Studies On Independent Component Analysis For Watermarking And Nonlinear Blind Source Separation

Nguyen Viet Thang

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2007

Acknowledgements

First and foremost, I would like to express my gratitude and sincere appreciation to my supervisor, Dr. Jagdish Chandra Patra, for his guidance, continuous encouragement, thoughtful criticism and invaluable suggestions. Without his support and invaluable patience, the thesis would not be possible. He has also always kept on pushing me to advance to higher research levels.

It is also my pleasure to thank and acknowledge the technicians and graduate students in the Centre for Computational Intelligence (C2I) for their helpful support and encouragement. I would like to thank the Nanyang Technological University (NTU) for providing me the financial support to do research in Singapore. I want to thank all my friends and colleagues, who assisted me in many ways throughout the duration of the research.

Last but not least, my deepest gratitude to my parents, my brother and my beloved for their love, support and patience. I would not have had these days without their endless encouragement.

Table of Contents

List of Tables

List of Figures

Abbreviations and Symbols

We use uppercase boldface letters to denote matrices, lowercase boldface letters for vectors, and lowercase letters for their entries. For example, a vector $\mathbf{x}$ with m elements is given by $\mathbf{x} = [x_1, x_2, ..., x_m]^T$, and a matrix of size $m \times n$ is given by $\mathbf{A} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]^T$ where $\mathbf{x}_i = [x_{i1}, x_{i2}, ..., x_{im}]^T$. Uppercase letters are used for the images (in the digital image watermarking section). Acronyms and specific notations frequently encountered in the thesis are listed below.

BSS	Blind Source Separation
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DWT	Discrete Wavelet Transform
DWM	Digital Watermarking
FA	Factor Analysis
HVS	Human Visual System
IC	Independent Component
ICA	Independent Component Analysis
KL	Kullback-Leibler divergence
LMS	Least Mean Square
MLP	Multi Layer Perceptron
NVF	Noise Visibility Function
PCA	Principal Component Analysis
pdf	Probability density function
PNL	Post Nonlinear Mixture
PP	Projection Pursuit
PSNR	Peak Signal to Noise Ratio
WMicaT	Watermarking by ICA using image Transpose

WMicaD	Watermarking by ICA using Dual watermarks
$\lfloor \cdot \rfloor$	Floor operator
$\bullet$	Element-by-element product
$\alpha, \beta, \gamma, \delta, \sigma, \lambda$	Watermark embedding parameters
$\mathbf{A}$	Mixing matrix
$\mathbf{A}^T$	Transpose of matrix $\mathbf{A}$
$\mathcal{C}_{1\to 2}$	1-dimension to 2-dimension converter
$\mathcal{C}_{2\to 1}$	2-dimension to 1-dimension converter
$\mathcal{D}$	Down-sizing operator
$E[\cdot]$	Expectation
$\mathcal{E}_i$	kPNL linear extraction module
$\mathcal{F}$	Nonlinear multivariate mixing function
$\mathcal{G}$	Nonlinear multivariate demixing function
$\mathcal{H} = \mathcal{G} \circ \mathcal{F}$	Global transformation
$H(\cdot)$	Differential entropy
$I(\cdot)$	Mutual information
I, I^+, I^*	Original image, watermarked image, test image
$J(\cdot)$	Negentropy
$\kappa_i(\cdot)$	i-th order cumulant
K, K_P	Key image, public image
$\mathcal{L}$	kPNL cost function
$\mathcal{M}_1, \mathcal{M}_2$	Watermark modification functions
$\mathcal{N}_i$	kPNL neural network component
$\overline{p_1 p_2}$	Line connecting points p_1 and p_2
$r, \lvert r \rvert$	Correlation coefficient, absolute correlation coefficient
$\mathbf{s} = [s_1, s_2, ..., s_n]^T$	Unknown original source signals
S	Signature image
$\mathcal{S}$	Graph in a multi-dimensional coordinate system
$\mathcal{U}$	Up-sizing operator
$\mathcal{V}$	Visual mask generator
V	Visual mask
W	Watermark

$\mathbf{W}$	Demixing/Separating matrix
$\mathbf{x} = [x_1, x_2, ..., x_n]^T$	Observed signals (observations)
$\mathbf{y} = [y_1, y_2, ..., y_n]^T$	Extracted/estimated signals
$\mathbf{z}^i = [z_1^i, z_2^i, ..., z_n^i]^T$	input signals of the i-th linear extraction module in kPNL
$\mathbf{z} = [z_1, z_2, ..., z_n]^T$	gPNL linearized signal

Abstract

Independent Component Analysis (ICA) is one of the important methods in statistics and signal processing that deals with the data representation (transformation) problem. From the observed data, the goal of ICA is to estimate the mixing system and the underlying components that create those observed data. The most interesting thing is that ICA does it 'blindly' without prior knowledge of either the sources or the mixing system. Due to these attractive characteristics, ICA has been applied in many fields of science and engineering, for example, noise reduction, biomedicine, audio systems, telecommunication, and many others.

In this thesis, we first propose a novel application of linear ICA in an emerging field, *i.e.*, the Digital Image Watermarking. The basic idea of applying ICA to digital watermarking is to consider the original image and the watermarks as independent data sources. The watermarked image then becomes a mixture of these sources and therefore, the watermark extraction can be carried out by ICA techniques. From this idea, we develop and contribute two algorithms for image watermarking: WMicaT (WaterMarking by ICA using image Transpose)and WMicaD (WaterMarking by ICA using Dual watermarks). The former exploits the independence between the image and its transpose to develop a simple and robust watermarking technique. The latter algorithm makes use of some special properties of the watermarks to develop a flexible and computational effective watermarking method. In addition, WMicaD is not only robust to various attacks but also able to detect the tampered area on the attacked image.

Secondly, in order to extend ICA model to cover a broader class of applications, nonlinear ICA is studied for Blind Source Separation (BSS) problem. Following the well-known Post Nonlinear (PNL) mixture model, two algorithms named kPNL (kurtosis-based PNL) and gPNL (geometric PNL) have been developed. The first

algorithm, kPNL, utilizes a modified unsupervised Multi-Layer Perceptron (MLP) network with kurtosis measurement to guide the learning process. This algorithm provides a simple sequential extraction of the source signals that does not require any knowledge about the number of underlying components. It is useful for the applications that only need to extract few most important components, for example, in the case of dimension reduction. The second algorithm, gPNL, is developed to deal with hard nonlinear problems. The gPNL algorithm employs a two-stage separating system using geometric properties of the mixtures to eliminate the nonlinearity in the first stage. Then a basic linear ICA algorithm is applied in the second stage to extract the unknown components. With the geometric two-stage approach, gPNL provides a good performance in extracting hard nonlinear mixtures, and allows users to choose a suitable linear ICA method that yield the best performance for their applications.

Chapter 1

Introduction

1.1 Independent Component Analysis

The problem of finding a good representation of multivariate data by a suitable transformation has always been an important study in statistics, signal processing and data analysis [45, 87, 92, 133]. The presentation is used for revealing certain essential characteristics of the underlying components. Consequently, it facilitates the data analysis in pattern recognition, feature extraction, data compression and so on [8, 14, 20, 23, 32].

A representation method generally assumes that the observed data is a mixture of some underlying meaningful components. Mathematically, it can be modeled as

$$\mathbf{x} = \mathcal{F}(\mathbf{s}) \tag{1.1.1}$$

where $\mathbf{x} = [x_1, x_2, ..., x_n]^T$ is a vector of the observed data (signals), $\mathcal{F}$ represents a multivariate transformation function, and $\mathbf{s} = [s_1, s_2, ..., s_m]^T$ is a vector of hidden components (variables) being determined. Currently, most of the representation methods are carried out on linear transformation because of its computational and conceptual simplicity. A linear transformation is written as

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{1.1.2}$$

where $\mathbf{A}$ denotes a mixing matrix that is supposed to be found.

Data representation can be applied in various areas such as signal processing [50, 95], pattern recognition [190], image processing [45, 136], biomedicine [119], face recognition [22, 186] and so on. Hence, because of its importance, many approaches have been proposed for data representation such as Factor Analysis (FA) [77, 114],

Principal Component Analysis (PCA) [102, 103], Projection Pursuit (PP) [53, 70, 71], and especially Independent Component Analysis (ICA) [45, 51, 92].

The classical Principal Component Analysis is aimed for reducing the dimension of data with the least loss of information. Using second-order statistical properties, PCA projects the multi-dimensional original data into a sub-space of orthogonal Principal Components (PCs) while optimizing the data reconstruction. Another method that is closely related to PCA is the FA. It also uses second-order statistics to search for the factors (latent variables) that can facilitate the interpretation of the data. FA can be extended to recover the noisy representation by adding noise component into the model (1.1.2). The PP technique is different from FA and PCA as it applies higher-order statistics to find 'interesting' projections. PP can be used for optimal visualization, density estimation, regression or dimension reduction [71, 85].

Independent Component Analysis is an emerging technique for representing data by Independent Components (ICs) [6, 45]. The goal of ICA is to eliminate the dependency in the observed data by transforming the observations into a data with components that are statistically independent from each others. It is rooted from an argument that independent variables should have the most interesting statistical properties because each component does not reveal any information of the others. In contrast to correlation-based transformations such as PCA, ICA not only decorrelates the data (second-order statistics) but also reduces higher-order statistical dependencies. The development of ICA is closely related to Blind Source Separation (BSS) problem. The goal of BSS is to recover the hidden source signals only from their mixtures that are recorded through the sensors, *i.e.*, the observed signals. Since in BSS problems, it is usually assumed that the sources are independent, ICA can be applied in BSS problems.

A classic example for ICA and BSS is the 'cocktail party' problem in Fig 1.1. Imagine that there are m people chatting simultaneously in a music hall, and there are n microphones around the room to record the voices. The mission here is how to retrieve back the speech of each person from the mixture signals recorded by the microphones. The problem is more challenging when nonlinear mixture, time delay and noises are taken into account.

Until now, linear ICA is the most extensively-studied subject since its model is

Figure 1.1: The 'cocktail party': an illustration of the ICA and BSS problem.

simple and its algorithms are computational effective. The linear ICA model is good enough in several applications such as in signal processing, image processing, communication, biomedicine, feature extraction, pattern recognition, data compression, and so on [20, 22, 115, 182, 221]. In many other applications, however, there is a need of a more general model, *i.e.*, the nonlinear ICA model, but the problem is quite challenging because of the complexity and indeterminacies of nonlinear ICA [106, 107]. Therefore, studies on nonlinear ICA are being carried on in parallel with finding of new applications for ICA.

1.2 Watermarking by ICA

Recently, digital watermarking has gained popularity together with the explosion of the Internet and digital multimedia [54]. Watermarking is an effective tool to protect the ownership and to authenticate digital objects by embedding visible or invisible special marks of the owner (called the Watermarks) on the original object (so called the Work). The incredibly fast growth of online distribution and digital production have made copying of a Work easier than ever. In addition, the degradation in quality, which was always happened in the hard copying process, does not happen with

digital copying. The duplication of a digital Work is perfectly the same as the original. The good news is that it facilitates the distribution, but the bad news is that pirating becomes a critical issue. A normal person can do an illegal copy without any special tool or advanced skill. Hence, the owners of the digital objects are looking for effective techniques that can protect their copyright and ownership. In this regard, digital watermarking becomes one of the important studies that gain attention of researchers all over the world. Watermarking can be found in various applications like broadcast monitoring, owner identification, ownership validation, transaction tracking, copy control and content authentication [116,179,223]. During the last ten years, there have been a number of methods proposed for digital watermarking using various techniques such as Least Significant Bit (LSB) check-sum, Quantization Index Modulation (QIM) [41], Fourier transform [55], Wavelet transform [121], and so on. However, because of different requirements for watermarking, the problem of finding an effective method is still open for further researches.

Because of effective separation capability of ICA, it has a lot of potential in watermarking applications. The general idea is to consider the schemes in watermarking as the processes of ICA. The embedding scheme where the watermarks are inserted into the Work is considered as mixing of independent components (the Work and the Watermarks), and the extraction scheme where the watermarks are recovered is considered as the separation process in ICA. Hence, by applying ICA method on the watermarked object, we can extract the watermark effectively. Preliminary experimental result in [32, 75, 195, 230] is very promising, and it encourages researchers to have further studies on applying ICA for watermarking. Therefore, we would like to propose our ICA study on digital image watermarking as a small contribution to the growth of ICA applications.

1.3 Nonlinear ICA

Nonlinear ICA is a natural extension of the basic linear model. The reason is simple, linear ICA has its computational advantage but its simple model is not adequate to describe the real-world applications whose environment usually contains nonlinearity.

The demand for an extended version of ICA that can deal with nonlinearity has attracted many researchers to develop effective nonlinear algorithms. Early approaches for general nonlinear ICA were introduced in [34, 60, 110, 161]. The goal is to estimate a multi-dimensional nonlinear function $\mathcal{G}$ so that the components of its output $\mathbf{y} = \mathcal{G}(\mathbf{x})$ are statistically independent with each others. The estimation, however, is not simple. The computational complexity increases exponentially with the increase of the dimension of data. Moreover, unlike linear ICA where the solution is unique, there always exists more than one solution for nonlinear ICA [88,107]. Hence, in order to provide a unique solution, some additional constraints are imposed.

One of the most popular constrained nonlinear models is the Post Nonlinear (PNL) mixtures model introduced in [202]. It restraints the nonlinearity to a set of one-dimensional nonlinear functions f_i. Assuming that there are m sources and n observations, the PNL mixing model is represented as

$$x_i = f_i(\sum_{j=1}^{m} a_{ij} s_j) \qquad i = 1, 2, ..., n \tag{1.3.1}$$

where x_i are the observed data, s_j are the hidden components and a_{ij} is the (i, j)-th element of the mixing matrix $\mathbf{A}$. The complexity of the model is simple enough for computation, and the most important thing is that the PNL gives a unique solution. With these advantages, PNL model has gained widespread attention in many ICA studies. Different approaches have been proposed [1, 202, 237] and applications have been found [119, 164, 181]. The second part of the thesis highlights our contribution in the nonlinear BSS problem.

1.4 Major Contributions

Our contributions include two parts: the first contribution is about application of ICA for digital watermarking for image, and the second contribution is the development of new ICA algorithms that deal with nonlinear BSS problem. Major contributions in this thesis are listed below:

WMicaT: *ICA-based Watermarking using image transpose* provides a robust method for watermarking. The watermark and its transpose are used as the independent components so that ICA can be applied for extraction of the watermark. The

additional information needed for watermark extraction is the mixture of the original image and its transpose, and it may be made publicly available.

WMicaD: *ICA-based Watermarking using two watermarks* provides another approach for watermarking. The algorithm applies a two-watermark scheme in embedding and extraction, and exploits the special characteristics of the two watermarks when using down-sampling and up-sampling operations on them. The result is a robust watermarking method with computational advantage. This method can be used for checking the ownership, tracking the copy number and image authentication.

kPNL: *Post nonlinear mixture ICA by Kurtosis.* This kurtosis-based algorithm is extended from the kurtosis-based linear ICA algorithm. To deal with the nonlinearity, we modify a classical Multi-Layer Perceptron (MLP) network to an unsupervised MLP network. The whole learning process is guided by the kurtosis measurement. With kPNL method, hidden sources are extracted sequentially and therefore, we do not need any prior information about the number of sources (which is usually assumed to be known in most ICA algorithms).

gPNL: *Geometric post nonlinear ICA* aims to deal with hard nonlinear problems. The gPNL is tailored to PNL model and clearly divides the separation scheme into two independent stages: the linearizing stage whose goal is to eliminate the nonlinearity, and the linear demixing stage that carries out the separation on linearized signals. Geometric transformations are applied in the first stage so that it can linearize the hard nonlinear signals. Then in the second stage, any classical linear ICA can be applied to extract the unknown signals. The gPNL, therefore, provides a flexible and effective method for nonlinear ICA.

1.5 Organization of The Thesis

The thesis is organized as follows. Chapter 2 introduces ICA problem and explains its relationships with other statistical representation techniques. We briefly review the classical linear ICA approach and applications of the ICA. At the end of the chapter, we introduce a generalized model of linear ICA, the nonlinear model.

In Chapter 3, we investigate one novel ICA application, ICA for digital image watermarking. We begin with the explanation of watermarking problem under ICA point of view. After that, we propose our two novel ICA-based watermarking methods, named WMicaT and WMicaD. The first one, WMicaT, applies an image transpose technique in the embedding and extraction processes to construct a robust watermarking algorithm. The second one, WMicaD, applies a two-watermark embedding scheme to deliver a method that can do both robust watermarking and authentication.

In Chapter 4, we approach the ICA for nonlinear BSS problem in detail, analyze its difficulties and introduce two novel algorithms. The Kurtosis-based algorithm, named as kPNL, is extended from a sequential linear ICA method using a modified unsupervised MLP networks. The other algorithm, named as gPNL, approaches the nonlinear BSS problem from geometric viewpoint and derives a robust two-stage extraction scheme. Finally, Chapter 5 provides conclusion about the work and discussions on the promising directions for further studies.

Chapter 2

Independent Component Analysis: A Review

Independent Component Analysis (ICA) is a method for finding hidden components from the observed multidimensional data. In signal processing, ICA sometimes is referred as Blind Source Separation (BSS) method. ICA differs from the other representation (transformation) methods because it searches for the components that are statistically independent. In this chapter, we briefly review the definitions, applications, and algorithms for ICA.

2.1 ICA Problem

Before going into the introduction of ICA, let us take a brief review of the classical transformation and representation methods that are commonly used in signal processing, pattern recognition, data analysis and many other applications.

2.1.1 Classical Transformation and Representation Methods

Principal Component Analysis

Principal Component Analysis (PCA) [102,103] is probably the most well-known representation method in signal processing, statistics and neural computation. The details on PCA could be found in [102], in which PCA (also called the discrete Karhunen-Loève transform) is described as a decorrelation-based method, working effectively on Gaussian sources. The basic idea in PCA is to find m components $s_1, s_2, ..., s_m$ that can maximally represent the amount of variance, i,e, the content of data in term of

expectation. Mathematically, PCA can be defined by a recursive formulation. Let $\mathbf{w}_1$ be a vector representing the direction of the first principal component, then it is identified by

$$\mathbf{w}_1 = \arg \max_{\|\mathbf{w}\|=1} E[(\mathbf{w}^T\mathbf{x})^2] \tag{2.1.1}$$

where $\mathbf{x}$ is the random data vector and $E[\cdot]$ denotes the expectation. Assuming the first $k-1$ principal components have been identified, the direction vector of the k-th principal component is identified by

$$\mathbf{w}_k = \arg \max_{\|\mathbf{w}\|=1} E[(\mathbf{w}^T(\mathbf{x} - \sum_{i=1}^{k-1} \mathbf{w}_i\mathbf{w}_i^T\mathbf{x}))^2] \tag{2.1.2}$$

After that, the principal components are computed as $s_i = \mathbf{w}_i^T\mathbf{x}$. A practical way to calculate $\mathbf{w}_i$ is to take $\mathbf{w}_i$ $(i = 1, 2, ..., n)$ as the eigenvectors corresponding to n largest eigenvalues of the covariance matrix $\mathbf{C} = E[\mathbf{x}\mathbf{x}^T]$.

PCA is mostly used for dimension reduction. By choosing $m \ll n$ where n is the original dimensions, one can greatly reduce the data representation dimension and consequently reduce the computational workload. It has been shown that the PCA representation is an optimal linear dimension reduction solution in term of mean square [102]. However, PCA is based only on second-order statistics. The transformed data is uncorrelated but is not necessarily independent. Therefore, in many situations, PCA does not provides the best solution, especially in blind source separation [133].

Factor Analysis

Factor analysis [77, 114] is another second-order representation method similar to PCA. In factor analysis, the data is represented by

$$\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{n} \tag{2.1.3}$$

where $\mathbf{x}$ is the vector of the observed variables, $\mathbf{s}$ is the vector of the hidden variables, *i.e.*, the factors, which is assumed to be unknown, $\mathbf{A}$ is a constant matrix and $\mathbf{n}$ is noise. $\mathbf{s}$ and $\mathbf{n}$ are assumed to be Gaussian, and $\mathbf{s}$ usually has a lower dimension than $\mathbf{x}$. Hence, factor analysis is basically a dimension reduction method, and the algorithm developed for factor analysis is considered as a modification of PCA.

Projection Pursuit

PCA and factor analysis, nevertheless, are second-order statistic methods. Their primary goals, as discussed in [92], are 'to find a faithful representation of the data, in the sense of reconstruction mean square error'. On the contrary, the following higher-order methods aim 'to find a meaningful representation'. Although the definition of meaningfulness is very user-dependent, these higher-order methods are applied in many different type of applications [51, 70, 87]. In the following paragraphs, we briefly review some well-known higher-order methods such as Projection Pursuit (PP), redundancy reduction and blind deconvolution.

The goal of Projection Pursuit (PP) [53,70,71,85,103] is to find the interesting projection of data. It is mostly used for optimizing data visualization and for dimension reduction. Unlike PCA where the objective is to have a faithful dimension-reduced representation, PP tries to reduce the dimensions in such a way that the important (or so called interesting) features of the data are preserved [71]. The illustration in Fig. 2.1 shows the difference between PCA and PP. A two dimensional (2D) data in the figure is clearly separated into two clusters. The first principal component obtained by using PCA is in the vertical direction. It does not reveal the two-cluster structure of the data. Whereas, by looking into the projection derived from PP (the horizontal direction), one can easily observe this feature.

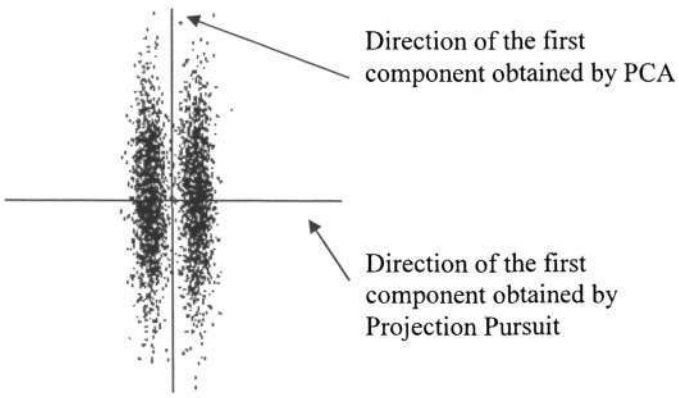


Figure 2.1: Illustration of the difference between PCA and Projection Pursuit (PP). The principal component derived from PCA (the vertical line) does not reveal any meaningful feature of the data while the projection made by PP (the horizontal line) is able to reveal the two-cluster structure of data [92].

The basic idea of finding 'interesting' direction in PP is to maximize the non-Gaussianity of the variables. As it is discussed in [85, 103], Gaussian distribution is a

least interesting one, and the direction that yields the most non-Gaussian distribution should be the most interesting direction. A common measure for non-Gaussianity is the differential entropy, $H(\mathbf{y})$, defined as

$$H(\mathbf{y}) = -\int p(\mathbf{y}) \log p(\mathbf{y}) d\mathbf{y} \tag{2.1.4}$$

where $p(\mathbf{y})$ denotes the probability density function (*pdf*) of the random vector $\mathbf{y}$. Assuming that $p(\mathbf{y})$ has zero-mean and fixed covariance, then $H(\mathbf{y})$ is maximized when $\mathbf{y}$ has Gaussian distribution. That is, the Projection Pursuit direction can be found by minimizing the differential entropy $H(\mathbf{w}^T\mathbf{x})$ with respect to $\mathbf{w}$ ($\mathbf{y} = \mathbf{w}^T\mathbf{x}$). In practice, however, the density of $\mathbf{w}^T\mathbf{x}$ that needed in (2.1.4) is hard to estimate. This problem can be overcome by some other non-Gaussianity measurements [53, 70], or by the approximations of the entropy [90, 103].

Redundancy Reduction

Redundancy reduction has been reported in literature [10, 17–19, 61, 67] as one of the important features of the sensory processing in the brain. To reduce redundancy, the data is represented by components that are as independent from each other as possible.

A common method for redundancy reduction is sparse coding [16, 19, 67]. The data is represented by a set of neurons such that only a small number of neurons is activated at any time. If the data is sparse in term of statistical properties, this representation method leads to approximate redundancy reduction [67]. A second redundancy reduction approach, called predictability minimization [192] is based on the observation that two independent random variables provide no information that could be used to predict one variable using the other one. Therefore, representing the data by independent components minimize the redundancy.

2.1.2 Linear Independent Component Analysis

We have reviewed some popular representation methods including two second-order ones and two higher-order ones. The ICA is another higher-order representation method that makes the representing features independent. In line with the conventional definition in many studies, the term ICA used in this chapter implies the linear

Independent Component Analysis.

Definition 2.1 Linear Independent Component Analysis of a random vector $\mathbf{x}$ consists of estimating the following generative model for the data

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{2.1.5}$$

where $\mathbf{x} = [x_1, x_2, ..., x_n]^T$ is a vector of n observed signals, $\mathbf{A}_{n\times m}$ is a real value matrix called mixing matrix, and $\mathbf{s} = [s_1, s_2, ..., s_m]^T$ is a vector of m hidden statistically independent variables (components). In the case of noisy ICA, the above model (2.1.5) is changed to

$$\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{n} \tag{2.1.6}$$

where $\mathbf{n}$ is a n-dimensional vector of random noise.

To estimate the hidden components s_i $(i = 1, 2, ..., m)$, the linear ICA methods try to find a linear transformation

$$\mathbf{y} = \mathbf{W}\mathbf{x} \tag{2.1.7}$$

where $\mathbf{W}$ is a matrix of size $m \times n$ so that the components y_i $(i = 1, 2, ..., m)$ of the output vector $\mathbf{y}$ are as independent as possible. However, the problem is how to measure the independence? As we know, components y_i are strictly independent if and only if

$$p(\mathbf{y}) = p(y_1)p(y_1)...p(y_m) \tag{2.1.8}$$

where $p(\mathbf{y})$ and $p(y_i)$ $(i = 1, 2, ..., m)$ are the *pdf* functions of $\mathbf{y}$ and y_i, respectively. This independent criterion, in practice, is very hard to measure and is usually replaced by some approximation measurements which will be reviewed in Section 2.2.

One of the important issues in ICA is the identifiability. As it was discussed in [40, 45, 51, 92], the following constraints should be imposed in order to guarantee the identifiability of ICA algorithm.

1. All the independent components s_i, with the possible exception of one component, must be non-Gaussian.

2. The number of observed mixtures n must be equal or larger than the number of independent components m.

3. The mixing matrix $\mathbf{A}$ must be a full column rank matrix.

These restrictions are important but not always necessary. Sometimes, it is still possible to estimate the non-Gaussian components even if there are more than one Gaussian variables. Also, there have been several works on over-complete ICA, *i.e.*, $n < m$, [38, 205, 207, 209], however, in most of ICA studies, the second restriction is imposed.

Having satisfied with the above constraints, independent components can be identified up to a scalar and a permutation [45, 87, 133]. That is, the estimated variables $y_1, y_2, ..., y_m$ is a permutation of the original variables $s_1, s_2, ..., s_m$ multiplied by some constants

$$\mathbf{y} = \mathbf{DPs} \tag{2.1.9}$$

where $\mathbf{D}_{m\times m} = diag[d_1, d_2, ...d_m]$ denotes a diagonal matrix and $\mathbf{P}_{m\times m}$ denotes a permutation matrix. In a simple example with two independent sources, s_1 and s_2, we can easily see that if $y_1 = -s_2$ and $y_2 = 10s_1$, then they are still independent of each other. Fortunately, the above-mentioned indeterminacies are not significant in most of the applications.

Finally, for the noisy ICA model (2.1.6), if the noise $\mathbf{n}$ is assumed to be independent with the source $\mathbf{s}$, then having the above three restrictions still assure the identifiability of the algorithm [92]. In addition, to facilitate the ICA extraction process in term of computation time and performance, observed variables and sources are usually assumed to be centered.

2.1.3 Relationships Between ICA and Other Methods

With the definition and formulation of ICA, one can see its relationship with the other representation methods. The connections between ICA and the others are illustrated in Fig. 2.2.

Firstly, by definition, ICA is certainly a kind of redundancy reduction. Several studies on neurophysiological data [26, 159] have shown that ICA provides the directions that is compatible with the actual redundancy reduction observed in experiments. Secondly, using the definition of noise-free ICA model (2.1.5), one can consider

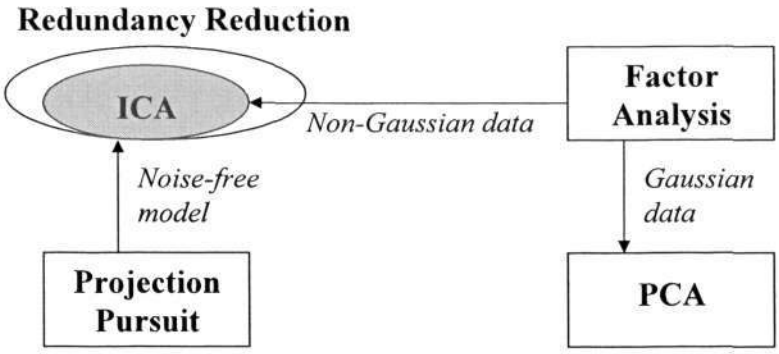


Figure 2.2: Relationships between ICA and some classical transformation methods, including Principal Component Analysis, Factor Analysis, Projection Pursuit and redundancy reduction [92].

ICA as a special case of Projection Pursuit. As we will see in Section 2.2, the criteria used in Projection Pursuit coincide with the criteria used in ICA for finding independent components.

Comparing with Factor Analysis, ICA can be considered as a non-Gaussian factor analysis. Equations (2.1.3) and (2.1.6) are the same except the fact that the hidden variables $\mathbf{s}$ in (2.1.3) are Gaussian while those in (2.1.6) are assumed to be non-Gaussian. Besides, ICA has a close relation with PCA because both methods need to formulate an objective function that define the interestingness of a linear representation and then maximize this function [92]. The difference is, however, PCA only uses second-order statistic to formulate the function while ICA applies higher-order statistic to define the objective function.

Finally, there is a close relationship between ICA and a well-known problem in signal processing, the Blind Source Separation (BSS) problem. The goal of BSS is to separate source signals from their mixtures without prior knowledge of the sources as well as the mixing system. It is proven that in the linear mixing situation, BSS and ICA problems are equivalent. In other words, ICA is a method to do Blind Source Separation.

2.2 Approaches for ICA Algorithms

The ICA problem is all about estimating the hidden variables by using independence property. First, we formulate an objective function and define a measure of independence, and secondly maximize (or minimize) the function by using the defined

measure. The most common approaches are non-Gaussianity, maximum likelihood, minimum mutual information, nonlinear decorrelation and nonlinear PCA.

2.2.1 ICA by Maximizing Kurtosis Measure

Kurtosis-based ICA is one of the methods that are tailored for the non-Gaussianity approach. The non-Gaussianity approach for ICA is rooted from the well-known Central Limit Theorem [163].

Theorem 2.1. *Let $s_1, s_2, ..., s_m$ be a set of m independent random variables and each s_i has an arbitrary probability distribution $p(s_i)$ with mean μ_i and a variance σ_i^2. Then the normal form variable*

$$\bar{s} = \frac{\sum_{i=1}^{m}(s_i - \mu_i)}{\sqrt{\sum_{i=1}^{n}\sigma_i^2}} \tag{2.2.1}$$

has a limiting cumulative distribution function which approaches a normal (Gaussian) distribution.

Intuitively speaking, Central Limit Theorem implies that the distribution of an average of the independent random variables tends to be more Gaussian than any distributions of these independent variables. And this is the basic idea to carry out ICA by maximizing Gaussianity. Using this Gaussianity-based approach, the hidden components will be extracted sequentially one after another.

From the definition of ICA, the observed vector $\mathbf{x}$ is a mixture of the hidden independent variables $\mathbf{x} = \mathbf{As}$ where $\mathbf{s} = [s_1, s_2, ..., s_m]^T$. Our aim is to estimate the hidden variables s_i.

An estimated data, denoted by y, is given by $y = \mathbf{w}^T\mathbf{x}$ where $\mathbf{w}$ denotes an n-dimensional demixing vector. Our goal is to find $\mathbf{w}$ so that y converges to one of the hidden variables, s_i. Using the model of $\mathbf{x}$ in (2.1.5) we have

$$y = \mathbf{w}^T\mathbf{As} = \mathbf{q}^T\mathbf{s} = \sum_{i=1}^{m} q_i s_i \tag{2.2.2}$$

with $\mathbf{q}^T = \mathbf{w}^T\mathbf{A}$. Clearly, y is a combination of the m independent variables, $s_1, s_2, ..., s_m$. From the Central Limit Theorem, we know that the distribution of y is more Gaussian than any of the variables, s_i, $i = 1, 2, ..., m$. Therefore, maximizing the non-Gaussianity of $y = \mathbf{w}^T\mathbf{x}$ with respect to vector $\mathbf{w}$ will make y converge to one of

the independent components, s_i. Since the methods tailored for this non-Gaussianity approach usually estimate the hidden components one by one, they are referred as sequential ICA extraction [48,49] or the one-unit contrast function methods [92]. The fundamental issue of this approach is how to measure the non-Gaussianity. Here, we review two major quantitative measures of non-Gaussianity: the kurtosis and negentropy.

Kurtosis is the fourth-order cumulant of a random variable. The kurtosis of a zero-mean random variable, y, denoted by $\kappa_4(y)$ is defined by [163]

$$\kappa_4(y) = E[y^4] - 3(E[y^2])^2 \tag{2.2.3}$$

where $E[y^4]$ and $E[y^2]$ denote the fourth- and second-order moments, respectively. For a Gaussian variable, we have $E[y^4] = 3(E[y^2])^2$, *i.e.*, the kurtosis equals to zero. Consequently, a non-zero kurtosis implies that the variable is non-Gaussian and higher the absolute value of kurtosis, the more non-Gaussian the variable is.

Now we apply the definition of kurtosis to find the independent components. Our goal is to find a demixing vector $\hat{\mathbf{w}}$ that maximizes the kurtosis of the output. That is

$$\hat{\mathbf{w}} = \arg \max_{\|\mathbf{w}\|=1} \kappa_4(\mathbf{w}^T\mathbf{x}) \tag{2.2.4}$$

Kurtosis-based criterion has been used widely in many ICA methods [48,58,86,89] because of its simplicity and possibility of providing a global convergence [92, 93]. However, it has been argued in [6,48] that kurtosis is not a very good measure of non-Gaussianity. In many cases, the kurtosis-based algorithms do not provide accurate estimation. The reason is, kurtosis measure is not affected much by main structure in the middle of the distribution but it is highly sensitive to the outliers (values at the tail of the distribution) [85].

2.2.2 ICA by Negentropy Measure

From a fundamental result of information theory, it is shown that a Gaussian variable has the largest entropy among all random variables [120, 163]. It means entropy can be used as a measure of non-Gaussianity. The negentropy is one of the most widely used measures in the non-Gaussianity-based ICA methods.

First, let us introduce the definition of mutual information, an elemental quantity in information theory. The mutual information between m random variables $y_1, y_2, ..., y_m$ is defined by

$$I(y_1, y_2, ..., y_m) = \sum_{i=1}^{m} H(y_i) - H(\mathbf{y}) \tag{2.2.5}$$

where $\mathbf{y} = [y_1, y_2, ..., y_m]^T$, $H(y_i)$ and $H(\mathbf{y})$ denote the differential entropy (2.1.4) of the variables, y_i and $\mathbf{y}$, respectively. Naturally, mutual information is a measure of the dependence between random variables. It is always non-negative, and zero if and only if all the variables are statistically independent. Recalling that $\mathbf{y} = \mathbf{W}^T\mathbf{x}$, the following is derived from (2.2.5)

$$I(y_1, y_2, ..., y_m) = \sum_{i=1}^{m} H(y_i) - H(\mathbf{x}) - \log|\det \mathbf{W}| \tag{2.2.6}$$

Looking into (2.2.6), it seems that independent components can be found directly by minimizing the entropy $H(y_i) = H(\mathbf{w}_i^T\mathbf{x})$ with respect to $\mathbf{w}_i$. However, it is not that easy since the entropy is not invariant for scale transformation [62, 92] and therefore, suitable modifications are needed. In order to obtain an invariant version of entropy, *i.e.*, a measure of non-Gaussianity that is zero for a Gaussian variable and always non-negative, a normalized version of entropy, called negentropy, J, is defined as follows

$$J(\mathbf{y}) = H(\mathbf{y}_{gauss}) - H(\mathbf{y}) \tag{2.2.7}$$

where $\mathbf{y}_{gauss}$ is a Gaussian random vector of the same covariance matrix as $\mathbf{y}$. Negentropy is also non-negative and is zero if and only if $\mathbf{y}$ has Gaussian distribution. Combining the definition of negentropy (2.2.7) and mutual information (2.2.5), we get the following relationship [87, 92]

$$I(y_1, y_2, ..., y_m) = J(\mathbf{y}) - \sum_{i=1}^{m} J(y_i) + \frac{1}{2}\log\frac{\prod C_{ii}^y}{\det \mathbf{C}^y} \tag{2.2.8}$$

where $\mathbf{C}^y$ is the covariance matrix of $\mathbf{y}$, C_{ii}^y are its diagonal elements, and $i = 1, 2, ..., m$. An important property of negentropy is its invariance under linear transformations [51]. Therefore, finding maximum negentropy direction where elements of the sum $J(y_i)$ in (2.2.8) are maximized, is equivalent to the direction where mutual information is minimized, *i.e.*, the direction of the independent component.

The estimation of negentropy, however, is as difficult as the estimation of mutual information. One of the attempts for estimating negentropy is to use higher-order cumulants for approximating the negentropy. For example, negentropy can be approximated as [87, 103]

$$J(y_i) \approx \frac{1}{12}\kappa_3(y_i)^2 + \frac{1}{48}\kappa_4(y_i)^2 \tag{2.2.9}$$

where the random variable, y_i, $i = 1, 2, ..., m$, is assumed to be zero mean and unit variance. This approximation, in fact, often leads to the use of kurtosis as in the previous section. Hence, it also suffers from the same problem of inaccuracy and outlier sensitivity. To overcome this problem, a generalized approximation is introduced. In a special simple case, two nonquadratic functions G^1 and G^2 are used. These functions are selected such that G^1 is odd and G^2 is even. Then we have the following approximation [87]

$$J(y) \approx k_1(E[G^1(y)])^2 + k_2(E[G^2(y)] - E[G^2(y_{gauss})])^2 \tag{2.2.10}$$

where k_1 and k_2 are positive constants, y_{gauss} is a Gaussian variable of zero mean and unit variance. The random variable, y, is assumed to be zero mean and unit variance. Clearly, the negentropy measure in (2.2.9) is a particular case of (2.2.10) with $G^1(y) = y^3$ and $G^2(y) = y^4$. Using a suitable selection of the two functions G^1 and G^2, one can significantly increase the ICA performance.

2.2.3 ICA by Maximizing Likelihood Estimation

In the previous Sections, we have seen how independent components are estimated sequentially by using non-Gaussianity measures. The sequential extraction methods are usually simple but not robust, for example, it is sensitive to the outlier values. In the following, we review another approach that extract all the independent components simultaneously. Likelihood and mutual information are the commonly-used measures for the simultaneous extraction methods.

From the well-known results in probability and statistics, we can derive the following formula for the probability density of the observed data

$$p_x(\mathbf{x}) = |\det \mathbf{A}^{-1}| \prod_i p_i(s_i) \tag{2.2.11}$$

where $\mathbf{x} = \mathbf{As}$, $\mathbf{s} = [s_1, s_2, ..., s_m]^T$ and p_i denote the *pdf* of the independent component, s_i. Denoting the inverse matrix of $\mathbf{A}$ by $\mathbf{W}$ and assuming there are N observations of $\mathbf{x}$ denoted by $\mathbf{x}(1), \mathbf{x}(2), ..., \mathbf{x}(N)$, the likelihood can be obtained as

$$L(\mathbf{W}) = \prod_{t=1}^{N} \prod_{i=1}^{n} p_i(\mathbf{w}_i^T \mathbf{x}(t)) |\det \mathbf{W}| \tag{2.2.12}$$

where $\mathbf{w}_i$ $(i = 1, 2, ...n)$ denotes the column vector of $\mathbf{W}$. Because of the computational advantage, it is more practical to use the logarithm of the likelihood. Taking logarithm of the equation (2.2.12) and then average it, we have the following log-likelihood formula

$$\frac{1}{N} \log L(\mathbf{W}) = \sum_{i=1}^{n} E[\log p_i(\mathbf{w}_i^T \mathbf{x}(t))] + \log |\det \mathbf{W}| \tag{2.2.13}$$

Hence, by maximizing the log-likelihood (2.2.13), one can obtain the demixing matrix $\mathbf{W}$ and the independent components [72, 173, 176, 177].

Another very similar approach is the Infomax algorithm introduced by Bell and Sejnowski [25]. The method is derived from neural network viewpoint, using information maximization principle. Assuming $\mathbf{x}$ as the network inputs, the outputs of the network are of the form $y_i = \phi_i(\mathbf{w}_i^T \mathbf{x})$, $i = 1, 2, ..., n$, where ϕ_i are nonlinear scalar functions. The independent components are obtained by maximizing the entropy of the outputs

$$H(\mathbf{y}) = H(\phi_1(\mathbf{w}_1^T \mathbf{x}), \phi_2(\mathbf{w}_2^T \mathbf{x}), ..., \phi_n(\mathbf{w}_n^T \mathbf{x})) \tag{2.2.14}$$

Applying the formula of the entropy transformation on (2.2.14) yields

$$H(\mathbf{y}) = H(\mathbf{x}) + \sum_{i=1}^{n} E[\log \phi_i'(\mathbf{w}_i^T \mathbf{x}(t))] + \log |\det \mathbf{W}| \tag{2.2.15}$$

where ϕ_i' denotes the derivative of the network function, ϕ_i. Comparing equations (2.2.13) and (2.2.15), we can see that the two are very similar. Hence, if the network function ϕ_i is chosen as the cumulative distribution function corresponding to the *pdf*, p_i, *i.e.*, $\phi_i'(\cdot) = p_i(\cdot)$ then the entropy of $\mathbf{y}$ (2.2.15) is the same as the log-likelihood function (2.2.13). That is, the Infomax method is equivalent to maximum likelihood estimation [39,87,133]. In addition, it is shown that the maximum likelihood algorithm for ICA is almost identical to the nonlinear decorrelation algorithms introduced in [49].

2.2.4 ICA by Minimizing Mutual Information

Theoretically, minimization of mutual information is probably the most suitable optimizing method for simultaneous estimation of independent components [51]. The mutual information (2.2.6) is always non-negative and equal zero if and only if all the components (variables) are independent to each other. However, as stated in Section 2.2.2, it is difficult to estimate mutual information in practice, since we have to estimate all the density functions. In some works [7, 51], the authors approximate mutual information by applying polynomial density expansion which leads to the use of higher-order cumulants. For example, mutual information can be approximated by the following expression [7]

$$I(\mathbf{y}) \approx C + \frac{1}{48}\sum_{i=1}^{m}(4\kappa_3(y_i)^2 + \kappa_4(y_i)^2 + 7\kappa_4(y_i)^4 - 6\kappa_3(y_i)^2\kappa_4(y_i)) \tag{2.2.16}$$

where C is a constant, κ_i denotes the i-th order cumulant, and y_i are assumed to be uncorrelated. Another approximation can be carried out by negentropy as discussed in Section 2.2.2.

Minimization of mutual information has a close relation with other approaches. Considering the log-likelihood function in (2.2.13), if the approximation density function p_i is equal the actual *pdf* function of the original variables, then the first term in (2.2.13) is equal to the sum of the entropy of the original variables, that is

$$\sum_{i=1}^{n} E[\log p_i(\mathbf{w}_i^T\mathbf{x}(t))] = -\sum_{i=1}^{m} H(y_i)$$

The log-likelihood now can be rewritten as

$$\frac{1}{T}\log L(\mathbf{W}) = -\sum_{i=1}^{m} H(y_i) + \log|\det \mathbf{W}| \tag{2.2.17}$$

By looking at the two equations (2.2.17) and (2.2.6), one can clearly see that log-likelihood is equal to the negative of mutual information with an additive constant.

Furthermore, it has been discussed in Section 2.2.2 that maximizing negentropy is equal to minimizing mutual information. That is, the ICA approaches are closely related to each others [40]. For detail discussions on these relationships, please see [6, 45, 156].

2.2.5 ICA by Nonlinear Decorrelation

The nonlinear decorrelation approach is first introduced by Jutten and Herault in [105] using the principle of canceling nonlinear cross-correlation between variables. The basic principle is as follows. If y_i and y_j are two independent components under the assumption that y_i and y_j have symmetric density, then their nonlinear cross-correlation, $E[g_1(y_i)g_2(y_j)]$, is zero where g_1 and g_2 are two odd nonlinear functions. Often the nonlinear functions g_1 and g_2 are chosen according to the *pdf* of the independent components. Using this principle, the authors [105] derive the learning rule

$$\Delta w_{ij} \propto g_1(y_i)g_2(y_j), \qquad i \neq j \tag{2.2.18}$$

where $i, j = 1, 2, ..., n$, and g_1 and g_2 are two nonlinear functions. Output variables y_i are updated at every iteration as $\mathbf{y} = (\mathbf{I} + \mathbf{W})^{-1}\mathbf{x}$ with all the diagonal entries w_{ii} set to zero. After the convergence, the outputs, y_i, become independent components. However, the algorithm converges only under some strict restrictions [63].

To improve the performance of this nonlinear decorrelation approach, several algorithms have been introduced [35, 46, 49]. For example, in [49], the computation has been reduced by avoiding the matrix inversion

$$\Delta \mathbf{W} \propto (\mathbf{I} - g_1(\mathbf{y})g_2(\mathbf{y}))\mathbf{W} \tag{2.2.19}$$

where $\mathbf{y} = \mathbf{W}\mathbf{x}$ and the nonlinear functions g_1 and g_2 are applied separately on every component of $\mathbf{y}$.

2.2.6 ICA by Nonlinear PCA

The idea of this approach is to introduce the nonlinearity into the objective function used in PCA [109, 110, 158]. By adding a nonlinear function, g, into equation 2.1.1, we have

$$\mathbf{w}_1 = \arg \max_{\|\mathbf{w}\|=1} E[g(\mathbf{w}^T\mathbf{x})^2] \tag{2.2.20}$$

It is argued that if the nonlinearity is suitably chosen as the *pdf* of independent components, and data is sphered, then optimizing 2.2.20 will lead to the estimation of independent components. The idea of introducing nonlinearity into PCA model

can also be applied to other PCA objective functions. For example, the following simplified bigradient learning rule for nonlinear PCA has been introduced in [222]

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \mu(t)g(\mathbf{W}(t)\mathbf{x}(t))\mathbf{x}(t)^T + \alpha(\mathbf{I} - \mathbf{W}(t)\mathbf{W}(t)^T)\mathbf{W}(t) \qquad (2.2.21)$$

where $\mu(t)$ is the learning rate and α is a constant on the range $[0.5, 1]$. The nonlinear function g is applied separately on every component of the vector $\mathbf{y} = \mathbf{W}\mathbf{x}$, and the data is assumed to be sphered. More details on this algorithm can be found in [94,222].

2.2.7 ICA by Higher-Order Cumulant Tensors

The use of higher-order cumulant tensors is another approach for ICA [37,38,52,131, 147]. Tensors, by definition, can be considered as generalization of matrices. Cumulant tensors are then considered as generalization of covariance matrix. For ICA, the commonly used tensor is the fourth-order cumulant tensor. Denote $\text{cum}(x_i, x_j, x_k, x_l)$ as the fourth-order cumulant, then the fourth-order cumulant tensor is defined as a linear transformation from the space of $m \times m$ matrices to itself. The (i, j)-th element of the matrix given by the transformation is computed as

$$T_{i,j}(\mathbf{M}) = \sum_{k,l} m_{kl}\text{cum}(x_i, x_j, x_k, x_l) \qquad (2.2.22)$$

where m_{kl} is the (k, l)-th element of the matrix $\mathbf{M}$. Since it is a symmetric linear operator, the fourth-order cumulant tensor has eigenvalues that correspond to the eigenmatrices. By solving the eigenvectors for such eigenmatrices, we can obtain the independent components (ICs). Examples of the tensorial ICA methods include the Fourth-Order Blind Identification (FOBI) [36] and the Join Approximate Diagonalization of Eigenmatrices (JADE) [69] algorithms.

Advantage of the cumulant tensor-based approach is that it does not require knowledge of the probability density of the independent components. However, it also has some drawbacks. As the dimension of the data increases, the memory units needed for storing the tensors and the computational workload increase exponentially. Therefore, it is impractical to apply this approach for data with large dimension. In addition, the statistical properties of the estimators are not as good as those using non-polynomial cumulants or likelihood methods.

2.2.8 Other Algorithms

Besides the major methods reviewed in the previous Sections, there have been many other algorithms developed for ICA. They can be a modified and improved version of a basic algorithm, a generalized method that can work on different types of independent components, or a specific method that takes advantages of the particular patterns of the independent components. In the following paragraphs, we briefly introduce these three approaches.

As discussed in [92], the optimization algorithm, *i.e.*, the method to update the model, contributes a big part in the performance and convergence of an ICA method. Therefore, some authors tried to improve the algorithm performance by modifying the optimization technique. For example, in [5, 7], natural gradient has been used in replacement of the traditional gradient descent. An equivalent method using relative gradient was independently introduced in [35]. Approaching ICA from different viewpoints is also an interesting topic as found in many studies, such as ICA by Support Vector Machine (SVM) [184], ICA by Genetic Algorithms [43, 228], and geometric ICA [206, 208, 231].

In another approach, researchers attempt to deliver a generalized algorithm that is suitable for different types of independent components. Lee et al. developed a unifying framework that can apply on both sub-Gaussian and super-Gaussian variables [134, 135]. The FastICA algorithm [86, 89] applies fixed-point optimization to speed up the convergence, and a generalized objective functions to enhance the robustness.

Finally, the are several ICA methods that are tailored for specific kind of variables. The most commonly used feature is the time relation of the signals (variables). By using this temporal structure, one can obtain an algorithm with less complexity and computation. Examples of these algorithms can be found in [28, 57, 104, 112, 145, 236, 238]. Variation of ICA is also considered, for example, Tree-dependent Component Analysis (TCA) [13], ICA for stationary and non-stationary variables [174, 175], and dynamic online ICA [24].

In summary, linear ICA has been extensively studied with variety of different algorithms. Depending on the application environment, one can choose a suitable ICA method to take the full advantages of the algorithm for best performance.

2.3 Applications of ICA

2.3.1 ICA for Signal Processing

Signal processing is the area where ICA finds its major applications. In the signal processing field, ICA is usually classified under blind source separation/deconvolution category. In fact, the history of ICA goes together with the development of blind source separation.

The BSS is one of the emerging fields in signal processing. The goal of BSS is to recover original sources given only their observed mixtures. The term 'blind' indicates that we do not have any *a priori* knowledge about either the sources or the mixing system. To carry out the separation, most of the models employed in BSS assume that the original sources are independent. Therefore, under this hypothesis, BSS problem can be considered as a particular case of ICA. A classic example of BSS is the cocktail party. Theoretically, it equals to an ICA model where x_i, $i = 1, 2...$, is the record of the i-th microphone, and s_j, $j = 1, 2, ...$, is the voice of the j-th person. It is said that 'most of the advances in ICA have been motivated by the search for solutions to the BSS problem and, the other way around, advances in ICA have been immediately applied to BSS problem [185].

Practical applications of ICA in signal processing include speech, voice and signal separation [44, 115, 194, 231]. A variation of signal separation is noise reduction in which the noisy ICA models have been applied. In [182, 188], ICA model is proposed for wire and wireless communication applications. The ICA can also be applied for blind deconvolution by assuming that the source signal values $s(t)$ at different time instant t are non-Gaussian and statistically independent [100]. From a single source deconvolution, ICA can be extended to solve the problem of convolutive mixtures where several source signals with different time delay are mixed together [2, 210].

2.3.2 ICA for Feature Extraction

Motivated by the theory of redundant reductions, the basic idea is to consider the observations as a combination (mixture) of independent hidden features, and in such cases ICA can be applied to estimate these features. When modeled, the columns of

the mixing matrix $\mathbf{A}$ represent features, s_i represents the coefficient of the i-th feature that forms the observed data $\mathbf{x}$.

An important study in [159] has shown that ICA and sparse coding give equivalent methods for estimating features of natural images. It is also shown that the estimated features are closely related to those observed in the primary visual cortex [213, 214]. The results are reconfirmed in some other studies [26, 96, 108]. The similarity between features obtained by ICA and those obtained from Gabor analysis [91] and wavelet transformation [140] clearly show the ability of ICA in feature extraction. ICA for feature extraction can be found in various applications such as face recognition [22, 65, 186], image processing [20, 83, 98, 197], and others [15, 146, 190].

An emerging application that utilizes the feature extraction property of ICA is digital watermarking [32, 75, 195, 230]. At first, ICA is applied to the object (image, video, audio) to extract the independent components (ICs). The extracted ICs can be used to represent the objects. Next, watermarks are embedded into either the features or the coefficients. Finally, the modified ICs are recombined to produce the watermarked object. More details on Watermarking and ICA for Watermarking are provided in Chapter 3.

2.3.3 ICA for Biomedicine

ICA has been successfully applied to biomedical applications such as blind separation of electroencephalographic (EEG) and magnetoencephalographic (MEG) [180, 216]. In order to record the brain activities, EEG and MEG methods use the electrodes that are attached to the scalp to get the signals. The number of observed signals in EEG and MEG are 23 and 122, respectively.The recorded data are assumed to be instantaneous mixtures of independent source signals and therefore, ICA can be applied [87].

Besides the separation of signals, ICA also provides promising results for the important task of canceling the noise (artifacts) that are not corresponding to the brain signals [74, 215]. In addition, ICA has been used for decomposition of evoked field potentials and measurement of cortical flow patterns on EEG and MEG data [9, 217].

Functional Magnetic Resonance Imaging (fMRI) is another biomedical application

where ICA has been successfully applied [8, 30, 113, 143]. It is used to separate independent spatial activity patterns [143], independent temporal activation patterns or both [8,30]. More ICA applications for biomedicine can be found in [21, 191, 198, 224].

2.3.4 Other Applications

With the potential of blind source separation and projection pursuit, ICA can be applied to many other applications. For example, in [111, 136, 178], the authors have shown the ability to utilize ICA for image classification. In other studies, ICA is applied for analyzing weather data [23], stock portfolio [14], cash flow [117], and time series prediction [139, 170].

2.4 Nonlinear ICA

We have reviewed the linear ICA model and its applications. However, in many practical situations, this linear model is too simple to describe the observed data adequately. For example, microwave channels which include a filter and a nonlinear amplifier [181]. A natural way is to extend the basic linear ICA model to a nonlinear mixing model. A general nonlinear mixing model is formulated as

$$\mathbf{x} = \mathcal{F}(\mathbf{s}) \tag{2.4.1}$$

where $\mathbf{s} = [s_1, s_2, ..., s_m]^T$ is the vector of independent sources and $\mathbf{x} = [x_1, x_2, ..., x_n]^T$ is the vector of observed data. Here, the mixing matrix $\mathbf{A}$ of linear model (2.1.5) has been replaced by an unknown real valued n-component mixing function $\mathcal{F}$. The problem of ICA now becomes finding a mapping $\mathcal{G} : \mathbb{R}^n \to \mathbb{R}^m$ that gives the output

$$\mathbf{y} = \mathcal{G}(\mathbf{x}) \tag{2.4.2}$$

so that the components y_i of the vector $\mathbf{y} = [y_1, y_2, ..., y_m]^T$ are statistically independent. The solutions for the nonlinear ICA model in (2.4.2), however, is not simple. A fundamental characteristic of the nonlinear ICA problem is that in the general case, solutions always exist, and they are non-unique. One reason for this is that if y_1 and y_2 are two independent random variables, then for any two arbitrary functions

$g(\cdot)$ and $h(\cdot)$, $g(y_1)$ and $h(y_2)$ are also independent [88, 106, 107]. The existence and non-uniqueness of nonlinear ICA solutions will be discussed in detail in Chapter 4.

Due to the complexity and indeterminacy of nonlinear ICA problem, it is impossible to have a generalized effective algorithm. General nonlinear ICA is stated as an ill-posed problem and the choice of algorithm is depended on the environment where it is applied [3, 4]. The first approach to the general nonlinear problem was introduced in [161, 162] and then extended in different studies [4, 128, 226]. Other approaches try to reduce the indeterminacies by constraining the nonlinear model to a sub-system [11, 66]. The constraints are imposed either to the unknown sources or to the mixing systems. For examples, in [11], the source distribution is assumed to be bounded in a parallelogram.

An important special sub-system for nonlinear ICA is the post-nonlinear mixture (PNL) model. Mathematically, each observed data is formulated by

$$x_i = f_i(\sum_{j=1}^{m} a_{ij} s_j) \qquad i = 1, 2, ..., n \tag{2.4.3}$$

where s_j, $j = 1, 2..., m$, are the hidden variables, a_{ij} is the (i, j)-th entry of the mixing matrix $\mathbf{A}$, and f_i are the nonlinear functions. Under this scheme, the sources s_j are first mixed linearly, after that the nonlinear functions f_i are applied to them to obtain the final observations x_i. As it is shown in [202], the indeterminacies of PNL model are usually the same as those of a basic linear model. Thus, it makes PNL to be an attractive model. In addition, the post-nonlinearity assumption is useful and plausible in many applications such as sensor arrays [164], microwave channels [181] and biomedical systems [119]. In Chapter 4, we will come back to nonlinear ICA problem in detail.

Chapter 3

Independent Component Analysis for Digital Image Watermarking

Digital Watermarking (DWM) is being considered as an effective tools to protect the ownership and authentication of a digital intellectual Work. In recent years, many studies on DWM have been carried out, for example, theoretical background [54, 132, 148, 193], performance measurements [123, 172] and problem analysis [56]. It leads to the development of various techniques [41, 55, 79, 122, 187, 218] and the utilization of DWM in many practical applications [116, 179, 223]. In the following Section, we will give a brief introduction to DWM. Section 3.2 summaries different ICA-based approaches for watermarking. The next two Sections 3.3 and 3.4, we present details of our two ICA-based watermarking techniques. In the last Section, we comment on the merits and demerits of the two techniques, and the future directions of ICA for DWM.

3.1 Digital Watermarking: A Brief Introduction

Today in digital age, the growth of computers and the Internet has changed the ways people create and distribute the media [144]. Almost everything can be digitized, uploaded and downloaded without loss of quality and at a little cost. The Internet has become an excellent distribution system for digital media. It is inexpensive, maintains the quality, eliminates warehousing, and delivers the media almost instantaneously [54]. However, the content owners are facing a rapidly-increasing risk of piracy of their intellectual properties. The situation is getting worse by the development of high-capacity storage devices [54, 79].

Previously when the media content and intellectual property have always been contained in some instruments, duplication or modification were not easy and the quality of a pirated copy was usually lower than the original. Hence, the owners were benefited from their original work and were not affected much by piracy. However, things are totally changed in the era of digital technologies. Pirates can digitize the media easily with very little degradation, duplicate them without any quality loss and distribute them without paying to the actual owners. The content owners, therefore, eagerly need techniques that can protect their rights.

Cryptography is probably the first and most common method of protecting digital contents. An encrypted work can be safely transferred to the users and then be decrypted by a key that users purchased from the owner. However, once decrypted, the content is no longer protected. Thus, watermarking comes as an alternative solution because it includes the information within the content. The information can be designed to survive through all common processes like decryption, compression, re-encryption and conversion. The benefits of watermarking help it find place in many applications.

3.1.1 Watermarking Process

Watermarking, as defined in [54], is 'the practice of imperceptibly altering a Work to embed a message about that Work'. The term 'Work' here refers to a specific copy of a digital content, such as text, song, video or picture. The original unwatermarked Work is also referred as original data, cover data or cover work. The embedded message is termed as the Watermark and the Work after being embedded is referred as the watermarked Work or watermarked data.

An illustration of a watermarking process is shown in Fig. 3.1. Here, we follow the arguments in [144] to separate a watermarking process into four consecutive stages

1. The embedding stage.
2. The distribution stage.
3. The extraction stage.
4. The decision stage.

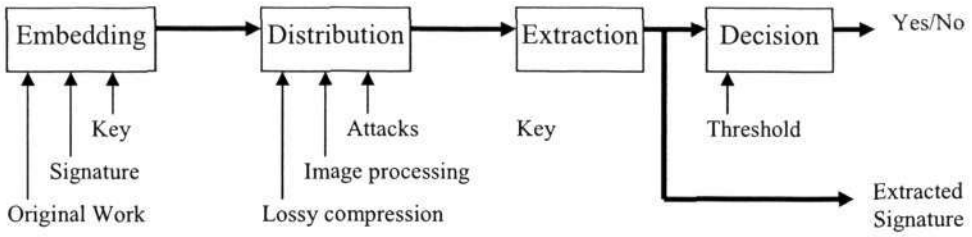


Figure 3.1: A general watermarking process includes four stages: embedding, distribution, extraction and decision.

Mission of the embedding stage is to insert the owner information into the Work to produce a watermarked version of the Work. The supplement data and security information that may need in the extraction are also created in this embedding stage. These additional data are referred as the Key.

When owners distribute the watermarked Works around, the Works may undergo many modifications due to lossy compression, transmission errors, image, audio, video processing and corrections, and especially the intentional attacks to remove the watermark.

Extraction stage is where we get back the watermarks from the Works with the help of all available information. The extracted watermarks are then shifted to the decision stage for analysis. Depending on the type of application, the outputs can be a simple Yes/No decision, the owner information or the tampered area in the Work.

The demands for watermarking are varied according to the characteristics of the applications. Here, we provide the basic requirements in watermarking [79].

1. A watermark shall convey as much information as possible, which means that the amount of the embedded data rate should be high.

2. A watermark should in general be secret and should only be accessible by authorized parties. This requirement is referred to as security of the watermark and is usually achieved by the use of cryptographic keys.

3. A watermark should stay in the host data regardless of whatever happens to the host data, including all possible modifications that may occur, and including all hostile attacks that unauthorized parties may attempt. This requirement is referred to as robustness of the watermark. It is an important requirement for copyright protection and conditional access applications. However, it is less

important for some applications where the watermarks are not required to be cryptographically secured, for example, for applications in which watermarks convey public information.

4. A watermark should, though being irremovable, be imperceptible.

3.1.2 Watermarking Classification

There are many ways to classify a watermarking technique in term of its application and purpose. In Table 3.1 we summarize a general classification of watermarking techniques which is viewed from different aspects as introduced in [132]. Details on these classifications are given below.

Table 3.1: Classification of watermarking techniques from different viewpoints.

Classification	**Characteristics**
Media type	text, image, video, audio
Perceptivity of watermark	visible, invisible
Robustness of watermark	robust, semi-fragile, fragile
Type of watermark	random sequence, image
Additional data	private, semi-private, public
Processing method	spatial domain, frequency domain

Classification by Media Type

Text watermarking and image watermarking have been being the most extensive studies to date. With the fast development of the Internet and large-scale storage devices, video and audio are the next two media that require effective watermarking techniques.

Classification by Watermark Perceptivity

The watermark can be embedded into the host data either visibly or invisibly. On one hand, visible watermarks are robust, *i.e.*, it is hard to be removed, and perceptually confirms the ownership. The visible watermark embedding is fast but it degrades

the quality of the original Work. On the other hand, invisible watermarks does not make noticeable modification on the Work but it is usually less robust than visible watermarks. The trade off between invisibility and robustness of the watermarks has been always an important topic for the researchers.

Classification by Watermark Robustness

Robustness is the most important factor in almost every watermarking techniques. Robust watermarks need to be strong enough to survive through various types of attack. Fragile watermark, in contrast, is designed to be vulnerable to any slightest modification on the watermarked Work. Fragile watermarking is usually used for authentication purpose. The third type of watermarking lies between robust and fragile watermarking. Semi-fragile watermarking can tolerate some degree of change in the watermarked Work, and it is capable of detecting the change in content of the Work.

Classification by Watermark Type

There are two types of content commonly used as watermarks. The first one, called noise type, is a random sequence, including pseudo noise, Gaussian noise and chaotic noise. The second one is a meaningful image, usually a logo or label.

Classification by Additional Data

Based on the availability of the additional data (the data needed for extraction), watermarking can be divided into three categories: public (blind) watermarking, semi-private watermarking and private watermarking. The first type requires neither original image nor watermark for the extraction. The second type does not need original image but needs the watermark. And the third type needs both original image and watermark.

Classification by Processing Method

Watermark embedding and extraction can be done either directly on spatial domain or on frequency domain (transform domain). The commonly used transform domain

include Discrete Cosine Transform (DCT), Discrete Fourier Transform (DFT) and Discrete Wavelet Transform (DWT). Spatial domain methods are robust against cropping and translation modifications while domain-transformed methods are generally robust against noise, compression and image processing.

3.1.3 Watermark Attacks

To verify the performance of a watermarking algorithm, one will apply modifications (so called attacks) to the watermarked Work and then try to extract the watermark from this attacked Work. With suggestions from [79, 123, 219], popular attacks can be categorized as shown in Table 3.2.

Table 3.2: Classification of attacks on digital watermarks.

Attacks	**Characteristics**
Removal attacks	completely remove the watermark information
Geometric attacks	impair the watermark detection
Cryptographic attacks	try to crack the watermarking method
Protocol attacks	create the ambiguity of the true ownership
Estimation-based attacks	estimate original work or watermark

Removal attacks aim to eliminate completely the watermark information so that users cannot recover the watermark from attacked Work. This type of attack includes denoising, quantization, remodulation and collusion attacks.

The goal of geometric attacks is not to remove the watermark information but to damage (distort) it so that the detector cannot recover the watermark. Geometric attacks include both global and local geometric distortions, such as rotation, scaling, shifting or jittering.

The third type of attack is cryptographic attack. It targets at cracking the watermarking schemes, finding a way to remove the embedded watermark or to embed a fake watermark. The Oracle attack and exhaustive search of watermark information belong to this kind of attacks [219].

In contrast of the above attacks which are trying to remove or impair the watermark information, the idea of protocol attacks is to create an ambiguity about the

true ownership of the Work. For example, attacker may add his own watermark to the watermarked Work, and then extract it and claim the Work to be his own.

The last kind of attack is called estimation-based attacks which is introduced by Voloshynovskiy et al. [219]. Attackers may exploit the knowledge of watermarking as well as the statistical property of original Work and watermarks in order to successfully estimate either the watermark information or the original Work. Estimation-based attacks can be classified into removal, protocol and desynchronization attacks.

3.1.4 Watermarking Applications

Probably watermarking can be applied to any place that needs a protection of intellectual property for digital media. Applications of watermarking are seen in every kind of digital media: text documents [33, 138, 142], images [42, 144, 160, 223, 233], video [64, 126, 199], audio [200, 211] and others [99, 116, 157].

Text Document Watermarking

Text watermarking finds its applications wherever digital documents are distributed, the digital library, for example [33, 142]. Most of the techniques are based on hiding the watermark in the layout and format of the documents. There are three common methods to embed the information: word-shift coding, line-shift coding and feature coding. In line-shift coding, single line is shifted either up or down by a small amount. Similarly, word-shift coding changes the space between two consecutive words [33]. Feature coding is somewhat different, instead of changing the distance between words or lines, it slightly modifies the shape of the characters. These methods, however, can be defeated by exhaustively retyping the entire document manually or automatically by using optical character recognition programs [138].

Image Watermarking

Until now, major watermarking studies and applications are for image since there are so many digital images circling all over the world that need a protection. As it is our goal to develop a novel method for image watermarking, details on digital image watermarking will be introduced in Section 3.1.5.

Video Watermarking

As video is a series of consecutive still image, the general problem of video watermarking is similar to that of still image. However, there also have some differences which are specific for video watermarking. First, video watermarking usually require real-time processing. Second, there are several particular attacks for video, for example, frame averaging, frame dropping or frame swapping. Watermarking on video, therefore, is much more complex [64, 199].

In order to reduce the complexity, several authors apply the watermark on compressed domain so that the embedding and extraction are compatible with the video encoding and decoding. For example, the DCT-based watermarking methods are compatible with the video DCT-based hybrid compression scheme: MPEG-2, MPEG-4, and H.263 [84, 125]. By doing this, the complexity of the algorithm is about the same with the video compression and decompression complexity. Some others try to reduce the computation workload by embedding the watermarks directly on spatial domain. Current state-of-the-art on video watermarking methods can be found in [79, 126].

Audio Watermarking

Most of the watermarking techniques for audio are done either directly on the audio signal or on the bit stream where the audio is represented in a compressed format. Audio watermarking requirements are quite similar to image watermarking, such as imperceptibility (inaudibility), robustness to signal modifications like compression, filtering, and Analog-to-Digital and Digital-to-Analog conversions. Several approaches for audio watermarking have been introduced such as spread spectrum technique, echo coding, and phase coding [31, 211, 227]. The Human Auditory System (HAS) is also considered in order to improve the watermarking methods [200].

3.1.5 Digital Image Watermarking

The number of publications on image watermarking is too large to completely cover in this Section. Here, we just point out the common approaches and ideas. The design of an algorithm include three issues: the embedded signal, the embedding stage and the extraction stage. Many image watermarking algorithms, in fact, differ only in

parts of these three issues [54].

The embedded information, the watermark, can be a random, pseudo-random data sequence or a meaningful image like the logo or trademark. In order to increase its invisibility, watermark is modified in shape so that it becomes less strong in areas of the image where it could be easily exposed. The watermark is often constructed in spatial domain, but sometimes also in transform domain like DCT transform domain. Depending on the application requirement, the amount of information that need to be embedded can vary widely [79].

The embedding process is an addition of the watermark to the host image, either on luminance or color channels. It can take place on spatial domain [124, 127] or transform domains such as full-image DCT transform domain, block DCT transform domain, DFT transform domain, and wavelet domain [55,144,168,169]. While watermarking on spatial domain is computationally advantageous, embedding on transform domain is usually claimed to be more invisible and secure. In practice, the embedding and watermark generating processes are normally treated as one, especially in those methods that use image dependent watermarks, *i.e.*, the watermarks are adapted to the content of the original image [218].

The extraction (or extraction and decision) stage is the place where embedded information is separated from watermarked image and then compared with the original watermark. Extraction is usually done in the same domain with the embedding stage and often need supporting data like original image or the Key. Based on availability of the original image, watermark extraction can be classified into blind, semi-blind or non-blind extraction. The Key can be classified into private, public or asymmetric key.

3.2 ICA-Based Approaches to Image Watermarking: A Review

The similarity between an ICA scheme and a DWM scheme can be seen in Fig. 3.2. The original image and watermarks can be seen as the inputs for ICA. The embedding process is then considered as an ICA mixing process that creates the watermarked image and key image from the inputs.

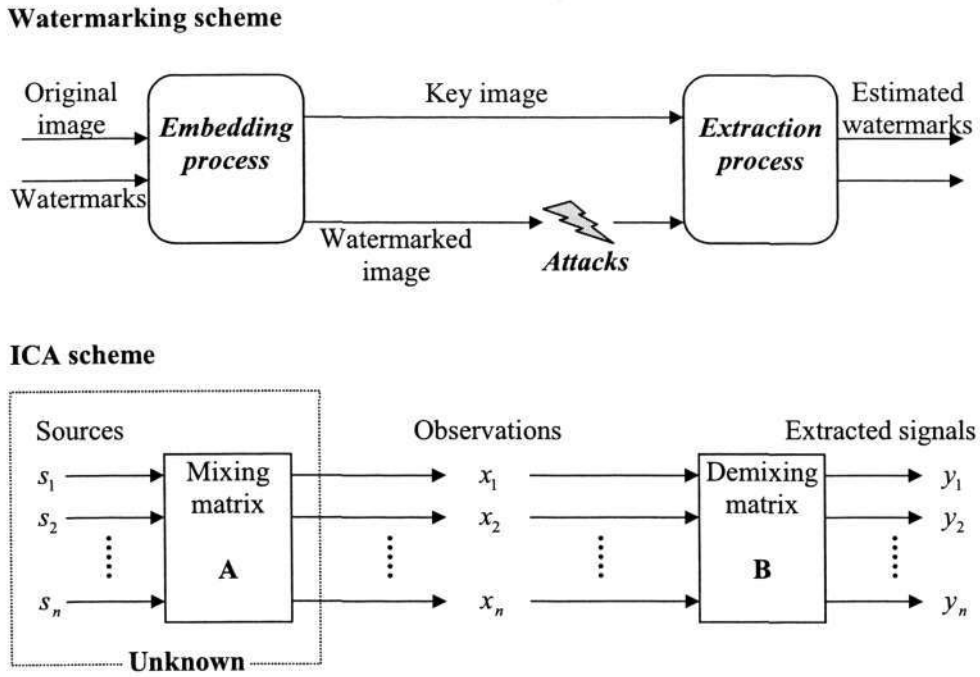


Figure 3.2: The similarity between an ICA scheme and a DWM scheme. From ICA point of view, the embedding process is similar to an ICA mixing system. Hence, watermark extraction can be carried out by an ICA demixing system.

At the extraction site, the estimation of watermarks can be carried out by an ICA demixing system with the watermarked image and key image are seen as the observations. In addition, the modifications made by the attacks on watermarked image can be seen as the noise that occurs during the transmission. Such analogy is the basic principle to develop an ICA-based watermarking algorithm. Until now, there have been several watermarking methods that utilize ICA techniques. We roughly classify such methods into two categories: the block-based methods and full image approach.

3.2.1 Block-Based Approach

In [26], Bell and Sejnowski proposed that the Independent Components (ICs) can be considered as the edge filter for a natural scene. That is, a natural image can be represented as a combination of the basis functions (the ICs). This approach is illustrated in Fig. 3.3.

The ICA block-based watermarking methods are based on the above principle. At first, the ICs are extracted using ICA and the original image is represented in term of these ICs. Next step, watermarks are embedded into some ICs and the modified ICs

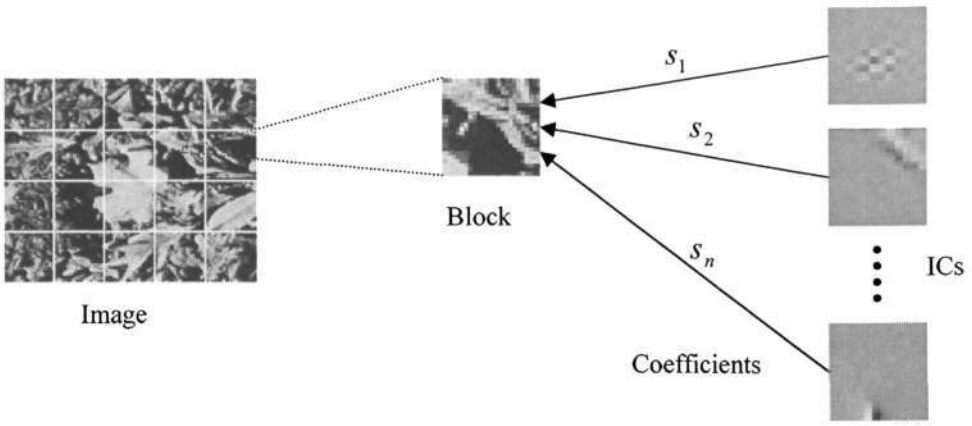


Figure 3.3: Representation of an image as the combination of the Independent Components (ICs). Original image is divided into blocks (patches) and each block is formed as a linear combination of the basis functions (the ICs).

are remixed to create the watermarked image. At the watermark extraction site, ICA is applied again on watermarked image to recover the ICs and then the watermarks are estimated. For example, in [75], the authors sorted the ICs by their energy, and then replaced the least significant ICs by the watermarks. In [32], the authors used a Quantization Index Modulation (QIM) technique [41] to embed the watermarks into the ICs.

With block-based approach, the watermarks are distributed all over the image. The watermarks, thus, are harder to be perceived. It does not require the original image and needs only few additional information to carry out the watermark extraction. Big disadvantages of this approach, however, are the need of a large number of ICs and the high computation workload. A solution to reduce the computation time is to lose the image quality by choosing only a few important ICs. Besides, as discussed in [26], using ICs as the edge filters is good for the natural images only. And finally, the ICA block-based watermarking methods are not very robust against different attacks [75].

3.2.2 Full-Image Approach

Another ICA approach for DWM, which we call full-image approach, considers original image, watermarks, key images (if any), each as an individual independent source signals. The watermarked image is a combination of these signals. A general scheme of the approach is shown in Fig. 3.4. The full-image approach is simpler and usually

more robust than the block-based one.

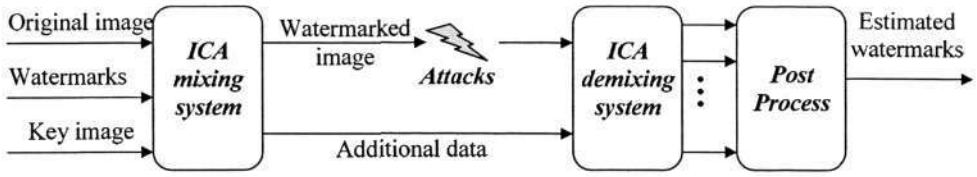


Figure 3.4: The ICA full-image watermarking scheme. The watermarked image is considered as a mixture of the original image, watermarks and key images.

In the ICA demixing model used in watermark extraction, the number of observed signals has to be equal or greater than the number of source signals. That is, besides the watermarked image, we need at least one more observation with a fairly large amount of data. Even more, many reported methods require the availability of original image at the extraction site. For example, in [195, 230], the authors used the original image, I, as well as a key image, K, to extract the watermarks. In other proposed schemes [137, 229], the original image is not required during extraction of watermark, however, a size-equivalent image is used.

In summary, if we can avoid the use of the original image or large data, the ICA-based full-image approach will provide a good solution for digital image watermarking. In the next Section, we propose two new algorithms: one uses image transpose to hide the original image and the other uses up/down-sizing technique to solve the issue of large additional data.

3.3 WMicaT: ICA-based Watermarking Using Image Transpose

We propose a novel method called WMicaT (Watermarking by ICA using image Transpose) that aims to eliminate the problem faced many ICA-based algorithms, *i.e.*, the need of original image during extraction process. By using the property of transpose of an image, WMicaT produces extra information called 'key image' during the embedding stage, and later uses this key image in the extraction stage. Unlike the original image which has to keep secretly, the key image can be made publicly available. A short report of this method has been published in [153].

3.3.1 Use of Image Transpose as An Additional Signal

Let us denote the original image by I, the watermark by W and the watermarked image by I^+. We consider an image as a square matrix of pixel intensities. The idea of using image transpose is emerged from the observation: in general an image is independent with its transpose, and the transpose of its transpose is the image itself. Considering watermarked image I^+ as a mixture of the original image and watermark, it can be expressed as

$$I^+ = I + \alpha W + \beta W^T \tag{3.3.1}$$

where α and β are scalar coefficients. Then its transpose is expressed by

$$(I^+)^T = I^T + \alpha W^T + \beta W \tag{3.3.2}$$

where T denotes transpose of a matrix.

Now we introduce the additional information, the key image K, as a linear combination of the original image and its transpose. That is

$$K = \gamma I + \delta I^T \tag{3.3.3}$$

where γ and δ are scalar coefficients. Its transpose, therefore, is expressed by

$$K^T = \gamma I^T + \delta I \tag{3.3.4}$$

In ICA terminology, I^+, I^{+T}, K and K^T are the four linear mixtures (observations) of the four original sources I, I^T, W and W^T. Hence, applying ICA on these four mixtures can produce the original sources. Note that as we assume that the original image and its transpose are independent, this scheme is not applicable if the original image is symmetrical.

3.3.2 WMicaT Embedding Scheme

A complete embedding scheme of WMicaT is shown in Fig. 3.5. From a small-sized image S that represents the owner signature, an initial watermark W_0 is created by tiling the signature to the size of the original image I. As discussed in [122, 218], in order to make the watermark more imperceptible and well embedded, we create a visual mask V from the original image I. Then we apply a modification function $\mathcal{M}$

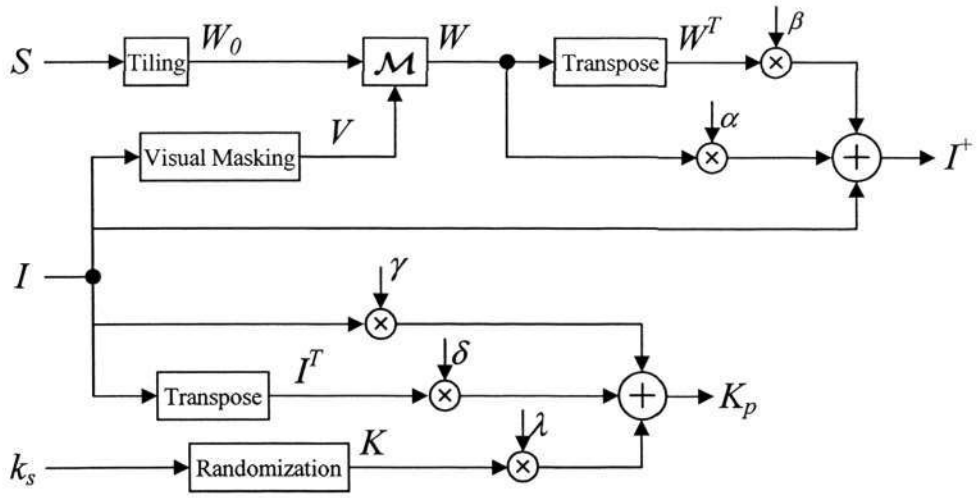


Figure 3.5: The WMicaT embedding scheme. The watermarked image I^+ and the public key K_P are generated from the original image I, the signature S, and a secret key k_s.

to the initial watermark and visual mask to get the final watermark W. The purpose of a visual mask is to identify the significant areas of the host image, *i.e.*, the texture and edge regions, in which the watermark can be more strongly embedded. With the help of a visual mask, one can increase the watermark strength considerably while maintaining the original image quality as well as the watermark imperceptibility. The details on different visual masking methods can be found in [122, 218]. Finally, the watermark W and its transpose W^T are inserted into the original image to form the watermarked image I^+ given by

$$I^+ = I + \alpha W + \beta W^T \tag{3.3.5}$$

where α and β are called 'embedding strengths'. The values of this two parameters will determine how strongly the watermarks are embedded.

The public image K_P, *i.e.*, the additional data needed during extraction process, is also built during the embedding process. It is a mixture of the original image, the original image transpose and a key image. The public image is computed by

$$K_P = \gamma I + \delta I^T + \lambda K \tag{3.3.6}$$

where K is the key image created using a pseudo random sequence with a seed k_s that satisfies $K = K^T$. The parameters γ, δ and λ are the 'key-image coefficients'. These parameters can be any non-zero values in the range of $[-1, 1]$. Without knowing the

seed k_s, one can not extract the original image I from public image K_P. The public image, therefore, can be made available online if required.

In this scheme, we apply a visual mask to adapt the watermark to the original image so that the embedded watermark has higher strength while remaining imperceptible. The visual mask is computed by using a Noise Visibility Function (NVF) [218]. The (m,n)-th entry of the visual mask V is obtained from the original image, I and is given by

$$V_{(m,n)} = \frac{1}{1+\sigma_I^2(m,n)} \tag{3.3.7}$$

where $\sigma_I^2(m,n)$ denotes the local variance of the image in a window centered on the pixel $I_{(m,n)}$. The local variance is calculated as

$$\sigma_I^2(m,n) = \frac{1}{(2L+1)^2}\sum_{i=-L}^{L}\sum_{j=-L}^{L}(I_{(i+m,j+n)} - \bar{I}_{(m,n)})^2 \tag{3.3.8}$$

where $\bar{I}_{(m,n)}$ denotes the mean of the pixels lying inside the window

$$\bar{I}_{(m,n)} = \frac{1}{(2L+1)^2}\sum_{i=-L}^{L}\sum_{j=-L}^{L} I_{(i+m,j+n)} \tag{3.3.9}$$

where a window of size $(2L+1)\times(2L+1)$ is used in the computation of $V_{(m,n)}$. Note that with this computation, the value of $V_{(m,n)}$ remains between 0 and 1.

The different steps involved in the embedding process are summarized below.

1. Generate the initial watermark W_0 from the author signature S by tiling it to the size of the image I.

2. Generate a visual mask V from the original image by (3.3.7).

3. Create the watermark W from W_0 and V using modification function $\mathcal{M}$ which is given by

$$W = W_0 - W_0 \bullet V \tag{3.3.10}$$

where '$\bullet$' denotes the element-by-element product, for example, the (m,n)-th entry of this product is given by $(W_0 \bullet V)_{(m,n)} = W_{0(m,n)}.V_{(m,n)}$.

4. Generate the watermarked image I^+ by embedding the watermark and its transpose to the original image using (3.3.5).

5. Compute key image K from a chosen seed k_s such that $K = K^T$.

6. Generate the public image K_p using (3.3.6).

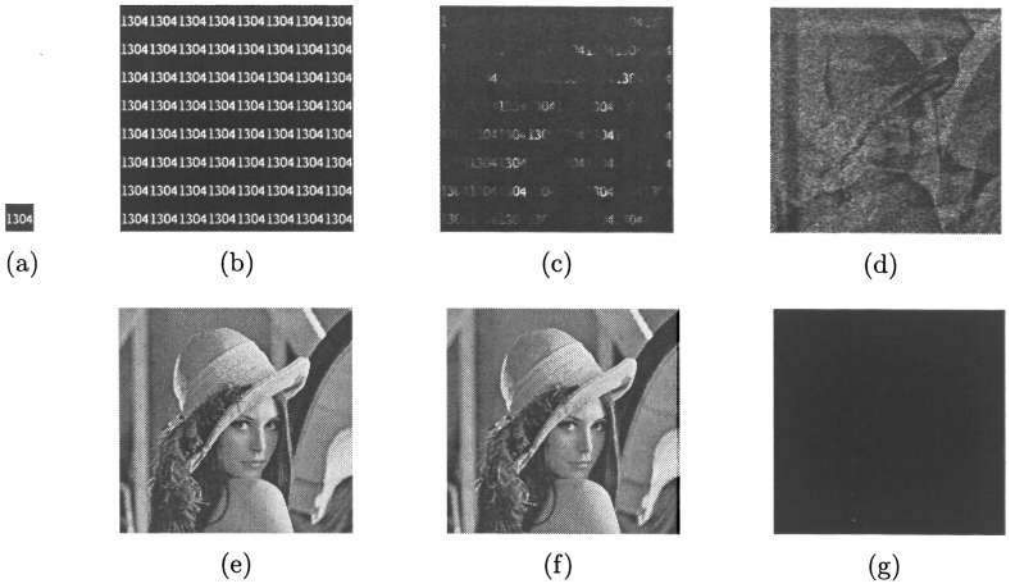


Figure 3.6: An example of the WMicaT embedding scheme. First row: (a) signature S, (b) initial watermark W_0, (c) watermark W, and (d) public image K_P. Second row: (e) original image I, (f) watermarked image I^+, and (g) difference between I and I^+. The size of S is 64×64 and the size of all other images is 512×512.

An example of the embedding scheme is shown in Fig. 3.6. The signature, the initial watermark, the watermark and the public image are shown in Fig. 3.6(a) to 3.6(d), respectively. The owner's signature is a binary image of size 64×64 and the other images 256 gray-scale of size 512×512. In this example, the value of the coefficients are $\alpha = 0.06$, $\beta = 0.015$, $\gamma = -0.7$, $\delta = 0.49$, $\lambda = 0.82$ and $L = 10$. The second row contains the original Lena image, its watermarked image and the difference between these two images. As we can see from Fig. 3.6(f), there is no visible sign of the watermark in the watermarked image . From the public image shown in Fig. 3.6(d), we can observe that the contents of the original image and its transpose are hidden behind a noisy scene.

3.3.3 WMicaT Extraction Scheme

The goal of the extraction scheme is to extract the previously-embedded signature S from the watermark image I^+. Besides the watermarked image, the other information available to us are the public image K_P and the secret key k_s. Using k_s, the seed to the random generator, we are able to generate the key image K. Therefore, the

task now is to extract the signature S with the knowledge of I^+, K_P and K. The extraction scheme can be divided into three stages. The aim of the first stage is to extract the original image I from K and K_P by applying ICA technique. The second stage applies ICA once again on the estimated original image and the watermarked image to extract the watermarks. After that, in stage three, a post processing scheme is applied to obtain the owner's signature from the estimated watermark. A WMicaT extraction scheme is depicted in Fig. 3.7.

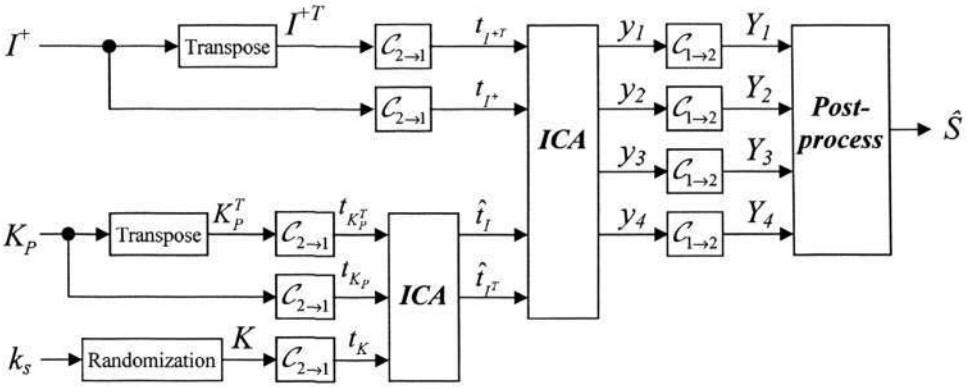


Figure 3.7: The WMicaT extraction scheme. I^+, K_P and k_s represent the watermarked image, public key image and secret key, respectively. $\mathcal{C}_{2\to1}$ and $\mathcal{C}_{1\to2}$ are 2D-to-1D and 1D-to-2D operators, and $\hat{S}$ is the estimate of the owner's signature.

As discussed earlier, the most important task in all ICA-based watermarking methods is to generate enough observations from the available data. Our solution is to use the image transpose. We have two input images: K_P and I^+, and we want to generate at least four signals to extract the watermark. For this purpose, in the first stage, we reconstruct the key image K using a random generator with seed number k_s. The three images K, K_P and K_P^T are converted into one dimensional (1D) signals t_K, t_{K_P}, and $t_{K_P^T}$, respectively, by 2D-to-1D operators $\mathcal{C}_{2\to1}$. Applying (3.3.6) and noting that $K^T = K$, the inputs to the first ICA block can be expressed as

$$\begin{aligned} t_{K_P} &= \mathcal{C}_{2\to1}(K_P) = \mathcal{C}_{2\to1}(\gamma I + \delta I^T + \lambda K) \\ t_{K_P^T} &= \mathcal{C}_{2\to1}(K_P^T) = \mathcal{C}_{2\to1}(\gamma I^T + \delta I + \lambda K) \\ t_K &= \mathcal{C}_{2\to1}(K) \end{aligned} \tag{3.3.11}$$

Denote the 1D signals of I and I^T by t_I and t_{I^T}, respectively. The above equation

(3.3.11) can be rewritten in a matrix form as

$$\begin{bmatrix} t_{K_P} \\ t_{K_P^T} \\ t_K \end{bmatrix} = \begin{bmatrix} \gamma & \delta & \lambda \\ \delta & \gamma & \lambda \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} t_I \\ t_{I^T} \\ t_K \end{bmatrix} \tag{3.3.12}$$

Clearly, (3.3.12) represents an ICA mixing model. That is, by applying ICA technique on the observed signals $[t_{K_P}, t_{K_P^T}, t_K]^T$, we can estimate the 1D signals $\hat{t}_I$ and $\hat{t}_{I^T}$, the original image and its transpose.

The second stage is the main step to extract the watermark. We have in total four 1D observations: t_{I+}, t_{I+^T}, $\hat{t}_I$ and $\hat{t}_{I^T}$. Using (3.3.5) the four mixtures can be expressed as

$$\begin{aligned} t_{I+} &= \mathcal{C}_{2\to 1}(I + \alpha W + \beta W^T) \\ t_{I+^T} &= \mathcal{C}_{2\to 1}(I^T + \alpha W^T + \beta W) \\ \hat{t}_I &= \mathcal{C}_{2\to 1}(I) \\ \hat{t}_{I^T} &= \mathcal{C}_{2\to 1}(I^T) \end{aligned} \tag{3.3.13}$$

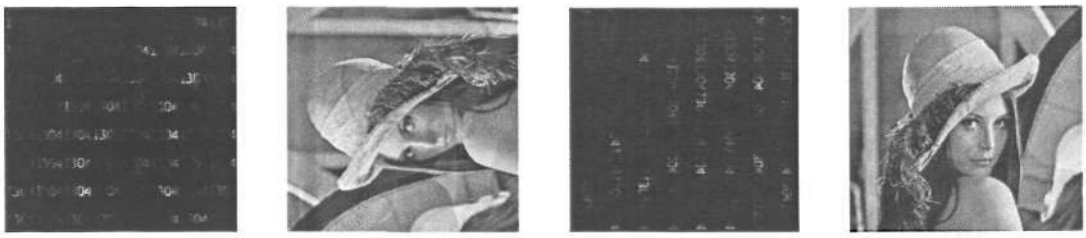
(3.3.14)

or in the matrix format as

$$\begin{bmatrix} t_{I+} \\ t_{I+^T} \\ \hat{t}_I \\ \hat{t}_{I^T} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \alpha & \beta \\ 0 & 1 & \beta & \alpha \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} t_I \\ t_{I^T} \\ t_W \\ t_{W^T} \end{bmatrix} \tag{3.3.15}$$

where t_W and t_{W^T} denote the 1D signals of the watermark W and its transpose W^T, respectively.

Equation (3.3.15) clearly matches the ICA mixing model $\mathbf{x} = \mathbf{As}$. Hence, applying ICA technique on $[t_{I+}, t_{I+^T}, \hat{t}_I, \hat{t}_{I^T}]^T$ results in four outputs y_1, y_2, y_3, and y_4, which correspond to the 1D estimates of the original image I, the watermark W and their transposes I^T and W^T (but may not be in same order). To these 1D signals, we apply 1D-to-2D operators $\mathcal{C}_{1\to 2}$ to generate four estimated images. Figure 3.8 illustrates the ICA output images obtained from the watermarked image and public key image created in the example shown in the previous section. The original Lena image, the watermark and their transposes can be seen clearly in Fig. 3.8.



(a) (b) (c) (d)

Figure 3.8: An example of the WMicaT extraction scheme. The four output images, Y_1, Y_2, $Y3$, and Y_4 are extracted by ICA technique.

3.3.4 The Post-Processing Scheme

The ICA technique, however, only provides a set of images that contains the watermark but is not able to identify it. It means that the output Y_1 does not necessarily correspond to the estimate of original image I. It can be the estimate of any one of the four source signals I, I^T, W and W^T. For this reason, in the third stage of the extraction scheme, we develop a post-processing algorithm to obtain the owner's signature from the images $Y_1, ..., Y_4$. We apply the correlation coefficients in post-processing scheme to identify which output Y_i corresponds to which source signal.

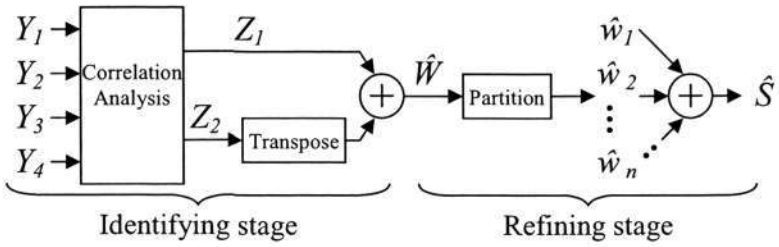


Figure 3.9: The WMicaT post-processing scheme. An identifying stage utilizes the correlation coefficients to obtain the watermark and its transpose. A refining stage produces the owner's signature from these watermark estimates.

The detail scheme of the post-process is shown in Fig. 3.9. It includes two stages, an identifying stage and a refining stage. The first stage filters out the watermarks from the four image inputs. Then the second stage uses the estimated watermarks to extract the owner's signature.

To identify the watermarks, we use the correlation coefficients between each output and the watermarked image. Let us consider two images X and Y, each of size $M \times N$. The absolute correlation coefficient $|r_{X,Y}|$ between two images $X_{M\times N}$ and $Y_{M\times N}$ is

defined as

$$|r_{X,Y}| = \frac{|s_{xy}|}{\sqrt{s_{xx}s_{yy}}} \tag{3.3.16}$$

where

$$\begin{aligned} s_{xy} &= \sum_{i=1}^{M}\sum_{j=1}^{N}(X_{(i,j)} - \bar{X})(Y_{(i,j)} - \bar{Y}) \\ s_{xx} &= \sum_{i=1}^{M}\sum_{j=1}^{N}(X_{(i,j)} - \bar{X})^2 \\ s_{yy} &= \sum_{i=1}^{M}\sum_{j=1}^{N}(Y_{(i,j)} - \bar{Y})^2 \end{aligned} \tag{3.3.17}$$

and

$$\begin{aligned} \bar{X} &= \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}X_{(i,j)} \\ \bar{Y} &= \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}Y_{(i,j)} \end{aligned} \tag{3.3.18}$$

The absolute correlation coefficient, $|r_{X,Y}|$ gives a measure of the similarity between two images, X and Y. When two images are totally different, $|r_{X,Y}| \approx 0$. When X and Y are identical to each other, $|r_{X,Y}| \approx 1$.

Our identification method is based on the following observations. In watermarking, since the watermarked image I^+ is supposed to be highly correlated with the original image I, their absolute correlation coefficient is near to 1, *i.e.*, $|r_{I,I^+}| \approx 1$. Similarly, the absolute correlation coefficient between their transposes, $|r_{I^T,I^{+T}}|$ is also near to 1. On the other hand, the watermark W is considered to be independent from both I^+ and I^{+T}. The values $|r_{W,I^+}|$ and $|r_{W,I^{+T}}|$, therefore, are close to 0. That is, by checking the absolute correlation coefficient between an output Y_i and the watermarked image I^+, we can identify which output is the estimate of the watermark.

Let us denote the absolute correlation coefficient between the watermarked image I^+ and the output Y_i by $|r_{I^+,Y_i}|$. Similarly, denote the absolute correlation coefficient between the transpose of the watermarked image I^{+T} and the output Y_i by $|r_{I^{+T},Y_i}|$, for $i = 1, ..., 4$. Let $\bar{r}_i$ be the sum of these two values, *i.e.*, $\bar{r}_i = |r_{I^+,Y_i}| + |r_{I^{+T},Y_i}|$. From these observations, it is clear that the sum $\bar{r}_i$ will be close to 0 if its corresponding output Y_i is the estimate of the watermark or the watermark transpose. Thus, by computing all the sum $\bar{r}_i$, $i = 1, ..., 4$, we can find out the estimates of the watermark and its transpose.

A numerical example of the correlation coefficients for the results shown in Fig. 3.8 and the sum $\bar{r}_i$ are provided in Table 3.3. From Table 3.3, it can be seen that the ICA outputs, Y_1 and Y_3 are the estimates of W and W^T, respectively.

After successfully estimating the watermark and its transpose by choosing the two

Table 3.3: The correlation coefficient table used for WMicaT post-processing scheme.

	Y_1	Y_2	Y_3	Y_4
$\lvert r_{I^+,Y_i}\rvert$	0.0034	0.0895	0.0374	0.9953
$\lvert r_{I^{+T},Y_i}\rvert$	0.0408	0.9762	0.0059	0.2130
$\bar{r}_i$	**0.0442**	1.0657	**0.0433**	1.2083

outputs that yield the smallest correlation coefficient sum, $\bar{r}_i \approx 0$, we continue to the next stage. The aim of this refining stage is to compute owner's signature from the two extracted watermarks. From the fact that the watermark is a multiple duplication of the signature, we apply a reverse process, splitting the watermark estimate into sub-images and then averaging them to retrieve the owner's signature.

Now let us denote the two outputs that are the estimates of the watermark and its transpose by Z_1 and Z_2. First, we calculate an average watermark, $\hat{W}$ by

$$\hat{W} = (Z_1 + Z_2^T)/2 \tag{3.3.19}$$

Second, we partition the image $\hat{W}$ into l sub-images $\hat{W}_{s1}$, $\hat{W}_{s2}$,...,$\hat{W}_{sl}$ each of size $M_S \times M_S$, where $M_S \times M_S$ is the size of the owner's signature. Third, we compute the estimate of the signature as the average of these sub-images

$$\hat{S} = \frac{1}{l}(\hat{W}_{s1} + \hat{W}_{s2} + \ldots + \hat{W}_{sl}) \tag{3.3.20}$$

An illustration of the refining process is shown in Fig. 3.10. The average watermark is separated into 9 sub-images. These images are then averaged to generate the estimate of the owner's signature. As it can be seen in the figure, the quality of the estimated signature is very good.

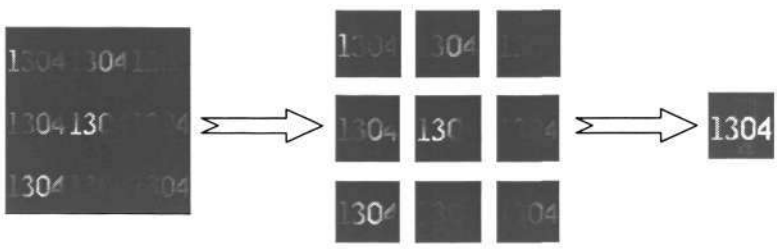


Figure 3.10: An example of the refining stage. The estimated watermark $\hat{W}$ is partitioned into small images. These sub-images are then averaged to generate the estimate of the owner's signature $\hat{S}$.

Here, we summarize the extraction of WMicaT in the following steps:

1. Use the seed k_s to generate the key image K.
2. Apply the 2D-to-1D operator $\mathcal{C}_{2\to1}$ to generate the 1D signals t_{K_P}, $t_{K_P^T}$ and t_K, from K_P, K_P^T and K, respectively.
3. Apply ICA technique on $[t_{K_P}, t_{K_P^T}, t_K]^T$ to extract the 1D signals $\hat{t}_I$ and $\hat{t}_{I^T}$.
4. Utilize a 2D-to-1D operator $\mathcal{C}_{2\to1}$ to generate the 1D signals t_{I^+} and $t_{I^{+T}}$ from the watermarked image I^+ and its transpose I^{+T}, respectively.
5. Apply ICA technique on $[t_{I^+}, t_{I^{+T}}, \hat{t}_I, \hat{t}_{I^T}]^T$ to extract four signals y_1,y_2, y_3, and y_4.
6. Utilize a 1D-to-2D operator $\mathcal{C}_{1\to2}$ to convert y_i to image Y_i, $i = 1, ..., 4$.
7. Apply the post-processing scheme on Y_i to retrieve an estimate of the owner's signature $\hat{S}$.

3.3.5 WMicaT Performance Analysis

We carried out several experiments to verify the robustness of the proposed method under different attacks with different original images and watermarks. We implement Expt1 and Expt2 on a medium-textured Lena image with two different watermarks. The first watermark was an image of an university logo and the second watermark was an image containing three letters 'NTU'. In the Expt3, the same 'NTU' image was selected and embedded in a highly-textured Baboon image.

Experimental settings

The original images (Lena and Baboon) are gray-scale images of size 512×512 with 256 intensity levels. The owner's signatures are binary images; the university logo is of size 128×128 and the university's name 'NTU' is of size 64×64. The embedding strengths α and β were controlled so that the watermarked images have a high quality in term of the Peak Signal-to-Noise Ratio ($PSNR$). The Peak Signal-to-Noise Ratio between an original image I and the modified image $\hat{I}$ is defined as

$$PSNR = 20log_{10}(\frac{255}{RMSE}) \ \ dB \tag{3.3.21}$$

with the RMSE (Root Mean Square Error) is given by

$$RMSE = \sqrt{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(I_{(i,j)} - \hat{I}_{(i,j)})^2} \tag{3.3.22}$$

where $I_{(i,j)}$ and $\hat{I}_{(i,j)}$ denote the (i,j)-th pixel intensity (gray) level of the original and modified images, respectively, and $M \times N$ is the size of the images. The numerical values used for different parameters in the three experiments are provided in Table 3.4. With the chosen parameter values, the watermark was generated and embedded into the host images as described in Section 3.3.2.

Table 3.4: The configuration table for the three experiments. α and β are the embedding strengths. γ, δ and λ are the key-image coefficients. L is the window half-length used in the making of visual mask.

	$I + W$	α	β	γ	δ	λ	L	$PSNR(dB)$
Expt1	Lena+Logo	0.073	-0.010	-0.70	0.49	0.82	10	41.45
Expt2	Lena+NTU	0.060	0.015	-0.70	0.49	0.82	10	43.99
Expt3	Baboon+NTU	0.040	-0.010	-0.70	0.49	0.82	10	42.91

An illustration of the signature S, the original image I, watermark W, the watermarked image I^+, and the public image K_P are shown in Fig. 3.11. With the help of the visual mask and the selected embedding strengths in Table (3.4), high quality watermarked images ($PSNR > 40dB$) were produced. The watermarked images are almost identical to the original ones and the embedded marks are imperceptible to the eyes.

We conduct the simulations by letting the watermarked images I^+ undergo various modifications (attacks) before carrying out watermark extraction. The modified image is called the test image I^*. Major attacks include JPEG compression, gray-scale reduction and image resizing. To carry out the ICA process in the extraction, we use SOBI (Second Order Blind Identification) algorithm, introduced by Belouchrani et al. [28], which provides a fast and effective extraction. For details on SOBI, please see Appendix A. To measure and compare the quality of the estimated signature, we evaluate the absolute correlation coefficient $|r_{S,\hat{S}}|$ between the original owner's signature S and its estimate $\hat{S}$.

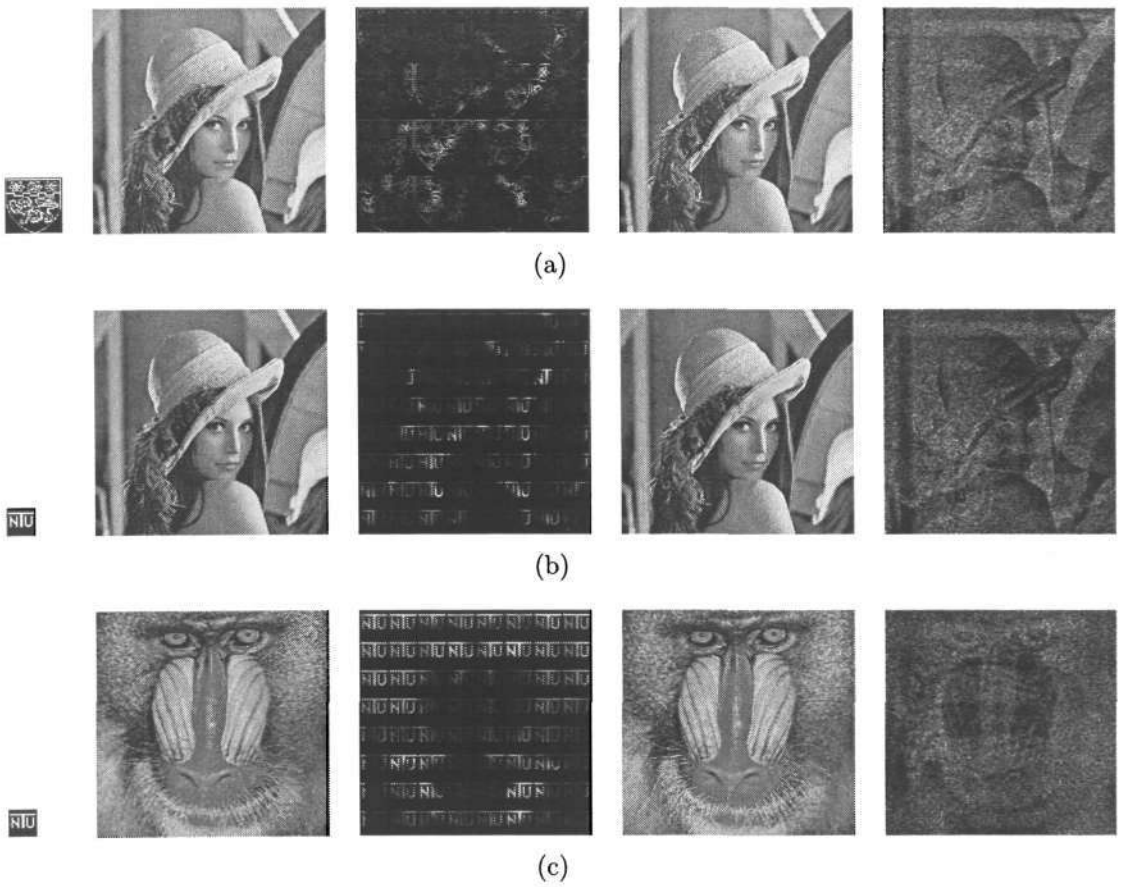


Figure 3.11: The images used in the WMicaT experiments. From left to right: owner's signature S, original image I, watermark W, watermarked image I^+, and public image K_P. (a) Expt1, (b) Expt2, and (c) Expt3.

JPEG Compression Test

In this test, we use a JPEG encoding tool to compress the watermarked image I^+ with different compression quality factors ranging from 90% to 10% to produce the test image I^*. After that, we extract the owner's signature from this JPEG-encoded image I^*. Next, we compute the absolute correlation coefficient $|r_{S,\hat{S}}|$ between the extracted image $\hat{S}$ and the original owner's signature S.

The performance results of WMicaT in three experiments are illustrated in Fig. 3.12. It can be seen that the proposed algorithm provided good performance on all the experiments. The quality of the estimated signatures were high even when the JPEG quality factor was lowered drastically. Only in Expt1, where the watermark was a relatively complex image (university logo), and when the compression quality factor was reduced to the lowest level (= 10%), the estimated signature was unrecognizable. Illustrations of the estimated signatures are shown in Fig. 3.13.

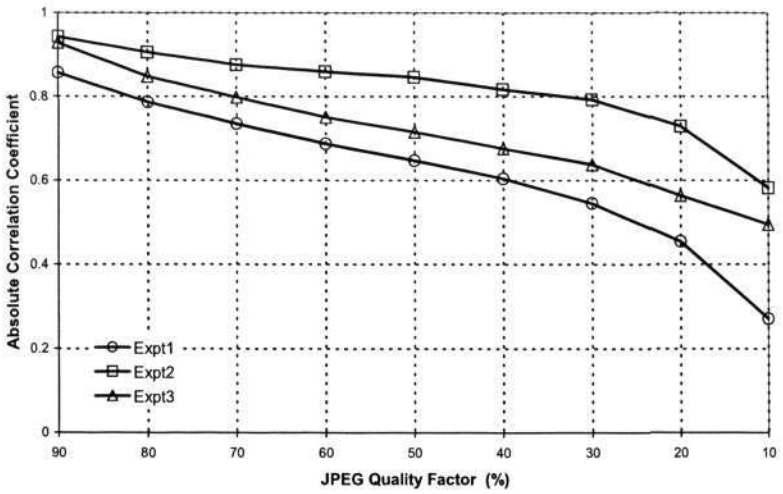


Figure 3.12: WMicaT results for the JPEG compression test. Quality of the estimated signature is measured by the absolute correlation coefficient, $|r_{S,\hat{S}}|$. The compression quality ranges from 90% to 10%.

(a) (b) (c)

Figure 3.13: The estimated signatures of WMicaT in JPEG compression test. (a) Expt1, (b) Expt2 and (c) Expt3. In each figure, from left to right: the outputs of JPEG compression test with quality factor of 90%, 50% and 20%.

Gray-Scale Reduction Test

The gray-scale test is aimed to analyze the extraction ability of WMicaT in the case when the number of different values used to represent the pixel intensity is reduced. In this test, the gray level of the watermarked image I^+ was reduced from 256 down to $128, 64, ..., 4$ level.

As it is shown in Fig. 3.14, the algorithm offered excellent results in the gray-scale reduction test. The performance index, $|r_{S,\hat{S}}|$ in all experiments were high, showing a strong correlation between the estimated image and the owner's signature. It can be seen that WMicaT was able to extract the signature successfully in all the cases where the gray level is greater or equal 8. When the gray level went down to 4, however, the performance of WMicaT degraded and the signature was not recognizable. Figure 3.15 shows examples of the estimated signatures extracted from the test images with gray level of 128, 32 and 8.

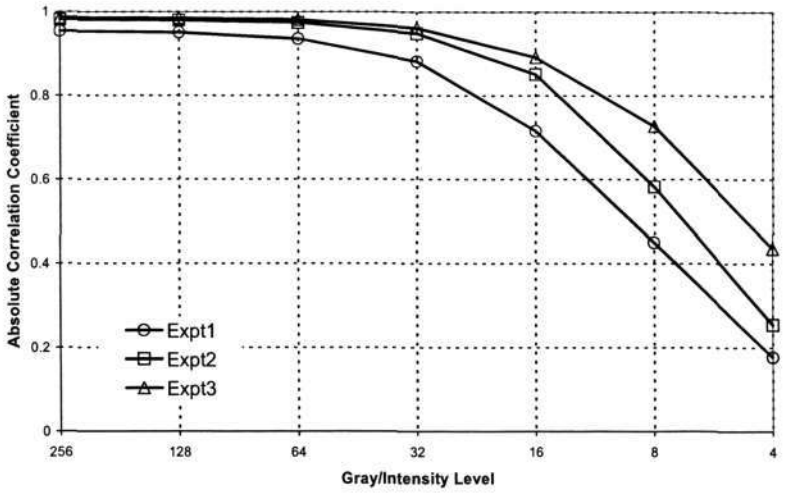


Figure 3.14: WMicaT results for the gray-scale reduction test. Quality of the estimated signature is measured by the absolute correlation coefficient, $|r_{S,\hat{S}}|$. The pixel gray level of the watermarked image is reduced from 256 down to 4 level.

(a) (b) (c)

Figure 3.15: The estimated signatures of WMicaT in gray level reduction test. (a) Expt1, (b) Expt2 and (c) Expt3. In each figure, from left to right: the outputs of the test with gray level reduced to 128, 32 and 8.

When comparing the results among the three experiments, we found out that Expt1's results were inferior to the others because of the relatively complex signature and the watermark. However, unlike the previous JPEG compression test, Expt3 yielded a slightly better performance in comparing with Expt2, especially when the gray level was low. The difference is partially because there is no distortion in the gray level test. The experiment on Baboon image (Expt3), where the watermark was more strongly embedded in textured areas, provided a better result.

Image Resizing Test

Image resizing is one of the most common modification to an image. In this test, we resized the watermarked image to different size and then applied WMicaT to estimate the signature. Since the image is resized, we need to synchronize all the input images (the watermarked image and the public image) to the same size before executing the

extraction scheme. There are two approaches for the synchronization. In the first approach *m1*, we resize all images to the size of the test image I^* before carrying out the ICA-based extraction. After that, the outputs of ICA process, $Y_1, ..., Y_4$ are resized again to the size of the public image and at the end, the post-process scheme is carried out. In the second approach *m2*, the test image I^* is resized to the public image's size and the whole extraction scheme is executed normally.

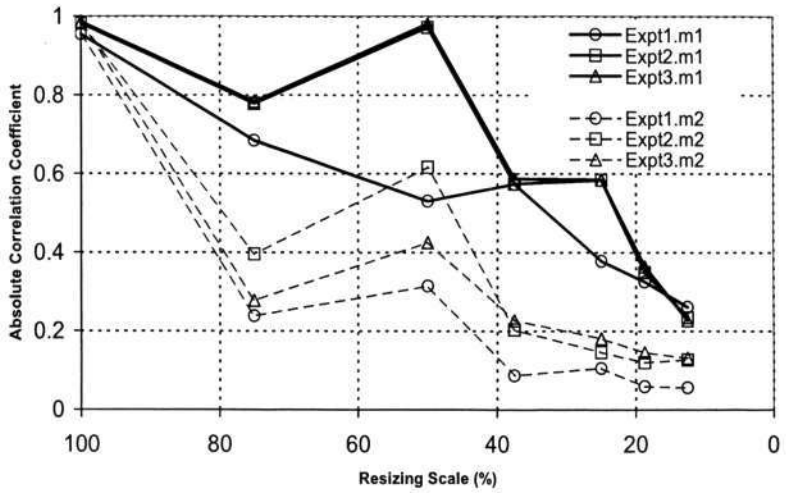
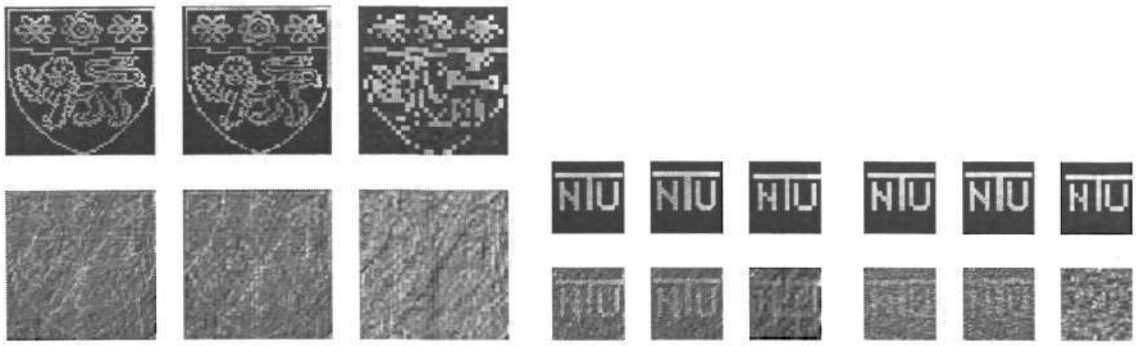


Figure 3.16: WMicaT results for the resizing test. The image is resized from 512 × 512 (100%) down to 64 × 64 (12.5%). In the figure, *m1* and *m2* denote the two synchronization approaches.

We down-scaled the watermarked image I^+ to different sizes: 384 × 384 (75%), 256 × 256 (50%), 192 × 192 (37.5%), 128 × 128 (25%), 96 × 96 (18.75%) and 64 × 64 (12.5%) and then carried out all the three experiments with both synchronization approaches. The experiment results are illustrated in Fig. 3.16. The difference in size synchronization results in different performance of WMicaT. The first approach *m1*, in which the test image I^* is kept intact displays a superior result (Fig. 3.17) in comparison with the second approach *m2*, where the test image is resized before going through the extraction. In fact, in *m2*, the test image has been resized two times, one during the attack and another one during the synchronization. The interpolation technique which is usually involved in the resizing process has modified or even removed most of the watermark content that was embedded in the image. Hence, as it is shown in Fig. 3.17, WMicaT algorithm performed poorly in all three experiments with the second synchronization approach.



(a) (b) (c)

Figure 3.17: The estimated signatures of WMicaT in resizing test. (a) Expt1, (b) Expt2 and (c) Expt3. In each figure, from left to right: the estimated signatures with test image of size 384×384, 256×256 and 128×128. First row: *m1*, second row: *m2*.

Comparison of WMicaT With Other Watermarking Methods

For further investigation, we compared the proposed method with other watermarking techniques that work on different processing domains [144]. These techniques include two Discrete Cosine Transform algorithms *Cox-DCT* [55] and *Koch-DCT* [118], two spatial-domain algorithms *Langelaar-spa* [127] and *Kutter-spa* [124], and two Discrete Wavelet Transform algorithms *Kundur-DWT* [121] and *Wang-DWT* [220]. The result of these techniques are taken from [144] and compared with the result of our second experiment, Expt2 (the Lena embedded with 'NTU' signature).

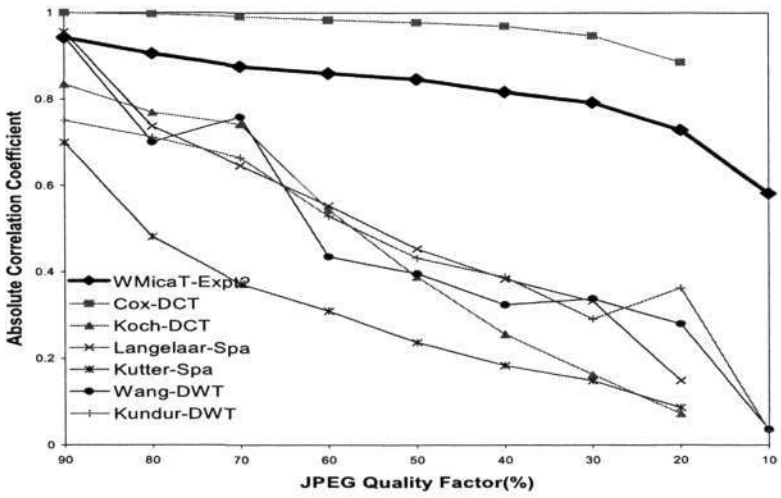


Figure 3.18: Performance comparison between WMicaT and other techniques for JPEG compression test.

Illustrations in Fig. 3.18, Fig. 3.19 and Fig. 3.20 show impressive results of WMicaT in comparing with the other techniques. It outperforms most of the referenced

algorithms in all three image attacking tests: JPEG compression, gray-scale reduction and image resizing. Unlike some methods that are only robust against several specific attacks, the proposed method provides a steady performance throughout all the tests. In JPEG compression test, WMicaT outperforms the other algorithms and only inferior to on DCT-based method (the *Cox-DCT*). It is an advantage of WMicaT since the spatial-based techniques like WMica usually perform poorly on JPEG test [54].

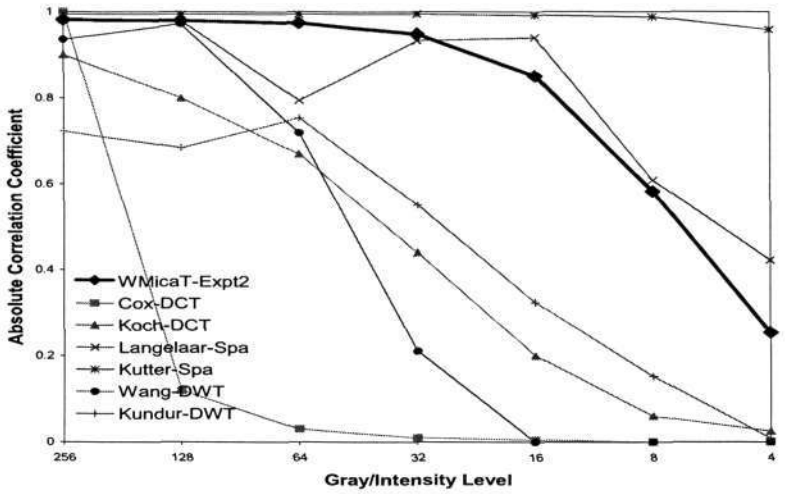


Figure 3.19: Performance comparison between WMicaT and other techniques for gray-scale reduction test.

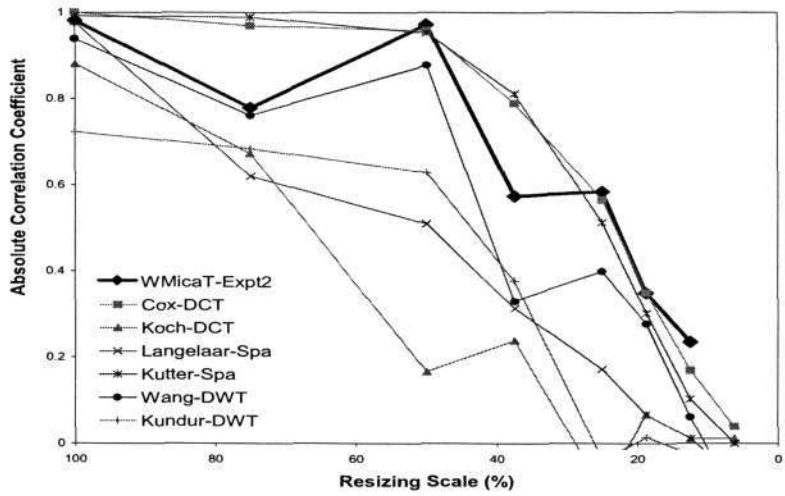


Figure 3.20: Comparison of WMicaT performance on resizing test with other watermarking algorithms.

Again, in the gray-scale reduction test (Fig. 3.19), the WMicaT provides good performance and is one of the three methods that yields the best result while the *Cox-DCT* method could not provide adequate result. A similar situation can be

observed in the third test in Fig. 3.20 on image resizing, WMicaT is again one of the top three methods that provide the best estimation of the owner's signature.

3.3.6 Discussion on WMicaT Algorithm

The proposed WMicaT method has shown impressive results against several common attacks such as JPEG compression, gray-scale reduction and image resizing. It has provided a consistent performance on different host images through out all the experiments. In comparison with other watermarking methods, WMicaT illustrates very good performance with high quality estimation of the owner's signature.

The effect of watermark content to the algorithm can be observed from the experimental results. An image embedded by a complex watermark that contains curves and discontinued areas seems to be more vulnerable to the attack than that embedded by a watermark with straight lines or smooth areas. Therefore, in the same test, the quality of the estimates of complex watermarks (the university logo, for example) is not as high as those of a simple watermark. However, the performance of WMicaT in the experiments with complex watermark is still very good. The estimated images are highly correlated with the original owner's signature. The algorithm fails to recognize the signature only when the test image is modified severely.

The WMicaT algorithm is a fast and robust watermarking technique. However, it has a disadvantage of using a large size supporting image, *i.e.*, the public image. The size of the public image is as big as the size of the original image. Therefore, it is not very convenient in storing and transferring this additional image. In our second watermarking scheme (WMicaD), we address to remove these problems.

3.4 WMicaD: ICA-based Watermarking Using Dual Watermarks

In this Section, we introduce another ICA-based watermarking method called WMicaD. This method aims to achieve two goals: (i) to reduce the amount of additional data needed, and (ii) to separate the watermark into a fixed part and variable part. The reason of separating a watermark into two parts is explained as follows.

Let us begin with an example. Assuming that an owner want to distribute his Work online securely. He needs to make digital copies of his Work, embeds them with his own information and then upload them to the Internet. His information, therefore, usually contains two parts: one for ownership verification and one for tracking the copy identity. The first part should be identical in all copies while the second part should be unique for each copy, for example, a copy ID number. Because of this characteristic, we name the first part as the 'fixed watermark' and then second one as the 'variable watermark'. Our WMicaD is developed to fulfill the above requirements by exploiting ICA separation technique and applying a special watermark construction scheme. Short reports of this method have been presented in [151, 152].

3.4.1 Special Property of The Dual Watermarks

From Section 3.2, an ICA-based watermarking algorithm needs additional information to generate the inputs for the ICA separating component. Our previous WMicaT method utilizes a public image as the supporting data for watermark extraction, but the size of the public image is as large as the original Work. In the WMicaD method, we attempt to reduce the size of the supporting image and to carry out all the extracting operations at the size-reduced level. First, we review the two operators that are used to resize the images. Second, we introduce the modification scheme which is applied to the watermarks so that the watermarks can reveal different contents at different size.

Down-sizing and Up-sizing Operators

Let $I_{M\times N}$ be an image of size $M \times N$ whose (m,n)-th entry is represented as $I_{(m,n)}$, $m = 0, 1, ..., M-1$, $n = 0, 1, ..., N-1$. Denote the down-sizing operator by $\mathcal{D}$, then the $k-time$ down-sized version of $I_{M\times N}$ is defined as $I_{[k]\frac{M}{k}\times\frac{N}{k}} = \mathcal{D}(I_{M\times N}, k)$ whose (m,n)-th entry is computed by

$$I_{[k](m,n)} = \frac{1}{k^2}\sum_{i=0}^{k-1}\sum_{j=0}^{k-1} I_{(km+i,kn+j)} \tag{3.4.1}$$

for all $m = 0, 1, ..., (\frac{M}{k}-1)$ and $n = 0, 1, ..., (\frac{N}{k}-1)$, and k is a non-zero positive integer, called 'resizing factor'. In sort, $I_{[k](m,n)}$ is the average of the gray values inside a window of size $k \times k$ in the original matrix $I_{M\times N}$.

The up-sizing operator $\mathcal{U}$, in contrast, duplicates an element of $I_{M\times N}$ to every element in a window of size $k \times k$. The $k-time$ up-sized version of $I_{M\times N}$ is defined as $I^{[k]}_{kM\times kN} = \mathcal{U}(I_{M\times N}, k)$ whose (m,n)-th entry is computed by

$$I^{[k]}_{(m,n)} = I_{(\lfloor \frac{m}{k} \rfloor, \lfloor \frac{n}{k} \rfloor)} \tag{3.4.2}$$

for all $m = 0, 1, ..., (kM-1)$ and $n = 0, 1, ..., (kN-1)$, and k is the 'resizing factor'. The 'floor' operator $\lfloor x \rfloor$ truncates the number x to a nearest smaller integer.

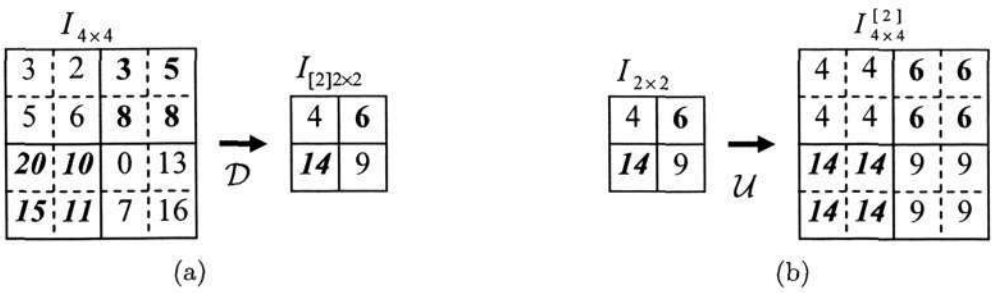


Figure 3.21: An example of the down-sizing and up-sizing operators with resizing factor $k = 2$. (a) $I_{[2]2\times2}$ is down-sized from $I_{4\times4}$ and (b) $I^{[2]}_{4\times4}$ is up-sized from $I_{2\times2}$.

An illustration of the up- and down-sizing operators are shown in Fig 3.21. From an image of size 4×4, $I_{4\times4}$ the down-sizing operator generates an image $I_{[2]2\times2}$ (Fig. 3.21(a)) with resizing factor, $k = 2$. Fig. 3.21(b) illustrates an image $I^2_{4\times4}$ obtained by up-sizing the image $I_{2\times2}$ by a resizing factor of 2.

Watermark Modification

In order to extract the two watermarks by ICA technique, we need to have at least three images as the mixtures. However, we only have two available images: a watermarked image and a key image (created in the embedding process). Simple linear combination of these two images can not create three different mixtures. Hence, our solution is to modify the watermarks with certain conditions so that they reveal different information at different image scales.

The first watermark W_1 is modified in such a way that when it is down-sized by a factor k_1, it produces a small-sized watermark $W_{1[k_1]}$. But, when W_1 is down-sized by a factor k_1k_2, it produces a null-matrix. Mathematically, this property can be

expressed as

$$\emptyset_{[k_1 k_2]} = \mathcal{D}(W_1, k_1 k_2) \tag{3.4.3}$$

$$W_{1[k_1]} = \mathcal{D}(W_1, k_1) \tag{3.4.4}$$

where $\emptyset$ denotes a null matrix, $\mathcal{D}$ is the down-sizing operator and k_1 and k_2 are the two resizing factors.

The second watermark W_2 is modified so that when we down-size and subsequently up-size it with the same factor, the watermark remains unchanged. Mathematically, this property can be expressed as

$$\mathcal{D}(W_2, k_1) = \mathcal{U}(\mathcal{D}(W_2, k_1 k_2), k_2) \tag{3.4.5}$$

The use of watermarks W_1 and W_2 will be explained in detail in the extraction scheme. In the following paragraphs, we introduce a simple technique to create these two watermarks.

First Watermark Modification: $\mathcal{M}_1$

The goal of the first watermark modification function $\mathcal{M}_1$ is to generate a watermark W_1 from the owner's signature so that the watermark satisfies the equations (3.4.3) and (3.4.4). An illustration of the modification scheme is shown in Fig. 3.22 in which a watermark of size 8×8 is created from a signature of size 2×2. Details of the scheme are provided in the following paragraphs.

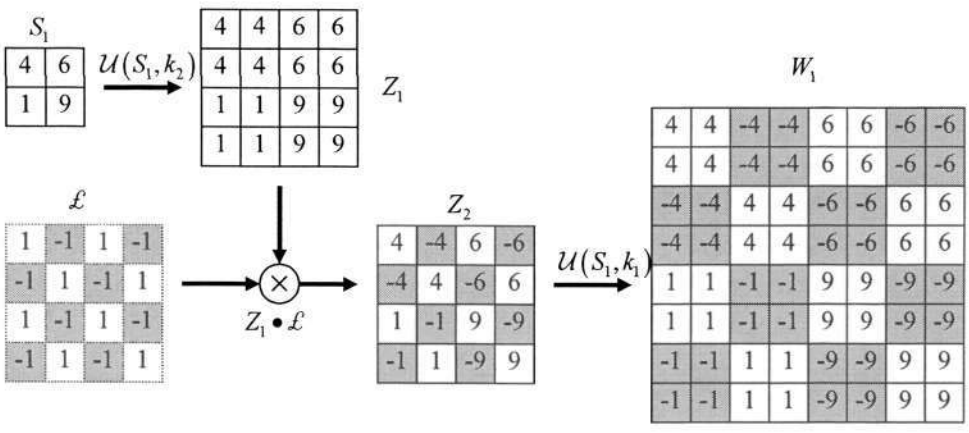


Figure 3.22: An example of the first watermark modification scheme $\mathcal{M}_1$. A watermark W_1 of size 8×8 is generated from an author signature S_1 of size 2×2. Resizing factors $k_1 = k_2 = 2$.

Let S_1 be an image of size $\frac{M}{k_1k_2} \times \frac{N}{k_1k_2}$ that represents the owner's signature. The scheme to construct the watermark W_1 from S_1 is described mathematically as

$$Z_1 = \mathcal{U}(S_1, k_2) \tag{3.4.6}$$

$$Z_2 = Z_1 \bullet \pounds \tag{3.4.7}$$

$$W_1 = \mathcal{U}(Z_2, k_1) \tag{3.4.8}$$

At first, the signature represented by a matrix S_1 is up-sized by a factor k_2 ($k_2 = 2$ in this example) to create a matrix Z_1. Second, the newly created matrix Z_1 is multiplied by a 'chessboard' matrix $\pounds$ element-by-element. Finally, the product of this multiplication (Z_2) is up-sized by a factor k_1 ($k_1 = 2$ in the illustration) to produce the watermark W_1. It can be seen that when the generated watermark W_1 is downsized by k_1k_2, it will result a null matrix satisfying (3.4.3).

In this scheme, the chessboard matrix, $\pounds$ is a matrix whose (m, n)-th entry is defined by

$$\pounds_{(m,n)} = \begin{cases} 1 & if \ \ (m+n) = even \\ -1 & otherwise \end{cases} \tag{3.4.9}$$

and the (m, n)-th entry of the element-by-element product $\bullet$ is computed by

$$Z_{2(m,n)} = Z_{1(m,n)} \pounds_{(m,n)} \tag{3.4.10}$$

Thus, in the proposed scheme, $\mathcal{M}_1$ will generate the watermark W_1 that satisfies both (3.4.3) and (3.4.4).

Second Watermark Modification: $\mathcal{M}_2$

The second modification function $\mathcal{M}_2$ is to create a watermark W_2 that satisfies (3.4.5), that is

$$\mathcal{D}(W_2, k_1) = \mathcal{U}(\mathcal{D}(W_2, k_1k_2), k_2) \tag{3.4.11}$$

The scheme of generating W_2 is simpler than that of W_1. Beginning with a signature S_2 of size $\frac{M}{k_1k_2} \times \frac{N}{k_1k_2}$, we apply the up-sizing operator $\mathcal{U}$ on S_2 with resizing factors k_1k_2 to obtain

$$W_2 = \mathcal{U}(S_2, k_1k_2) \tag{3.4.12}$$

Figure 3.23 shows an illustration of the second modification scheme, $\mathcal{M}_2$. The second watermark W_2 of size 8×8 is constructed from a signature S_2 of size 2×2 by

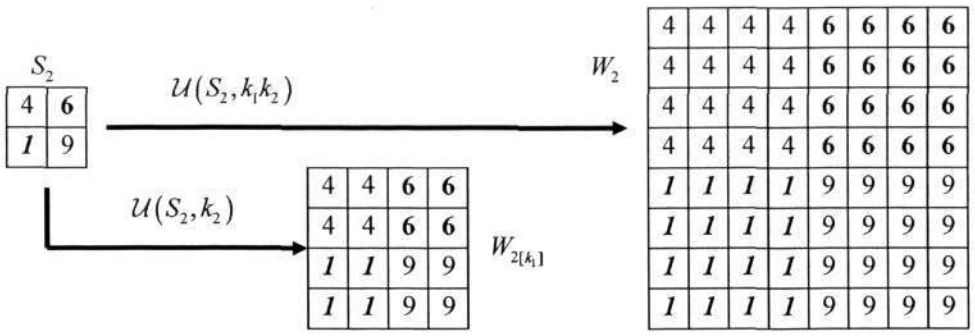


Figure 3.23: An example of the second watermark modification scheme $\mathcal{M}_2$. A watermark W_2 of size 8×8 is generated from an author signature S_2 of size 2×2. Resizing factors $k_1 = k_2 = 2$.

an up-sizing operator $\mathcal{U}$ with the resizing factors $k_1 = 2$ and $k_2 = 2$. It is easy to see that the generated watermark W_2 satisfies (3.4.5), *i.e.*, $W_{2[k_1]} = \mathcal{U}(S_2, k_2)$.

3.4.2 WMicaD Embedding Scheme

The WMicaD embedding scheme generates two images: a watermarked image I^+ and a key image K. I^+ is constructed by inserting the two watermarks W_1 and W_2 into the original image I. The small-sized key image K is generated as a supporting image that is used during the watermark extraction. Details of the scheme are shown in Fig. 3.24.

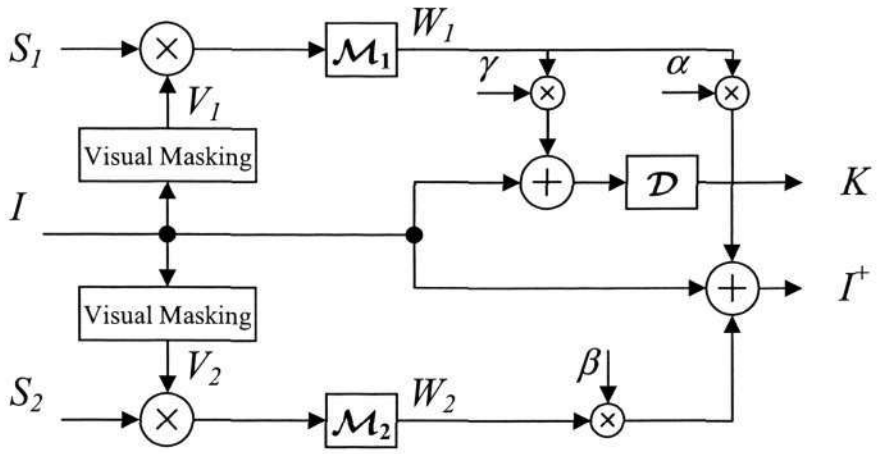


Figure 3.24: The WMicaD embedding scheme. Special watermarks W_1 and W_2 are created from the owner's signatures by the modification functions and are embedded in the original image to create the watermarked image, I^+. The key image is generated by down-sizing the combination of W_1 and I.

At first, we create two visual masks V_1 and V_2 from the original image. The visual

masks can be different from each other by choosing different masking windows half-lengths, L_1 and L_2. As discussed in Section 3.3.2, the visual masks help to increase the embedding strength of the watermarks. Next, we generate the first watermark W_1 by applying the visual mask V_1 on the owner signature S_1 and then passing it through the modification function $\mathcal{M}_1$. The generated watermark W_1, therefore, satisfies the conditions (3.4.3) and (3.4.4). Similarly, we apply the visual mask V_2 on the copy ID S_2, and then apply function $\mathcal{M}_2$ to generate the second watermark W_2 that satisfies (3.4.5).

In the next step, the two watermarks W_1 and W_2 are inserted into the original image I to produce the watermarked image I^+. Mean while, the key image K is generated by down-sizing a combination of I and the first watermark W_1. Note that by the down-sizing operation, the size of the key image can be substantially reduced. In summary, the steps involved in the embedding scheme are given below.

1. Create two visual masks V_1 and V_2 by NVF method introduced in Section 3.3.2

$$V_i = \mathcal{V}(I, L_i) \tag{3.4.13}$$

   where L_i is the window length, $i = 1, 2$.

2. Use $\mathcal{M}_1$ to create W_1 that satisfies (3.4.3) and (3.4.4)

$$W_1 = \mathcal{M}_1(S_1, V_1, k_1, k_2) \tag{3.4.14}$$

3. Use $\mathcal{M}_2$ to create W_2 that satisfies (3.4.5)

$$W_2 = \mathcal{M}_2(S_2, V_2, k_1, k_2) \tag{3.4.15}$$

4. Create the watermarked image I^+

$$I^+ = I + \alpha W_1 + \beta W_2 \tag{3.4.16}$$

5. Compute the key image K

$$K = \mathcal{D}(I + \gamma W_1, k_1) \tag{3.4.17}$$

Parameters α and β are called 'embedding strengths' and γ is called 'key-image coefficient'. These parameters can be any non-zero values in the range of $[-1, 1]$.

3.4.3 WMicaD Extraction Scheme

The goal of the extraction process is to estimate two embedded signatures from the watermarked image I^+. In order to apply ICA in the extraction, we need to generate three inputs from two available sources: the watermarked image I^+ and the key image K. The whole process is carried out on the down-sized images, hence, it significantly reduces the computational workload.

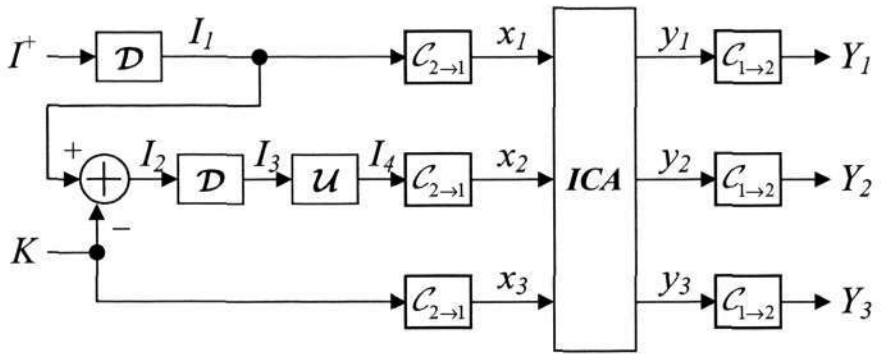


Figure 3.25: The WMicaD extraction scheme. I^+ and K are the watermarked image and the key image, respectively. The up-sizing and down-sizing operators are denoted by $\mathcal{U}$ and $\mathcal{D}$. $\mathcal{C}_{2\to1}$ and $\mathcal{C}_{1\to2}$ are the 2D-to-1D and 1D-to-2D conversion operators, respectively.

The details of the WMicaD extraction scheme are shown in Fig. 3.25. The extraction is initiated by down-sizing I^+ to the size of key image K. The key image is subtracted from the down-sized version of I^+, denoted by I_1, to generate another image I_2. Next, we apply the down-sizing operator followed by an up-sizing operator, both with the same resizing factor, on I_2. By doing that, we are able to exploit the special properties of the two watermarks to generate the necessary third input I_4 needed for the ICA block.

In the next step, the three inputs I_1, K and I_4 are converted to one dimensional signal by a 2D-to-1D operator and passed through the ICA block. Finally, a 1D-to-2D operator is applied to convert the ICA outputs back to 2D images. The steps involved in the WMicaD extraction are described as follow

1. Down-size the watermarked image I^+ to the size of the key image K with resizing factor k_1

$$I_1 = \mathcal{D}(I^+, k_1) \tag{3.4.18}$$

2. Create the image I_4 from I_1 and K by applying up-sizing and down-sizing operators with a resizing factor k_2

$$I_4 = \mathcal{U}(\mathcal{D}(I_1 - K, k_2), k_2) \tag{3.4.19}$$

3. Create 1D signals from I_1, I_4 and K

$$[x_1, x_2, x_3]^T = [\mathcal{C}_{2\to1}(I_1), \mathcal{C}_{2\to1}(I_4), \mathcal{C}_{2\to1}(K)]^T \tag{3.4.20}$$

where $\mathcal{C}_{2\to1}$ denotes a 2D-to-1D operator.

4. Apply an ICA technique on $\mathbf{x} = [x_1, x_2, x_3]^T$ to get three outputs $\mathbf{y} = [y_1, y_2, y_3]^T$.

5. Convert back the outputs $\mathbf{y}$ to images

$$Y_i = \mathcal{C}_{1\to2}(y_i) \tag{3.4.21}$$

where $i = 1, 2, 3$, and $\mathcal{C}_{1\to2}$ is a 1D-to-2D operator.

To explain the meaning of each step in the extraction process and to see how ICA technique can work on these mixed signals, we begin with I_1 (3.4.18). With respect to (3.4.16), I_1 can be expressed as

$$\begin{aligned} I_1 &= \mathcal{D}(I, k_1) + \alpha\mathcal{D}(W_1, k_1) + \beta\mathcal{D}(W_2, k_1) \\ &= I_{[k_1]} + \alpha W_{1[k_1]} + \beta W_{2[k_1]} \end{aligned} \tag{3.4.22}$$

Referring to (3.4.17), the key image K can be rewritten as

$$K = I_{[k_1]} + \gamma W_{1[k_1]} \tag{3.4.23}$$

Since $I_2 = I_1 - K$, using (3.4.22) and (3.4.23) we get

$$I_2 = (\alpha - \gamma)W_{1[k_1]} + \beta W_{2[k_1]} \tag{3.4.24}$$

Let $I_3 = \mathcal{D}(I_2, k_2)$. Since W_1 satisfies (3.4.3) and (3.4.4), and W_2 satisfies (3.4.5), I_3 can be refined as

$$\begin{aligned} I_3 &= (\alpha - \gamma)\mathcal{D}(W_{1[k_1]}, k_2) + \beta\mathcal{D}(W_{2[k_1]}, k_2) \\ &= (\alpha - \gamma)\emptyset + \beta W_{2[k_1k_2]} \\ &= \beta W_{2[k_1k_2]} \end{aligned} \tag{3.4.25}$$

Finally, substituting (3.4.24) and (3.4.25) in (3.4.19) yields

$$\begin{aligned} I_4 &= \mathcal{U}(I_3, k_2) \\ &= \beta W_{2[k_1]} \end{aligned} \tag{3.4.26}$$

By using equations (3.4.22), (3.4.23) and (3.4.26), the input vector $\mathbf{x} = [x_1, x_2, x_3]^T$ in (3.4.20) can be represented in the matrix form as

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & \alpha & \beta \\ 0 & 0 & \beta \\ 1 & \gamma & 0 \end{bmatrix} \begin{bmatrix} t_I \\ t_{W1} \\ t_{W2} \end{bmatrix} \tag{3.4.27}$$

where t_I, t_{W1} and t_{W2} are the three 1D signals, converted from $I_{[k_1]}$, $W_{1[k_1]}$ and $W_{2[k_1]}$ by the 2D-to-1D converters, respectively. As it is shown, (3.4.27) matches the ICA mixing model $\mathbf{x} = \mathbf{As}$. It means that by applying ICA technique on $\mathbf{x} = [x_1, x_2, x_3]^T$, we can compute the estimates of $I_{[k_1]}$, $W_{1[k_1]}$ and $W_{2[k_1]}$. Most importantly, since ICA algorithm is applied on the down-sized version of I^+, it provides substantial computational advantage over those methods that are executed on full-size images.

3.4.4 The Post-Processing Scheme

Because of utilization of ICA in extraction, WMicaD algorithm faces the same problem that WMicaT has; the problem of identifying the watermarks. We know that the outputs of ICA components, Y_1, Y_2 and Y_3 are the estimates of $I_{[k_1]}$, $W_{1[k_1]}$ and $W_{2[k_1]}$ but they are in arbitrary order. Therefore, we need a post-processing scheme to identify which output is the corresponding estimate of the watermarks.

Again, we apply the absolute correlation coefficient, $|r_{X,Y}|$ defined in (3.3.16) to identify the correct estimates. Firstly, the absolute correlation coefficients $|r_{I^+_{[k_1]},Y_i}|$, $i = 1, 2, 3$, between the watermarked image $I^+_{[k_1]}$ and each of the outputs Y_i are computed. On one hand, the output Y_k that corresponds to the original image $I_{[k_1]}$ is highly correlated with the test image $I^+_{[k_1]}$, *i.e.*, $|r_{I^+_{[k_1]},Y_k}| \approx 1$. On the other hand, as the watermarks are assumed to be independent from the original image, the correlation of the two remaining outputs (corresponding to the watermarks) with the test image will be very low, *i.e.* $|r_{I^+_{[k_1]},Y_j}| \approx 0$, $j \neq k$. Hence, taking the two outputs that have lowest $|r_{I^+_{[k_1]},Y_i}|$ gives us the estimates of the down-sized watermarks $\hat{W}_{[k_1],1}$ and $\hat{W}_{[k_1],2}$.

In the next step, we obtain the original signatures from the watermark estimates. Since the watermarks are created by replicating the owner's signature S_1 and the copy ID number S_2, we partition the image Y_i into l sub-images $Y_{i1}, ..., Y_{il}$ each of size $M_S \times M_S$, where $M_S \times M_S$ is the size of the owner's signature. Averaging these sub-images yields the estimate of the signature.

$$\hat{S} = \frac{1}{l}(\hat{W}_{i1} + \hat{W}_{i2} + \ldots + \hat{W}_{il}) \tag{3.4.28}$$

3.4.5 WMicaD Performance Analysis

Simulations was carried out to test the robustness and the tamper detection ability of the proposed algorithm. The experiments were done on two well-known images, the Lena and Baboon, with major attacks: JPEG compression, gray-scale reduction, noise addition, resizing and rotation. In addition, we also carried out an experiment to verify the tamper detection ability of WMicaD.

Experimental settings

The experiments were implemented on the gray-scale images Lena and Baboon of size 512×512. Two small-sized binary images of size 16×64 shown in Fig. 3.26 were used as the author's signature and the copy ID. Two watermarks of size 512×512 were created from the author's signature and copy ID images. In the simulations, Peak Signal to Noise Ratio ($PSNR$) was chosen as the criterion for measuring and controlling the quality of the watermarked image. The $PSNR$ between an image I and its modification $\hat{I}$ has been defined in (3.3.21).

Figure 3.26: The two original signatures, S_1 and S_2, used in the simulations. Both images are of size 16×64.

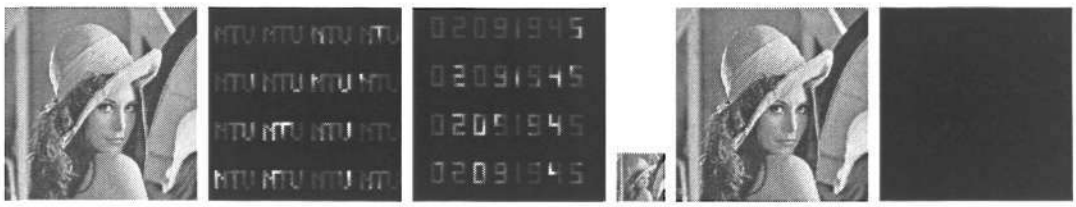
In order to maintain the quality of the watermarked image and the imperceptibility of the watermarks, the embedding coefficients α, β and the window half-length L (used in the visual mask function $\mathcal{V}$) were chosen so that $PSNR \geq 43dB$ in all the experiments. The resizing factors k_1 and k_2 were appropriately selected so that the

key image K is small enough while the watermarks still have adequate details. Details of the selected parameters are provided in Table 3.5.

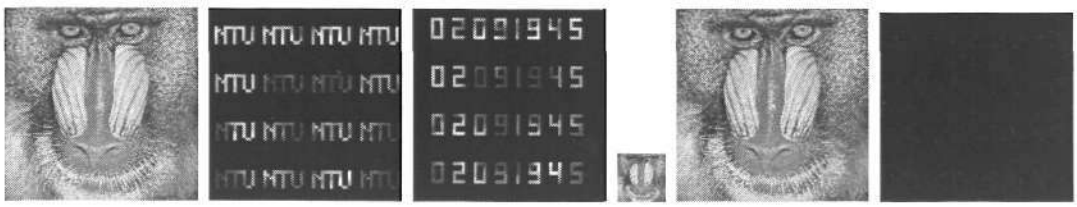
Table 3.5: The configuration table for the two experiments. α, β are the embedding strengths and γ is the key-image coefficient. L_1 and L_2 are the window half-lengths used in the making of two visual masks. k_1 and k_2 are the resizing factors.

	α	β	γ	k_1	k_2	$L_1 = L_2$	$PSNR$
Lena	$\frac{-7}{256}$	$\frac{11}{256}$	$\frac{9}{256}$	4	2	12	44.99
Baboon	$\frac{-6}{256}$	$\frac{9}{256}$	$\frac{9}{256}$	4	2	10	43.14

The original image I, watermarks W_1 and W_2, key image K and watermarked image I^+ are shown in Fig. 3.27. As the down-sizing factors was chosen at $k_1 = 4$ and $k_2 = 2$, the key image is of size 128×128. From the difference image between I^+ and I (as shown in the last column of Fig. 3.27), one can observe that very little distortion has been done to the host image due to the watermarking.



(a)



(b)

Figure 3.27: The images used in the WMicaD experiments. From left to right: original image I, watermarks W_1 and W_2, key image K, watermarked image I^+, and the difference between I^+ and I. The size of the key image K is 128×128 and the size of the other images is 512×512. (a) Expt1: Lena image. (b) Expt2: Baboon image.

The simulations were run through four stages. First, the embedding scheme was carried out to generate watermarked image and key image. Second, the watermarked image was attacked by the above-mentioned modifications (JPEG compression, gray-scale reduction, *etc.*). Third, the extraction and post-processing scheme were carried

out on the attacked image, *i.e.*, the test image. Finally, the estimated signatures were compared with the original ones to evaluate the their quality and the performance of our algorithm.

We used SOBI (Second Order Blind Identification) algorithm [28] (see Appendix A for details) to carry out the ICA process. To measure and compare the quality of the estimated signature, we computed the absolute correlation coefficient (3.3.16), $|r_{S,\hat{S}}|$ between the original owner's signature S and its estimate $\hat{S}$.

JPEG Compression Test

The watermarked image I^+ was compressed using JPEG compression tool with different quality factors (from 90% down to 10%). The watermark extraction was done on the compressed image. The performance index, in term of correlation coefficient, $|r_{S,\hat{S}}|$, was computed for each quality factor and is shown in Fig. 3.28.

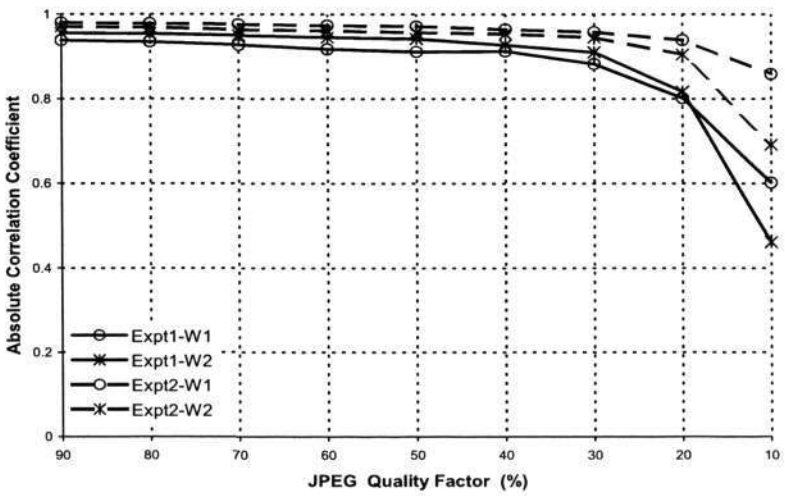
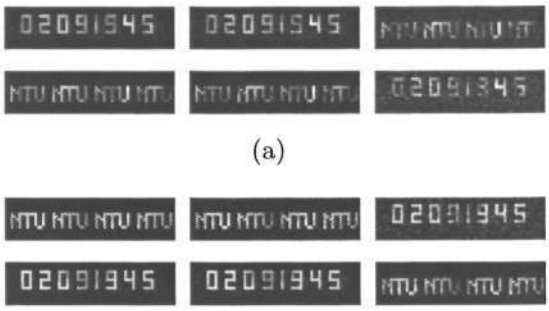


Figure 3.28: WMicaD results for the JPEG compression test. The compression quality ranges from 90% to 10%. Expt1: Lena image. Expt2: Baboon image.

In the plot, solid lines represent the results of the experiment with Lena image (Expt1). Dashed lines represent the results of the experiment with Baboon image (Expt2). The circle symbol ($\circ$) represents the results of the first watermark W_1, and the star symbol ($*$) represents the results of the second watermark W_2. As it is shown, the proposed algorithm provided good performance on the two experiments. The qualities of the estimates are high even when the JPEG compression quality factor is reduced awfully.



(a)

(b)

Figure 3.29: The estimated signatures of WMicaD in JPEG compression test. From left to right: the outputs of JPEG compression test with quality factor of 90%, 50% and 20%. (a) Expt1 and (b) Expt2.

The estimated signatures when the test image was compressed with the quality factor of 90%, 50% and 20% are shown in Fig. 3.29. As we can see, the estimates are clearly visible in the figure. Even in the lowest quality settings, all the signatures are still recognizable.

Gray Scale Reduction Test

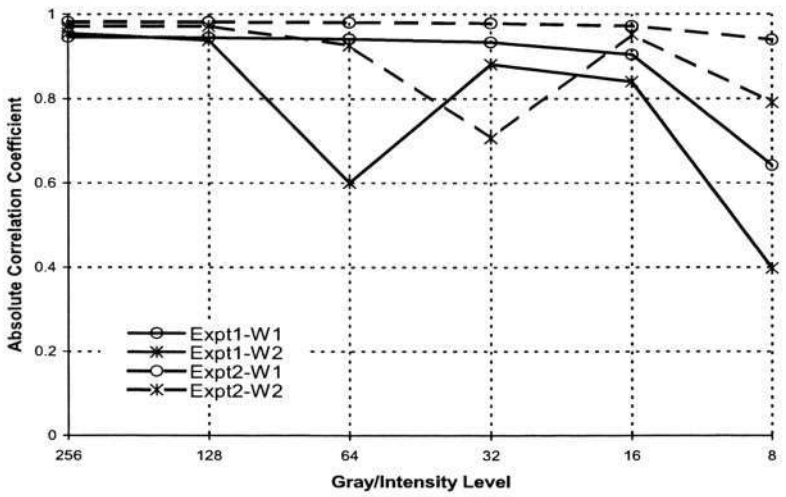


Figure 3.30: WMicaD results for the gray level reduction test. The gray level of the watermarked image is reduced from 256 down to 8 level. Expt1: Lena image. Expt2: Baboon image.

In this test, the intensity (gray) level of the watermarked image I^+ was reduced from original 256 level down to $128, 64, ..., 8$ level. As it is shown in Fig. 3.30, the proposed algorithm offered good results in the test with high correlation between the estimated signatures and the original ones. The proposed algorithm is able to

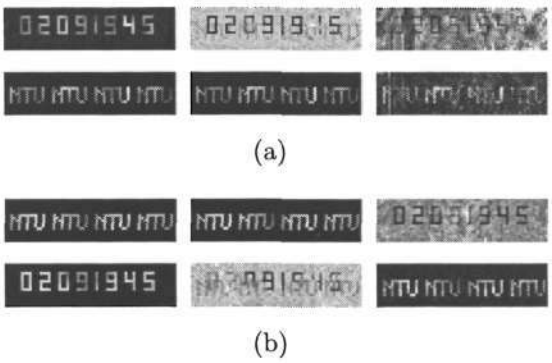


Figure 3.31: The estimated signatures of WMicaD in gray level reduction test. From left to right: the outputs of the test with gray level reduced to 128, 32 and 8. (a) Expt1 and (b) Expt2.

extract the signature successfully when the gray scale level is reduced up to a level of 8. Figure 3.31 shows the clear estimates when the test image gray level was reduced from original 256 level to 128, 32 and 8.

Addition-of-Noise Test

We introduce the results of WMicaD in the noise test with three different types of noise. It includes a Gaussian white noise with zero mean and variance σ^2 ranging from 0 to 0.05, a 'salt & pepper' noise with noise density ranging from 0 to 0.05, and a multiplicative noise. In multiplicative noise test, a uniformly distributed random noise u with zero mean and variance σ^2 ranging 0 to 0.05 was multiplied and added to the image by the equation

$$I^* = I^+ + uI^+ \tag{3.4.29}$$

The performance results of WMicaD on the three noise tests: Gaussian, 'salt & pepper' and multiplicative noise are shown in Fig. 3.32, Fig. 3.34 and Fig. 3.36, respectively. The WMicaD did not perform well on the Gaussian noise test. However, it provides good results on the other two tests in term of correlation coefficient. Examples of the estimated signatures of the three noise tests are shown in Fig. 3.33, Fig. 3.35 and Fig. 3.37.

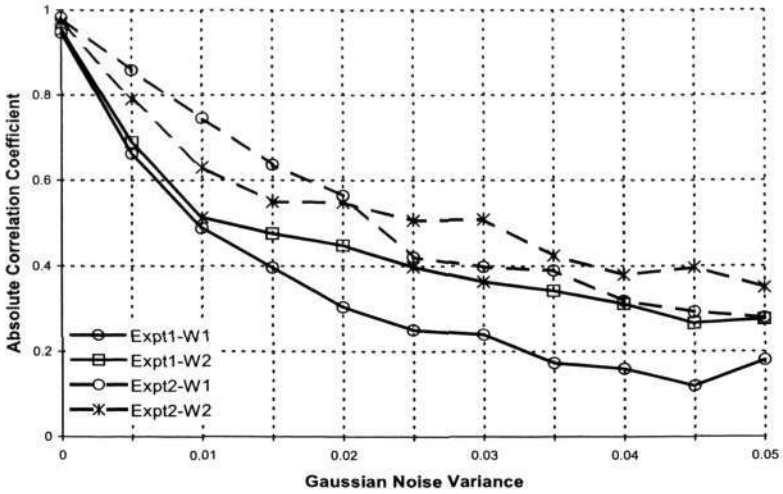
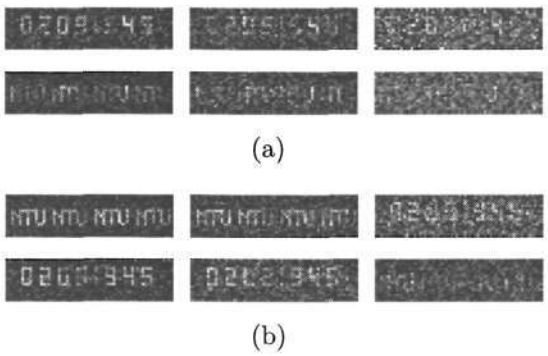


Figure 3.32: WMicaD results for the Gaussian noise test. Gaussian noise is zero mean and variance ranging from 0 to 0.05. Expt1: Lena image. Expt2: Baboon image.



(a)

(b)

Figure 3.33: The estimated signatures of WMicaD in Gaussian noise test. From left to right: the outputs of the noise test with noise variance of 0.005, 0.01 and 0.03. (a) Expt1 and (b) Expt2.

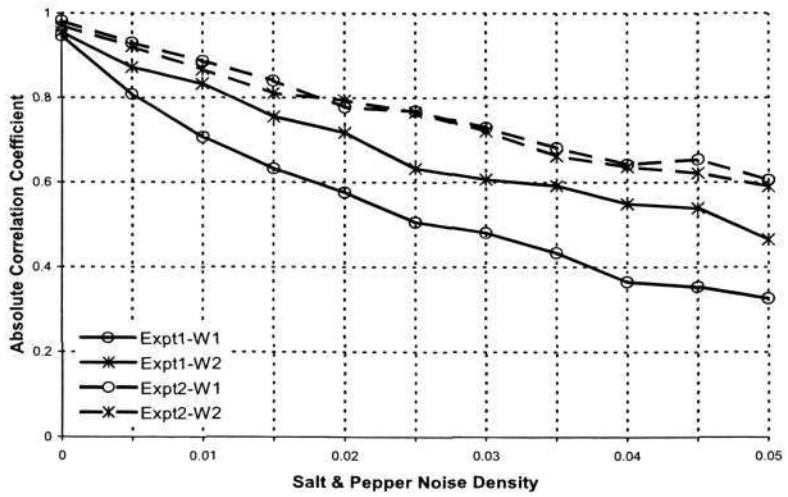
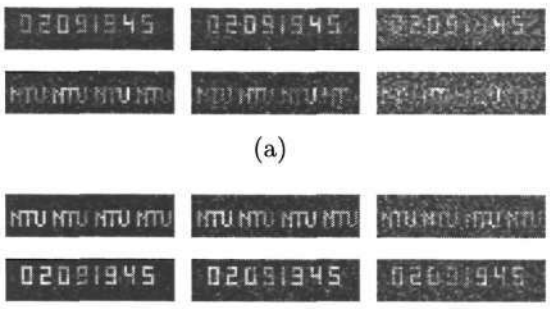


Figure 3.34: WMicaD results for the 'salt & pepper' noise test. Noise density is ranged from 0 to 0.05. Expt1: Lena image. Expt2: Baboon image.



(a)

(b)

Figure 3.35: The estimated signatures of WMicaD in 'salt & pepper' noise test. From left to right: the outputs of the noise test with noise density of 0.005, 0.01 and 0.03. (a) Expt1 and (b) Expt2.

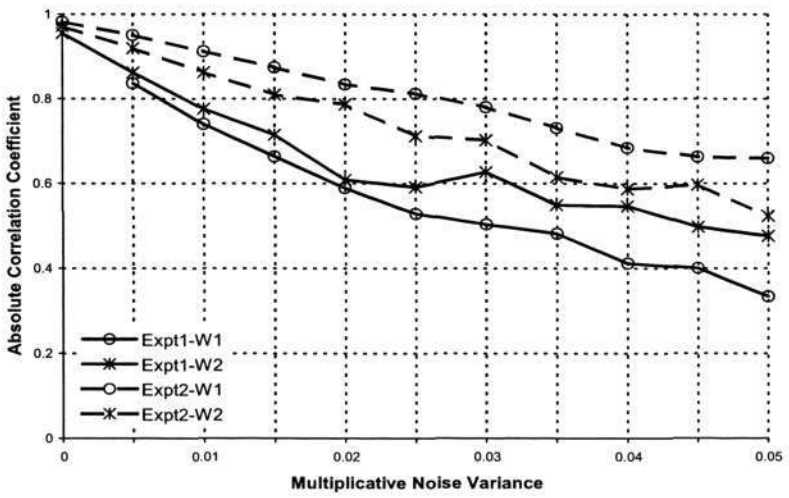


Figure 3.36: WMicaD results for the multiplicative noise test. Noise is uniformly distributed with zero mean and variance ranging from 0 to 0.05. Expt1: Lena image. Expt2: Baboon image.

(a)

(b)

Figure 3.37: The estimated signatures of WMicaD in multiplicative noise test. From left to right: the outputs of the noise test with noise variance of 0.005, 0.01 and 0.03. (a) Expt1 and (b) Expt2.

Resizing and rotation test

Here, we present the results of WMicaD in resizing and rotation tests. In resizing test, WMicaD was tested on both size-enlarged and size-reduced images. The test image size was varied from 200% (1024 × 1024) down to 25% (128 × 128) of the original size (in both dimensions). The performance of this test are shown in Fig. 3.38 and Fig. 3.39.

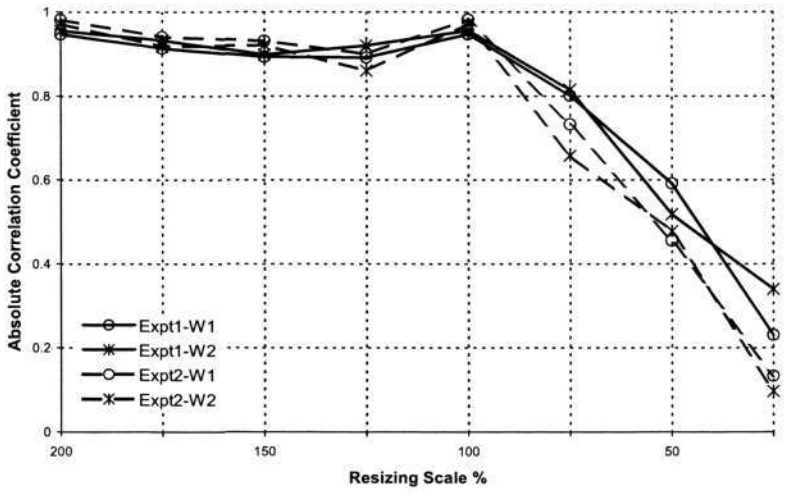
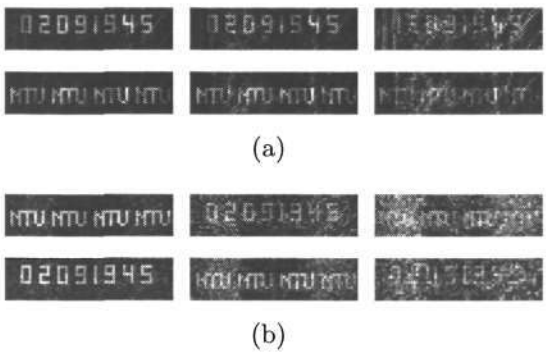


Figure 3.38: WMicaD results for the resizing test. The image size is varied from 200% to 25% of the original size in both dimensions. Expt1: Lena image. Expt2: Baboon image.



(a)

(b)

Figure 3.39: The estimated signatures of WMicaD in resizing test. From left to right: the outputs of resizing test with image size of 150% (768 × 768), 75% (384 × 384)and 50% (256 × 256) degree. (a) Expt1 and (b) Expt2.

In the rotation test (Fig. 3.40), the watermarked image was assumed to be rotated imperceptibly by a small angle before going to the extraction. The performance of WMicaD in the test was encouraging (Fig. 3.41). The extracted watermarks are still recognizable even when the rotate angle was up to 0.25 degree.

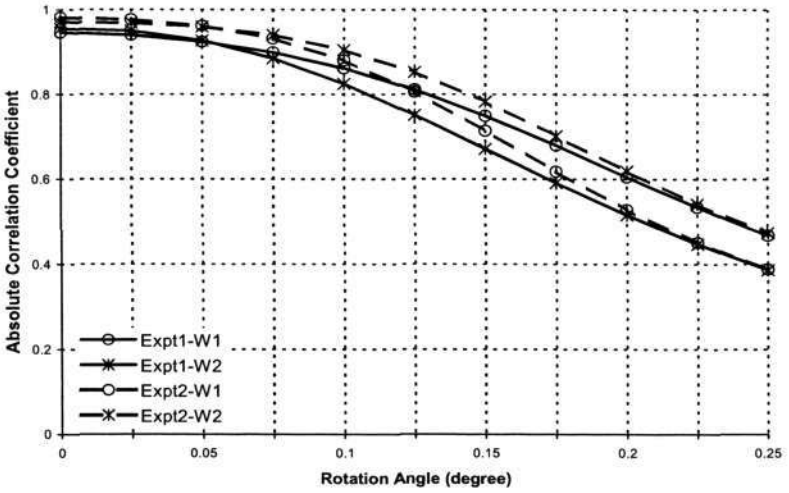
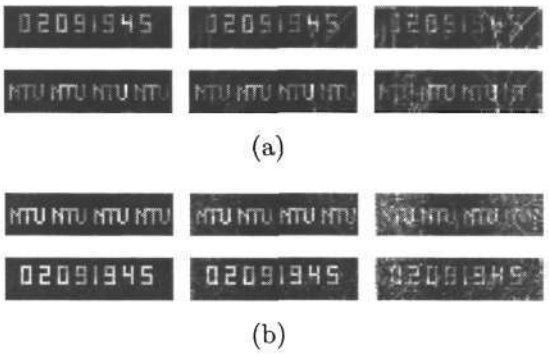


Figure 3.40: WMicaD results for the rotation test. The estimate quality is measured by the absolute correlation coefficient, $|r_S|$, between the signature estimate and its original one. The rotation angle is varied from 0 to 0.25 degree. Expt1: Lena image. Expt2: Baboon image.



(a)

(b)

Figure 3.41: The estimated signatures of WMicaD in rotation test. From left to right: the outputs of rotation test with rotating angle of 0.05, 0.10 and 0.20 degree. (a) Expt1 and (b) Expt2.

3.4.6 More on Robustness Experiments

The previous sections have provided detailed experimental analysis of WMicaD on the most commonly used images: Lena and Baboon. In this Section, we provide more statistical data on the performance of WMicaD using several experiments.

Five 512×512 gray images of different types were used as the original images, including Lena (portrait), Baboon (face), Grass (texture), Goldhill (scene) and Barbara (indoor). The embedding parameters were identical in all five simulations, with embedding strengths $\alpha = -7$, $\beta = 11$, key-image coefficient $\gamma = 9$, window half-lengths $L_1 = L_2 = 12$ and resizing factors $k_1 = 4$, $k_2 = 2$. The signatures used for embedding were two 16×64 binary images representing the word 'NTU' and the

(a) (b) (c) (d) (e)

Figure 3.42: The five watermarked images. (a) Lena image, (b) Baboon image, (c) Goldhill image, (d) Grass image and (e) Barbara images.

number '02091945'. The signatures are shown in Fig. 3.26. The watermarked images are shown in Fig. 3.42.

Each of the five test images went through four different attacks (JPEG compression, gray level reduction, image resizing and image rotation). The absolute correlation coefficients (3.3.16) for each attack were computed for all the five images and are tabulated in Tables 3.6 to 3.9. From these Tables, once can observe consistent resutls for all the attacks and all the test images.

The average absolute correlation coefficient over the five images for different attacks are shown in Figs. 3.43 to 3.46. From these Figures, we can see that WMicaD performed well on all the tests. It confirms our observations on the robustness of WMicaD as discussed in Section 3.4.5.

Table 3.6: Results of WMicaD on rotation test. Performance is measured by the absolute correlation coefficient.

Rotation	Lena		Baboon		Goldhill		Grass		Barbara		**Average**	
in degree	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$\overline{W1}$	$\overline{W2}$
0.025	0.94	0.95	0.98	0.97	0.98	0.98	0.99	0.99	0.97	0.96	0.97	0.97
0.05	0.92	0.93	0.96	0.96	0.96	0.94	0.98	0.98	0.95	0.92	0.96	0.95
0.075	0.90	0.88	0.93	0.94	0.93	0.89	0.96	0.96	0.92	0.85	0.93	0.91
0.10	0.86	0.82	0.88	0.90	0.87	0.81	0.96	0.93	0.88	0.77	0.89	0.85
0.125	0.81	0.75	0.80	0.85	0.79	0.73	0.89	0.90	0.81	0.67	0.82	0.78
0.15	0.75	0.67	0.71	0.78	0.69	0.65	0.83	0.84	0.74	0.58	0.74	0.70
0.175	0.68	0.59	0.62	0.70	0.59	0.57	0.75	0.78	0.65	0.49	0.66	0.63
0.20	0.60	0.52	0.53	0.62	0.49	0.50	0.67	0.71	0.57	0.41	0.57	0.55
0.225	0.53	0.45	0.45	0.54	0.40	0.44	0.59	0.64	0.50	0.34	0.50	0.48
0.25	0.47	0.39	0.39	0.47	0.33	0.39	0.52	0.58	0.43	0.29	0.43	0.42

Table 3.7: Results of WMicaD on JPEG compression test. Performance is measured by the absolute correlation coefficient.

JPEG	Lena		Baboon		Goldhill		Grass		Barbara		**Average**	
	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$\overline{W1}$	$\overline{W2}$
90	0.94	0.95	0.98	0.97	0.98	0.98	0.99	0.99	0.97	0.97	0.97	0.97
80	0.93	0.95	0.98	0.97	0.98	0.98	0.99	0.99	0.97	0.96	0.97	0.97
70	0.93	0.95	0.98	0.96	0.98	0.98	0.99	0.99	0.96	0.96	0.97	0.97
60	0.92	0.95	0.97	0.96	0.97	0.97	0.99	0.99	0.96	0.95	0.96	0.96
50	0.91	0.94	0.97	0.96	0.97	0.96	0.99	0.98	0.95	0.94	0.96	0.96
40	0.91	0.93	0.96	0.95	0.96	0.94	0.98	0.98	0.95	0.93	0.95	0.95
30	0.88	0.91	0.96	0.94	0.94	0.90	0.98	0.96	0.93	0.90	0.94	0.92
20	0.80	0.82	0.94	0.90	0.88	0.76	0.96	0.92	0.89	0.80	0.90	0.84
10	0.60	0.46	0.86	0.69	0.60	0.36	0.86	0.67	0.67	0.49	0.72	0.54

Table 3.8: Results of WMicaD on resizing test. The size of original image is 512×512. Performance is measured by the absolute correlation coefficient.

Resizing	Lena		Baboon		Goldhill		Grass		Barbara		**Average**	
	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$\overline{W1}$	$\overline{W2}$
200%	0.95	0.96	0.98	0.97	0.98	0.98	0.99	0.99	0.97	0.97	0.97	0.97
175%	0.91	0.93	0.94	0.92	0.98	0.98	0.99	0.99	0.97	0.97	0.96	0.96
150%	0.89	0.90	0.93	0.92	0.98	0.98	0.99	0.99	0.97	0.97	0.95	0.95
125%	0.89	0.92	0.90	0.86	0.99	0.98	0.99	0.99	0.97	0.97	0.95	0.95
100%	0.95	0.96	0.98	0.97	0.99	0.99	0.99	0.99	0.97	0.97	0.98	0.97
75%	0.80	0.81	0.73	0.66	0.98	0.98	0.99	0.99	0.97	0.97	0.89	0.88
50%	0.59	0.52	0.45	0.48	0.95	0.97	0.98	0.97	0.90	0.94	0.78	0.77
25%	0.23	0.34	0.13	0.10	0.70	0.77	0.78	0.85	0.66	0.73	0.50	0.56

Table 3.9: Results of WMicaD on gray level reduction test. Performance is measured by the absolute correlation coefficient.

Gray	Lena		Baboon		Goldhill		Grass		Barbara		**Average**	
Level	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$W1$	$W2$	$\overline{W1}$	$\overline{W2}$
256	0.95	0.96	0.98	0.97	0.99	0.99	0.99	0.99	0.97	0.97	0.98	0.97
128	0.94	0.94	0.98	0.97	0.99	0.98	0.99	0.94	0.97	0.95	0.98	0.96
64	0.94	0.60	0.98	0.93	0.98	0.96	0.99	0.99	0.97	0.90	0.97	0.87
32	0.93	0.88	0.98	0.71	0.98	0.98	0.99	0.99	0.97	0.97	0.97	0.90
16	0.90	0.84	0.97	0.95	0.95	0.88	0.99	0.97	0.93	0.84	0.95	0.90
8	0.64	0.40	0.94	0.79	0.66	0.44	0.93	0.87	0.75	0.44	0.79	0.59

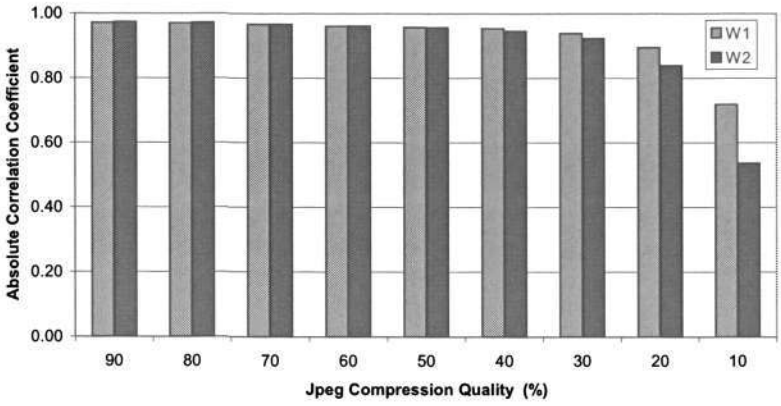


Figure 3.43: Average result of WMicaD on the JPEG compression test.

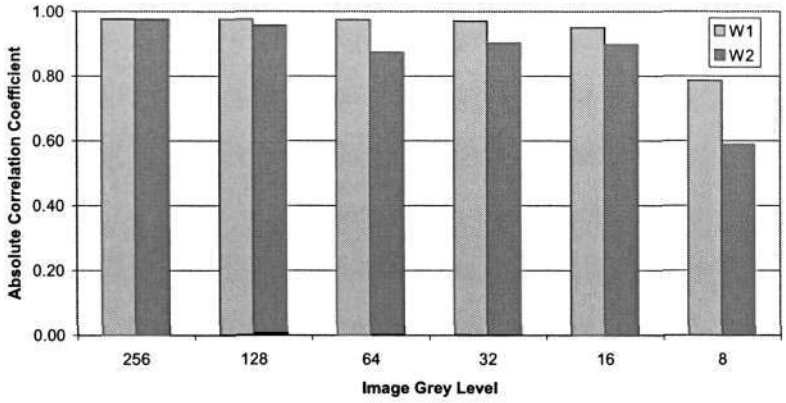


Figure 3.44: Average result of WMicaD on the gray level reduction test.

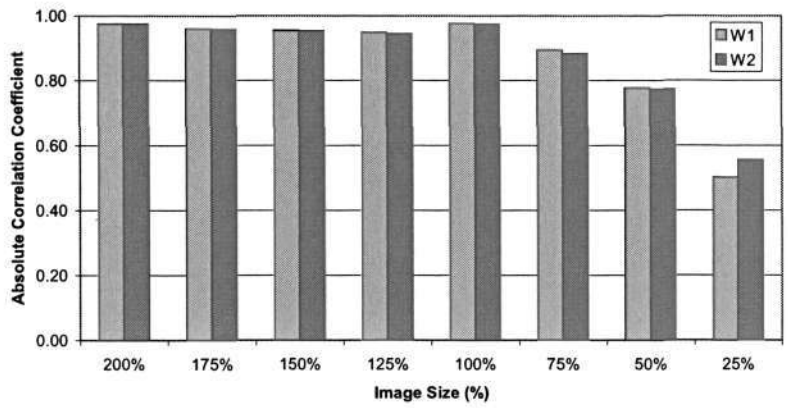


Figure 3.45: Average result of WMicaD on the image resizing test.

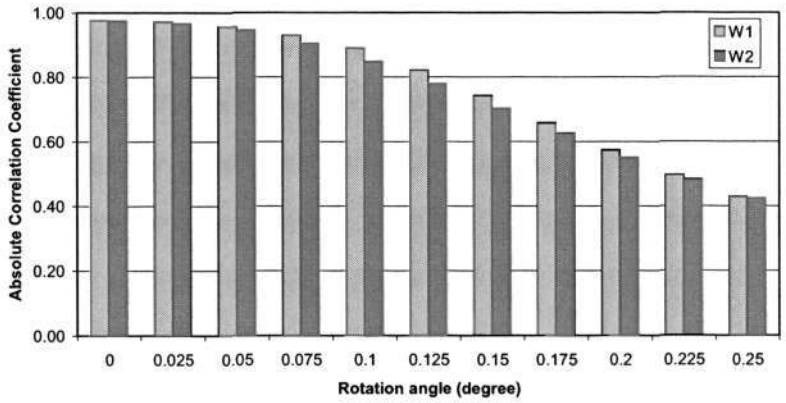


Figure 3.46: Average result of WMicaD on the image rotation test.

3.4.7 WMicaD for Detection of Tampered Area

The results in the previous Section have shown a very good performance of WMicaD method against severe modifications. This Section, we carry out an authentication test to illustrate the ability of WMicaD in detecting the intentionally tempered areas. Figure 3.47 shows a tampered Lena image in which a small portion of the image (the feather portion on the hat's tail area) was copied and maliciously overwritten to another place. Our goal is to detect this tampered area through the extraction process.

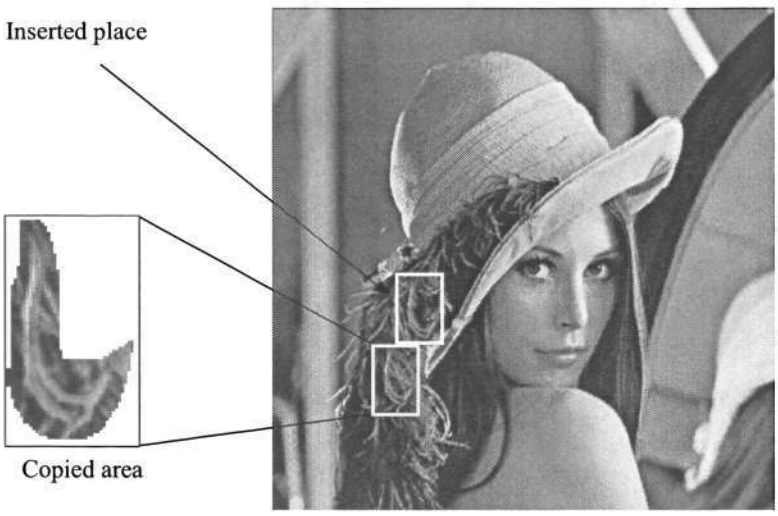


Figure 3.47: The image used in the tampering test. A small portion of the Lena image in the feather area of the hat's tail is copied and inserted to another place (the tampering area is magnified and shown on the left side of the tampered image).

Detecting the Tampered Area

As it is shown in Fig. 3.47, a small area from the watermarked image was copied and imperceptibly overwritten to another location. The tampered image was then put into the WMicaD extraction and we received three output images as shown in Fig. 3.48. The detection process was carried out on these images and subsequently putting them to the post-process. As it can be seen in Fig. 3.48, the tampered area is clearly noticeable in the watermark estimates, with the pixel values of the tampered area are much higher than the rest of the image.

(a) (b) (c)

Figure 3.48: Three output images Y_1, Y_2 and Y_3 of WMicaD in the tampering test (all images are of size 128×128). The tampered area can be discovered in the output Y_2 and Y_3 which correspond to the two watermark estimates.

Recovering the signatures

After successfully detecting the tampered area, we replaced the value of these tampered pixels (the area where pixel values are higher) in the estimated images (Fig. 3.48(b) and Fig. 3.48(c)) by the image average values. The newly-modified images were quantized to 256 gray level and then used as the inputs for the post-processing scheme. As it is shown in Fig. 3.49, the estimated watermarks and signatures are clearly visible after the replacement and quantization.

(a) (b) (c)

Figure 3.49: Two estimated watermarks and signatures after the tampered area is replaced. (a) The first watermark, (b) the second watermark and (c) the two estimated signatures.

3.4.8 Discussion on WMicaD Algorithm

We have introduced another novel ICA-based watermarking method called WMicaD. By following the full image approach mentioned in Section 3.2.2, the proposed method is robust against major attacks. The performance of WMicaD is stable in all the simulation on different tests with different host images. By utilizing the up-sizing and down-sizing operators, we are able to reduce the size of the additional support data (the key image) and consequently cut down the computational workload for the extraction considerably. Moreover, the special dual watermarks scheme provides a convenient way to embed into the original image, add the useful information and

make the key image public available. And last but not least, the dual watermarks brings the ability of detecting the tampered area in the WMicaD.

3.5 Conclusion

Watermarking by ICA is an interesting topic since the blind separation property of ICA algorithms provides many advantages for the embedding and extraction processes. On one hand, the block-based approach embeds the watermark information in the independent components (ICs). Thus, the information is distributed imperceptibly all over the image. The forgers, therefore, have difficulty in detecting the watermarks. The trade-off is, however, the watermarks are not very robust against strong modifications, and the method needs a lot of computation to calculate the ICs.

On the other hand, the second approach considers the full image and watermarks, each as one independent source, and the watermarked image as a mixture of the original image and the watermarks. The similarity between a watermarking model and an ICA model as shown in Section 3.2.2 makes it easy to build up the algorithm. General characteristic of those algorithms that follow this full image approach is a high performance in many robust tests. The trade-off is, however, the algorithms usually need original image or some large-sized image as the additional data for the extraction.

In this chapter, we propose two different methods for the watermarking problem by utilizing the advantage of ICA technique. Both methods, WMicaT and WMicaD, are based on the full image ICA-based approach. Hence, their experiments yield very good results on different tests. The watermarks can survive several major attacks. Comparing to the other reported methods, both WMicaT and WMicaD overcome the need of original image during the extraction by utilizing the image transpose (in WMicaT) and dual watermarks (in WMicaD). In addition, their additional data, the key image, can be made public available.

The use of up-sizing and down-sizing operators in embedding and extraction schemes brings more advantages to WMicaD. First, it allows the algorithm to work on small-sized images, reducing the computation and saving the storage space. And second, it gives WMicaD the ability to detect the tampered area, one of the very

useful abilities for image authentication.

Our two proposed methods, however, do have their disadvantages. WMicaT does not work on the image which is similar to its transpose, *i.e.*, $I \approx I^T$ since in this situation, I and I^T can not be considered as two independent sources anymore. Thus, the ICA technique will not be able to operate. Using dual special watermarks instead of image transpose, WMicaD does not suffer from the above issue, but it does not work well on the small image since it needs to down-size the image to at least 4×4 time smaller than the original. A summary of the advantages and disadvantages of the two proposed algorithms, WMicaT and WMicaD, are provided in Table 3.10.

Finally, the ICA approach is not only limited to digital image watermarking. In [101] and [232], the authors suggest an application of ICA as a tool for attacking and analyzing the robustness of a watermarking algorithm. Recent papers also mention the use of ICA for watermarking on different media such as ICA for video watermarking [199] and ICA for audio signal watermarking [211]. With more and more studies, we can see a promising future of ICA in watermarking applications.

Table 3.10: Summary the advantages and disadvantages of the two algorithm: WMicaT and WMicaD.

	Advantages
WMicaT and WMicaD	- Do not require the original image at the extraction site. - Additional data can made public available. - Robust against salient attacks.
WMicaD	- Working on size-reduced images brings computational advantage and storage advantage. - Can detect the tampered area. - Dual watermarks contain both fixed and variable information of the owner and the image.
	Disadvantages
WMicaT	- Can not work on the host image whose transpose is similar to the image, $I \approx I^T$.
WMicaD	- May not work well on small size image.

Chapter 4

Blind Source Separation by Nonlinear Independent Component Analysis

In the previous chapters we have shown the ability of linear ICA in various kinds of applications, especially in BSS. However, simple linear model appears to be inapplicable in many practical situations, for example, a digital satellite or microwave channels which are composed of a linear filter followed by a nonlinear amplifier [107, 181], or sensor arrays [164]. When linear models fail, the nonlinear ICA models appear to be powerful tools for modeling such BSS systems because of their higher approximation capabilities [106, 107, 202].

Despite great potential of nonlinear models, there have been only a few studies on nonlinear ICA. Complexity in computation [161, 162] and non-uniqueness of solutions [4, 88, 201] are the most challenging problems for researchers to extend and generalize ICA to nonlinear BSS models. Several approaches have been introduced either by applying constraints on sources signals [11, 78, 130, 141, 207] or by limiting nonlinear ICA to some specific models [4, 128, 202, 203, 237]. Among them, post nonlinear separation (PNL) introduced by Taleb and Jutten [202], is probably the most attractive model.

In this chapter, we provide our two contributions inspired by PNL model to solve nonlinear ICA problem. Our first method aims to provide a sequential online solution [149] while the second one aims to produce a more effective result in a hard nonlinear environment [150, 154, 155].

4.1 From Linear ICA to Nonlinear ICA: A Review

The linear mixing ICA model described in chapter 2 is modeled as

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{4.1.1}$$

where $\mathbf{A}_{n\times m}$ is an unknown mixing matrix, $\mathbf{s} = [s_1, s_2, ..., s_m]^T$ is a source vector whose elements represent m independent hidden source signals, and $\mathbf{x} = [x_1, x_2, ..., x_n]^T$ is a data vector whose elements represent n observed mixtures. The goal of ICA is to separate the hidden sources from their mixtures by using the independent criteria.

A straightforward extension of (4.1.1) is to replace the linear mixing matrix by a general nonlinear function. The model is then described by

$$\mathbf{x} = \mathcal{F}(\mathbf{s}) \tag{4.1.2}$$

where $\mathbf{x}$ and $\mathbf{s}$ denote the data and source vectors as above, and $\mathcal{F}$ is an unknown real-valued m-component mixing function. To simplify the problem, let us assume that the number of independent sources equals the number of observations, that is $m = n$. The goal of retrieving unknown sources is accomplished by constructing a function $\mathcal{G} : \mathbb{R}^n \to \mathbb{R}^n$ that yields components

$$\mathbf{y} = \mathcal{G}(\mathbf{x}) \tag{4.1.3}$$

which are statistically independent. In other words, the goal is to obtain

$$y_i = k_i s_{\sigma(i)}, \qquad i = 1, 2, ..., n \tag{4.1.4}$$

where σ is a permutation on $\{1, 2, ..., n\}$, and k_i is a scalar [107]. The fundamental problem of nonlinear ICA, however, is the non-uniqueness of solutions. Without additional assumptions on source signals or mixing model, there always exists infinite outputs that satisfy the independent criteria but are not the estimates of the hidden sources. Details on the existence and uniqueness of nonlinear ICA are given in the following section.

4.1.1 Existence and Uniqueness of Nonlinear ICA

Because of its vital role in the construction of nonlinear ICA algorithms, issues on existence and uniqueness of solutions have been addressed by many authors [4, 66, 88, 107, 201]. We have to clear three questions:

1. Existence: Can the model produce independent components as output?

2. Uniqueness: Is this output unique?

3. Separability: Is the output the estimate of the unknown sources?

The answers are believed to be the direct consequences of Darmois's work on factorial analysis [59].

Existence

In general nonlinear ICA problem, there exists at least one solutions. Given a data vector $\mathbf{x}$, there always has a function $\mathcal{G}$ so that $\mathbf{y} = \mathcal{G}(\mathbf{x})$ in (4.1.3) have independent components. The following example, introduced by Hyvãrinen and Pajunen in [88], will illustrate a constructive procedure to form a solution.

The construction, seen as a generalization of Gram-Schmidt orthogonalization, is defined recursively. Let us assume that m independent random variables $y_1, y_2, ..., y_m$ are available and they follow a joint uniform distribution in the unit cube $[0,1]^m$ (it is not a restriction of y_i to have uniform distribution, this follows directly from the recursion). Let x be a random variable, and $a_1, a_2, ..., a_m$ and b be some scalars. Define

$$\begin{aligned} g(a_1, a_2, ..., a_m, b; p_{y,x}) &= P(y_{m+1} \leq b | y_1 = a_1, y_2 = a_2, ..., y_m = a_m) \quad (4.1.5) \\ &= \frac{\int_{-\infty}^{b} p_{y,x}(a_1, a_2, ..., a_m, \xi) d\xi}{p_y(a_1, a_2, ..., a_m)} \end{aligned}$$

where $p_y(\cdot)$ is the marginal probability density of $(y_1, y_2, ..., y_m)$ and $p_{y,x}(\cdot)$ is the probability density of $(y_1, y_2, ..., y_m, x)$, and $P(\cdot|\cdot)$ denotes the conditional probability. The term $p_{y,x}$ is to remind that g depends on the joint probability distribution of $y_1, y_2, ..., y_m$ and x. When $m = 0$, g becomes the cumulative distribution of x. With the definition of g in (4.1.5), we have the following theorem [88]:

Theorem 4.1. *Assume that $y_1, y_2, ..., y_m$ are independent scalar random variables which follow a joint uniform distribution in the unit cube $[0,1]^m$. Let x be any scalar random variable (such that the joint distribution of $y_1, y_2, ..., y_m, x$ has a probability density with respect to Lebesgue measure of $\mathbb{R}^{m+1}$. Define g as in (4.1.5), and set*

$$y_{m+1} = g(y_1, y_2, ..., y_m, x; p_{y,x}) \quad (4.1.6)$$

Then y_{m+1} is independent from the $y_1, y_2, ..., y_m$. In particular, the variables $y_1, y_2, ..., y_{m+1}$ are jointly uniformly distributed in the unit cube $[0,1]^{m+1}$.

Clearly, the theorem provides a construction of solution for nonlinear ICA. It affirms the statement: there always exists at least one solution to the problem.

Uniqueness and Separability

The above construction not only ensures the existence of a solution but also shows the non-uniqueness of nonlinear ICA. There may have more than one result that satisfy the independent criteria, but may not satisfy the goal of retrieving the unknown sources stated in (4.1.4). Hence, in what conditions, ICA model yields unique result? In other words, which ICA model can achieve the separability? Several authors [88, 107, 201] have tried to identify the issue from different viewpoints. Their studies are summarized in the following paragraphs.

Definition 4.1 A one-to-one mapping $\mathcal{H}$ is called trivial, if it transforms any random vector $\mathbf{s}$ with independent components into a random vector with independent components.

Let us denote the set of trivial transformation by R. Trivial transformation is characterized by the following proposition

Proposition 4.1 A one-to-one mapping $\mathcal{H}$ is trivial if and only if it can be written as

$$\mathcal{H}_i(u_1, u_2, ..., u_n) = h_i(u_{\sigma(i)}), \qquad i = 1, 2, ..., n \tag{4.1.7}$$

where h_i are arbitrary functions, and σ is any permutation over $\{1, 2, ..., n\}$.

Let $\mathcal{H} = \mathcal{G} \circ \mathcal{F}$ be a global transformation, then the output of the model can be represented by $\mathbf{y} = \mathcal{G} \circ \mathcal{F}(\mathbf{s}) = \mathcal{H}(\mathbf{s})$. It becomes clear that the goal of our nonlinear ICA problem is, using independent assumption, to find $\mathcal{G}$ such that the global transformation $\mathcal{H}$ is trivial. However, it is impossible to satisfy the separability requirement without imposing additional constraints on $\mathcal{H}$ as it is shown below.

In the general case where $\mathcal{H}$ does not have any particular form, there is a well-known statistical result shows that preserving independence is not a strong enough

constraint to ensure separability in the context of (4.1.7). The result has been established by Darmois in [59]. In his work, the author introduced a simple constructive method for decomposing any random vector by a non-trivial mapping into independent variables. That is, there exist non-trivial transformations $\mathcal{H}$ which mix the variables while preserving their statistical independence. Hence, separability is impossible for general nonlinear transformations by only using the statistical independent assumptions.

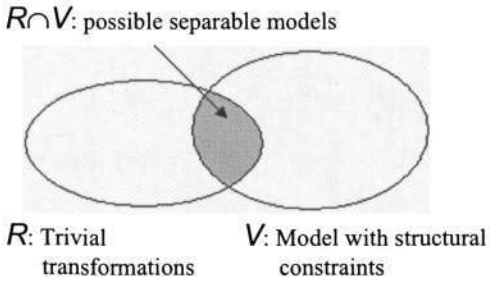


Figure 4.1: The intersection between structural constrained transformation and trivial transformation. R and V denote the sets of trivial transformation and model with structural constraints, respectively. Shaded area $R \cap V$ represents the set of possibly separable models [201].

The previous negative result is only for general nonlinear ICA where there is no constraint applied on the transformation $\mathcal{H}$. In [201], the authors argued that by constraining $\mathcal{H}$ to a certain set of transformation, denoted by V, can dramatically reduce the strong indeterminacies and eventually lead to the separability. Figure 4.1 illustrates the interaction between these two sets where their intersection is the set of possibly separable models. In summary, sources are separable and able to be restored up to a trivial transformation belonging to $R \cap V$.

4.1.2 Different Approaches to Nonlinear ICA

Probably the first attempt to introduce a nonlinear ICA algorithm was reported in [34] where nonlinearity is assumed to be known. Deco and Parra et al. [60, 62, 165–167] developed nonlinear methods that were based on volume conserving symplectic transformations. Their constraint of volume conservation, however, is somewhat arbitrary, and hence the methods are usually not able to recover the original sources [106].

A general nonlinear ICA algorithm was introduced by Pajunen et al. in [161] using Self-Organizing Map (SOM). SOM learns a nonlinear mapping from the data

to a rectangular grid. With suitable modifications, the mapping provided by SOM is roughly uniformly distributed on the grid. The marginal densities along the sides of the rectangular grid become then statistically independent [161,162]. However, disadvantage of using SOM for nonlinear ICA is that computational complexity increases rapidly with the number of the sources, preventing the method from large-scale nonlinear applications. Other results on SOM-based method for linear and nonlinear ICA are given in [81].

The variational Bayesian (or Bayesian ensemble learning) approach [128,129] utilizes an approximation which is fitted to the posterior distribution of the parameters to be estimated. Because of the simplicity, Gaussian distribution is usually selected as the approximation. The approximative posterior distribution is fitted to the posterior distribution estimated from the data using the Kullback-Leibler (KL) divergence [45, 95]. The algorithm is based on the fact that KL divergence is sensitive to the mass of the distributions rather than to some peak value, resulting in robust estimates. In [128], a MLP network with one hidden layer is used for modeling nonlinearity. The necessary regularization for nonlinear ICA is achieved by choosing the model and sources that have most probably generated the observed data.

In addition to the ensemble learning approach, MLP networks have been used in some other nonlinear ICA methods. Autoassociative MLP networks introduced in [80] have been tried in solving nonlinear ICA. Both generative model and its inversion are learned simultaneously, but separately without utilizing the fact that the models are connected. Autoassociative MLPs have shown some success in nonlinear data representation but generally they suffer from slow learning. Yang et al. [226] proposed another algorithm using mutual information and applying natural gradient method on a MLP network and experimented with post nonlinear mixtures. Fisher and Principle [68] contributed another MLP structure with maximum entropy method for nonlinear ICA.

Kernel methods can also be used for linear and nonlinear ICA. An early kernel-based ICA algorithm can be found in [12], and more ICA methods with kernels are reported in [78, 141]. However, problem of such algorithms is how an appropriate nonlinear transformation is chosen. A Radial Basic Function (RBF) network approach for ICA is introduced in [204] by Tan and Zurada. Their contrast function consists of

mutual information and partial moments of the estimated sources, which are used to provide the regularization needed in nonlinear ICA. The authors have shown several experiments on artificial mixture sets with good result. And many more different nonlinear ICA approaches can be mentioned here such as mutual information [3, 4], spline network [196] and genetic algorithms for improving the estimation of parameters of the separating mapping. Several authors also contribute their initial ideas without presenting a complete study, for example, general Bayesian Ying-Yang framework [225], energy-based probability density model [82], and Gaussianization technique [235, 237]. For more information, please see [106].

In addition to general nonlinear ICA approaches, methods applied for a particular constrained nonlinear ICA have gained a lot of attention. As discussed previously in Section 4.1.1, one must set some constraints on the model in order to receive a unique solution. In [11], the authors proved that nonlinear mixtures having distribution contained in a parallelogram can be uniquely separated, and using independent criteria, nonlinearity can be estimated. Finally, Post Nonlinear mixture (PNL) model, introduced by Taleb et al. [202], is probably the most well-studied specific model which constrains the nonlinearity to a univariate nonlinear function. Many algorithms tailored for PNL model can be seen in [1, 97, 189, 201, 207, 234, 237]. The details of this model will be given in the next section.

4.2 Post Nonlinear Mixtures Model

Post Nonlinear model (PNL), first introduced by Taleb and Jutten in [202], is probably the most widely studied model to date. The model is simple and realistic enough to simulate many practical systems, for example, the microwave communication application [181] and biological system [119].

The post nonlinear mixture as shown in Fig. 4.2 models the unknown environment as a two-stage mixing system. The hidden original signals are assumed to be linearly mixed in the first stage. Nonlinear distortions happen in the second stage where each nonlinearity is applied independently on each signal. Mathematically, the nonlinear

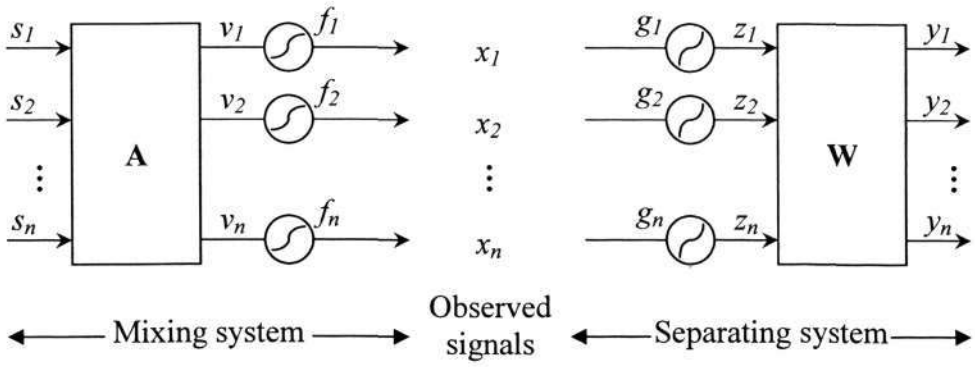


Figure 4.2: Post Nonlinear (PNL) mixing-separating model. Mixing system (assumed to be hidden) includes a linear mixing stage followed by a componentwise nonlinear stage. Separating system is an inverse process. A linearization stage eliminates the nonlinearity of the observed signals. The next stage linearly separates the hidden sources from linearized signals.

observations have the following form

$$x_i(t) = f_i(\sum_{j=1}^{n} a_{ij} s_j(t)) \qquad i = 1, 2, ...n \tag{4.2.1}$$

or in a more specific two-stage form

$$v_i(t) = \sum_{j=1}^{n} a_{ij} s_j(t) \tag{4.2.2}$$

$$x_i(t) = f_i(v_i(t)) \qquad i = 1, 2, ..., n \tag{4.2.3}$$

where s_j denote hidden sources, a_{ij} represents the (i, j)-th scalar entry of the mixing matrix $\mathbf{A}$, and f_i is an invertible nonlinear function. Time index t is presented in the equation to indicate the time relation[1]. The PNL model (4.2.1) corresponds to systems in which the transmission across a channel is linear and memoryless, whereas sensors and their preamplifiers introduce memoryless nonlinear distortions [202]. It belongs to L-ZMNL[2] family which suits perfectly for many real-world applications, such as sensor array processing [164], satellite and microwave communication [181], and biological systems [119].

To estimate the mixing system and to retrieve original signals, an inverse two-stage model, *i.e.*, a separating system, is constructed. First, the observed signals are linearized to eliminate their nonlinearity. Second, a normal linear ICA algorithm is applied to separate the source signals from these linear mixtures. The separating

[1] In some expressions, t can be dropped off to make the equation compact.
[2] L stands for Linear and ZMNL stands for Zero-Memory NonLinearity.

model, therefore, can be expressed by the following form

$$y_i(t) = \sum_{j=1}^{n} w_{ij} g_j(x_j(t)) \qquad i = 1, 2, ...n \tag{4.2.4}$$

or a two-stage form

$$z_j(t) = g_j(x_j(t)) \qquad j = 1, 2, ..., n \tag{4.2.5}$$

$$y_i(t) = \sum_{j=1}^{n} w_{ij} z_j(t) \qquad i = 1, 2, ..., n \tag{4.2.6}$$

where w_{ij} is the (i, j)-th entry of the demixing matrix, $\mathbf{W}$.

4.2.1 Separability of PNL

The PNL model attracts a lot of researchers because of its practicality and separability. As mentioned above, separability is a hard issue of nonlinear ICA. In [202], the authors have given detail proof of the separability of PNL model. The following lemma and its proof is restated from their arguments.

Lemma 4.2. *Let* $(\mathbf{f}, \mathbf{A})$ *be a PNL mixture and* $(\mathbf{g}, \mathbf{W})$ *be a PNL separation structure, where we have the following*

- $\mathbf{A}$ *is a regular matrix and has at least two nonzero entries per row or per column.*
- f_i, $i = 1, 2, ..., n$ *are differentiable invertible functions.*
- $\mathbf{W}$ *is a regular matrix.*
- $h_i = g_i \circ f_i$ *satisfies* $\forall u \in \mathbb{R}, h_i(u) \neq 0$, *for all* $i = 1, 2, ..., n$.

Suppose that each source s_i *accepts a density function that vanishes at one point at least. Then, the output* $\mathbf{y}$ *of the separation structure has mutually independent components if and only if the* h_i *components are linear, and* $\mathbf{W}$ *is a separating matrix.*

The proof is given in the Appendix C. With this lemma, PNL is a separable model, *i.e.*, this model produces unique result.

4.2.2 Algorithms Tailored for PNL Model

The initial algorithm was introduced in the same paper with PNL model [202]. The authors use mutual information between the output components $y_1, y_2, ..., y_n$ as the cost function for updating both the linear and nonlinear systems. The cost function is estimated from output vector $\mathbf{y}$. The probability density function is calculated first, then it is used to compute the cost function. The authors suggest that estimating the probability density function (*pdf*) by Gram-Charlier expansion is appropriate only for mild post-nonlinear distortions. For hard nonlinear, kerned-based estimation is probably preferable.

Some other authors propose different approaches for PNL model. In [1], Achard et al. introduce a parameterization of the mutual information criterion with simple piecewise constant functions for parameterization. Another approach by Ziehe et al. [234] uses an alternating conditional expectation (ACE) method for approximating the inversion of the post-nonlinearities f_i. In [235, 237], Ziehe et al. improve the method by computing the inverse functions directly from observed data. The key linearization process approximates linear mixtures with gaussian distributions. From the linearized mixtures, original signals are estimated by a linear ICA algorithm, TDSEP [236].

In [171] Peng et al. introduce a semi-parametric hybrid neural network model that bases on MLP network for separating post-nonlinear mixtures. Their method has advantage in dealing with cross-channel post-nonlinearities. Similarly, another PNL-tailored model using adaptive spline neural networks is proposed in [196]. In [183], Puntonet et al. combine a geometric approach with neural network learning for a special class of post-nonlinear mixtures, which are assumed to be powers of linear mixtures of the source signals. Another geometrical approach for separating bounded sources in PNL mixtures has been proposed by Zadeh et al. in [11]. A good review of post nonlinear algorithms can be found in [106].

4.3 kPNL: Kurtosis-based Post Nonlinear ICA

Our first contribution to nonlinear ICA was inspired by the practical PNL model. The proposed algorithm takes advantage of the Multi-Layer Perceptron (MLP) networks

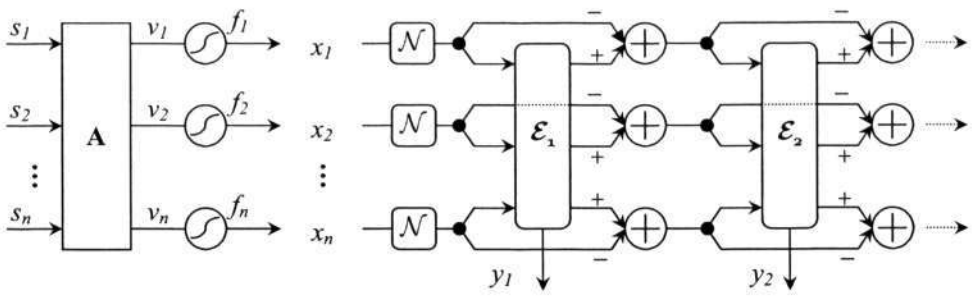


Figure 4.3: Kurtosis-based PNL mixing-separating model. Separating system include neural network components $\mathcal{N}$ and a chain of sequential linear extraction modules $\mathcal{E}_i$, $i = 1, 2, \ldots$.

with back-propagation training algorithm to simulate the nonlinear functions and a sequential extraction ICA algorithm to estimate the hidden signals. A scheme of the model is shown in Fig. 4.3 with a set of neural network components $\mathcal{N}_i$ $(i = 1, 2, ..., n)$, used for eliminating the nonlinearity in the observed signals, and a sequence of linear extraction modules, $\mathcal{E}_i$ $(i = 1, 2, ...)$.

Sequential ICA extraction [6, 48, 49] is a process that extract the hidden sources one by one. When an original source is estimated, it is subtracted from the observed data to obtain a new set of input data. The extraction is then repeated on this new input data. Comparing to simultaneous separation, sequential extraction does not require information about the number of sources, and is more flexible for us to choose how many signals to be estimated. Apparently, this one by one extraction approach is more suitable in practice than the simultaneous one, since in real world applications, it is not easy to know exactly the number of source signals. Besides, in many applications, it is not necessary to obtain all the signals, only the most significant signals are supposed to be estimated.

Taking these advantages from sequential extraction, our Kurtosis-based sequential PNL method approach, named as kPNL [149], has the following characteristics

1. Unlike simultaneous separation, kPNL does not require information about the number of sources.

2. Only important signals need to be extracted. For example, if the source signals are mixed with many noise sources, users may be able to extract the signals with some desired properties.

3. Kurtosis-based updating function can be computed directly from data. It is less

complex than some updating functions used in simultaneous separation methods such as Kullback-Leibler (KL) divergence [202].

4.3.1 kPNL Sequential Extraction Model

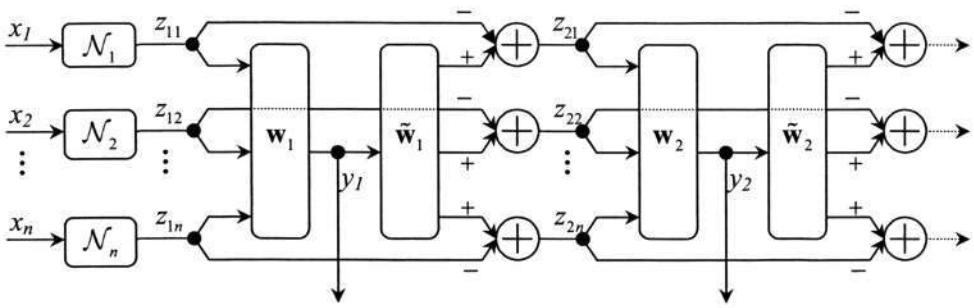


Figure 4.4: kPNL full extraction model. n MLP networks are used to eliminate nonlinearity of the observed signals. From the linearized signals, estimates of the original hidden sources are separated sequentially by kurtosis-based linear ICA model.

Shown in Fig 4.4 is the detail of kPNL separating system, including n neural network components $\mathcal{N}_i$ followed by a sequence of linear extraction modules $\mathcal{E}_i$ ($i = 1, 2, ...$). Each module $\mathcal{E}_i$ is combined by a demixing vector $\mathbf{w}_i$ and a deflation vector $\tilde{\mathbf{w}}_i$. In the first step, each input x_i is taken by a corresponding neural network $\mathcal{N}_i$. The neural network is trained to simulate the inverse function $h_i = f_i^{-1}$. In the second step, the outputs of neural networks $\mathbf{z}_1 = [z_{11}, z_{12}, ..., z_{1n}]^T$ are passed through a demixing vector $\mathbf{w}_1$ to extract the first signals y_1. Third step, a deflation vector $\tilde{\mathbf{w}}_1$ is applied to subtract y_1 out of z_{1i} ($i = 1, 2, ..., n$) and we get a new input $\mathbf{z}_2 = [z_{21}, z_{22}, ..., z_{zn}]^T$. The demixing and deflating processes continue with $\mathbf{z}_2$ and so on.

In this work, we use the Multi Layer Perceptron (MLP) with a modified back propagation learning algorithm as the neural networks, $\mathcal{N}_i$. Traditional MLP applies a supervised learning algorithm on desired data set to train the model. However, in the nonlinear ICA problem, there is no desired data. Hence, kPNL utilizes an unsupervised learning scheme with kurtosis-based cost function for training. The same kurtosis-based cost function is also used for training the linear extraction model, i.e, for learning the demixing vector $\mathbf{w}_i$.

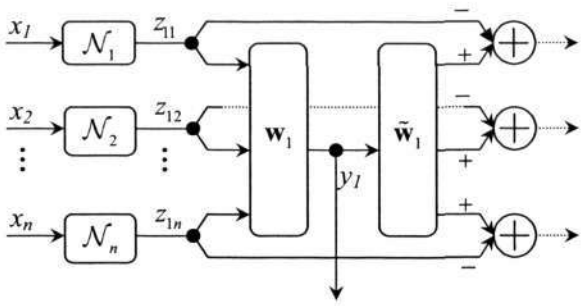


Figure 4.5: Extraction of the first signal y_1. Kurtosis criterium is used for training n MLP components $\mathcal{N}_i$, $i = 1, ..., n$, and the first demixing vector $\mathbf{w}_1$.

Extraction of the First Signal

The first stage of kPNL method is to train the MPLs, $\mathcal{N}_i$ $(i = 1, ..., n)$, and to extract the first signal y_1. Scheme of the first signal extraction is shown in Fig 4.5. To simplify the problem, the input $\mathbf{x}$, is assumed to be centered and whitened by a pre-process. Given the nonlinear observations x_i, the linearized outputs z_{1i} are expressed as

$$z_{1i} = \mathcal{N}_i(x_i) \qquad i = 1, 2, ..., n \tag{4.3.1}$$

and the first output y_1 is expressed as

$$y_1 = \mathbf{w}_1^T \mathbf{z}_1 = \sum_{j=1}^{n} w_{1i} z_{1i} \tag{4.3.2}$$

where $\mathbf{z}_1 = [z_{11}, z_{12}, ..., z_{1n}]^T$. To extract the signal, we extend the Kurtosis-based method proposed in [48] to our nonlinear case. By minimizing the following cost (loss) function, we receive the first estimate

$$\mathcal{L}_1(\mathbf{w}_1) = -\frac{1}{4}|\kappa_4(y_1)| \tag{4.3.3}$$

where $\kappa_4(y_1)$ is normalized kurtosis defined by

$$\kappa_4(y_1) = \frac{E[y_1^4]}{E[y_1^2]^2} - 3 \tag{4.3.4}$$

As discussed in [6], it is easy to show that for the pre-whitened input signals $\mathbf{x}$, the normalized kurtosis satisfies the following relations

$$\begin{aligned} \kappa_4(y_1) &= \kappa_4(\textstyle\sum_{i=1}^n w_{1i} g_i(x_i)) \\ &= \kappa_4(\textstyle\sum_{i=1}^n w_{1i} g_i \circ f_i(\sum_{j=1}^n a_{ij} s_j)) \\ &= \kappa_4(\mathbf{w}_1^T \mathbf{A}\mathbf{s}) = \kappa_4(\mathbf{e}^T \mathbf{s}) \\ &= \textstyle\sum_{i=1}^n \frac{e_i^4}{\|\mathbf{e}\|} \kappa_4(s_i) \end{aligned} \tag{4.3.5}$$

where $\mathbf{e}^T = \mathbf{w}_1^T\mathbf{A}$ and $\|\mathbf{e}\| = 1$. Hence, the normalized kurtosis satisfies the lemma given in [63]. The cost function does not contain any spurious local minima, thus all the minima correspond to the source signals [6]. Applying standard stochastic gradient descent to minimize the cost function (4.3.3) derives the following learning rule [48]

$$\mathbf{w}_1(k+1) = \mathbf{w}_1(k) + \mu_1(k)\varphi(y_1(k))\mathbf{z}_1(k) \tag{4.3.6}$$

where $\mathbf{z}_1$ is pre-whitened vector, μ_1 denotes the learning rate and $\varphi(y_1)$ is computed as

$$\varphi(y_1) = sgn(\kappa_4(y_1))\left(y_1 - \frac{m_2(y_1)}{m_4(y_1)}y_1^3\right)\frac{m_4(y_1)}{m_2^3(y_1)} \tag{4.3.7}$$

where sgn denotes the sign function, $sgn(u) = 1$ if $u \geq 0$ and $sgn(u) = -1$ if $u < 0$. The higher order moments m_2 and m_4 can be estimated online through the following formula

$$m_p(k+1) = (1-\mu)m_p(k) + \mu|y_1(k)|^p \tag{4.3.8}$$

with parameter $\mu > 0$ and $m_p(k) \cong E[|y_1(k)|^p]$ where $E[\cdot]$ denotes the expectation.

After updating the demixing vector $\mathbf{w}_1$, the next important step is to update the MLP components. Since the same network structure is used in all n components, we only need to construct one learning rule and then apply to all n MLPs. For more information on MLP structure and its back-propagation learning algorithm, please see [29, 76, 80]. In this work, we use the notations and matrix-based formulas taken from [76] whose details are given in Appendix B.

Assuming that our MLP component is a m-layered neural network. Let $\mathbf{V}^i$ and $\mathbf{b}^i$ be the weight matrix and the bias vector of the i-th layer, respectively. Applying the back-propagation algorithm yields the following learning rules

$$\mathbf{V}^i(k+1) = \mathbf{V}^i(k) - \alpha\mathbf{v}^i(\mathbf{a}^{i-1})^T \tag{4.3.9}$$

$$\mathbf{b}^i(k+1) = \mathbf{b}^i(k) - \alpha\mathbf{v}^i \tag{4.3.10}$$

where $\mathbf{V}^i$, $\mathbf{b}^i$ and $\mathbf{a}^i$ denote the weight matrix, the bias vector and the output vector of the i-th layer, respectively. The sensitivity function, $\mathbf{v}^i$, is computed as

$$\mathbf{v}^i = \dot{\mathbf{F}}^i(\mathbf{s}^i)(\mathbf{V}^{i+1})\mathbf{v}^{i+1} \qquad i = m-1, ..., 1 \tag{4.3.11}$$

where $\mathbf{s}^i$ denotes the sum (also called net input vector) of the i-th layer, and with

$$\dot{\mathbf{F}}^i(\mathbf{s}^i) = diag[\dot{f}^i(s_1^i), ..., \dot{f}^i(s_{N^i}^i)] = \begin{bmatrix} \dot{f}^i(s_1^i) & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \dot{f}^i(s_{N^i}^i) \end{bmatrix} \tag{4.3.12}$$

where $\dot{f}^i(\cdot)$ is the derivative of the activation function $f^i(\cdot)$. More details on MLP structures and the above notations are given in Appendix B.

Until know, we are following the standard back-propagation to derive the learning rule. However, in order to adapt this supervised algorithm to our unsupervised kPNL model, we apply a major change in the computation of the last sensitivity function $\mathbf{s}^m$. Instead of using the desired outputs, which is not available in nonlinear ICA model, we utilize the cost function $\mathcal{L}_1$ in (4.3.3) to construct the learning algorithm. Using cost function $\mathcal{L}_1$ and stochastic gradient descent, the last sensitivity function $\mathbf{v}^m$ is computed by

$$\mathbf{v}^m = \dot{\mathbf{F}}^m(\mathbf{s}^m)\mathbf{w}_1(k)\varphi(y_1) \tag{4.3.13}$$

where $\mathbf{w}_1(k)$, $\varphi(y_1)$ and $\dot{\mathbf{F}}^m(\mathbf{s}^m)$ are respectively defined in (4.3.6), (4.3.7) and (4.3.12). With this modified learning scheme, MLP components can be trained without any supervised signal. The updating algorithm for estimating the nonlinear inverse functions and the first demixing vector was fully constructed.

Extraction of the Remaining Signals

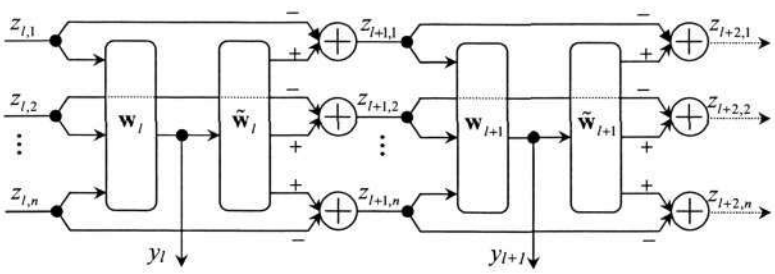


Figure 4.6: Extraction of the remaining signals. Linear kurtosis-based ICA algorithm is used to estimate the hidden sources one-by-one.

After extraction of the first signal, the next step is to remove it from the mixtures by a deflation procedure that is shown in Fig. 4.6. The subtracted data are used as input for extracting the next signal. The procedure is applied recursively to estimate the remaining signals.

Assuming that the first l signals, $y_1, y_2, ..., y_l$ have been successfully extracted, the goal now is to obtain the next signal, y_{l+1}. First, the l-th estimate, y_l, is subtracted from the mixtures to generate inputs, $\mathbf{z}_{l+1}$, for the next linear extraction module. The subtraction can be expressed by

$$\mathbf{z}_{l+1}(k) = \mathbf{z}_l(k) - \tilde{\mathbf{w}}_l(k) y_l(k) \tag{4.3.14}$$

where the deflation vector, $\tilde{\mathbf{w}}_l$, is obtained by minimizing the following cost function

$$\mathcal{L}_l(\tilde{\mathbf{w}}_l) = \frac{1}{p}\|\mathbf{z}_{l+1}\|^p \tag{4.3.15}$$

with $\mathbf{z}_l = [z_{l1}, z_{l2}, ..., z_{ln}]^T$ and

$$y_l = \mathbf{w}_l^T \mathbf{z}_l \tag{4.3.16}$$

$$\mathbf{w}_l(k+1) = \mathbf{w}_l(k) + \mu_l(k)\varphi(y_l)\mathbf{z}_l(k) \tag{4.3.17}$$

$$\varphi(y_l) = sgn(\kappa_4(y_l)) \left(y_l - \frac{m_2(y_l)}{m_4(y_l)} y_l^3 \right) \frac{m_4(y_l)}{m_2^3(y_l)} \tag{4.3.18}$$

where sgn denotes the sign function, m_2 and m_4 denote the second and fourth order moments. Minimizing cost function in (4.3.15) yields a learning rule

$$\tilde{\mathbf{w}}_l(k+1) = \tilde{\mathbf{w}}_l(k) + \tilde{\mu}_l(k) y_l(k) \mathbf{u}[\mathbf{z}_{l+1}(k)] \qquad l = 1, 2, ... \tag{4.3.19}$$

where $\tilde{\mu}_l$ is the learning rate, and

$$\mathbf{u}[\mathbf{z}_{l+1}(k)] = \frac{\partial \mathcal{L}_l}{\partial \mathbf{z}_{l+1}} = \begin{bmatrix} u(z_{l+1,1}) \\ u(z_{l+1,2}) \\ \vdots \\ u(z_{l+1,n}) \end{bmatrix} = \begin{bmatrix} sgn(z_{l+1,1})|z_{l+1,1}|^{p-1} \\ sgn(z_{l+1,2})|z_{l+1,2}|^{p-1} \\ \vdots \\ sgn(z_{l+1,n})|z_{l+1,n}|^{p-1} \end{bmatrix} \tag{4.3.20}$$

When the subtraction is completed, extraction scheme 4.3.2 is carried out on $\mathbf{z}_{l+1}$ to get y_{l+1}. The whole procedure can be continued with $y_{l+2}, y_{l+3}, ...$ until all the desired signals are estimated. That is, the amplitude of each signal $\mathbf{z}_{l+1}$ is lower than certain threshold.

4.3.2 kPNL Complete Algorithm

With the learning rules (4.3.9), (4.3.10) and (4.3.13) for the MLP components; and (4.3.17) and (4.3.19) for the linear extraction modules, we are able to write up a complete kurtosis-based PNL ICA algorithm. The following pseudo-code will describe the proposed kPNL algorithm

==========

kPNL: the Kurtosis-Based PNL Algorithm

Input

$x_1, x_2, ..., x_n$: n observed signals with N samples each.

Parameter

μ_i, $\tilde{\mu}_i$: learning rates.

ξ: threshold to stop the linearization.

Output

$y_1, y_2, ...$: estimated signals.

function kPNL()
{
  For $k = 1$ **to** N {
    Calculate $y_1(k)$ by (4.3.2);
    Compute $k_4(y_1)$, $m_2(y_1)$, $m_4(y_1)$ by (4.3.4) and (4.3.8);
    Compute $\varphi(y_1)$ by (4.3.7);
    Update $\mathbf{w}_1(k+1)$ ***by (4.3.6)***;
    Calculate $\mathbf{v}^m$ by (4.3.13);
    For $j = m-1$ **downto** 1
      Calculate $\mathbf{v}^j$ by (4.3.11);
    For $i = m-1$ **downto** 1 {
      Update $\mathbf{V}^i(k+1)$ ***by (4.3.9)***;
      Update $\mathbf{b}^i(k+1)$ ***by (4.3.10)***;
    } /* end of i loop */
    Let $l = 0$;
    Repeat {
      $l = l + 1$;
      Calculate $\mathbf{z}_{l+1}(k)$ by (4.3.14);
      Calculate $\mathbf{u}[\mathbf{z}_{l+1}(k)]$ by (4.3.20);
      Update $\tilde{\mathbf{w}}_l(k+1)$ ***by (4.3.19)***;
    } **Until** $(\|\mathbf{z}_{l+1}\| < \xi)$
  } /* end of k loop */
} /* *end of kPNL* */

==========

4.3.3 kPNL Performance Analysis

In this section, we review two experiments that were carried out to verify the extraction ability of kPNL. The first simulation was run on three gray-scaled images

with mild post nonlinear mixing. The second simulation was carried out with hard nonlinear mixing. The model configuration was kept fixed in all simulations so that we can verify the adaptation of kPNL on different data sets and on different mixing situations.

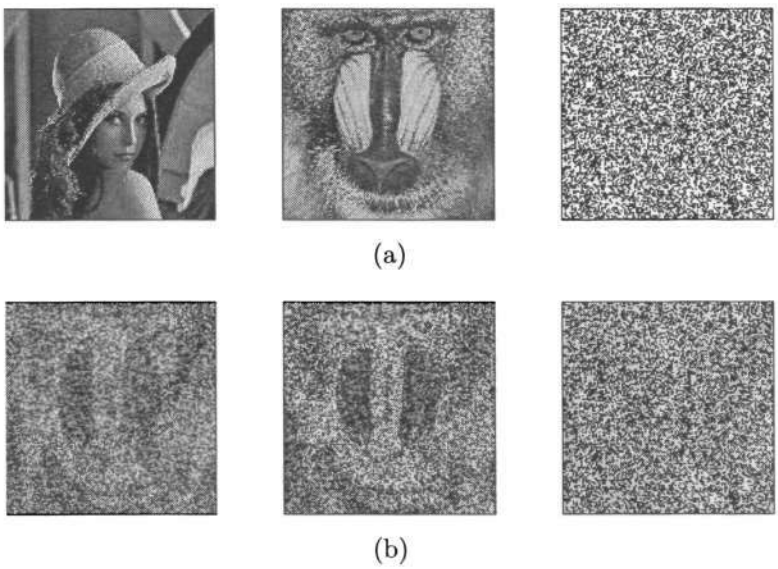
Simulation 1: Mixture of Three Images

To create the PNL mixtures, three images (shown in Fig. 4.7(a)) including two real pictures (Lenna and Baboon) and a noise picture, all of size 128×128, are first linearly mixed by a random mixing matrix $\mathbf{A}$, and then passed through nonlinear functions $f_i(.)$. In this specific simulation, mixing matrix and nonlinear functions are as follows

$$\mathbf{A} = \begin{bmatrix} 0.177 & -0.373 & -0.402 \\ 0.084 & -0.537 & 0.345 \\ 0.307 & -0.167 & 0.876 \end{bmatrix} \tag{4.3.21}$$

$$\begin{aligned} f_1(x) &= x + 0.1x^3 \\ f_2(x) &= \tfrac{1}{2.5+x} \\ f_3(x) &= 0.8x - 0.1650x^3 + 0.53x^5 \end{aligned} \tag{4.3.22}$$

The original images and their PNL mixtures are shown in Fig. 4.7. All images are of size 128×128.



(a) (b)

Figure 4.7: The three source images (Lena, Baboon and noise) and their nonlinear mixtures, all of size 128×128. (a) Unknown sources, and (b) PNL mixtures.

Two-layer MLP with identical architecture $1-20-1$ was chosen as the nonlinear components (one input, one output and 20 neurons in hidden layer). Tangent hyperbolic function was chosen as activation function, f^i. The values of demixing vectors $\mathbf{w}_i$ and deflation vectors $\widetilde{\mathbf{w}}_i$ $(i = 1, 2, 3)$, and all parameters of MLP modules were initialized randomly. From the three observed images, data is converted into one dimensional signals and kPNL algorithm is applied on it. The outputs are then converted back to images. As shown in Fig. 4.8, the original pictures have been successfully estimated. Comparing with the source images in Fig. 4.7(a), the estimates are similar to the original ones.

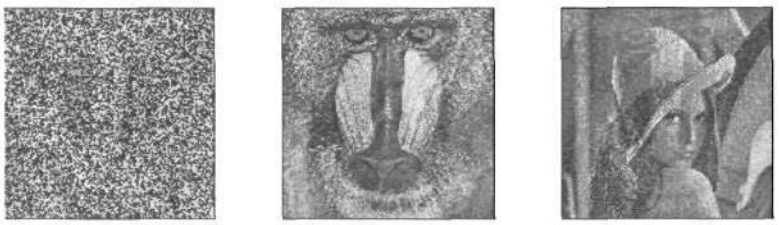


Figure 4.8: The three images that are estimated by kPNL.

Table 4.1: Simulation 1: Mixture of three images - Absolute correlation coefficient between the original sources and the outputs.

$\lvert r_{s,y}\rvert$	s_1	s_2	s_3
y_1	0.0318	0.1989	**0.9776**
y_2	0.2388	**0.9426**	0.1576
y_3	**0.9086**	0.3490	0.0377

Quantitative performance is provided in Table 4.1 in term of the absolute correlation coefficient. The absolute correlation coefficient between s and y, denoted by $|r_{s,y}|$, is computed by

$$|r_{s,y}| = \left| \frac{\sum_{t=1}^{N}(s(t)-\overline{s})(y(t)-\overline{y})}{\sqrt{\sum_{t=1}^{N}(s(t)-\overline{s})^2 \sum_{t=1}^{N}(y(t)-\overline{y})^2}} \right| \tag{4.3.23}$$

where $\overline{s} = \frac{1}{N}\sum_{t=1}^{N} s(t)$, $\overline{y} = \frac{1}{N}\sum_{t=1}^{N} y(t)$, with N is the number of the samples. The higher $|r_{s,y}|$, the more correlation between s and y. When two signals are identical, their absolute correlation coefficient $|r_{s,y}| = 1$. Whereas, the correlation coefficient $|r_{s,y}| \approx 0$ when two signals are not correlated to each others. As it is clearly shown in

the table, kPNL provides a very good result with correlation coefficients between the source signal and its corresponding estimate (the bold numbers in the table) are all > 0.9.

Simulation 2: Mixture of Four Sinusoidal Signals

In this simulation, we test kPNL algorithm with hard nonlinear distortions. Using four sinusoidal artificial signals, we create linear mixtures and then put them through four hard nonlinear functions to generate the PNL mixtures. The source signals, linear mixing matrix and nonlinear functions are chosen as

$$s_i(t) = sin(0.02\pi(2i-1)t) \qquad (i = 1, 2, 3, 4) \tag{4.3.24}$$

$$\mathbf{A} = \begin{bmatrix} 0.787 & -0.198 & -0.066 & -0.081 \\ -0.201 & 0.748 & -0.203 & -0.296 \\ -0.262 & -0.100 & 0.987 & -0.117 \\ -0.087 & -0.108 & -0.132 & 0.822 \end{bmatrix} \tag{4.3.25}$$

$$\begin{aligned} x_1(t) &= (1.8v_1(t))^3 \\ x_2(t) &= 4tanh(5v_2(t)) \\ x_3(t) &= arctan(4v_3(t)) \\ x_4(t) &= 4logsig(7v_4(t)) - 2 \end{aligned} \tag{4.3.26}$$

The illustrations of source signals and their PNL mixtures are shown in Fig. 4.7(a) and Fig. 4.7(b). The $1 - 20 - 1$ MLP networks (as in previous simulations) were used in order to cancel the nonlinearities of the observed signals. 1000 samples of the PNL mixtures were chosen as inputs for the kPNL algorithm. After executing the algorithm, four outputs were estimated and are shown in Fig. 4.9(c).

Unfortunately, kPNL fails to estimate the original signals. The kurtosis-based learning/updating rule for MLP parameters and the two demixing and deflation matrices does not work well in this experiment. MLP components can not eliminate the nonlinearity of the inputs, leading to the wrong estimations in the following steps. It is suggested that we need a more effective method for canceling the nonlinearity in addition to the current updating rule.

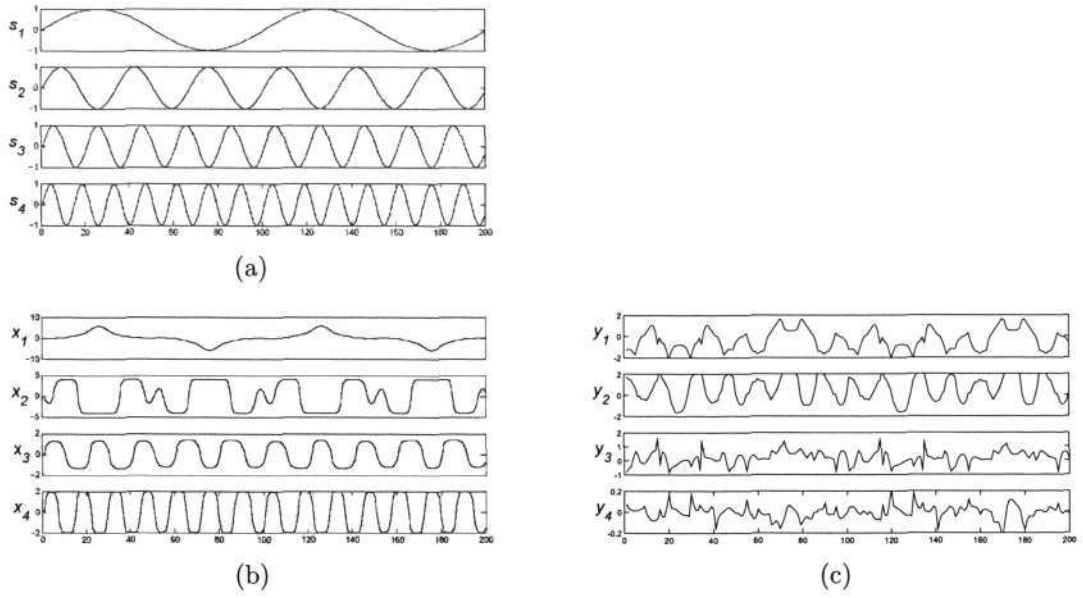


Figure 4.9: Time series plots of the signals (200 samples). (a) Unknown original signals, (b) PNL mixtures and (c) kPNL estimates.

4.3.4 kPNL Summary

We have introduced a nonlinear ICA approach by modifying and applying a linear ICA sequential extraction algorithm to the Post Nonlinear (PNL) model. Taking the advantage of the PNL model, our kPNL ensures a unique solution for the problem. With the use of sequential extraction algorithm, no prior knowledge of the number of sources is needed and the unknown sources can be estimated conveniently one by one. In addition, the simplicity of the Kurtosis-based updating rule allows kPNL to reduce the computational cost and to avoid approximating the probability density functions. In short, kPNL is a fast and online-learning algorithm.

The simplicity of the updating rule, however, also brings disadvantage to kPNL. As we have experienced in the simulation (Section 4.3.3), training all the kPNL model with the Kurtosis-based learning only is not enough to eliminate the nonlinear and estimate the correct original signals. Hence, some alternative methods are required to improve the kPNL performance.

4.4 gPNL: Geometric Post Nonlinear ICA

The kPNL algorithm in previous section introduced a convenient way to extract the source signals sequentially from their nonlinear mixtures. It does, however, have the

disadvantages. There has been argument that kPNL does not work well on hard nonlinear distorted signals and the Kurtosis criterium is sensitive to the outlier. In this section, we propose a novel approach to the PNL problem, the geometry-based approach.

Our algorithm, called gPNL (geometric Post Nonlinear), divides the separating system into two stages: the first stage applies geometric transformations to linearize the input signals, *i.e.* to transform the PNL observations into linear mixtures; and the second stage carries out blind separation on this linearized signals by applying a normal linear ICA algorithm.

Our major contribution [150, 154, 155], the geometric linearizing technique, comes from the following observation: in a multi-dimensional space, the graph representing nonlinear mixture is a hypersurface. In the special case, when the mixture is a linear one, its graph becomes a hyperplane. Hence, the objective of converting a given nonlinear mixture to a linear one can be achieved by transforming the corresponding hypersurface (the geometric representation of the nonlinear mixture) to a hyperplane (the geometric representation of the linear mixture). Thereafter, any algorithm which is applicable to linear ICA can be applied on the linearized mixtures to extract the original signals. With this two-stage geometric approach, our algorithm, gPNL, has the following advantages:

- Linearization can be done without any knowledge or assumption on the number or distribution of the unknown sources, the linear mixing system or the observed signals.

- After completion of the linearizing stage, one can choose the most suitable linear ICA method to accomplish the separating process. It is very useful as each ICA method works well only on specific type of applications.

- The linearization is carried out geometrically, therefore, gPNL does not have any constraint on the nonlinear distortion.

4.4.1 Geometric Approach for PNL Model

In this section, we will redescribe the PNL problem by geometric concepts. The nonlinear and linear mixtures are represented in terms of hypersurfaces and hyperplanes (also referred to as surfaces and planes for short) in a multi-dimensional space. The linearizing process that changes PNL mixtures into linear ones, is presented as a transformation of nonlinear surfaces into planes. The approach is first described in a particular case of two sources and two observed signals (the $2-2$ case) so that it can be illustrated in a three dimensional (3D) space. Solution for a general PNL with more than two sources and observations is then generalized from this particular case. To begin with, we recall the two basic definitions that will be used extensively in our method: the definitions of a surface and a plane.

Definition 4.2 Given $z = f(x, y)$, a graph of $f(x, y)$ is a set of all points $(x, y, f(x, y))$ in a 3D space spanned by x, y and z, and is called a surface. The function $f(x, y)$ can be any arbitrary function of x and y.

Definition 4.3 In a special case, when the function $f(x, y)$ is a linear one, *i.e.*, $f(x, y) = ax + by + c$, the graph of $f(x, y)$ in the 3D space is called a plane.

Let us look at a specific $2-2$ case, the linear mixtures in (4.2.2), v_i, can be expressed as

$$\begin{aligned} v_1 &= a_{11}s_1 + a_{12}s_2 \\ v_2 &= a_{21}s_1 + a_{22}s_2 \end{aligned} \tag{4.4.1}$$

and the mixtures in (4.2.3), x_i, $i = 1, 2$, can be expressed as

$$\begin{aligned} x_1 = f_1(v_1) &= f_1(a_{11}s_1 + a_{12}s_2) \\ x_2 = f_2(v_2) &= f_2(a_{21}s_1 + a_{22}s_2) \end{aligned} \tag{4.4.2}$$

Considering a 3D space spanned by s_1, s_2 and z where z can be assumed to be either v_i or x_i. Let us denote the graphs of v_i and x_i, $(i = 1, 2)$ by $\mathcal{S}_{v_i}$ and $\mathcal{S}_{x_i}$, respectively. Since v_i is described as a linear equation (4.4.1), its graph $\mathcal{S}_{v_i}$ in this 3D space is clearly a plane. The PNL mixture x_i on the other hand, is represented by a nonlinear function (4.4.2) and its graph $\mathcal{S}_{x_i}$ is in the form of a nonlinear surface. The PNL model is then described in term of the transformation from planes to surfaces and illustrated geometrically in Fig. 4.10.

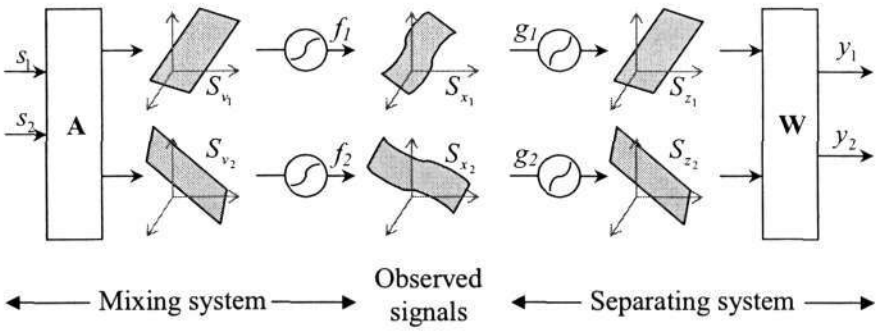
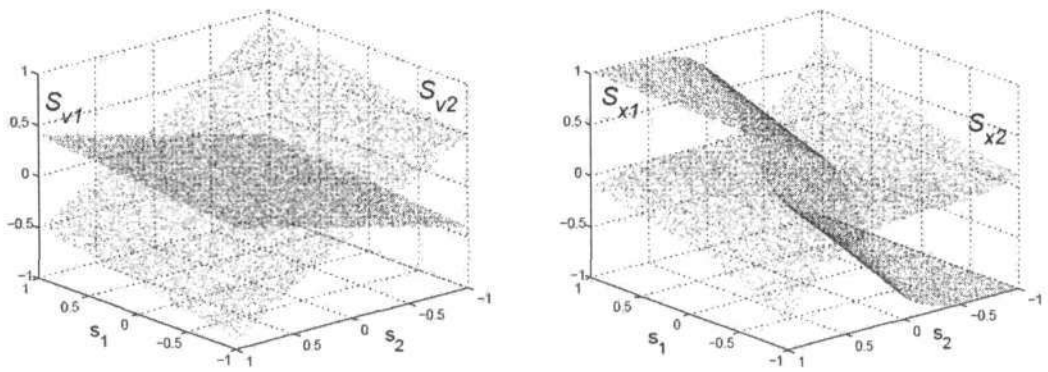


Figure 4.10: Geometric description of a PNL mixing and separating system in a 3D space. Linear mixtures are represented by planes, v_i and z_i. Nonlinear mixtures are represented by surfaces, x_i, $i = 1, 2$.

An example of the PNL mixing system is depicted in Fig. 4.11. From 10000 samples in range $[-1, 1]$ of random signals s_1 and s_2, graphs of two linear mixtures $v_1 = 0.1s_1 + 0.3s_2$ and $v_2 = 0.2s_1 - 0.7s_2$ were created and are shown in Fig. 4.11(a). Next, the PNL mixtures $x_1 = tanh(10v_1)$ and $x_2 = v_2^3$ were constructed and their graphs, $\mathcal{S}_{x_1}$ and $\mathcal{S}_{x_2}$, are shown in Fig. 4.11(b). These two observed graphs $\mathcal{S}_{x_1}$ and $\mathcal{S}_{x_2}$ will be the inputs for the PNL separating system.

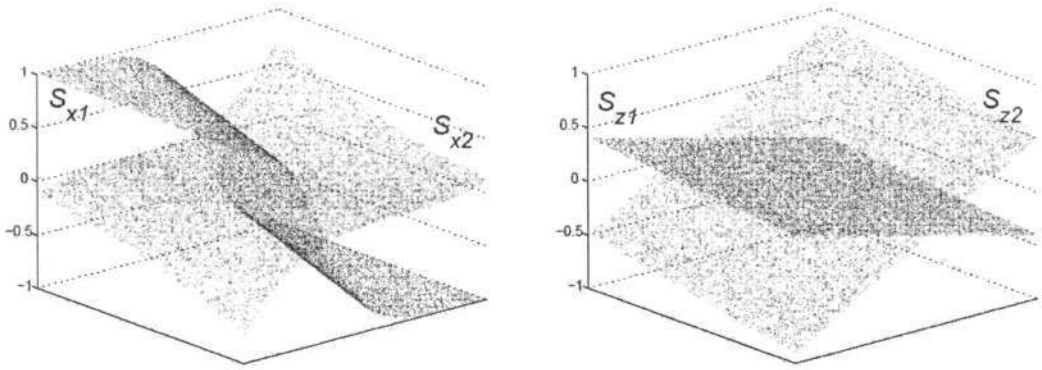


(a) (b)

Figure 4.11: Example of a PNL mixing system in 3D space. (a) Planes $\mathcal{S}_{v_1}$ and $\mathcal{S}_{v_2}$ correspond to linear mixtures v_1 and v_2. (b) Nonlinear surfaces $\mathcal{S}_{x_1}$ and $\mathcal{S}_{x_2}$ correspond to PNL mixtures x_1 and x_2.

In the separation stage, the only available data are the nonlinear mixed signals. That is, in the graph of the observed signals $\mathcal{S}_{x_i}$, the values of the first two coordinates are unknown and only the value of the third coordinate x_i can be accessed. Denote by '_' the unknown coordinate, then points on surface $\mathcal{S}_{x_i}$ can be represented by $(\_, \_, x_i)$, $i = 1, 2$. Illustration of these input surfaces are shown in Fig. 4.12(a) where the information on the first two axes are hidden, and our objective is to retrieve the

values of these two unknown coordinates.



(a) (b)

Figure 4.12: Example of the PNL separating system in 3D space. (a) Nonlinear surface $\mathcal{S}_{x_1}$ and $\mathcal{S}_{x_2}$ correspond to the observations. (b) Planes $\mathcal{S}_{z_1}$ and $\mathcal{S}_{z_2}$ correspond to linearized mixtures z_1 and z_2.

To achieve the goal, a unique geometric transformation process is applied in the first stage of the separating system. This critical linearization stage is accomplished by transforming the nonlinear input surfaces $\mathcal{S}_{x_i}$ into planes $\mathcal{S}_{z_i}$ using geometric modifications. An example of the linearized signal z_i, represented as plane $\mathcal{S}_{z_i}$ in 3D space is shown in Fig. 4.12(b). Note that the labels s_1 and s_2 appeared in Fig. 4.11 has been removed from Fig. 4.12 in order to reflect the unavailability of the knowledge of source signals, *i.e.*, the first two coordinates.

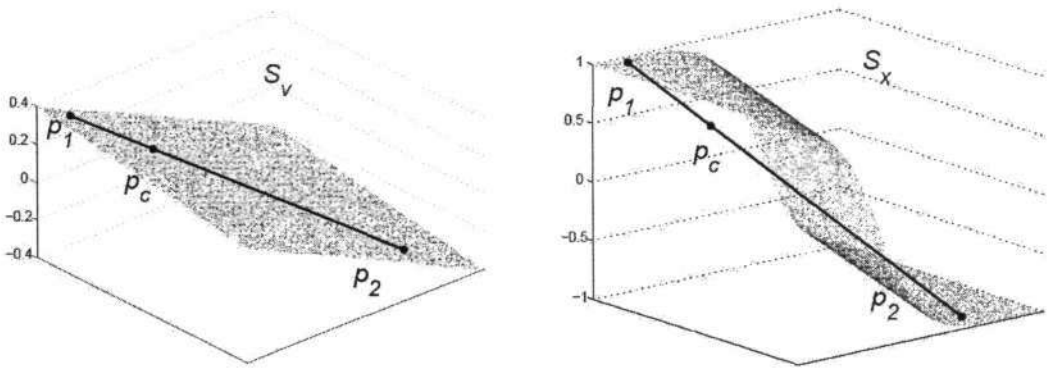
The problem of transforming surface to plane would be quite easy if the coordinates s_1 and s_2 were known. The construction of $\mathcal{S}_{z_i}$ can be achieved by letting $z_i = w_{i1}s_1 + w_{i2}s_2$ and selecting any arbitrary nonzero scalar value for w_{i1} and w_{i2} $(i = 1, 2)$. In PNL problem described above, however, this kind of information is not available. Thus, to carry out the transformation $\mathcal{S}_{x_i}$ into $\mathcal{S}_{z_i}$, the following issues need to be resolved

1. To identify whether a given graph $\mathcal{S}_z$ is a plane.
2. To evolve a mechanism to generate $\mathcal{S}_{z_i}$ from $\mathcal{S}_{x_i}$ without prior knowledge of the coordinates s_1 and s_2 .

Identification of a Plane

From the nature of plane and surface in 3D space, the following property is observed

Proposition 4.2 Let p_1 and p_2 be two arbitrary points lying on a surface $\mathcal{S}$ in a 3D space. Let $\overline{p_1p_2}$ be a straight line joining these two points and let p_c be an arbitrary point lying on the straight line $\overline{p_1p_2}$. $\mathcal{S}$ is a plane if and only if for all $p_c \in \overline{p_1p_2}$ and for all p_1 and p_2 on $\mathcal{S}$, p_c lies on $\mathcal{S}$.



(a) (b)

Figure 4.13: An example showing the difference between a plane and a nonlinear surface. (a) Arbitrary point $p_c \in \overline{p_1p_2}$ lies on the plane $\mathcal{S}_v$. (b) Point $p_c \in \overline{p_1p_2}$ falls out of nonlinear surface $\mathcal{S}_x$.

An illustration of Proposition 4.2 is shown in Fig. 4.13. In Fig. 4.13(a), two points p_1 and p_2 lie on $\mathcal{S}_v$ and p_c is an arbitrary point on $\overline{p_1p_2}$. Since $\mathcal{S}_v$ is a plane ($v = 0.1s_1 + 0.3s_2$ in the example), we have $p_c \in \mathcal{S}_v$. Whereas, in Fig. 4.13(b), since $\mathcal{S}_x$ is not a plane ($z = tanh(10v) = tanh(10(0.1s_1 + 0.3s_2))$ in the example), p_c does not lie on $\mathcal{S}_x$. With the above proposition, we can identify which one is a plane.

Transformation of a Surface to a Plane

In order to retrieve a plane $\mathcal{S}_z$ from a given nonlinear surface $\mathcal{S}_x$ without information of the first two coordinates, we propose a heuristic constructive method that utilizes properties of a line in 3D space. First, we introduce the following definitions and propositions which will be used latter in our algorithm.

Definition 4.4 The equation of a line containing a point $p_1(x_1, y_1, z_1)$ is written in the form of $\frac{x-x_1}{a} = \frac{y-y_1}{b} = \frac{z-z_1}{c}$ where a, b, c are non-zero scalars.

Definition 4.5 Given $p_1(x_1, y_1, z_1)$ and $p_2(x_2, y_2, z_2)$ any two arbitrary points in a 3D space. p_1 is a companion of p_2 (and vice versa) if and only if $x_1 = x_2$ and $y_1 = y_2$.

Using Definitions 4.4 and 4.5, the following property is observed

Proposition 4.3 Given two arbitrary points $p_1(\_, \_, z_{p1})$ and $p_2(\_, \_, z_{p2})$. Let us denote theirs companions by $q_1(\_, \_, z_{q1})$ and $q_2(\_, \_, z_{q2})$, respectively. Let $p_c(\_, \_, z_{pc})$ and $q_c(\_, \_, z_{qc})$ be the two arbitrary points lying on $\overline{p_1p_2}$ and $\overline{q_1q_2}$, respectively. If p_c is the companion of q_c then the following expression holds true

$$\frac{z_{pc} - z_{p1}}{z_{p2} - z_{p1}} = \frac{z_{qc} - z_{q1}}{z_{q2} - z_{q1}} \tag{4.4.3}$$

Proof of Proposition 4.3. Since p_1 is the companion of q_1, their respective co-ordinates are specified by (x_1, y_1, z_{p1}) and (x_1, y_1, z_{q1}). Similarly, coordinates of p_2, q_2, p_c and q_c are specified by (x_2, y_2, z_{p2}), (x_2, y_2, z_{q2}), (x_c, y_c, z_{pc}) and (x_c, y_c, z_{qc}), respectively.

Referring to Definition 4.4, as p_c lies on $\overline{p_1p_2}$, the following equation is derived

$$\frac{x_c - x_1}{x_2 - x_1} = \frac{y_c - y_1}{y_2 - y_1} = \frac{z_{pc} - z_{p1}}{z_{p2} - z_{p1}} \tag{4.4.4}$$

Similarly, as q_c lies on $\overline{q_1q_2}$, we have

$$\frac{x_c - x_1}{x_2 - x_1} = \frac{y_c - y_1}{y_2 - y_1} = \frac{z_{qc} - z_{q1}}{z_{q2} - z_{q1}} \tag{4.4.5}$$

Combining (4.4.4) and (4.4.5) yields

$$\frac{z_{pc} - z_{p1}}{z_{p2} - z_{p1}} = \frac{z_{qc} - z_{q1}}{z_{q2} - z_{q1}} \tag{4.4.6}$$

□

Proposition 4.3 presents the key point of our method. Given five points p_1, p_2, q_1, q_2 and p_c, we can locate the position of the sixth point, q_c, by using (4.4.3). Assume that a plane $\mathcal{S}_v$ is given, and our mission is to transform a nonlinear surface $\mathcal{S}_x$ into a plane $\mathcal{S}_z$ with the help of $\mathcal{S}_v$. The following transformation scheme will detail the steps to transform $\mathcal{S}_x$ into $\mathcal{S}_z$ using $\mathcal{S}_v$. The plane $\mathcal{S}_v$, for this reason, is referred to as 'reference plane'.

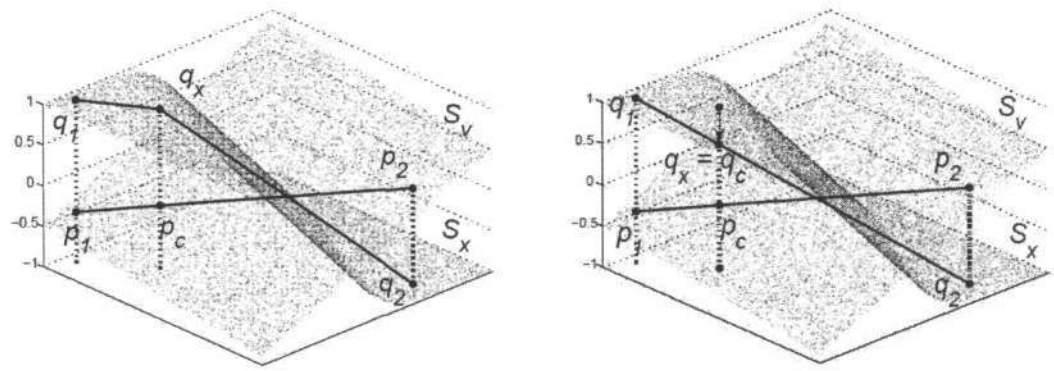
==========

Transformation Scheme

- Pick up any two arbitrary points p_1 and p_2 on the reference plane, $\mathcal{S}_v$.

- Locate their respective companions q_1 and q_2 on $\mathcal{S}_x$.
- Select an arbitrary point p_c on $\overline{p_1 p_2}$.
- Find a companion, $q_c \in \overline{q_1 q_2}$, of p_c by using equation (4.4.3).
- Locate a point q_x on $\mathcal{S}_x$ such that $\{p_c, q_x\}$ is a companion pair.
- Pull q_x towards q_c.
- Use Proposition 4.2 to check whether $\mathcal{S}_x$ is a plane.
- If $\mathcal{S}_x$ is not a plane then repeat all the above steps, otherwise stop.

==========



(a) (b)

Figure 4.14: An example showing one iteration of the linearizing process using reference plane. (a) The point $q_x \in \mathcal{S}_x$ before transformation. (b) After the transformation, q_x was pulled to $q_c \in \overline{q_1 q_2}$.

Figure 4.14 illustrates the movement of q_x in one iteration of the linearizing process. The old location of q_x is shown in Fig. 4.14(a). After one single transformation, q_x moves towards q_c (Fig. 4.14(b)) which lies on the same straight line joining q_1 and q_2. The iteration is repeated for every point $q_x \in \mathcal{S}_x$ and in the end, the surface $\mathcal{S}_x$ is transformed to a plane $\mathcal{S}_z$.

4.4.2 gPNL Algorithm

In this section, we apply the above transformation scheme to the construction of our geometric linearizing method. However, the following issues have to be solved:

1. Since the first two coordinates are not available (s_1 and s_2 are unknown), we cannot apply Definition 4.5 directly to identify the companion of a given point.

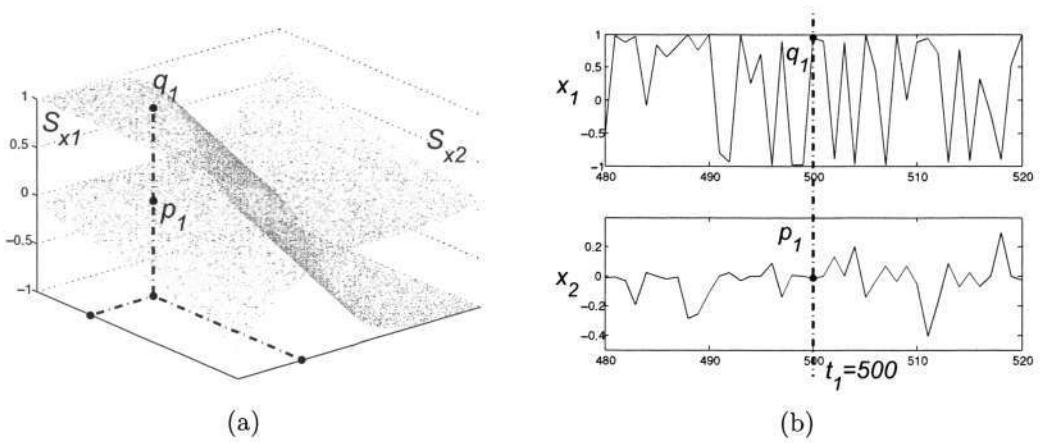
2. In PNL model, there is no plane to use as reference plane.

3. We have to deal with the ambiguity of point $q_c \in \overline{q_1 q_2}$. Proposition 4.3 implies that if q_c is a companion of p_c then (4.4.3) holds true, but the reverse does not apply always.

Identifying Companion by Time Index

To identify a companion of a given point, we take advantage of the time index mentioned in (4.2.1) and (4.2.4) with the proposition 4.4 described below. Let $v_1(t_1)$ be a sample of linear mixture v_1 at time $t = t_1$, then this sample corresponds to a point $(s_1(t_1), s_2(t_1), v_1(t_1))$ on the surface $\mathcal{S}_{v_1}$. Similarly, coordinates of the points $v_2(t_1)$, $x_1(t_1)$ and $x_2(t_1)$ are given by $(s_1(t_1), s_2(t_1), v_2(t_1))$, $(s_1(t_1), s_2(t_1), x_1(t_1))$ and $(s_1(t_1), s_2(t_1), x_2(t_1))$, respectively. Considering our model (4.2.1), relationship among these points are represented by the following proposition:

Proposition 4.4 Let $p_1(\_, \_, v_1(t_1)) \in \mathcal{S}_{v_1}$ and $q_1(\_, \_, x_1(t_2)) \in \mathcal{S}_{x_1}$ be the two points representing the samples $v_i(t_1)$ and $x_j(t_2)$, respectively. If $t_1 = t_2$ then p_1 is a companion of q_1.



(a) (b)

Figure 4.15: An example of companion points in relation with time index. (a) In a 3D space plot, p_1 and q_1 have the same coordinates $s_1(t_1)$ and $s_2(t_1)$. (b) In time series plot, p_1 and q_1 are at the same time instant $t = t_1$.

An example of the Proposition 4.4 is shown in Fig. 4.15. In a 3D space (Fig. 4.15(a)),

the p_1 and its companion q_1 are represented as two points having the same coordinates $s_1(t_1)$ and $s_2(t_1)$. Meanwhile, the companion points are shown in time series in Fig. 4.15(b) as two signal samples at same time instant $t = t_1$. Clearly with Proposition 4.4, companion points can be identified without knowing the coordinates s_1 and s_2.

Replacing The Reference Plane by A Fake Plane

For the second issue of finding a reference plane, the proposed solution is to employ a 'fake plane', *i.e.*, to assume a surface as reference plane and use it to transform the other surface. The two surfaces will alternatively play the role of 'fake plane' during the transformation process. For this reason, the process mentioned in Section 4.4.1 is modified to adapt with the fake plane. Let $\mathcal{S}_{x_2}$ be the fake plane and $\mathcal{S}_{x_1}$ be the surface to be transformed. Select two pairs of companions points $\{p_1(\_, \_, x_2(t_1)), q_1(\_, \_, x_1(t_1))\}$ and $\{p_2(\_, \_, x_2(t_2)), q_2(\_, \_, x_1(t_2))\}$ with $p_i \in \mathcal{S}_{x_2}$ and $q_i \in \mathcal{S}_{x_1}$, $i = 1, 2$. Let p_c be a point being located at $(\_, \_, x_2(t_c))$. Then the value on the third coordinate of the companion of p_c, $q_c(\_, \_, z_1(t_c))$, can be computed from (4.4.3) as

$$z_1(t_c) = \frac{x_2(t_c) - x_2(t_1)}{x_2(t_2) - x_2(t_1)}(x_1(t_2) - x_1(t_1)) + x_1(t_1) \tag{4.4.7}$$

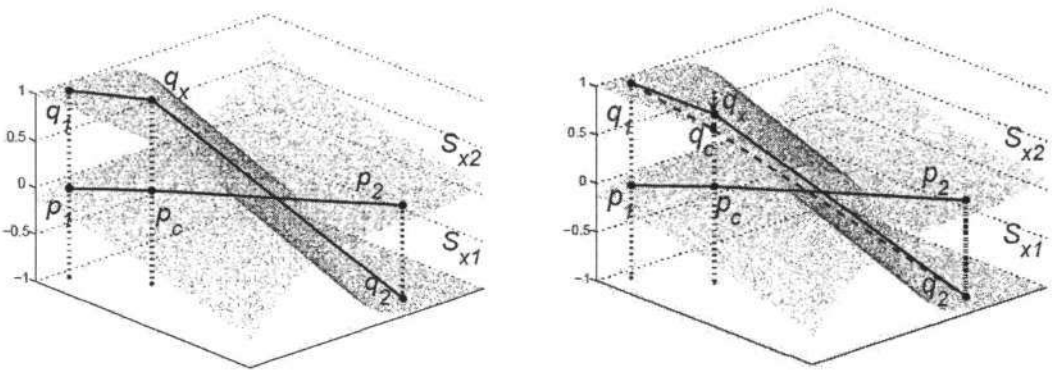
Reduction of Ambiguity by Local Transformation

In Proposition 4.3, we have stated that if q_c is a companion of p_c then (4.4.3) holds true. However, the reverse clause is not always true. There may have more than one point q_c that satisfy (4.4.3) but are not the companions of p_c. An incorrect selection of q_c will make q_x, the point that is supposed to be changed, move towards a wrong position. Unfortunately, there is no solution to eliminate this ambiguity completely. In order to reduce the inaccuracy, we apply a local transformation, *i.e.* the surface is divided into small regions and the transformation process is carried out within these regions. In addition, instead of moving q_x right to q_c, we apply a learning rate $\mu < 1$ to update q_x position. With this technique, the transformation take a longer time but steadily converge. The updating function is formulated as

$$x_1^{new}(t_c) = \mu z_1(t_c) + (1 - \mu)x_1^{old}(t_c) \tag{4.4.8}$$

Final gPNL Algorithm

With all the above solvents, the linearizing stage is finally addressed. Figure 4.16 illustrates an iteration of the proposed gPNL linearizing process. The surfaces $\mathcal{S}_{x_1}$ and $\mathcal{S}_{x_2}$ represent two nonlinear mixtures x_1 and x_2, respectively. $\mathcal{S}_{x_2}$ is assumed to be a 'fake plane' and is used to transform $\mathcal{S}_{x_1}$ into a plane. The old position of the selected point, q_x, before the change is illustrated in Fig. 4.16(a). The new location of q_x after the transformation is shown in Fig. 4.16(b). With the learning rate $\mu < 1$, q_x does not move right to q_c but to a location near q_c.



(a) (b)

Figure 4.16: An example of gPNL geometric linearizing method. Surface $\mathcal{S}_{x_1}$ is transformed by fake plane $\mathcal{S}_{x_2}$. (a) The old position of q_x. (b) The new position after transformation. Because the learning rate μ is smaller than 1, q_x is pulled to a new position near q_c.

After the linearizing process, we obtain two planes, $\mathcal{S}_{z_1}$ and $\mathcal{S}_{z_2}$, which represent for the linearized signals z_1 and z_2, respectively. To further improve the smoothness of these surfaces, a smoothing function is applied. In this work, we use an simple averaging technique. The signals are sorted in ascending order and then the sorted signals are smoothed by averaging. After that, it is restored back to the original order. The averaging function is formulated as

$$z_i^s(t) = \frac{1}{L} \sum_{j=-(L-1)/2}^{(L-1)/2} z_i^s(t+j) \qquad i = 1, 2 \tag{4.4.9}$$

where z_i^s denotes the sorted version of the signal z_i, $i = 1, 2$. The function replaces a value at time instant t by the average of the values inside a window of size L (L is an odd number) and centered at t.

Finally, the algorithm is extended for the case of $n > 2$ observations. In this case, each signal x_i is represented by a surface $\mathcal{S}_{x_i}$ $(i = 1, 2, ..., n)$. In one iteration, a surface $\mathcal{S}_{x_k}$ is randomly chosen and used as the 'fake plane' to transform the other surfaces. The equation (4.4.7), therefore, is updated to

$$z_i(t_c) = \frac{x_k(t_c) - x_k(t_1)}{x_k(t_2) - x_k(t_1)}(x_i(t_2) - x_i(t_1)) + x_i(t_1) \qquad i = 1, 2, ..., n \qquad (4.4.10)$$

Likewise, the updating equation (4.4.8) is modified as

$$x_i^{new}(t_c) = \mu z_i(t_c) + (1 - \mu)x_i^{old}(t_c) \qquad (4.4.11)$$

The stopping criteria '$\mathcal{S}_x$ is a plane' of the transformation scheme in Section 4.4.1 is also needed to be realized into a precise condition in order to complete gPNL algorithm. In this work, we measure the difference (error) between the values of the signal after and before the change in one iteration. In each iteration, these errors are accumulated and compared to a threshold ξ. The accumulated error, ϵ, is computed by

$$\epsilon = \frac{1}{nN_k}\sum_{i=1}^{n}\sum_{j=1}^{N_k}(x_i^{new}(j) - x_i^{old}(j))^2 \qquad (4.4.12)$$

where $x_i^{new}(j)$ and $x_i^{old}(j)$ are the signal values after and before the change, respectively, and N_k is the number of updated samples in each iteration. The linearizing process is stopped when ϵ goes below a threshold ξ. After that, a basic linear ICA algorithm is applied on the linearized signals, z_i $(i = 1, 2, ..., n)$, to produce the estimates of source signals, y_i. A framework of gPNL is described by the following pseudo-code

==========

gPNL: the Geometric PNL Algorithm

Input

$x_1, x_2, ..., x_n$: n observed signals each with N samples.

Parameters

μ: learning rate.

ξ: threshold to stop the linearization.

N_k: number of points to be updated in each iteration.

Outputs

$z_1, z_2, ..., z_n$: n linearized signals.

$y_1, y_2, ..., y_n$: n estimated signals.

function gPNL()
{
  Repeat {
    Randomly select a fake plane x_k.
    For $i = 1$ **to** n {
      For $j = 1$ **to** N_k {
        Randomly generate time indices t_1, t_2 and t_c.
        Compute $z_i(t_c)$ using (4.4.10).
        Update $x_i(t_c)$ using (4.4.11).
      } */* end of j loop */*
    } */* end of i loop */*
    Compute ϵ using (4.4.12).
  } **Until** ($\epsilon < \xi$)
  For $i = 1$ **to** n assign $z_i = x_i$.
  For $i = 1$ **to** n smooth z_i using (4.4.9).
  Apply a linear ICA algorithm on z_i to extract y_i.
} */* end of gPNL */*
==========

4.4.3 gPNL Performance Analysis

Computer simulations have been carried out on different data sets to evaluate the performance of gPNL method. The first test was done on a simple case of two sinusoidal sources so that the data can be visualized. The number of source signals is increased to four in the second and third simulations. In the last simulation, we repeat the test which was used in the kPNL analysis section and then compare the performance of the two algorithms. The simulations were carried out with the following steps

1. Load the source signals, generate a mixing matrix **A** randomly, and set the nonlinear functions f_i.

2. Generate linear mixtures v_i and PNL mixtures x_i.

3. Select parameters for the gPNL algorithm.

4. Run the linearizing process to get the signals z_i.

5. Apply a linear ICA method on z_i to estimate y_i.

6. Run competitive algorithms on the same inputs and compare their results with those of our method.

We have applied several typical linear ICA methods for the linear separating stage (step 5), such as SOBI (Second Order Blind Identification) [28], JADETD (Joint Approximate Diagonalization of Eigen matrices with time delays) [73], and FPICA (Fixed- Point ICA) [86]. As results of these methods were almost identical, we only provide the outcomes that were carried out with SOBI during the simulations.

To give a perspective comparison of the algorithm's performance, we compare our method with a linear ICA algorithm and a competitive PNL method. For the linear one, we choose SOBI, and for the PNL one, we choose Gauss-TD [237], one of the effective reported PNL methods. The performance was measured in term of the absolute correlation coefficient (4.3.23) between the original source and its estimate, $|r_{s,\hat{s}}|$.

Simulation 1: Mixture of Two Sinusoidal Signals

Using the sinusoidal signals (4.4.13), two linear mixtures v_1 and v_2 were generated by a mixing matrix $\mathbf{A}$ whose entries are the random numbers in range of $[-1, 1]$. The linear mixtures were then distorted by two nonlinear functions (4.4.15) to produce the observations x_1 and x_2. The source signals, mixing matrix and nonlinear functions are given below

$$\begin{aligned} s_1(t) &= sin(0.22t) \\ s_2(t) &= sin(2\pi(0.1t) + 6cos(2\pi(0.02t))) \end{aligned} \tag{4.4.13}$$

$$\begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix} = \begin{bmatrix} -0.605 & 0.152 \\ 0.625 & 0.056 \end{bmatrix} \begin{bmatrix} s_1(t) \\ s_2(t) \end{bmatrix} \tag{4.4.14}$$

$$\begin{aligned} x_1(t) &= (3v_1(t))^3 \\ x_2(t) &= tanh(10v_2(t)) \end{aligned} \tag{4.4.15}$$

At first, the linearizing process was run on 3000 samples ($N = 3000$) with the window size of smoothing function $L = 151$, learning rate $\mu = 0.2$ and error threshold $\xi = 0.002$. The 3D plots of the linearized signals, z_i, are shown in Fig. 4.17(c) and compared with the plots of linear mixtures, v_i, (Fig 4.17(a)) and PNL mixtures, x_i, (Fig. 4.17(b)). As it is expected, the graphs obtained by gPNL, z_i, are very similar

to the graphs of the unknown linear mixtures, v_i. Clearly, nonlinearity in x_i has been eliminated in z_i. Whereas, some nonlinearity is still visible in the graphs (Fig. 4.17(d)) obtained the competitive Gauss-TD algorithm.

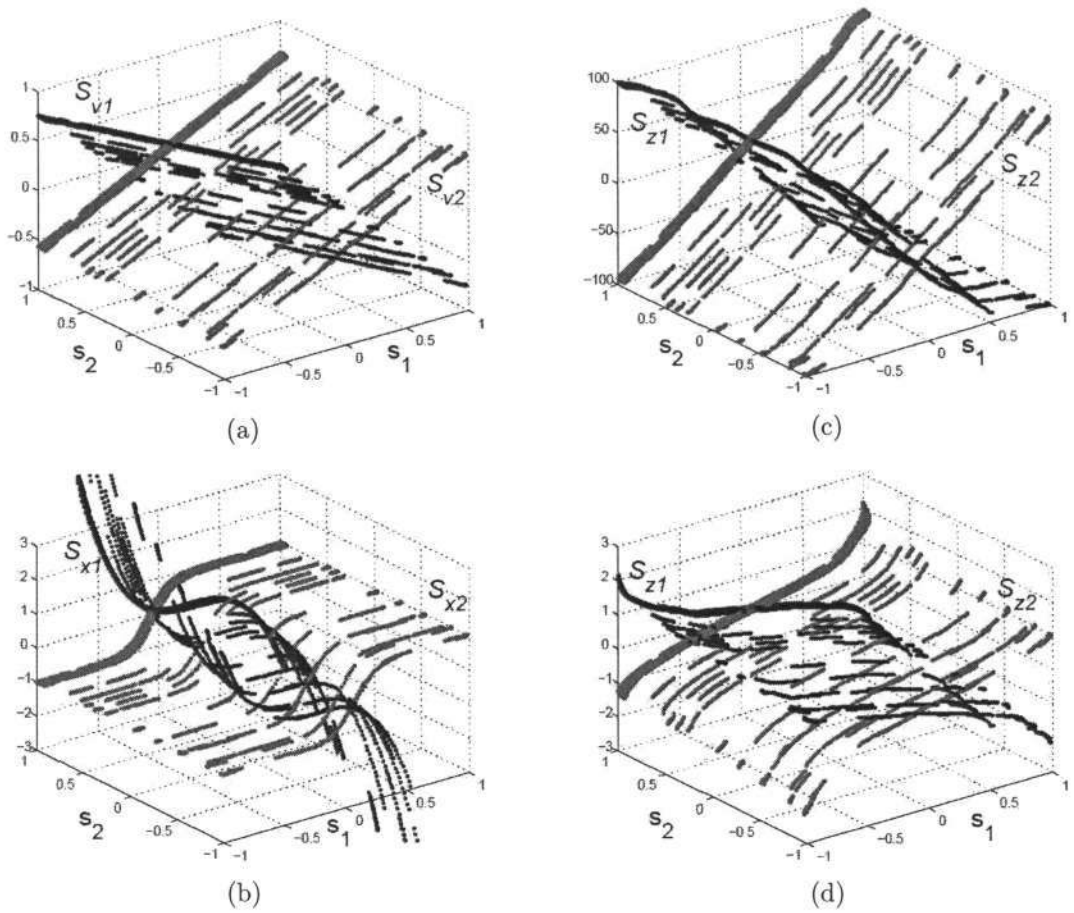


Figure 4.17: Geometric representation of signals in a 3D space. (a) Linear mixtures v_1 and v_2, (b) PNL mixtures x_1 and x_2, (c) gPNL linearized signals z_1 and z_2 and (d) Gauss-TD linearized signals.

In the next stage, a linear ICA method, SOBI, was applied on z_i to extract the estimates of original signals, y_i. The plots of gPNL estimates, y_i, are illustrated in Fig. 4.18, together with the plots of source signals, linear mixtures, PNL mixtures, SOBI direct estimates and Gauss-TD estimates. Comparing to those of SOBI and Gauss-TD, gPNL results resemble better to the original source signals. Among three competitors, linear ICA algorithm provided the worst performance (Fig. 4.18(d)).

A quantitative performance is carried out in term of the absolute correlation coefficient between a source signal and its estimate $\hat{s}_i$, $|r_{s_i,\hat{s}_i}|$. The estimate of the i-th source signal, $\hat{s}_i$, is one of the outputs y_j $(j = 1, 2, ..., n)$ whose absolute correlation coefficient, $|r_{y_j,s_i}|$, is the highest one. Using this index, $|r_{s_i,\hat{s}_i}|$, comparison between

gPNL and SOBI and Gauss-TD results are carried out and reported in Table 4.2. As it is shown in the table, the result obtained by gPNL is better than both SOBI and Gauss-TD results, showing the highest correlation coefficients, $|r_{\hat{s}_i,s_i}| \approx 1$.

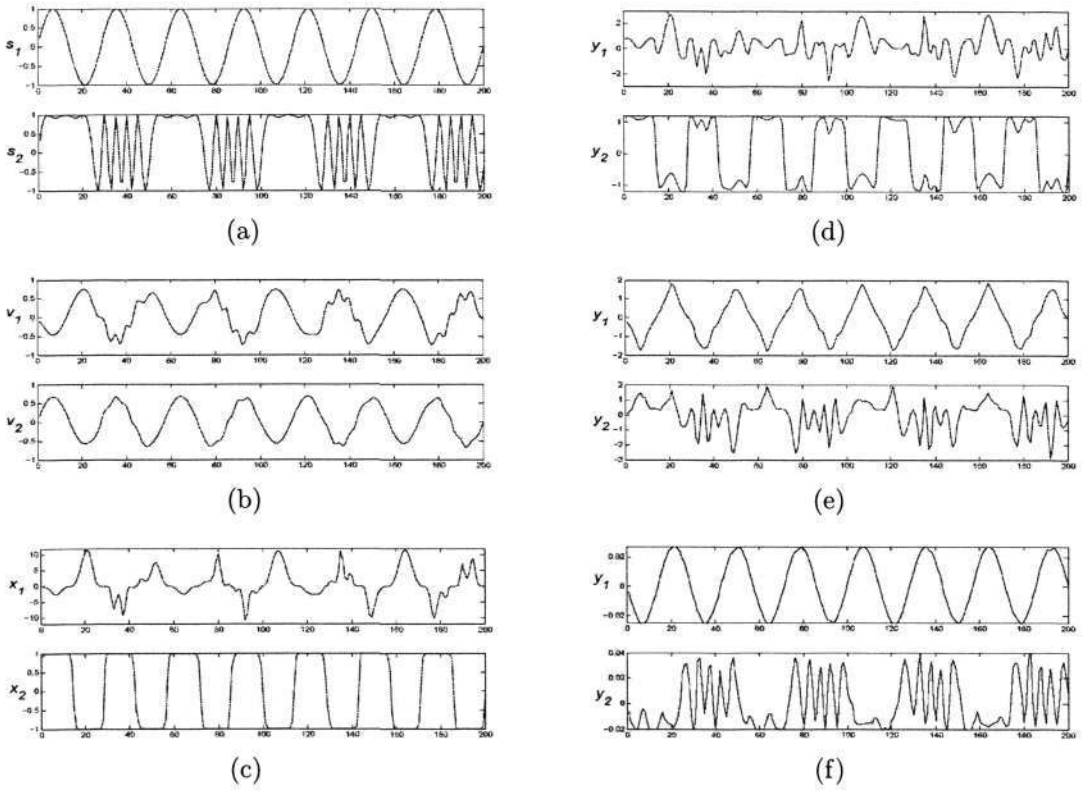


Figure 4.18: Time series plots of the signals (200 samples). (a) Unknown sources, (b) linear mixtures, (c) PNL mixtures, (d) SOBI estimates, (e) Gauss-TD estimates and (f) gPNL estimates.

Table 4.2: Simulation 1: Mixture of two sinusoidal signals - Absolute correlation coefficient between the original sources and their estimates, $|r_{s_i,\hat{s}_i}|$.

	$\lvert r_{s_1,\hat{s}_1}\rvert$	$\lvert r_{s_2,\hat{s}_2}\rvert$
SOBI	0.887	0.582
Gauss-TD	0.982	0.888
gPNL	0.999	0.966

Simulation 2: Mixture of Four Sinusoidal Signals

The simulation in this section aimed to test the linearizing ability of gPNL. For this purpose, the mixing matrix was chosen as a nearly identity matrix. The number of sources were increased to four sinusoidal signals (4.4.16) with $N = 1000$ samples [45,

47]. The linear and PNL mixtures are expressed in (4.4.17) and (4.4.18), respectively. Plots of the original signals, their linear mixtures and PNL mixtures are shown in Fig. 4.19.

$$s_i(t) = sin(0.02\pi(2i-1)t) \qquad (i = 1, 2, 3, 4), \tag{4.4.16}$$

$$\begin{bmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \\ v_4(t) \end{bmatrix} = \begin{bmatrix} 0.787 & -0.198 & -0.066 & -0.081 \\ -0.201 & 0.748 & -0.203 & -0.296 \\ -0.262 & -0.100 & 0.987 & -0.117 \\ -0.087 & -0.108 & -0.132 & 0.822 \end{bmatrix} \begin{bmatrix} s_1(t) \\ s_2(t) \\ s_3(t) \\ s_4(t) \end{bmatrix} \tag{4.4.17}$$

$$\begin{aligned} x_1(t) &= (1.8v_1(t))^3 \\ x_2(t) &= 4tanh(5v_2(t)) \\ x_3(t) &= arctan(4v_3(t)) \\ x_4(t) &= 4logsig(7v_4(t)) - 2 \end{aligned} \tag{4.4.18}$$

where *logsig* denotes a logistic sigmoid function $logsig(v) = 1/(1 + e^{-v})$.

The gPNL was run with the inputs shown in Fig. 4.19(c) and the parameters as $L = 101$, $\mu = 0.2$ and $\xi = 0.001$. The final estimates y_i are shown in Fig. 4.19(f) along with the estimates of SOBI and Gauss-TD. Clearly, gPNL results are better than these two methods. The nonlinearity has been removed and all the originals source signals were estimated accurately. The shapes of the output signals match closely with the sine wave shapes of the original signals.

Table 4.3: Simulation 2: Mixture of four sinusoidal signals - Absolute correlation coefficient between the original sources and their estimates, $|r_{s_i,\hat{s}_i}|$.

	$\lvert r_{s_1,\hat{s}_1}\rvert$	$\lvert r_{s_2,\hat{s}_2}\rvert$	$\lvert r_{s_3,\hat{s}_3}\rvert$	$\lvert r_{s_4,\hat{s}_4}\rvert$
SOBI	0.909	0.807	0.938	0.957
Gauss-TD	0.968	0.890	0.923	0.975
gPNL	0.969	0.961	0.986	0.992

From Table 4.3, one can see that the absolute correlation coefficient of gPNL is higher than that of SOBI and Gauss-TD. This indicates the superior performance of gPNL over the other two algorithms. In addition, as we have observed in Section 4.3, in the case of mixture of four sinusoidal signals (simulation 2), the kPNL algorithm failed to separate the signals. However, as seen in Table 4.3, gPNL has successfully separated the four signals. This fact shows superiority of gPNL over kPNL.

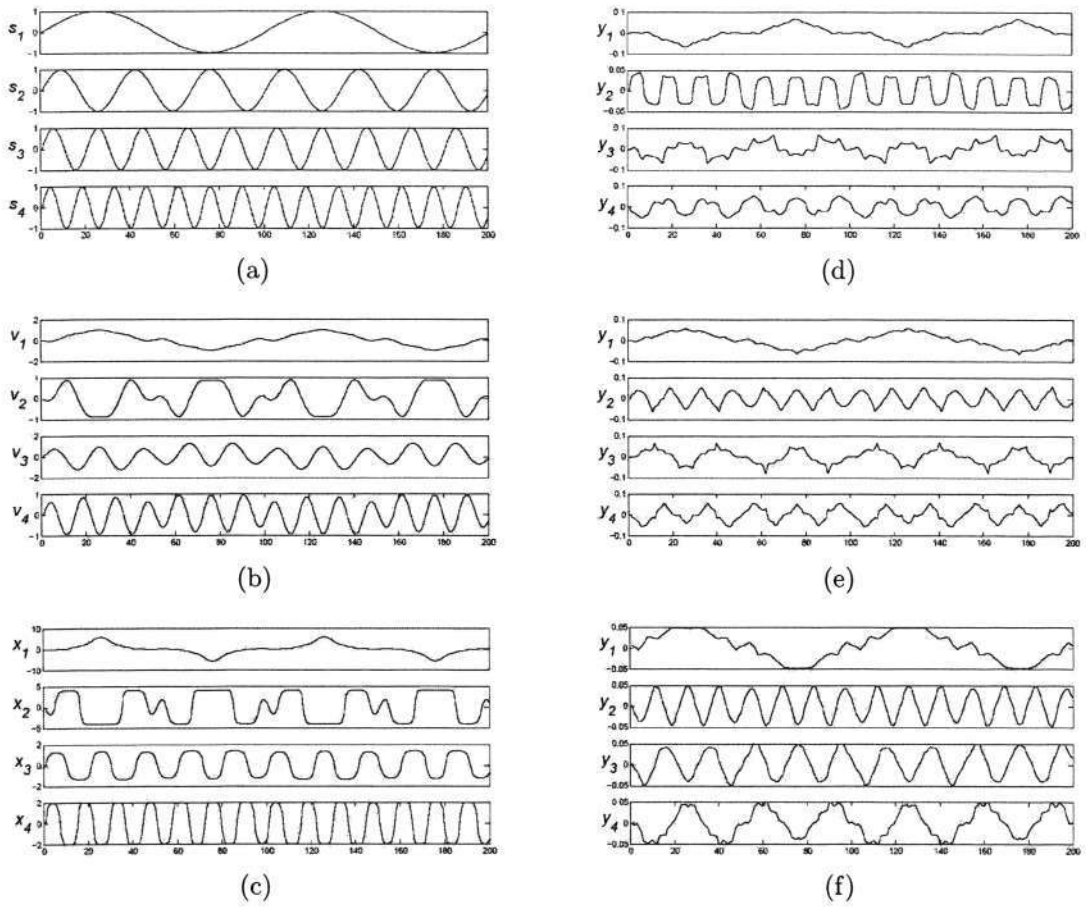


Figure 4.19: Time series plots of the signals (200 samples). (a) Unknown sources, (b) linear mixtures, (c) PNL mixtures, (d) SOBI estimates, (e) Gauss-TD estimates and (f) gPNL estimates.

Simulation 3: Mixture of Four Speech Signals

In this simulation, 5000 samples of four speeches (taken from [47]) were chosen as the original sources. Their linear and nonlinear mixtures are given by

$$\begin{bmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \\ v_4(t) \end{bmatrix} = \begin{bmatrix} -0.087 & 0.714 & -0.835 & 0.448 \\ 0.283 & -0.560 & -0.842 & -0.130 \\ -0.086 & 0.093 & -0.061 & 0.522 \\ -0.483 & -0.751 & 0.531 & 0.202 \end{bmatrix} \begin{bmatrix} s_1(t) \\ s_2(t) \\ s_3(t) \\ s_4(t) \end{bmatrix} \tag{4.4.19}$$

$$\begin{aligned} x_1(t) &= tanh(10v_1(t)) \\ x_2(t) &= 0.1v_2(t) + v_2(t)^3 \\ x_3(t) &= tanh(2v_3(t)) + v_3(t)^3 \\ x_4(t) &= v_4(t) + tanh(7v_4(t)) \end{aligned} \tag{4.4.20}$$

The plots of original speeches, linear mixtures and PNL mixtures are shown in

Fig. 4.20. Our proposed algorithm, gPNL, was run with setting $L = 151$, $\mu = 0.5$ and $\xi = 0.01$. The final outputs, y_i, are plotted in Fig. 4.20(f), next to the outputs of SOBI and Gauss-TD. Quantitative measure of gPNL, SOBI and Gauss-TD performance, in term of correlation coefficients between the estimates $\hat{s}_i$ and their original signals s_i, are provided in Table 4.4.

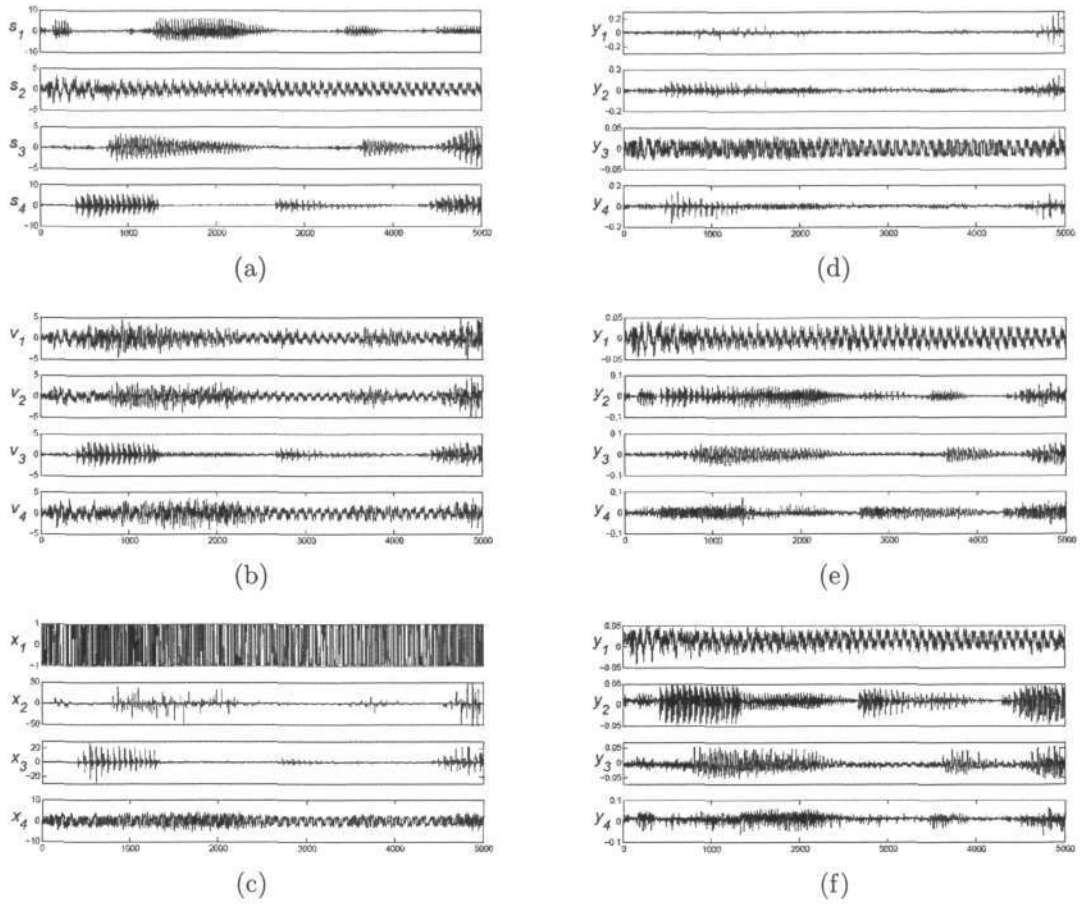


Figure 4.20: Time series plots of the signals (5000 samples). (a) Unknown speech signals, (b) linear mixtures, (c) PNL mixtures, (d) SOBI estimates, (e) Gauss-TD estimates and (f) gPNL estimates.

Figure 4.20 shows a very good performance of gPNL. The linear ICA method (SOBI) was not able to separate the nonlinear mixtures. It could only manage to estimate only one speech signal with adequate quality (with $r_{(s_2,\hat{s}_2)} \approx 0.8$). Comparing Fig. 4.20(a) with Fig. 4.20(f), one can see that gPNL successfully estimated all the four signals. Our proposed algorithm continues to provide good performance with high quality estimated signals; even the poorest output of gPNL has its correlation coefficient over 0.72. gPNL also provides a similar performance, compare to the Gauss-TD algorithm which uses additional assumption of the Gaussianity of the mixtures.

Table 4.4: Simulation 3: Mixture of four speech signals - Absolute correlation coefficient between the original sources and their estimates, $|r_{s_i,\hat{s}_i}|$.

	$\lvert r_{s_1,\hat{s}_1}\rvert$	$\lvert r_{s_2,\hat{s}_2}\rvert$	$\lvert r_{s_3,\hat{s}_3}\rvert$	$\lvert r_{s_4,\hat{s}_4}\rvert$
SOBI	0.404	0.816	0.590	0.695
Gauss-TD	0.770	0.984	0.956	0.611
gPNL	0.724	0.946	0.875	0.926

4.4.4 gPNL Summary

With a novel geometric approach to nonlinear ICA problem, gPNL has shown its impressive performance through various simulations. By considering the different characteristics between a plane and a nonlinear surface in multi-dimensional space, a simple linearizing process has been developed to transform the Post Nonlinear mixtures into linear mixtures without any additional assumption. From the linearized signals, the unknown sources can be estimated by any linear ICA algorithm. In addition, gPNL can perform the extraction effectively on various data sets without changing the method parameters.

Besides the advantages, several issues do exist and require further investigations to improve the algorithm. The first issue comes from the heuristic criterion in (4.4.3) that may leads gPNL to select an incorrect updating point. The convergence conditions and the criteria for selecting the algorithm's parameters are also other issues that need more study in future.

4.5 Conclusion

In this chapter, we have briefly reviewed through the challenging nonlinear ICA for BSS and contributed two novel methods to solve the problem. In contrast to the simple linear ICA situation in which a sole independent criterium is enough to extract the unknown signals, nonlinear ICA is much more complex and requires additional constraints. As it was discussed, general nonlinear ICA methods can not guaranty a unique results; therefore, researchers approach the non-uniqueness by limiting nonlinear ICA to different sub-models, either by assuming the source distributions or by constraining nonlinearity to a specific model. One of the most practical sub-models

is the Post Nonlinear (PNL) mixtures [202] whose nonlinear distortions are assumed to be the single variable functions.

Our first nonlinear method, the kPNL algorithm, follows the simple but practical PNL model. A modified Kurtosis-based updating rule is applied to eliminate the nonlinearity and to extract the unknown source signals. This approach is inspired from a linear ICA sequential extraction algorithm [6]. Using this simple and computational effective rule, the kPNL is trained to extract the signals one by one without any assumption of the number of the hidden sources.

gPNL is the second proposed approach to PNL ICA problem. In this method, the signal input samples are viewed as points in a multi-dimensional coordinate system and the linearizing process is considered as a geometric transformation of a nonlinear surface to a plane. By doing this, we are able to divide the separating system into two stages (i) a linearizing stage that uses geometric transformation to eliminate the nonlinearity and (ii) a linear separating stage where any normal linear ICA method can be applied to extract the hidden original signals. Impressive performance of gPNL has been shown through simulations on different kinds of data set.

Besides their advantages, both algorithms have issues that need further study. kPNL provides a convenient way to extract the signals but only works well on mild nonlinear mixtures. The sole Kurtosis-based learning function is not sufficient for both linearization and extraction. The gPNL, in contrast, provides good estimation but suffers from the ambiguity of the updating points. Therefore, we may need restrictions on the model in order to ensure the convergence of the algorithm.

Several proposals can be applied to improve the performance of gPNL and kPNL. First, we may assume that the number of observed signals is greater than the source signals. This assumption significantly reduces the ambiguity in gPNL method. Second, kPNL can be combined with gPNL to create a robust partial online sequential extraction algorithm that can deal with hard nonlinearity. The MLP components in kPNL can be replaced by the geometric approximations. Some initial input samples can be used in the geometric transformation, after which both geometric transformation and Kurtosis-based sequential extraction can be carried out at the same time. Finally, geometric approach is not limited to PNL model only. Extending it to general nonlinear ICA problems with multi-variate nonlinearity is worthy of future work.

Chapter 5

Conclusions

Independent Component Analysis (ICA) has become one of the most emerging method in the field of data representation and transformation. Unlike other representation methods that only exploit correlation (second order statistical dependence), ICA goes for higher-order independence criteria and therefore, provides a better transformation for data. The data, being represented by independent components, are more significant in term of statistical characteristics. The potential of ICA is shown by an increasing number of its applications in different fields, such as blind source separation, data analysis, and biomedicine.

Besides the development of linear ICA applications, study on nonlinear ICA is another interesting direction that attracts many researchers. It comes from the demand of an ICA system that works on those environments which could not be described properly by a linear model. In the thesis, we have presented our studies on these two major issues of ICA. In Chapter 3, two novel ICA-based techniques for watermarking were introduced, and in Chapter 4, we proposed two new approaches of nonlinear ICA for BSS problem.

5.1 ICA and Watermarking

The methods presented in Chapter 3 are our contributions to ICA application in the field of watermarking. The boom of digital multimedia and the Internet communications in recent years have raised a crucial issue on copyright and authentication and therefore, made watermarking become an important study. Various techniques including ICA have been applied in watermarking. One can easily see the similarities

between ICA mixing and watermark embedding, and between ICA demixing and watermark extraction. These are the key reason that makes ICA a potential technique for watermarking.

As ICA requires the number of observations to be equal or greater than the number of hidden sources, it is necessary to create additional observations in addition to the watermarked image. In the first watermarking method presented in this thesis, WMicaT, the above problem is solved by using image transpose. WMicaT considers the original image, the watermark and their transposes as original sources, then it creates the watermarked image and an additional key image as the mixtures. On the extraction side, these two images and their transposes act as mixtures for ICA to estimate the original image and the watermark.

The second watermarking method, WMicaD, approaches the problem from another direction. WMicaD utilizes two special watermarks that reveal different properties when being down-sized on different size. WMicaD also needs an additional key image but of much smaller size than the key image used for WMicaT. When the extraction is invoked, a down-sizing operation is applied on the watermarked image and key image, producing three images that correspond to the three mixtures of the original images and two watermarks.

With the help of ICA method, our two watermarking algorithms deliver robust extraction performance against strong attacks on the watermarked image. Experiments on various types of attack have shown that even when the watermarked image had been severely damaged, the algorithms could still extract watermarks with adequate quality. In comparison, WMicaT provides a slightly better performance in term of robustness, but WMicaD has computational and memory space advantages. In addition, WMicaD provides a two independent watermark embedding/extraction scheme that allows users to verify the ownership and to keep track of the copy number at the same time. The scheme also provides a capability of authentication for images.

5.2 Beyond Linear ICA

Simple linear ICA is good but not enough to fully describe the whole system in practice. Its extension, the nonlinear model, promises to be useful in a broader range

of application. Research works on nonlinear ICA have been increasing significantly in the last few years, focusing on the reduction of the computational complexity and the elimination of the indeterminacies. One of the effective ways is to study on the special cases, *i.e.*, the sub-models by applying constraints on the system. And among these sub-models, Post Nonlinear Mixtures (PNL) is probably the most popular one with many interesting characteristics.

In this thesis, we provided two novel algorithms, kPNL and gPNL that are tailored to PNL model. The first algorithm, the kPNL (kurtosis-based PNL) model, features modified MLP (Multi-Layer Perceptron) networks and a sequential linear ICA model. The MLPs and the linear system are updated in an unsupervised way by a kurtosis-based measurement. The hidden components are extracted one by one without any prior information about the number of sources. This feature is very useful in those applications where the users just need a few most important components and do not want to spend computer resources to extract all the unnecessary ones. Moreover, the kPNL provides an online extraction, *i.e.*, users do not need to collect all the data before extraction, the results are obtained right when the data is fed to the system.

The second algorithm was introduced in Chapter 4. The gPNL aims for a robust solution that can deal with hard nonlinear distortions. The gPNL (geometric PNL) model is divided into two stages. The first stage eliminates nonlinearity by using geometric transformation on the observed data. The second stage separates the linearized signals by using a classical linear ICA algorithm. Unlike many other PNL algorithms where only statistical independent criteria is used for both canceling nonlinearity and separating the components, gPNL exploits the representation difference in a multi-dimensional space between a nonlinear mixture and a linear one to transform the observations into linear mixtures. This additional characteristic allows gPNL to work on hard nonlinearity, and completely separates the second stage from the first stage, the choice of algorithm and independent measurement for second stage does not effect the performance of the first stage. Hence, the users have total freedom to select the most suitable linear ICA method for gPNL in their applications.

Chapter 6

Future Directions

WMicaT and WMicaD for image are just the first steps of applying ICA to watermarking. These two methods can be extended to other watermarking applications such as video and audio watermarking. The low computational workload of WMicaD is especially promising for video watermarking where instant extraction is a critical requirement. For more adaptive and robust watermarking, WMicaT and WMicaD can possibly be applied on transformed domain such as Fourier transform and wavelet transform. Combining WMicaT and WMicaD with other techniques like Quantization Index Modulation (QIM) is another considerable direction for future studies.

gPNL has shown its good performance in nonlinear separation. However, there are several issues that need more work to improve its performance. First of all, the ambiguity in gPNL geometric transformation has to be eliminated. One of the possible solutions is to have more observations than the number of hidden components so that the identification of the update point is unique. The next improvement can be a combination of kPNL and gPNL algorithms. The combined algorithm will inherit all the advantages of its preceding ones, including the abilities to work on hard nonlinearity and to extract the hidden components sequentially. Finally, considering geometric approach to a more general nonlinear model other than the PNL is another issue that needs further studies.

The mixing of 'pure' signals (speech, music) and images has promising applications. For example, the music can be embedded in a postcard to become a music card. Or, a small image representing owner signature can be hidden inside a song to protect the musician copyright. The digital images can be easily converted into one dimensional signals by reading the pixel values sequentially (in row-wise or column-wise). Studying

on this kind of applications is another interest for both linear ICA and nonlinear ICA researchers.

Separation of convolutive signals is a special branch of ICA study that requires different approaches with time shifting and temporal information. In the present form, our proposed algorithms may not work well for convolutive signals. However, with more thorough research, we can extend our approach to challenge the convolutive separation problem.

The question of whether it is good to employ nonlinear ICA for watermarking is still open for research. Replacing linear model with nonlinear model in current ICA applications is very interesting but requires further extensive study and adaptation. Even with the classical linear ICA, there are many applications that have not been thoroughly investigated. Until now, ICA is mostly applied for blind source separation. The use of the main characteristic of ICA for data transformation and representation, however, is just started to attract the research attention. Further studies on it will lead ICA to a whole new area in pattern recognition, feature extraction, data analysis and prediction.

Appendix A

The SOBI Algorithm

The SOBI (Second Order Blind Separation) algorithm, introduced by Belouchrani et al. [28], is a linear ICA algorithm that exploits the time coherence property of the sources signals. In contrast with other techniques that require higher-order statistics, the SOBI approach relies only on stationary second-order statistics that are based on a joint diagonalization of a set of covariance matrices.

At first, the observed signals in SOBI are modeled as the mixtures of source signals plus additive noise, given by

$$\mathbf{x}(t) = \mathbf{v}(t) + \mathbf{n}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t) \tag{A.0.1}$$

where $\mathbf{x}(t) = [x_1(t), ..., x_n(t)]^T$ is a noisy instantaneous linear mixture of source signals, $\mathbf{s}(t) = [s_1(t), ..., s_n(t)]^T$ is the vector of source signals, and $\mathbf{n}(t)$ denotes the additive noise vector. In this algorithm, $\mathbf{n}(t)$ is modeled as a stationary, temporally white, zero-mean complex random process independent of the source signals, and spatially white. That is

$$E[\mathbf{n}(t+\tau)\mathbf{n}^*(t)] = \sigma^2\delta(\tau)\mathbf{I} \tag{A.0.2}$$

where $\delta(\tau)$ is the Kronecker delta, $\mathbf{I}$ denotes the identity matrix, the superscript $*$ denotes the conjugate transpose and σ^2 denotes the noise variance.

The SOBI approach is started with an assumption that the source signal vector is either a deterministic ergodic sequence (hypothesis $H1$) or a stationary multivariate process (hypothesis $H1$).

Hypothesis $H1$:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbf{s}(t+\tau)\mathbf{s}^*(t) = E[\mathbf{s}(t+\tau)\mathbf{s}^*(t)] \tag{A.0.3}$$

$$= diag[\rho_1(\tau), ..., \rho_n(\tau)] \tag{A.0.4}$$

Hypothesis $H2$:

$$E[\mathbf{s}(t+\tau)\mathbf{s}^*(t)] = diag[\rho_1(\tau), ..., \rho_n(\tau)] \tag{A.0.5}$$

where $diag[\cdot]$ is the diagonal matrix formed with the elements of its vector valued argument. For convenience, the same notation is used for the deterministic averaging operation under hypothesis $H1$ and for ensemble averaging under $H2$. Assumptions $H1$ and $H2$ mean that the components $s_i(t)$, $i = 1, ..., n$ are mutually uncorrelated, and $\rho_i(\tau) = E[s_i(t+\tau)s_i^*(t)]$ denotes the autocovariance of $s_i(t)$. Under the above assumptions, the covariance matrices of the array output take the following structure

$$\mathbf{R}(0) = E[\mathbf{x}(t)\mathbf{x}^*(t)] = \mathbf{A}\mathbf{R}_\mathbf{s}(0)\mathbf{A}^* + \sigma^2\mathbf{I} \tag{A.0.6}$$

$$\mathbf{R}(\tau) = E[\mathbf{x}(t+\tau)\mathbf{x}^*(t)] = \mathbf{A}\mathbf{R}_\mathbf{s}(\tau)\mathbf{A}^* \qquad \tau \neq 0 \tag{A.0.7}$$

where $\mathbf{R}_\mathbf{s}$ denotes the covariance matrix of unknown sources $\mathbf{s}(t)$.

The key idea of SOBI is take from the following observation: blind identification is feasible by using spatial covariance matrices [27,28,212]. These covariance matrices present a simple structure that allows straightforward blind identification procedures based on eigendecomposition. In SOBI, the idea is implemented by using joint diagonalization of several covariance matrices. By using joint diagonalization, robustness of the algorithm is significantly increased at low additional computational cost in comparing with unique matrix solution as in [212].

Now let us consider a set $\mathcal{M} = \{\mathbf{M}_1, ..., \mathbf{M}_K\}$ of K matrices of size $n \times n$. The 'joint diagonality' (JD) criterion is defined, for any $n \times n$ matrix $\mathbf{V}$, as the following nonnegative function of $\mathbf{V}$:

$$JD(\mathcal{M}, \mathbf{V}) = \sum_{k=1}^{K} off(\mathbf{V}^*\mathbf{M}_k\mathbf{V}) \tag{A.0.8}$$

where $off(\cdot)$ of an $n \times n$ matrix $\mathbf{M}$ with entries M_{ij} is defined as

$$off(\mathbf{M}) = \sum_{1 \leq i \neq j \leq n} |M_{ij}|^2 \tag{A.0.9}$$

Clearly, $off(\mathbf{M})$ equals to zero if $\mathbf{M}$ is a diagonal matrix. A unitary matrix is said to be a 'joint diagonalizer' of the set $\mathcal{M}$ if it minimizes the JD criterion (A.0.8) over the set of all unitary matrices.

After having all the necessary definitions, we can describe the SOBI algorithm in detail. Basically, the SOBI procedure includes two main tasks: whitening the

$\mathbf{v}$ part (A.0.1) of the observed signals, and finding the joint diagonalizer over the set of covariance matrices of the observation. Whitening is carried out by finding a whitening matrix $\mathbf{W}$ that satisfies

$$E[\mathbf{W}\mathbf{v}(t)\mathbf{v}^*(t)\mathbf{W}^*] = \mathbf{W}\mathbf{R}_\mathbf{v}(0)\mathbf{W}^* = \mathbf{W}\mathbf{A}\mathbf{A}^*\mathbf{W}^* = \mathbf{I} \tag{A.0.10}$$

Steps involved in the algorithm are as follows

1. Estimate the covariance matrix $\mathbf{R}(0)$ from T data samples. Let us denote the n largest eigenvalues of $\mathbf{R}(0)$ by $\lambda_1, ..., \lambda_n$ and the corresponding eigenvectors by $\mathbf{h}_1, ..., \mathbf{h}_n$.

2. Under the white noise assumption, an estimate σ^2 of the noise variance is the average of the $m - n$ smallest eigenvalues of $\mathbf{R}(0)$. The whitened signals are $\mathbf{z}(t) = [z_1(t), ..., z_n(t)]^T$ which are computed by

$$z_i(t) = (\lambda_i - \sigma^2)^{-1/2}\mathbf{h}_i^*(t)\mathbf{x}(t) \qquad i = 1, ..., n \tag{A.0.11}$$

   This is equivalent to form a whitening matrix, $\mathbf{W}$ by

$$\mathbf{W} = [\lambda_1 - \sigma^2)^{-1/2}\mathbf{h}_i, ..., \lambda_n - \sigma^2)^{-1/2}\mathbf{h}_n]^* \tag{A.0.12}$$

3. Let $\overline{\mathbf{R}}(\tau)$ denote the spatially whitened covariance matrices of $\mathbf{R}(\tau)$, then $\overline{\mathbf{R}}(\tau)$ is defined as

$$\overline{\mathbf{R}}(\tau) = \mathbf{W}\mathbf{R}(\tau)\mathbf{W}^* \forall \tau \neq 0 \tag{A.0.13}$$

   Hence, we can estimate $\overline{\mathbf{R}}(\tau)$ by computing the covariance matrices of $\mathbf{z}$ for a fixed set of time lag $\tau \in \{\tau_j | j = 1, ..., K\}$.

4. Obtain a unitary matrix, $\mathbf{U}$ as the joint diagonalizer of the set $\{\overline{\mathbf{R}}(\tau_j) | j = 1, ..., K\}$.

5. The source signals are estimated as

$$\mathbf{s}(t) = \mathbf{U}^*\mathbf{W}\mathbf{x}(t) \tag{A.0.14}$$

   and the mixing matrix $\mathbf{A}$ is estimated as

$$\mathbf{A} = \mathbf{W}^{\#}\mathbf{U} \tag{A.0.15}$$

   where # denotes the Moore-Penrose pseudo–inverse.

Appendix B

MLP Structure and Notations

In this appendix, we specify the MLP network structure and the notations which are used in Chapter 4. These are adapted from [76].

B.1 The MLP Structure

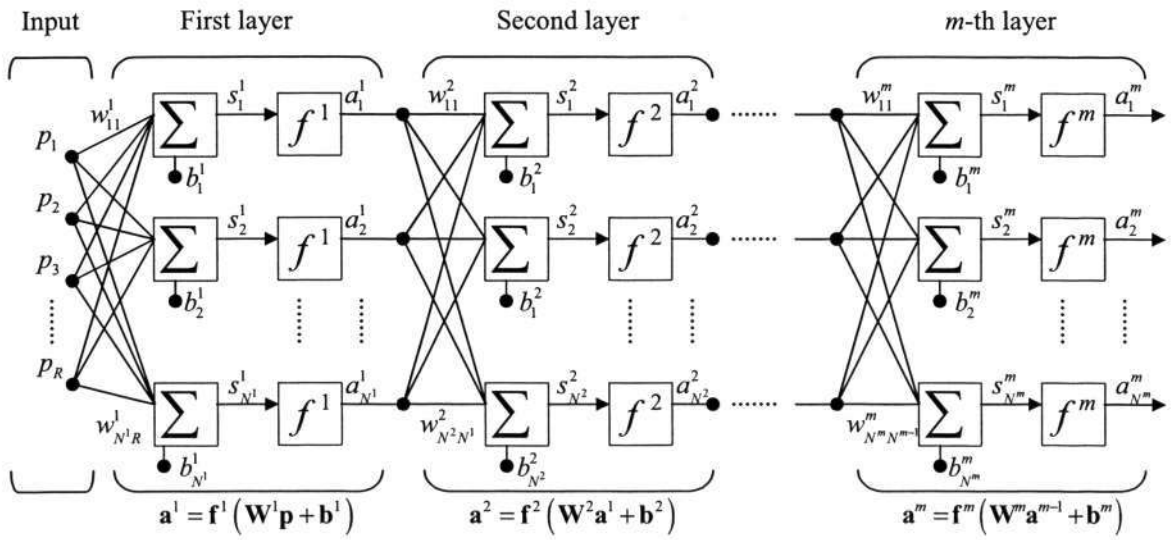


Figure B.1: Structure of a m layer MLP network [76]. The MLP has R inputs, $p_1, p_2, ..., p_R$, and l-th layer has N^l outputs, $a_1^l, a_2^l, ..., a_{N^l}^l$.

Figure B.1 represents structure of a m layer MLP network with the following notations

$\mathbf{p} = [p_1, p_2, ..., p_R]^T$: vector of R inputs.

$\mathbf{W}^l_{N^l \times N^{l-1}}$: weight matrix of the l-th layer.

$\mathbf{b}^l = [b_1, b_2, ..., b_{N^l}]^T$: bias vector or sum of the l-th layer.

$\mathbf{s}^l = [s_1, s_2, ..., s_{N^l}]^T$: net input vector or sum of the l-th layer.

f^l: logistic function of the l-th layer.

$\mathbf{a}^l = [a_1, a_2, ..., a_{N^l}]^T$: output vector of the l-th layer.

with $l = 1, 2, ..., m$ and N^l denote the number of neurons of the l-th layer (we define $N^0 = R$). An abbreviated version of the MLP structure which uses matrix notation is shown in Fig. B.2. This abbreviated representation will be used for deriving the back-propagation learning algorithm.

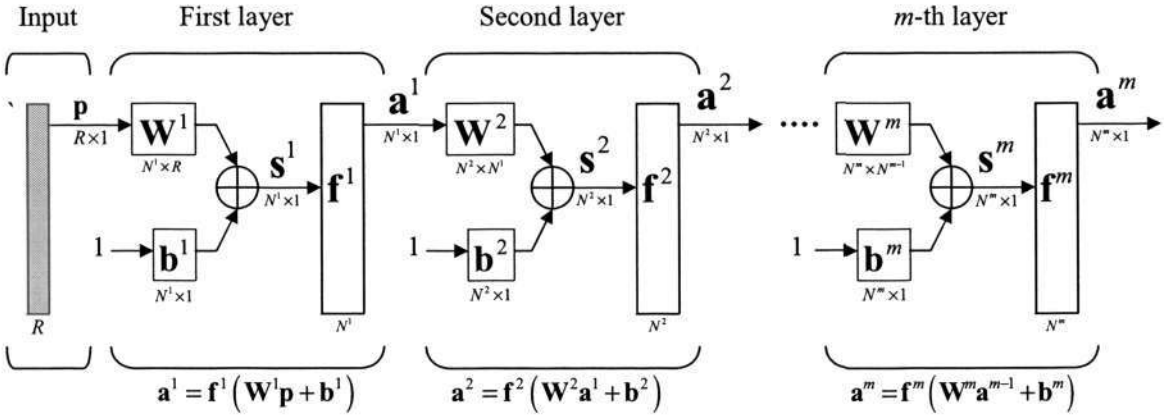


Figure B.2: Structure of a m layer MLP network presented by abbreviated matrix-based notations [76]. The MLP has R inputs, $p_1, p_2, ..., p_R$, and l-th layer has N^l outputs, $a_1^l, a_2^l, ..., a_{N^l}^l$.

Using the above notations, we derive the following formulas for calculating the net outputs

$$\mathbf{s}^l = \mathbf{W}^l\mathbf{a}^{l-1} + \mathbf{b}^l \tag{B.1.1}$$

$$\mathbf{a}^l = \mathbf{f}^l(\mathbf{s}^l) = \mathbf{f}^l(\mathbf{W}^l\mathbf{a}^{l-1} + \mathbf{b}^l) \tag{B.1.2}$$

$$\mathbf{a}^0 = \mathbf{p} \tag{B.1.3}$$

$$\mathbf{a} = \mathbf{a}^m \tag{B.1.4}$$

where

$$\mathbf{a}^l = \mathbf{f}^l(\mathbf{s}^l) = \begin{bmatrix} f^l(s_1^l) \\ f^l(s_2^l) \\ \vdots \\ f^l(s_{N^l}^l) \end{bmatrix} \tag{B.1.5}$$

for $l = 1, 2, ..., m$ and $\mathbf{a}$ denotes output vector of MLP network.

B.2 The Back-propagation Algorithm

The back-propagation algorithm for MLP is a generalization of the Least Mean Square (LMS) algorithm [76]. The algorithm uses mean square error as the performance index. Let us denote the MLP desired outputs by $\mathbf{d} = [d_1, d_2, ..., d_{N^m}]^T$, then our goal is to adjust the network parameters (weight matrices and bias vectors) so that they minimize the mean square error (MSE), $\mathcal{E}$, given by

$$\mathcal{E} = E[\mathbf{e}^T\mathbf{e}] = E[(\mathbf{d}-\mathbf{a})^T(\mathbf{d}-\mathbf{a})] \tag{B.2.1}$$

At iteration k, we approximate the MSE, $\mathcal{E}$, by

$$\hat{\mathcal{E}} = (\mathbf{d}(k)-\mathbf{a}(k))^T(\mathbf{d}(k)-\mathbf{a}(k)) \tag{B.2.2}$$

where $\hat{\mathcal{E}}$ denotes the approximate MSE. Applying steepest descent algorithm yield the following updating rule at iteration k

$$w_{ij}^l(k+1) = w_{ij}^l(k) - \alpha\frac{\partial\hat{\mathcal{E}}}{\partial w_{ij}^l} \tag{B.2.3}$$

$$b_i^l(k+1) = b_i^l(k) - \alpha\frac{\partial\hat{\mathcal{E}}}{\partial b_i^l} \tag{B.2.4}$$

where w_{ij}^l is the (i,j)-th entry of weight matrix $\mathbf{W}^l$, b_i^l is i-th element of bias vector $\mathbf{b}^l$, and α is the learning rate.

Applying chain rule to the last term in (B.2.3) and (B.2.4) yields

$$\frac{\partial\hat{\mathcal{E}}}{\partial w_{ij}^l} = \frac{\partial\hat{\mathcal{E}}}{\partial s_i^l}\times\frac{\partial s_i^l}{\partial w_{ij}^l} = v_i^l \times a_j^{l-1} \tag{B.2.5}$$

$$\frac{\partial\hat{\mathcal{E}}}{\partial b_i^l} = \frac{\partial\hat{\mathcal{E}}}{\partial s_i^l}\times\frac{\partial s_i^l}{\partial b_i^l} = v_i^l \times 1 \tag{B.2.6}$$

where v_i^l is defined as a sensitivity function (or sensitivity for short), $v_i^l \equiv \frac{\partial\hat{\mathcal{E}}}{\partial s_i^l}$. Replacing the two equations back into (B.2.3) and (B.2.4) derives the updating rules for w_{ij}^l and b_i^l at iteration k

$$w_{ij}^l(k+1) = w_{ij}^l(k) - \alpha v_i^l a_j^{l-1} \tag{B.2.7}$$

$$b_i^l(k+1) = b_i^l(k) - \alpha v_i^l \tag{B.2.8}$$

or in the matrix form

$$\mathbf{W}^l(k+1) = \mathbf{W}^l(k) - \alpha\mathbf{v}^l(\mathbf{a}^{l-1})^T \tag{B.2.9}$$

$$\mathbf{b}^l(k+1) = \mathbf{b}^l(k) - \alpha\mathbf{v}^l \tag{B.2.10}$$

where

$$\mathbf{v}^l \equiv \frac{\partial \hat{\mathcal{E}}}{\partial \mathbf{s}^l} = \begin{bmatrix} \frac{\partial \hat{\mathcal{E}}}{\partial s_1^l} \\ \frac{\partial \hat{\mathcal{E}}}{\partial s_2^l} \\ \vdots \\ \frac{\partial \hat{\mathcal{E}}}{\partial s_{N^l}^l} \end{bmatrix} \tag{B.2.11}$$

Now the remaining task is to compute the sensitivities, $\mathbf{v}^l$. We will try to calculate recursively $\mathbf{v}^l$ from $\mathbf{v}^{l+1}$. Let's consider the following Jacobian matrix

$$\frac{\partial \mathbf{s}^{l+1}}{\partial \mathbf{s}^l} \equiv \begin{bmatrix} \frac{\partial s_1^{l+1}}{\partial s_1^l} & \frac{\partial s_1^{l+1}}{\partial s_2^l} & \cdots & \frac{\partial s_1^{l+1}}{\partial s_{N^l}^l} \\ \frac{\partial s_2^{l+1}}{\partial s_1^l} & \frac{\partial s_2^{l+1}}{\partial s_2^l} & \cdots & \frac{\partial s_2^{l+1}}{\partial s_{N^l}^l} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial s_{N^{l+1}}^{l+1}}{\partial s_1^l} & \frac{\partial s_{N^{l+1}}^{l+1}}{\partial s_2^l} & \cdots & \frac{\partial s_{N^{l+1}}^{l+1}}{\partial s_{N^l}^l} \end{bmatrix} \tag{B.2.12}$$

whose (i,j)-th element can be expressed as

$$\begin{aligned} \frac{\partial s_i^{l+1}}{\partial s_j^l} &= \frac{\partial (\sum_{k=1}^{N^l} w_{ik}^{l+1} a_k^l + b_i^{l+1})}{\partial s_j^l} = w_{ij}^{l+1} \frac{\partial a_j^l}{\partial s_j^l} \\ &= w_{ij}^{l+1} \frac{\partial f^l(s_j^l)}{\partial s_j^l} \\ &= w_{ij}^{l+1} \dot{f}^l(s_j^l) \end{aligned} \tag{B.2.13}$$

where $\dot{f}^l(s_j^l)$ denotes the derivative of the activation function of the l-th layer, $\dot{f}^l(s_j^l) = \frac{\partial f^l(s_j^l)}{\partial s_j^l}$. Thus, the Jacobian matrix in (B.2.12) can be rewritten as

$$\frac{\partial \mathbf{s}^{l+1}}{\partial \mathbf{s}^l} = \mathbf{W}^{l+1} \dot{\mathbf{F}}^l(\mathbf{s}^l) \tag{B.2.14}$$

where

$$\dot{\mathbf{F}}^l(\mathbf{s}^l) = \begin{bmatrix} \dot{f}^l(s_1^l) & 0 & \cdots & 0 \\ 0 & \dot{f}^l(s_2^l) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \dot{f}^l(s_{N^l}^l) \end{bmatrix} \tag{B.2.15}$$

Finally, we can establish the recursive relation between the two consecutive sensitivities, $\mathbf{v}^l$ and $\mathbf{v}^{l+1}$. Applying chain rule in matrix form on $\mathbf{v}^l$ (B.2.11) and utilizing the result in (B.2.14) yield

$$\begin{aligned} \mathbf{v}^l &= \frac{\partial \hat{\mathcal{E}}}{\partial \mathbf{s}^l} = (\frac{\partial \mathbf{s}^{l+1}}{\partial \mathbf{s}^l})^T \frac{\partial \hat{\mathcal{E}}}{\partial \mathbf{s}^{l+1}} \\ &= \dot{\mathbf{F}}^l(\mathbf{s}^l)(\mathbf{W}^{l+1})^T \mathbf{v}^{l+1} \end{aligned} \tag{B.2.16}$$

Equation (B.2.16) clearly shows how the sensitivities are propagated backward through the network from the last layer to the first layer

$$\mathbf{v}^m \rightarrow \mathbf{v}^{m-1} \rightarrow \ldots \rightarrow \mathbf{v}^2 \rightarrow \mathbf{v}^1 \tag{B.2.17}$$

To complete the algorithm, the last sensitivity, $\mathbf{v}^m$, is obtained directly from the error function by

$$\begin{aligned} v_i^m &= \frac{\partial \hat{\mathcal{E}}}{\partial s_i^m} = \frac{\partial (\mathbf{d}-\mathbf{a})^T(\mathbf{d}-\mathbf{a})}{\partial s_i^m} \\ &= \frac{\partial \sum_{j=1}^{N^m}(d_j-a_j)^2}{\partial s_i^m} = -2(d_i-a_i)\frac{\partial a_i}{\partial s_i^m} \\ &= -2(d_i-a_i)\frac{\partial f^m(s_i^m)}{\partial s_i^m} \\ &= -2(d_i-a_i)\dot{f}^m(s_i^m) \qquad i=1,2,...,N^m \end{aligned} \tag{B.2.18}$$

This formula can be expressed in matrix form as

$$\mathbf{v}^m = -2\dot{\mathbf{F}}^m(\mathbf{s}^m)(\mathbf{d}-\mathbf{a}) \tag{B.2.19}$$

In summary, a back-propagation algorithm for a MLP network includes the following steps in each iteration

1. Propagate the inputs forward through the network

$$\mathbf{a}^0 = \mathbf{p} \tag{B.2.20}$$

$$\mathbf{a}^{l+1} = \mathbf{f}^{l+1}(\mathbf{W}^{l+1}\mathbf{a}^l + \mathbf{b}^{l+1}) \qquad l=1,...,m-1 \tag{B.2.21}$$

$$\mathbf{a} = \mathbf{a}^m \tag{B.2.22}$$

2. Propagate the sensitivities backward through the network

$$\mathbf{v}^m = -2\dot{\mathbf{F}}^m(\mathbf{s}^m)(\mathbf{d}-\mathbf{a}) \tag{B.2.23}$$

$$\mathbf{v}^l = \dot{\mathbf{F}}^l(\mathbf{s}^l)(\mathbf{W}^{l+1})^T\mathbf{v}^{l+1} \qquad l=m-1,...,1 \tag{B.2.24}$$

3. Update weights and bias by steepest descent rules

$$\mathbf{W}^l(k+1) = \mathbf{W}^l(k) - \alpha\mathbf{v}^l(\mathbf{a}^{l-1})^T \tag{B.2.25}$$

$$\mathbf{b}^l(k+1) = \mathbf{b}^l(k) - \alpha\mathbf{v}^l \tag{B.2.26}$$

Appendix C

Proof of Lemma 4.2

Lemma *Let* $(\mathbf{f}, \mathbf{A})$ *be a PNL mixture and* $(\mathbf{g}, \mathbf{W})$ *be a PNL separation structure, where we have the following*

- $\mathbf{A}$ *is a regular matrix and has at least two nonzero entries per row or per column.*
- f_i, $i = 1, 2, ..., n$ *are differentiable invertible functions.*
- $\mathbf{W}$ *is a regular matrix.*
- $h_i = g_i \circ f_i$ *satisfies* $\forall u \in \mathbb{R}, h_i(u) \neq 0$, *for all* $i = 1, 2, ..., n$.

Suppose that each source s_i *accepts a density function vanishes at one point at least, and then, the output* $\mathbf{y}$ *of the separation structure has mutually independent components if and only if the* h_i *components are linear, and* $\mathbf{B}$ *is a separating matrix [202].*

Proof of Lemma.

If $h_i = g_i \circ f_i$ are linear and $\mathbf{B}$ is a separating matrix, then components of $\mathbf{y}$ are independent.

Conversely, if components of $\mathbf{y}$, y_i, are mutually independent. Denote by p_i and q_i the probability density function (pdf) of s_i and y_i, respectively, then the pdf of random vector $\mathbf{s}$ can be expressed as

$$\begin{aligned} p_{\mathbf{s}}(\mathbf{s}) &= \prod_{i=1}^{n} p_i(s_i) \\ &= \prod_{i=1}^{n} q_i\Big(\sum_{j=1}^{n} b_{ij} h_j\Big(\sum_{k=1}^{n} a_{jk} s_k\Big)\Big) \cdot \left| \prod_{i=1}^{n} h_i'\Big(\sum_{j=1}^{n} a_{ij} s_j\Big) \det \mathbf{A} \det \mathbf{W} \right| \end{aligned} \quad \text{(C.0.1)}$$

for all $\mathbf{s} \in \mathbb{R}^n$.

As it was assumed, there exists a point $\mathbf{s}^0 \in \mathbb{R}^n$ such that $p_{\mathbf{s}}(\mathbf{s}^0) = 0$. From (C.0.1), given that $\forall i, \forall u \in \mathbb{R}, h_i'(u) \neq 0$, there exists a point $\mathbf{y}^0 \in \mathbb{R}^n$ such that

$$\prod_{i=1}^{n} q_i(y_i^0) = 0 \tag{C.0.2}$$

and consequently, there exists $l \in \{1, 2, ..., n\}$ such that

$$q_l(y_l^0) = 0 \tag{C.0.3}$$

Now let $\mathcal{H}_l \in \mathbb{R}^n$ be the hypersurface of the implicit function

$$\sum_{j=1}^{n} b_{lj} h_j(\sum_{k=1}^{n} a_{jk} s_k) = y_l^0 \tag{C.0.4}$$

From (C.0.1) we have

$$\forall \mathbf{s} \in \mathcal{H}_l : p_{\mathbf{s}}(\mathbf{s}) = 0 \tag{C.0.5}$$

where point $\mathbf{s}$ of $\mathcal{H}_l$ is a transform of point $\mathbf{x}$ of the hyperplane $\mathcal{P}_l$ of equation $\sum_{j=1}^{n} b_{lj} x_j = y_l^0$ by a transformation $\mathbf{A}^{-1}\mathbf{h}^{-1}(\cdot)$.

Suppose that $\mathcal{H}_l$ is not parallel to any of the n hyperplanes $s_i = 0$, $i = 1, 2, ..., n$. Then the projection of $\mathcal{H}_l$ onto each axis consists of $\mathbb{R}$. In fact, for each coordinate $s_i \in \mathbb{R}$, we can find $s_1, ..., s_{i-1}, s_{i+1}, ..., s_n \in \mathbb{R}^{n-1}$ such that $\mathbf{h}(\mathbf{As}) \in \mathcal{P}_l$. Hence, $p_{\mathbf{s}}(\mathbf{s}) = 0$ for all $\mathbf{s} \in \mathbb{R}^n$, which is impossible since $p_{\mathbf{s}}$ sum to 1.

Consequently, for each l such that (C.0.3) holds, there exists $m_l \in \{1, 2, ..., n\}$ such that $\mathcal{H}_l$ is parallel to the hyperplane $s_{m_l} = 0$. This impose that for all $s_1, s_2, ..., s_n$

$$\sum_{j=1}^{n} b_{lj} h_j(\sum_{k=1}^{n} a_{jk} s_k) = k_{m_l}(s_{m_l}) \tag{C.0.6}$$

where k_{m_l} is any function. Assume that (C.0.3) holds for all $l = 1, 2, ..., n$, then (C.0.6) is simplified by letting $m_l = l$. We have

$$\sum_{j=1}^{n} b_{lj} h_j(\sum_{k=1}^{n} a_{jk} s_k) = k_l(s_l) \tag{C.0.7}$$

Differentiating each equation with respect to s_i, $i = 1, 2, ..., n$ yields

$$\begin{bmatrix} k_1'(s_1) & & 0 \\ & \ddots & \\ 0 & & k_n'(s_n) \end{bmatrix} = \mathbf{W} \begin{bmatrix} h_1'(\sum_{i=1}^{n} a_{1i} s_i) & & 0 \\ & \ddots & \\ 0 & & h_n'(\sum_{i=1}^{n} a_{ni} s_i) \end{bmatrix} \mathbf{A} \tag{C.0.8}$$

Let us denote the above two matrices by $\mathbf{D}(\mathbf{s}) = diag[k_1'(s_1), ..., k_n'(s_n)]$ and $\mathbf{B}(\mathbf{s}) = diag[h_1'(\sum_{i=1}^n a_{1i}s_i), ..., h_n'(\sum_{i=1}^n a_{ni}s_i)])$, respectively. For $\mathbf{s}_1 \neq \mathbf{s}_2$, we have

$$\mathbf{D}(\mathbf{s}_1) = \mathbf{W}\mathbf{B}(\mathbf{s}_1)\mathbf{A} \tag{C.0.9}$$

$$\mathbf{D}(\mathbf{s}_2) = \mathbf{W}\mathbf{B}(\mathbf{s}_2)\mathbf{A} \tag{C.0.10}$$

Eliminating $\mathbf{W}$ yields

$$\mathbf{D}(\mathbf{s}_1)^{-1}\mathbf{D}(\mathbf{s}_2) = \mathbf{A}^{-1}\mathbf{B}(\mathbf{s}_1)^{-1}\mathbf{B}(\mathbf{s}_2)\mathbf{A} \tag{C.0.11}$$

Recall that $\mathbf{D}(\mathbf{s})$ and $\mathbf{B}(\mathbf{s})$ are two diagonal matrices, then (C.0.11) can be rewritten as

$$\mathbf{A}\mathbf{\Lambda}(\mathbf{s}_1, \mathbf{s}_2) = \mathbf{A}\mathbf{\Omega}(\mathbf{s}_1, \mathbf{s}_2) \tag{C.0.12}$$

where $\mathbf{\Lambda}$ and $\mathbf{\Omega}$ are diagonal matrices. Hence, for each i, j we have

$$a_{ij}(\lambda_{jj}(\mathbf{s}_1, \mathbf{s}_2) - \omega_{ii}(\mathbf{s}_1, \mathbf{s}_2)) = 0 \tag{C.0.13}$$

Since $\mathbf{A}$ is regular, for each column j, there exists at least one $i = \sigma(j)$, where σ is a permutation, such that $a_{\sigma(j)j} \neq 0$. Thus

$$\forall j : \lambda_{jj}(\mathbf{s}_1, \mathbf{s}_2) = \omega_{\sigma(j)\sigma(j)}(\mathbf{s}_1, \mathbf{s}_2) \tag{C.0.14}$$

Suppose that there exists another nonzero entry on column j: $a_{\alpha jj} \neq 0$, where α is a permutation, and $\alpha(j) \neq \sigma(j)$, then we have

$$\lambda_{jj}(\mathbf{s}_1, \mathbf{s}_2) = \omega_{\alpha(j)\alpha(j)}(\mathbf{s}_1, \mathbf{s}_2) = \omega_{\sigma(j)\sigma(j)}(\mathbf{s}_1, \mathbf{s}_2) \tag{C.0.15}$$

Fixing $\mathbf{s}_2$ at a constant value and setting $\mathbf{s}_1 = \mathbf{s}$, a variable, yields

$$h'_{\alpha(j)}(\sum_{i=1}^{n} a_{\alpha(j)i}s_i)) = C_j h'_{\sigma(j)}(\sum_{i=1}^{n} a_{\sigma(j)i}s_i)) \tag{C.0.16}$$

where C_j is a constant. Since $\alpha(j) \neq \sigma(j)$ and $\mathbf{A}$ is regular, the two linear forms involved in (C.0.16) are independent. Hence

$$h'_{\alpha(j)}(x) = C_j h'_{\sigma(j)}(y) \qquad \forall x, y \in \mathbb{R} \tag{C.0.17}$$

This equation clearly shows that $h_{\alpha(j)}$ and $h_{\sigma(j)}$ are linear. Similar results are obtained by considering two nonzero entries of the rows of $\mathbf{A}$.

Appendix C: Proof of Lemma 4.2

It was supposed that functions h_j and their inverses are defined all over $\mathbf{R}$. It is nonrestrictive since if an h_j was defined only on a subset $K_j \subset \mathbb{R}$, then the pdf $p_{\mathbf{s}}$ would be zero whenever $\mathbf{As} \notin \mathbb{R}^{n-1} \times K_j$. The same reasoning holds for the inverse, h_j^{-1}.

Since h_i are linear, the problem reduces to a linear source separation problem, and $\mathbf{W}$ is a separating matrix. □

Bibliography

[1] S. Achard, D. T. Pham, and C. Jutten, "Blind source separation in post nonlinear mixtures," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'01)*, San Diego, CA, USA, 2001, pp. 295–300.

[2] R. Aichner, H. Buchner, F. Yan, and W. Kellermann, "Real-time convolutive blind source separation based on a broadband approach," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 840–848.

[3] L. B. Almeida, "Linear and nonlinear ICA based on mutual information," in *Proc. IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium (AS-SPCC'00)*, Alberta, Canada, Oct. 2001, pp. 117–122.

[4] ——, "MISEP - linear and nonlinear ICA based on mutual information," *Journal of Machine Learning Research*, vol. 4, pp. 1297–1318, Dec. 2003.

[5] S.-I. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, vol. 10, no. 2, pp. 251–276, 1998.

[6] S.-I. Amari and A. Cichocki, "Adaptive blind signal processing - neural network approaches," *Proceedings of the IEEE*, vol. 86, no. 10, pp. 2026–2048, 1998.

[7] S.-I. Amari, A. Cichocki, and H. H. Yang, "A new learning algorithm for blind signal separation," in *Proc. of the Conf. in Advances in Neural Information Processing Systems 8*, 1996, pp. 757–763.

[8] J. Anemüller, J.-R. Duann, T. J. Sejnowski, and S. Makeig, "Unraveling spatio-temporal dynamics in fMRI recordings using complex ICA," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 1103–1110.

[9] J. Anemüller, T. J. Sejnowski, and S. Makeig, "Reliable measurement of cortical flow patterns using complex independent component analysis of electroencephalographic signals," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 1009–1016.

[10] J. Atick, "Entropy minimization: A design principle for sensory perception?" *International Journal of Neural Systems*, vol. 3, pp. 81–90, 1992.

[11] M. Babaie-Zadeh, C. Jutten, and K. Nayebi, "A geometric approach for separating post nonlinear mixtures," in *Proc. of the XI European Signal Processing Conf. (EUSIPCO'02)*, vol. 02, Toulouse, France, 2002, pp. 11–14.

[12] F. R. Bach and M. I. Jordan, "Kernel independent component analysis," *Journal of Machine Learning Research*, vol. 3, pp. 1–48, 2002.

[13] ——, "Beyond independent components: Trees and clusters," *Journal of Machine Learning Research*, vol. 4, pp. 1205–1233, Dec. 2003.

[14] A. D. Back and A. S. Weigend, "A first application of independent component analysis to extracting structure from stock returns," *International Journal of Neural Systems*, vol. 8, no. 4, pp. 473–484, 1997.

[15] K. Baek, B. A. Draper, J. R. Beveridge, and K. She, "PCA vs. ICA: A comparison on the FERET data set," in *Proc. Int. Joint Conference on Information Science (JCIS'02)*, North Carolina, USA, Mar. 2002, pp. 824–827.

[16] H. B. Barlow, "Single units and sensation: A neuron doctrine for perceptual psychology?" *Perception*, vol. 1, pp. 371–394, 1972.

[17] ——, "Possible principles underlying the transformations of sensory messages," in *Sensory Communication*, W. A. Rosenblith, Ed. MIT press, 1961, pp. 217–234.

[18] ——, "Unsupervised learning," *Neural Computation*, vol. 1, pp. 295–311, 1989.

[19] ——, "What is the computational goal of the neocortex?" in *Large-scale neuronal theories of the brain*, C. Koch and J. L. Davis, Eds. Cambridge, MA: MIT press, 1994, pp. 1–22.

[20] M. Barret and M. Narozny, "Application of ICA to lossless image coding," in *Proc. Int. Symposium on Independent Component Analysis and Blind Signal Separation (ICA'03)*, Nara, Japan, Apr. 2003, pp. 855–859.

[21] A. K. Barros, R. Vigário, V. Jousmäki, and N. Ohnishi, "Extraction of event-related signals from multichannel bioelectrical measurements," *IEEE Trans. Biomed. Eng.*, vol. 47, no. 5, pp. 583–588, 2000.

[22] M. S. Bartlett, J. R. Movellan, and T. J. Sejnowski, "Face recognition by independent component analysis," *IEEE Trans. Neural Networks*, vol. 13, no. 6, pp. 1450–1464, Nov. 2002.

[23] J. Basak, A. Sudarshan, D. Trivedi, and M. S. Santhanam, "Weather data mining using independent component analysis," *Journal of Machine Learning Research*, vol. 5, pp. 239–253, Mar. 2004.

[24] G. Basalyga and M. Rattray, "Statistical dynamics of on-line independent component analysis," *Journal of Machine Learning Research*, vol. 4, pp. 1393–1410, Dec. 2003.

[25] A. J. Bell and T. J. Sejnowski, "An information-maximisation approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, no. 6, pp. 1129–1159, 1995.

[26] ——, "The 'independent components' of natural scenes are edge filters," *Vision Research*, vol. 37, pp. 3327–3338, 1997.

[27] A. Belouchrani, K. Abed-Meraim, J. F. Cardoso, and E. Moulines, "Second-order blind separation of correlated sources," in *Proc. Int. Conf. on Digital Signal Processing*, Cyprus, 1993, pp. 346–351.

[28] ——, "A blind source separation technique using second-order statistics," *IEEE Trans. Signal Processing*, vol. 45, no. 2, pp. 434–444, Feb. 1997.

[29] C. M. Bishop, *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[30] B. B. Biswal and J. L. Ulmer, "Blind source separation of multiple signal sources of fMRI data sets using independent component analysis," *Journal of Computer Assisted Tomography*, vol. 23, no. 2, pp. 265–271, 1999.

[31] L. Boney, A. H. Tewfik, and K. N. Hamdy, "Digital watermarks for audio signals," in *Proc. IEEE Int. Conf. on Multimedia Computing and Systems (ICMCS'96)*, Hiroshima, Japan, June 1996, pp. 473–480.

[32] S. Bounkong, B. Toch, D. Saad, and D. Lowe, "ICA for watermarking digital images," *Journal of Machine Learning Research*, vol. 4, pp. 1471–1498, Dec. 2003.

[33] J. T. Brassil, S. Low, and N. F. Maxemchuk, "Copyright protection for the electronic distribution of text documents," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1181–1196, July 1999.

[34] G. Burel, "Blind separation of sources: A nonlinear neural algorithm," *Neural Network*, vol. 5, pp. 937–947, 1992.

[35] J. F. Cardoso and B. H. Laheld, "Equivariant adaptive source separation," *IEEE Trans. Signal Processing*, vol. 44, no. 12, pp. 3017–3030, 1996.

[36] J. F. Cardoso, "Source separation using higher order moments," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP'89)*, Glasgow, UK, 1989, pp. 2109–2112.

[37] ——, "Eigen-structure of the fourth-order cumulant tensor with application to the blind source separation problem," in *Proc. of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP'90)*, NM, USA, 1990, pp. 2655–2658.

[38] ——, "Super-symmetric decomposition of the fourth-order cumulant tensor. blind identification of more sources than sensors," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'91)*, 1991, pp. 3109–3112.

[39] ——, "Infomax and maximum likelihood for source separation," *IEEE Letters on Signal Processing*, vol. 4, pp. 112–114, 1997.

[40] ——, "Dependence, correlation and gaussianity in independent component analysis," *Journal of Machine Learning Research*, vol. 4, pp. 1177–1203, Dec. 2003.

[41] B. Chen and G. W. Wornell, "Quantization index modulation: a class of provably good methods for digital watermarking and information embedding," *IEEE Trans. Inform. Theory*, vol. 47, no. 4, pp. 1423–1443, May 2001.

[42] P.-C. Chen, "Modeling, performance analysis, and applications of digital image watermarking systems," Master's thesis, National Tsing Hua University, May 1999.

[43] Y. W. Chen, X. Y. Zeng, and Z. Nakao, "Blind signal separation by an evolutionary neural network with higher-order statistics," in *Proc. IEEE International Workshop on Knowledge-Based Intelligent Engineering Systems and Allied Technologies*, vol. 2, Brighton, UK, Aug. 2000, pp. 566–571.

[44] C. Choi, G.-J. Jang, Y. Lee, and S. R. Kim, "Adaptive cross-channel interference cancellation on blind source separation outputs," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 857–864.

[45] A. Cichocki and S.-I. Amari, *Adaptive blind signal and image processing.* John Wiley & Sons Ltd, 2002.

[46] A. Cichocki, R. Bogner, L. Moszczynski, and K. Pope, "Modified Herault-Jutten algorithms for blind separation of sources," *Digital Signal Processing*, vol. 7, pp. 80–93, 1997.

[47] A. Cichocki *et al.* (2003) Icalab toolboxes. [Online]. Available: http://www.bsp.brain.riken.jp/ICALAB/

[48] A. Cichocki, R. Thwonmal, and S.-I. Amari, "Sequential blind signal extraction in order specified by stochastic properties," *Electronics Letters*, vol. 33, no. 1, pp. 64–65, 1997.

[49] A. Cichocki and R. Unbehauen, "Robust neural networks with on-line learning for blind identification and blind separation of sources," *IEEE Trans. Circuits Syst. I*, vol. 43, no. 11, pp. 894–906, Nov. 1996.

[50] P. Comon, "Separation of stochastic processes," in *Proc. Workshop on Higher-Order Spectral Analysis*, Colorado, US, 1989, pp. 174–179.

[51] ——, "Independent component analysis, a new concept?" *Signal Processing*, vol. 36, no. 3, pp. 287–314, 1994.

[52] P. Comon and B. Mourrain, "Decomposition of quantics in sums of powers of linear forms," *Signal Processing*, vol. 53, no. 2, pp. 93–107, 1996.

[53] D. Cook, A. Buja, and J. Cabrera, "Projection pursuit indexes based on orthonormal function expansions," *Journal of Computational and Graphical Statistics*, vol. 2, no. 3, pp. 225–250, 1993.

[54] I. J. Cox, M. L. Miller, and J. A. Bloom, *Digital Watermarking*, 1st ed., E. Fox, Ed. Morgan Kaufmann, Oct. 2001.

[55] I. J. Cox, J. Kilian, T. Leighton, and T. G. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Trans. Image Processing*, vol. 6, no. 12, pp. 1673–1687, 1997.

[56] S. Craver, N. D. Memon, B.-L. Yeo, and M. M. Yeung, "Can invisible watermarks resolve rightful ownerships?" in *Storage and Retrieval for Image and Video Databases (SPIE)*, 1997, pp. 310–321.

[57] S. A. Cruces and A. Cichocki, "Combining blind source extraction with joint approximate diagonalization: Thin algorithms for ICA," in *Proc. Int. Symposium on Independent Component Analysis and Blind Signal Separation (ICA'03)*, Nara, Japan, Apr. 2003, pp. 463–468.

[58] S. A. Cruces, L. R. Castedo, and A. Cichocki, "Robust blind source separation algorithms using cumulants," *Neurocomputing*, vol. 49, pp. 87–118, Dec. 2002.

[59] G. Darmois, "Analyse des liaisons de probabilité," in *Proc. Int. Statistics Conferences*, vol. III A, Washington DC, USA, 1951, pp. 231–232.

[60] G. Deco and W. Brauer, "Nonlinear higher-order statistical decorrelation by volume-conserving neural architectures," *Neural Network*, vol. 8, pp. 525–535, 1995.

[61] G. Deco and D. Obradovic, "Linear redundancy reduction learning," *Neural Networks*, vol. 8, no. 5, pp. 751–755, 1995.

[62] ——, *An Information-Theoretic Approach to Neural Computing.* NJ, USA: Springer Verlag, 1996.

[63] N. Delfosse and P. Loubaton, "Adaptive blind separation of independent sources: A deflation approach," *Signal Processing*, vol. 45, pp. 59–83, 1995.

[64] G. Doerr and J.-L. Dugelay, "A guide tour of video watermarking," *Signal processing: Image Communication*, vol. 18, no. 4, pp. 263–282, 2003.

[65] B. A. Draper, K. Baek, M. S. Bartlett, and J. R. Beveridge, "Recognizing faces with PCA and ICA," *Computer Vision and Image Understanding, Special Issue on Face Recognition*, vol. 91, no. 1, pp. 115–137, July 2003.

[66] J. Eriksson and V. Koivunen, "Blind identifiability of class of nonlinear instantaneous ica models," in *Proc. of the XI European Signal Proc. Conf. (EUSIPCO'02)*, vol. 2, Toulouse, France, Sept. 2002, pp. 7–10.

[67] D. J. Field, "What is the goal of sensory coding?" *Neural Computation*, vol. 6, pp. 559–601, 1994.

[68] J. Fisher and J. Principe, "Entropy manipulation of arbitrary nonlinear mappings," in *Proc. IEEE Int. Workshop on Neural Networks for Signal Processing VII*, 1997, pp. 14–23.

[69] J. François and A. Souloumiac, "Blind beamforming for non gaussian signals," *IEE Proceedings–F*, vol. 140, no. 6, pp. 362–370, 1993.

[70] J. H. Friedman, "Exploratory projection pursuit," *Journal of the American Statistical Association*, vol. 82, pp. 249–266, 1987.

[71] J. H. Friedman and J. W. Tukey, "A projection pursuit algorithm for exploratory data analysis," *IEEE Trans. Comput.*, vol. C–23, no. 9, pp. 881–890, 1974.

[72] M. Gaeta and J. L. Lacoume, "Source separation without prior knowledge: the maximum likelihood solution," in *Proc. of the European Signal Processing Conference (EUSIPCO'90)*, 1990, pp. 621–624.

[73] P. G. Georgiev and A. Cichocki, "Robust independent component analysis via time-delayed cumulant functions," *IEICE Transactions on Fundamentals*, vol. E86-A, no. 3, pp. 573–579, 2003.

[74] K. A. Glass, G. A. Frishkoff, R. M. Frank, C. Davey, J. Dien, A. D. Malony, and D. M. Tucker, "A framework for evaluating ICA methods of artifact removal from multichannel EEG," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 1033–1040.

[75] F. J. Gonzalez-Serrano, H. Y. Molina-Bulla, and J. J. Murillo-Fuentes, "Independent component analysis applied to digital image watermarking," in *Proc. IEEE Int. Conference on Acoustics, Speech, and Signal Processing, 2001 (ICASSP'01)*, vol. 3, 2001, pp. 1997–2000.

[76] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*, 1st ed. Boston, MA: PWS Publishing, 1995.

[77] H. H. Harman, *Modern Factor Analysis*, 2nd ed. University of Chicago Press, 1967.

[78] S. Harmeling, A. Ziehe, M. Kawanabe, and K. R. Müller, "Kernel-based nonlinear blind source separation," *Neural Computation*, vol. 15, no. 5, pp. 1089–1124, 2003.

[79] F. Hartung and M. Kutter, "Multimedia watermarking techniques," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1079–1107, July 1999.

[80] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Prentice Hall, 1998.

[81] M. Herrmann and H. H. Yang, "Perspectives and limitations of self-organising maps in blind separation of source signals," in *Proc. Int. Conf.on Neural Information Processing (ICONIP'96)*, Hong Kong, Sept. 1996, pp. 1211–1216.

[82] G. Hinton, M. Welling, Y. Teh, and S. Osindero, "A new view of ICA," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'01)*, San Diego, CA, USA, 2001, pp. 746–751.

[83] P. O. Hoyer and A. Hyvärinen, "Independent component analysis applied to feature extraction from colour and stereo images," *Network: Computation in Neural Systems*, vol. 11, no. 3, pp. 191–210, 2000.

[84] C.-T. Hsu and J.-L. Wu, "Digital watermarking for video," in *Proc. IEEE Int. Conf. on Digital Signal Proccessing (DSP'97)*, Santorini, Greece, July 1997, pp. 217–220.

[85] P. J. Huber, "Projection pursuit," *The Annals of Statistics*, vol. 13, no. 2, pp. 435–475, 1985.

[86] A. Hyvärinen, "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Trans. Neural Networks*, vol. 10, no. 3, pp. 626–634, May 1999.

[87] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis.* John Wiley & Sons Ltd, 2001.

[88] A. Hyvärinen and P. Pajunen, "Nonlinear independent component analysis: Existence and uniqueness results," *Neural Networks*, vol. 12, no. 3, pp. 429–439, 1999.

[89] 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, Apr. 1997, pp. 3917–3920.

[90] ——, "New approximations of differential entropy for independent component analysis and projection pursuit," in *Proc. of the 1997 conference on Advances in Neural Information Processing Systems 10*, 1998, pp. 273–279.

[91] ——, "Sparse code shrinkage: Denoising of nongaussian data by maximum likelihood estimation," *Neural Computation*, vol. 11, no. 7, pp. 1739–1768, 1999.

[92] ——, "Survey on independent component analysis," *Neural Computing Surveys*, vol. 2, pp. 94–128, 1999.

[93] A. Hyvärinen and E. Oja, "A fast fixed-point algorithm for independent component analysis," *Neural Computation*, vol. 9, no. 7, pp. 1483–1492, 1997.

[94] ——, "Independent component analysis by general nonlinear hebbian-like learning rules," *Signal Processing*, vol. 64, no. 3, pp. 301–313, 1998.

[95] ——, "Independent component analysis: Algorithms and applications," *Neural Networks*, vol. 13, pp. 411–430, 2000.

[96] A. Hyvärinen, E. Oja, P. Hoyer, and J. Hurri, "Image feature extraction by sparse coding and independent component analysis," in *Proc. IEEE Int. Conf. on Pattern Recognition (ICPR'98)*, vol. 2, Brisbane, Australia, Aug. 1998, pp. 1268–1273.

[97] A. Ilin and A. Honkela, "Post-nonlinear independent component analysis by variational bayesian learning," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 766–773.

[98] M. Inki, "ICA features of image data in one, two and three dimensions," in *Proc. Int. Symposium on Independent Component Analysis and Blind Signal Separation (ICA'03)*, Nara, Japan, Apr. 2003, pp. 861–866.

[99] A. K. Jain and U. Uludag, "Hiding biometric data," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 25, no. 11, pp. 1494–1498, Nov. 2003.

[100] G.-J. Jang and T.-W. Lee, "A maximum likelihood approach to single-channel source separation," *Journal of Machine Learning Research*, vol. 4, pp. 1365–1392, Dec. 2003.

[101] D. Jiang, H. K. Lee, and Y. H. Suh, "BSS: A new approach for watermark attack," in *IEEE 4th International Symposium on Multimedia Software Engineering (MSE'2002)*, California, USA, Dec. 2002, pp. 182–187.

[102] I. Jolliffe, *Principal Component Analysis.* Springer Verlag, 1986.

[103] M. Jones and R. Sibson, "What is projection pursuit?" *Journal of the Royal Statistical Society, ser. A*, vol. 150, pp. 1–36, 1987.

[104] A. Jung and A. Kaiser, "Considering temporal structures in independent component analysis," in *Proc. Int. Symposium on Independent Component Analysis and Blind Signal Separation (ICA'03)*, Nara, Japan, Apr. 2003, pp. 95–100.

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

[106] C. Jutten and J. Karhunen, "Advances in nonlinear blind source separation," in *Proc. Int. Symposium on Independent Component Analysis and Blind Signal Separation (ICA'03)*, Nara, Japan, Apr. 2003, pp. 245–256.

[107] C. Jutten and A. Taleb, "Source separation: From dusk till dawn," in *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA'00)*, Finland, 2000, pp. 15–26.

[108] J. Karhunen, A. Hyvärinen, R. Vigario, and E. Oja, "Applications of neural blind separation to signal and image processing," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'97)*, Munich, Germany, 1997, pp. 131–134.

[109] J. Karhunen and J. Joutsensalo, "Representation and separation of signals using nonlinear PCA type learning," *Neural Networks*, vol. 7, no. 1, pp. 113–127, 1994.

[110] J. Karhunen, E. Oja, L. Wang, R. Vigário, and J. Joutsensalo, "A class of neural networks for independent component analysis," *IEEE Trans. Neural Networks*, vol. 8, no. 3, pp. 486–504, 1997.

[111] J. Karvonen and M. Similä, "Independent component analysis for sea ice SAR image classification," in *Proc. IEEE Int. Geoscience and Remote Sensing Symposium (IGARSS'01)*, vol. 3, Sydney, Australia, July 2001, pp. 1255–1257.

[112] M. Kawamoto, K. Matsuoka, and M. Oya, "Blind separation of sources using temporal correlation of the observed signals," *IEICE Trans. Fundamentals*, vol. E80-A, no. 4, pp. 695–704, 1997.

[113] I. R. Keck, F. J. Theis, P. Gruber, E. W. Lang, K. Specht, and C. G. Puntonet, "3D spatial analysis of fMRI data on a word perception task," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 977–984.

[114] M. Kendall, *Multivariate Analysis.* Charles Grinffin&Co., 1975.

[115] M. Kermit and O. Tomic, "Independent component analysis applied on gas sensor array measurement data," *IEEE Sensors J.*, vol. 3, no. 2, pp. 218–228, Apr. 2003.

[116] S. Kishk and B. Javidi, "3D object watermarking by a 3D hidden object," *Optics Express*, vol. 11, no. 8, pp. 874–888, Apr. 2003.

[117] K. Kiviluoto and E. Oja, "Independent component analysis for parallel financial time series," in *Proc. Int. Conf. on Neural Information Processing (ICONIP'98)*, Tokyo, Japan, 1998, pp. 895–898.

[118] E. Koch and J. Zhao, "Towards robust and hidden image copyright labeling," in *Proc. IEEE Int. Workshop on Nonlinear Signal and Image Processing*, Marmaras, Greece, June 1995, pp. 452–455.

[119] M. J. Korenberg and I. W. Hunter, "The identification of nonlinear biological systems: LNL cascade models," *Biol. Cybernetics*, vol. 55, pp. 125–134, 1996.

[120] E. Kreyszig, *Advanced Engineering Mathematics*, 8th ed. John Wiley & Sons Ltd, 1999.

[121] D. Kundur and D. Hatzinakos, "Digital watermarking using multiresolution wavelet decomposition," in *Int. Conference on Acoustic, Speech and Signal Processing (ICASSP'98)*, vol. 5, Washington, USA, May 1998, pp. 2969–2972.

[122] M. Kutter, S. K. Bhattacharjee, and T. Ebrahimi, "Towards second generation watermarking schemes," in *6th International Conference on Image Processing (ICIP'99)*, vol. 1, Kobe, Japan, Oct. 1999, pp. 25–28.

[123] M. Kutter and F. A. P. Petitcolas, "A fair benchmark for image watermarking systems," in *Proc. SPIE Security and Watermarking of Multimedia Contents*, vol. 3657, San Jose, CA, USA, Jan. 1999, pp. 226–239.

[124] M. Kutter, F. Jordan, and F. Bossen, "Digital signature of color images using amplitude modulation," in *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases*, vol. 2952, San Jose, USA, Feb. 1997, pp. 518–526.

[125] G. C. Langelaar, R. L. Lagendijk, and J. Biemond, "Realtime labeling methods for MPEG compressed video," in *Proc. 18th Symp. Information Theory in the Benelux*, Veldhoven, The Netherlands, May 1996, pp. 25–32.

[126] G. C. Langelaar, I. Setyawan, and R. L. Lagendijk, "Watermarking digital image and video data: A state-of-the-art overview," *IEEE Signal Processing Mag.*, vol. 17, no. 5, pp. 20–46, 2000.

[127] G. C. Langelaar, R. L. Lagendijk, and J. Biemond, "Robust labeling methods for copy protection of images," in *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases*, vol. 3022, San Jose, USA, Feb. 1997, pp. 289–309.

[128] H. Lappalainen, "Nonlinear independent component analysis using ensemble learning: Theory," in *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA'00)*, Finland, 2000, pp. 251–256.

[129] H. Lappalainen, X. Giannakopoulos, A. Honkela, and J. Karhunen, "Nonlinear independent component analysis using ensemble learning: Experiments and discussion," in *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA'00)*, Finland, 2000, pp. 351–356.

[130] A. Larue, C. Jutten, and S. Hosseini, "Markovian source separation in post-nonlinear mixtures," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 702–709.

[131] L. D. Lathauwer, P. Comon, B. D. Moor, and J. Vandewalle, "Higher-order power method, application in independent component analysis," in *Proc. Int. Symp. on Nonlinear Theory and its Applications (NOLTA'95)*, Nevada, USA, 1995, pp. 10–14.

[132] S.-J. Lee and S.-H. Jung, "A survey of watermarking techniques applied to multimedia," in *Proc. IEEE Int. Symposium on Industrial Electronics, 2001 (ISIE'01)*, vol. 1, 2001, pp. 272–277.

[133] T.-W. Lee, *Independent Component Analysis : Theory and Applications.* Kluwer Academic Publishers, 1998.

[134] T.-W. Lee, M. Girolami, A. J. Bell, and T. J. Sejnowski, "A unifying information-theoretic framework for independent component analysis," *Computers and Mathematics with Applications*, vol. 31, no. 11, pp. 1–12, 2000.

[135] T.-W. Lee, M. Girolami, and T. J. Sejnowski, "Independent component analysis using an extended infomax algorithm for mixed sub-gaussian and super-gaussian sources," *Neural Computation*, vol. 11, no. 2, pp. 417–441, 1999.

[136] T.-W. Lee and M. S. Lewicki, "Unsupervised image classification, segmentation, and enhancement using ICA mixture models," *IEEE Trans. Image Processing*, vol. 11, no. 3, pp. 270–279, Mar. 2002.

[137] J. Liu, X. Zhang, M. Najar, and M. A. Lagunas, "A robust digital watermarking scheme based on ICA," in *Proc. IEEE Int. Conf. on Neural Networks and Signal Processing*, vol. 2, Nanjing, China, 2003, pp. 1481–1484.

[138] S. Low and N. F. Maxemchuk, "Performance comparison of two text marking and detection methods," *IEEE J. Select. Areas Commun.*, vol. 16, no. 4, pp. 561–572, May 1998.

[139] S. Malaroiu, K. Kiviluoto, and E. Oja, "Time series prediction with independent component analysis," in *Proc. Int. Conf. on Advanced Investment Technology*, Gold Coast, Australia, 2000.

[140] S. G. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, pp. 674–693, 1989.

[141] D. Martinez and A. Bray, "Nonlinear blind source separation using kernels," *IEEE Trans. Neural Networks*, vol. 14, no. 1, pp. 228–235, Jan. 2003.

[142] N. F. Maxemchuk and S. Low, "Marking text documents," in *Proc. IEEE Int. Conf. on Image Processing (ICIP'97)*, vol. 3, CA, USA, Oct. 1997, p. 13.

[143] M. McKeown, S. Makeig, S. Brown, T.-P. Jung, S. Kindermann, A. J. Bell, V. Iragui, and T. Sejnowski, "Blind separation of functional magnetic resonance imaging (fMRI) data," *Human Brain Mapping*, vol. 6, no. 5-6, pp. 368–372, 1998.

[144] P. Meerwald, "Digital watermarking in the wavelet transform domain," Master's thesis, University Salzburg, Austria, Jan. 2001.

[145] L. Molgedey and G. Schuster, "Separation of a mixture of independent signals using time delayed correlations," *Physical Review Letters*, vol. 72, no. 23.

[146] F. C. Morabito, "Independent component analysis and neural approaches to the extraction of features from NDT/NDE," in *Proc. Int. Joint Conference on Neural Network (IJCNN'99)*, vol. 6, Washington, USA, July 1999, pp. 4021–4026.

[147] E. Moreau and O. Macchi, "High order contrasts for self-adaptive source separation," *International Journal of Adaptive Control and Signal Processing*, vol. 10, no. 1, pp. 19–46, 1996.

[148] P. Moulin and R. Koetter, "Data-hiding codes," *Proceedings of the IEEE*, vol. 93, no. 12, pp. 2083–2126, Dec. 2005.

[149] T. V. Nguyen, J. C. Patra, and E.-L. Ang, "Blind image extraction from nonlinear mixtures using MLP-based ICA," in *Proc. IEEE Int. Conf. on Multimedia and Expo (ICME'03)*, vol. 1, Baltimore, USA, July 2003, pp. 249–252.

[150] T. V. Nguyen, J. C. Patra, and A. Das, "A geometric approach to post nonlinear mixture in blind source separation," in *Proc. IEEE Int. Conf. on Communications Systems (ICCS'04)*, vol. 1, Singapore, Sept. 2004, pp. 260–264.

[151] T. V. Nguyen and J. C. Patra, "WMica: An ICA-based digital watermarking technique," in *Proc. IEEE International Symposium on Signal Processing and Information Technology (ISSPIT'04)*, vol. 1, Rome, Italy, Dec. 2004, pp. 306–309.

[152] ——, "Digital image watermarking using independent component analysis," in *Proc. IEEE of the 5th Pacific Rim Conference on Multimedia (PCM'04)*, Japan, Nov. 2004, pp. 364–371.

[153] T. V. Nguyen, J. C. Patra, and N. S. Chaudhari, "A novel digital image watermarking scheme using blind source separation," in *Proc. IEEE Int. Conf. on Image Processing (ICIP'04)*, vol. 1, Singapore, Oct. 2004, pp. 2649–2652.

[154] T. V. Nguyen, J. C. Patra, and A. Das, "A post nonlinear geometric algorithm for independent component analysis," *Digital Signal Processing*, vol. 15, no. 3, pp. 276–294, May 2005.

[155] T. V. Nguyen, J. C. Patra, A. Das, and G. S. Ng, "Post nonlinear blind source separation by geometric linearization," in *Proc. of the International Joint Conference on Neural Networks (IJCNN'05)*, vol. 1, Montreal, Canada, Aug. 2005, pp. 244–249.

[156] D. Obradovic and G. Deco, "Information maximization and independent component analysis: Is there a difference?" *Neural Computation*, vol. 10, pp. 2085–2101, 1998.

[157] R. Ohbuchi, H. Masuada, and M. Aono, "Watermarking three-dimensional polygonal models through geometric and topological modifications," *IEEE J. Select. Areas Commun.*, vol. 16, no. 4, pp. 551–560, 1998.

[158] E. Oja, "The nonlinear PCA learning rule in independent component analysis," *Neurocomputing*, vol. 17, no. 1, pp. 25–46, 1997.

[159] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, pp. 607–609, 1996.

[160] J. J. ÓRuanaidh and T. Pun, "Rotation, scale and translation invariant spread spectrum digital image watermarking," *Signal Processing*, vol. 66, no. 3, pp. 303–317, 1998.

[161] P. Pajunen, A. Hyvärinen, and J. Karhunen, "Nonlinear blind source separation by self-organizing maps," in *Proc. Int. Conf. on Neural Information Processing (ICONIP'96)*, vol. 2, Hong Kong, China, 1996, pp. 1207–1210.

[162] P. Pajunen and J. Karhunen, "A maximum likelihood approach to nonlinear blind source separation," in *Proc. Int. Conf. on Artificial Neural Networks (ICANN'97)*, Lausanne, Switzerland, 1997, pp. 541–546.

[163] A. Papoulis, *Probability, random variables, and stochastic processes.* NY, USA: McGraw-Hill, 1984.

[164] A. Parashiv-Ionescu, C. Jutten, A. Ionescu, A. Chovet, and A. Rusu, "High performance magnetic field smart sensor arrays with source separation," in *Proc. of the 1st Int. Conf. on Modeling and Simulation of Microsystems (MSM'98)*, CA, USA, Apr. 1998, pp. 666–671.

[165] L. Parra, "Symplectic nonlinear component analysis," in *Advances in Neural Information Processing Systems*, vol. 8. Cambridge, Mass: MIT Press, 1996, pp. 437–443.

[166] L. Parra, G. Deco, and S. Miesbach, "Redundancy reduction with information-preserving nonlinear maps," *Network*, vol. 6, pp. 61–72, 1995.

[167] ——, "Statistical independence and novelty detection with information-preserving nonlinear maps," *Neural Computation*, vol. 8, pp. 260–269, 1996.

[168] J. C. Patra, K. K. Ang, and E. L. Ang, "Hierarchical multiple image watermarking for image authentication and ownership verification," in *Proc. IEEE Int. Conf. on Image Processing (ICIP'04)*, Singapore, Oct. 2004, pp. 2661–2664.

[169] J. C. Patra, C. H. Tan, and E. L. Ang, "Bilevel image watermarking for image authentication surviving JPEG lossy compression," in *Proc. IEEE Int. Conf. on Image Processing (ICIP'04)*, Singapore, Oct. 2004, pp. 713–716.

[170] K. Pawelzik, K.-R. Müller, and J. Kohlmorgen, "Prediction of mixtures," in *Proc. Int. Conf. on Artificial Neural Networks (ICANN'96)*, Bochum, Germany, 1996, pp. 127–132.

[171] H. Peng, Z. Chi, and W. Siu, "A semi-parametric hybrid neural model for nonlinear blind signal separation," *Int. Journal of Neural Systems*, vol. 10, no. 2, pp. 79–94, 2000.

[172] F. Petitcolas, "Watermarking schemes evaluation," *IEEE Signal Processing Mag.*, vol. 17, no. 5, pp. 58–64, 2000.

[173] D. T. Pham, "Blind separation of instantaneous mixture sources via an independent component analysis," *IEEE Trans. Signal Processing*, vol. 44, no. 11, pp. 2768–2779, 1996.

[174] ——, "Mutual information approach to blind separation of stationary sources," *IEEE Trans. Inform. Theory*, vol. 48, no. 7, pp. 1935–1946, July 2002.

[175] D. T. Pham and J. F. Cardoso, "Blind separation of instantaneous mixtures of non stationary sources," *IEEE Trans. Signal Processing*, vol. 49, no. 9, pp. 1837–1848, Sept. 2001.

[176] D. T. Pham and P. Garat, "Blind separation of mixture of independent sources through a quasi-maximum likelihood approach," *IEEE Trans. Signal Processing*, vol. 45, no. 7, pp. 1712–1725, July 1997.

[177] D. T. Pham, P. Garrat, , and C. Jutten, "Separation of a mixture of independent sources through a maximum likelihood approach," in *Proc. of the European Signal Processing Conference (EUSIPCO'92)*, 1992, pp. 771–774.

[178] V. S. Poblador, E. M. Moreno, and J. S. Casals, "ICA as a preprocessing technique for classification," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 1165–1172.

[179] C. I. Podilchuk and E. J. Delp, "Digital watermarking: algorithms and applications," *IEEE Signal Processing Mag.*, vol. 18, no. 4, pp. 33–46, July 2001.

[180] A. Polonsky and M. Zibulevsky, "MEG/EEG source localization using spatio-temporal sparse representations," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 1001–1008.

[181] S. Prakriya and D. Hatzinakos, "Blind identification of LTI-ZMNL-LTI nonlinear channel models," *IEEE Trans. Signal Processing*, vol. 43, no. 12, pp. 3007–3013, Dec. 1995.

[182] S. Punnoose, X. Zhu, and A. K. Nandi, "Layered space frequency equalisation for MIMO-MC-CDMA systems in frequency selective fading channels," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 1181–1188.

[183] C. Puntonet, M. Alvarez, A. Prieto, and B. Prieto, "Separation of sources in a class of post-nonlinear mixtures,," in *Proc. 6th European Symp. on Artificial Neural Networks (ESANN98)*, Apr. 1998, pp. 321–326.

[184] C. G. Puntonet, J. M. Górriz, M. Salmerón, and S. H. Mellado, "Theoretical method for solving BSS-ICA using SVM," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 256–262.

[185] C. G. Puntonet and A. Prieto, Eds., *Proceeding of the fifth International Conference on Independent Component Analysis and Blind Signal Separation.* Granada, Spain: Springer-Verlag, Germany, Sept. 2004.

[186] Y. Qi, D. Doermann, and D. DeMenthon, "Hybrid independent component analysis and support vector machine learning scheme for face detection," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'01)*, vol. 3, Utah, USA, May 2001, pp. 1481–1484.

[187] C. Rey and J.-L. Dugelay, "A survey of watermarking algorithms for image authentication," *EURASIP Journal on Applied Signal Processing*, vol. 2002, no. 6, pp. 613–621, June 2002.

[188] T. Ristaniemi, K. Raju, J. Karhunen, and E. Oja, "Inter-cell interference cancellation in CDMA array systems by independent component analysis," in *Proc. Int. Symposium on Independent Component Analysis and Blind Signal Separation (ICA'03)*, Nara, Japan, Apr. 2003, pp. 739–744.

[189] F. Rojas, C. G. Puntonet, M. R. Alvarez, I. Rojas, and R. M. Clemente, "Blind source separation in post-nonlinear mixtures using competitive learning, simulated annealing, and a genetic algorithm," *IEEE Trans. Syst., Man, Cybern. C*, vol. 34, no. 4, pp. 407–416, Nov. 2004.

[190] H. S. Sahambi and K. Khorasani, "A neural-network appearance-based 3-D object recognition using independent component analysis," *IEEE Trans. Neural Networks*, vol. 14, no. 1, pp. 138–149, Jan. 2003.

[191] I. Schiessl, M. Stetter, J. W. W. Mayhew, N. McLoughlin, J. S. Lund, and K. Obermayer, "Blind signal separation from optical imaging recordings with extended spatial decorrelation," *IEEE Trans. Biomed. Eng.*, vol. 47, no. 5, pp. 573–577, 2000.

[192] J. Schmidhuber, M. Eldracher, and B. Foltin, "Semilinear predictability minimization produces well-known feature detectors," *Neural Computation*, vol. 8, pp. 773–786, 1996.

[193] A. Sequeira and D. Kundur, "Communication and information theory in watermarking: A survey," 2001.

[194] C. Serviere and D.-T. Pham, "A novel method for permutation correction in frequency-domain in blind separation of speech mixtures," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 807–815.

[195] M. Shen, X. Zhang, L. Sun, P. J. Beadle, and F. H. Y. Chan, "A method for digital image watermarking using ICA," in *Proc. Int. Symposium on Independent Component Analysis and Blind Signal Separation (ICA'03)*, Nara, Japan, Apr. 2003, pp. 209–214.

[196] M. Solazzi, F. Piazza, and A. Uncini, "Nonlinear blind source separation by spline neural networks," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'01)*, vol. 5, Utah, USA, May 2001, pp. 2781–2784.

[197] I. Stainvas and D. Lowe, "A generative model for separating illumination and reflectance from images," *Journal of Machine Learning Research*, vol. 4, pp. 1499–1519, Dec. 2003.

[198] P. F. Stetson, "Independent component analysis of pulse oximetry signals based on derivative skew," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 1072–1078.

[199] J. Sun, J. Liu, and M. A. Lagunas, "An blind video watermarking scheme based on ICA," in *Proc. IEEE Int. Conf. on Neural Networks and Signal Processing*, vol. 2, Nanjing, China, 2003, pp. 1127–1130.

[200] M. D. Swanson, B. Zhu, A. H. Tewfik, and L. Boney, "Robust audio watermarking using perceptual masking," *Signal Processing*, vol. 66, no. 3, pp. 337–355, 1998.

[201] A. Taleb, "A generic framework for blind source separation in structured nonlinear models," *IEEE Trans. Signal Processing*, vol. 50, no. 8, pp. 1819–1830, 2002.

[202] A. Taleb and C. Jutten, "Source separation in post-nonlinear mixtures," *IEEE Trans. Signal Processing*, vol. 47, no. 10, pp. 2807–2820, Oct. 1999.

[203] Y. Tan and J. Wang, "Nonlinear blind source separation using higher order statistics and a genetic algorithm," *IEEE Trans. Evol. Comput.*, vol. 5, no. 6, pp. 600–612, Dec. 2001.

[204] Y. Tan, J. Wang, and J. M. Zurada, "Nonlinear blind source separation using a radial basis function network," *IEEE Trans. Neural Networks*, vol. 12, no. 1, pp. 124–134, Jan. 2001.

[205] Y. W. Teh, M. Welling, S. Osindero, and G. E. Hinton, "Energy-based models for sparse overcomplete representations," *Journal of Machine Learning Research*, vol. 4, pp. 1235–1260, Dec. 2003.

[206] F. J. Theis, A. Jung, E. W. Lang, and C. G. Puntonet, "A theoretic model for linear geometric ICA," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'01)*, California, USA, Dec. 2001, pp. 349–354.

[207] F. J. Theis and S.-I. Amari, "Postnonlinear overcomplete blind source separation using sparse sources," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 718–725.

[208] F. Theis, A. Jung, C. Puntonet, , and E. Lang, "Linear geometric ICA: Fundamentals and algorithms," *Neural Computation*, vol. 15, pp. 419–439, 2003.

[209] F. Theis, E. Lang, and C. Puntonet, "A geometric algorithm for overcomplete linear ICA," *Neurocomputing*, vol. 56, pp. 381–398, 2004.

[210] H.-L. N. Thi and C. Jutten, "Blind source separation for convolutive mixtures," *Signal Processing*, vol. 45, pp. 209–229, 1995.

[211] B. Toch, D. Lowe, and D. Saad, "Watermarking of audio signals using independent component analysis," in *Proc. Int. Conf. on WEB Delivering of Music (WEDELMUSIC'03)*, Leeds, UK, Sept. 2003, pp. 71–74.

[212] L. Tong, V. C. Soon, R. Liu, and Y. Huang, "AMUSE: A new blind identification algorithm," in *Proc. IEEE Int. Symp. on Circuits and Systems (ISCAS'90)*, LA, USA, 1990, pp. 1784–1786.

[213] J. H. van Hateren and D. L. Ruderman, "Independent component analysis of natural image sequences yields spatiotemporal filters similar to simple cells in

primary visual cortex," *Proc. Royal Society London, Series B*, vol. 265, pp. 2315–2320, 1998.

[214] J. H. van Hateren and A. van der Schaaf, "Independent component filters of natural images compared with simple cells in primary visual cortex," *Proc. Royal Society London, Series B*, vol. 265, pp. 359–366, 1998.

[215] R. Vigário, "Extraction of ocular artifacts from EEG using independent component analysis," *Electroenceph. Clin. Neurophysiol*, vol. 103, no. 3, pp. 395–404, 1997.

[216] R. Vigário, J. Särelä, V. Jousmäki, M. Hämäläinen, and E. Oja, "Independent component approach to the analysis of EEG and MEG recordings," *IEEE Trans. Biomed. Eng.*, vol. 47, no. 5, pp. 589–593, 2000.

[217] R. Vigário, J. Särelä, V. Jousmäki, and E. Oja, "Independent component analysis in decomposition of auditory and somatosensory evoked fields," in *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA'99)*, Aussois, France, 1999, pp. 167–172.

[218] S. Voloshynovskiy, A. Herrigel, N. Baumgaertner, and T. Pun, "A stochastic approach to content adaptive digital imagewatermarking," in *Proc. of International Workshop on Information Hiding*, Dresden. Germany, Oct. 1999, pp. 212–236.

[219] S. Voloshynovskiy, S. Pereira, T. Pun, J. J. Eggers, and J. K. Su, "Attacks on digital watermarks: classification, estimation based attacks, and benchmarks," *IEEE Commun. Mag.*, vol. 39, no. 8, pp. 118–126, Aug. 2001.

[220] H.-J. Wang, P.-C. Su, and C.-C. J. Kuo, "Wavelet-based digital image watermarking," *Optics Express*, vol. 3, no. 12, pp. 491–496, Dec. 1998.

[221] J. Wang and C.-I. Chang, "Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis," *IEEE Trans. Geosci. Remote Sensing*, vol. 44, no. 6, pp. 1586–1600, 2006.

[222] L. Wang and J. Karhunen, "A unified neural bigradient algorithm for robust PCA and MCA," *International Journal of Neural Systems*, vol. 7, no. 1, pp. 53–67, 1996.

[223] M. Wu and B. Liu, "Data hiding in binary image for authentication and annotation," *IEEE Trans. Multimedia*, vol. 6, no. 4, pp. 528–538, Aug. 2004.

[224] G. Wubbeler, A. Ziehe, B. Mackert, K. Müller, L. Trahms, and G. Curio, "Independent component analysis of noninvasively recorded cortical magnetic dc-field in humans," *IEEE Trans. Biomed. Eng.*, vol. 47, no. 5, pp. 594–599, 2000.

[225] L. Xu, "Temporal BYY learning for state space approach, hidden markov model, and blind source separation," *IEEE Trans. Signal Processing*, vol. 48, no. 7, pp. 2132–2144, 2000.

[226] H. Yang, S.-I. Amari, and A. Cichocki, "Information-theoretic approach to blind separation of sources in nonlinear mixture," *Signal Processing*, vol. 64, pp. 291–300, 1998.

[227] H. Yang, J. C. Patra, and C. W. Chan, "An artificial neural network-based scheme for robust watermarking of audio signals," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICISSP'02)*, Florida, USA, May 2002, pp. 1029–1032.

[228] M. Yoshioka and S. Omatu, "Signal separation method using genetic algorithms," in *Proc. of IEEE Int. Joint Conf. on Neural Networks (IJCNN'98)*, vol. 2, AK, USA, 1998, pp. 909–912.

[229] D. Yu and F. Sattar, "A new blind image watermarking technique based on independent component analysis," in *Springer-Verlag Computer Science Lecture Series*, vol. 2613, Jan. 2003, pp. 51–63.

[230] D. Yu, F. Sattar, and K.-K. Ma, "Watermark detection and extraction using independent component analysis method," *EURASIP Jounal on Applied Signal Processing - Special Issue on Nonlinear Signal and Image Processing (PART II)*, vol. 2002, no. 1, pp. 92–104, Jan. 2002.

[231] M. B. Zadeh, A. Mansour, C. Jutten, and F. Marvasti, "A geometric approach for separating several speech signals," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'04)*, Granada, Spain, 2004, pp. 798–806.

[232] S. Zhang and P. K. Rajan, "Independent component analysis of digital image watermarking," in *Proc. of IEEE International Symposium on Circuits and Systems (ISCAS'02)*, vol. 3, May 2002, pp. 217–220.

[233] Y. Zhao, "Dual domain semi-fragile watermarking for image authentication," Master's thesis, University of Toronto, 2003.

[234] A. Ziehe, M. Kawanabe, S. Harmeling, and K. R. Müller, "Blind separation of post-nonlinear mixtures using ACE and temporal decorrelation," in *Proc. Int. Conf. on Independent Component Analysis and Blind Signal Separation (ICA'01)*, California, USA, Dec. 2001, pp. 433–438.

[235] ——, "Blind separation of post-nonlinear mixtures using gaussianizing transformations and temporal decorrelation," in *Proc. Int. Symposium on Independent Component Analysis and Blind Signal Separation (ICA'03)*, Nara, Japan, Apr. 2003, pp. 269–274.

[236] A. Ziehe and K. Müller, "TDSEP–an efficient algorithm for blind separation using time structure," in *Proc. Int. Conf. on Artificial Neural Networks (ICANN'98)*, pp. 675–680.

[237] A. Ziehe, M. Kawanabe, S. Harmeling, and K. R. Müller, "Blind separation of post-nonlinear mixtures using linearizing transformations and temporal decorrelation," *Journal of Machine Learning Research*, vol. 4, pp. 1319–1338, Dec. 2003.

[238] A. Ziehe, P. Laskov, G. Nolte, and K.-R. Müller, "A fast algorithm for joint diagonalization with non-orthogonal transformations and its application to blind source separation," *Journal of Machine Learning Research*, vol. 5, pp. 777–800, July 2004.

Author's Publications

Journal Publications

1. T. V. Nguyen, J. C. Patra, and A. Das, "A post nonlinear geometric algorithm for independent component analysis," *Digital Signal Processing*, vol. 15, no. 3, pp. 276–294, May 2005.

2. T. V. Nguyen, J. C. Patra, and S. Emmanuel, "gpICA: A Novel Nonlinear ICA Algorithm Using Geometric Linearization," *EURASIP Journal on Applied Signal Processing*, vol. 2007, Article ID 31951, 12 pages, 2007.

3. T. V. Nguyen, J. C. Patra, and P. K. Meher"WMicaD: A New Digital Watermarking Technique Using Independent Component Analysis," *EURASIP Journal on Applied Signal Processing*, final revision, 2007.

4. T. V. Nguyen and J. C. Patra, "A Simple ICA-based Digital Image Watermarking Scheme," *Digital Signal Processing*, submitted, July 2006.

Conference Publications

1. T. V. Nguyen, J. C. Patra, and E.-L. Ang, "Blind image extraction from nonlinear mixtures using MLP-based ICA," in *Proc. IEEE Int. Conf. on Multimedia and Expo (ICME'03)*, vol. 1, Baltimore, USA, July 2003, pp. 249–252.

2. T. V. Nguyen, J. C. Patra, and A. Das, "A geometric approach to post nonlinear mixture in blind source separation," in *Proc. IEEE Int.Conf. on Communications Systems (ICCS'04)*, vol. 1, Singapore, Sept. 2004, pp. 260–264.

3. T. V. Nguyen, J. C. Patra, and N. S. Chaudhari, "A novel digital image watermarking scheme using blind source separation," in *Proc. IEEE Int. Conf. on*

Image Processing (ICIP'04), vol. 1, Singapore, Oct. 2004, pp. 2649–2652.

4. T. V. Nguyen and J. C. Patra, "Digital image watermarking using independent component analysis," in *Proc. IEEE of the 5th Pacific Rim Conference on Multimedia (PCM'04)*, Japan, Nov. 2004, pp. 364–371.

5. T. V. Nguyen and J. C. Patra, "WMica: An ICA-based digital watermarking technique," in *Proc. IEEE Int. Symposium on Signal Processing and Information Technology (ISSPIT'04)*, vol. 1, Rome, Italy, Dec. 2004, pp. 306–309.

6. T. V. Nguyen, J. C. Patra, A. Das, and G. S. Ng, "Post nonlinear blind source separation by geometric linearization," in *Proc. of the Int. Joint Conf. on Neural Networks (IJCNN'05)*, vol. 1, Montreal, Canada, Aug. 2005, pp. 244–249.