Winter 12-2018

Different Estimation Methods for the Basic Independent Component Analysis Model

Zhenyi An
Washington University in St. Louis

Washington University in St. Louis

Department of Mathematics and Statistics

Different Estimation Methods for the Basic Independent Component Analysis Model

by

Zhenyi An

A thesis presented to
The Graduate School
of Washington University in
partial fulfillment of the
requirements for the degree
of Master of Arts

Dec 2018

Saint Louis, Missouri

Contents

List of Figures

Acknowledgments

I'd like to appreciate my supervisor Dr. Figueroa-Lopez's help during my preparation process of the thesis, whose patience and consideration really impressed me. Sincerely thanks Dr. Ding for offering the inspiration and feedback on the oral defense. Until the final step of the graduation, there's always sufficient support from faculty and staff in Mathematics Department. Every acknowledgment during the program period can't be achieved without all of your help. I am also thankful for the understanding and consideration from my family, even though they didn't stay by my side physically, I still can feel their support.

Zhenyi An

Washington University in St. Louis

Dec 2018

ABSTRACT OF THE THESIS

Different Estimation methods for the Basic Independent

Component Analysis Model

by

Zhenyi An

Master of Arts in Statistics

Washington University in St. Louis, December 2018

Research Advisor: Professor José E. Figueroa-López

Inspired by classic cocktail-party problem, the basic Independent Component Analysis (ICA) model is created. What differs Independent Component Analysis (ICA) from other kinds of analysis is the intrinsic non-Gaussian assumption of the data. Several approaches are proposed based on maximizing the non-Gaussianity of the data, which is measured by kurtosis, mutual information, and others. With each estimation, we need to optimize the functions of expectations of non-quadratic functions since it can help us to access the higher-order statistics of non-Gaussian part of the data. In this thesis, our goal is to review the one of the most efficient estimation methods, that is, the Fast Fixed-Point Independent Component Analysis (FastICA) algorithm, illustrate it with some examples using an R package.

Chapter 1

Introduction to the *Independent Component Analysis (ICA)*

1.1 Introduction

In this section, we discuss the connection between Independent Component Analysis (ICA) and some other popular approaches such as Principle Component Analysis (PCA), decorrelation and whitening transformation. This part is based on Chapter 7 from the book *Independent Component Analysis by Hyvarinen et al.*

Firstly, we introduce the cocktail-party problem, which is a prime example of the power of Independent Component Analysis. Imagine that you are in a room where there are three people speaking at the same time. There are also three microphones in three different areas. So, the microphones would record and offer you three recorded time signals, which are denote by $x_1(t)$, $x_2(t)$ and $x_3(t)$ at time index t. Each recorded signals is a weighted sum of the speech signals from the three speakers, which we denote by $s_1(t)$, $s_2(t)$, and $s_3(t)$. Let's express this as a system of linear equations:

$$x_1(t) = a_{11}s_1(t) + a_{12}s_2(t) + a_{13}s_3(t), \tag{1.1}$$

$$x_2(t) = a_{21}s_1(t) + a_{22}s_2(t) + a_{23}s_3(t), \tag{1.2}$$

$$x_3(t) = a_{31}s_1(t) + a_{32}s_2(t) + a_{33}s_3(t), \tag{1.3}$$

where the a_{ij} with $i, j = 1, \ldots, 3$ are some parameters depending on the distances of the microphones from the speakers. If we were able to estimate the original speech signals $s_1(t)$, $s_2(t)$, and $s_3(t)$ using only the recorded signals $x_i(t)$, that would be fine, however, we know neither the a_{ij} nor the $s_i(t)$, so the problem becomes more difficult to solve.

One approach to solving this problem is to use some information on the statistical properties of the signals $s_i(t)$ to estimate both the a_{ij} and the $s_i(t)$. Actually, we can see that it suffices to assume that $s_1(t)$, $s_2(t)$, and $s_3(t)$ are, at each time instant t, statistically independent. This is a reasonable assumption to fit in many given cases. In general, in the Independent Component Analysis (ICA) model, it's common to estimate the a_{ij} based on the independency of the information already given, which allows us to approximately separate the three original signals, $s_1(t)$, $s_2(t)$, and $s_3(t)$, from their mixtures, $x_1(t)$, $x_2(t)$, and $x_3(t)$.

The basic Independent Component Analysis (ICA) model consist of n random variables [C. Jutten et al.-1991] , $x_1, ..., x_n$, which are modeled as linear combinations of random variables some $s_1, ..., s_n$:

$$x_i = a_{i1}s_1 + a_{i2}s_2 + ... + a_{in}s_n,$$

for all $i = 1, \ldots, n$, where a_{ij}, $i, j = 1, ..., m$ are some unknown real coefficients. By definition, the s_i's are statistically mutually independent. It's more convenient to use vector-matrix notation. Let us denote by $\mathbf{x}$ the random column vector whose elements are the mixtures $x_1, ..., x_n$ and likewise by $\mathbf{s}$ be the random column vector with elements $s_1, ..., s_n$. Let us denote by $\mathbf{A}$ the matrix with elements a_{ij}. Thus, the mixing model can be written as

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{1.4}$$

or

$$\mathbf{x} = \sum_{i=1}^{n} \mathbf{a}_i s_i, \tag{1.5}$$

where $\mathbf{a}_i$ is the ith column of the matrix $\mathbf{A}$. The basic Independent Component Analysis (ICA) model can be estimated when the following assumptions are satisfied.

Assumption 1. *We assume that the independent components $s_1, \ldots, s_n$ are statistically independent.*

Assumption 2. *The independent components $s_1, \ldots, s_n$ must have non-Gaussian distributions with finite moments of any degree.*

Assumption 3. *For simplicity of the matrix operation, we assume that the unknown mixing matrix is square.*

There also exist some indeterminacies that would hold for the Independent Component Analysis (ICA).

1. We can't determine the variances of the independent components. This is because both $\mathbf{s}$ and $\mathbf{A}$ are unknown, any scalar multiplier in one of the sources s_i could always be canceled by dividing the corresponding column $\mathbf{a}_i$ of $\mathbf{A}$ by the same scalar, say $\mathbf{a}_i$:

$$\mathbf{x} = \sum_i \left(\frac{1}{\alpha_i}\mathbf{a}_i\right)(s_i\alpha_i) \tag{1.6}$$

2. We can't determine the order of the independent components either. Since both $\mathbf{s}$ and $\mathbf{A}$ being unknown, we can freely change the order of the terms in the summation

$$\mathbf{x} = \sum_{i=1}^{n} \mathbf{a}_i s_i \tag{1.7}$$

and rename the independent components.

Next, let's discuss about the connection between the whitening transformation [J.-F. Cardoso-1989] and the Independent Component Analysis (ICA). Whitening process is stronger than decorrelation. A zero-mean random vector after whitening transformation, say $\mathbf{y}$, means that its components are uncorrelated and its covariance matrix equals the identity matrix:

$$E\{\mathbf{y}\mathbf{y}^{\mathbf{T}}\} = \mathbf{I}. \tag{1.8}$$

So, whitening transformation means that we linearly transform the observed data vector $\mathbf{x}$ by multiplying it with some matrix $\mathbf{V}$:

$$\mathbf{z} = \mathbf{V}\mathbf{x}, \tag{1.9}$$

so that we obtain a new vector z that is "white". The whitening process transforms the mixing matrix into a new one. Thus, we have from (1.4) and (1.5) that:

$$\mathbf{z} = \mathbf{V}\mathbf{A}\mathbf{s} = \tilde{\mathbf{A}}\mathbf{s} \tag{1.10}$$

Meanwhile, let's consider an orthogonal transformation $\mathbf{U}$ of $\mathbf{z}$:

$$\mathbf{y} = \mathbf{U}\mathbf{z} \tag{1.11}$$

Due to the orthogonality of $\mathbf{U}$, we have

$$E\{\mathbf{y}\mathbf{y}^{\mathbf{T}}\} = E\left\{\mathbf{U}\mathbf{z}\mathbf{z}^{\mathbf{T}}\mathbf{U}^{\mathbf{T}}\right\} = \mathbf{U}\mathbf{I}\mathbf{U}^{\mathbf{T}} = \mathbf{I} \tag{1.12}$$

which is to say, $\mathbf{y}$ is "white" as well. Thus, we can't tell if the independent components are given by $\mathbf{z}$ or $\mathbf{y}$ after using the whitening transformation alone. Since $\mathbf{y}$ could be any orthogonal transformation of $\mathbf{z}$, whitening transformation gives the independent components only up to an orthogonal transformation. Otherwise, whitening transformation is useful as a preprocessing step in Independent Component Analysis (ICA). The utility of whitening process resides in the fact that the new mixing matrix

$$\tilde{\mathbf{A}} = \mathbf{V}\mathbf{A} \tag{1.13}$$

is orthogonal. This can be seen from

$$E\{\mathbf{z}\mathbf{z}^{\mathbf{T}}\} = \tilde{\mathbf{A}}E\{\mathbf{s}\mathbf{s}^{\mathbf{T}}\}\tilde{\mathbf{A}}^{T} = \tilde{\mathbf{A}}\tilde{\mathbf{A}}^{\mathbf{T}} = \mathbf{I} \tag{1.14}$$

As the properties shown above, we can restrict our search for the mixing matrix to the space of orthogonal matrices. Instead of having to estimate the n^2 parameters that are the elements of the original matrix $\mathbf{A}$, we only need to estimate an orthogonal mixing matrix $\tilde{\mathbf{A}}$, which contains $n\left(n-1\right)/2$ degrees of freedom. For example, in two dimensions, an orthogonal transformation is determined by a single parameter; in larger dimensions, an orthogonal matrix contains only about half of the number of parameters of an arbitrary matrix. Thus we can conclude that whitening process solves half of the problem of Independent Component Analysis (ICA). Hence, whitening transformation helps to reduce the complexity of the problem.

Otherwise, whitening transformation also makes it clear why Gaussian variables are not allowed in Independent Component Analysis (ICA). Suppose that the joint distribution of two independent components, s_1 and s_2, is Gaussian. This means that their joint pdf is given by

$$p(s_1, s_2) = \frac{1}{2\pi} \exp\left(-\frac{s_1^2 + s_2^2}{2}\right) = \frac{1}{2\pi} \exp\left(-\frac{||\mathbf{s}||^2}{2}\right) \tag{1.15}$$

Now, assume that the mixing matrix $\mathbf{A}$ is orthogonal. For example, we could assume that this is so because the data has been processed through whitening transformation. By using the formula of transforming probability density function, we obtain:

$$p_y(y) = \frac{1}{|\det Jg(g^{-1}(y))|} p_x(g^{-1}(y)), \tag{1.16}$$

where $y = g(x)$ and noting that for an orthogonal matrix $\mathbf{A^{-1}} = \mathbf{A^T}$ holds, we obtain the joint density of the mixtures x_1 and x_2 as density is given by

$$p(x_1, x_2) = \frac{1}{2\pi} exp(-\frac{||\mathbf{A}^T\mathbf{x}||^2}{2})|\det \mathbf{A}^T| \tag{1.17}$$

Due to the orthogonality of $\mathbf{A}$, we have $||\mathbf{A^T x}||^\mathbf{2} = ||\mathbf{x}||^2$ and $|\det \mathbf{A}| = 1$; note that if $\mathbf{A}$ is orthogonal, so is $\mathbf{A}^T$. Thus, we have

$$p(x_1, x_2) = \frac{1}{2\pi} exp(-\frac{||\mathbf{x}||^2}{2}), \tag{1.18}$$

Thus we see that the orthogonal mixing matrix does not change the probability density function, since it does not appear in the probability density function (1.18) above at all and the original and mixed distributions keep the same. Hence we are unable to infer the mixing matrix from the mixtures.

We conclude that in the case of Gaussian independent components, we can only estimate the Independent Component Analysis (ICA) model up to an orthogonal transformation.

In the case that some of the components are Gaussian and others are non-Gaussian, we can estimate all the non-Gaussian components, but the Gaussian components can't be separated from each other. Only in the case that there 's just one Gaussian component, we can estimate the model, since the single Gaussian component does not have any other Gaussian components that it could be mixed with.

1.2 Mathematical Preliminaries

In this thesis, there will be some high-order statistics concepts and properties needed to further research.

1.2.1 Cumulants

First, we proceed to the general definition of cumulants. Assume that x is a real-valued, zero-mean, continuous scalar random variable with probability density function $p_x(x)$.

We know that the first characteristic function $\varphi(w)$ of x is defined as the following:

$$\varphi(w) = E\{\exp(jwx)\} = \int_{-\infty}^{\infty} \exp(jwx) p_x(x) dx \tag{1.19}$$

where $j = \sqrt{-1}$ and w is the transformed variable corresponding to x. Each probability distribution is uniquely specified by its characteristic function. Expanding the characteristic function $\varphi(w)$ into its Taylor series yields

$$\varphi(w) = \int_{-\infty}^{\infty} (\sum_{k=0}^{\infty} \frac{x^k (jw)^k}{k!}) p_x(x) dx = \sum_{k=0}^{\infty} E\{x^k\} \frac{(jw)^k}{k!} \tag{1.20}$$

Thus the coefficient terms of this expansion are moments $E\{x^k\}$ of x. For this reason, the characteristic function $\varphi(w)$ is also called moment generating function.

The second characteristic function $\phi(w)$ of x is given by the natural logarithm of the first characteristic function (1.19) :

$$\phi(w) = \ln(\varphi(w)) = \ln(E\{\exp(jwx)\}) \tag{1.21}$$

The cumulants κ_k of x are defined as the coefficients of the Taylor series expansion of the second characteristic function (1.21):

$$\phi(w) = \sum_{k=0}^{n} \kappa_k \frac{(jw)^k}{k!} \tag{1.22}$$

where the kth cumulant is obtained as the derivative

$$\kappa_k = (-j)^k \frac{d^k \phi(w)}{dw^k} \mid_{w=0} \tag{1.23}$$

For a zero mean random variable x, the first four cumulants are

$$\kappa_1 = 0,\ \kappa_2 = E\{x^2\},\ \kappa_3 = E\{x^3\},\ \text{and} \tag{1.24}$$

$$\kappa_4 = E\{x^4\} - 3[E\{x^2\}]^2 \tag{1.25}$$

Hence the first three cumulants are equal to the respective moments, and the fourth cumulant κ_4 is recognized to be the kurtosis defined earlier in (2.4).

Then, the respective expressions for the cumulants when the mean $E\{x\}$ of x is nonzero is listed below.

$$\kappa_1 = E\{x\} \tag{1.26}$$

$$\kappa_2 = E\{x^2\} - [E\{x\}]^2 \tag{1.27}$$

$$\kappa_3 = E\{x^3\} - 3E\{x^2\}E\{x\} + 2[E\{x\}]^3 \tag{1.28}$$

$$\kappa_4 = E\{x^4\} - 3[E\{x^2\}]^2 - 4E\{x^3\}E\{x\} + 12E\{x^2\}[E\{x\}]^2 - 6[E\{x\}]^4 \tag{1.29}$$

Now consider the multivariate case. Let $\mathbf{x}$ be a random vector and $p_{\mathbf{x}}(\mathbf{x})$ its probability density function. The characteristic function of $\mathbf{x}$ is again

$$\varphi(w) = E\{\exp(jw\mathbf{x})\} = \int_{-\infty}^{\infty} \exp(jw\mathbf{x}) p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \tag{1.30}$$

where w is now a row vector having the same dimension as $\mathbf{x}$, and the integral is computed over all components of $\mathbf{x}$. The moments and cumulants of $\mathbf{x}$ are obtained in a similar manner to the one-unit case.

The moments of $\mathbf{x}$ are coefficients of the Taylor series expansion of the first characteristic function $\varphi(w)$, and the cumulants are the coefficients of the expansion of the second characteristic function $\phi(w) = \ln(\varphi(w))$. In the multivariate case, the cumulants are usually called cross-cumulants. It can be shown that the second, third, and fourth order cumulants for a zero mean random vector $\mathbf{x}$ are [M. Girolami-1999]

$$\operatorname{cum}(x_i, x_j) = E\{x_i x_j\} \tag{1.31}$$

$$\operatorname{cum}(x_i, x_j, x_k) = E\{x_i x_j x_k\} \tag{1.32}$$

$$\operatorname{cum}(x_i, x_j, x_k, x_l) = E\{x_i x_j x_k x_l\} - E\{x_i x_j\}E\{x_k x_l\} \tag{1.33}$$

$$- E\{x_i x_k\}E\{x_j x_l\} - E\{x_i x_l\}E\{x_j x_k\} \tag{1.34}$$

Hence the second cumulant is equal to the second moment $E\{x_i x_j\}$, which in turn is the correlation r_{ij} covariance c_{ij} between the variables x_i and x_j. Similarly, the third cumulant $\mathrm{cum}(x_i, x_j, x_k)$ is equal to the third moment $E\{x_i x_j x_k\}$. However, the fourth cumulant differs from the fourth moment $E\{x_i x_j x_k x_l\}$ of the random variables x_i, x_j, x_k, and x_l. In general, higher-order moments correspond to correlations used in second-order statistics, and cumulants are the higher-order counterparts of covariances. Both moments and cumulants contain the same statistical information since cumulants can be represented in terms of sums of products of moments. But, it's more recommended to utilize cumulants because they present in a clearer way the additional information provided by higher-order statistics. In particular, it can be shown that cumulants have the following properties not shared by moments.

1. Let $\mathbf{x}$ and $\mathbf{y}$ be statistically independent random vectors having the same dimension, then the cumulant of their sum $\mathbf{z} = \mathbf{x} + \mathbf{y}$ is equal to the sum of the cumulants of $\mathbf{x}$ and $\mathbf{y}$. This property also holds for the sum of more two independent random vectors.

2. If the distribution of the random vector or process $\mathbf{x}$ is multivariate gaussian, all its cumulants of order three and higher are identically zero.

Thus higher-order cumulants measure the departure of a random vector from a gaussian random vector with an identical mean vector and covariance matrix, which makes it possible to use cumulants for extracting the non-Gaussian part of a signal. example, they make it possible to ignore additive gaussian noise corrupting a non-Gaussian signal using cumulants.

Chapter 2

Independent Component Analysis (ICA) Estimation

2.1 Independent Component Analysis (ICA) by Non-Gaussianity Maximization

2.1.1 Non-gaussian is independent

As shown in the last section, it's possible for us to estimate non-Gaussian variables but Gaussian variables. Hence, non-Gaussianity [N.Delfosse-1995] could be used as a leading principal in Independent Component Analysis (ICA) estimation.

Let's recall the Central Limit Theorem first, which says, the distribution of a sum of independent random variables tends toward a Gaussian distribution under certain conditions. Let us now assume that the data vector $\mathbf{x}$ is distributed as the following

Independent Component Analysis (ICA) data model:

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{2.1}$$

i.e., it is a mixture of independent components. Estimation of the independent components can be accomplished by finding the right linear combinations of the mixture variables, since we can solve the equation as

$$\mathbf{s} = \mathbf{A}^{\mathbf{-1}}\mathbf{x} \tag{2.2}$$

Thus, to estimate one of the independent components, we can consider a linear combination of the x_i. Let us denote this by $\mathbf{y} = \mathbf{b}^{\mathbf{T}}\mathbf{x} = \sum_i b_i x_i$, where b is a vector to be determined. Note that we also have $\mathbf{y} = \mathbf{b}^{\mathbf{T}}\mathbf{A}\mathbf{s}$. Thus, $\mathbf{y}$ is a certain linear combination of the s_i, with coefficients given by $\mathbf{b}^{\mathbf{T}}\mathbf{A}$. Let us denote this vector by $\mathbf{q}$. Then we have

$$\mathbf{y} = \mathbf{b}^{\mathbf{T}}\mathbf{x} = \mathbf{q}^{\mathbf{T}}\mathbf{s} = \sum_i q_i s_i. \tag{2.3}$$

If $\mathbf{b}$ were one of the rows of the inverse of $\mathbf{A}$, this linear combination $\mathbf{b}^{\mathbf{T}}\mathbf{x}$ would actually equal one of the independent components. In that case, the corresponding $\mathbf{q}$ would be such that just one of its elements is 1 and all the others are zero. Actually it's hard for us to determine exactly since there's no information about matrix $\mathbf{A}$. Still we can find a good approximation of the estimator.

We can vary the coefficients in $\mathbf{q}$, and see how the distribution of $\mathbf{y} = \mathbf{q}^{\mathbf{T}}\mathbf{s}$ changes. The idea behind this is that since a sum of even two independent random variables is

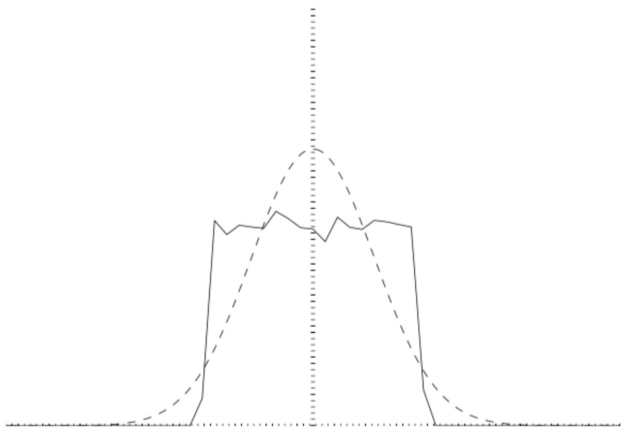

Figure 2.1: A single uniformly distributed independent component with real-lined compared with gaussian distribution with dashed line

more gaussian than the original variables, $\mathbf{y} = \mathbf{q^T s}$ is usually more gaussian than any of the s_i. Note that this is strictly true only if the s_i's have identical distributions. In this case, obviously only one of the elements q_i of $\mathbf{q}$ is nonzero.

In practice, there's no need to know the values of $\mathbf{q}$, because $\mathbf{q^T s} = \mathbf{b^T x}$ by the definition of $\mathbf{q}$. We can just let $\mathbf{b}$ vary and look at the distribution of $\mathbf{b^T x}$.

Therefore, we could take as $\mathbf{b}$ a vector that maximizes the non-Gaussianity of $\mathbf{b^T x}$. Such a vector would necessarily correspond to a $\mathbf{q} = \mathbf{A^T b}$. This means that $\mathbf{y} = \mathbf{b^T x} = \mathbf{q^T s}$ equals one of the independent components. Maximizing the non-Gaussianity of $\mathbf{b^T x}$ thus gives us one of the independent components. We can explain why maximizing the non-Gaussianity equals to obtaining the independent component through the Figure 2.1 and Figure 2.2.

In classical central limit theorem, it says the sample mean of the random variables

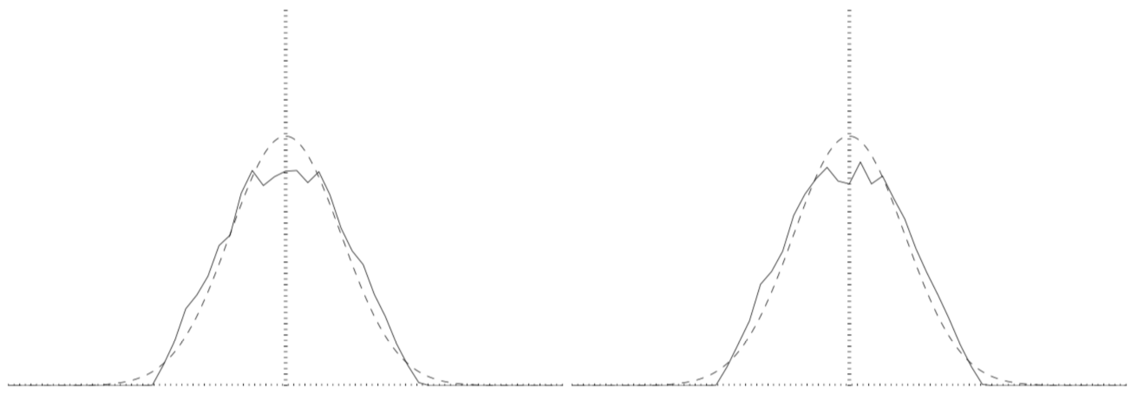

Figure 2.2: The real-lined mixture of independent components compared with dashed lined gaussian distribution

tends to be more gaussian as the sample size increases. Here, the mixture of the independent components also can be seen as the weighted sum of the independent random variables. Thus, there's no doubt that as the number of the independent components increase, the mixture should be more gaussian when compared with the original single independent component. The plot above can tell the fact.

2.2 Measuring Non-Gaussianity by kurtosis

2.2.1 Extrema of kurtosis give independent components

In order to measure the non-Gaussianity in a quantitative way, we introduce the *kurtosis* [A. Hyvarinen-1999]. The kurtosis of y, denoted by kurt (y) is defined by

$$\mathrm{kurt}(\mathrm{y}) = E\{\mathrm{y}^4\} - 3(E\{\mathrm{y}^2\})^2 \tag{2.4}$$

Remember that all the random variables here have zero mean. For a Gaussian $\mathbf{y}$, the fourth moment equals $3(E\{\mathbf{y}^2\})^2$ and then kurtosis is zero. Typically, non-Gaussianity is measured by the absolute value of kurtosis. The square of kurtosis can also be used. Due to its simplicity, kurtosis is widely used in Independent Component Analysis (ICA) and related fields. Theoretical analysis is simplified because of the following linearity property: If x_1 and x_2 are two independent random variables, it holds

$$\operatorname{kurt}(x_1 + x_2) = \operatorname{kurt}(x_1) + \operatorname{kurt}(x_2), \tag{2.5}$$

and

$$\operatorname{kurt}(\alpha x_1) = \alpha^4 \operatorname{kurt}(x_1), \tag{2.6}$$

where α is a constant.

2.2.2 Gradient algorithm using kurtosis

Many of the Independent Component Analysis (ICA) criteria have the basic form of minimizing a cost function $J(\mathbf{W})$ with respect to a parameter matrix $\mathbf{W}$.

In gradient descent [A. Hyvarinen-1998], we minimize a function $J(\mathbf{W})$ iteratively by starting from some initial point $\mathbf{w}(0)$, computing the gradient of $J(\mathbf{W})$ at this point, and then moving in the direction of the negative gradient by a suitable distance. Then we repeat the same procedure at a new point, and so forth. For $t = 1, 2, ...$, we have the update rule

$$w(t) = w(t-1) - \alpha(t)\frac{\partial J(w)}{\partial w}|_{w=w(t-1)} \tag{2.7}$$

with the gradient taken at the point $w(t-1)$. The parameter $\alpha(t)$ here gives the length of the step in the negative gradient direction. It's often called the *step size* or *learning rate*. The iteration (2.7) is continued until it converges.

Denote the difference between the new and old value by

$$w(t) - w(t-1) = \Delta w \tag{2.8}$$

We can then write the rule (2.8) either as

$$\Delta w = -\alpha\frac{\partial J(w)}{\partial w} \tag{2.9}$$

or

$$\Delta w \propto -\frac{\partial J(w)}{\partial w} \tag{2.10}$$

where α is a shorthand notation of the *step size* or *learning rate* and the symbol $\propto$ is read "is proportional to"; it is then understood that the vector on the left-hand side, Δw, has the same direction as the gradient vector on the right-hand side.

In order to maximize the kurtosis absolute value, we would start from some vector $\mathbf{w}$, then compute the direction in which the absolute value of the kurtosis of $\mathbf{y} = \mathbf{w^T z}$ is growing most strongly and then move the vector $\mathbf{w}$ in that direction. Then the gradient of the absolute value of kurtosis of $\mathbf{w^T z}$ can be simply computed as

$$\frac{\partial|\text{kurt}(\mathbf{w^T x})|}{\partial \mathbf{w}} = 4sign(\text{kurt}(\mathbf{w^T z}))[E\{\mathbf{z}(\mathbf{w^T z})^{\mathbf{3}}\} - \mathbf{3w}||\mathbf{w}||^{\mathbf{2}}] \tag{2.11}$$

Since the data is handled through whitening transformation, we have $E\{(\mathbf{w^T z})^2\} = ||\mathbf{w}||^2$.

We are optimizing this function on the unit sphere $||\mathbf{w}||^2 = 1$, the gradient method must be complemented by projecting **w** on the unit sphere after every step. This can be done by dividing **w** by its norm. Since the latter term in brackets in (2.11) would simply change the norm of **w** in the gradient algorithm, and not its direction, it can be omitted. This is because only the direction of **w** is interesting, and any change in the norm is not significant because the norm is normalized to unity anyway.

For the unconstrained problem of minimizing a multivariate function, the most classic approach is gradient descent. When the solution is a vector **w**; the matrix case goes through in a completely analogous fashion.

Then, we obtain the gradient algorithm in the multivariate case:

$$\Delta \mathbf{w} \propto \text{sign}(\text{kurt}(\mathbf{w^T z}))E\{\mathbf{z}(\mathbf{w^T z})^3\} \tag{2.12}$$

$$\mathbf{w} \leftarrow \mathbf{w}/||\mathbf{w}|| \tag{2.13}$$

And an adaptive(data-driven) version of this algorithm:

$$\Delta \mathbf{w} \propto \text{sign}(\text{kurt}(\mathbf{w^T z}))\mathbf{z}(\mathbf{w^T z})^3 \tag{2.14}$$

$$\mathbf{w} \leftarrow \mathbf{w}/||\mathbf{w}|| \tag{2.15}$$

where every observation $z\,(t)$ can be used in the algorithm at once. However, it must be noted that when computing sign(kurt($\mathbf{w^T x}$)), the expectation operator in the definition of kurtosis can't be omitted. Instead, the kurtosis must be properly estimated from a time-average; of course, this time-average can be estimated on-line. Denoting by γ the estimate of the kurtosis, we could use

$$\Delta\gamma \propto ((\mathbf{w^T z})^4 - 3) - \gamma \tag{2.16}$$

This gives the estimate of kurtosis as a kind of a running average. Actually, in many cases we assume one acknowledges the nature of the distributions of the independent components, i.e., whether they are subgaussian or supergaussian. Then we can simply plug the correct sign of kurtosis in the algorithm and avoid its estimation.

2.2.3 A fast fixed-point algorithm using kurtosis

We have obtained a gradient method for maximizing the non-Gaussianity measured by the absolute value of kurtosis. The advantage of such gradient methods is that the inputs $z(t)$ can be used in the algorithm at once, which enables fast adaptation in a nonstationary environment. But, in this case, the convergence rate is slow and it depends on a good choice of the learning rate sequence.

In the case of making the learning rate radically faster and more reliable, the fixed-point iteration algorithms can be a good option. To derive a more efficient fixed-point iteration, we note that at a stable point of the gradient algorithm, the gradient must point in the direction of $\mathbf{w}$, that is, the gradient must be equal to $\mathbf{w}$ multiplied by some scalar constant. Only in such way, adding the gradient to $\mathbf{w}$ does not alternate its direction. And the vector $\mathbf{w}$ converges.

The statement above means that after normalization to unit norm, the value of $\mathbf{w}$ is not changed except perhaps by changing its sign. Equating gradient of kurtosis in (2.11) with $\mathbf{w}$, this means that we should have

$$\mathbf{w} \propto [E\{\mathbf{z}(\mathbf{w}^{\mathbf{T}}\mathbf{z})^3\} - 3||\mathbf{w}||^2\mathbf{w}] \tag{2.17}$$

This equation immediately suggests a fixed-point algorithm where we first compute the right-hand side, and give this as the new value for $\mathbf{w}$:

$$\mathbf{w} \leftarrow E\{\mathbf{z}(\mathbf{w}^{\mathbf{T}}\mathbf{z})^3\} - 3\mathbf{w} \tag{2.18}$$

After every fixed-point iteration, $\mathbf{w}$ is divided by its norm to remain on the constraint set. The final vector $\mathbf{w}$ gives one of the independent components as the linear combination $\mathbf{w}^{\mathbf{T}}\mathbf{z}$.

Note that convergence of this fixed-point iteration indicates that the old and new values of $\mathbf{w}$ point are in the same direction. It is not necessary that the vector converges to a single point, since $\mathbf{w}$ and $-\mathbf{w}$ define the same direction. This kind of algorithm is called Fast Fixed-Point Algorithm for Independent Component Analysis (FastICA), whose convergence is cubic, and there's no learning rate or other adjustable parameters needed in the algorithm.

2.2.4 Negentropy as non-Gaussianity measure

Negentropy is defined in terms of the information-theoretic quantity of differential entropy, which we just call it entropy here. The entropy of a random variable is related to the information that the observation of the variable gives. The more "random", i.e., unpredictable and unstructured the variable is, the larger its entropy. The (differential) entropy H of a random vector $\mathbf{y}$ with density $p_y\left(\eta\right)$ is defined as

$$H(\mathbf{y}) = -\int p_y(\eta) \log p_y\left(\eta\right) d\eta \tag{2.19}$$

Remark 2.2.1. A Gaussian variable has the largest entropy among all random variables of equal variance. This means that entropy could be used as a measure of non-Gaussianity.

To obtain a measure of non-Gaussianity that is zero for a gaussian variable and always nonnegative, one often uses a normalized version of differential entropy, called negentropy. Negentropy J is defined as follows

$$J(\mathbf{y}) = H(\mathbf{y}_{gauss}) - H(\mathbf{y}) \tag{2.20}$$

where $\mathbf{y}_{gauss}$ is a gaussian random variable of the same covariance matrix as $\mathbf{y}$.

But, the problem in using negentropy is that it's computationally hard. Estimating negentropy using the definition requires an estimate (possibly nonparametric) of the probability density function. In practice, we only need approximation of 1-D (neg)entropies, so we consider the scalar case only. The classic method of approximating negentropy is based on higher-order cumulants, using the polynomial density expansions, which gives the approximation

$$J(y) \approx \frac{1}{12}E\{y^3\}^2 + \frac{1}{48}\text{kurt}(y)^2, \tag{2.21}$$

in which the random variable y is assumed to be of zero mean and unit variance. One useful approach is to generalize the high-order cumulant approximation so that it uses expectations of general nonquadratic functions. As a simple special case, we can take any two non-quadratic functions G^1 and G^2, so that G^1 is odd and G^2 is even, and consider the following approximation:

$$J(y) \approx k_1(E\{G^1(y)\}^2 + k_2(E\{G^2(y)\} - E\{G^2(v)\})^2 \tag{2.22}$$

where k_1 and k_2 are positive constants, v is a Gaussian variable of zero mean and unit variance and for non-quadratic functions G^i, i is an index, not a power. If we use only one non-quadratic function G, the approximation becomes

$$J(y) \propto [E\{G(y)\} - E\{G(\nu)\}]^2 \tag{2.23}$$

for practically any non-quadratic function G. Thus, we obtain approximations of negentropy that offer a good compromise between the properties of the two classic non-Gaussianity measures presented by kurtosis and negentropy.

2.2.5 Gradient algorithm using negentropy

As with kurtosis, we can derive a simple gradient algorithm for maximizing negentropy. Taking the gradient of the approximation of negentropy [A. Hyvarinen-1999-1] in (2.23) with respect to $\mathbf{w}$, and taking the normalization $E\{(\mathbf{w^T z})^2\} = ||\mathbf{w}||^2 = 1$ into account, one obtains the following algorithm

$$\Delta\mathbf{w} \propto \gamma E\{\mathbf{z}g(\mathbf{w^T z})\}, \tag{2.24}$$

$$\mathbf{w} \leftarrow \mathbf{w}/||\mathbf{w}|| \tag{2.25}$$

where $\gamma = E\{G(\mathbf{w^T z})\} - E\{G(\nu)\}$, ν being a standardized gaussian random variable. The normalization is necessary to project $\mathbf{w}$ on the unit sphere to keep the variance of $\mathbf{w^T z}$ constant. The function g is the derivative of the function G used in the approximation of negentropy. The expectation could be omitted to obtain an adaptive stochastic gradient algorithm. The constant γ, which gives the algorithm a kind of "self-adaptation" quality, can be easily estimated on-line follows:

$$\Delta\gamma \propto (G(\mathbf{w^T z}) - E\{G(\nu)\}) - \gamma \tag{2.26}$$

This constant corresponds to the sign of kurtosis in (2.11).

2.2.6 A Fast Fixed-point algorithm using negentropy

As with kurtosis, a much faster method for maximizing negentropy than that given by the gradient method, can be found using a fixed-point algorithm. The resulting Fast Fixed-Point Algorithm for Independent Component Analysis (FastICA) finds a direction, i.e., a unit vector $\mathbf{w}$, such that the projection $\mathbf{w^T z}$ maximizes non-Gaussianity. Non-Gaussianity is here measured by the approximation of negentropy $J(\mathbf{w^T z})$ given in (2.23).

Looking at the gradient method in (2.24) immediately suggests the following fixed-point iteration:

$$\mathbf{w} \leftarrow E\{\mathbf{z}g(\mathbf{w^T z})\} \tag{2.27}$$

which would be followed by the normalization of $\mathbf{w}$. The coefficient γ can be omitted because it would be eliminated by the normalization anyway. Then after the derivation process, we obtain the following approximative Newton iteration:

$$\mathbf{w} \leftarrow \mathbf{w} - [E\{\mathbf{z}g(\mathbf{w}^T\mathbf{z})\} + \beta\mathbf{w}]/[E\{g'(\mathbf{w}^T\mathbf{z})\} + \beta]. \tag{2.28}$$

This algorithm can be further simplified by multiplying both sides of (2.28) by $\beta + E\{g'(w^T z)\}$. This gives, after some simple algebraic simplification:

$$\mathbf{w} \leftarrow E\{\mathbf{z}g(\mathbf{w^T z}) - E\{g'(\mathbf{w^T z})\}\mathbf{w}\}. \tag{2.29}$$

This is the basic fixed-point iteration in fast fixed-point algorithm.

In this section, we have so far estimated only one independent component. This is why these algorithms are sometimes called "one-unit" algorithms. The key to extending the method of maximum non-Gaussianity to estimate more independent component is based on the property that the vectors $\mathbf{w}_i$ corresponding to different independent components are orthogonal after whitening transformation space as shown in the Section 1. To recapitulate, the independence of the components requires that they are uncorrelated, and after whitening transformation space we have $\mathrm{E}\{(\mathbf{w}_i^{\mathbf{T}}\mathbf{z})(\mathbf{w}_j^{\mathbf{T}}\mathbf{z})\} = \mathbf{w}_i^{\mathbf{T}}\mathbf{w}_j$, and therefore uncorrelatedness equals to orthogonality. The $\mathbf{w}_i'$s are in fact the rows of the inverse of the mixing matrix by definition, and these are equal to the columns of the mixing matrix, because by orthogonality $\mathbf{A}^{-1} = \mathbf{A}^T$.

To summarize, in order to estimate several independent components, we need to run any of the one-unit algorithms several times with vectors $\mathbf{w}_1, ..., \mathbf{w}_n$, and, to prevent different vectors from converging to the same maxima, we must orthogonalize the vectors $\mathbf{w}_1, ..., \mathbf{w}_n$ after every iteration. We present in the following different methods for achieving decorrelation.

2.2.7 Orthogonalization Methods

2.2.7.1 Deflationary Orthogonalization

When we have estimated p independent components, or p vectors $\mathbf{w}_1, ..., \mathbf{w}_p$, we alternate the following steps:

1. Choose m, the number of Independent Components to estimate. Set $p \leftarrow 1$.
2. Initialize $\mathbf{w}_p$ (e.g. randomly)
3. Do an iteration of a one-unit algorithm on $\mathbf{w}_p$.

4. Do the following orthogonalization:

$$\mathbf{w}_p \leftarrow \mathbf{w}_p - \sum_{j=1}^{p-1} \left(\mathbf{w}_p^T \mathbf{w}_j\right) \mathbf{w}_j \tag{2.30}$$

5. Normalize $\mathbf{w}_p$ by dividing by its norm.

6. If $\mathbf{w}_p$ has not converged, go back to Step 3.

7. Set $p \leftarrow p + 1$. If p is not greater than the desired number of Independent Components, go back to Step 2.

2.2.7.2 Symmetric Orthogonalization

In certain applications, it may be desirable to use a symmetric decorrelation, in which no vectors are privileged over others. This means that the vectors $\mathbf{w}_i$ are not estimated one by one; instead, they are estimated in parallel. One motivation for this is that the deflationary method has the drawback that estimation errors in the first vectors are cumulated in the subsequent ones by the orthogonalization. Another one is that the symmetric orthogonalization methods enable parallel computation of Independent Components.

Symmetric orthogonalization is done by first doing the iterative steps of the one-unit algorithm on every vector $\mathbf{w}_i$ in parallel, and afterwards orthogonalizing all the $\mathbf{w}_i$ by special symmetric methods. Specifically, we follow these steps:

1. Choose the number of independent components to estimate, say m;
2. Initialize the $\mathbf{w}_i$, $i = 1, ..., m$ (e.g., randomly);
3. Do an iteration of a one-unit algorithm on every $\mathbf{w}_i$ in parallel;
4. Do a symmetric orthogonalization of the matrix $\mathbf{W} = (\mathbf{w}_1, ...\mathbf{w}_m)^T$;
5. If not converged, go back to step 3.

2.2.8 Independent Component Analysis (ICA) and Projection Pursuit

Projection pursuit [M.C. Jones-1987] is a technique developed in statistics for finding interesting projections of multidimensional data. The projection pursuit is usually performed by finding the most non-Gaussian projections of the data. In the formulation of projection pursuit, no data model or assumption about the independent components is made. If the Independent Component Analysis (ICA) model holds, optimizing the non-Gaussianity measures produce independent components; if the model does not hold, then what we get are the projection pursuit directions.

Chapter 3

Independent Component Analysis (ICA) by Maximum Likelihood Estimation

3.1 Deriving the likelihood

3.1.1 Likelihood Estimation

We can easily derive the likelihood for a noise-free Independent Component Analysis (ICA) model as in (1.4). Let us briefly recall the basic Independent Component Analysis (ICA) model.

Denote by $\mathbf{x}$ the random vector whose elements are the mixtures $x_1, ..., x_n$ and likewise by $\mathbf{s}$ be the random vector with elements $s_1, ..., s_n$. Let us denote by $\mathbf{A}$ the matrix with elements $\mathbf{a}_{ij}$. Thus, the mixing model can be written as [M. Gaeta-1990]

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{3.1}$$

whose density p_x of the mixture vector can be formulated as

$$p_x(\mathbf{x}) = |\det \mathbf{B}| p_s(\mathbf{s}) = |\det \mathbf{B}| \prod_i p_i(s_i) \tag{3.2}$$

where $\mathbf{B} = \mathbf{A}^{-1}$, and the p_i denote the densities of the independent components. Its expression can be given as a function of $\mathbf{B} = (\mathbf{b}_1, ..., \mathbf{b}_n)^T$and $\mathbf{x}$ as follows:

$$p_x(\mathbf{x}) = |\det \mathbf{B}| \prod_i p_i(\mathbf{b}_i^T \mathbf{x}) \tag{3.3}$$

Assume that we have T observations of $\mathbf{x}$, denoted by $\mathbf{x}(1), \mathbf{x}(2), ..., \mathbf{x}(T)$. Then the likelihood can be obtained as the product of this density evaluated at T points. This is denoted by L and considered as a function of $\mathbf{B}$:

$$L(\mathbf{B}) = \prod_{\mathbf{t=1}}^{\mathbf{T}} \prod_{\mathbf{i=1}}^{\mathbf{n}} p_i(\mathbf{b}_i^T \mathbf{x}(t)) |\det \mathbf{B}| \tag{3.4}$$

The log-likelihood is given by

$$\log L(\mathbf{B}) = \sum_{t=1}^{T} \sum_{i=1}^{n} \log p_i(\mathbf{b}_i^T \mathbf{x}(t)) + T \log |\det \mathbf{B}| \tag{3.5}$$

To simplify notation and to make it consistent to what was used in the previous section, we can denote the sum over the sample index t by an expectation operator, and divide the likelihood by T to obtain [D.-T.Pham-1992]

$$\frac{1}{T}\log L(\mathbf{B}) = E\{\sum_{i=1}^{n}\log p_i(\mathbf{b}_i^T\mathbf{x})\} + \log|\det \mathbf{B}| \tag{3.6}$$

The expectation here is not the theoretical expectation, but an average computed from the observed sample. Surely, in the algorithms the expectations are eventually replaced by sample averages.

3.1.2 Estimation of the densities

We have expressed the likelihood as a function of the parameters of the model, which are elements of the mixing matrix. And we used the elements of the inverse **B** of the mixing matrix which can be directly computed from its inverse.

Still, we need to estimate the densities of the independent components in the Independent Component Analysis (ICA) model. In fact, the likelihood is a function of these densities too. However, the estimation becomes so complicated that it's hard for us to solve. This is because the estimation of densities is a nonparametric problem generally. Here, nonparametric problems mean that it can't be reduced to the estimation of a finite parameter set. Thus, the estimation of the Independent Component Analysis (ICA) model has also a nonparametric part, which is why the estimation is sometimes called "semi-parametric".

Since nonparametric problems have an infinite number of parameters, which are hardest to estimate, we'd like to avoid this kind of estimation in the Independent Component Analysis (ICA). There are two ways to avoid it.

First, in some cases that we have known the densities of the independent components already, we can just use these prior densities in the likelihood. Then the likelihood will really be a function of $\mathbf{B}$ only. If reasonably small errors in the specification of these prior densities have little influence on the estimator, it will be a suitable result.

A second approach is that we can approximate the densities of the independent components by a family of densities that are specified by a limited number of parameters. If it is the case that it's possible to use a very simple family of densities to estimate the Independent Component Analysis (ICA) model for any densities p_i, we will get a simple solution. For instance, we may use an extremely simple parameterization of the p_i, consisting of the choice between two densities, that is, a single binary parameter.

3.1.3 A simple density family

It turns out that in maximum likelihood estimation, it's sufficient to use just two approximations of the density of an independent component. For each component, we just need to determine which one of the two approximations is better. It shows that we can make small errors when we fix the densities of the components, since it's sufficient for us to use a density that is in the same half of the space of probability densities. Also, it shows that we can estimate the independent components using models made up of only two densities.

The validity of these approaches in shown in the following theorem. This theorem is basically a corollary of the stability theorem in (2.2.5).

Theorem 3.1.1. *Denote by $\tilde{p}_i$ the assumed densities of the independent components, and*

$$g_i(s_i) = \frac{\partial}{\partial s_i} \log \tilde{p}_i(s_i) = \frac{\tilde{p_i'(s_i)}}{\tilde{p_i(s_i)}} \tag{3.7}$$

Constrain the estimates of the independent components $y_i = \mathbf{b}_i^T \mathbf{x}$ to be uncorrelated and to have unit variance. Then the ML estimator is locally consistent, if the assumed densities $\tilde{p}_i$ fulfill

$$E\{s_i g_i(s_i) - g'(s_i)\} > 0 \tag{3.8}$$

for all i.

Since sufficiently small changes don't change the sign in (3.8), the theorem above rigorously shows that small misspecifications in the densities p_i do not affect the local consistency of the maximum likelihood estimator.

Moreover, the theorem show show to construct families consisting of only two densities, so that the condition in (3.8) is true for one of these densities.

Example 3.1.1. Consider the following log-densities:

$$\log \tilde{p}_i^+(s) = \alpha_1 - 2 \log cosh(s) \tag{3.9}$$

$$\log \tilde{p}_i^-(s) = \alpha_2 - [s^2/2 - \log \cosh(s)] \tag{3.10}$$

where α_1, α_2 are positive parameters that are fixed so as to make these two functions logarithms of probability densities. Actually, these constants can be ignored in the following. The factor 2 in (3.9) is not important, but it's usually used here; also, the factor 1/2 in (3.10) could be changed.

The motivation for these functions is that $\tilde{p}_i^+$ is a supergaussian density, since the $\log\cosh$ function is close to the absolute value that would give the Laplacian density. The density given by $\tilde{p}_i^-$ is subgaussian, because it is like a gaussian log-density, $-s^2/2$ plus a constant, that has been somewhat "flattend" by the $\log\cosh$ function.

Simple computations show that the value of the non-polynomial moment in (3.8) is for $\tilde{p}_i^+$

$$2E\{-\tanh(s_i)s_i + (1 - \tanh(s_i)^2)\} \tag{3.11}$$

and for $\tilde{p}_i^-$ it is

$$E\{\tanh(s_i)s_i - (1 - \tanh(s_i)^2)\} \tag{3.12}$$

since the derivative of $\tanh(s)$ equals $1 - \tanh(s)^2$, and $E\{s_i^2\} = 1$ by definition. We see that the signs of these expressions are always opposite. Thus, for practically any distributions of the s_i, one of these functions fulfills the condition, i.e., as the desired sign, and estimation is possible. Of course, for some distributions of the s_i the non-polynomial moment in the condition could be zero, which corresponds to the case of zero kurtosis in cumulant-based estimation; such cases can be considered to be very rare.

Thus we can just compute the non-polynomial moments for the two prior distributions in (3.9) and (3.10), and choose the one that fulfills the stability condition in (3.8). This can be done on-line during the maximization of the likelihood. This always provides a locally consistent estimator, and solves the problem of semi-parametric estimation.

3.2 Algorithms For Maximum Likelihood Estimation

3.2.1 Gradient algorithms

Algorithm 1. *The Bell-Sejnowski algorithm*

The simplest algorithms for maximizing likelihood are obtained by gradient methods. Using the gradient method mentioned in last section, the stochastic gradient of the log-likelihood in (3.6) as:

$$\frac{1}{T}\frac{\partial \log L}{\partial \mathbf{B}} = [\mathbf{B}^T]^{-1} + E\{g(\mathbf{B}\mathbf{x})\mathbf{x}^T\} \tag{3.13}$$

here, $g(\mathbf{y}) = (g_i(y_i), ..., g_n(y_n))$ is a component-wise vector function that consists of the so-called score functions g_i of the distributions of s_i, defined as

$$g_i = (\log p_i)' = \frac{p_i^{'}}{p_i} \tag{3.14}$$

This immediately gives the following algorithm for Maximum Likelihood estimation:

$$\Delta\mathbf{B} \propto [\mathbf{B}^T]^{-1} + E\{g(\mathbf{Bx})\mathbf{x}^T\} \tag{3.15}$$

A stochastic version of this algorithm could be used as well. This means that the expectation is omitted, and in each step of the algorithm, only one data point is used:

$$\Delta\mathbf{B} \propto [\mathbf{B}^T]^{-1} + g(\mathbf{Bx})\mathbf{x}^T. \tag{3.16}$$

This algorithm is often called the Bell-Sejnowski algorithm [A.J.Bell et al-1995]. The algorithm in Eq. (3.15) converges very slowly, however, especially due to the inversion of the matrix **B** that is needed in every step. The convergence can be improved by whitening the data, and especially by using the natural gradient.

Algorithm 2. *The natural gradient algorithm*

The natural gradient method [S.-I. Amari et al-1996] simplifies the maximization of the likelihood considerably, and makes it better conditioned. The principal of the natural gradient is based on the geometrical structure of the parameter space, and is related to the principal of relative gradient, which uses the Lie group structure of the Independent Component Analysis (ICA) problem. In the case of basic Independent

Component Analysis (ICA) model, both of these principles amount to multiplying the right-hand side of (3.15) by $\mathbf{B}^T\mathbf{B}$. Thus we obtain

$$\Delta\mathbf{B} \propto (\mathbf{I} + E\{g(\mathbf{y})\mathbf{y}^T\})\mathbf{B} \tag{3.17}$$

This algorithm can be interpreted as nonlinear decorrelation. The idea is that the algorithm converges when $E\{g(\mathbf{y})\mathbf{y}^T\} = \mathbf{I}$, which means that y_i and $g_i(y_i)$ are uncorrelated for $i \neq j$.

3.2.2 A Fast fixed-point algorithm

In Eq. (2.28) in previous section we had the following form of the fast fixed-point algorithm (for whitened data):

$$\mathbf{w} \leftarrow \mathbf{w} - [E\{\mathbf{z}g(\mathbf{w}^T\mathbf{z})\} + \beta\mathbf{w}]/[E\{g'(\mathbf{w}^T\mathbf{z})\} + \beta] \tag{3.18}$$

where β can be computed from (2.28) as $\beta = -E\{y_i g(y_i)\}$. If we write this in matrix form, we obtain:

$$\mathbf{W} \leftarrow \mathbf{W} + diag(\alpha_i)[diag(\beta_i) + E\{g(\mathbf{y})\mathbf{y}^T\}]\mathbf{W} \tag{3.19}$$

where α_i is defined as $-1/E\{g'(\mathbf{w}^T\mathbf{z}) + \beta_i\})$, and $\mathbf{y} = \mathbf{W}\mathbf{z}$. To express this using non-whitened data, as we have done in this section, it is enough to multiply both sides of (3.19) from the right by the whitening matrix. This means simply that we replace the $\mathbf{W}$ by $\mathbf{B}$, since we have $\mathbf{Wz} = \mathbf{WVx}$ which implies $\mathbf{B} = \mathbf{WV}$.

Thus, we obtain the basic iteration of fast fixed-point algorithm as:

$$\mathbf{B} \leftarrow \mathbf{B} + diag(\alpha_i)[diag(\beta_i) + E\{g(\mathbf{y})\mathbf{y}^T\}]\mathbf{B} \tag{3.20}$$

where $\mathbf{y} = \mathbf{Bx}$, $\beta_i = -E\{y_i g(y_i)\}$, and $\alpha_i = -1/(\beta_i + E\{g'(y_i)\})$.

After every step, the matrix $\mathbf{B}$ must be projected on the set of whitening matrices. This can be accomplished by the classic method involving matrix square roots,

$$\mathbf{B} \leftarrow (\mathbf{BCB}^T)^{-1/2}\mathbf{B} \tag{3.21}$$

where $\mathbf{C} = E\{\mathbf{xx}^T\}$ is the covariance matrix of the data.

3.3 The Infomax Principle

The infomax principle [J.-F. Cardoso-1997] is very closely related to maximum likelihood principle for Independent Component Analysis (ICA). It is based on maximizing the output entropy, or information flow, of a neural network with nonlinear outputs. Hence the name infomax.

Suppose that $\mathbf{x}$ is the input to the neural network whose outputs are of the form

$$y_i = \phi_i(\mathbf{b}_i^T\mathbf{x}) + \mathbf{n} \tag{3.22}$$

where the ϕ_i are some nonlinear scalar functions, and the $\mathbf{b}_i$ are the weight vectors of the neurons. The vector $\mathbf{n}$ is additive gaussian white noise. One then wants to maximize the entropy of the outputs:

$$H(\mathbf{y}) = H(\phi_1(\mathbf{b}_1^T\mathbf{x}), ..., \phi_n(\mathbf{b}_n^T\mathbf{x})) \tag{3.23}$$

This can be motivated by considering information flow in a neural network. Efficient information transmission requires that we maximize the mutual information between the inputs $\mathbf{x}$ and the outputs $\mathbf{y}$. We have the entropy of a transformation formula,

$$H(\mathbf{y}) = H(\mathbf{x}) + E\{\log|\det J\mathbf{f}(\mathbf{x})|\} \tag{3.24}$$

where $\mathbf{y}$ is an invertible transformation of the random vector $\mathbf{x}$, say $\mathbf{y} = \mathbf{f}(\mathbf{x})$.

Using formula above, we obtain

$$H(\phi_1(\mathbf{b}_1^T\mathbf{x}), ..., \phi_n(\mathbf{b}_n^T\mathbf{x})) = H(\mathbf{x}) + E\{\log|\det \frac{\partial \mathbf{F}}{\partial \mathbf{B}}(\mathbf{x})|\} \tag{3.25}$$

where $\mathbf{F}(\mathbf{x}) = (\phi_1(\mathbf{w}_1^T\mathbf{x}), ..., \phi_n(\mathbf{w}_n^T\mathbf{x}))$ denotes the function defined by the neural network. We can simply calculate the derivative to obtain

$$E\{\log|\det \frac{\partial \mathbf{F}}{\partial \mathbf{B}}(\mathbf{x})|\} = \sum_i E\{\log \phi_i'(\mathbf{b}_i^T\mathbf{x})\} + \log|\det \mathbf{B}| \tag{3.26}$$

Now, we can see the output entropy is of the same form as the expectation of the likelihood as in (3.6). So if the nonlinearities ϕ_i used in the neural network are chosen as the cumulative distribution functions corresponding to the densities p_i, i.e., $\phi_i'(.) = p_i(.)$, the output entropy is actually equal to the likelihood. This means that infomax is equivalent to maximum likelihood estimation.

Chapter 4

Independent Component Analysis (ICA) by Minimization of Mutual Information

4.1 Defining Independent Component Analysis (ICA) by mutual information

4.1.1 Information-theoretic concepts

Briefly recall the basic definitions of information theory presented in (2.2.4). The differential entropy H of a random vector $\mathbf{y}$ with density $p(\mathbf{y})$ is defined as:

$$H(\mathbf{y}) = -\int p(\mathbf{y}) \log p(\mathbf{y}) d\mathbf{y} \tag{4.1}$$

Entropy is closely related to the code length of the random vector. A normalized version of entropy is given by negentropy J, which is defined as follows

$$J(\mathbf{y}) = H(\mathbf{y}_{gauss}) - H(\mathbf{y}) \tag{4.2}$$

where $\mathbf{y}_{gauss}$ is a gaussian random vector of the same covariance matrix as $\mathbf{y}$. Negentropy is always nonnegative, and zero only for gaussian random vectors. Mutual information I between m random variables, y_i, $i = 1, ..., m$ is defined as follows

$$I(y_1, y_2, ..., y_m) = \sum_{i=1}^{m} H(y_i) - H(\mathbf{y}). \tag{4.3}$$

4.1.2 Mutual information as measure of dependence

Mutual information [P.Comon-1994] considers the whole dependence structure of the variables, not just the covariance, like principal component analysis and related methods. It's always nonnegative, and zero if and only if the variables are statistically independent.

Therefore, mutual information is used to find the Independent Component Analysis (ICA) representation. This approach is an alternative to the model estimation approach. We define the Independent Component Analysis (ICA) of a random vector $\mathbf{x}$ as an invertible transformation:

$$\mathbf{s} = \mathbf{B}\mathbf{x} \tag{4.4}$$

where the matrix $\mathbf{B}$ is determined so that the mutual information of the transformed components s_i is minimized. If the data follows the Independent Component Analysis (ICA) model, this allows estimation of the data model. Otherwise, in this definition, there's no need to assume that the data follows the model. Actually, minimization of mutual information can give the maximally independent components in any case.

4.2 Mutual information and non-Gaussianity

Before discussing the connection between the mutual information and non-Gaussianity, let's do some derivation about the entropy of the linear transformation of the random vector $\mathbf{x}$.

Consider an invertible transformation of the random vector $\mathbf{x}$, say

$$\mathbf{y} = \mathbf{f}(\mathbf{x}) \tag{4.5}$$

In the following, we want to show the connection between the entropy of $\mathbf{y}$ and that of $\mathbf{x}$.

Denote by $J\mathbf{f}(\xi)$ the Jacobian matrix of the function $\mathbf{f}$, i.e., the matrix of the partial derivatives of $\mathbf{f}$ at point ξ. The classic relation between the density p_y of $\mathbf{y}$ and the density p_x of $\mathbf{x}$, can then be formulated as

$$p_y(\eta) = p_x(\mathbf{f}^{-1}(\eta))|\det J\mathbf{f}(\mathbf{f}^{-1}(\eta))|^{-1} \tag{4.6}$$

Now, expressing the entropy as an expectation

$$H(\mathbf{y}) = -E\{\log p_y(\mathbf{y})\} \tag{4.7}$$

we obtain

$$E\{\log p_y(\mathbf{y})\} = E\{\log[p_x(\mathbf{f}^{-1}(\mathbf{y}))|\det J\mathbf{f}(\mathbf{f}^{-1}(\mathbf{y}))|^{-1}]\} \tag{4.8}$$

$$= E\{\log[p_x(\mathbf{x})|\det J\mathbf{f}(\mathbf{x})|^{-1}]\} = E\{\log p_x(\mathbf{x})\} - E\{\log|\det J\mathbf{f}(\mathbf{x})|\} \tag{4.9}$$

Thus we obtain the relation between the entropies as

$$H(\mathbf{y}) = H(\mathbf{x}) + E\{\log|\det J\mathbf{f}(\mathbf{x})|\} \tag{4.10}$$

Then, by using the formula for the differential entropy of a transformation as given in (4.10), we can get a corresponding result for mutual information. We have for an invertible linear transformation $\mathbf{y} = \mathbf{B}\mathbf{x}$:

$$I(y_1, y_2, ..., y_n) = \sum_i H(y_i) - H(\mathbf{x}) - \log|\det \mathbf{B}| \tag{4.11}$$

Next, let's consider what happens if we constrain the y_i to be uncorrelated and of unit variance. This means $E\{\mathbf{y}\mathbf{y}^T\} = \mathbf{B}E\{\mathbf{x}\mathbf{x}^T\}\mathbf{B}^T = \mathbf{I}$, which implies

$$\det \mathbf{I} = 1 = \det(\mathbf{B}E\{\mathbf{x}\mathbf{x}^T\}\mathbf{B}^T) = (\det \mathbf{B})(\det E\{\mathbf{x}\mathbf{x}^T\})(\det \mathbf{B}^T) \tag{4.12}$$

and this implies that $\det \mathbf{B}$ must be constant since $\det E\{\mathbf{x}\mathbf{x}^T\}$ does not depend on $\mathbf{B}$. Moreover, for y_i of unit variance, entropy and negentropy differ only by a constant and the sign, as can be seen (4.2), Thus we obtain,

$$I(y_1, y_2, ...y_n) = \mathbf{const.} - \sum_i J(y_i) \tag{4.13}$$

where the constant term does not depend on $\mathbf{B}$. This shows the basic relation between negentropy and mutual information.

We see in (4.13) that finding an invertible linear transformation $\mathbf{B}$ that minimizes the mutual information equals to finding directions in which the negentropy is maximized. We have seen previously that negentropy is a measure of non-Gaussianity. Thus, we have the conclusion as stated below.

Conclusion 1. Independent Component Analysis (ICA) estimation by minimization of mutual information equals to maximizing the sum of non-Gaussianities of the estimates of the independent components, when the estimates are constrained to be uncorrelated.

However, there exist some important differences between these two criteria.

Note 1. The deflationary, i.e., one-by-one, estimation of the independent components is available in negentropy since we are able to look for the maxima of non-Gaussianity of a single projection $\mathbf{b}^T\mathbf{x}$. But, it's impossible with mutual information.

While using non-Gaussianity, we force the estimates of the independent components to be uncorrelated. This is not necessary since we can use the form in (4.11) directly while using mutual information.

4.3 Mutual information and likelihood

Mutual information and likelihood are closely related. To see the relation between likelihood and mutual information, consider the expectation of the log-likelihood in (3.5):

$$\frac{1}{T}E\{\log L(\mathbf{B})\} = \sum_{i=1}^{n} E\{\log p_i(\mathbf{b}_i^T\mathbf{x})\} + \log|\det \mathbf{B}| \tag{4.14}$$

If the p_i were equal to the actual probability density functions of $\mathbf{b}_i^T\mathbf{x}$, the first term would be equal to $-\sum_i H(\mathbf{b}_i^T\mathbf{x})$. Thus the likelihood would be equal, up to an additive constant given by the total entropy of $\mathbf{x}$, to the negative of mutual information as given in Eq. (4.11).

The connection may be just as strong, or even stronger practically. Since in practice we don't know the distributions of the independent components that are needed in maximum likelihood estimation. So a reasonable approach is to estimate the density of $\mathbf{b}_i^T\mathbf{x}$ as part of the maximum likelihood estimation method, and use this as an approximation of the density of s_i. This is what we did in Section 3. Then the p_i in this approximation of likelihood are indeed equal to the actual probability density functions $\mathbf{b}_i^T\mathbf{x}$. Thus equivalency holds.

Conversely, to approximate mutual information, we could take a fixed approximation of the densities y_i, and plug this in the definition of entropy. Denote the probability density functions by $G_i(y_i) = \log p_i(y_i)$. Then we could approximate (4.11) as

$$I(y_1, y_2, ..., y_n) = -\sum_i E\{G_i(y_i)\} - \log|\det\mathbf{B}| - H(\mathbf{x}) \tag{4.15}$$

Chapter 5

R Implementation Examples using FastICA Algorithm

5.1 Un-mixing two independent signals

The original matrix **s** which consists of two independent signals (columns) are given. The first signal (column) is made up with sin series and the second one is repeated series from -0.99 to 1. Then the mixing matrix which is denoted by **A** is shown below:

$$\mathbf{A} = \begin{bmatrix} 0.3019 & -0.5539 \\ 0.7567 & 0.5673 \end{bmatrix}$$

After doing matrix multiplication between matrix we obtained **x**, the mixture of two independent signals which is an 1000×2 matrix. What we next do is that apply fastICA function in **R** to un-mixing the mixture **x**. The result is as shown in the

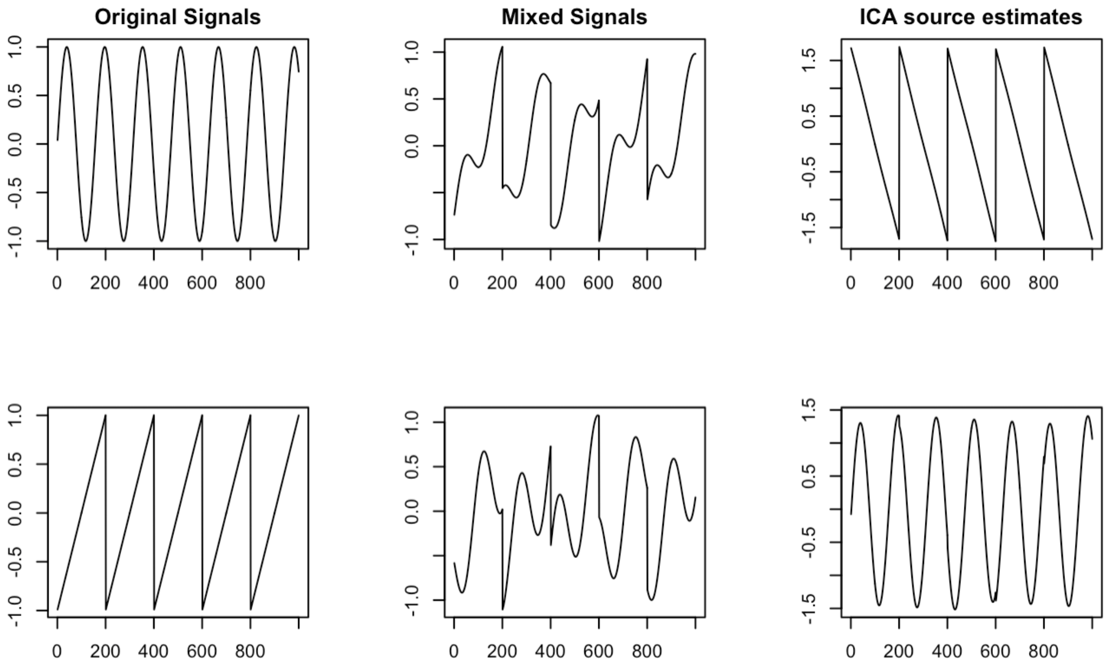

Figure 5.1: Result of implementation of the fastICA algorithm

Figure 5.1.

We can see from the Figure 5.1, the shape of the signals we obtained after implementing the fastICA algorithm method is so similar that we can conclude that the result is not bad. But, the scale here has changed and the order of the original signals has been switched. In this **R** function, the negentropy has been chosen as a measure of non-Gaussianity of the data, and $\log\cosh$ function has been used for the approximation of the negentropy. We has introduced that the convergence of the fastICA algorithm is much faster than gradient based algorithm, which is cubic. In this example, we can find from the **R** output, only after three iterations, the tolerance is small enough, that is, $3.036508e-07$ so that we can see the iteration converges.

5.2 Un-mixing two mixed independent uniformly distributed components

The original matrix **s** is two uniformly distributed components. And the mixing matrix A is denoted by:

$$\mathbf{A} = \begin{bmatrix} 2 & 2 \\ -1 & 3 \end{bmatrix}$$

After doing matrix multiplication between matrix we obtained **x**, the mixture of two uniformly independent signals which is a 5000×2 matrix. As the previous example, what we will do is that apply fastICA function in **R** to un-mixing the mixture **x**. The result is as shown in the following.

What we obtained is shown in the Figure 5.2. The first graph indicates pre-processed data. Second one is the plot of two independent uniformly distributed components.

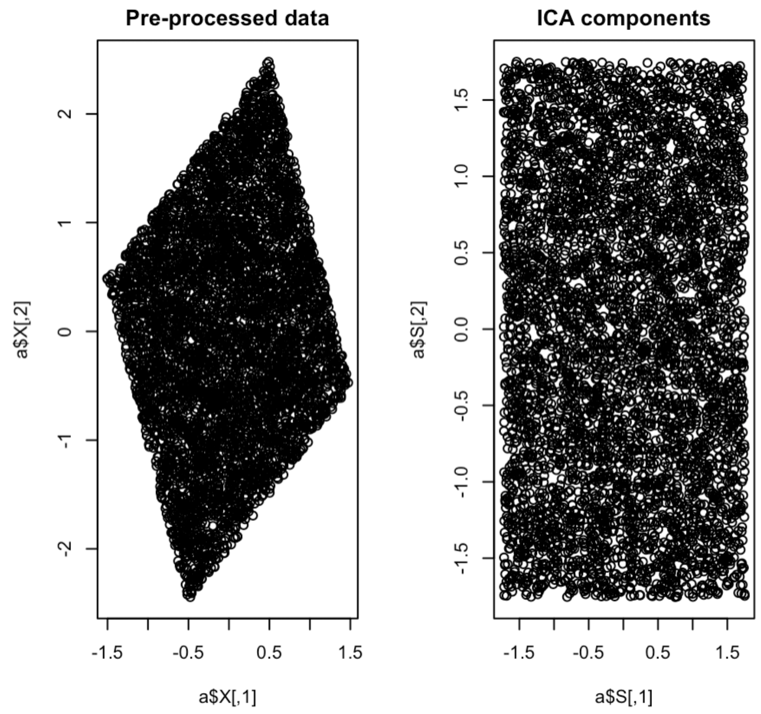

Figure 5.2: Un-mixing two mixed independent uniforms

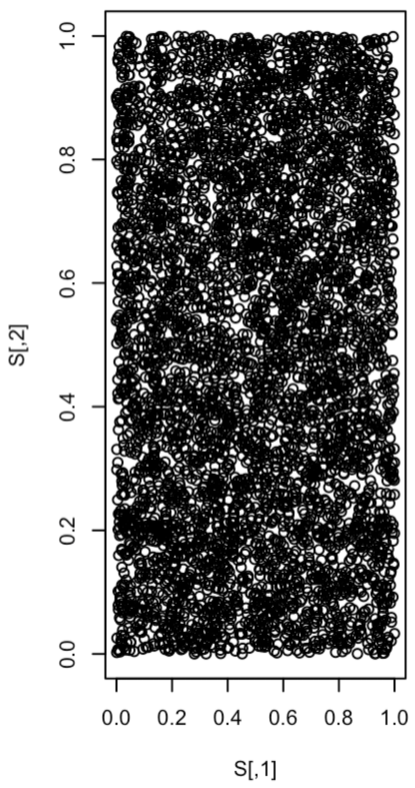

Figure 5.3: Plot of the original two independent uniforms

To compare the result after implementing the fastICA algorithm method with the original one shown in Figure 5.3, the shape is so similar even though the graph scale changes. Here again, the negentropy is the measure of non-Gaussianity of the data, and $\log\cosh$ function has been used for the approximation of the negentropy. We can find through the **R** output, the tolerance is small enough, which is, 0.000004 after three iteration steps so that we conclude the iteration converges.

Bibliography

[1] C. Jutten and J. He´rault. Blind separation of sources, partI:An adaptive algorithm based on neuromimetic architecture. Signal Processing, 24:1–10, 1991.

[2] M. Girolami. Self-Organising Neural Networks - Independent Component Analysis and Blind Source Separation. Springer-Verlag, 1999.

[3] J.-F. Cardoso. Source separation using higher order moments. In Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'89), pages 2109–2112, Glasgow, UK, 1989.

[4] N. Delfosse and P. Loubaton. Adaptive blind separation of independent sources: a deflation approach. Signal Processing, 45:59–83, 1995.

[5] A. Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. IEEE Trans. on Neural Networks, 10(3):626–634, 1999.

[6] A. Hyvarinen and E. Oja. Independent component analysis by general nonlinear Hebbian- like learning rules. Signal Processing, 64(3):301–313, 1998.

[7] M.C. Jones and R. Sibson. What is projection pursuit? J. of the Royal Statistical Society, Ser. A, 150:1–36, 1987.

[8] M. Gaeta and J.-L. Lacoume. Source separation without prior knowledge: the maximum likelihood solution. In Proc. EUSIPCO'90, pages 621–624, 1990.

[9] D.-T. Pham, P. Garrat, and C. Jutten. Separation of a mixture of independent sources through a maximum likelihood approach. In Proc. EUSIPCO, pages 771–774, 1992.

[10] A. J. Bell and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. Neural Computation, 7:1129–1159, 1995.

[11] J.-F. Cardoso. Infomax and maximum likelihood for source separation. IEEE Letters on Signal Processing, 4:112–114, 1997.

[12] S.-I. Amari, A. Cichocki, and H.H. Yang. A new learning algorithm for blind source separation. In Advances in Neural Information Processing Systems 8, pages 757–763. MIT Press, 1996.

[13] P. Comon. Independent component analysis—a new concept? Signal Processing, 36:287–314, 1994.

[14] A. Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. IEEE Trans. on Neural Networks, 10(3):626634, 1999.