Real-time Noise Cancellation Using ICA-PSO-PE

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Remziye İrem Bor

June 2012

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Yusuf Ziya İder(Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Orhan Arıkan

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. İbrahim Körpeoğlu

Approved for the Graduate School of Engineering and Sciences:

Prof. Dr. Levent Onural
Director of Graduate School of Engineering and Sciences

ABSTRACT

Real-time Noise Cancellation Using ICA-PSO-PE

Remziye İrem Bor
M.S. in Electrical and Electronics Engineering
Supervisor: Prof. Dr. Yusuf Ziya İder
June 2012

A real-time implementable noise cancellation algorithm is developed. Speech and noise sources are not known but only their mixtures are observed. A mobile radio system is modelled with instantaneous mixture model as the environment where noise cancellation is performed. A combination of independent component analysis (ICA) and particle swarm optimization (PSO) algorithms is used to separate speech and noise signals. However, ICA has an ambiguity such that it is not possible to know which one of the separated signals is speech or noise. To overcome this ambiguity problem, a pitch extraction (PE) algorithm is developed and combined with ICA-PSO. The ICA-PSO-PE algorithm is implemented in MATLAB. Signals are synthetically mixed with a mixing matrix and provided in frames of 40 ms to simulate the real-time behaviour. Pre-processing steps except centering is bypassed to fasten the process and objective functions of ICA are slightly modified to reduce computational cost. Rule of convergence for PSO is changed in a way to rely on global best solution highly and a very small swarm is used. In order to increase accuracy of separation, a learning period is introduced. Experiments show that ICA-PSO-PE is a real-time implementable and robust noise cancellation algorithm in the sense that it is computationally efficient, accurately extracts speech signal from its mixtures, even with very

low SNR levels. The proposed noise cancellation algorithm is compared with FastICA by Hyvärinen *et al* and the subtraction method. Simulations show that our algorithm outperforms FastICA in the sense of real-time implementability and outperforms subtraction method in the sense of robustness.

Keywords: Noise cancellation, ICA, PSO, pitch extraction

ÖZET

ICA-PSO-PE İLE GERÇEK ZAMANLI GÜRÜLTÜ GİDERİMİ

Remziye İrem Bor
Elektrik ve Elektronik Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Prof. Dr. Yusuf Ziya İder
Haziran 2012

Gerçek-zamanlı çalışabilecek bir gürültü giderimi algoritması geliştirilmiştir. Konuşma ve gürültü kaynakları bilinmemekte, ancak bunların karışımları gözlenebilmektedir. Bir telsiz sistemi, gürültü giderimi yapılacak ortam olarak anlık karışım modeli ile modellenmiştir. Bağımsız bileşen analizi (BBA) ve parçacık sürü optimizasyonu (PSO) algoritmaları konuşma ve ses sinyallerini ayrıştırmak için birlikte kullanılmıştır. Buna ek olarak, BBA ile ayrıştırılmış sinyallerin hangisinin konuşma veya hangisinin gürültü olduğunu bilinmemektedir. Bu belirsizlik sorunu aşmak için bir ses perdesi özütleme (SPÖ) algoritması BBA-PSO ile birleştirilmiştir. BBA-PSO-SPÖ algoritması MATLAB ile gerçeklenmiştir. Ses ve gürültü işaretleri, bir karışım matrisi ile sentetik olarak karıştırılmış ve gerçek zamanlı davranışa benzetim yapmak için 40 ms'lik çerçeveler halinde kullanılmıştır. Merkezleme hariç bütün ön işlemler atlanmış ve BBA'nın bazı amaç fonksiyonları hesaplama maliyetini azaltmak için basitleştirilmiştir. PSO için yakınsama kuralı, sürüdeki en iyi çözüme güçlüce dayanacak şekilde değiştirilmiş ve çok küçük bir sürü kullanılmıştır. Ayrıştırma doğruluğunu artırmak için bir öğrenme süreci tanıtılmıştır. Deneyler, BBA-PSO-SPÖ algoritmasının düşük işlem maliyeti ile gerçek zamanlı uygulanabilir ve çok düşük sinyal-gürültü oranlarında dahi doğru ayrıştırma sağlaması ile

dayanıklı bir gürültü giderimi yöntemi olduğunu göstermiştir. Önerilen gürültü giderimi algoritması Hyvärinen'in FastICA yöntemi ve çıkarma yöntemi ile karşılaştırılmıştır. Simülasyonlar göstermiştir ki BBA-PSO-SPÖ algoritması gerçek zamanlı uygulanabilirliği bakımından FastICA'dan ve dayanıklılık anlamında çıkarma yönteminden daha iyi performans gösterir.

Anahtar Kelimeler: Gürültü giderimi, bağımsız bileşen analizi, parçacık sürü optimizasyonu, ses perdesi özütleme

ACKNOWLEDGMENTS

I would like to thank my advisor Prof. İder and Prof. Arıkan for their guidance and support. I have learned a lot from them, not only theoretically, but also how to be an engineer in practice. Having the opportunity to observe their approach to any kind of problems is one of the greatest benefits that I gained during my masters studies.

Special thanks to Dr. Erdem Ertan for his contributions on pitch extraction. Beyond simply contributing by his former research, he willingly did his best to improve my vision and make me believe in myself and my studies.

I would also like to thank my family and friends for their support and encouragement. They were always there with their smiling faces to give me hope. They made me feel safe and loved which provided me the strength to carry on.

Finally, I would like to thank Aselsan Inc. for supporting my master studies and DSP group for giving me the chance to experience team work.

Contents

List of Figures

List of Tables

Dedicated to Gözen, Aynur and Kani...

Chapter 1

INTRODUCTION

In mobile communication, especially background noise may be enhanced by voice coding algorithms. Therefore, suppressing the background noise may not be sufficient to provide communication with good quality. Noise cancellation, which is an active research area, may be a solution for this problem.

In this thesis, we assume "noise" is additive and everything except the desired signal is considered as "noise". More specifically, we refer to "background noise" signal which is added to speech signal on a mobile radio. Besides "noise cancellation", there are many names referring to enhancing the quality of observed signal, i.e. making it resemble the source signal as much as possible. De-noising and noise suppression are some of the most frequently used names. The reason that we preferred the word "cancellation" is because our aim is literally "cancelling out" the noise component, instead of trying to suppress it. Before going into the details of how we perform cancellation, it is important to address the system that gains advantage from noise cancellation.

Aim of this thesis is to put noiseless transmission into practice for a mobile radio or any similar system. Generally, noise of signal to be transmitted is suppressed in mobile radio systems. However, voice coding algorithms in mobile radios may enhance formerly suppressed noise. So, beyond suppression, mobile radio communication requires real-time and computationally efficient noise cancellation. That also increases the feeling of quality by providing crystal clear speech. Since there are also other necessary modulations on signal to be transmitted, the time left for noise cancellation algorithm is very short. Another challenge for such a system is the erratic character of noise signal.

Not only statistical properties but also amplitude of noise signal changes with respect to the environment. For instance, assuming that this radio is being used by a fireman, the user of the mobile radio may be travelling in the fire truck, where car noise or may be more noise due to siren is present. Then they arrive at the venue of fire, say a hotel is burning. When they go into the building noise signal is the noise of fire and its statistical properties and amplitude is completely different than the former noise signal, car noise. Another example can be a policeman using the mobile radio during police patrol. He may be travelling in a police car passing by a plaza or stuck in heavy traffic where the noise signal is always changing. So, the noise cancellation algorithm must be adaptive.

In order to simulate such a system, linear instantaneous mixture model is used. In this model, observed signals consist of instantaneous mixtures of source signals, which are noise and speech in this case. The model can be used for n sources and m receivers but we used it only for the case $n = m = 2$ since we have two receivers and two source signals. Then the model becomes

$$x_1(t) = \alpha_{11} s_1(t) + \alpha_{21} s_2(t) \tag{1.1}$$

$$x_2(t) = \alpha_{12} s_1(t) + \alpha_{22} s_2(t) \tag{1.2}$$

where

- $s_1(t)$ and $s_2(t)$ are source signals,
- $x_1(t)$ and $x_2(t)$ are observed signals,
- α_{ij},are mixing coefficients belonging to i^{th} source signal j^{th} receiver.

We need two different observations because we are trying to analyse mixture of two unknown signals by observing their mixtures only. Note that, not only the source signals, but also the mixing coefficients are unknown. Thus, we use *blind source separation* techniques since little or no information on source signals is present.

In real life, observed signals may be obtained from the receivers at the top and bottom of a mobile radio, as shown in Figure 1.1. Generally, such an orientation causes the main microphone (receiver 1) to receive speech signal with a higher amplitude than the sub-microphone (receiver 2). On the other hand, since noise is coming from far-away, it is received almost equally by both microphones.

In such systems, the most common way of background noise cancellation is subtracting the signal obtained at sub-microphone from the one received by main microphone. Though this method is computationally efficient, unless both receivers' amplitude gains must be matched at a certain level to obtain good results. Generally, noise is cancelled by hardware. Software solutions for noise cancellation are not that common. Some companies claim to perform noise cancellation but either their algorithms are patented and they are not willing to provide information or the system where noise cancellation performed is different from ours. So, it is not possible to claim that there is a certain solution for this problem since performance of methods strongly depends on experimental

Figure 1.1: The mobile radio and its receivers

conditions.

One of the works which is old but compares some algorithms to reduce car noise is provided by Liberti, Rappaport and Proakis in 1991 [1]. Their system is different from ours because they use a reference microphone listening to noise and a primary microphone mostly receiving the speech. In addition to moving the reference microphone to several places in the car, they used some additional hardware (like adding a foam to primary microphone). Some methods they investigated, like two-microphone adaptive noise cancellation, does not perform well when noise is in the speech band. On the other hand, methods with high noise reduction levels are computationally inefficient.

A more recent and comprehensive study on speech enhancement using BSS is provided in [2]. In his work BSS is combined with spectral subtraction (SS),

which is a widely used speech enhancement technique. He obtained fairly good results for separating various sources with reverberation and latency. He also developed a real-time implementable algorithm. However, he uses frequency-domain ICA with some additional filters which increases the computational burden and uses an array of microphones which is not possible in our mobile radio case.

In this thesis, we developed a hybrid noise cancellation algorithm using independent component analysis (ICA), particle swarm optimization (PSO) and pitch extraction (PE). ICA is a *blind source separation* technique to analyse multivariate data using statistical independence and nongaussianity properties of data. ICA is a linear transformation of data in which the desired representation minimizes statistical independence and maximizes nongaussianity. In this context, representation means that we transform the data in order to make its essential content accessible.

There are other linear transformation methods like principal component analysis and projection pursuit but ICA is more recently introduced. The technique of ICA is first introduced by J. Hérault, C. Jutten and B. Ans but it was not efficient. Important contributions to algorithm are made by J. F. Cardoso [3] and P. Comon [4]. Their works are extend by A. Cichocki and R. Unbehauen [5, 6] and ICA gained wider attraction after A.J. Bell and T.J. Sejnowski published their approach based on information-maximization principle [7] in 1995. A. Hyvärinen and E. Oja presented a fixed point algorithm, FastICA [8] in 1997. FastICA is computationally very efficient and allowed use of ICA on large-scale problems [9].

PSO is a heuristic problem solving method based on swarm intelligence (SI). It was first proposed by Kennedy and Eberhart [10] as a general optimization

tool which simulates simplified social life models like fish schools and bird flocks. In this thesis, PSO is used to find extrema of objective functions provided by ICA. Generally, ICA is used with gradient-based optimization methods [8] but we have shown here that PSO performs as well as gradient based methods or even faster. Combined ICA-PSO algorithms recently became popular in various research fields. A detailed survey on ICA-PSO (a.k.a PSO-ICA) is provided in Section 4.1.

PE algorithm is a fork of pitch period estimation algorithm in [11]. Pitch-period is a property of voiced speech signal and one of the most important parameters of parametric coders because incorrect estimation of it can cause audible artefacts in the synthesized speech (Section 5.1).

Our contributions can be summarized as follows:

- We bypass the preprocessing steps of ICA to provide computational efficiency and fasten the process of noise cancellation.
- We modified an objective function of ICA in order to reduce computational burden.
- Besides combining PSO with ICA, the rule of convergence of PSO is changed such that instead of waiting for all particles to accumulate at the same point in space, we checked whether the point providing global best remains constant. In order to prevent premature convergence, we carefully determined parameters of PSO.
- The frame based structure of ICA-PSO algorithm makes it real-time implementable.

- Changes in convergence procedure and objective functions of ICA enabled working with an extremely small swarm.
- A unique PE algorithm is combined with ICA-PSO.

In this thesis, Chapter 2 and Chapter 3 addresses the details and background information on our two main methods, ICA and PSO, respectively. In Chapter 4, we provide a survey on former ICA-PSO algorithms and clarify our modifications. Chapter 5 addresses working principles and details of PE algorithm. In Chapter 6 besides testing the performance of algorithm under various conditions, we compare and contrast our proposed algorithm with the existing algorithms in [8] and Section 2.3, and the method known as subtraction method provided in Section 6.6.2.

Chapter 2

INDEPENDENT COMPONENT ANALYSIS

Independent component analysis (ICA) is a blind source separation technique based on statistical properties of signals. It is a computational technique for revealing hidden factors that underlie sets of random variables, measurements or signals. ICA is used for extracting independent components in a signal which are mixed by an unknown mixing system. Since there is a little or no information on signals and the system mixing them, ICA is a "blind" technique.

In this chapter, first of all, basic ICA model is covered and solved in Section 2.1 by emphasizing its restrictions and ambiguities. Our discussion continues with addressing the ICA model that we used in this thesis in Section 2.2. One of the most widely used ICA methods, FastICA is given in Section 2.3 with detail since we use it in our comparisons. Finally, other ICA methods are addressed in Section 2.4 for the sake of completeness.

2.1 Basic Independent Component Analysis

Assume that the obtained data, $x(t)$, consists of m observations of different elements by T. Those elements (or variables) can be signals emitted by some physical objects or sources such as, telecommunication signals or voices of people talking on a room. Actually, cocktail-party problem, is one of the basic problems in which source signals are recordings of people talking simultaneously in a room. Assuming two people (the number of people are arbitrary but of course must be larger than 1) were talking and their voices are recorded by two microphones, the system can be modelled as

$$x_1(t) = \alpha_{11} s_1(t) + \alpha_{12} s_2(t) \tag{2.1}$$

$$x_2(t) = \alpha_{21} s_1(t) + \alpha_{22} s_2(t) \tag{2.2}$$

where $m = 2$ (number of observations), $t = 1, ..., T$ and α_{ij} are unknown mixing coefficients (weights). Another unknown in this system is the source signals, $s_i(t)$ since the problem is to find original signals from their mixtures,$x_1(t)$ and $x_2(t)$. This is the *blind source separation problem* where *blind* means we have no or very little prior information about the original signals.

One of the essential assumptions of ICA is that mixing matrix is invertible. Let us denote source signals and mixing matrices by $\mathbf{s}$ and $\mathbf{A}$

$$\mathbf{s} = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} \qquad and \qquad \mathbf{A} = \begin{bmatrix} \alpha_{11} & \alpha_{21} \\ \alpha_{12} & \alpha_{22} \end{bmatrix} \tag{2.3}$$

where the assumption is that the mixing coefficients α_{ij} are different enough to make $\mathbf{A}$ invertible. Denoting inverse of $\mathbf{A}$ as $\mathbf{W}$, which exists due to the assumption, source signals can be separated as

$$y_1(t) = \omega_{11} x_1(t) + \omega_{12} x_2(t) \tag{2.4}$$

$$y_2(t) = \omega_{21} x_1(t) + \omega_{22} x_2(t) \tag{2.5}$$

where $y_1(t)$ and $y_2(t)$ are *demixed signals*.

In addition, $\mathbf{W}$ corresponds to *demixing directions* which are

$$\mathbf{\Theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} \tag{2.6}$$

and

$$\mathbf{W} = \begin{bmatrix} \omega_{11} & \omega_{12} \\ \omega_{21} & \omega_{22} \end{bmatrix} = \begin{bmatrix} \beta_1 \cos(\theta_1) & \beta_1 \sin(\theta_1) \\ \beta_2 \cos(\theta_2) & \beta_2 \sin(\theta_2) \end{bmatrix} \tag{2.7}$$

where projections of data are parametrized by angles of Θ. Since we are looking for projections of data where its contents are extracted, parametrizing projections by angles makes it easier to search the space because instead of looking for four points (ωs), we can look for 2 angles (θs). Also note that finding β_1 and β_2 is not important since we are looking for directions.

According to ICA, if $y_1(t)$ and $y_2(t)$ are independent they are equal to $s_1(t)$ and $s_2(t)$. $y_1(t)$ and $y_2(t)$ possibly correspond to scaled versions of source signals and, in practice, $y_1(t)$ does not necessarily correspond to $s_1(t)$ but may correspond to $y_2(t)$ as well, which is one of the ambiguities of ICA.

2.1.1 Restrictions and Ambiguities

The basic ICA model must satisfy the following assumptions and restrictions to be able to estimate source signals.

- Source signals must be *statistically* independent, which means information on the value of s_i does not provide any information on the value of s_j if $i \neq j$.

- Source signals must have *nongaussian* distributions. Actually, unless both of source signals have gaussian distributions, they can be separated (if there are two source signals).

- Assuming a square mixing matrix is required for simplicity. This means that we assume number of sources is equal to the number of sensors. However, in some cases there are more observations (dimensions) than number of independent components and then dimensionality can be reduced. On the contrary, number of independent component can be larger than number of observations, which is the case of *over-complete bases* [9, Chapter 16].

- Mixing matrix must be invertible.

As another simplification, independent components are assumed to have zero mean, in other words, *centered*. Subtracting the mean, in other words *centering*, is a preliminary step for ICA algorithms in order not to cause loss of generality. Since both $\mathbf{s}$ and $\mathbf{A}$ are unknowns for us, satisfying all of those assumptions cannot prevent the following ambiguities of ICA

- Variances of independent components cannot be determined. As a result, magnitudes of independent components can be fixed such that $E\{s_i^2\} = 1$. Note that, ambiguity of the sign remains.

- Order of independent components cannot be determined as was pointed in Section 2.1.

Another preprocessing step used by many ICA algorithms frequently is *whitening* which is representing observed signals such that they are uncorrelated. However, since being uncorrelated does not imply independence, further steps defined by ICA must be taken. More information about whitening can be found in Appendix A.

Not a preprocessing step but a mid-processing step is *orthogonalization*. Basis vectors are orthogonal in theory but iterative algorithms do not always protect orthogonality. Thus, among iterations, orthogonalization methods are applied. Those methods are either *sequential* or *symmetric*

- Gram-Schmidt orthogonalization (GSO) is on of the classical sequential orthogonalization methods in which

$$\mathbf{w}_1 = \mathbf{a}_1 \tag{2.8}$$

$$\mathbf{w}_j = \mathbf{a}_j - \sum_{i=1}^{j-1} \frac{\mathbf{w_i}^T + \mathbf{a}_j}{\mathbf{w}_i^T \mathbf{w}_i} \mathbf{w}_i \tag{2.9}$$

  where $\mathbf{a}_1, ..., \mathbf{a}_m$ are n dimensional independent vectors and $m \leq n$, $\mathbf{w}_1, ..., \mathbf{w}_m$ are a set of orthogonal vectors that span the same subspace with the former set [12]. In other words, each $\mathbf{w}_j$ is a linear combination of $\mathbf{a}_j$. As a result of Eq.(2.9), $\mathbf{w}_j^T \mathbf{w}_i = 0$ if $i \neq j$. Note that, in this sequential process, first k matrices for $k < j$ are already orthogonal and the summation simplifies. Also, each $\mathbf{w}_j$ is divided by its norm, making them *orthonormal*, in other words, they are both orthogonal and they have unit Euclidean norm. The problem with the sequential methods is cumulation of error.

- In symmetric orthogonalization methods, all $\mathbf{a}_i$ are considered in the same way such that finding any orthogonal basis that spans the same subspace with $\mathbf{a}_i$ is enough. Of course this is not a unique solution if there are no other constraints. First forming the matrix $\mathbf{A} = (\mathbf{a}_1 ... \mathbf{a}_m)$ and then finding eigendecomposition of symmetric matrix $(\mathbf{A}^T \mathbf{A})^{-1/2}$ and finally putting $\mathbf{W} = \mathbf{A}(\mathbf{A}^T \mathbf{A})^{-1/2}$ provides orthonormal basis. Note that it is orthonormal because $\mathbf{W}^T \mathbf{W} = \mathbf{I}$ holds for $\mathbf{W}$. This method is preferred to be used in gradient algorithms.

2.2 ICA by Maximization of Nongaussianity

As was mentioned in previous chapters, gaussianty is crucial for independent component analysis because it is not possible to analyse components if both of them have gaussian distributions. In other words, without nongaussianity separation is not possible. Thus, intuitively, maximization of nongaussianity can be used as a measure of independence.

2.2.1 Gaussian Distributed Components Cannot Be Analyzed

Gaussian distribution has unique properties making it significant for independent component analysis (ICA). If $\mathbf{x}$ is a gaussian distributed n-dimensional random variable, it has the following density: where m_x and C_x correspond to mean and covariance matrices, respectively.

$$p_{\mathbf{x}}(\mathbf{x}) = \frac{1}{(2\pi)^{n/2} det(\mathbf{C_x})^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{m_x})^T {\mathbf{C_x}}^{-1}(\mathbf{x}-\mathbf{m_x})\right) \tag{2.10}$$

If x is a one dimensional random variable ($n = 1$) gaussian density is the following:

$$p(\mathbf{x}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x-\mu}{2\sigma^2}\right) \tag{2.11}$$

where σ is the variance and μ is mean of the random variable, x.

Some properties of gaussian distribution is important for ICA:

- Linear transformations of gaussian distributed random variables are gaussian distributed, too.
- Uncorrelatedness means independence.
- Knowledge of statistics higher than second order is not needed.

- Gaussian distribution is the most random distribution among all other distributions having the same mean and covariance matrices. In an information theoretic view, gaussian distribution has the largest entropy.

While the fist two property makes gaussian distributed random variables unidentifiable by ICA, last ones make it a measure of independence. I would like to explain the effect of the first property in this part.

Assume that we have two gaussian distributed sources, s_1 and s_2. We do not have any pripor information about the sources but we observe two linear mixtures of them via two separate receivers. Let us denote source signals and mixing matrices by $\mathbf{s}$ and $\mathbf{A}$.

$$\mathbf{s} = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} \qquad and \qquad \mathbf{A} = \begin{bmatrix} \alpha_{11} & \alpha_{21} \\ \alpha_{12} & \alpha_{22} \end{bmatrix} \tag{2.12}$$

Thus, received signals become $\mathbf{r} = \mathbf{As}$. In other words, $\mathbf{r}$ is a linear mixture of the source signals. Now, let us further make two assumptions:

- s_1 and s_2 are jointly gaussian. According to (1.1) source signals have the following distribution:

$$p(s_1, s_2) = \frac{1}{2\pi} \exp\left(-\frac{s_1^2 + s_2^2}{2}\right) = \frac{1}{2\pi} \exp\left(-\frac{\|\mathbf{s}\|^2}{2}\right) \tag{2.13}$$

- Mixing matrix, $\mathbf{A}$, is orthogonal

On the one hand, assuming an orthogonal mixing matrix does not cause loss of generality because whitening, one of ICA preprocessing steps, turns any mixing matrix into an orthogonal one. More information about preprocessing steps can be found in the Appendix. On the other hand, this assumption is very because for an orthogonal matrix $\mathbf{A}^{-1} = \mathbf{A}^T$ holds. Thus, $\mathbf{s} = \mathbf{A}^T\mathbf{r}$. If we re-write the distribution of source signals:

$$p(r_1, r_2) = \frac{1}{2\pi} \exp\left(-\frac{\left\|\mathbf{A^T r}\right\|^2}{2}\right) \left|\det \mathbf{A^T}\right| \tag{2.14}$$

The determinant term comes from linear and nonsingular transformations of probability density function (pdf). If $\mathbf{y} = \mathbf{Ax}$ and $\mathbf{x} = \mathbf{A}^{-1}\mathbf{y}$ then:

$$p_{\mathbf{y}}(\mathbf{y}) = \frac{1}{|det\mathbf{A}|} p_{\mathbf{x}}(\mathbf{A}^{-1}\mathbf{y}) \tag{2.15}$$

Since $\mathbf{A}$ is orthogonal, $\left\|\mathbf{A^T x}\right\|^2 = \|\mathbf{x}\|^2$ and $\left|\det \mathbf{A^T}\right| = 1$. So, (1.5) turns into (1.7) not providing any information about the mixing matrix, $\mathbf{A}$.

$$p(r_1, r_2) = \frac{1}{2\pi} \exp\left(-\frac{\|\mathbf{r}\|^2}{2}\right) \tag{2.16}$$

Thus, it is not possible to identify the mixing matrix for gaussian random variables. As a result, it is not possible to separate gaussian random variables from each other. All we can do is to obtain an orthogonal transformation of the received signals. In other words, gaussian distributed components cannot be analyzed. However, mixture of a gaussian and a nongaussian component can be analyzed.

2.2.2 Nongaussianity means independence

As central limit theorem shows sum of two independent nongaussian random variables is more gaussian than any of them. That is the basic idea of relating independence to nongaussianity. Note that we have linear mixtures (summations) of independent component. Let us try to find the inverse of mixing matrix by trial-and-error. If we find the exact inverse, since the outcome will consist of separated components,instead of their mixtures, it will be the most nongaussian one. All other outcomes will be more gaussian because they will contain addition of two independent components.

Since we observe linear mixtures of source signals, $\mathbf{r} = \mathbf{As}$. Let us denote inverse of the mixing matrix, $\mathbf{A}$, as $\mathbf{W}$. So,

$$\mathbf{W} = \mathbf{A}^{-1} \qquad and \qquad \mathbf{W} = \begin{bmatrix} \omega_{11} & \omega_{21} \\ \omega_{12} & \omega_{22} \end{bmatrix} \tag{2.17}$$

If $\mathbf{W}$ can be found, $\mathbf{s} = \mathbf{A}^{-1}\mathbf{r} = \mathbf{W}\mathbf{r}$. Thus, separated components are also linear mixtures of received signals. Now, let $\mathbf{y}$ vector denote one of the separated IC and note that it is a linear combination of received signals:

$$\mathbf{y} = \mathbf{b}^T\mathbf{r} = \sum_i b_i x_i \tag{2.18}$$

Here, $\mathbf{b}$ vector corresponds to one of the columns of the mixing matrix, $\mathbf{A}$. For instance, if $\mathbf{A}$ is 2×2, $\mathbf{b}$ is 2×1 and $\mathbf{r}$ is 2×N. Expressing $\mathbf{r}$ in terms of $\mathbf{s}$, $\mathbf{y}$ becomes linear combination of independent components:

$$\mathbf{y} = \mathbf{b}^T\mathbf{A}\mathbf{s} = \mathbf{q}^T\mathbf{s} = \sum_i q_i s_i \tag{2.19}$$

If $\mathbf{b}$ is exact inverse of one of the columns of $\mathbf{A}$, $\mathbf{q}^T\mathbf{s}$ must give one of the independent components. In other words, one of the elements of $\mathbf{q}$ must be 0 and the other must be 1:

$$\mathbf{s}_i = \begin{bmatrix} q_1 & q_2 \end{bmatrix} \mathbf{s} \tag{2.20}$$

So, if we take $\mathbf{b}$ as a vector that maximizes nongaussianity of $\mathbf{b}^T\mathbf{r}$, it corresponds to $\mathbf{q} = \mathbf{A}^T\mathbf{b}$ with only one nonzero component. As a result, we can say that nongaussianity is a measure of independence.

2.2.3 Measures of Nongaussianity

Robust measures of nongaussianity is necessary to decide whether independent components are separated or not. It is possible to use two measures of nongaussianity:

1. Kurtosis
2. Negentropy

Both measures depend on higher order statistics than second order because higher order statistics of gaussian distributed random variables does not provide any information as shown in Section 2.2.1. However, in practice, negentropy is a more

robust measure of negentropy. In this context, robustness is being insensitive to outliers, fast and adaptive.

Kurtosis

Kurtosis is the name of the fourth-order cumulant of a random variable. Cumulants κ_k of x (1.13) are the coefficients of the Taylor series expansion of the second order characteristic function (1.12). The second order characteristic equation is the following:

$$\phi(\omega) = ln\left(\varphi(\omega)\right) = ln\left(\mathbf{E}\left\{\exp(j\omega x)\right\}\right) \tag{2.21}$$

Taylor series expansion is:

$$\phi(\omega) = \sum_{k=0}^{n} \kappa_k \frac{(j\omega)^k}{k!} \tag{2.22}$$

Finally kth order cumulant becomes:

$$\kappa_k = (-j)^k \left.\frac{d^k \phi(\omega)}{d\omega^k}\right|_{\omega=0} \tag{2.23}$$

Since one of the pre-processing steps is centering, consider first two cumulant of a zero mean random variable:

$$\begin{aligned} \kappa_1 =& 0, \qquad \kappa_2 = \mathbf{E}\left\{x^2\right\}, \qquad \kappa_3 = \mathbf{E}\left\{x^3\right\} \\ \kappa_4 =& \mathbf{E}\left\{x^4\right\} - 3\left[\mathbf{E}\left\{x^2\right\}\right]^2 \end{aligned} \tag{2.24}$$

Also, if variance of the random varible is 1 (a normalized random variable), the fourth order cumulant simplifies to a normalized version of the fourth moment, $\kappa_4 = \mathbf{E}\left\{x^4\right\} - 3$. Fourth moment of gaussian distributed random variables are $3\left(Ey^2\right)^2$. Thus, kurtosis and all higher order cumulants of gaussian distributed random variables are zero, as mentioned in Section 2.2.1. For other distributions kurtosis is either positive or negative.

If kurtosis of a random variable is positive, it is *supergaussian*, otherwise it is *subgaussian*. Laplacian density is one of the supergaussian densities. Speech resembles to Laplacian density. Its pdf is given by

$$p(y) = \frac{1}{\sqrt{2}} \exp\left(\sqrt{2}\,|y|\right) \tag{2.25}$$

Absolute value of Kurtosis is used to measure nonguassianity. It is zero for gaussian distribution and larger than zero for other distributions.

Negentropy

Negentropy originates from differential entropy, a concept of information theory. Entropy is the measure of randomness of a random variable. As mentioned in Section 2.2.1 Gaussian random variable has the largest entropy, in other words, it is the most *random* random variable. Thus, entropy can be used as a measure of nongaussianity. Differential entropy of a random vector $\mathbf{x}$ with density $p_y(\eta)$ is defined as

$$H(\mathbf{y}) = -\int p_y(\eta) \log p_y(\eta) d\eta \tag{2.26}$$

Negentropy, J, of a random vector $\mathbf{y}$ is defined as

$$J(\mathbf{y}) = H(\mathbf{y}_{\mathbf{gauss}}) - H(\mathbf{y}) \tag{2.27}$$

where $\mathbf{y}_{\mathbf{gauss}}$ has the same covariance and mean matrix with $\mathbf{y}$ and also it is gaussian distributed. Note that negentropy is zero for a gaussian distributed random variable and negative for other distributions. Negentropy is an optimal measure of nongaussianity, both in theory and in practice. However, it is computationally very difficult. As a result, some approximations of negentropy are used.

From this point on, I would like to consider scalar cases for simplicity. There are two approximations of negentropy:

- cumulant based approximation
- approximation via nonpolynomial moments

The cumulant based approximation ends up with a kurtosis like function, as expected:

$$J(y) \approx \frac{1}{12} E\{y^3\}^2 + \frac{1}{48} kurt(y)^2 \tag{2.28}$$

This approximation is very similar to using kurtosis such that it is the squared version of it. Thus, that is not a robust approximation. Again, it is sensitive to outliers and mainly measures the tails of distribution and largely unaffected by structure near the centre of distribution. As a result, we need a more sophisticated approximation and that is provided by nonpolynomial moments.

In this approach, we extend the cumulant based approach so that it uses expectations of general nonquadratic functions or *non-polynomial moments* [13, 14, 15]. Basically, we change y^3 and y^4 with non-quadratic functions, G^i where i is an index, not a power. Then we can approximate negentropy based on the expectations of G^i by choosing G^i wisely, which is very important. They must have the following properties in order to estimate negentropy in a robust way:

- $E\{G^i\}$ must be insensitive to outliers so, G^i must grow slower than quadratically
- G^i must contain the source signal's statistical properties related to entropy. For instance, if $p_y(\eta)$ were known, G^i would be $\log p_y(\eta)$. So that $E\{G^i\}$ would be exactly entropy of $p_y(\eta)$.
- G^i must be linearly independent

As a simple case, taking an odd G^1 and en even G^2 the following approximation is obtained

$$J(y) \approx k_1(E\{G^1(y)\})^2 + k_2(E\{G^2(y)\}) - (E\{G^2(v)\})^2 \tag{2.29}$$

where k_1 and k_2 are positive constants and v is a gaussian random variable with the same mean and variance of y. Even if the approximation is not very accurate, it is still a good measure for nongaussianity since it is zero for a gaussian random variable and always negative for other distributions. If we use only one nonquadratic function Eq. (2.29) becomes

$$J(y) \propto [E\{G(y)\} - E\{G(v)\}]^2 \tag{2.30}$$

The following choices of G are proved very useful:

$$G_1(y) = \frac{1}{a_1} \log \cosh a_1 y \tag{2.31}$$

$$G_2(y) = -\exp\left(-\frac{y^2}{2}\right) \tag{2.32}$$

Both approximations of negentropy and kurtosis provide measures of non-gaussianity which are objective functions for ICA algorithms. One of the most widely used algorithms for optimizing those objective functions is a fast fixed-point ICA algorithm, FastICA, first introduced by Hyvärinen et al. in [8] and then generalized for various objective functions in the following years [13, 14, 15]. The objective function that we use is explained in detail in Section 4.2.1.

2.3 FastICA

For whitened data, $\mathbf{z}$, the one-unit FastICA algorithm has the following form [16]

$$\mathbf{w}(k) = E\{\mathbf{z}g(\mathbf{w}(k-1)^T\mathbf{z})\} - E\{g'(\mathbf{w}(k-1)^T\mathbf{z})\}\mathbf{w}(k-1) \tag{2.33}$$

where $\mathbf{w}$ is demixing matrix, k is iteration number and g is the derivative of any G defined in Section 2.2.3. Note that, sample mean is used as the expectation of data so, number of samples, i.e. window size, must be large enough. The basic one-unit algorithm using a gradient based optimization method can be summarized as follows [9, Chapter 8]

1. Center data such that $\mathbf{x} = \mathbf{x} - E\{\mathbf{x}\}$ where $E\{\mathbf{x}\}$ is the sample mean.

2. Whiten data (obtain $\mathbf{z}$)

3. Initialize $\mathbf{w}$. Initial value can be random or depend on a guess about the original signal. Note that $\mathbf{w}$ has unit norm.

4. Let $\mathbf{w} \leftarrow E\{\mathbf{z}g(\mathbf{w}^T\mathbf{z})\} - E\{g'(\mathbf{w}^T\mathbf{z})\}\mathbf{w}$

5. Normalize $\mathbf{w}$ such that $\mathbf{w} \leftarrow \frac{\mathbf{w}}{\|\mathbf{w}\|}$

6. If not converged, go back to step 4

Also, a FastICA algorithm without whitening as a preprocessing step is presented, too [8]. If the aim is to estimate several independent components, FastICA either consists of several iterations of a one-unit algorithm or all components can be estimated via a parallel process, according to type of used orthogonalization. If deflation based orthogonalization methods are used, components are estimated one-by-one. The other option is to use sequential orthogonalization in which all data are estimated in parallel. In this case, no data has privilege over one another. So, FastICA is a general algorithm that can optimize either one-unit or multi-unit objective functions [17].

Other methods of independent component analysis are presented in the following chapter for completeness.

2.4 Other Methods

After deciding on which objective function to use, the proper ICA algorithm for optimization must be chosen. Different methods can be compared with respect to stability, convergence speed, memory requirement or whatever critical for a

certain application.

The pioneering work on ICA is Jutten-Hérault algorithm which is inspired by neural networks [18]. Since Jutten-Hérault algorithm could converge under severe restrictions, many algorithms upon them are developed [16]

- Non-linear decorrelation algorithms [6, 5, 19] and [20, 21] reduced computational overhead and increased stability
- Algorithms for maximum likelihood or infomax estimation constitute and important class of ICA approximations. In [7, 22, 23] natural gradient is used for maximizing likelihood whereas [24] proposes a Newton method.
- Non-linear PCA algorithms were introduced in [25]
- Some neural algorithms are relevant to ICA such as [26] using kurtosis and [27] working on non-whitened data.
- Some adaptive (neural) algorithms are also applied to ICA like exploratory projection pursuit algorithms [28] and least-squares type algorithms in [29].
- Tensor based algorithms [30, 31, 32, 33, 4] which are batch algorithms and not suitable for using with large dimensional data.
- Weighted covariance methods [3]

So, two general branches of ICA algorithms are *adaptive* algorithms and *batch-mode* algorithms. While adaptive algorithms changes its behaviour according to data in an on-line manner, batch-mode algorithms evaluate blocks of data. FastICA algorithm is not adaptive since it uses sample averages computed over larger samples of the data. It is a very efficient batch algorithm which can be used with both one-unit and multi-unit objective functions.

A more recent optimization method used on ICA is *Particle Swarm Optimization* (PSO) which I explain in detail in the following chapter.

Chapter 3

PARTICLE SWARM OPTIMIZATION

Beginning with a basic question, why do we need optimization, the concept of optimization is discussed in Section 3.1. Afterwards, swarm intelligence, which is the origin of PSO, is discussed in Section 3.2. Discussion on PSO is finalized by investigating its basic forms in Section 3.3 and some improvements and modifications on PSO algorithms in Section 3.4.

3.1 Optimization

A general definition of optimization is that it is the process of adjusting a system to get the best possible outcome. The system is not necessarily a mathematical function. For instance, all engineering design processes are optimization since aim of them is to choose design parameters to improve some objective. Also, many business decisions, like supply chains and investment portfolios are optimization processes. Varying decision parameters lead to higher profit. Moreover, from a psychological point of view, negotiations to solve problems among people

too can be considered as optimization. Actually, optimization is a natural consequence of problem solving business of both evolution and mind.

If a function is considered, optimization process is driven in three spaces: Parameter space, function space and fitness space. *Parameter space* contains all elements entering to the function and also known as *search space*. *Function space* consists of results of operations on elements. Though two former spaces can be multidimensional according to the elements, *fitness space* is one dimensional and contains only 'goodness' information. Goodness or error is the degree of success of parameters on optimizing the problem via the values in the function space.

Optimization process aims to minimize error and maximize goodness for the system. However, this may involve maximisation or minimisation of tasks. Considering problems as a kind of task, systems of functions can be investigated. Maximization of a function f can be seen as minimizing $-f$, terms of *maximization*, *minimization* and *optimization*can be used interchangeably. A general optimization problem can be defined as minimizing the *objective function*, f_0, with respect to n design parameters, x. Note that if the same problem was a maximization problem of objective function, g would be equal to $-f$.

There are many optimization algorithms with important considerations depending on special cases of problems. For instance, optimization can be *linear* or *non-linear* according to system's model. There are efficient linear programming methods to solve linear optimization problems but non-linear problems are harder to deal with, which is also our case. Another consideration is dealing with *constrained* or *unconstrained* tasks. Unconstrained tasks are easier to deal with and generally defined as

$$\text{Given } f : \mathbb{R}^n \to \mathbb{R}$$

$$\text{find } x^* \text{ such that } f(x^*) \leq f(x), \forall x \in \mathbb{R}$$

One of the simplest constraints is known as *box-constraint* or *bound* constraint such that $x_k^{min} < x_k < x_k^{max}$. Constraints like non-negativity of all parameters are harder problems.

Another consideration is *multimodality* and its opposite *unimodality* of functions. A multimodal function has more than one unique global optima. For instance, $x^2 = 25$ is a multimodal function because it has two optima $x = 5$ and $x = -5$. On the other hand, a unimodal function has only one optimum solution, e.g $x - 4 = 0$. There are other considerations like convexity and differentiability but all techniques with those considerations can be investigated in two main categories: Local optimization and global optimization.

3.1.1 Local Optimization

As the name implies, local optimization targets an area (or subset), B in the search space, instead of the whole space, S. A *local optimizer*,$f(x^*)$ is defined as

$$f(x^*) \leq f(x), \forall x \in \mathbb{B} \tag{3.1}$$

If the optimization is unconstrained, $S = \mathbb{R}^n$. Note that S can contain other proper regions such that $B_i \cap B_j = 0$ unless $i \neq j$. However, local minimums of different regions can have the same values in function space. In other words, $f(x_i^*) = f(x_j^*)$ when $i \neq j$ is possible. Many local optimization algorithms uses an initial point, $\mathbf{z}_0 \in S$ to search locally around it. It is expected for a local optimization algorithm to find the minimum in the same subset with $\mathbf{z}_0$. But some algorithms only guarantee that they will find a local minimum which is not necessarily the closest one to $\mathbf{z}_0$ and can be in another subset.

3.1.2 Global Optimization

A global optimizer is described in a similar but different way to the local optimizer in Eq. (3.1)

$$f(x^*) \leq f(x), \forall x \in \mathbb{S} \tag{3.2}$$

where S is the search space and $S = \mathbb{R}^n$ if optimization is unconstrained. Similar to local optimization algorithms, global optimization algorithms generally use an initial point $\mathbf{z}_0 \in S$. Though the term *global optimization* is used to mean the process of finding x^* in Equation 3.2 in this thesis, it sometimes mean finding x^* in $\mathbf{B}$ without depending on position of $\mathbf{z}_0$. Such algorithms first take *global* steps to find a region $\mathbf{B}_i$ where it is possible to find the minimum of $\mathbf{B}_i$ via taking *local* steps.

3.1.3 No Free Lunch Theorem

No Free Lunch Theorem (NFL), introduced by Wolpert and Macready [34] states that no optimization algorithm is better than the others, averaged over all objective functions in a finite search space. For the programmers who were trying to develop an algorithm which would be a first choice for any kind of problems, NFL was very interesting since it claims that a blind guess is as good as a special algorithm. Though it is thought that NFS will not be valid in small subsets of *all* functions, it is shown to be valid in smaller subsets [35]. So, optimization algorithms can be superior to one another for a specific type of problems, rather than being superior at all kinds of possible problems. For the problems we discuss in this thesis, the closest competitor of PSO is the gradient-based optimization algorithms, which are used with ICA frequently.

3.2 Swarm Intelligence

As well as individual intelligence, there is intelligence of a society because thinking is social. Swarm intelligence (SI) is defined as "the emergent collective intelligence of groups of simple agents." by Bonabeau et al [36]. Population of simple agents interacts both with their environment and locally with each other in SI systems. Ant colonies, bird flocking, animal herding, bacteria molding and fish schooling are examples of SI systems in nature. Five basic principles of swarm intelligence are proposed by Mark Millonas [37], who develops swarm models for artificial life applications:

- Proximity: Ability to perform basic space and time computations.
- Quality: Ability to respond to quality factors in the environment.
- Diverse response: Activity of population must be spread along various channels.
- The principle of stability: The population must not change very rapidly.
- The principle of adaptability: Ability to change behaviour mode when it is worth the "computational price".

All five of Millonas' principles describe particle swarms. Why Kennedy and Eberhart preferred the word, "particle" is explained in Section 3.3. Since all agents disperse through out different regions of search space, such population-heuristic methods are less likely to trap into locally optimum points. However, in some cases all agents may stiff into the same region before finding the global optimizer. That is called *premature convergence*, and the agents are said to be *prematurely convergenced*. In order to avoid that, population-heuristic methods, including swarm intelligence, try to add some randomness into search process. Premature convergence is discussed in more detail in the following sections, especially in Section 3.4.

3.2.1 Adaptive Culture Model

Adaptive Culture Model (ACM) is a computational model of dissemination of culture, introduced by Robert Axelrod in 1997 [38]. Humans not only consider their own experiences but also learn from models introduced by others' experiences. Those models enable knowledge and skills spread within a population, as naturally as learning from one another, making it converge to an optimal process. That adaptation system operates on a pattern among individuals like three circles, enlarging from close to distant individuals, simultaneously:

- Individuals learn from their neighbours. Interacting with their neighbours and exchanging experiences are the most local part of this phenomenon.
- Group level processes, emerge from spread of knowledge through social learning. At this point, it would be advantageous to remember story about the six blind men and the elephant by Jhon Godfrey Saxe (1869-1936). The story describes that each one of the blind men discovers a certain part of the elephant, like tusks and legs, but think that the whole elephant consists of that part only. If they are not also deaf and able to communicate, they can discover that elephant is a creature containing, legs like trees and tusks like spears. This short story describes that the society is able to benefit from individuals' partial knowledge and construct a culture, beyond experiences of any individual.
- Culture optimizes cognition and reaches distant individuals. Insights and innovations are carried by culture and combination of various innovations makes better models appear. This is the most global effect.

In other words, the idea states that interactions among individuals spread within society and result effective models. That whole process is called "cognitive optimization" by Eberhart and Kennedy [39, Chapter 6]. On the one hand, "particle

swarm adaptation" (PSA), which is computer simulations of societies exchanging experiences in a multivariate real-number space, has this point of view, too. On the other hand, ACM and PSA are two branches of the same tree because ACM simulates societies in terms of discrete variables while PSA is simulated in continuous or binary space. Both of them consists of individuals imitating successful others' to reach optimal solution but the space they evolve differs. That is also why we preferred PSO, a version of PSA, to work the real-number space. Though ACM can find optimal solutions, it is only designed to show effectiveness of imitating better individuals. However, PSO is designed to focus on "the ability of social interaction to result in optimization of hard problems" [39, Chapter 6].

3.3 Particle Swarm

Note that the "circles" in Section 3.2.1 are the higher level of cultural adaptation since they show the patterns among individuals. However, properties of individuals, in other words, their behaviours must be taken into consideration, too. Kennedy summarizes them in terms of three principles [39, Chapter 7]:

- Evaluate: The ability to evaluate a very fundamental concept that even the most basic organisms can evaluate certain conditions of the environment surrounding them. Also, evaluation is necessary for learning such that "learning could even be defined as a change that enables the organism to improve the average evaluation of its environment". In other words, learning cannot occur if the organism cannot evaluate.

- Compare: Comparison enable individuals to measure themselves and reorganize their position in population. This is a key ability to motivate individuals to imitate their neighbours at better positions.

- Imitate: Imitation is rarely found in nature, because it is not simply behaving the same way but understanding the reasons and using it when necessary. For humans, true imitation is central to sociality, acquisition and maintenance of mental abilities.

The view point of Eberhart and Kennedy is different than cognitive viewpoint because they think mind is not isolated from the society but it is a "public phenomenon". The swarm that we are talking about consists of "particles" instead of other options like "agents" or "points". The term "agent" is too comprehensive for the swarm members which tend to be homogeneous and follow their programs explicitly. On the other hand, the term "point" is not proper for individuals moving with certain velocity, though the individuals are almost volumeless and massless.

3.3.1 Particle Swarm in Binary Search Space

Assume that our swarm consists of very simple individuals, only can decide "yes" (1) or "no" (0), which are binary decisions. Those simple individuals know how well their decision and their neighbours' decisions performed and keep in mind the best, in other words the most positive performances ever. If they were humans, they would be talking with their neighbours about performances and trying to imitate their neighbours if their performance is better. They also know the best performance in the whole swarm, even if it belongs to the most distant member. Note that individuals are only influenced by the best performances. This approach may be too simple for actual swarms but its catches the basic principles.

Individuals can be connected to each other via various patterns, in other words, connections among individuals can vary. Most particle swarm algorithms use either one of the following sociometric principles or both of them:

- *gbest*: This is the "globally best" performance, performed by any member of the swarm. Obviously, "g" stands there for mentioning "global". This concepts, actually, connects all individuals since all of them are influenced by *gbest*.

- *lbest*: This is the "locally best" performance, performed by neighbours' of the particle, in other words, k nearest particles the it is connected with. Similarly, "l" stands there for mentioning "local" best. For instance, if $k = 2$, the particle i is connected to (knows performances of) particles $i-1$ and $i+1$. Various topologies are possible and they cause various effects.

Note that the particle also knows its own best position. Thus, the particles must be able to evaluate (their choices), compare (with their neighbours) and imitate (best decisions) a number of binary choices in order to make consistent decisions. From the psychological point of view, concept of cognitive dissonance for humans can be used to explain the sense of tension when consequent decisions are inconsistent. When we feel (evaluate) discomfort, we feel motivated to change the situation, in other words, improve the evaluation. Goodness of that cognitive evaluation can be measured by only a single measure, as provided in Festinger's description of cognitive dissonance, like "fitness" being a single measure of genetic or phenotypic goodness [39].

There are plenty of theories about improving cognitive fitness. We will not go into the details of those theories but we are interested in *subjective norm*, described by Ajzen and Fishbein's *Reasoned Action Model (1980)* [40]. The individual's subjective norm toward a behaviour consists of the others judgements on the action and its motivation to perform with them. Note that this is a very social concept. It can be formulated as sum of the products of individuals' beliefs that certain others (neighbours) think they should or should not perform the behaviour (their judgement), multiplied by the motivation to agree with each

of those others:

$$SN_0 = \sum_{i=1}^{n} b_i m_i \tag{3.3}$$

where b_i are outcomes of behaviours and m_i is the motivation of the individual. On the other hand, there is a more personal part in *Reasoned Action Model*, which is called *attitude*. It is a combination of individual's belief that certain action will result some outcomes b_i and individual's evaluation of those outcomes e_i:

$$SN_0 = \sum_{i=1}^{n} b_i e_i \tag{3.4}$$

Both of those concepts, *subjective norm* and *intend* has a root in Boyd and Richerson's cultural transmission model [39]. This model has two terms:

- Individual term: Attitude toward a behaviour, in other words, individual learning
- Social term: This term corresponds to subjective norm, in other words, cultural transmission

Eberhart and Kennedy theorize that those two terms are key to human intelligence since knowledge from individual experiences and from others' experiences provide an intellectual advantage. As an addition to previous factors affecting individual's decisions, current position of the individual's attitude towards the issue must be taken into account. For instance, if the initial attitude of individual is negative, positive experiences should occur over and over to change the attitude into positive. On the other hand, the more extreme the position is, the lower tendency has the individual to change its position by trying another alternative.

All factors affecting the individual's binary decisions and considered up to this point are formulated in mathematical terms as a function of social and personal

factors by Kennedy and Eberhart (1997) as the following:

$$P(x_{id}(t) = 1) = f(x_{id}(t-1)), v_{id}(t-1), p_{id}, p_{gd}) \tag{3.5}$$

where

- i indicates the individual
- d indicates the site of the bitstring formed by ith individual's decisions. Note that the individual makes a number of binary decisions, forming a bitstring like "10110101110"
- t is the current time step and t-1 is the previous step
- $P(x_{id}(t) = 1)$ is the probability that individual's decision will be positive or "yes" or 1 for the bit at dth side of the bitstring.
- x_{id} is the current state of the bitstring site d
- $v_{id}(t-1)$ is the latest disposition of the individual. In other words, it is the probability of choosing 1.
- p_{id} is the best decision given so far. If best result is obtained when decision was 1, p_{id} is 1. Otherwise, it is 0.
- p_{gd} is the neighbourhood's or global best, depending on the topology used. Similarly to p_{id}, it is 1 if the best result is obtained when decision was 1, otherwise it is 0.

On the one hand, stochastic structure of the decisions provide greater ability to discover new opportunities for the individual. On the other, it can cause exploitation of certain patterns near best particles, making the particle search less. The uncertainty of decisions can be used to balance among those two situations.

Desired probabilistic adjustment can be gathered via $v_{id}(t)$, which is the particle's predisposition to decide. The higher $v_{id}(t)$, the more particle is likely to decide 1 or vice versa. Since particles' decisions are influenced by their own and their neighbours' best positions, $v_{id}(t)$ must depend on both of them. In addition, we previously mentioned that particles' current positions affect their decisions. Thus, $v_{id}(t)$ could be simply summed up by $(p_{id} - x_{id}(t))$ and $(p_{gd} - x_{id}(t))$. However, in any situation we do not know whether personal or social influence is superior. Weighting both personal and social terms with random numbers, each of one of them will be stronger from time to time.

Binary decision is formulated in [39] as the following:

$$v_{id}(t) = v_{id}(t-1) + \varphi_1(p_{id} - x_{id}(t-1) + \varphi_2(p_{gd} - x_{id}(t-1)) \tag{3.6}$$

$$\text{if } \rho_{id} < s(v_{id}(t)) \text{ then } x_{id}(t) = 1; \text{ else } x_{id}(t) = 0$$

where the symbol φ represents a positive random number selected from a uniform distribution with a predefined upper limit, ρ_{id} is a vector of random numbers uniformly distributed in [0, 1] and $s(v_{id}) = \frac{1}{1+\exp(-v_{id})}$ which is the sigmoid function. The sigmoid function is used to provide a decision threshold such that if $v_i dt$ is higher, the particle is more likely to choose 1 and if it is lower, particle is more likely to choose 0. Also, v_{id} must not be close to either 0 or 1, so it can be limited by a constant parameter, V_{max}. So, decision can flip and v_{id} does not move toward infinity. V_{max} is set at ± 4, practically, because $s(V_{max}) = 0.0180$ that a bit will flip. In this model, each particle search for a better solution by making decisions influenced by its own success and neighbours' success. As a particle imitates its neighbours' successful decisions, it may come up with a better result and this process is performed thorough out the population. Thus, good decisions spread thorough the population and a culture is formulated, as was explained in Section 3.2.1. The pseudo-code of the algorithm maximizing goodness is given as the following in [39]

loop

for $i = 1 \rightarrow$ number of individuals **do**

if $G(\vec{x_i}) > G(\vec{p_i})$ **then**

for $d = 1 \rightarrow dimension$ **do**

$\{\rho_{id}$ is best so far$\}$

$\rho_{id} = x_{id}$ Next d

end for

end if

$g = i$

for $j =$ indexes of neighbours **do**

if $G(\vec{p_i}) > G(\vec{p_g})$ **then**

$g = j$

end if Next j

end for

for $d = 1 \rightarrow$ number of dimensions **do**

$v_i(t) = v_{id}(t-1) + \varphi_1(\rho_{id} - x_{id}(t-1)) + \varphi_2(\rho_{gd} - x_{id}(t-1))$

$v_{id} \in (-V_{max}, +V_{max})$

if $\rho_{id} < s(v_{id}(t))$ **then**

$x_{id}(t) = 1$

else

$x_{id}(t) = 0$

end if Next d

end for Next i

end for

Until criterion

end loop

3.3.2 Particle Swarm in Continuous Numbers

Up to this point, particle swarm algorithm originating from ACM are explained in a basic form, binary PSO. However, particle swarm, as introduced in [10] , is an optimization algorithm searching for the optimal solution in n dimensional search space, R^n.

Particles move in a heterogeneous space such that some regions of the search space are more advantageous, providing the particles better solutions. This situation is valid for both psychology and mathematical function systems such that when a vector of cognitive or mathematical parameters is evaluated, presence of some attractive regions is expected. Thus, current position of the particle has an influence on its attitude.

The particles in the swarm move towards the optimal solution with a velocity. Though parameters of a function could be conceptualized as point, velocity and acceleration are properties of particles, more than points. Particles behave like individuals in a society, so their movements have sociological basis, as partly explain in Section 3.2.1 and 3.3.1. Particles are influenced by their neighbours' attitudes towards cases. A sociological insight to this action is that particles moves toward one another like people searching agreement with their neighbours. Note that, there are two steps of action before moving towards each other, evaluation and comparison. While evaluation is fundamental for learning, comparison is necessary for being social.

Position of particle i is indicated with $\vec{xi}$ which is an algebraic vector of any size. Displacement of a particle is explained by velocity, $\vec{v_i}$ and new position is

found by:

$$\vec{x_i}(t) = \vec{x_i}(t-1) + \vec{v_i}(t) \tag{3.7}$$

The critical point is to define $\vec{v_i}$ because algorithm samples the space with the movement of particles. As mentioned in previous chapters, individuals are influenced by their own and their neighbours' behaviours, according to social-psychological theory. The neighbourhood relation depends on topological closeness, instead of the one in parameter space. For instance, there may be a person who has same opinions with you but you or the people you know have never met. So, that person has no influence on you. Similar to the binary case, a neighbourhood is defined for particles in a topological array. So, neighbours' and personal best solutions must be taken into account while the displacement is being evaluated. As a result, new position of the particle is formulated as the following:

$$\vec{x_i}(t) = f(\vec{x_i}(t-1), \vec{v_i}(t-1), p_i, p_g) \tag{3.8}$$

Though this continuous case is very similar to binary case, there are some key differences such that rate of change is in terms of velocity instead of probability. Displacement of a particle is a function of the its evaluation of its and its neighbours' best position and the comparison of those evaluations with particle's current position. Evaluation corresponds to knowledge or learning whereas comparison is simple the differences between particles current position and its and its neighbours' best positions. Thus, the formulation of displacement is very similar to formulation of probability of flipping in the binary case:

$$\vec{v_i}(t) = \vec{v_i}(t-1) + \varphi_1(\vec{p_i} - \vec{x_i}(t-1)) + \varphi_2(\vec{p_g} - \vec{x_i}(t-1)) \tag{3.9}$$

$$\vec{x_i}(t) = \vec{x_i}(t-1) + \vec{v_i}(t) \tag{3.10}$$

Again similarly to binary case φ_1 and φ_2 are used to construct a balance between social and personal comparisons such that the particle cycles unevenly around:

$$\frac{\varphi_1\vec{p_i} + \varphi_2\vec{p_g}}{\varphi_{21} + \varphi_2} \tag{3.11}$$

whose location changes at every iteration. Note that the sub index d, indicating each dimension, is not used up to this point because all evaluations are vectorial containing all dimensions of variables. In order not to exploit, each dimension $\vec{v_i}$ is limited by V_{max} in the following way:

$$\text{if } v_{id} > V_{max} \text{ then } v_{id} = V_{max}$$

$$\text{else if } v_{id} < -V_{max} \text{ then } v_{id} = -V_{max}$$

Thus, particles do not fly away but fly in certain boundaries and still search the space. The pseudocode for PSO in continuous numbers is provided in [39]:

loop
  for $i = 1 \rightarrow$ number of individuals **do**
    if $G(\vec{x_i}) > G(\vec{p_i})$ **then**
      for $d = 1 \rightarrow dimension$ **do**
        $\{\rho_{id}$ is best so far$\}$
        $\rho_{id} = x_{id}$ Next d
      end for
    end if
    $g = i$
    for $j =$ indexes of neighbours **do**
      if $G(\vec{p_i}) > G(\vec{p_g})$ **then**
        $g = j$
      end if
    end for
    for $d = 1 \rightarrow$ number of dimensions **do**
      $v_{id}(t) = v_{id}(t-1) + \varphi_1(\rho_{id} - x_{id}(t-1)) + \varphi_2(\rho_{gd} - x_{id}(t-1))$
      $v_{id} \in (-V_{max}, +V_{max})$
      $x_{id}(t) = x_{id}(t-1) + v_{id}(t)$
    end for

end for

Until criterion

end loop

Note that the most important change is in the last loop. Instead of probabilistic decision in binary case, displacement is updated by $v_{id}(t)$. There are some implementation issues which can be summarized below:

- initializing the population
- number of particles to use

Initialization of the population is, actually, initializing velocities and positions of the particles. They can be randomly initialized, which is a common approach. The randomness is bounded by $\pm V_max$ for velocity and by the dynamic range of each dimension for the positions. As another option, position can be initialized according to initial guesses.

Number of particles to use depends on practical facts like the properties of the problem and computational efficiency. For instance, while Kennedy prefers 10 and Eberhart prefers 50 particles, we used 5-7 particles.

With a final look on terms of Eq. (3.9), we comprehend that

- $\vec{v_i}$ is the inertia term. It is often multiplied by an inertia weight, ω.
- $\phi_1(\vec{p_i} - \vec{x_i})$ is the cognitive component. It is the personal part of evaluation, comparison and imitation.
- $\phi_2(\vec{p_g} - \vec{x_i})$ is the social component. Mind of a particle becomes social and culture spreads throughout the swarm via this term. How it spreads depends on the topology of neighbourhood.

Actually, there are various PSO algorithms produced by different approaches to those three terms, different parameter selections, neighbourhood topologies and some other aspects. I also use a slightly modified version of the original PSO algorithm, described up to this point. Variations of PSO algorithms are investigated in more detail in the following chapter.

3.4 Variations of PSO

There have been numerous contributions to PSO algorithm from by engineers, mathematicians, physicists, biochemists and psychologists. Either by changing parameters or investigating adaptive sytems, they aim to overcome a corresponding shortcoming of the PSO algorithm. Also, applications and implementations of the algorithm revealed surprising improvements. I will not go into details of all improvements but rather focus on the ones I used.

3.4.1 Velocity Clamping

Velocity is updated in a stochastic way by velocity update equation given in Eq. (3.9). Thus, it can go beyond functional range unintentionally, if the velocity grows excessively. Thus, Eberhart and Kennedy introduced velocity clamping into PSO algorithm [41]. Generally, the following constraint is implemented:

$$\text{if } v_{id} > V_{max} \text{ then } v_{id} = V_{max}$$

$$\text{else if } v_{id} < -V_{max} \text{ then } v_{id} = -V_{max}$$

In order to clarify the effect of V_{max}, Kennedy simplified the algorithm [42] by reducing the dimensionality to 1.0 and making the weighted best point, p, static:

$$p = \frac{\phi_1 p_i + \phi_1 p_g}{\phi_1 + \phi_2} \tag{3.12}$$

Thus the simplified formula becomes

$$v = v + \phi(p - x)x = x + v \tag{3.13}$$

where $\phi = \phi_1 + \phi_2$. Note that vector signs are also dropped since the dimension is only one. Experiments show that when v is not clamped, it increases dramatically, beyond the region of interest. However, when v is clamped, it moves in a useful region.

Thus, appropriate choose of V_{max} makes particle search in a useful region and prevents explosion. Sometimes V_{max} is chosen as the upper range of the search space. However, such V_{max} can be problematic for some problems since it assumes that center of the search space lies at origin of Euclidean space. For instance, if the range is [-100, 100] this assumption is valid. However, search space could be [100,300] and explosion would be inevitable. V_{max} must be chosen according to range of the problem such that it must be large enough to make particles escape from local optima and must be small enough to find the best solution. The n dimensional search space can be defined as

$$\Omega = \left[{x_1}^L, {x_1}^U\right[\times \left[{x_2}^L, {x_2}^U\right[\times ... \left[{x_n}^L, {x_n}^U\right[\subset \mathbf{R}^n$$

where ${x_d}^L$ and ${x_d}^U$ are the lower and upper bounds of the search space,$\mathbf{R}^n$. Then ${V_{max}}^d$, which is the maximum velocity for the particle in dth dimension, as a function of range of search space becomes

$${V_{max}}^d = \lambda \text{range}_d(\Omega)$$

$$\lambda \in (0, 1]$$

$$\text{range}_d(\Omega) = {x_d}^U - {x_d}^L$$

$$\text{for } d = 1, 2, ..., n$$

λ is generally used as 0.5 (fifty percent), it is not optimized by theory yet. For instance, Liu et al suggested fifteen percent [43] for λ. Success of fifteen percent is shown empirically.

3.4.2 Control Parameter

The control parameter, ϕ, determines the trajectory of the particle. Actually, the first mathematical analysis about the trajectory of a particle is published by Ozcan and Mohan [44], without taking V_{max} into consideration. They analyzed one-dimensional, nonrandom particle with constant p in Eq. (3.12) and concluded that particles 'surf' in the space looking for another wave to carry them to the best point, instead of 'flying' as was inspired by bird flocks [10].

ϕ is also known as "acceleration constant". If $\phi = 0.0$, $v = v + 0$ and $x = x + v$ which increases linearly. On the other hand, if ϕ is very small, like 0.01, x increases and decreases slowly. It seems like a sine wave whose frequency increases if ϕ increases and vice versa. Also, when ϕ is at a moderate level, like 1 or 10, the movement of particles looks random. However, increasing ϕ up to large values, like 100, prevents the particle to search the whole space but makes it visit the same points over and over because V_{max} prevent explosion. Obviously, without V_{max} and a large ϕ, particles explode quickly and inevitably. This is similar to stretching the strings of a guitar: The more you stretch the strings (increase ϕ), their oscillation increases (particles change directions faster). At a moderate level you hear the best sound (particles can move randomly). If you stretch it more and more (very large values of ϕ), they broke (explosion).

3.4.3 Constriction Factor

In order to answer the question of how to control explosion Clerc and Kennedy [45] introduced constriction coefficients. They proposed a generic case in which there are numerous ways to control explosion and convergence. They studied the

following deterministic system by defining $y_t = p - x_t$ where t is the time index.

$$\begin{cases} v_{t+1} = v_t + \varphi y_t \\ y_{t+1} = -v_t + (1-\varphi) y_t \end{cases}$$

The matrix representation of the current state becomes

$$P_t = \begin{bmatrix} v_t y_y \end{bmatrix}$$

and

$$M = \begin{bmatrix} 1 & \varphi \\ -1 & 1-\varphi \end{bmatrix}$$

is matrix of the system. Those matrices are used to provide a generalized definition to the system. Since $P_{t+1} = MP_t$, or $P_t = M^t P_0$, the system is completely defined by M. Clerc also introduced the following generalized particle swarm model, which provides numerous ways to control explosion and convergence

$$\begin{cases} v_{t+1} = \alpha v_t + \beta \varphi y_t \\ y_{t+1} = -\gamma v_t + (\delta - \eta\varphi) y_t \end{cases}$$

where α, β, γ and η can be adjusted to control explosion and convergence. One of the ways of controlling explosion is Clerc's simplest constriction coefficient, Type 1". It determines a system similar to Eq. (3.13)

$$v(t) = \chi(v(t-1) + \varphi(p - x(t-1))) \tag{3.14}$$

$$x(t) = x(t-1) + v(t) \tag{3.15}$$

where χ is the constriction coefficient which can be calculated as

$$\chi = \frac{2\kappa}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|} \text{ if } \varphi > 4.0 \tag{3.16}$$

where $\kappa \in [0, 1]$. Constriction coefficient is not determined for $\varphi \leq 4.0$ in Type 1". Assume $\kappa = 0$ then, as can be seen in Eq. (3.15), particle does not search. Obviously κ has a damping effect on velocity. On the other hand, as κ grows larger and larger, that damping effect reduces. Thus the particle becomes more exploratory. However, κ is not the only factor that affects exploratory behaviour

of the particle.

Remembering p in Eq. (3.15) is stochastically weighted average of p_i and pg and their distance affects trajectory of the particle significantly. For instance if they are very close to each other, particle travels around them, which is possibly a small part of the search space. On the other hand, if p_i and pg are far from each other, particle travels the search space. In other words, when neighbours' best are in a different region than global best, there is something wrong and particle keeps on searching until everything seems correct. That is a slow convergence but the particle is sure that everything is as it should be. The more members go to a specific region, the more narrower trajectories has the particle. Still, particles can turn back into exploratory mode if a neighbour finds a new optimum in a different region.

Clerc's model does not explain interactions among particles. Advantages of constriction coefficients are compared with ones of velocity clamping [45] and inertia weight [46], occasionally. However, it has no greater advantage over them.

3.4.4 Inertia Weight

Inertia weight, ω, is one of the earliest contributions to PSO algorithm. It is introduced by Shi and Eberhart [47] as one of the methods to improve rate of convergence. It results the following velocity update equation:

$$\vec{v_i}(t) = \omega\vec{v_i}(t-1) + \varphi_1(\vec{p_i} - \vec{x_i}(t-1)) + \varphi_2(\vec{p_g} - \vec{x_i}(t-1)) \tag{3.17}$$

ω weighs the influence of previous velocity on the current velocity. Inertia weight forces the particle to save its previous direction. It is like the effect of a person's prejudice or intention on a subject. For instance, consider a person walking on road A. When he comes to a crossroad among roads A and B, if his inertia

weight is high, he is more likely to keep walking on road A, even if there are some positive information about road B. On the other hand, if his inertia is low and there are some information indicating road B is better, he is more likely to choose B. Another way of explaining effect of inertia weight can be simply setting $\phi_1 = \phi_2 = 0$ in Eq. (4.23). When ω is more than 1.0, velocity increases up to V_{max} and remains constant there. On the other hand, when ω is less than 1.0, the particle will slow down until its velocity becomes 0.

Inertia weight is shown to be effective in [0,1.4] or can be used in a time-varying manner [42]. Shi and Eberhart's results show that choosing $\omega \in [0.8, 1.2]$ improves rate of convergence, whereas choosing higher ω causes failures to converge. It is preferable to set ω close to 1.0. Also, ω in a decreasing manner can be advantageous, too. Shi and Eberhart made experiments investigating the relation among V_{max} and inertia weight in [42] and effects of ω for various functions in [47]. The results show that optimal ω in first one is 0.8 but the best results in both experiments are observed with a time varying ω, decreasing from 0.9 to 0.4. Such a choice of inertia weight provides a leery characteristic to the particles at the beginning, since they do not fly unbounded. While iterations are being finalized, particles can fly more boldly and reassure the optimal solution they found.

Shi and Eberhart recently introduced a fuzzy inertia weight approach [48] which is adapting inertia weight dynamically using a fuzzy controller. Though it is a promising technique, there are some implementation difficulties since properties of fuzzy controller is hard to define.

3.4.5 Neighbourhood Topologies

In the previous chapter, effects of being social on finding best solutions is discussed. Communication of individuals helps spreading culture thorough out in a society. Thus, mind is *social*, instead of being *isolated* [39]. Similar to the social case, article swarm algorithm is designed in a way that decisions (or movements) of a particle is influenced by its k nearest neighbours and/or the best-performing particle in the swarm. It is shown that *isolated* particles perform very poorly, compared to the social ones. At this point, it is beneficial to remember to main type of neighbourhoods for the sake of completeness: *gbest* and *lbest*.

gbest Neighbourhood

The *gbest* contains a *global best particle* that attracts all others towards itself. If global best particle is not updated in a way to find the best solution, the swarm may converge prematurely. The update equations are the same with Eq. (3.9) which is provided here for completeness:

$$\vec{v_i}(t) = \vec{v_i}(t-1) + \varphi_1(\vec{p_i} - \vec{x_i}(t-1)) + \varphi_2(\vec{p_g} - \vec{x_i}(t-1)) \tag{3.18}$$

$$\vec{x_i}(t) = \vec{x_i}(t-1) + \vec{v_i}(t) \tag{3.19}$$

Note that in this case, all dimensions of $\vec{p_g}$ consists of the the global best particle's repeated values. This one is the Wheel topology, which is shown in Figure 3.1.

lbest Neighbourhood

The *lbest* neighbourhood provides multiple attraction points in order to prevent premature convergence. In this topology, particles have a neighbourhood and they are only affected by their neighbours' performances. Figure 3.2 shows *k=2* neighbourhood which is the Circle topology. In Figure 3.2, particle 1 is only influenced by particles 2 and 7. Similarly, particle 2 is influenced by 1 and 3 and

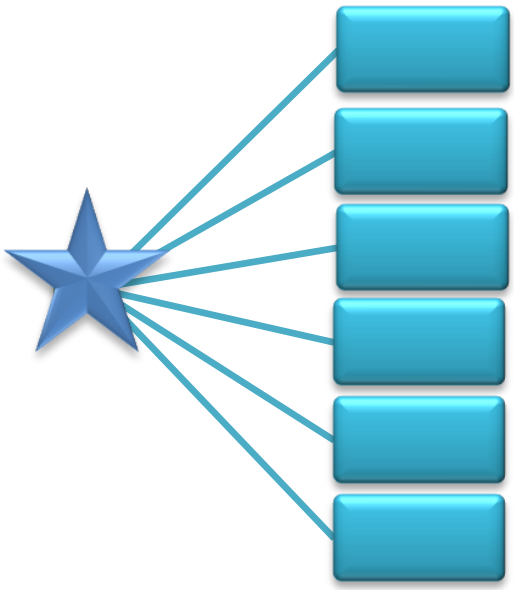


Figure 3.1: gbest

so on. *lbest* neighbourhood provides the velocity and position update equations in Eq. (3.9) but this time, ith element of $\vec{p_{gi}}$ is the best-performing particle in the neighbourhood of ith particle. Note that the particles are connected according to their indexes, instead of their places in the search space. Defining neighbourhoods in the index space is computationally inexpensive since it does not require clustering and enables spreading of information throughout the swarm objectively, regardless of the position of the particle in the search space.

Performances of Various Neighbourhood Topologies

"Small world" phenomenon in sociology indicates that a person shares information with a large number of people unintentionally. Researches by Milgram [] showed that people in United States are only 5 people apart. In other words, two randomly selected people can find each other with a small number of other in between. Moreover, Watts and Strogetz showed that changing some parts of the ring topology randomly, can decrease the average path length. Taking those sociological researches into consideration, Kennedy developed alternative topologies that could affect the information flow [49]. Kennedy proposed the

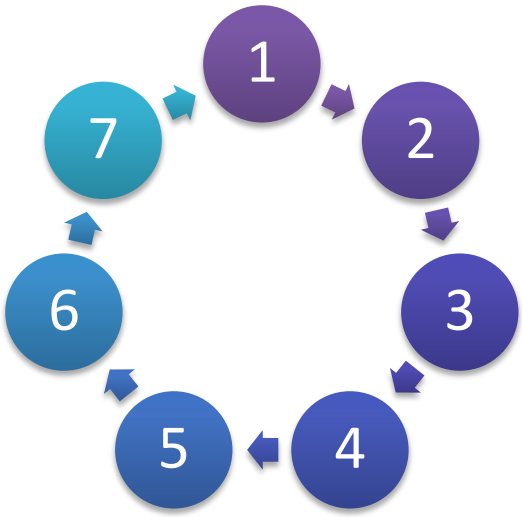


Figure 3.2: lbest

Wheel and Circle topologies and their versions with randomly changed edges. Note that Wheel topology corresponds to *gbest* neighbourhood and the Circle topology corresponds to *lbest* neighbourhood. Kennedy showed that topology significantly affects performance of the optimizer in the following manners:

- The Circle configuration was more powerful when functions with many local minima are used because *gbest* neighbourhood is trapped to local minima. Kennedy explains that case with the slower rate of spreading of information in *lbest* neighbourhood. On the contrary, information spreads faster in *gbest* neighbourhood and many particles are likely to be the best performer.

- The Wheel structure performs better with easier functions, preferably unimodal ones. Though fast transmission of information is a problem with functions with many local optima, it is advantageous for unimodal functions since all other particles follow the leader to the best solution.

Note that, as well as the topology, testing function too affects performance of the optimizer.

Chapter 4

COMBINED ICA-PSO ALGORITHM

Idea of combining two useful algorithms, ICA and PSO, attracted researchers from various fields including chemistry and industrial engineering. As well as the fields of researchers, their methods of using ICA and PSO varies. For instance, a vast majority of researchers work with objective functions based on mutual information whereas the ones based on maximization of negentropy are used by some researchers as as we did in this work.

ICA and PSO are not brand new methods but combining them is a relatively new method. I would like to provide a brief survey on ICA-PSO focused on clarifying similarities and dissimilarities of proposed algorithms. Then I would like to continue with explaining our modifications on ICA and PSO to combine them in a real-time implementable way.

4.1 Survey on ICA-PSO

The first paper combining ICA and PSO is published by Krusienski and Jenkins [50] in March 2005. They used mutual information approach and implemented a batch-mode algorithm, i.e. they estimate all independent component simultaneously. Since minimizing mutual information requires optimizing non-linear performance functions, gradient algorithms provide suboptimal solutions and require multiple restarts. Krusienski and Jenkins states that another difficulty for gradient based algorithms occurs when sources have multimodal distributions because number of local minima on performance surface increases. As mentioned in the previous chapters (2), assuming a linear instantaneous mixing model, $\mathbf{x} = \mathbf{As}$ of N independent components, nonparametric density estimation they use the following objective function

$$L(W) = -\frac{1}{M}\sum_{i=1}^{N}\sum_{k=1}^{M}\left[\frac{1}{Mh}\sum_{m=1}^{M}\varphi\left(\frac{w_i\left(x^{(k)} - x^{(m)}\right)}{h}\right)\right] - \log|\det W| \qquad (4.1)$$

where $\varphi(.)$ is the Gaussian kernel, h is the kernel bandwidth and $x^{(m)}$ is the m^{th} columns of mixture x.

In addition, Krusienski and Jenkins use a modified PSO in which inertia weight is adaptive

$$\omega_i(n) = \frac{1}{1 + \exp^{\frac{-\delta J_1(n)}{S}}} \qquad (4.2)$$

where $\omega_i(n)$ is the inertia weight of i^{th} particle, $\delta J_i(n)$ is the change in particle fitness between current and last generation and S is a constant to control the transition slope. Their experiments showed that stochastic optimization algorithms have a better performance for separating various benchmark functions compared to gradient based optimization algorithms.

In the field of industrial engineering, ICA-PSO is used for fault detection, which is a method to automate industrial inspection. In 2006, Tsai et. al. [51] presented an ICA-PSO algorithm to detect defects on low contrast surfaces like backlight

panels and glass substrates in thin film transistor-liquid crystal (TFT-LCD) displays. Their approach was maximization of negentropy with a constrained ICA model. They used exponential approximation of negentropy as objective function, which was previously provided in Eq.(2.31). They combined ICA algorithm with basic PSO algorithm without any constriction factor. The convolution filter produced by using ICA-PSO algorithm performed better for that certain experimental setup, compared to widely used convolution filters like Wiener filter.

Another application of ICA-PSO was on MEG data analysis by Xie and Wu in 2006 [52]. Their approach was maximization of negentropy with the following approximation

$$G(y) = \frac{1}{a_1} \log \cosh a_1 y \tag{4.3}$$

where $1 \leq a_1 \leq 2$. Similar to the formerly presented work of Tsai et. al., they used a basic PSO algorithm without any constriction factor. However, they obtained better convergence behaviour from gradient methods, so, suggested use of FastICA [8]. after the PSO algorithm. At the end, their experiments showed that ICA-PSO algorithms have better performance since gradient-based algorithms needs re-runs in order to make sure a global optima is found instead of a local one.

In 2009, Nian et al. [38] used an improved version of PSO on ICA in order to process noisy speech signal for speaker recognition (SR) purposes. Though it is possible to obtain very accurate SR results with noiseless speech signals, features related to noise sources decrease the performance of SR algorithms significantly. So, extracting features of noise signals grim speech signals is important. For this purpose, the improved version of PSO takes effects of evolution speed factor, h, and aggregation degree factor, s, of the swarm into consideration. h and s affect the search course of the swarm [53]. So, inertia weight (ω) in velocity update equation in Eq. (4.23) becomes a function of h and s such that

$$\omega_i^t = g(h_i^t, s) \tag{4.4}$$

The idea is to make particles move slow and sure (with a larger inertia weight) at the beginning of searching process and then refine the search results by enabling bold movements providing possibility of escaping from local extrema (a smaller inertia weight). $g(h_i^t, s)$ must be a function such that inertia weight must be decreased by h and increased by s. When possibility of finding a good position increases, it is better to slow down and search around instead of hurrying to the next position. On the other hand, such a movement increases aggregation factor around possible good solutions and that may result trapping into local optima. Finally,

$$\omega_i^t = \omega_{ini} - \alpha(1 - h_i^t) + \beta s \tag{4.5}$$

is introduced where ω_{ini} is initial weight of inertia, $h, s \in [0, 1]$ and choice of α and β is typically within the range [0,1], too. This method is called *dynamic inertia weight* PSO (DPSO).

Nian et al. adopted their objective function from mutual information approach in ICA. In their first experiment, kurtosis of original and reconstructed signals are compared as a measure of quality. They showed that DPSO-ICA performed better with less convergence steps. In their second experiment they compared accuracy of extracted speech features in noisy environments that contains either car noise or babble noise. They showed that DPSO-ICA again performed better than FastICA.

Another study on PSO-ICA was on extracting image features for indoor surveillance by Tsai and Lai [54]. Starting from approximation of negentropy, they reach a non-differentiable objective function. So, they introduce PSO to find optima of the objective function. They search for the optima of a 2x2 demixing matrix, assuming that particles fly in a 4-D space. They use the basic velocity update equation provided in Eq. (4.23) wtih $\omega = 1$. Their results show that PSO search process converges faster over 150 iterations. Their ICA-PSO algorithm can recover highly correlated signals, instead of independent ones. On

the contrary, FastICA algorithm using approximations of negentropy recovers independent signals faster. If ICA-PSO is run for a large number of iterations, results resemble to the ones of FastICA, though.

In 2010, Zhang and Zhang used another version of ICA-PSO for fault detection of non-gaussian processes [55]. They used maximization of negentropy via its non-linear approximations. They combined it with a PSO algorithm that has a slight variation on weight of inertia, ω

$$\omega(t) = \omega_{max} - \frac{(\omega_{max} - \omega_{min}) \times t}{t_{max}} \tag{4.6}$$

where t is iteration number, ω_{max} is initial value of ω and ω_{min} is the final value of ω. Generally, ω_{max} is 0.9 and ω_{min} is in [0.3, 0.4]. This method depends on the same idea explained for Nian et. al. which provides slower but sure convergence in the beginning of searching process and ability to jump out of local optima near the end of search process.

Igual et al. introduced a PSO-ICA algorithm that performs better than FastICA where more than one source signals have gaussian-like distributions [56]. They used Clerc's constriction method for PSO and mutual information approach of ICA. They showed that PSO managed to converge in some cases where gradient-based optimization algorithms failed.

4.2 ICA-PSO Algorithm

The algorithm introduced in this work has some modifications on the methods proposed in chapters 2 and 3. On ICA, negentropy based objective functions are used but they are slightly modified such that peakiness of signal's distribution is the main concern. On PSO, swarm size is made extremely small, effect of *global best* is emphasized (Sec. 3.4.2) and effect of *inertia* is suppressed (Sec. 3.4.4).

Those modifications are to overcome handicaps of ICA and PSO efficiently. Details of modifications are provided in the following sections.

4.2.1 Modifications on ICA

The negentropy based objective function is modified so that peakiness of distribution is measured and effect of its tails are suppressed. In order to clarify that modification, I would like to re-visit approximations of negentropy, which is previously told in Sec. 2.2.3, with more detail.

Recall that in ICA, our aim is to measure non-gaussianity. Negentropy is a robust measure of nongaussianity, both theoretically and practically. As was told in Sec. 2.2.3, negentropy is an information theoretical concept, which is defined as

$$J(\mathbf{y}) = H(\mathbf{y_{gauss}}) - H(\mathbf{y}) \tag{4.7}$$

where $\mathbf{y_{gauss}}$ has the same covariance and mean matrix with $\mathbf{y}$. Also negentropy is zero for a gaussian distributed random variable and non-zero for other distributions (Sec. 2.2.3). If distribution of y is unknown, it is computationally very hard to calculate negentropy. So, approximations of negentropy are used.

There are two main approaches to approximate negentropy:

- Cumulants
- Non-polynomial moments

The first approach is based on using expansions like Taylor expansion which is taken for the pdf of random variable. Assuming pdf of a zero-mean and unit variance random variable, $p_x(\xi)$ is near the standardized gaussian density allows us to make Taylor-like expansions

$$\varphi(\xi) = \frac{\exp(-\xi^2)}{\sqrt{2\pi}} \tag{4.8}$$

Using Gram-Charlier expansions we reach Chebyshev-Hermite polynomials, which are derivatives of $\varphi(\xi)$

$$\frac{\partial^i \varphi(\xi)}{\partial \xi^i} = (-1)^i H_i(\xi)\varphi(\xi) \tag{4.9}$$

where H_i indicates Chebyshev-Hermite polynomials an i is a non-negative index. These polynomials form an orthonormal system and Gram-Charlier expansion of $p_x(\xi)$ becomes

$$p_x(\xi) \approx \hat{p}_x(\xi) = \varphi(\xi)(1 + \kappa_3(x)\frac{H_3(\xi)}{3!} + \kappa_4(x)\frac{H_4(\xi)}{4!}) \tag{4.10}$$

where $\kappa_3(x)$ and $\kappa_4(x)$ were defined in Eq. (2.24). Using $\hat{p}_x(\xi)$ in definition of entropy

$$H(x) \approx -\int \hat{p}_x(\xi)\log \hat{p}_x(\xi)d\xi \tag{4.11}$$

Using mathematical manipulations, approximations and Eqn. 4.10 we reach to

$$H(x) \approx -\int \varphi(\xi)\log \varphi(\xi)d\xi - \frac{\kappa_3(x)^2}{2\times 3!} + \frac{\kappa_4(x)^2}{2\times 4!}) \tag{4.12}$$

and

$$J(x) \approx \frac{1}{12}E\{x^3\}^2 + \frac{1}{48}kurt(x)^2 \tag{4.13}$$

So, we found the cumulant based approximation in Eq.(2.28). Since high order moments are calculated in this approximation, it is more sensitive to tails of the distribution than its center values.

As seen in Fig. (4.1), tails of distribution has a greater effect compared to its center values. This kind of approximation of negentropy can be very sensitive to outliers, which are rarely or accidentally observed values. In addition, finite number of samples are considered here. That causes negentropy approximations be more incorrect.

More robust approximations of negentropy can be obtained via approximating maximum entropy. Using linearly independent functions F^i and again making

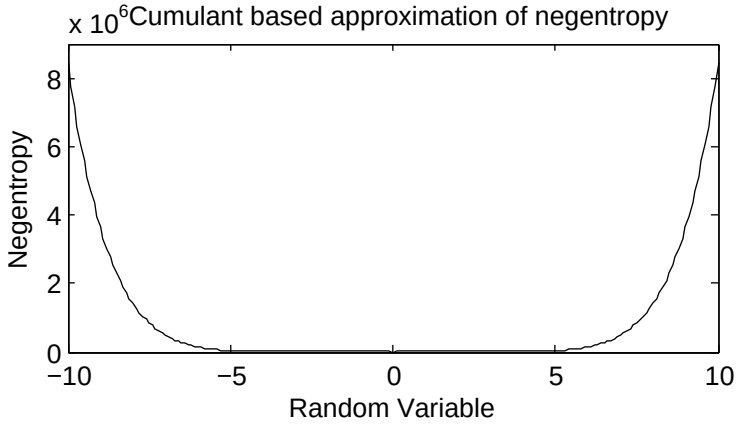


Figure 4.1: The cumulant based approximation of negentropy. It emphasizes importance of tails of distribution

the assumption that distribution is near gaussian one and some additional mathematical manipulations ([9]) we can find *approximative maximum entropy density*, $\hat{p}(\xi)$

$$\hat{p}(\xi) = \varphi(\xi)(1 + \sum_{i=1}^{n} c_i F^i(\xi)) \tag{4.14}$$

where $c_i = E\{F^i\}$ and φ is the metric used to define an orthonormal system of F^i. Using Eq.(4.11) and some algebraic manipulations

$$J(x) \approx \frac{1}{2} \sum_{i=1}^{n} E\{F^i\}^2 \tag{4.15}$$

This approximation shows that negentropy can be approximated using *non-polynomial moments*. The question is to choose proper F^i.

Choosing a set of linearly independent functions (G^i) and applying Gram-Schmidt orthonormalization to that set can provide F^i. There are 3 criteria for choosing proper G^i:

1. $E\{G^i\}$ must be insensitive to outliers.

2. G^i must grow slower than quadratically according to theory of maximum entropy distributions

3. G^i must contain the source signal's statistical properties related to entropy. For instance, if $p_x(\xi)$ were known, G^i would be $\log p_x(\xi)$. So that $E\{G^i\}$ would be exactly entropy of $p_x(\xi)$.

It is possible to use a set of G^i or a single G to approximate negentropy. According to the property desired to be measured, choice of G^i varies. For instance choosing an odd G^1 and even G^2, it is possible to measure skewness and kurtosis, respectively, by making $G^1 = x^3$ and $G^2 = x^4$. So, the resulting approximation resembles to the cumulant-based ones. Note that Eq.(4.15) turns into

$$J(x) \approx k_1(E\{G^1(x)\})^2 + k_2(E\{G^2(x)\} - E\{G^2(v)\}) \tag{4.16}$$

where k_1 and k_2 are positive constants. For measuring sparsity following choices can be useful ([9])

$$G^2(x) = |x| \tag{4.17}$$

$$G^2(x) = \exp(\frac{-x^2}{2}) \tag{4.18}$$

where the second function is smoother and more useful. For measuring asymmetry it is possible to use

$$G^1(x) = x\exp(\frac{-x^2}{2}) \tag{4.19}$$

Figure 4.2 explains the relation between functions and the properties they measure. Since (a) has larger values around center of distribution, it can be used to measure peakiness since values of x around center are more important. For instance, a super-gaussian distribution would have large values whereas a sub-gaussian one would have smaller ones when $G^2(x)$ is used as an approximation. On the contrary, a super-gaussian distribution would have a smaller values compared to the sub-gaussian one's values, if we use (c). Using (b), it is possible to measure amount of shifting around center of distribution.

Sometimes G^1 and G^2 are used together to measure both properties of distributions. Some useful choices were shown in Eq. (2.31). On the other hand, it is possible to use only one of them. If some information about signals is known,

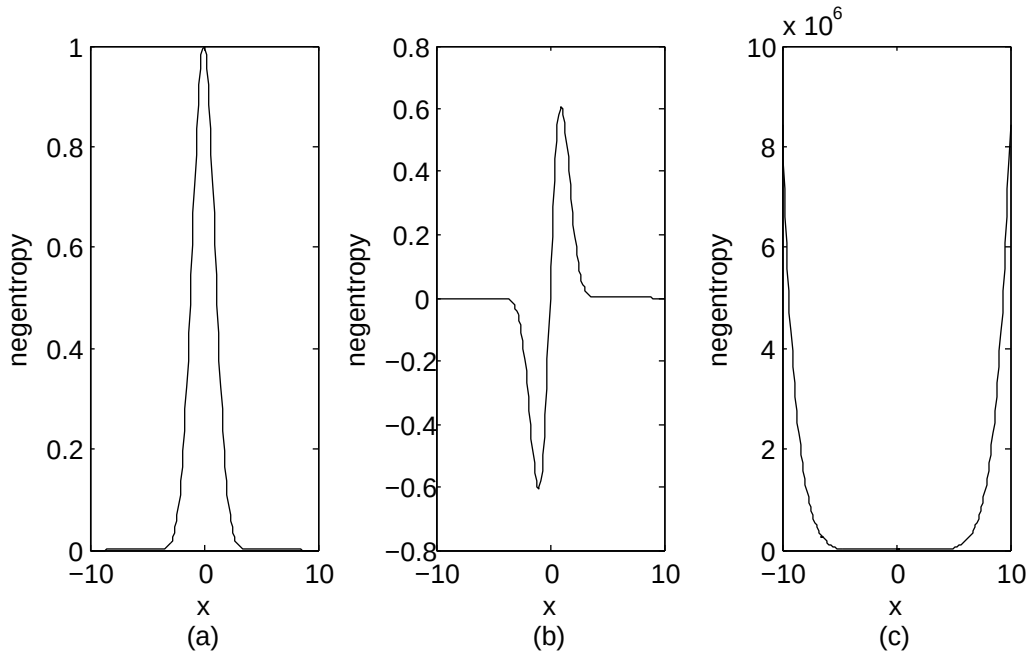


Figure 4.2: (a) $G^2(x)$ measuring peakiness, (b) $G^1(x)$ measuring bimodality, (c) Cumulant based approximation in Eq. (4.13) measuring tails of distribution

the function to be used can be chosen effectively.

We know that one of our signals is speech and its distribution resembles Laplace distribution, which is peaky. In addition, background noise may be gaussian whose distribution has heavier tails. Those distributions are shown in the Figure 4.3. In Figure 4.3, speech and noise samples are taken from SISEC 2010 database. So, it is clever to use an objective function (approximation of negentropy) that focuses on measuring peakiness. So, using G^2 can be useful.

In this case, negentropy is generally approximated as was shown in Eq. (2.31) and provided here for sake of completeness

$$J(x) \propto [E\{-\exp\left(-\frac{x^2}{2}\right)\} - E\{-\exp\left(-\frac{v^2}{2}\right)\}]^2 \tag{4.20}$$

where $G^2(x) = -\exp\left(-\frac{y^2}{2}\right)$ and v is a gaussian random variable who has the same mean covariance matrices with x. Practically, it is both possible and useful to omit v because $E\{G^2(v)\}$ is a constant. Thus, the first modification on this

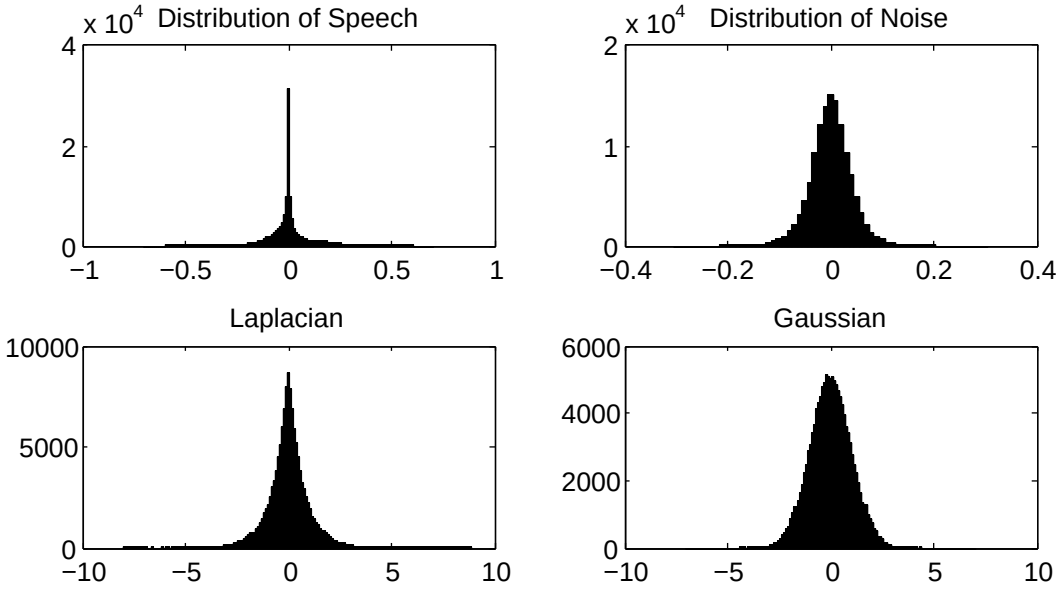


Figure 4.3: Similarities among distributions of speech signal and Laplace distribution, as well as the one among noise signal and gaussian distribution are clear

objective function is omitting $E\{G^2(v)\}$

$$J(x) \propto [E\{\exp\left(-\frac{x^2}{2}\right)\}]^2 \tag{4.21}$$

After that point, squaring is not necessary. So, the objective function becomes the one shown in Figure 4.2 (a) and

$$J(x) \propto [E\{\exp\left(-\frac{x^2}{2}\right)\}] \tag{4.22}$$

Since separating two signals is aimed in this work, decorrelation too is not needed because signals can be obtained from extrema. The most peaky signal can be obtained from maxima and the other one from minima. In addition, since there are less calculations modified objective function is easier to compute compared to the other ones.

Computational efficiency of Eq. (4.22) is important because the overall method of noise cancellation is supposed work real-time. Cancelling gaussian distributed term and squaring and omitting decorrelation step both simplifies the objective function and makes it more efficient. They are useful modifications because the

objective function becomes specialized to separate speech from other signals and its computational cost decreases.

4.2.2 Modifications on PSO

Modifications on PSO are made to reduce computational cost and number of iterations to converge, at the same time. Reducing computational cost is only possible by using the simplest velocity update equation and reducing the swarm size. On the other hand, those changes must not make convergence harder. There is a trade of among computational efficiency and fast convergence.

First of all, neighbourhood topology and thus velocity update equation must be determined. In this work, we preferred *gbest* topology which requires minimum amount of computation and yet performs good enough. The *gbest* topology was shown in Figure (3.1). In this topology, particles know about their own best performance and the best performance in the swarm as was explained in Section 3.4.5.

In velocity update equation, *inertia weight* is used to determine adventurous characteristic of particles such that they are adventurous if they dare to continue looking for optima with the same velocity. In other words, inertia term determines the effect of current velocity on the velocity to be calculated. Combining *gbest* topology with inertia, we get the following velocity update equation for particle i at time t

$$\vec{v_i}(t) = \omega\vec{v_i}(t-1) + \varphi_1(\vec{p_i} - \vec{x_i}(t-1)) + \varphi_2(\vec{p_g} - \vec{x_i}(t-1)) \tag{4.23}$$

That update equation is widely used and generally ω is either close to 1.0 [42] or varies according to a certain rule [47]. However, we choose ω as 0.5 to search the space bravely.

The reasoning of such an approach depends on that we do not wait for all particles to be gathered at optima but we expect that the best point, found by whichever particle, remains constant. Particles can be adventurous and look for optima

bravely. A particle may find optima at one of the iterations and loose it in the next one but that optima can be found by another particle. Comparing the best performances of every iteration (*gbest*), we decide optimal solution. Consequently, *gbest* becomes the most important term.

Importance of *gbest* can be emphasized in velocity update equation by making φ_2 larger than φ_1 in Eqn. (4.23). However, modifications up to this point may trigger premature convergence, which was discussed in Section (3.2). So, a randomization factor is added to velocity update equation to prevent premature convergence

$$\vec{v_i}(t) = \omega\vec{v_i}(t-1) + c_1 r_1(\vec{p_i} - \vec{x_i}(t-1)) + c_2 r_2(\vec{p_g} - \vec{x_i}(t-1)) \tag{4.24}$$

where r_1 and r_2 are random values drawn from the standard uniform distribution on the open interval(0,1)and c_1 and c_2 are used to balance effect of personal and global terms. So, we choose c_2 larger than c_1 to make particle's decisions biased in favour of *gbest*. In the mean time, r_1 and r_2 prevents being convinced so fast and adds suspicion to characteristics of particles by randomly effecting their decisions.

Since we constructed the velocity update equation as in Eq. (B.1), it is time to decide number of particles to fly in search space, i.e. *swarm size*. It must be as small as possible because the larger the number is, the more expensive the computational cost becomes. Eberhart and Shi suggest that swarm size must be at least 10 but we reduced it up to 7 particles. It can be reduced further but after 5 particles, both number of iterations to find optima and error rate increase. So, 7 particles are efficient.

For a very small swarm size which is 7, *gbest* topology is used with an efficient velocity update equation. Velocity update equation became efficient by emphasizing effect of *gbest* and introducing randomization factors at the same time. Choosing a relatively small and constant value for ω in Eqn. (B.1) increased searching capacity of particles. As a result, PSO is combined with ICA in an efficient way such that it is possible to implement algorithm in real-time.

Chapter 5

PITCH EXTRACTION

In this chapter, basics of the pitch extraction algorithm (PE) are addressed. In Section 5.1 some properties of speech signal is provided. In Section 5.2 the idea behind PE is explained in detail.

5.1 Some Properties of Speech Signal

Speech is formed by air forced from the lungs through the vocal cords and along the vocal tract. The vocal tract extends from the opening in the vocal cords (called the glottis) to the mouth and lips. Although there are many possible speech sounds which can be produced, the shape of the vocal tract and its mode of excitation change relatively slowly. Speech sounds' modes of excitation constitute three classes:

- *Voiced sounds* are produced when the vocal cords vibrate open and closed. So, the flow of air from the lungs to the vocal tract is interrupted. As a result, quasi-periodic pulses of air are produced as the excitation. The rate of the opening and closing is the pitch of the sound. Voiced sounds show a

high degree of periodicity at the pitch period, which is typically between 2 and 20 ms. Voiced signals tend to be louder like the vowels /a/, /e/, /i/, /u/, /o/.

- *Unvoiced sounds* occur when the excitation is a noise-like turbulence produced by forcing air at high velocities through a constriction in the vocal tract while the glottis is held open. So, it does not entail the use of the vocal cords. Unvoiced signals tend to be more abrupt like the stop consonants /p/, /t/, /k/.

- *Plosive sounds* result when vocal tract is completely closed and the air pressure behind the closed vocal tract released suddenly.

Some sounds are a mixture of the above classes and do not belong to any of them. Voiced fricatives can be a good example for that case. They occur when both vocal cord vibration and a constriction in the vocal tract are present.

As mentioned above, *pitch-period* is a property of voiced speech originating from its quasi-periodic nature. Though shapes and periods of quasi-periodic signals slowly varies with time, that change is quite slow making them resemble to periodic signals. The length of each consecutive cycle in speech are called *pitch-period*, and one period signal is called the *pitch-cycle*, whose frequency is also known as *fundamental frequency*.

5.2 Pitch Extraction

The pitch-period estimation method proposed in [11] is a time-domain method based on correlation. Since autocorrelation function has the same period with the signal, the lag with maximum correlation corresponds to pitch period of signal.

However, there are some problems causing incorrect estimation of pitch period and finding of voicing state.

First problem is the quasi-periodic nature of speech which is more specifically effective at the end of words. Another problem concerning end of words depends on vocal fry, which is a speech signal observed when sub-glottal pressure is not enough to maintain a quasi-periodic signal. This problem affects finding of voicing state whereas the previous one affects estimation of pitch period. There are also pitch-halving and pitch-doubling problems but since our main aim is not to find exact pitch-period but to distinguish among speech and noise, they are not effective. Those problems based on high correlation at multiples of pitch-period.

Generally, the normalized autocorrelation, ρ_τ at pitch lag τ is calculated as

$$\rho = \frac{\langle x_0, x_\tau \rangle}{\sqrt{\langle x_0, x_0 \rangle \langle x_\tau, x_\tau \rangle}} \tag{5.1}$$

where $\langle x_k, x_l \rangle$ is defined as

$$\langle x_k, x_l \rangle = \sum_{n=0}^{L-1} x[n \pm k] x[n \pm l] \tag{5.2}$$

where $x[n]$ is the analysed signal and L is the number of samples used. In order to reduce estimation errors, the frame of analysed signals can be divided into multiple subframes by pitch estimation algorithms. Using forward and backward correlation in each frame, a pitch track can be obtained.

In this work, main analysis window is twice the size of maximum pitch period. Furthermore, it is divided into two regions to allow correlation which is calculated as

$$\langle x_{\tau_1}, x_{\tau_2} \rangle = \sum_{k=\frac{L}{2}-1}^{L} x[k - \tau_1] x[k - \tau_2] \tag{5.3}$$

where τ_1 and τ_2 are pitch lags. After calculating ρ for each pitch lag from 5 to 20 ms, the pitch lag at which maximum ρ is calculated is chosen as the pitch period.

In order to label the reconstructed signals as speech or noise, maximum ρ of both reconstructed signals are compared. If voiced speech is reconstructed, ρ is close to 1 with one of allowed τ. On the other hand, autocorrelation function of reconstructed noise signal cannot be as high as speech's autocorrelation, unless, background noise is speech, too. If there are two people talking simultaneously, it is not possible to label one of them as noise and the other one as speech.

In addition to Eq. (5.1), energy of signals can be used to differentiate among signals. Since energy of voiced speech is relatively higher than unvoiced one's energy, the distinction between them becomes clearer. Moreover, energy of noise is supposed change in a slower fashion. So, energy of signals can be calculated and used as

$$P(x) = \sum_{k=\frac{L}{2}-1}^{L} x[k]^2 \tag{5.4}$$

$$R = P(x)\rho \tag{5.5}$$

where L is the length of frame. However, using energy of signals requires that energies must be at levels enabling correct comparison. For instance, if energy of noise is very high compared to the energy of speech, it can be labelled as speech.

In Figure 5.1, 10 frames of speech and noise signals, taken from SISEC 2010 database, are shown. Since sampling rate is 16 kHz and maximum allowed pitch lag is 20 ms, frame length (L) is 640 samples. Notice that some frames of speech only contain voiced speech (e.g. 1, 2, 8), some frames contain unvoiced speech (e.g. 5, 10) and some frames contain both parts (e.g. 7, 9). Figure 5.2 shows maximum correlation levels at each frame. There is a significant difference among

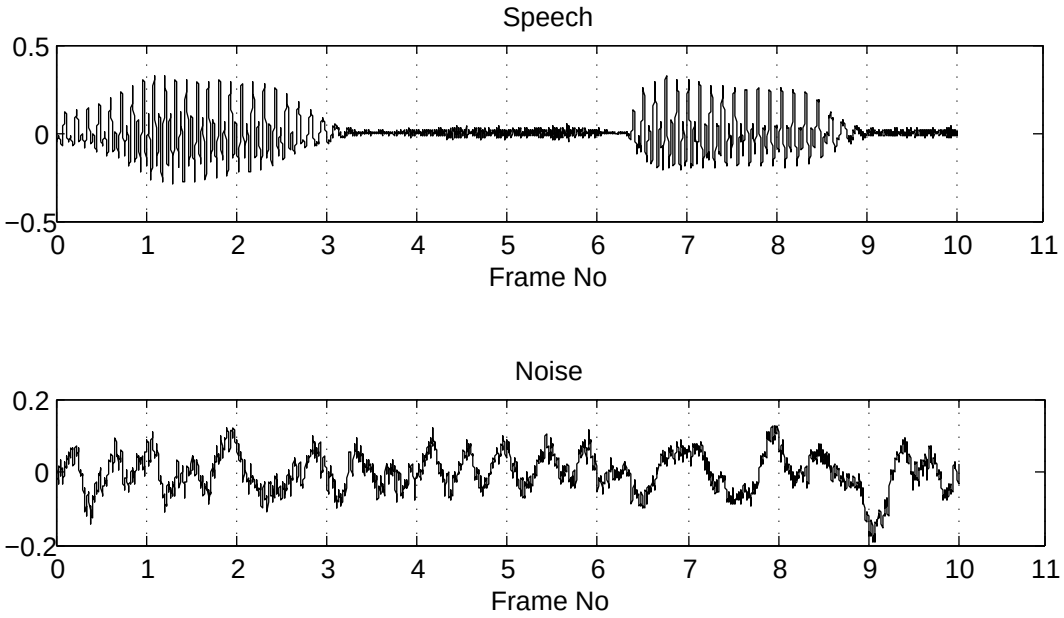


Figure 5.1: 10 frames of speech and noise signals

voiced and unvoiced speech. On the other hand, noise signal has a slowly varying course of autocorrelation.

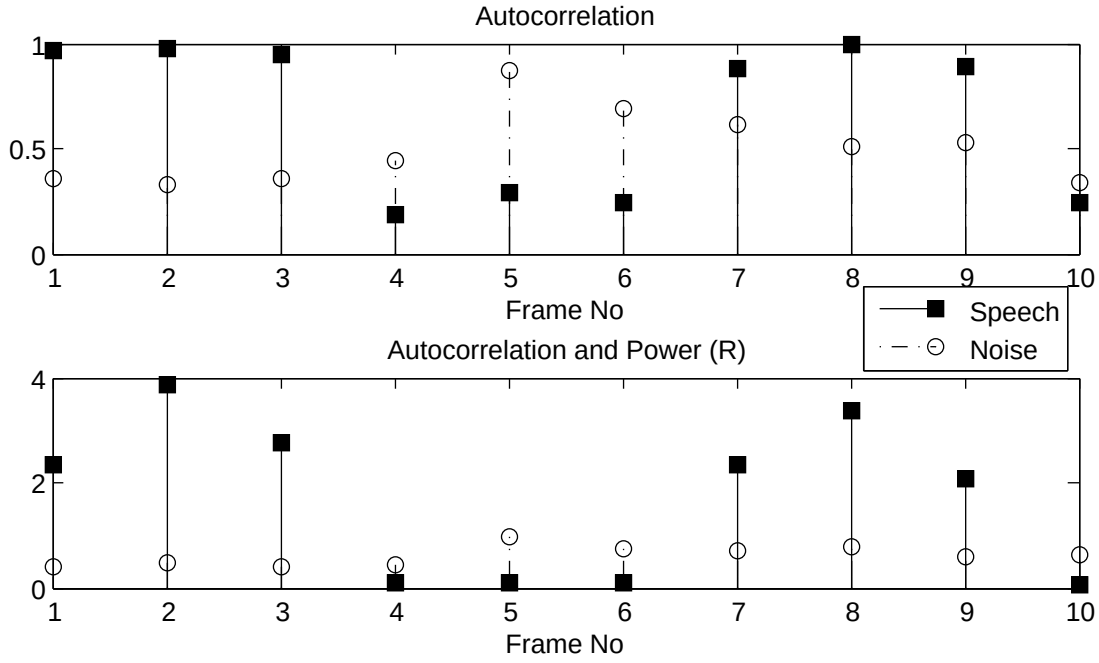


Figure 5.2: Maximum ρ and R values for each frame of speech and noise signals above

Chapter 6

SIMULATIONS AND RESULTS

The methods addressed up to this point constitute a hybrid algorithm for noise cancellation. Each of these methods has a specific function in this algorithm where

- **Independent Component Analysis (ICA)** provides objective functions
- **Particle Swarm Optimization (PSO)** finds extrema of objective functions
- **Pitch Extraction (PE)** labels separated signals either as speech or noise

The overall process is shown in Figure 6.1 basically. The aim is to analyse speech and noise by observing their mixtures only (Chapter (1)). In order to cancel out noise, we are looking for de-mixing directions (Θ) which corresponds to elements of de-mixing matrix, W (Section (2.1)). There are always two sources (noise and speech), two channels (1 and 2) and we are looking for two de-mixing directions ($\theta 1$ and $\theta 2$) in the experiments in this chapter.

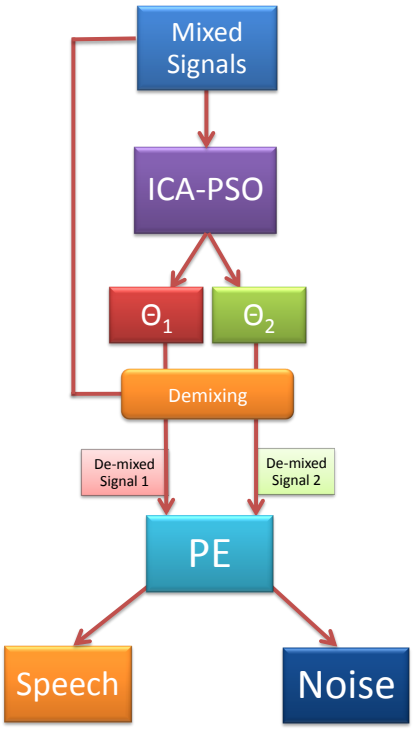


Figure 6.1: Overall System

Algorithm decides for θs for speech and noise at the end of a learning process. During the learning process θs are found for speech and noise in each frame. In order to prevent incorrect decisions on de-mixing directions, θs are collected and grouped as the ones corresponding to speech and the ones corresponding to noise. When enough θs are collected, the most frequent directions are chosen from histograms of each group of θ. Note that because of the ambiguity problem of ICA (Section 2.1.1), it is not possible to know whether a θ corresponds to speech or noise. So, PE is used to group θ.

In Section 6.1, performance of different objective functions of ICA are compared from points of view of accuracy and speed. In the following section, Section 6.2, advantage of using PE is shown. In Sections 6.3, 6.4 and 6.5 experiments on SNR levels, effect of source signals and duration of learning period are presented, respectively. In the last part, Section 6.6, performance of ICA-PSO-PE is compared with a widely used ICA algorithm (FastICA) and a frequently used

noise cancellation method (subtraction method).

Experimental conditions are explained in each section but they do not have significant differences. Generally, data from Signal Separation and Evaluation Campaign (SISEC) 2010 database is used in experiments. The database contains background noise recorded at a plaza, a subway and a cafeteria. There are male and female speakers. Generally, voice of a female speaker with background noise of a plaza (cars passing by, people talking and other noises) is used in experiments since it is thought to be one of the problematic cases. Speech and background noise samples are mixed synthetically with a mixing matrix. Unless otherwise stated, there is no latency among channels.

In order to simulate real-time radio communication system, data is used frame-by-frame. In other words, instead of using the whole 8 seconds of data, it is divided into frames of 40 ms. After the analysis of a frame is completed, the next frame is imported and analysed and so on. For each frame, ICA-PSO-PE algorithm re-starts without adopting information from previous frames. Note that since the information on signals decreases with decreasing number of samples in each frame, the analysis results become more inaccurate if the duration of a frame becomes shorter. Though our experiments showed that it is possible to reduce duration of frame up to 20 ms, pitch extraction algorithm requires a frame size of twice the maximum allowed pitch lag, which corresponds to 40 ms.

In addition to frame-by-frame structure, the objective functions of ICA and all other necessary computations are modified to reduce the cost of computation. Those modifications are explained in detail in Chapter 4.

All the simulations in MATLAB are designed to test the real-time implementable algorithm. Sampling rate is 16 kHz, which results to 640 samples per frame. Note that frames are 40 ms long, which is acceptable since it is not possible to perceive a 40 ms throughput delay by listening. Since the algorithm begins providing results after the first frame and the process on the consecutive frames take less than 40 ms, all the latency is due to the processing of first frame. As a final note, learning period takes 50 frames, which is 2 seconds. In all figures, signal after 2s represents the separated data, except the ones in Section 6.5.

6.1 Performances of Objective Functions

There are many objective functions used by ICA to approximate negentropy. Performance measures for each objective function are accuracy and computational efficiency. Accuracy depends on how correctly θs are found even at very low SNRs and with various sources. Computational efficiency depends on the speed of algorithm with that objective function.

Objective functions are compared with each other from the perspectives mentioned above. In this experiment, various sources of speech and noise are used from SISEC 2010 database. All objective functions are tested under various conditions. The most significant and distinctive results are provided in this section.

The following objective functions are compared with each other

- **Power 4** is a simple and Kurtosis-like function.

$$g(x) = x^4 \tag{6.1}$$

- **Exponential** objective function is the slightly modified version explained in Chapter 4. It is provided here for completeness.

$$g(x) = \exp\left(\frac{x^2}{2}\right) \tag{6.2}$$

- **Hyperbolic cosine** is widely used by ICA-PSO algorithms such as [52]

$$g(x) = \frac{1}{a_1}\log\cosh(a_1 x) \tag{6.3}$$

  where a_1 is chosen to be 1 generally.

- **Hyperbolic tangent** (tanh) is used in FastICA algorithm, released in October 19 2005 and its copyright belongs to Hugo Gävert, Jarmo Hurri, Jaakko Särelä, and Aapo Hyvärinen. Actually, it is the modified version of Eq. 6.3 to use with gradient-based methods. However, it can be used with PSO, too.

$$g(x) = \tanh(x) \tag{6.4}$$

To discuss the performance of the objective functions more clearly, consider their plots given in Figure 6.2. Note that all objective functions, except the exponential one, emphasize the effect of tails of distribution, i.e. values which are away from zero. On the other hand, exponential function focuses on the values around zero (center). When number of values around the center is higher than the number of ones at tails, i.e. when one of the independent components' distribution has a peaky structure and the other one is more flat, exponential objective function can more successfully differentiate them. Hyperbolic tangent (tanh) is supposed to provide accurate results when one of the distributions is shifted to one side of center, i.e., it measures skewness.

In the first experiment, objective functions are tested under very low SNR conditions where $SNR_1 = -10.4650$ and $SNR_2 = -12.2259$. SNR_1 and SNR_2 represent the SNR levels on channels 1 and 2, respectively. Results of the first experiment are presented in Table 6.1. All the objective functions can find θ_1

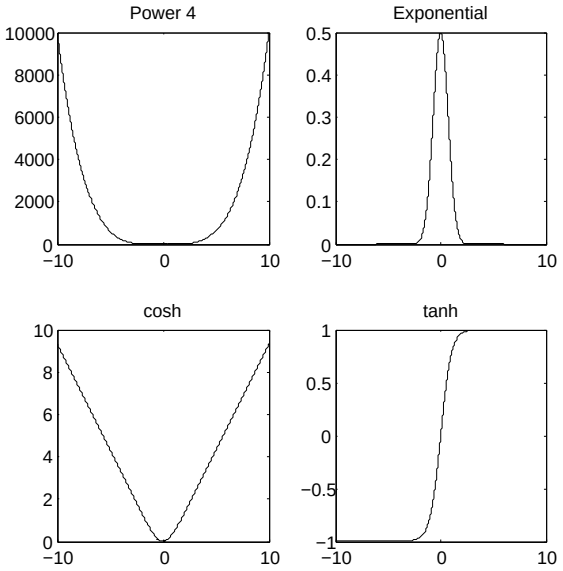


Figure 6.2: Plots of all objective functions in [-10,10]

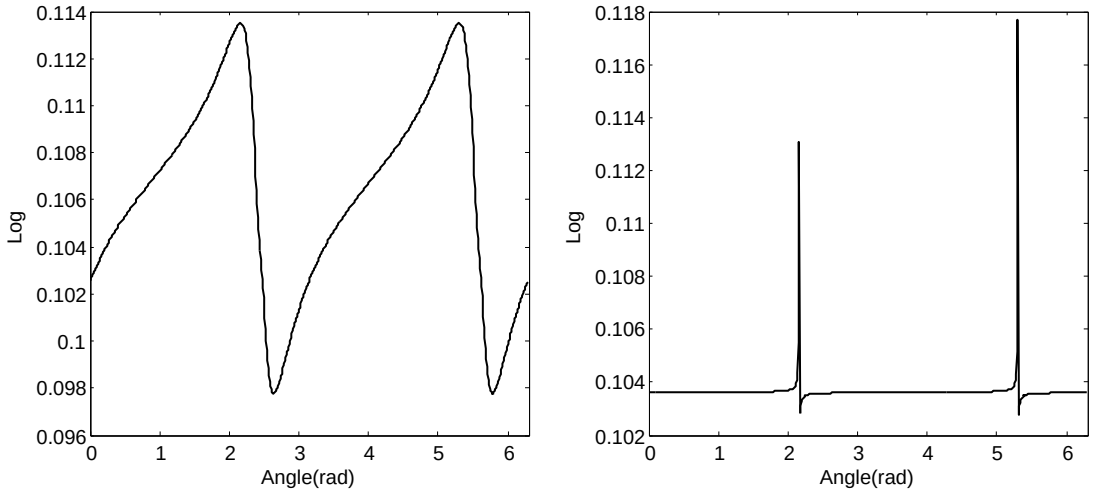
successfully but it is observed that θ_2 is problematic in the sense that none of the functions can find θ_2 accurately. Besides inaccurate results of all other objective functions, hyperbolic cosine finds it the same as the first direction. The reason behind such a behaviour can be understood by comparing Figures 6.3a and 6.3b. These figures show values of objective function, evaluated at every angle from 0 to 2*pi, with the same data used in this experiment. In other words, the data is de-mixed at every possible direction as shown in Equation 6.5

$$\mathbf{y} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \end{bmatrix} * \mathbf{X} \tag{6.5}$$

where $\mathbf{X}$ is the observed signal and $\mathbf{y}$ is the de-mixed signal. Note that θ is in [0, 2π] and each y becomes the input of the objective function to obtain the plots in Figures 6.3a and 6.3b. In Figure 6.3a SNR level is high and a clear distinction between directions, i.e. maximum and minimum points of objective function, can be made. On the other hand, in Figure 6.3b, SNR level is very low, objective function became very sharp that there is no difference among maximum and minimum points. So, when SNR is very low, both directions converge to one of them, explaining the result of hyperbolic cosine in Table 6.1.

Objective function	θ_1	θ_2
Power 4	-0.7854	-0.8761
Exponential	-0.7854	-0.9151
Hyperbolic cosine	-0.7854	-0.7854
Hyperbolic tangent	-0.7854	-0.8330

Table 6.1: Performance of objective functions at a low SNR level. SNR levels are $SNR_1 = -10.4650$ and $SNR_2 = -12.2259$. Theoretical θs are $\theta_1 = -0.7854$ and $\theta_2 = -0.9828$



(a) High SNR (b) Low SNR

Figure 6.3: Examples of changing behaviour of an objective function under high and low SNR conditions. In this example, hyperbolic cosine objective function is used but such behaviour is valid for all objective functions, only at different SNR levels.

In the second experiment, a very hard to analyse data set is used. Since noise of cafeteria includes voices of other people, statistical structures of speech and noise signals resemble. In addition, it is hard for PE to label data. Thus, recordings at cafeteria is a challenging data set. As a result, most of the objective functions fail at high SNR levels. In addition to possible lowest SNR level that the signals can be separated, total time required until the end of learning period is important. Results of this experiment are shown in Table 6.2.

Objective function	θ_1	θ_2	$\mathbf{SNR_1}$	$\mathbf{SNR_2}$	**Time (s)**
Power 4	-0.7854	-0.9828	1.8546	0.0937	0.9463
Exponential	-0.7854	-0.9151	0.8856	-0.8753	0.7149
Hyperbolic cosine	-0.7854	-0.7854	9.4243	7.6634	1.0352
Hyperbolic tangent	-0.8330	-0.9828	4.8650	3.1040	0.9228

Table 6.2: Performance of objective functions with challenging source signals. SNR_1 and SNR_2 are lowest possible SNR levels that separation is accurate. Theoretical θs are $\theta_1 = -0.7854$ and $\theta_2 = -0.9828$

Time in Table 6.2 is not numerically important because the time required complete a task depends on properties of CPUs and development environments. However, it can be used to compare speed of objective functions because they were run in the same environment.

SNR levels at Table 6.2 indicate the lowest possible SNR level at which successful separation is observed. Results in Table 6.2 show that exponential objective function is optimum when SNR levels and time is a concern. On the other hand, hyperbolic tangent performs better at low SNR. Also note that cosh fails even at a high SNR level.

When both experiments are considered, exponential and hyperbolic tangent objective functions seem to be the best performers under low SNR conditions and with challenging source signals. If there is information on statistical structures about source signals, objective function can be chosen with the consideration of that information.

6.2 Benefit of PE

A pitch extraction algorithm is introduced to label reconstructed data either as speech or noise. Thus, it is possible to group θ as the ones belonging to speech

and the ones belonging to noise. Otherwise, due to ambiguity problem, θ cannot be grouped and histograms become scattered.

In order to obtain more unscattered histograms, tangents of angles, instead of angles themselves, are plotted. Otherwise different angles pointing at the same direction cause a scattered histogram.

The following figures show benefit of using PE such that cumulation at correct θ is larger in simulations with PE. In these experiments a women is speaking and background noise is recorded in a plaza. Those data are taken from SISEC 2010 database. The source signals used in this experiment can be seen in Figure 6.9b where the speaker is "female speaker 2" in Section 6.4. Mixing matrix and corresponding demixing directions are

$$\mathbf{A} = \begin{bmatrix} .9 & .5 \\ .6 & .5 \end{bmatrix} \qquad and \qquad \mathbf{\Theta} = \begin{bmatrix} -0.7854 \\ -0.9828 \end{bmatrix} \tag{6.6}$$

where $\tan(\theta_1) = -1.0$ and $\tan(\theta_1) = -1.5$. Experimental conditions are the same with the ones in Section 6.10. In addition, noise signal is enhanced 5 times in Figures 6.4, and 15 times in Figure 6.5. SNR_1 and SNR_2 indicates SNR levels at channels 1 and 2, respectively. For channel i SNR is calculated as

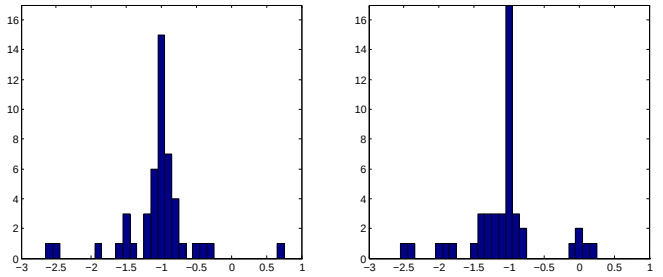
$$SNR_i = 10 \log \left| \frac{P(\alpha_{i1} s_1)}{P(\alpha_{i2} s_2))} \right| \tag{6.7}$$

where s_1 and s_2 are source signals, α_{ij} are mixing coefficients corresponding to signal j at channel i and power of a signal x consisting of L samples is calculated as

$$P(x) = \sum_{k=1}^{L} x[k]^2 \tag{6.8}$$

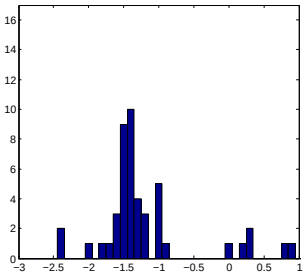
Though in Figures 6.4a and 6.4b number of θs found around -1 are almost equal for both simulations with and without PE, it is different in Figure 6.4d. That difference is important because number of θs around -1 and -1.5 are equal and

in such a case it is not possible to decide which value is correct. A similar but worse problem is seen Figure 6.5d such that number of θs around -1 is larger the one around -1.5. So, the decision of θs would be wrong. However, in Figure 6.5d, it is possible to find correct θs clearly. In addition number of θs around -1 is larger in Figure 6.5a than 6.5b.

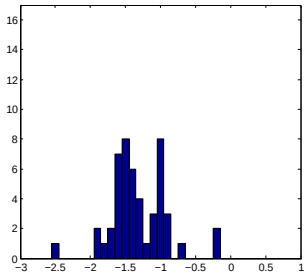


(a) Clustering of θ_1 with PE

(b) Clustering of θ_1



(c) Clustering of θ_2 with PE



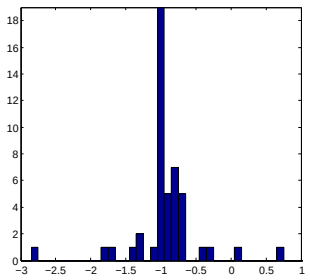
(d) Clustering of θ_2 without PE

Figure 6.4: $SNR_1 = 1.7851$, $SNR_2 = 0.0241$ where $tan(\theta_1) = -1$ and $tan(\theta_2) = -1.5$

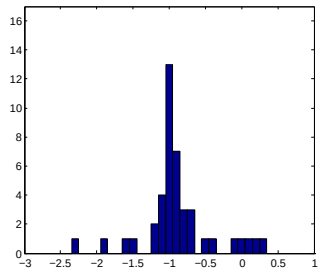
Experiments show that it is beneficial to use PE since it increases the probability of finding de-mixing directions correctly by providing better clustering at correct values.

6.3 Effect of SNR on Histograms of θ

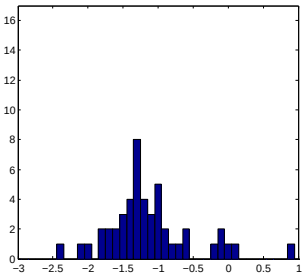
The success of separation depends on correctly finding demixing directions, θs. As SNR increases, it becomes harder to find direction of speech signal. On the other hand, finding direction of noise signal becomes easier with high SNR. That



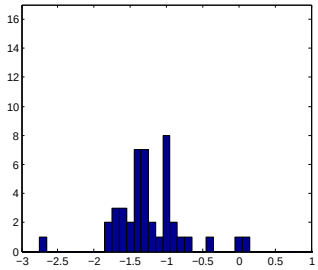
(a) Clustering of θ_1 with PE



(b) Clustering of θ_1 without PE



(c) Clustering of θ_2 with PE



(d) Clustering of θ_2 without PE

Figure 6.5: $SNR_1 = -2.98$, $SNR_2 = -4.74$ where $tan(\theta_1) = -1$ and $tan(\theta_2) = -1.5$

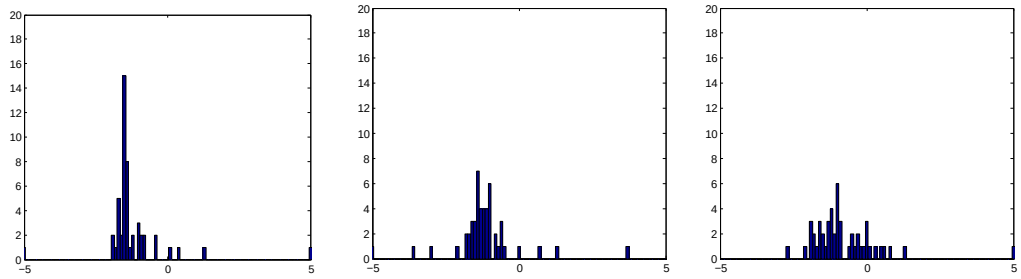
effect can be observed by investigating histograms of $\theta 1$ and $\theta 2$ in Figures 6.6 and 6.7, which are demixing directions of speech and noise sources, respectively.

In these experiments a women is speaking and background noise is recorded in a plaza. Those data are taken from SISEC 2010 database. The source signals used in this experiment can be seen in Figure 6.9b where the speaker is "female speaker 2" in Section 6.4. Mixing matrix and corresponding demixing directions are

$$\mathbf{A} = \begin{bmatrix} .9 & .5 \\ .6 & .5 \end{bmatrix} \qquad and \qquad \mathbf{\Theta} = \begin{bmatrix} -0.7854 \\ -0.9828 \end{bmatrix} \tag{6.9}$$

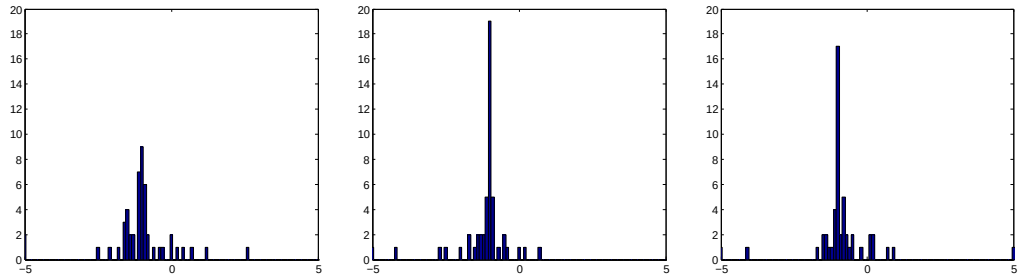
where $\tan(\theta_1) = -1.0$ and $\tan(\theta_1) = -1.5$.

In Figure 6.6, histogram becomes more scattered with increasing SNR. On the other hand, in Figure 6.7 clustering at correct demixing direction increases with SNR. Note that, in those figures, SNR is increased by enhancing noise signal, so, the noise enhancement factor is provided. In Table 6.3 relation among noise enhancement factor and SNR level is provided.



(a) noise enhancement factor = 1 (b) noise enhancement factor = 10 (c) noise enhancement factor = 20

Figure 6.6: Histograms of θ_1 for various SNR levels



(a) noise enhancement factor = 1 (b) noise enhancement factor = 10 (c) noise enhancement factor = 20

Figure 6.7: Histograms of θ_2 for various SNR levels

noise enhancement	**SNR_1**	**SNR_2**
1	8.77	7.01
10	-1.22	-2.98
20	-4.24	-5.99

Table 6.3: SNR level with respect to noise enhancement factor of noise signal

SNR level has opposite effects on histograms of speech and noise. It is seen that Figure 6.6c and 6.7a are similar such that there are 6 and 9 values clustered

in correct θ, respectively. However, for speech signal, when SNR is high as in Figure 6.6a there are 16 values out of 50 trials clustered in correct direction. Similarly, there are 17 values acclustered in correct direction for noise when SNR is low as seen in Figure 6.7c. As seen in Figures 6.6b and 6.7b, number of clustered values for speech signal decreases more rapidly than the one for noise signal.

As expected, SNR has a scattering effect on histograms of θ of speech. As SNR increases, information of speech signal becomes more and more buried into noise and it becomes harder to extract speech out of the mixture. On the other hand, it becomes easier to extract noise as SNR increases. That can be advantageous because extracting noise and subtracting it out from the mixture can provide speech signal. However, since ICA cannot recover amplitudes of signals, this is not a straightforward solution.

6.4 Performance of the ICA-PSO-PE Algorithm with Various Sources

Aim of this experiment is to investigate the performance of ICA-PSO-PE with various noise and speech sources. In this experiment, records from SISEC 2010 database are used as source signals. Source signals are synthetically mixed using the following mixing matrix, $\mathbf{A}$

$$\mathbf{A} = \begin{bmatrix} .9 & .5 \\ .6 & .5 \end{bmatrix} \qquad and \qquad \mathbf{\Theta} = \begin{bmatrix} -0.7854 \\ -0.9828 \end{bmatrix} \tag{6.10}$$

where $\mathbf{\Theta}$ consists of corresponding de-mixing directions. Here, tangent of first element of $\mathbf{\Theta}$, θ_1 is -1.0 and tangent of second component,θ_2, is -1.5.

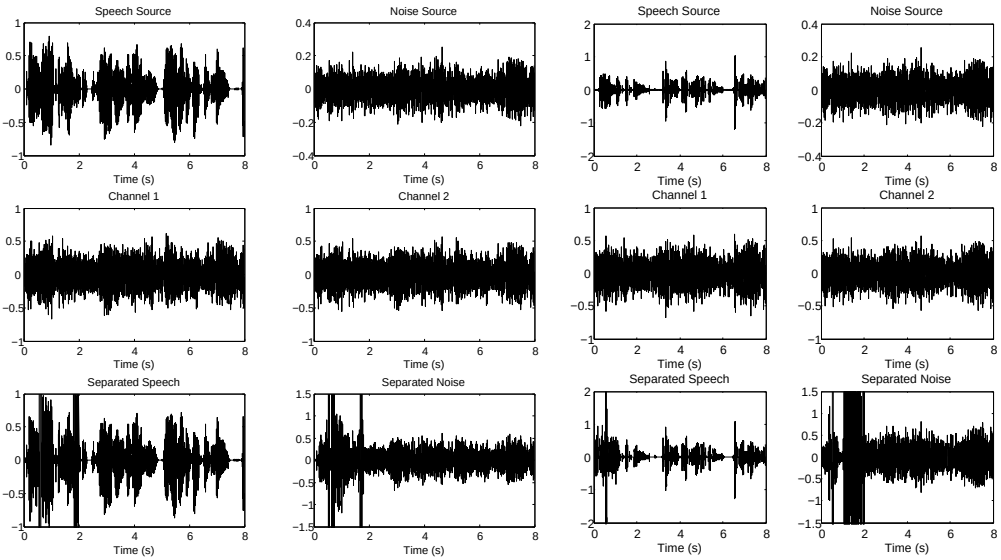
Recordings contain voices of male and female speakers and noises of a cafeteria, a subway and a plaza. The recordings in cafeteria contains voices of other speakers mainly. The noise recorded at subway consists of the noise of a train passing by. The one recorded at plaza is a mixture of noises of cars passing by, voices of people and children.

The sentences read by speakers are the following

- Male speaker 1 (m_1):“ “This food’s too spicy”, he complained. Young men can be very arrogant and rude. So, Marcus owned the big shipping company. Their eyes met across the table.”
- Male speaker 2 (m_2): “Time is running out for the scientists. If you knew July like I knew July. Your new dress is breath-taking darling. Her first book was published last year. ”
- Female speaker 1 (f_1): “An enormous quake rocked the island. Eventually, he hopes to solve all the problems. Fault installation can be blamed for this.”
- Female speaker 2 (f_2): “Among them are the canvases by a young artist. Building from the ground up is very costly. Next year we’ll see several more exhibitions. The number of works on view will increase.”

Since the main concern here is the effect of statistical properties of signals, SNR levels are around -2 dB and -7dB. Such SNR values are still challenging but fair enough to emphasize statistical properties of signals instead of effects like scattering of directions due to SNR (Section 6.3). SNRs on channels 1 and 2 are provided in Table 6.4 as SNR_1 and SNR_2, respectively. Source signals, observed signals on channels 1 and 2, and separated signals are shown in Figures 6.8a through 6.10b. As a final note, *exponential* objective function is used in all

experiments.



(a) with speaker m_2 (b) with speaker f_1

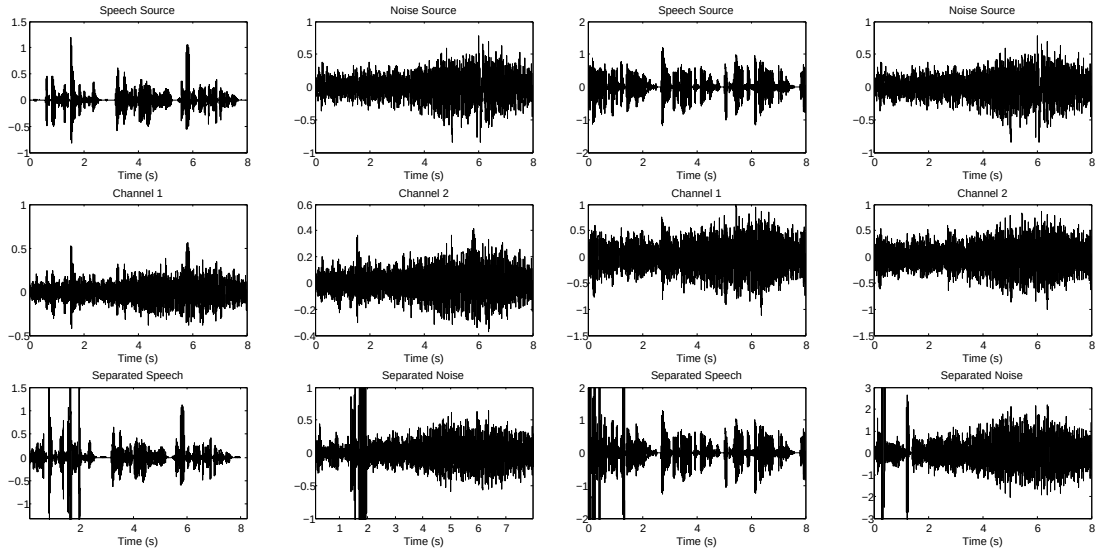
Figure 6.8: Noise of Cafeteria

Noise Source	**Speech Source**	$\mathbf{SNR_1}$	$\mathbf{SNR_2}$
Cafeteria	m_2	-2.4355	-5.2431
Cafeteria	f_1	-3.4872	-4.1964
plaza	m_1	-3.1190	-4.8799
plaza	f_2	-3.4753	-5.2362
Subway	m_1	-2.0152	-3.7762
Subway	f_2	-5.3818	-7.1427

Table 6.4: SNR levels of various mixtures

Though a few examples of experiments are demonstrated here, it is observed that ICA-PSO-PE successfully separates signals when SNR levels are up to approximately -12 dB to -15 dB. However, m_1 is a problematic recording that it cannot be separated when SNR level at main microphone is lower than -2.4 dB.

Results of experiments show that ICA-PSO-PE can separate all source signals correctly. Since working principles of ICA-PSO-PE depend on statistical properties of the signals, there can be signals that ICA-PSO-PE fails to separate their



(a) with speaker m_1 (b) with speaker f_2

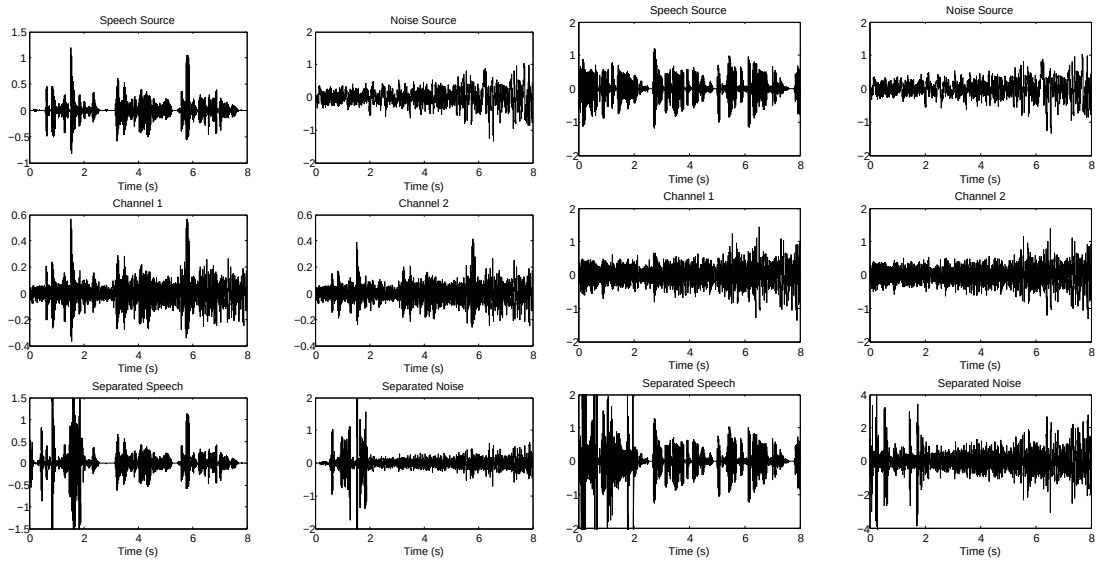
Figure 6.9: Noise of plaza

mixtures. Straightforward examples for such a case are mixture of two gaussian distributed signals or mixtures of two speakers' voices. However, for mixtures of speech and background noise ICA-PSO-PE performs fairly good.

6.5 Duration of Learning Period

Duration of learning period affects the probability of choosing the correct direction for each signal. The decision of correct directions of separation (θ_1 and θ_2) is given based on clustering of θs in histograms. Direction with maximum clustering is chosen as the direction of separation. As shown in Section 6.3, histograms become scattered as SNR increases. So, number of trials before decision must be large enough to overcome scattering and enable clustering on correct directions.

The number of trials before deciding directions of separation are called *learning period*. During the learning period, θs are collected and labelled using PE. There are two possible labels for each θ: It is either separation direction of speech



(a) with speaker m_1 (b) with speaker f_2

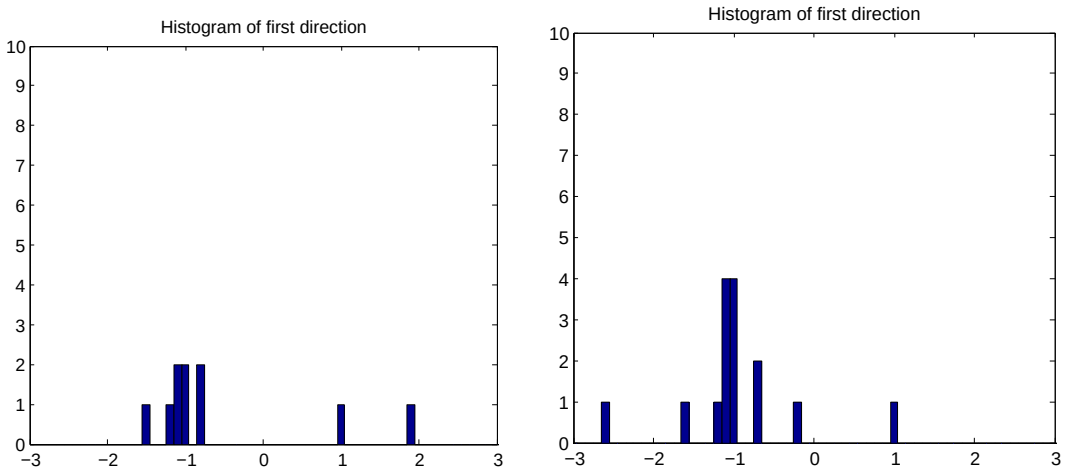
Figure 6.10: Noise of subway

or separation direction of noise. The algorithm produces an output but PE may not always label θ correctly for each frame because some frames are transition frames that contain both voiced and unvoiced speech which were addressed in detail in Section 5.1. At transition frames, PE gives similar values for speech and noise signals, which may lead to erroneous labelling of signals. Normally, the larger value provided by PE is used to label speech signal and the smaller one is used for noise signal. When those values are close, signals may be labelled incorrectly.

In practice, labelling is very important because if signals are incorrectly labelled, communication is not possible. For instance, if noise signal is labelled as speech, it is transmitted instead of speech. In addition, when some noise frames are transmitted instead of speech frames, it is very difficult to understand the speech since erroneous frames cause disturbing voices.

In order to increase quality of communication, labelling must be done with high accuracy. So, a learning period is used to give consistent decisions on separation directions of speech and noise. Among the collected and labelled θs during the learning period, the directions with maximum clustering is chosen as the direction of separation for each signal. Experiments showed that 50 frames, which is 2 seconds with frames of 40 ms, is an optimal duration that gives good results even at very low SNR values. In those expriments, voice of a female speaker with background noise of a plaza (cars passing by, people talking and other noises) is used from SISEC 2010 database.

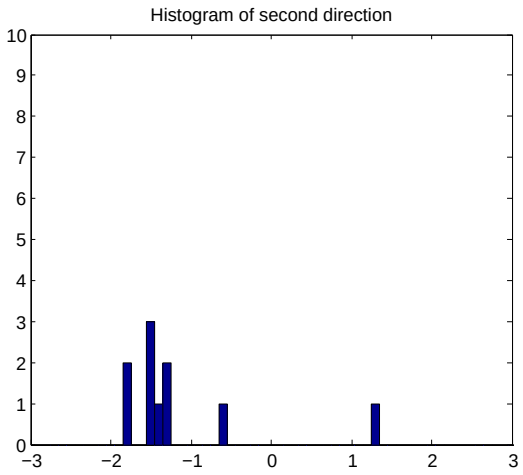
As can be seen in Figure 6.11a, it is not possible to make a decision. Though a duration of 10-frames learning period is useful for finding second direction, scattering effect of decreasing SNR makes 10-frames learning period insufficient.



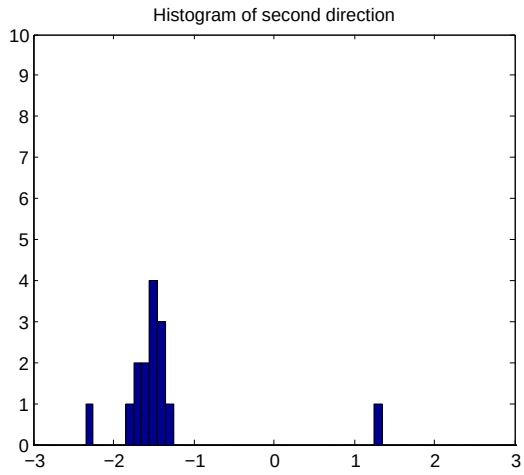
(a) 10 frames, channel 1

(b) 15 frames, channel 1

Figure 6.11: Noise is not enhanced and $tan(\theta_1) = -1$



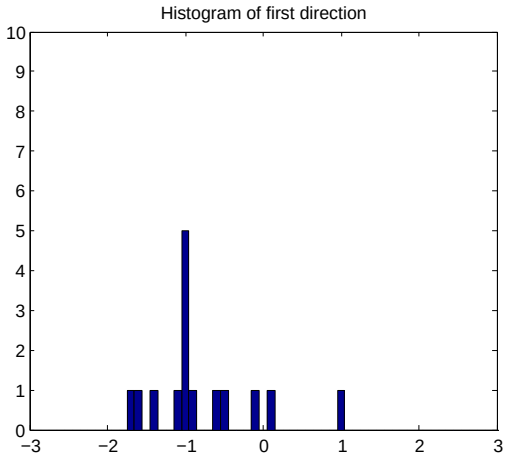
(a) 10-frames period, channel 1



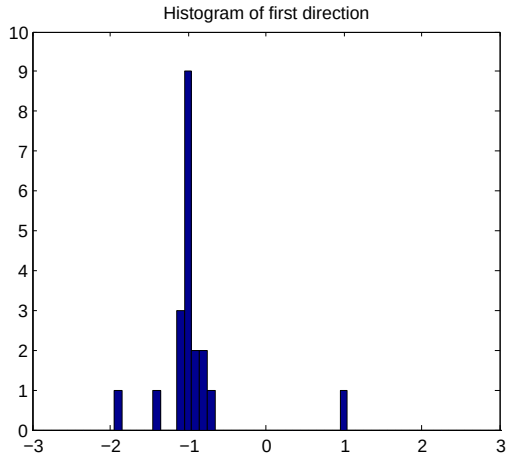
(b) 15-frames period, channel 1

Figure 6.12: Noise is not enhanced and $tan(\theta_2) = -1.5$

In the next experiment, noise is 10 times enhanced and learning duration is increased to 15 and 20 frames. Figures 6.13 and 6.14 show that, a learning duration of 15 frames is not enough to overcome scattering effect. On the other hand, a duration of 20 frames seem to be sufficient to decide on the correct direction for both channels.



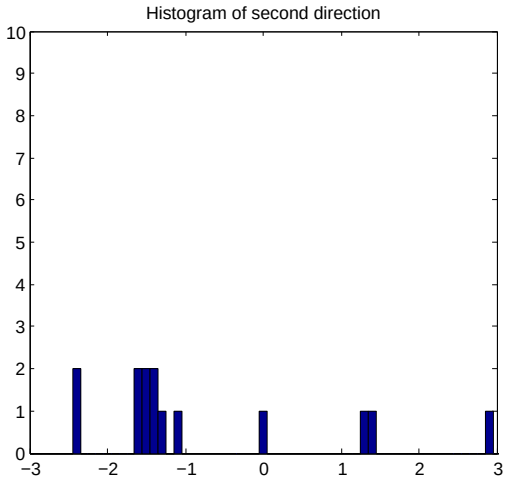
(a) 15-frames period, channel 1



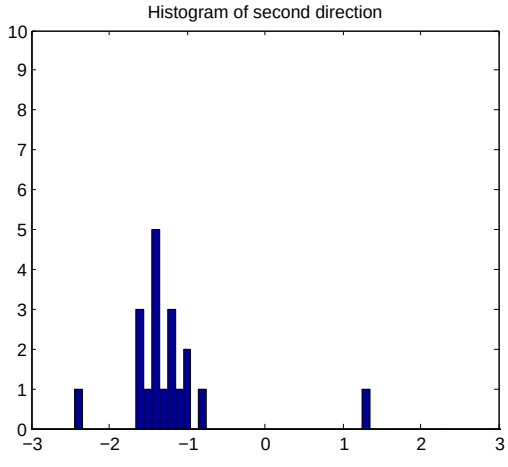
(b) 20-frames period, channel 1

Figure 6.13: Noise enhancement factor is 10 and $tan(\theta_1) = -1$

However, increasing noise enhancement factor to 15 times, decreases success of 20 frames. As seen in Figure 6.18a, 20 frames cannot perform well enough to



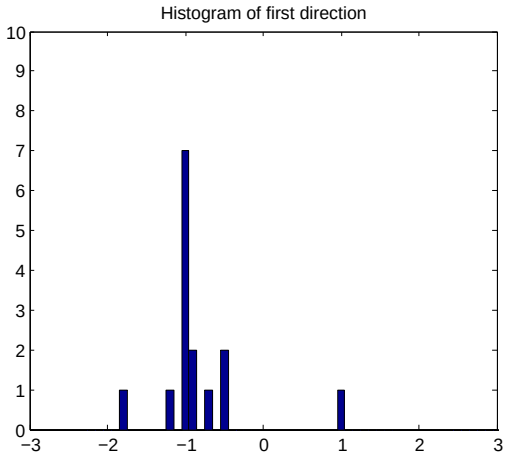
(a) 15-frames period, channel 2



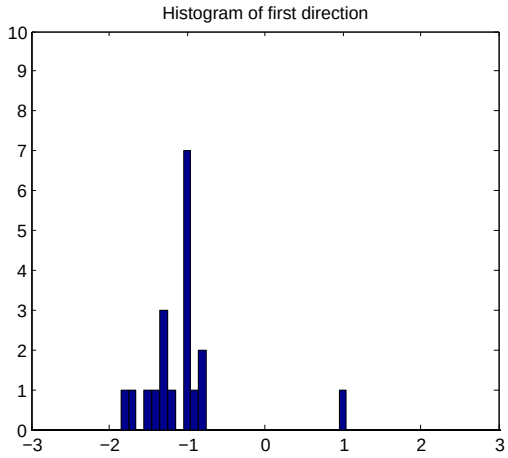
(b) 20-frames period, channel 2

Figure 6.14: Noise enhancement factor is 10 and $tan(\theta_2) = -1.5$

choose a correct θ for the second direction. On the other hand, duration of 15 frames is still useful at first direction but there is no opportunity to decide at the second one.



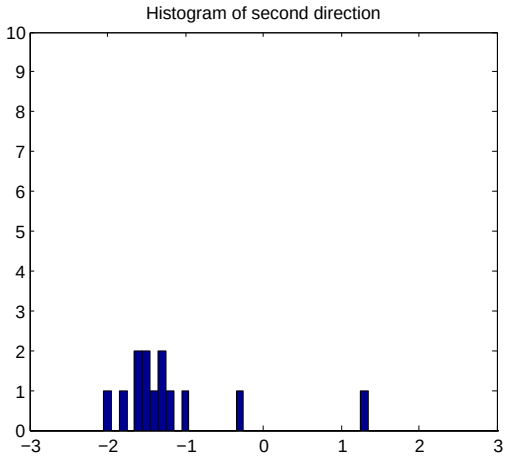
(a) 15-frames period, channel 1



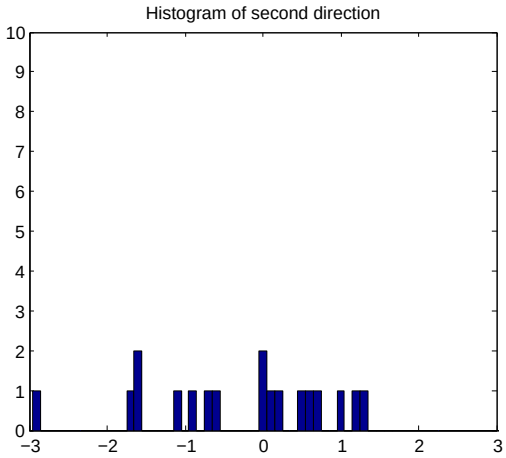
(b) 20-frames period, channel 1

Figure 6.15: Noise enhancement factor is 15 and $tan(\theta_1) = -1$

Increasing learning duration to 30 frames provides good results for an noise enhancement factor of 15 times. On the other hand, 20 frames learning duration keeps on providing good results for the first direction but it is not possible to choose a direction for the second one.

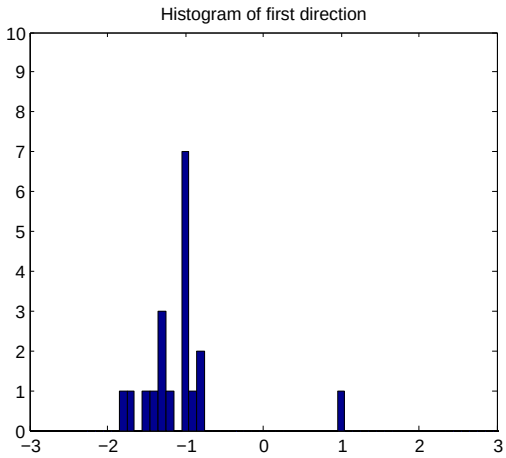


(a) 15-frames period, channel 2

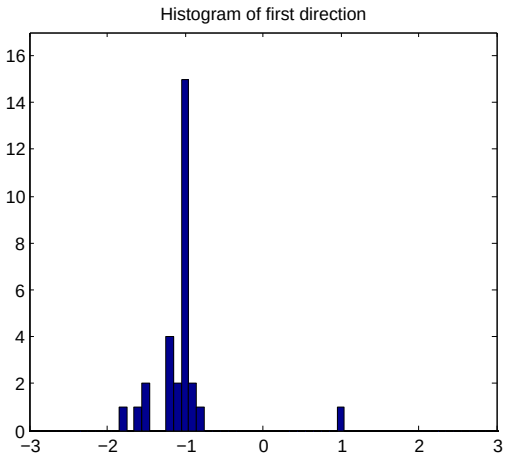


(b) 20-frames period, channel 2

Figure 6.16: Noise enhancement factor is 15 and $tan(\theta_2) = -1.5$



(a) 20-frames period, channel 1

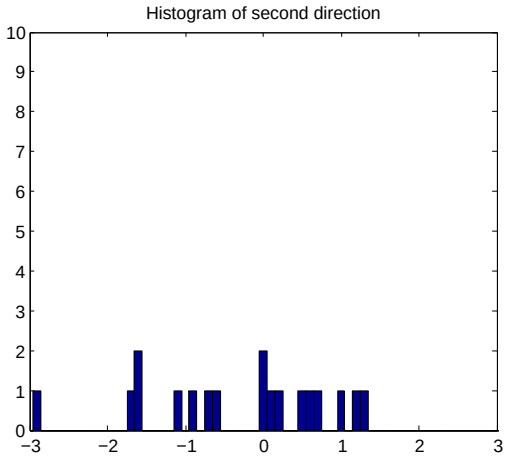


(b) 30-frames period, channel 1

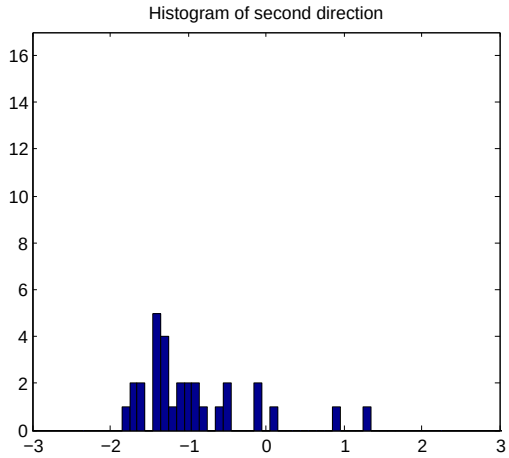
Figure 6.17: Noise enhancement factor is 15 and $tan(\theta_1) = -1$

In Figures 6.19a and 6.20a, it is seen that 30 frames of learning duration is not successful to decide second θ, though it is possible to choose first one. On the other hand, 50 frames of learning duration makes it possible for both directions.

Increasing noise enhancement factor above 20 times does not help to analyse effects of learning duration because there exist problems due to theoretical factors

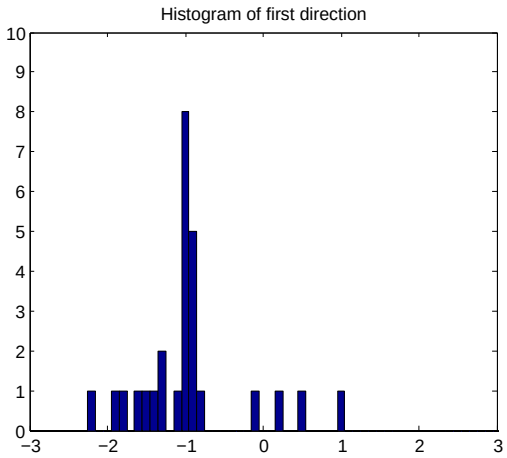


(a) 20-frames period, channel 2

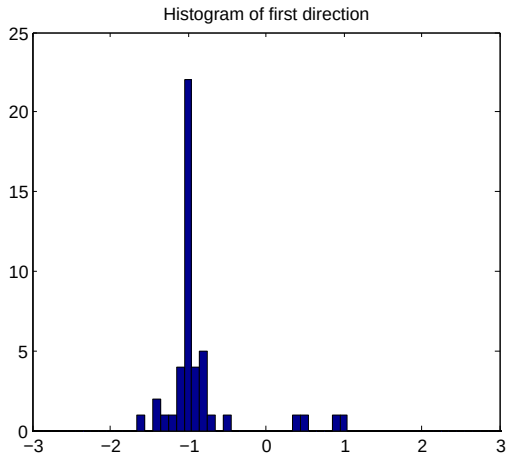


(b) 30-frames period, channel 2

Figure 6.18: Noise enhancement factor is 15 and and $tan(\theta_2) = -1.5$



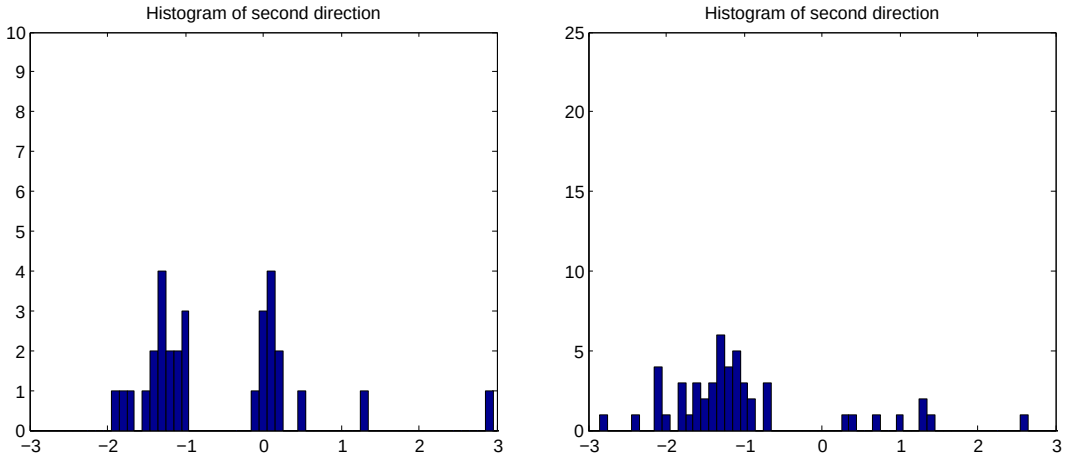
(a) 30-frames period, channel 1



(b) 50-frames period, channel 1

Figure 6.19: Noise enhancement factor is 20 and $tan(\theta_1) = -1$

which cannot be solved by increasing learning duration. So, 50 frames is an optimal duration of learning.



(a) 30-frames period, channel 2

(b) 50-frames period, channel 2

Figure 6.20: Noise enhancement factor is 20 and $tan(\theta_2) = -1.5$

Noise Enhancement Factor	**Duration (frames)**	**SNR_1**	**SNR_2**
1	10	9.4588	7.6979
1	15	9.4787	7.7177
10	15	-0.5213	-2.2822
10	20	-0.8506	-2.6115
15	15	-2.2822	-4.0431
15	20	-2.6115	-4.3724
15	30	-2.6814	-4.4423
20	30	-3.9308	-5.6917
20	50	-4.1878	-5.9487

Table 6.5: SNR levels during learning periods with respect to noise enhancement factor

6.6 Comparisons with Other Noise Cancellation Methods

In the first part, performance of ICA-PSO-PE is compared with a widely used ICA algorithm, FastICA. Since FastICA is not real-time and it cannot overcome ambiguities of ICA, their performance are compared on 200 frames. In other words, frame-based performance of ICA-PSO-PE is compared with performance of FastICA on all frames.

As a real-time noise cancellation method, the inverse of the signal in the second channel is summed up with the one in first channel. Actually, this method is the most efficient one if gains of microphones are matched and there is no latency at one of the channels. However, when those ideal conditions are not satisfied, this method does not perform well.

There are several conditions effecting the performance of the algorithm like objective functions, optimization method and parameter selection. For a fair comparison, those conditions are made the same as much as possible.

6.6.1 Comparisons with FastICA

For this experiment, FastICA for Matlab 7.x and 6.x version 2.5 is used. It was released in October 19 2005 and its copyright belongs to Hugo Gävert, Jarmo Hurri, Jaakko Särelä, and Aapo Hyvärinen. The packet of FastICA contains two main functions *fastica* and *fasticag* where *fastica* is the command line version of main function and *fasticag* is the one with graphical user interface. There are several other functions called by main functions which perform additional processes like whitening and PCA. I used *fasticag* to provide data and choose the objective function to be used.

There are several factors, which effect the performance of algorithms and prevent a totally fair comparison among FastICA and ICA-PSO-PE. The differences are

- Frame-based structure and learning period: FastICA is not real-time and it uses all the data to separate signals. In other words, FastICA has more information about signals compared to ICA-PSO-PE because the latter

has a frame-based structure that provides the information of only 40 ms of data. Since the information used by ICA-PSO-PE is limited, it has a learning period which takes 50 (or less) frames to decide. Though ICA-PSO-PE produces an output during learning period, it is not as reliable as the ones produced after the end of learning period.

- Objective functions: The objective functions used by FastICA and ICA-PSO-PE are slightly different. In order to increase speed and reduce computational complexity, objective functions are simplified as much as possible with ICA-PSO-PE.

- Pre-processing steps: The pre-processing steps of ICA, which are previously mentioned in Section 2.1.1 are skipped to increase speed by reducing computational complexity. On the other hand, FastICA uses those pre-processing for separating signals correctly. In addition, FastICA uses a mid-processing step, orthogonalization, which is mentioned again in Section 2.1.1.

Experiments showed that FastICA and ICA-PSO-PE can perform equally good even when SNR levels are very low. Using the various mixing matrices, objective functions and SNR levels, it is shown that ICA-PSO-PE is as successful as the well-known FastICA.

In most of the experiments exponential objective function is used. In this one, besides exponential objective function, hyperbolic tangent is also used because FastICA uses it too and comparing with the same objective function is more fair. It is defined as

$$g(x) = \tanh(x) \tag{6.11}$$

Besides tanh, FastICA also uses Gaussian objective function, which is defined as

$$g(x) = x \exp\left(\frac{-a * x^2}{2}\right) \tag{6.12}$$

where a is a constant and generally chosen as 1. Gauss objective function is very similar to the exponential objective function used by ICA-PSO-PE which is defined in Eq. 4.22 and provided here for completeness.

$$g(x) = \exp\left(\frac{x^2}{2}\right) \tag{6.13}$$

In this part, exponential objective function for ICA-PSO-PE and gauss objective function for FastICA is used. In Figure 6.21, it is seen that results of ICA-PSO-PE and FastICA are the same after the end of learning duration (2 seconds), although SNR levels are significantly low. Actually, noise enhancement factor can be increased up to 70 times meaning that $SNR_1 = -11.9262$ and $SNR_2 = -13.6871$. Up to this point ICA-PSO-PE performs as good as FastICA. But after 70 times, FastICA keeps on performing incredibly good but ICA-PSO-PE fails to find one of the directions.

Using tanh, it is possible to increase noise enhancement factor up to 140 times meaning that $SNR_1 = -14.9365$ and $SNR_2 = -16.6974$. Results obtained with 140 times noise enhancement, signals observed at channels and source signals are shown in Figure 6.22. At 140 times noise enhancement, directions are accurately found by both methods. The directions found by ICA-PSO-PE are $\theta_1 = -0.8330$ and $\theta_2 = -0.7854$ whereas the theoretical ones are $\theta_1 = -0.9828$ and $\theta_2 = -0.7854$. Similar to the previous case, when noise enhancement factor is more than 140 times, ICA-PSO-PE cannot find one of the directions but FastICA keeps on separating signals correctly. However, note that at such SNR levels, it is not even possible to hear speech since it is buried deeply into the noise. That can be seen by looking at the signals observed at Channel 1 and 2 in Figures 6.21 and 6.22.

As the experiments show, ICA-PSO-PE performs as good as FastICA up to a certain level of SNR. On the one hand, FastICA uses pre-processing steps, mid

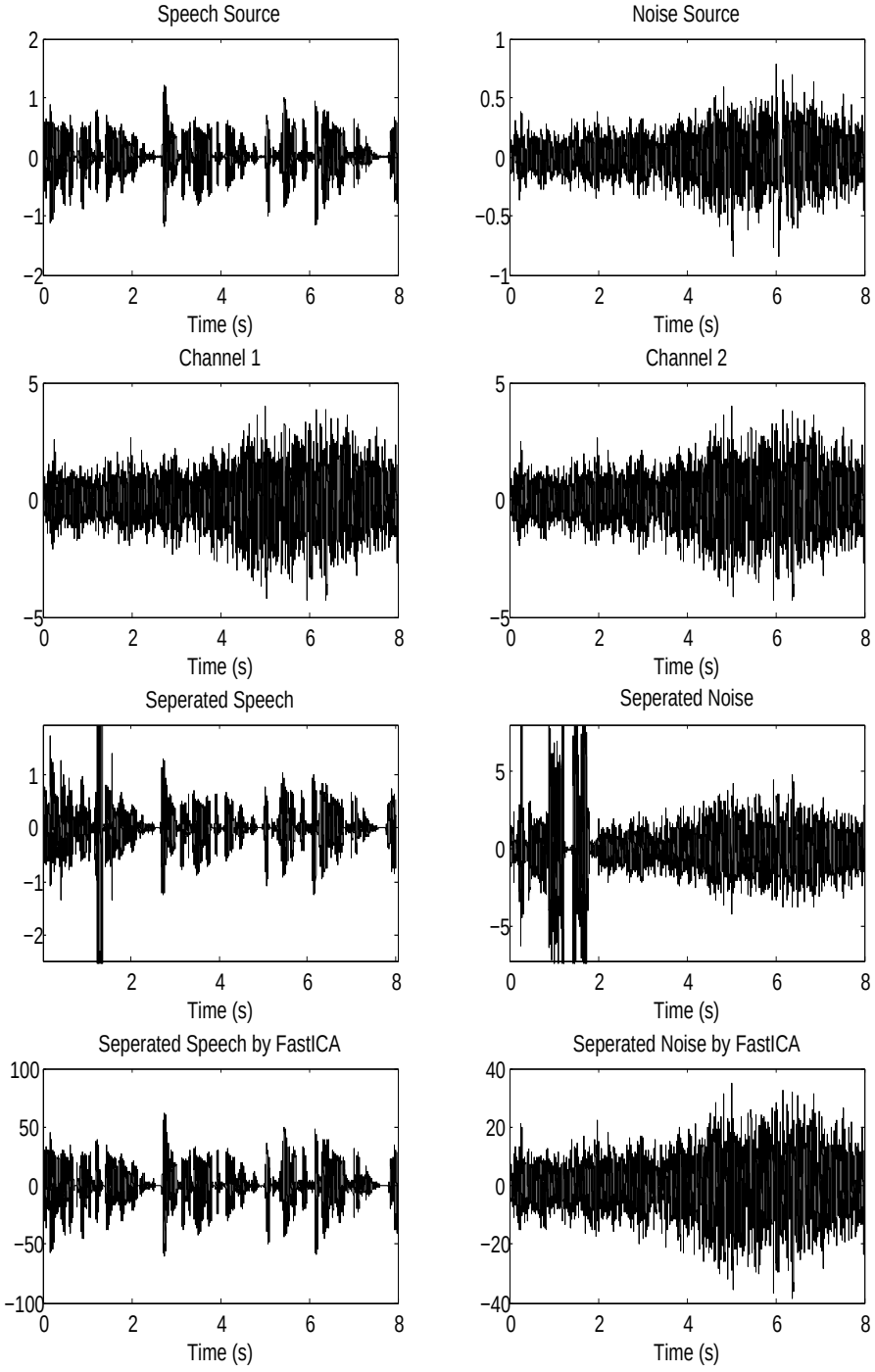


Figure 6.21: Noise enhancement factor is 50 times, $SNR_1 = -10.4650$ and $SNR_2 = -12.2259$, objective function is exponential for ICA-PSO-PE and gauss for FastICA

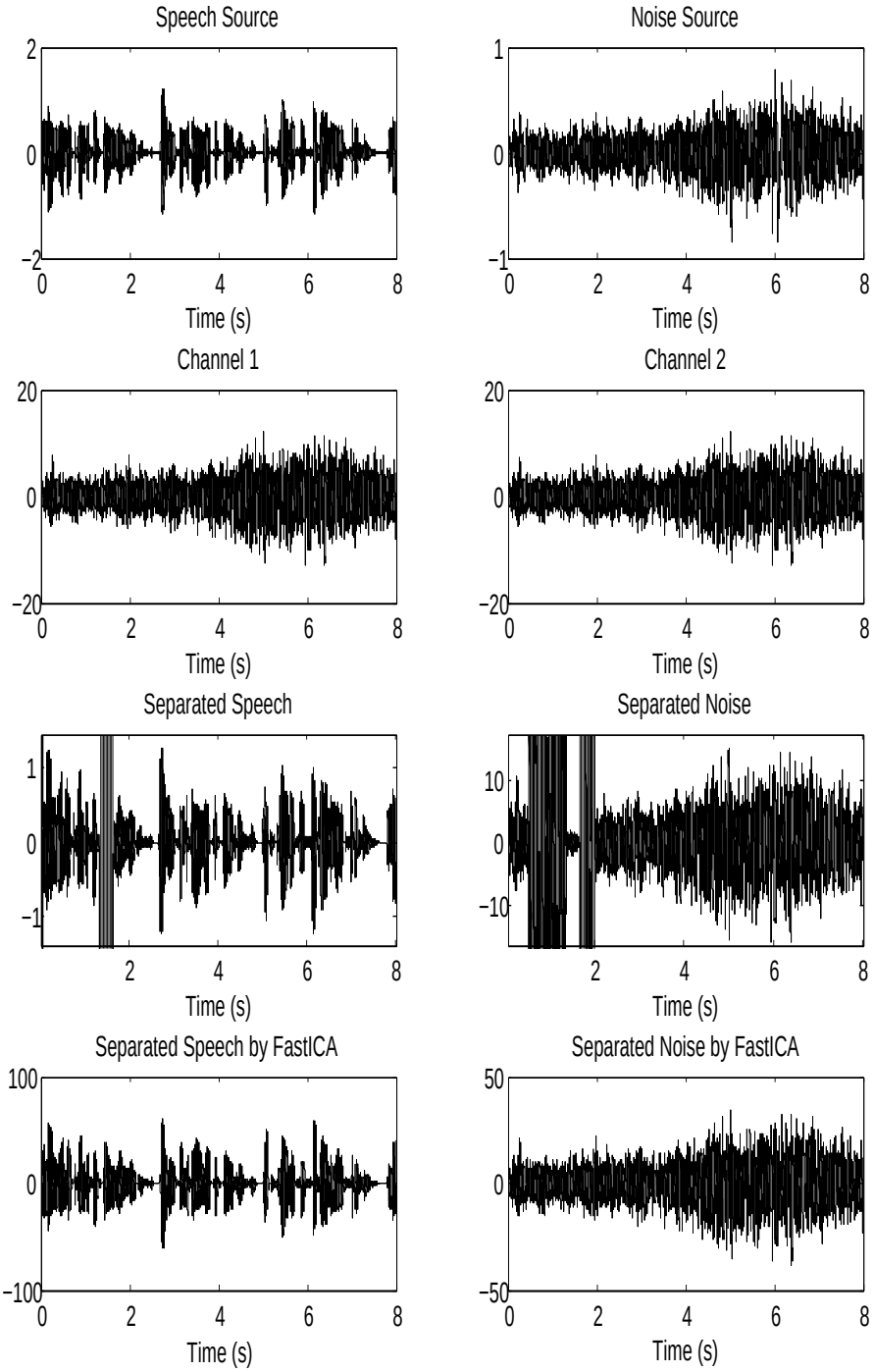


Figure 6.22: Noise enhancement factor is 140 times, $SNR_1 = -14.9596$ and $SNR_2 = -16.7205$, objective function is tanh for ICA-PSO-PE and FastICA

processing steps and 8 seconds long data for separation. On the other hand, ICA-PSO-PE uses 2 seconds of data in terms of 50 frames where each one of them is 40 ms long. It has no pre-processing or mid-processing steps. So, it is acceptable that ICA-PSO-PE fails at very low SNRs before FastICA. However, the SNR levels that ICA-PSO-PE fails are very low that it is hard to observe them in the real systems since the main microphone is expected to obtain a signal with high SNR. In addition, note that FastICA cannot label signals as speech or noise but ICA-PSO-PE directly decides which on is speech and which one is noise.

As a final note, it can be said that performance of both algorithms are effected by, in addition to the factors mentioned above, properties of source signals and coefficients of mixing matrix but under the same conditions, they are expected to perform similarly.

6.6.2 Comparisons with Subtraction Method

Subtraction method is a simple but efficient method for noise cancellation in systems like modelled in Chapter 1. In this method, signal received by Channel 2 (sub channel) is subtracted from the signal received by Channel 1 (main channel). This operation is

$$\mathbf{y} = \mathbf{x_1} - \mathbf{x_2} \tag{6.14}$$

where $\mathbf{y}$ is the demixed signal and $\mathbf{x_1}$ and $\mathbf{x_2}$ are observed signals at channels 1 and 2, respectively. In other words, $\mathbf{x_1}$ and $\mathbf{x_2}$ are rows of the matrix of observed signals, $\mathbf{X}$ mentioned previously.

If gains of receivers at both channels are 100% matched, this method can be very successful because Channel 1 and 2 receive noise at the same level but level of speech is larger at Channel 1. So, in case of subtraction, noise components

cancel each other and speech signal with a decreased level of amplitude remains.

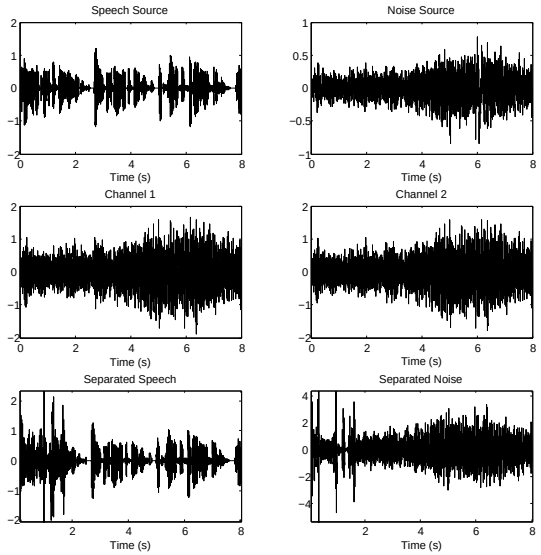
Since there is no operations except subtracting received signal from each other, this method is the fastest and simplest method. However, it performs worse than ICA-PSO-PE or FastICA when SNR level is very low or gain of receivers are not completely matched. In this experiment, performance of this method is compared with the one of ICA-PSO-PE. Voice of a female speaker and noise recorded in a plaza from SISEC 2010 database is used in this experiment. Speech and background noise samples are mixed synthetically with a the following mixing matrix

$$\mathbf{\Theta} = \begin{bmatrix} -0.7854 \\ -0.9828 \end{bmatrix} \qquad and \qquad \mathbf{A} = \begin{bmatrix} 0.9 & 0.5 \\ 0.6 & 0.5 \end{bmatrix} \tag{6.15}$$

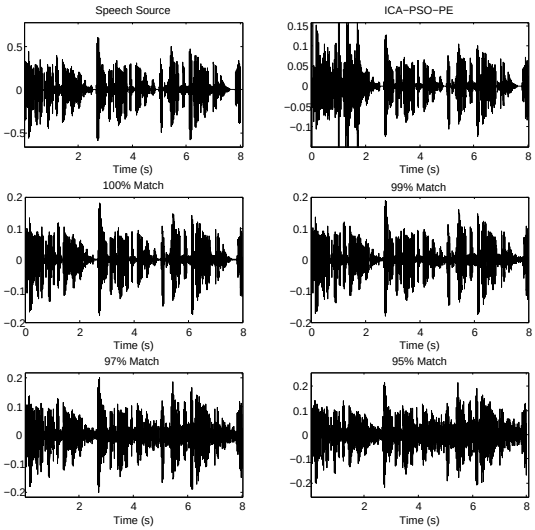
where Θ is the corresponding demixing directions to the mixing matrix, $\mathbf{A}$. There is no latency among channels and instantaneous mixing model is used.

Source signals, signals received in each channel and results obtained by ICA-PSO-PE using exponential objective function, with 20 and 50 times enhanced noise, are shown in Figures 6.23a and 6.24a for completeness. For comparison, source signals, results obtained using ICA-PSO-PE and subtraction method are provided in Figures 6.23b and 6.24b.

Note that, as matching percentage of receivers decreases, performance of subtraction method decreases too. In case of 20 times noise enhancement, performance of subtraction method is fairly good for even 95% matching (Figure 6.23b). However, when noise enhancement is 50 times, subtraction method cannot perform well at 97% matching (Figure 6.24b).



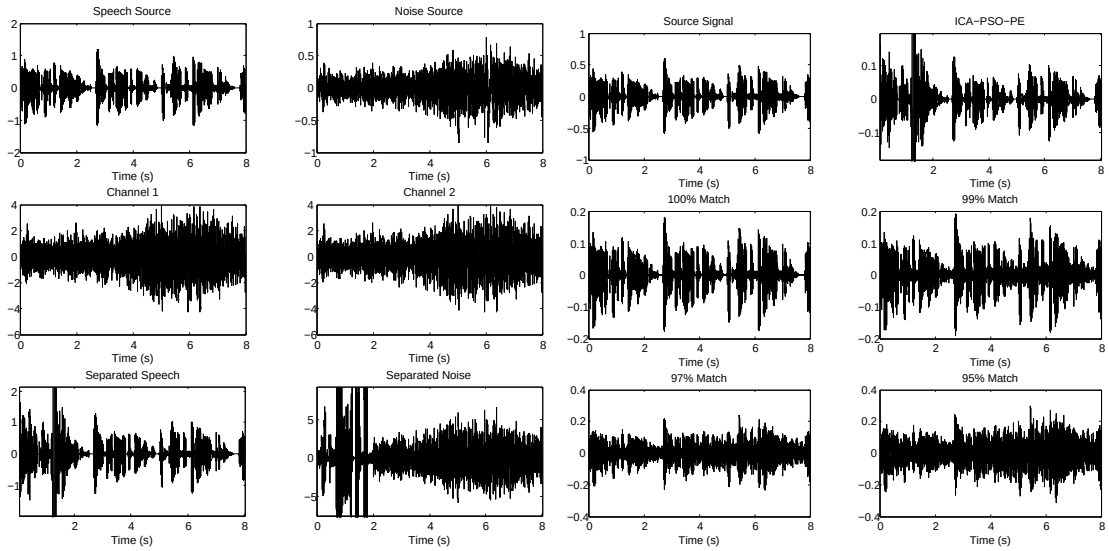
(a) Source signals, observed signals and results of ICA-PSO-PE



(b) ICA-PSO-PE vs Subtraction Method

Figure 6.23: noise enhancement is 20 times, learning duration is 2 s, $SNR_1 = -6.4855$ and $SNR_2 = -8.2464$, objective function is *exponential* for ICA-PSO-PE

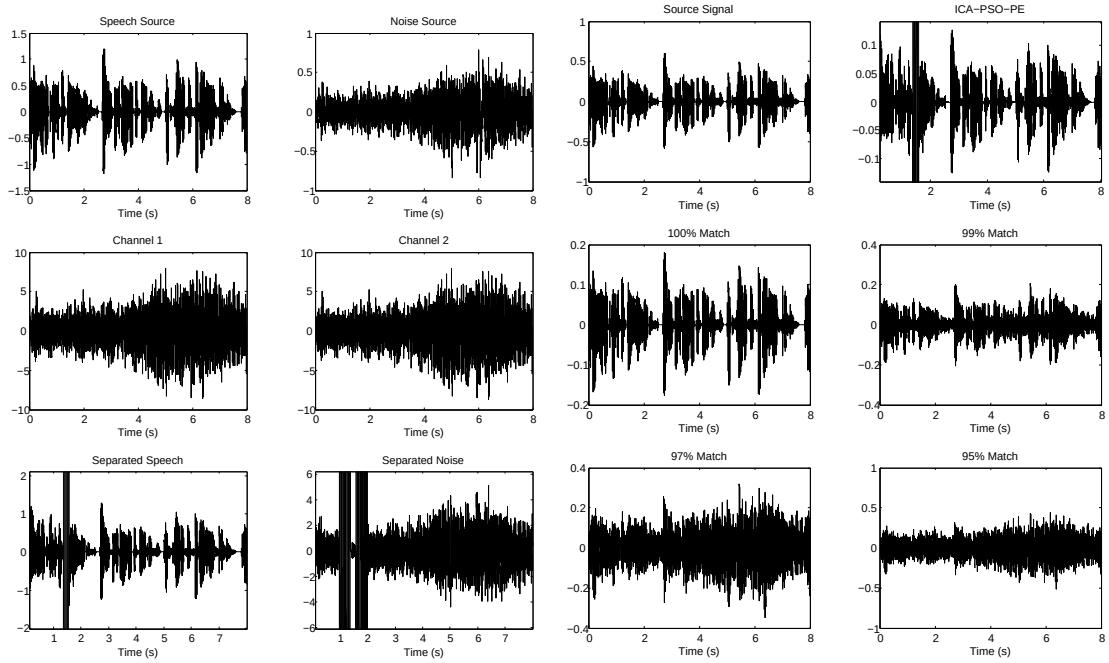
Using tanh objective function, noise enhancement is increased to 100 times. In Figures 6.25a and 6.25b, it is seen that even 99% matching results worse than ICA-PSO-PE. Even when SNR levels are very low, ICA-PSO-PE keeps on performing perfectly even at levels where 99% matched receivers fail. So, ICA-PSO-PE can be said to be more robust compared to subtraction method.



(a) Source signals, observed signals and results of ICA-PSO-PE

(b) ICA-PSO-PE vs Subtraction Method

Figure 6.24: noise enhancement is 50 times, $SNR_1 = -10.4649$ and $SNR_2 = -12.2258$, objective function is *exponential* for ICA-PSO-PE



(a) Source signals, observed signals and results of ICA-PSO-PE

(b) ICA-PSO-PE vs Subtraction Method

Figure 6.25: noise enhancement is 100 times, $SNR_1 = -13.4752$ and $SNR_2 = -15.2361$, objective function is tanh for ICA-PSO-PE

Chapter 7

CONCLUSIONS

The ICA-PSO-PE algorithm we proposed in this work is shown to be an efficient solution for noise cancellation problem on the given mobile communication system in Chapter 1 or similar systems. The only requirements for separating noise and speech signal are their statistical independence and at least one of them must have a non-gaussian distribution.

Performance of ICA-PSO-PE with various objective functions is tested. Experiments showed that exponential and hyperbolic tangent functions are superior to the others. Those objective functions are both It is shown that PE is a beneficial part of the algorithm since it labels data either as speech or noise. Thus, unscattered histograms of θs can be obtained. Especially with increasing SNR level, histograms of θs scatter significantly. In this case, duration of learning period must be chosen carefully. Our experiments showed that 50-frames period can be an optimum choice.

ICA-PSO-PE algorithm must be adaptive because speech and background noise signals can be very unstable. We conducted experiments with speech signals belonging to female and male speakers, and various background noises. ICA-PSO-PE succeeded at cancelling noise in each experiments, even at low SNR levels. In addition, ICA-PSO-PE is compared with FastICA and subtraction method. ICA-PSO-PE either performed as well as them or was superior.

ICA-PSO-PE is a real-time implementable algorithm due to its frame-based structure and computational efficiency. The expected computational cost of ICA-PSO-PE is calculated in Appendix B, which is 1 megacycle, or 20 ms. That means ICA-PSO-PE can be implemented in real-time because, remember that, each frame takes 40 ms.

Our future work will consist of implementing ICA-PSO-PE in a real-time environment like a digital signal processor (DSP) or field programmable gate array (FPGA). In addition, we used the instantaneous mixture model to simulate the mixing system of noise and speech signals. However, there can be latency among receivers due to their positional difference. So, A mixing model with latency at one of the receiver can be investigated.

Another future work for increasing accuracy of results may be introducing additional criteria before deciding θs. For instance, a measure can be introduced to measure the scattering level of θs' histograms. If that measure is at a certain level indicating that the histograms are very scattered, the learning period can be forced re-start. So, more accurate decisions on θs can be made. In the future, parameter selection for PSO can be investigated in more detail to increase speed of convergence or reduce probability of trapping into local extrema. In addition, information from consecutive frames can be used to provide faster convergence.

Since the ICA-PSO-PE algorithm must be adaptive, an alarm situation indicating a significant change in environmental conditions must be introduced. If the alarm is a valid one, learning period can re-start and ICA-PSO-PE adapts to the new conditions.

Improvement in the clarity of the transmitted speech can reduce number of repetitions significantly. This saves time, especially important in emergency situations, and eases battery energy requirements. Therefore, ICA-PSO-PE can be an innovative tool for communication systems where both time and energy constraints are critical. Performance of Performance of ICA-PSO-PE must be evaluated by considering all possible usage scenarios instead of just evaluating its noise cancellation performance in isolated signal segments.

APPENDIX A

WHITENING

Whitening is a preprocessing step for many ICA algorithms, including FastICA [13]. Because whitening only restricts the search space of mixing matrix, $\mathbf{A}$, it is not necessary but useful. For a random vector, $\mathbf{y} = (z_1, ..., z_n)^T$, *white* means that its elements are uncorrelated and have unit variances

$$E\{z_i z_j\} = \delta_{ij} \qquad and \qquad E\{\mathbf{z}\mathbf{z}^\mathbf{T}\} = \mathbf{I} \tag{A.1}$$

So, whitening is decorrelating and then scaling. Uncorrelatedness is related to independence but it is not enough to separate components. Note that two random variables are *uncorrelated* if their covariance is zero:

$$cov(y_1, y_2) = E\{y_1 y_2\} - E\{y_1\}E\{y_2\} \tag{A.2}$$

Whiteness is slightly stronger than uncorrelatedness because, as shown in A.1, variance of random variables are normalized, too. Of course, the strongest condition is independence since it also implies uncorrelatedness due to $E\{y_1 y_2\} = E\{y_1\}E\{y_2\}$ for independent random variables.

In order to whiten a random vector, $\mathbf{x}$, we can linearly transform it

$$\mathbf{y} = \mathbf{V}\mathbf{x} \tag{A.3}$$

Thus, we obtain a white random vector, $\mathbf{y}$. In order to find $\mathbf{V}$, we can use eigenvalues, $\mathbf{D} = diag(d_1, ..., d_n)$, and eigenvectors, $\mathbf{E} = (\mathbf{e}_1, ..., \mathbf{e}_n)$, of the covariance

matrix, $\mathbf{C_x} = E\{\mathbf{x}\mathbf{x}^\mathbf{T}\}$. After finding $\mathbf{D}$ and $\mathbf{E}$, a linear whitening transform is obtained as

$$\mathbf{V} = \mathbf{D}^{-\frac{1}{2}}\mathbf{E}^T \tag{A.4}$$

$\mathbf{V}$ exists if eigenvalues are positive. However, that is not really a restriction because $\mathbf{C_x}$ is positive semi-definite, making eigenvalues positive. Let us clarify why $\mathbf{V}$ is a whitening transform: Note that it is possible to write $\mathbf{C_x}$ in terms of its eigenvalue and vectors, $\mathbf{C_x} = \mathbf{E}\mathbf{D}\mathbf{E}^T$. Also, note that $\mathbf{E}$ is an orthogonal matrix, so that $\mathbf{E^T E} = \mathbf{I}$. Using A.5

$$\begin{aligned} E\{\mathbf{y}\mathbf{y}^\mathbf{T}\} =& \mathbf{V}E\{\mathbf{x}\mathbf{x}^\mathbf{T}\}\mathbf{V}^T \\ =& \mathbf{D}^{-\frac{1}{2}}\mathbf{E}^T\mathbf{E}\mathbf{D}\mathbf{E}^T\mathbf{E}\mathbf{D}^{-\frac{1}{2}} = \mathbf{I} \end{aligned}$$

So, $\mathbf{y}$ is whitened because its covariance matrix is identity matrix. However, $\mathbf{V}$ is not unique because any $\mathbf{U}\mathbf{V}$, where $\mathbf{U}$ is orthogonal, can whiten $\mathbf{x}$.

$$E\{\mathbf{x}\mathbf{x}^\mathbf{T}\} = \mathbf{U}\mathbf{V}^T E\{\mathbf{x}\mathbf{x}^\mathbf{T}\}\mathbf{V}^T\mathbf{U}^T \tag{A.5}$$

As a result, whitening provides only an orthogonal transformation of independent components and restricts the search space by solving half of the problem. Since whitening does not offer a unique solution, it is a preprocessing step before using ICA to find a unique mixing matrix.

APPENDIX B

COMPUTATIONAL COST OF ICA-PSO-PE ON TI C55x DSP

ICA-PSO-PE is supposed to work real-time in real life. We did not have the opportunity to implement it on real-time devices in this work. However, all experiments in MATLAB are conducted in a way resembling to real-life. For instance, 8 s of data (mixed signal) is divided into frames of 40 ms which is suitable for real-time environments. Note that frame length of 40 ms is due to PE algorithm and properties of speech signal, as investigated in Chapter 5 in detail.

Computational cost of ICA-PSO-PE is calculated in terms of clock cycles of the processor. Number of required clock cycles for performing necessary instructions are summed up to calculate overall computational cost. The most costly part of ICA-PSO-PE algorithm, which is the ICA-PSO loop running during the learning period, is considered in those calculations. Assuming a swarm size of 7, as widely used in our experiments, the following matrices are used in this process:

- $\mathbf{V}$ is the vector containing velocity of particles and its size is 7×1

- $\mathbf{P_{best}}$ is the vector containing personal best performance of each particle and its size is 7×1
- $\mathbf{S}$ is the vector of swarm and its size is 7×1
- $\mathbf{A}$ consists of repetition of the global best value and its size is 7×1
- $\mathbf{r_1}$ and $\mathbf{r_2}$ contain randomization factors for each particle and both of them are 7×1
- $\mathbf{W} = [\cos(\mathbf{S}) \ \sin(\mathbf{S})]$ is matrix of candidate demixing vectors and its size is 7×2
- $\mathbf{X'} = [\mathbf{x_1} \ \mathbf{x_2}]$ where $\mathbf{X}$ is the matrix of observed signals and $\mathbf{x_1}$ and $\mathbf{x_2}$ are the observed signals from channels 1 and 2, respectively. Since we use a frame duration of 40 ms and our sampling rate is 16kHz, size of $\mathbf{X}$ is 2×640
- $\mathbf{Y} = \mathbf{W} * \mathbf{X}$ where $\mathbf{Y}$ contains demixed data by using each particle. So, its size is 7×640 which is the larger matrix in our algorithm
- **fAll** contains value of objective function calculated for each row of $\mathbf{Y}$. So, its size is 7×1

For calculating the computational cost, we considered C55x digital signal processor (DSP) of Texas Instruments since those devices are widely used in mobile communication devices, especially radios. The TMS320C55x DSP is a fixed-point DSP in the TMS320 family, and it can use either of two forms of the instruction set: a mnemonic form or an algebraic form. Instructions and number of required to perform them are provided in Table B.1 based on *TMS320C55x DSP Library Programmer's Reference* and *TMS320C55x DSP Mnemonic Instruction Set Reference Guide*. In this table, n_x is the length of vector that will be used with that instruction.

In addition, note that generally each instruction like ADD or SUBT takes 1 cycle. If the clock of DSP is set to 50 MHz, 40 ms corresponds to 2 megacycles

Instruction	Explanation	Cost (Cycles)
expn	Exponential base e	11*n_x
mmul	Matrix multiplication where $\mathbf{R} = \mathbf{X_1} * \mathbf{X_2}$	if((row1==odd)(row1≥4)(col1≥2), $((col1 + 4) * 0.5 * (row1.1) + col1 + 12)col2$
neg	Vector negate	4*n_x
sine	$\sin(x)$ where $x \in [-\pi, \pi]$	19*n_x

Table B.1: Number of cycles to perform instructions

which is $2 * 10^6$ cycles. Computational cost of ICA-PSO per one iteration of a swarm with 7 particles is calculated step by step:

1. Calculate velocity: **77** cycles

$$\vec{v_i}(t) = \omega\mathbf{V} + c_1 r_1(\mathbf{P_{best}} - \mathbf{S}) + c_2 r_2(\mathbf{A} - \mathbf{S}) \tag{B.1}$$

   where $\omega\mathbf{V}$ costs 7 cycles,

   $(\mathbf{P_{best}} - \mathbf{S})$ costs 21 cycles,

   $c_1 r_1((\mathbf{P_{best}} - \mathbf{S}))$ costs 14 cycles and similarly

   $c_2 r_2(\mathbf{A} - \mathbf{S})$ costs $21 + 14 = 35$ cycles.

   By summing up all of them we find 77 cycles.

2. Update positions of particles: **7** cycles

$$\mathbf{S} = \mathbf{S} + \mathbf{V} \tag{B.2}$$

   which simply addition of 7 variables and takes 7 cycles.

3. Update $\mathbf{Y}$: **20480** cycles

$$\mathbf{Y} = \mathbf{WX} \tag{B.3}$$

   According mmul on Table B.1 and taking $row1 = 7$, $col1 = 2$, $row2 = 2$ and $col2 = 640$ we find 20480 cycles are required to perform this matrix multiplication.

4. Update function evaluations: **12160** cycles

$$result = \frac{sum\left(\exp\left(-\frac{\mathbf{Y}^2}{2}\right)\right)}{640} \tag{B.4}$$

where $\mathbf{Y}^2$, costs 640 cycles,

$\frac{\mathbf{Y}^2}{2}$, costs 640 cycles,

Negotiation, $-\frac{\mathbf{Y}^2}{2}$, costs 4*640 cycles,

Exponential, $\exp\left(-\frac{\mathbf{Y}^2}{2}\right)$, costs 11*640 cycles,

Summation, $sum\left(\exp\left(-\frac{\mathbf{Y}^2}{2}\right)\right)$, costs 640 cycles,

and division costs another 640 cycles.

Summing all of them, we find $19 * 640 = 12160$ cycles are required.

5. Rest of the process: **28** cycles

   There are no calculations or matrix multiplications after this point. Only, minimum or maximum of some matrices are found and those values are stored. Since they are 1-cycle-instructions, rest of the process is computationally easy.

In the end, approximately $77 + 7 + 266 + 20480 + 12160 + 28 = 33108$ cycles are required per each iteration of ICA-PSO with a swarm size of 7. In our experiments particles converge at 16-20 iterations. So, the computational cost of ICA-PSO is approximately **528288** - **660360** cycles, which is well below $2 * 10^6$ cycles that can be performed within 40 ms.

Once ICA-PSO is converged, PE is run to label separated data either as speech or noise. PE calculates autocorrelation coefficient at each pitch lag from 5 to 20 ms which corresponds to 241 integer pitch lags from 80 samples to 320 samples. Though the cost of PE highly depends on the way it is implemented, it is possible to say approximately **480000** cycles are required. Finally, cost of ICA-PSO-PE becomes approximately 1 megacycle.

If clock of DPS is 50 MHz, 1 megacycle takes 20 ms, which allows real-time noise cancellation. In addition, 50 MHz is not the fastest possible clock. There are many DSPs working faster and, for instance, ARM based microprocessors

can work with clocks faster than 450 MHz. In an FPGA, ICA-PSO-PE can run faster, too.

Bibliography

[1] T. S. R. Joseph C. Liberti and J. G. Proakis, "Evaluation of several adaptive algorithms for canceling acoustic noise in mobile radio environments," 1991.

[2] Y. Takahashi, *Blind Speech Enhancement with Independent Component Analysis and Spectral Subtraction*. PhD thesis, Dept. Elect. Eng., Nara Institute of Technology, Nara, Japan, 2010.

[3] J. F. Cardoso, "Source separation using higher order moments," *In Proc. ICASSP'89*, 1989.

[4] P. Comon, "Independent component analysis: A new concept," *Signal Processing, 36*, 1994.

[5] L. M. A. Cichocki, R. Unbehauen and E. Rummert, "A new on-line adaptive algorithm for blind separation of source signals," *In Proceedings of International Symposium on Artificial Neural Networks ISANN-94, Tainan, Taiwan*, 1994.

[6] A. Cichocki and R. Unbehauen, "Robust neural networks with on-line learning for blind identification and blind separation of sources," *IEEE Trans. on Circuits and Systems, 43(11)*, 1996.

[7] A. Bell and T. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation, 7*, 1995.

[8] A. Hyvärinen and E. Oja, "A fast fixed-point algorithm for independent component analysis," *Neural Computation*, 1997.

[9] J. K. A. Hyvärinen and E. Oja, *Independent Component Analysis*. John Wiley and Sons, Inc., 2001.

[10] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," *Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia*, pp. 1942–1948, 1995.

[11] A. E. Ertan, *Pitch-synchronous processing of speech signal for improving the quality of low bit rate speech coders*. PhD thesis, Dept. Elect. Eng., Georgia Institute of Technology, Atlanta, Georgia, 2004.

[12] D. Luenberger, *Optimization by Vector Space Methods*. John Wiley and Sons, Inc., 1969.

[13] A. Hyvärinen, "A family of fixed-point algorithms for independent component analysis," *In Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'97), Munich, Germany*, 1997.

[14] A. Hyvärinen, "New approximations of differential entropy for independent component analysis and projection pursuit," *In Advances in Neural Information Processing Systems 10*, 1998.

[15] A. Hyvärinen, "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Transactions on Neural Networks*, 1999.

[16] A. Hyvärinen, "Survey on independent component analysis," *Neural Computing Surveys 2*, 1999.

[17] A. Hyvärinen, "The fixed-point algorithm and maximum likelihood estimation for independent component analysis," *Neural Processing Letters*, 1999.

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

[19] L. M. A. Cichocki, R.E. Bogner and K. Pope, "Modified herault-jutten algorithms for blind separation of sources," *Digital Signal Processing 7*, 1997.

[20] B. Laheld and J.-F. Cardoso, "Adaptive source separation with uniform performance," *In Proceedings of EUSIPCO, Edinburgh, Scotland*, 1994.

[21] J. F. Cardoso and B. H. Laheld, "Equivariant adaptive source separation," *IEEE Trans. on Signal Processing 44(12)*, 1996.

[22] J. F. Cardoso, "Entropic contrasts for source separation," *In S. Haykin, editor, Adaptive Unsupervised Learning*, 1999.

[23] M. G. T.-W. Lee and T. J. Sejnowski, "Independent component analysis using an extended infomax algorithm for mixed sub-gaussian and super-gaussian sources," *Neural Computation*, 1998.

[24] P. G. D. T. Pham and C. Jutten, "Separation of a mixture of independent sources through a maximum likelihood approach," *In Proc. EUSIPCO*, 1992.

[25] H. O. E. Oja and J. Wangviwattana, "Learning in nonlinear constrained hebbian networks," *In Artificial Neural Networks, Proc. ICANN'91, Espoo, Finland*, 1991.

[26] N. Delfosse and P. Loubaton, "Adaptive blind separation of independent sources: a deflation approach," *Signal Processing, 45*, 1995.

[27] A. Hyvärinen and E. Oja, "Simple neuron models for independent component analysis," *Int. Journal ofNeural Systems 7(6)*, 1996.

[28] C. Fyfe and R. Baddeley, "Non-linear data structure extraction using simple hebbian networks," *Biological Cybernetics 72*, 1995.

[29] C. Fyfe and R. Baddeley, "Blind source separation using least-squares type adaptive algorithms," *In Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'97)*, 1997.

[30] J. Cardoso, "Eigen-structure of the fourth-order cumulant tensor with application to the blind source separation problem," *In Proc. ICASSP'90, Albuquerque, NM, USA*, 1990.

[31] J. F. Cardoso, "Super-symmetric decomposition of the fourth-order cumulant tensor. blind identification of more sources than sensors," *In Proc. ICASSP'91*, 1991.

[32] J. F. Cardoso and A. Souloumiac, "Blind beamforming for non gaussian signals," *IEE Proceedings-F, 140(6)*, 1993.

[33] J. F. Cardoso and P. Comon, "Independent component analysis, a survey of some algebraic methods," *In Proc. ISCASP'96*, 1996.

[34] D. H. Wolpert and W. G. Macready, "No Free Lunch theorems for optimisation," *IEEE Transactions on Evolutionary Computation*, 1997.

[35] C. S. M. D. Vose and L. D. Whitley, "The no free lunch and problem description length," *Proceedings of the Genetic and Evolutionary Computation Conference, SanFrancisco, USA*, 2001.

[36] M. D. E. Bonabeau and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, Santa Fe Institute Studies in the Sciences of Complexity, 1999.

[37] M. M. Millonas, "Swarms, phase transitions and collective intelligence," *Artificial Life III*, pp. 1942–1948, 1994.

[38] R. Axelrod, *The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*. Princeton University Press, 1997.

[39] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*. Morgan Kaufmann Publishers, 2001.

[40] *Proceedings of International Congress on Evolutionary Computation, Washington, US*, pp. 1939–1944, 1999.

[41] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," *Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan*, 1995.

[42] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," *Evolutionary Computation VII: Proceedings EP 98*, pp. 561–600, 1998.

[43] Y. J. F. T. B. Liu, L. Wang and D. Huang, "An improved particle swarm optimization combined with chaos," *Chaos, Solition and Fractals*, 2005.

[44] E. Ozcan and C. Mohan, "Surfing waves," *Proceedings of International Congress on Evolutionary Computation, Washington, US*, pp. 1939–1944, 1999.

[45] R. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," *Proceedings of Congress on Evolutionary Computation, La Jolla, CA*, 2000.

[46] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, 2002.

[47] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," *IEEE International Conference on Evolutionary Computation, Anchorage, Alaska*, 1998.

[48] Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," *Proceedings of International Congress on Evolutionary Computation, Seoul, Korea*, 2001.

[49] J. Kennedy, "Small worlds and mega minds: effects of neighbourhood topologies on particle swarm performance," *Proceedings of International Congress on Evolutionary Computation, Washington, US*, pp. 1931–1938, 1999.

[50] D. J. Krusienski and W. K. Jenkins, "Nonparametric density estimation based independent component analysis via particle swarm optimization," *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, Philadelphia, PA, USA*, March, 2005.

[51] S.-M. C. C.-H. Y. Du-Ming Tsai, Yan-Hsin Tseng, "An independent component analysis-based filter design for defect detection in low-contrast surface images," *Pattern Recognition, 39*.

[52] J. W. Lei Xi and, "Global optimal ica and its application in meg data analysis," *Neurocomputing, Volume 69, Issues 16-18*, October, 2006.

[53] J. K. M. Clerc, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. on Evolutionary Computation, vol.6*, February 2002.

[54] D.-M. Tsai and S.-C. Lai, "Independent component analysis-based background subtraction for indoor surveillance," *IEEE Transactions on Image Processing, Volume 18*, January, 2009.

[55] Y. Zhang and Y. Zhang, "Fault detection of non-gaussian processes based on modified independent component analysis," *Chemical Engineering Science, Volume 65*, May, 2010.

[56] R. L. J. M.-B. J. Igual, J. Ababneh and V. Zarzoso, "Solving independent component analysis contrast functions with particle swarm optimization," *ICANN 2010, Part II*, May, 2010.