	6.8377	70.3344	1.9562	4.2888	2.2398	49.3705	1.8923	6.3923
79972	0.80	6.8337	70.3167	1.9557	4.2895	2.2398	49.3705	1.8903	6.3882
79906	0.80	6.8231	70.2700	1.9541	4.2916	2.2451	50.0611	1.9028	6.3886
79801	0.80	6.8062	70.1958	1.9517	4.2948	2.2585	48.7808	1.8939	6.4090
79740	0.80	6.7964	70.1529	1.9503	4.2966	2.3363	54.9430	1.8846	6.1590
79733	0.80	6.7953	70.1482	1.9501	4.2969	2.3363	54.9430	1.8967	6.1833
79710	0.80	6.7916	70.1315	1.9496	4.2976	2.3363	54.9430	1.8910	6.1719
79592	0.80	6.7727	70.0487	1.9469	4.3012	2.2788	53.2744	1.8825	6.2296
79587	0.80	6.7719	70.0449	1.9467	4.3013	2.2788	53.2744	1.8825	6.2297
79450	0.80	6.7499	69.9484	1.9436	4.3055	2.8204	62.5814	1.8794	5.7327
79385	0.80	6.7394	69.9020	1.9420	4.3075	0.0260	16.1069	1.8778	8.2100
79066	0.80	6.6882	69.6768	1.9346	4.3173	0.0783	44.3657	1.8714	7.2379
78965	0.80	6.6719	69.6053	1.9323	4.3204	0.2730	55.9020	1.8664	6.7785
78927	0.80	6.6659	69.5788	1.9314	4.3215	0.2496	61.3293	1.8643	6.6012
77674	0.80	6.4652	68.6957	1.9023	4.3596	0.4025	58.0488	1.8373	6.6055
77313	0.80	6.4076	68.4420	1.8939	4.3705	0.6075	40.9716	1.8281	7.0879
76109	0.80	6.2155	67.5956	1.8659	4.4068	-0.5586	17.1745	1.7998	8.2134

With these parameters, we get the performance indices shown in Table 6.4. These values are obtained with the GA parameters provided in Table A 2.1.

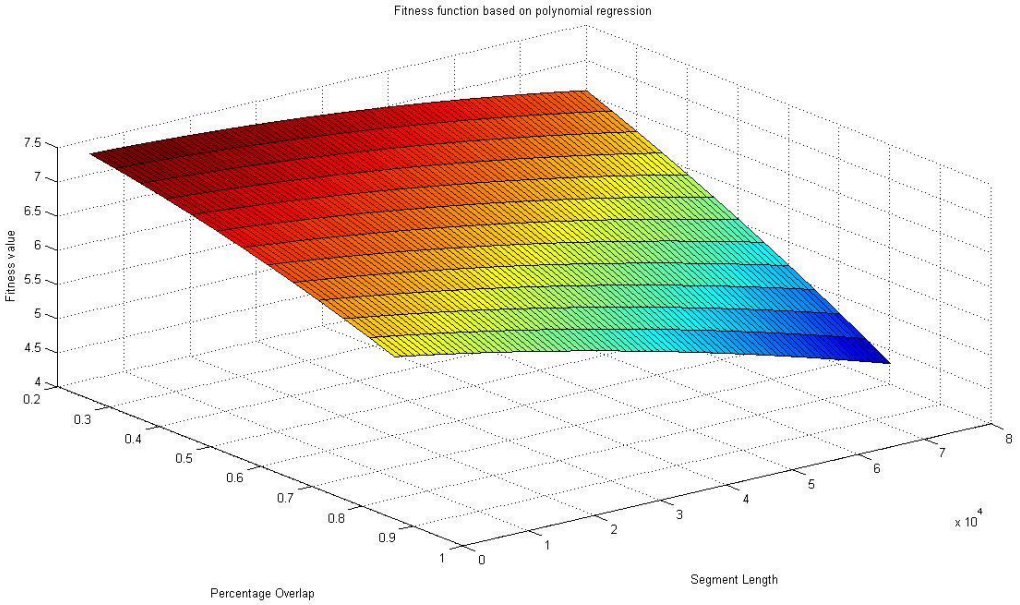

Figure 6.4 Optimization landscape of Polynomial regression based model

6.3.3 Results

The plot of the input signals, the mixed signals and the recovered signals for few samples of the Speech2D mixture is shown in Figure 6.5.

These values are applied for two other mixtures of voice samples and it provides good results proving that the optimized values hold generally true. Their performance indices are provided in Table 6.4. The plots of the source, mixed and recovered signals for the signals Speech2D and Speech_Noisy2D are shown in Figure 6.6 and Figure 6.7 respectively.

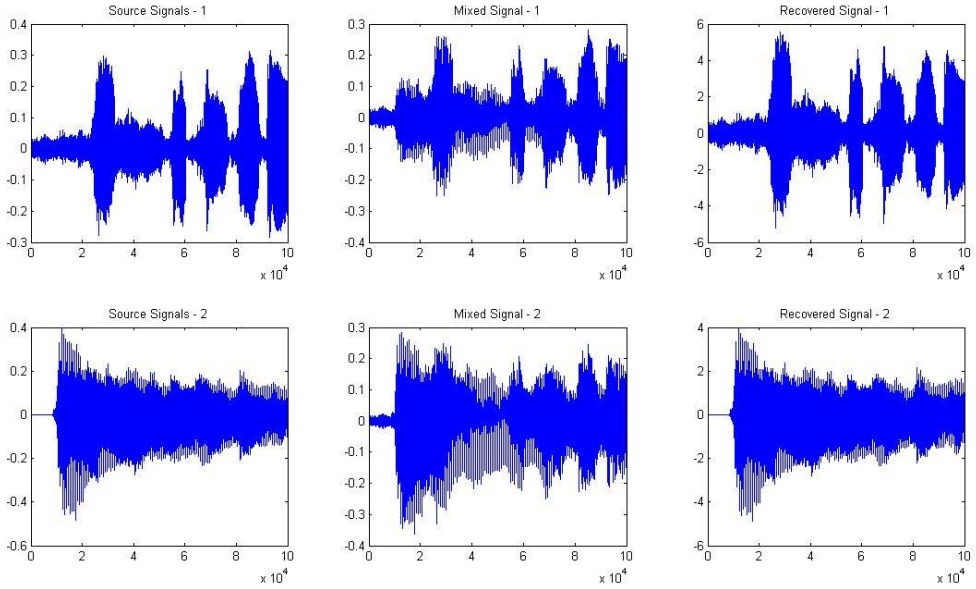

Figure 6.5 Polynomial regression based optimized parameters for Speech_Music2D

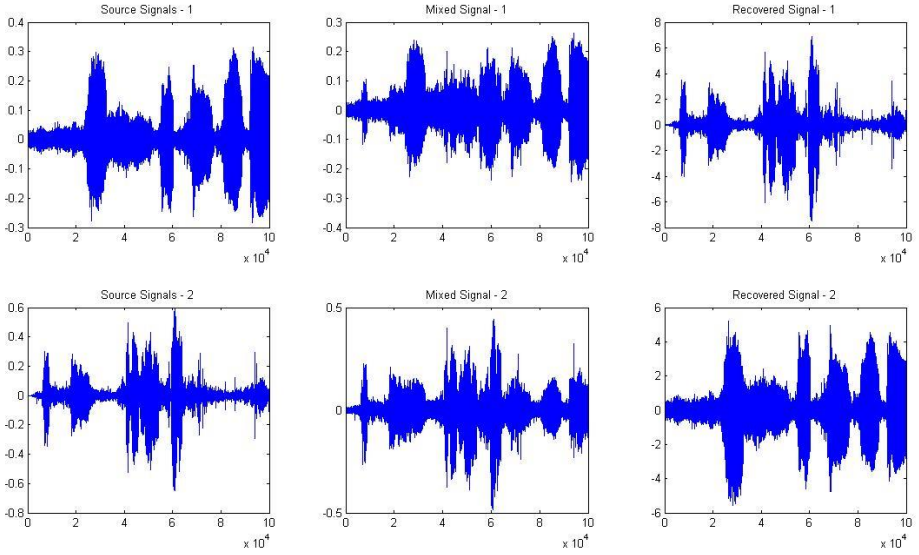

Figure 6.6 Polynomial regression based optimized parameters for Speech2D

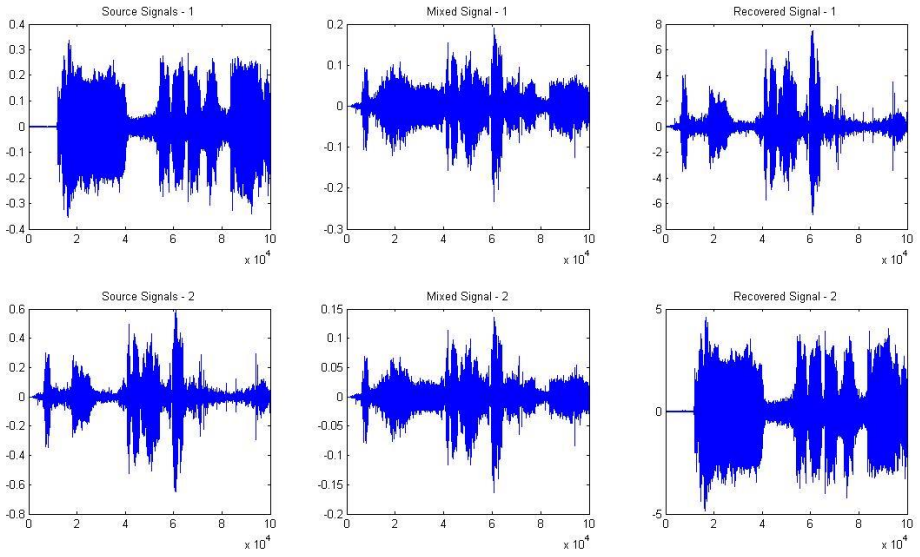

Figure 6.7 Polynomial regression based optimized parameters for Speech_Noisy2D

6.4 NEURAL NETWORKS

6.4.1 Neural networks model

A neural network model for the algorithm is constructed before proceeding for the optimization of the parameters. The training set used here is the same training set used in Polynomial regression. Normalization of input and output data are performed to ensure convergence and the settings are saved so that these settings can be applied to subsequent validation, testing and optimization runs.

The choice of the learning algorithm would be the next step. Since the neural network is not trained online, the time required to train the neural network is not a huge concern. The choice of the algorithm is influenced by the factors such as minimization of Mean Square Error (MSE), better convergence and avoidance of under-fitting and over-fitting. In this thesis, Levenberg-Marquardt (LM) algorithm is used to train a back-propagation neural network. LM algorithm is fast and has better convergence than gradient descent and other conjugate gradient methods. Also, it is robust than Gauss-Newton algorithm

but it tends to be slower than steepest descent algorithm. LM algorithm is an ideal learning algorithm for small and medium sized neural networks: when there are no more than few hundred weights [38]. Since, online BSS process is a small neural network; the computation of the Hessian matrix required for the LM algorithm is quite efficient [40].

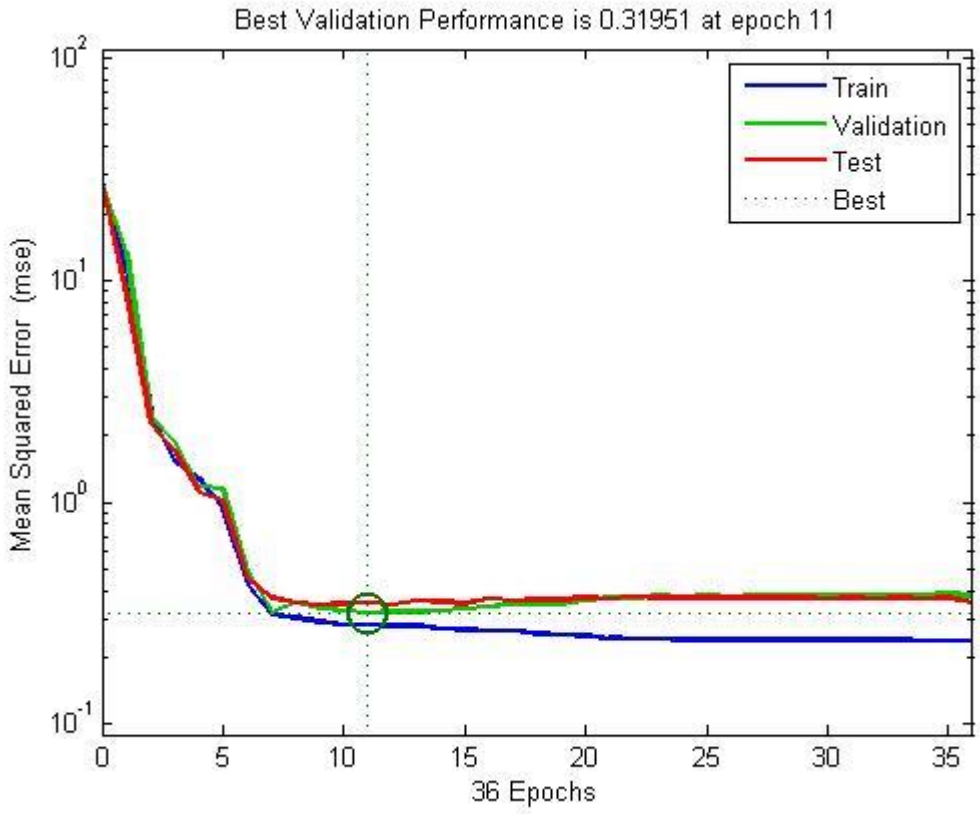

Figure 6.8 Convergence of Neural network trained with LM algorithm

6.4.2 Optimization of neural networks based model

The optimization landscape showing the fitness value plotted against Segment Length and Percentage Overlap is shown in Figure 6.9. As it can be observed from Figure 6.9, the optimization landscape has lot of local minima. To avoid getting stuck in local optima, we need to use a global optimization technique. In this thesis, we use GA based optimization technique to optimize the parameters: Segment Length and Percentage Overlap.

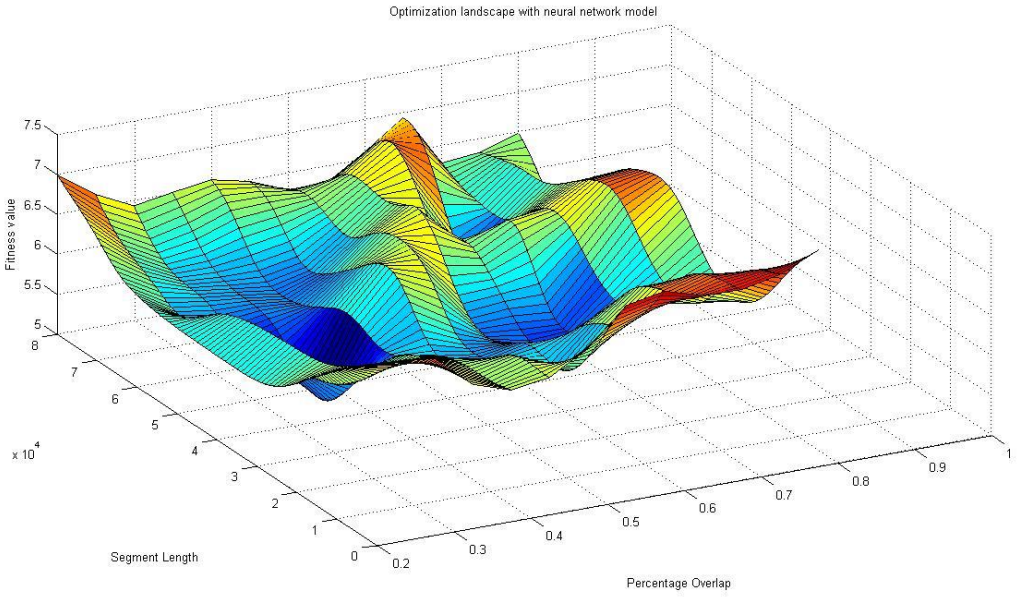

Figure 6.9 Optimization landscape for neural networks based model of online BSS

The optimization is performed with different parameter settings in GA algorithm. For each setting, GA is performed 10 times and the best performance among all the runs is chosen. These values are then compared with the optimized values of other parameter settings in GA and the best value is chosen. The best optimization parameters from Table 6.2 are:

$$Segment\ Length = 43800\ samples$$

$$Percentage\ Overlap = \ 34.18\ \% \tag{6.9}$$

Table 6.2 GA based optimization of neural networks based model for various parameters of the algorithm

Segment Length	% overlap	Theoretical				Simulation			
		SAR (db)	SIR (db)	Time (secs)	Fitness value	SAR (db)	SIR (db)	Time (secs)	Fitness value
43789	0.34	1.3712	49.7588	1.1197	5.1236	-1.8309	42.4619	1.0250	6.2449
43789	0.34	1.3712	49.7588	1.1197	5.1236	-1.8309	42.4619	1.0212	6.2373
43789	0.34	1.3712	49.7588	1.1197	5.1236	-1.8309	42.4619	1.0213	6.2375
43789	0.34	1.3712	49.7588	1.1197	5.1236	-1.8309	42.4619	1.0213	6.2374
43789	0.34	1.3713	49.7589	1.1197	5.1236	-1.8309	42.4619	1.0259	6.2468
43789	0.34	1.3712	49.7587	1.1197	5.1236	-1.8309	42.4619	1.0277	6.2502
43789	0.34	1.3713	49.7588	1.1197	5.1236	-1.8309	42.4619	1.0249	6.2448
43789	0.34	1.3711	49.7584	1.1196	5.1236	-1.8309	42.4619	1.0267	6.2483
43788	0.34	1.3711	49.7584	1.1196	5.1236	-1.8309	42.4619	1.0254	6.2458
43789	0.34	1.3712	49.7588	1.1197	5.1236	-1.8309	42.4619	1.0249	6.2447
43785	0.34	1.3706	49.7565	1.1195	5.1236	-1.8309	42.4619	1.0249	6.2446
43785	0.34	1.3706	49.7564	1.1195	5.1236	-1.8309	42.4619	1.0249	6.2447
43783	0.34	1.3704	49.7556	1.1195	5.1236	-1.8309	42.4619	1.0250	6.2449
43797	0.34	1.3722	49.7624	1.1199	5.1236	-1.8309	42.4619	1.0252	6.2453
43805	0.34	1.3733	49.7663	1.1201	5.1236	-1.8309	42.4619	1.0254	6.2457
43814	0.34	1.3746	49.7711	1.1204	5.1236	-1.8309	42.4619	1.0257	6.2463
43763	0.34	1.3678	49.7459	1.1189	5.1236	-1.8309	42.4619	1.0289	6.2527
43757	0.34	1.3669	49.7427	1.1187	5.1236	-1.8309	42.4619	1.0242	6.2433
43755	0.34	1.3667	49.7416	1.1186	5.1236	-1.8309	42.4619	1.0289	6.2528
43857	0.34	1.3802	49.7919	1.1217	5.1237	-1.2617	38.7569	1.0262	6.1811
43865	0.34	1.3812	49.7951	1.1220	5.1237	-1.2617	38.7569	1.0280	6.1847
43683	0.34	1.3569	49.7054	1.1164	5.1237	-1.3241	41.1397	1.0227	6.1155
43639	0.34	1.3510	49.6826	1.1151	5.1238	-0.7708	37.1589	1.0189	6.0560
43995	0.34	1.3981	49.8563	1.1259	5.1239	0.3533	42.1947	1.0297	5.5351
43583	0.34	1.3432	49.6532	1.1134	5.1239	-0.7708	37.1589	1.0209	6.0602
43582	0.34	1.3431	49.6517	1.1133	5.1239	-0.7708	37.1589	1.0218	6.0620
44030	0.34	1.4025	49.8719	1.1270	5.1240	0.3533	42.1947	1.0290	5.5337
43492	0.34	1.3304	49.6035	1.1106	5.1242	-0.4938	42.4804	1.0187	5.7859
43482	0.34	1.3291	49.5977	1.1103	5.1243	-0.4938	42.4804	1.0272	5.8029
44194	0.34	1.4230	49.9427	1.1319	5.1248	-0.1727	46.7454	1.0324	5.5642
44232	0.34	1.4277	49.9585	1.1331	5.1250	-0.1727	46.7454	1.0354	5.5702
44303	0.34	1.4363	49.9872	1.1352	5.1255	-1.3933	39.1557	1.0379	6.2351
43115	0.34	1.2755	49.3817	1.0990	5.1268	1.3224	34.7637	1.0098	5.4201
44540	0.34	1.4641	50.0759	1.1424	5.1275	-1.2819	38.4235	1.0420	6.2305
44602	0.34	1.4711	50.0979	1.1443	5.1282	-1.3309	41.2120	1.0409	6.1518
45039	0.34	1.5179	50.2298	1.1573	5.1344	-0.8032	35.8305	1.0502	6.1738

With these parameters, we get the performance indices shown in Table 6.4. These values are obtained with the GA parameters provided in Table A 2.2.

6.4.3 Results

Since the SAR and SIR are quite high with the time taken for processing approximately 1 sec, the optimization has provided with excellent set of parameters. The plot of the input signals, the mixed signals and the recovered signals for few samples of the Speech2D mixture is shown in Figure 6.10.

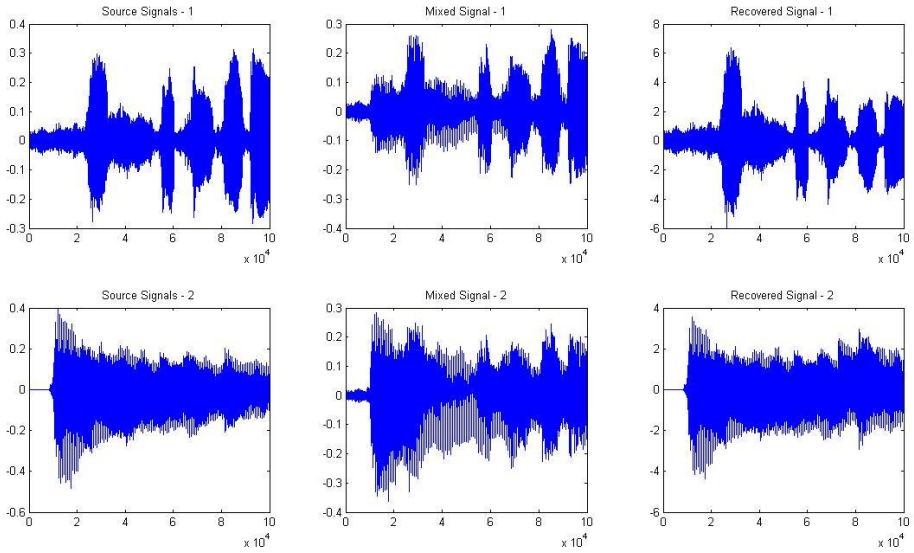

Figure 6.10 Neural networks based optimized parameters for Speech_Music2D

These values are applied for the other two mixtures of voice samples and it provides good results proving that the optimized values hold generally true. Their performance indices are provided in Table 6.4. The plots of the source, mixed and recovered signals for the signals Speech2D and Speech_Noisy2D are shown in Figure and Figure respectively.

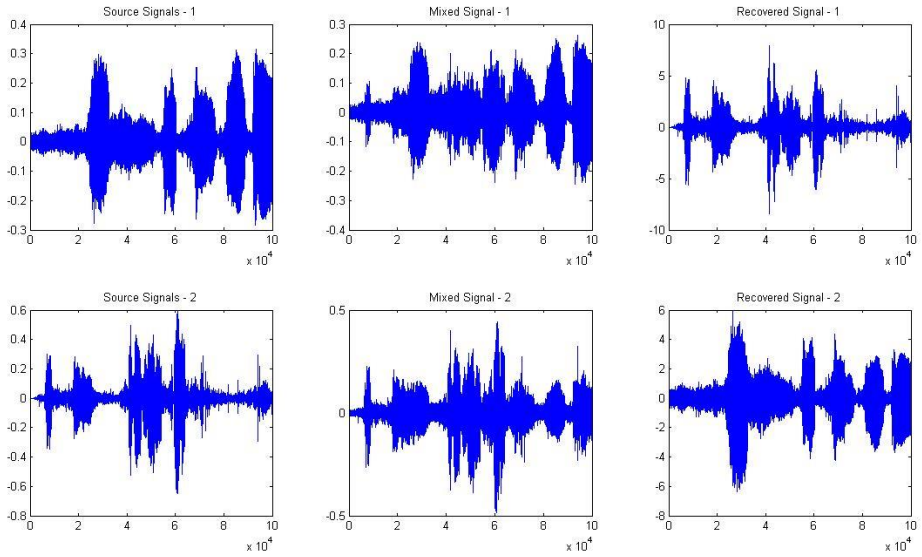

Figure 6.11 Neural networks based optimized parameters for Speech2D

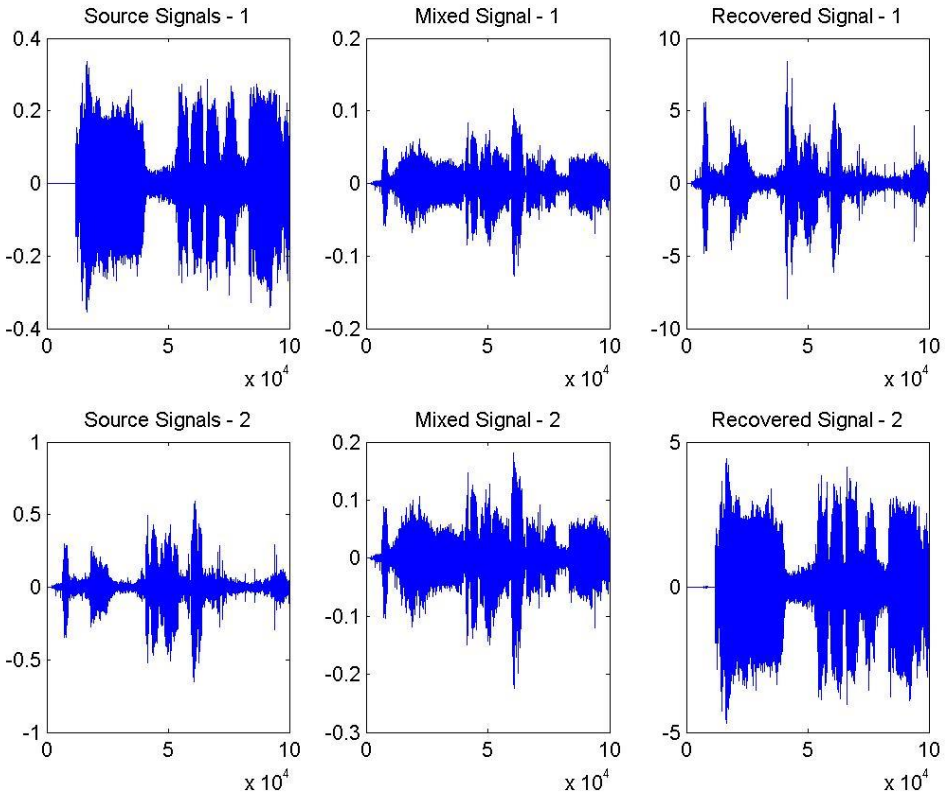

Figure 6.12 Neural networks based optimized parameters for Speech_Noisy2D

6.5 ADAPTIVE NEURO FUZZY INFERENCE SYSTEM

6.5.1 ANFIS Model

The ANFIS system is trained using the training data used in Polynomial regression. Figure 6.13 shows the SAR, SIR and Time of the actual system and the corresponding ANFIS trained model. It can be observed from the SAR and SIR plot of the ANFIS model that the ANFIS model extracts the general tendencies of the data and omits the local disturbances. It can also be observed that there is no 'overfitting' of the data.

The fitness function is then formed based on the formula defined in Chapter. 6.2. Figure shows the comparison of fitness function of the ANFIS model with the fitness function of the actual algorithm's data. It can be observed that ANFIS models the algorithm into a smooth fitness function so that the global minima can be found out easily.

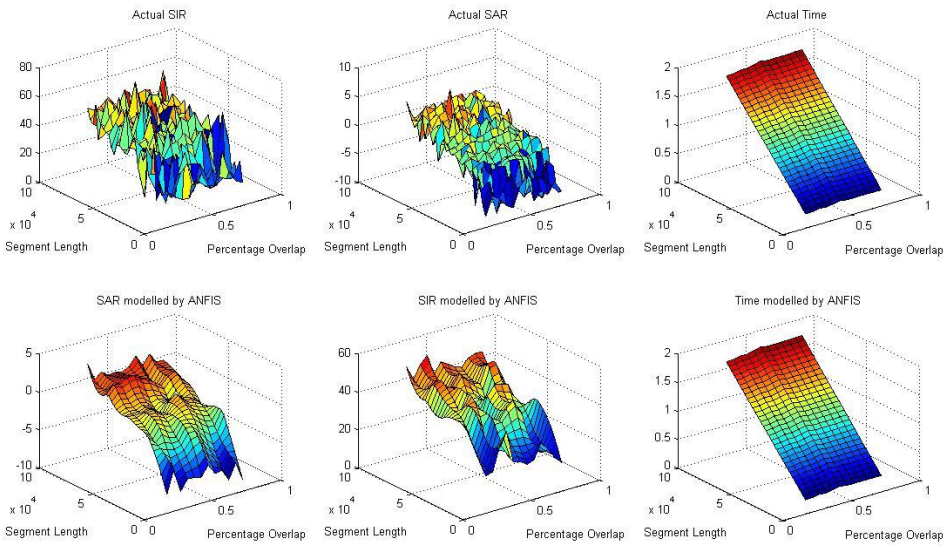

Figure 6.13 SAR, SIR and Time of the online BSS system and the corresponding ANFIS model

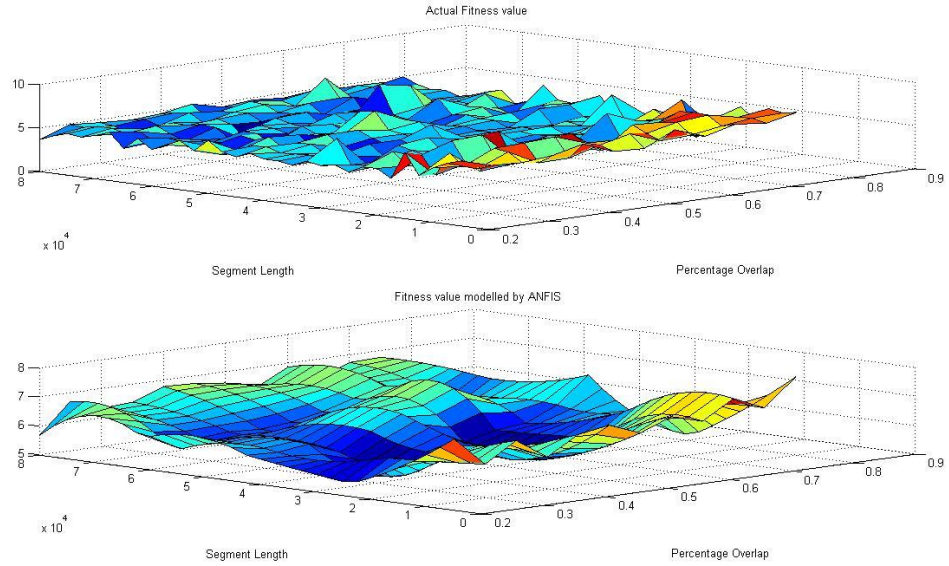

Figure 6.14 Comparison of fitness function of online BSS system and ANFIS model

6.5.2 Optimization of ANFIS based model

As it can be observed from Figure 6.14, the optimization landscape has lot of local minima. To avoid getting stuck in local optima, we need to use a global optimization technique.

The optimization is performed with the help Genetic Algorithm with different parameter settings. For each setting, GA is performed 10 times best performance among all the runs is chosen. These values are then compared with the optimized values of other parameter settings in GA and the best value is chosen.

The best optimized parameters obtained from Table 6.3 are:

$$Segment\ Length = \ 28200\ samples$$

$$Percentage\ Overlap = 31.8\ \% \qquad (6.10)$$

Table 6.3 GA based optimization of ANFIS based model for various parameters of the algorithm

Segment Length	% over lap	Theoretical				Simulation			
		SAR (db)	SIR (db)	Time (secs)	Fitness value	SAR (db)	SIR (db)	Time (secs)	Fitness value
28168	0.32	0.1739	48.9348	0.7433	4.7975	-0.0205	39.1164	0.6640	5.0310
28169	0.32	0.1740	48.9351	0.7433	4.7975	-0.0205	39.1164	0.6665	5.0360
28168	0.32	0.1739	48.9348	0.7433	4.7975	-0.0205	39.1164	0.6666	5.0361
28167	0.32	0.1738	48.9345	0.7433	4.7975	-0.0205	39.1164	0.6665	5.0360
28171	0.32	0.1743	48.9359	0.7434	4.7975	-0.0205	39.1164	0.6628	5.0286
28160	0.32	0.1731	48.9322	0.7431	4.7975	-0.0205	39.1164	0.6705	5.0439
28159	0.32	0.1730	48.9320	0.7431	4.7975	-0.0205	39.1164	0.6627	5.0283
28157	0.32	0.1728	48.9314	0.7431	4.7975	-0.0205	39.1164	0.6629	5.0287
28156	0.32	0.1727	48.9310	0.7430	4.7975	-0.0205	39.1164	0.6630	5.0291
28153	0.32	0.1724	48.9300	0.7430	4.7975	-0.0205	39.1164	0.6624	5.0278
28151	0.32	0.1722	48.9293	0.7429	4.7975	-0.0205	39.1164	0.6627	5.0283
28149	0.32	0.1719	48.9286	0.7429	4.7975	0.5939	32.9505	0.6626	5.0289
28141	0.32	0.1710	48.9256	0.7427	4.7975	0.5939	32.9505	0.6624	5.0284
28135	0.32	0.1704	48.9237	0.7425	4.7975	0.5939	32.9505	0.6731	5.0500
28134	0.32	0.1703	48.9235	0.7425	4.7975	0.5939	32.9505	0.6667	5.0371
28093	0.32	0.1657	48.9096	0.7415	4.7975	0.5939	32.9505	0.6620	5.0277
28265	0.32	0.1843	48.9667	0.7456	4.7975	-0.2945	50.7214	0.6649	4.7373
28307	0.32	0.1888	48.9802	0.7466	4.7976	-0.2945	50.7214	0.6697	4.7468
28325	0.32	0.1906	48.9857	0.7470	4.7976	-0.2945	50.7214	0.6668	4.7411
28388	0.32	0.1972	49.0048	0.7485	4.7978	-0.0313	30.8584	0.6682	5.3183
28447	0.32	0.2033	49.0223	0.7499	4.7980	-0.0313	30.8584	0.6695	5.3208
28467	0.32	0.2052	49.0289	0.7504	4.7980	0.3520	39.6462	0.6722	4.9056
27838	0.32	0.1371	48.8143	0.7355	4.7981	-1.7650	26.8458	0.6608	6.0151
28531	0.32	0.2118	49.0457	0.7519	4.7983	0.3520	39.6462	0.6743	4.9098
28627	0.32	0.2214	49.0714	0.7542	4.7988	0.5165	41.2030	0.6765	4.8074
27668	0.32	0.1173	48.7452	0.7315	4.7990	-0.7085	51.7164	0.6517	4.8157
27516	0.32	0.0991	48.6788	0.7279	4.8002	-0.7982	42.4912	0.6511	5.1519
28866	0.32	0.2443	49.1280	0.7598	4.8006	-0.5015	33.0301	0.6815	5.4291
29103	0.32	0.2659	49.1746	0.7654	4.8030	-1.2523	39.1012	0.6843	5.4826
27202	0.32	0.0598	48.5291	0.7205	4.8034	-0.6325	32.5413	0.6410	5.4082
29195	0.32	0.2740	49.1901	0.7676	4.8042	-3.6723	14.8909	0.6986	7.1250
29220	0.32	0.2761	49.1940	0.7682	4.8045	-3.6723	14.8909	0.6871	7.1019
27054	0.32	0.0406	48.4530	0.7170	4.8053	-1.8607	32.7596	0.6414	5.8110
29299	0.32	0.2828	49.2057	0.7700	4.8056	-0.4922	38.5433	0.6888	5.2569
30243	0.32	0.3529	49.2647	0.7923	4.8249	-0.6095	45.1336	0.7102	5.1190
24721	0.32	-0.3230	46.7500	0.6619	4.8731	-2.3877	31.6347	0.5872	5.9159

With these parameters, we get the performance indices shown in Table 6.4. These values are obtained with the GA parameters provided in Table A 2.3.

6.5.3 Results

We are able to achieve better performance at a time delay approximately 0.6 seconds as observed in Table 6.4. This performance is better than the optimization of the neural networks based online BSS model. The plot of the source signals, mixed signals and recovered signals is shown in Figure 6.15.

The performance indices of Speech2D and Speech_Noisy2D signal mixtures are shown in Table 6.4. These indices indicate that the performance of ANFIS for the other two signals are not as robust as that of neural networks based model. Figure 6.16 and Figure 6.17 show the sources, signal mixtures and the recovered signals of Speech2D and Speech_Noisy2D signal mixtures respectively.

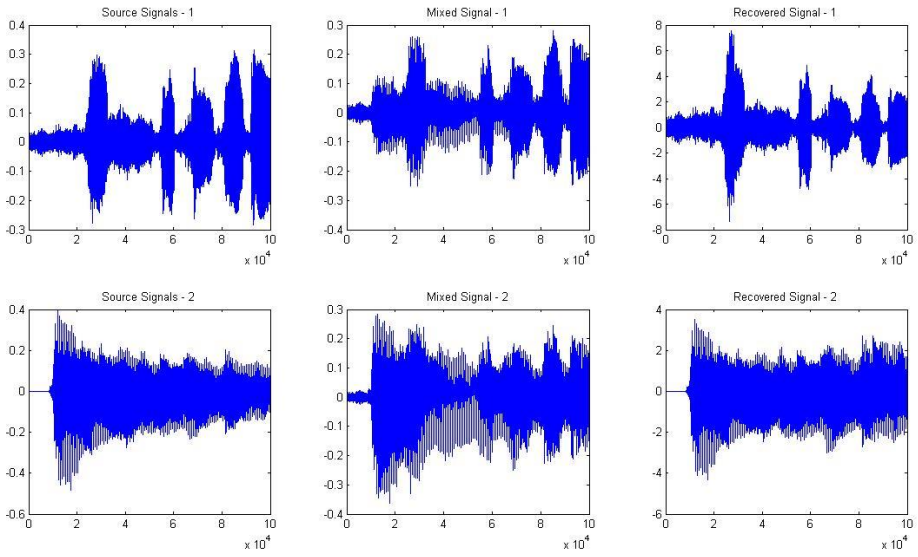

Figure 6.15 ANFIS based optimized parameters for Speech_Music2D

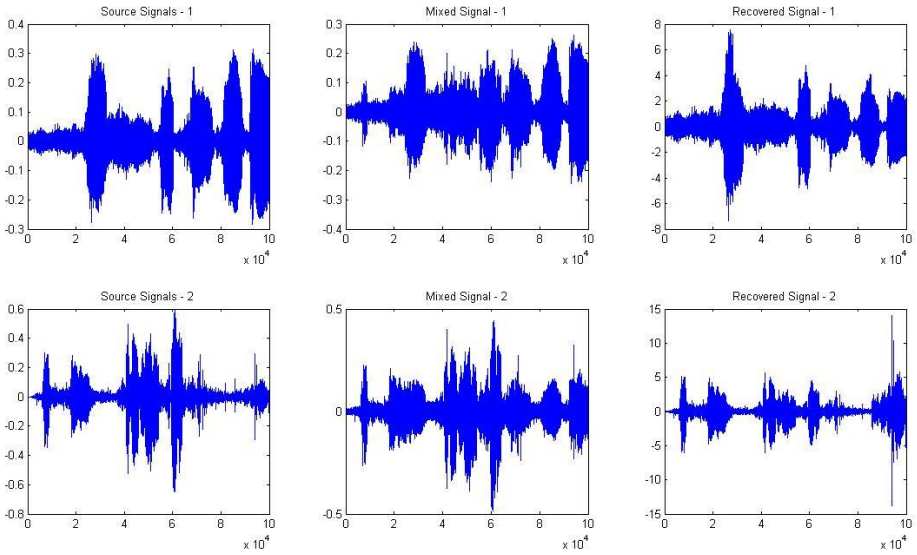

Figure 6.16 ANFIS based optimized parameters for Speech2D

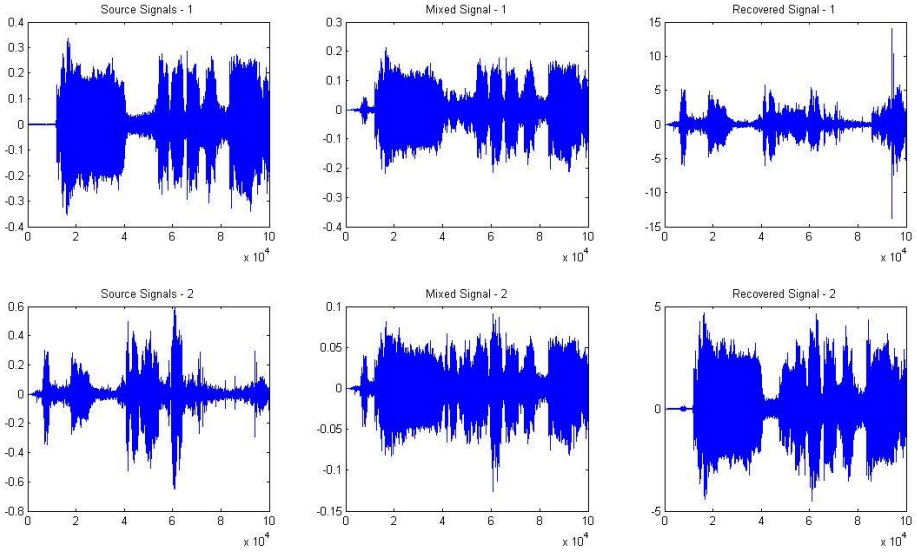

Figure 6.17 ANFIS based optimized parameters for Speech_Noisy2D

6.6 COMPARISON OF OPTIMIZATION OF ONLINE BSS

Table 6.4 Comparison of optimization based on Polynomial Regression, ANN and ANFIS based model

Parameter	Polynomial Regression			Neural Networks			ANFIS		
	Speech_Music2D	**Speech2D**	**Speech_Noisy2D**	**Speech_Music2D**	**Speech2D**	**Speech_Noisy2D**	**Speech_Music2D**	**Speech2D**	**Speech_Noisy2D**
Segment Length	80000			43800			28200		
Percentage Overlap	0.8			0.34			0.31		
SAR (db)	2.23	3.46	13.51	-1.83	2.85	-1.09	-0.02	-3.51	1.33
SIR (db)	49.37	25.84	49.99	42.46	29.83	39.18	39.12	8.37	15.20
Time (sec)	1.89	1.89	1.90	1.02	1.02	1.02	0.66	0.66	0.66
Fitness value	6.39	6.77	2.63	6.24	5.10	6.10	5.03	7.22	5.38

Table 6.4 shows the comparison of optimization based on Polynomial regression, neural networks and ANFIS based models' performance indices. Although Polynomial Regression provides a model with very high SAR and SIR, the Time parameter is quite high. In real-time, approximately 2 seconds delay for audio signal processing may not be acceptable. Although ANFIS based model provided the least Time, SAR is very low: the output signal would be highly distorted as can be observed in Figure 6.15, Figure 6.16 and, Figure 6.17. Neural networks can give a model better than other models and provides an optimization result that is consistent with all the three performance parameters.

CHAPTER 7
CONCLUSION AND SCOPE FOR FUTURE WORK

The proposed scheme to order the signals based on their SIRs of the overlapping samples is found to be an effective technique in solving the permutation problem. The optimized parameters obtained by GA are tested with two other signal mixtures: one with less noisy speech with lot of pauses and the other with more noisy speech with fewer pauses between words. The performance indices for these signals are quite high for neural networks based model compared to the polynomial regression and ANFIS based model indicating the robustness of neural networks based model for optimization process.

The technique discussed in this technique can be applied to real-time signal separation problems like ambient noise reduction in mobile phones, pre-processing for voice input systems, voice recognition and, authentication systems.

Real-time testing of the method of overlapping samples for time-domain BSS can be carried out in a DSP or FPGA kit before implementing in real-time embedded systems.

REFERENCES

[1] A. Hyvärinen, "Fast and robust fixed-point algorithms for independent component analysis," *IEEE transactions on neural networks*, pp. 626-634, 1999.

[2] A. Hyvärinen and E. Oja, "A fast-fixed point algorithm for Independent component analysis," *Neural Computation,* vol. 9, pp. 1483-1492, 1997.

[3] A. Hyvärinen and E.Oja, "Independent Component Analysis: Algorithms and Applications," *Neural Networks,* vol. 13, no. 4, pp. 411-430, 2000.

[4] L.Tong, Y. Inouye and R. Liu, "A finite step global convergence algorithm for the cumulant based parameter estimation of multichannel MA processes," in *Acoustics, Speech and Signal Processing*, 1991.

[5] L.Tong, Y. Inouye and R. Liu, "A finite step global convergence algorithm for the parameter estimation of multichannel MA processes," *IEEE transactions on Signal processing,* vol. 40, no. 10, pp. 2547 - 2558, October 1992.

[6] A. Belouchrani and A. Cichocki, "Robust whitening procedure in blind source separation context," *Electronics letters*, vol. 36, no. 24, pp. 2050 - 2053, 2000.

[7] J.-F. Cardosa, A. Belouchrani, K. A-M and E.Moulines, "A blind source separation technique using second order statistics," *IEEE transactions on signal processing,* vol. 45, no. 2, Febraury 1997.

[8] L. Tong, R.-W. Liu, V. C. Soon and Y.-F. Huang, "Indeterminacy and Identifiability of Blind Identification," *IEEE transactions on circuits and systems*, vol. 38, 1991.

[9] L. Tong, V. C. Soon, Y. Huang and R. Liu, "AMUSE: A new blind identification algorithm," *Circuits and Systems*, 1990.

[10] J. Herault and C. Jutten, "Blind separation of sources: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, pp. 1-10, 1991.

[11] R. Gribonval, L. Benaroya, E. Vincent and C. Févotte, "Proposals for performance measurement in source separation," in *4th Int. Symp. on Independent Component Anal. and Blind Signal Separation*, 2003.

[12] E. Vincent, R. Gribonval and C. Févotte, "Performance Measurement in Blind Audio Source Separation," *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, vol. 14, no. 4, 2006.

[13] J. Cardoso and A. Souloumiac, "Blind beamforming for non-Gaussian signals," *IEE Proceedings F (Radar and Signal Processing)*, vol. 140, no. 6, pp. 362-370, 1993.

[14] J. Cardoso, "Source separation using higher order moments," in *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89.*, 1989.

[15] P. Comon, "Independent component analysis, a new concept?," *Signal processing*, pp. 287-314, 1994.

[16] A. Bell and T. Sejnowski, "An information maximisation approach to blind separation and blind deconvolution," *Neural computation*, pp. 1129-1159, 1995.

[17] S. Amari, A. Cichocki and H. Yang, "A New Learning Algorithm for Blind Signal Separation," in *Advances in Neural Information processing 8 (Proc. NIPS '95)*, Cambridge, MA, 1996.

[18] V. C. Soon, L. Tong, Y. Huang and R. Liu, "An extended fourth order blind identification algorithm in spatially correlated noise," in *Acoustics, Speech, and*

Signal Processing. ICASSP-90., 1990.

[19] Y. Inouye and T. Matsui, "Cumulant based parameter estimation of linear systems of nonminimum phase," in *Workshop on Higher-Order Spectral Analysis, 1989.*, 1989.

[20] P. Comon, "Separation of stochastic processes," in *Workshop on Higher-Order Spectral Analysis, 1989*, 1989.

[21] B. Pearlmutter and L. Parra, "A context-sensitive generalization of ICA," in *International conference on neural information processing*, 1996.

[22] Y.Baram and Z.Roth, "Density shaping by neural networks with application to classification, estimation and forecasting.," Technion, Haifa, 1994.

[23] M. Girolami and C. Fyfe, "algorithms, Negentropy and kurtosis as projection pursuit indices provide generalised ICA," in *A. C, Back A (eds.), NIPS-96 Blind Signal Separation Workshop*, 1996.

[24] R. Linsker, "Local synaptic learning rules suffice to maximise mutual information in a linear network," *Neural Computation*, pp. 691-702, 1992.

[25] Y. Li, D. Powers and J. Peach, "Comparison of Blind Source Separation Algorithms," *Advances in neural networks and applications*, pp. 18-21, 2000.

[26] A. Cichocki, S. Amari, K. Siwek, T. Tanaka, A. H. Phan and e. al, "ICALAB toolbox," [Online]. Available: http://www.bsp.brain.riken.jp/ICALAB/. [Accessed 8 June 2014].

[27] A. Cichocki and S. Amari, Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications, Wiley, 2003.

[28] C. Cherry, "Some experiments on recognition of speech with one and two ears," *The journal of the acoustic society of America*, vol. 25, no. 5, pp. 975 - 979, September 1953.

[29] B. Arons, "A Review of The Cocktail Party Effect," *Journal of the American Voice I/O Society,* vol. 12, no. 7, pp. 35-50, 1992.

[30] F. &. S. H. G. Ehlers, "Blind separation of convolutive mixtures and an application in automatic speech recognition in a noisy environment," *IEEE Transactions on Signal processing*, vol. 45(10), pp. 2608-2612, 1997.

[31] E. Chaumette, P. Comon and D. Muller, "Application of ICA to airport surveillance," in *IEEE Signal Processing Workshop on Higher-Order Statistics*, 1993.

[32] M. Habl, C. Bauer, C. Ziegaus, E. W. Lang and F. Schulmeyer, "Analyzing brain tumor related EEG signals with ICA algorithms," *Artificial Neural Networks in Medicine and Biology,* pp. 131-136, 2000.

[33] A. Cichocki and W. Kasprzak, "Multi-layer neural networks with a local adaptive learning rule for blind separation of source signals," in *Proc. Int. Symp. Nonlinear Theory App.*, Las Vegas, 1995.

[34] A. Ypma and P. Pajunen, "Rotating machine vibration analysis with second-order independent component analysis," in *Proceedings of the First International Workshop on Independent Component Analysis and Signal Separation, ICA*, 1999.

[35] N. Thirion, J. Mars and J. L. Boelle, "Separation of seismic signals: A new concept based on a blind algorithm," in *EUSIPCO'96*, Triest, Italy, 1996.

[36] S. Boyd and L. Vandenberghe, Convex optimization, Cambridge University Press, 2009.

[37] D. M.Hawkins, "The problem of overfitting," *Journal of Chemical Information and Computational Sciences,* vol. 44, pp. 1-12, 2004.

[38] M. Hagan and M. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE transactions on Neural Networks,* pp. 989-993, 1994.

[39] J.-S. Jang, "ANFIS: Adaptive-network based fuzzy inference system," *IEEE transactions on Systems, Man and Cybernetics,* pp. 665-685, 1993.

[40] H. Yu and B. M. Wilamowski, "Levenberg-marquardt training," in *Industrial Electronics Handbook*, 2 ed., 2011, pp. 12-1 to 12-15.

[41] C. Fevotte, R. Gribonval and E. Vincent, "BSS_EVAL Toolbox user guide revision 2.0," IRISA, 2011.

APPENDIX I

This section explains the signals used in testing of the BSS algorithms.

1) ACsin4D

This signal mixture is formed by linear addition of 4 sine waves as shown in Equation A 1.1. The length of this signal is 1001 samples.

$$s_n = \sin\big((2n-1)\omega K\big), n = 1,2,3,4 \qquad \text{(A 1.1)}$$

The sources and the signal mixtures are shown in Figure

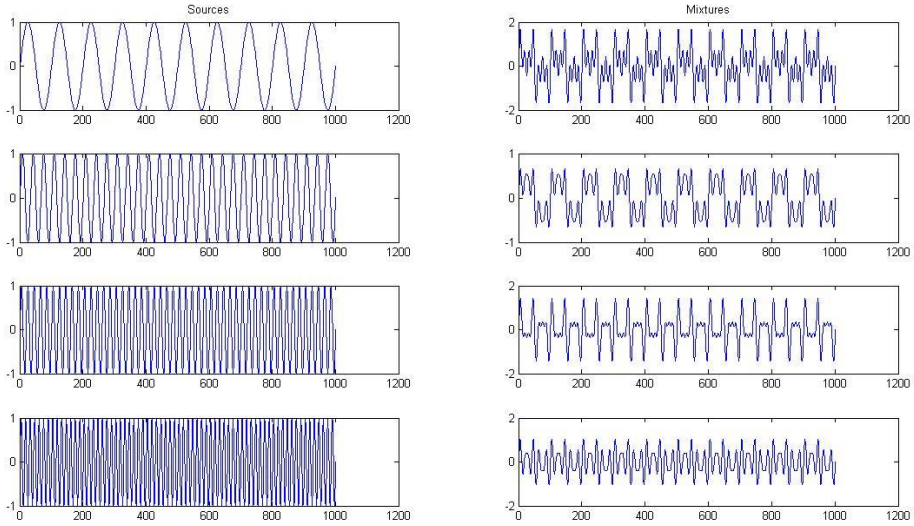

Figure A 1.1 Sources and Signal mixture of signal ACsin4D

2) Speech4D

Speech4D is a signal mixture with 4 speech and music sources[1]. The signal length is 5000 samples sampled at 44100 samples per sec. The sources and signal mixtures are shown in Figure A 1.2.

3) Speech_Music2D

This signal is formed by mixture of a speech signal[2] and a music signal. The length of each signal is 2200000 samples sampled at 44100 samples per sec. The source and signal waveforms are shown in Figure A 1.3.

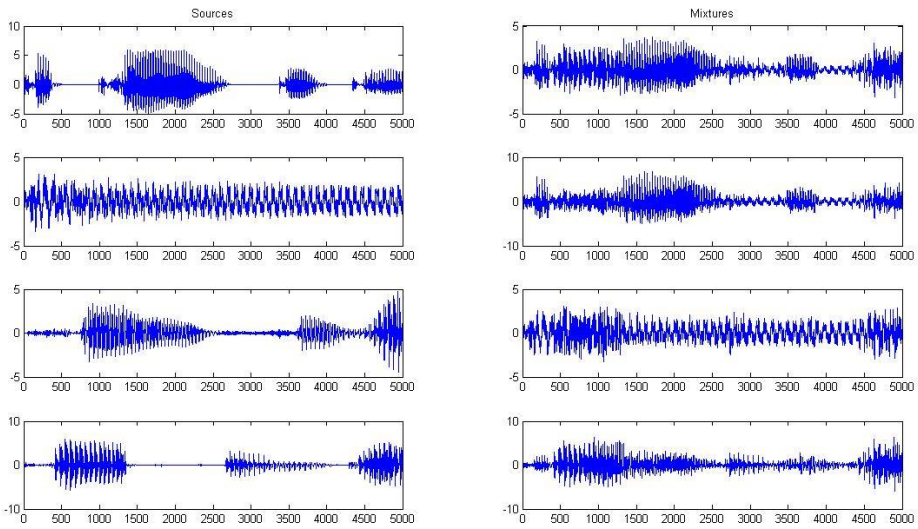

Figure A 1.2 Sources and Signal mixture of signal Speech4D

[1] Courtesy: http://www.bsp.brain.riken.go.jp/ICALAB/ICALABSignalProc/benchmarks
[2] Courtesy: https://ia600402.us.archive.org/29/items/MLKDream/MLKDream_64kb_mp3.zip

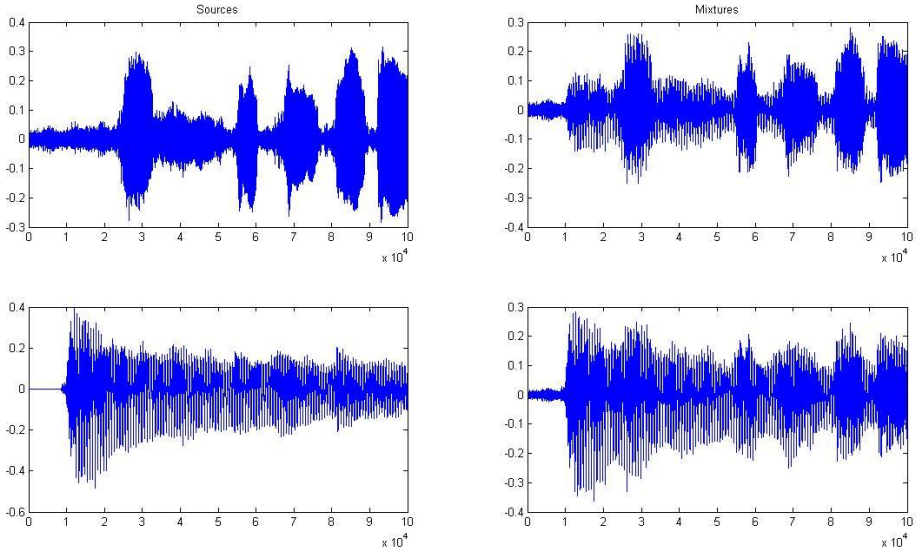

Figure A 1.3 Sources and Signal mixture of signal Speech_Music2D

4) Speech2D

This signal is formed by mixture of two speech signals[3]. The length of each signal is 2200000 samples sampled at 44100 samples per sec. The source and signal waveforms are shown in Figure A 1.4.

5) Speech_Noisy2D

This signal mixture is formed by mixture of two speech signals, one of which is very noisy[4]. Like the previous signals, these signals are also 2200000 samples long and sampled at 44100 samples per sec. The source and the signal mixtures are shown in Figure A 1.5.

[3] Courtesy: https://ia600402.us.archive.org/9/items/MahatmaGandhiSpeech/GandhiSpeech.mp3
[4] Courtesy: https://ia600200.us.archive.org/18/items/Greatest_Speeches_of_the_20th_Century/TheMoonLanding.mp3

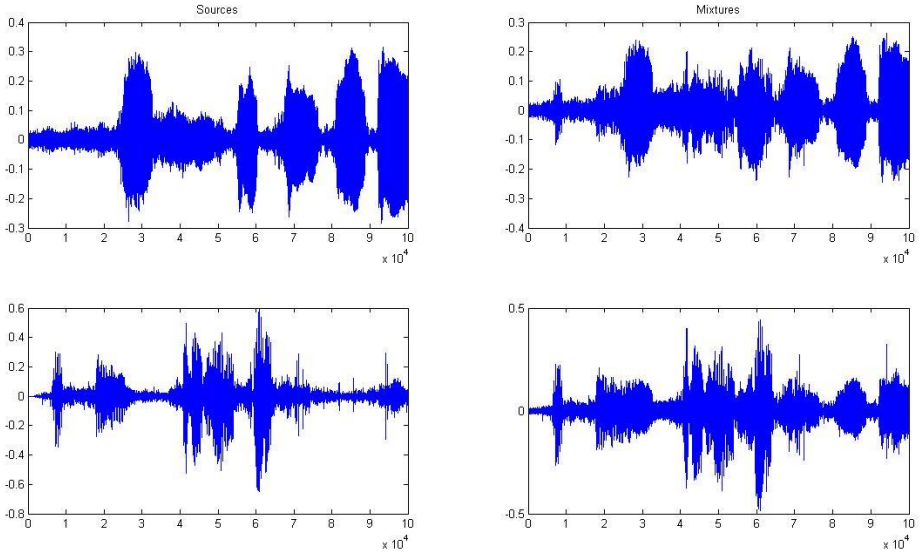

Figure A 1.4 Sources and Signal mixture of signal Speech2D

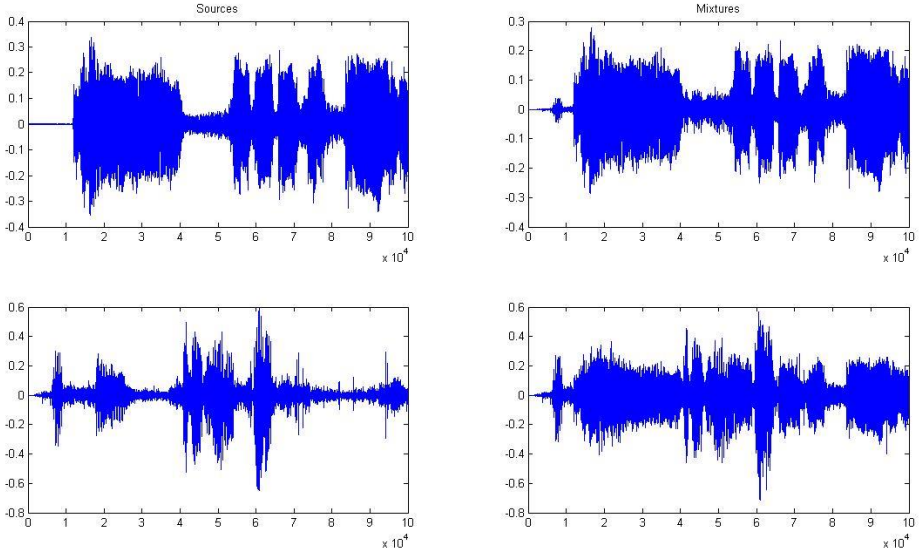

Figure A 1.5 Sources and Signal mixture of signal Speech_Noisy2D

APPENDIX II

Table A 2.1 Parameters of GA used to optimize Polynomial regression based Online BSS model

Parameter	**Value**
SelectionFcn	@selectionroulette (Roulette)
Population Size	25
EliteCount	4
MutationFcn	@mutationadaptfeasible (Adaptive feasible)
CrossoverFcn	@crossoverheuristic (Heuristic)
Generations	100
HybridFcn	@fmincon
CrossoverFraction	0.5

Table A 2.2 Parameters of GA used to optimize neural networks based Online BSS model

Parameter	**Value**
SelectionFcn	@selectionstochunif (Stochastic Uniform)
Population Size	25
EliteCount	4
MutationFcn	@mutationadaptfeasible (Adaptive feasible)
CrossoverFcn	@crossoverheuristic (Heuristic)
Generations	100
HybridFcn	@fmincon
CrossoverFraction	0.5

Table A 2.3 Parameters of GA used to optimize ANFIS based Online BSS model

Parameter	**Value**
SelectionFcn	@selectionroulette (Roulette)
Population Size	25
EliteCount	4
MutationFcn	@mutationadaptfeasible (Adaptive feasible)
CrossoverFcn	@crossoverheuristic (Heuristic)
Generations	100
HybridFcn	@fmincon
CrossoverFraction	0.8