Independent component analysis and beyond

Dissertation

zur Erlangung des akademischen Grades
doctor rerum naturalium
– Dr. rer. nat. –

eingereicht an der
Mathematisch-Naturwissenschaftlichen Fakultät
der Universität Potsdam

von
Stefan Harmeling

Potsdam, im August 2004

Summary

Independent component analysis (ICA) is a tool for statistical data analysis and signal processing that is able to decompose multivariate signals into their underlying source components. Although the classical ICA model is highly useful, there are many real-world applications that require powerful extensions of ICA. This thesis presents new methods that extend the functionality of ICA.

Reliability and grouping of independent components with noise injection. Usually noise is considered to be destructive. We present a new method that constructively injects noise to assess the reliability and the grouping structure of empirical independent component estimates. We generalize hereby earlier work on reliability assessment based on bootstrap to arbitrary ICA algorithms. Our method can be viewed as a Monte-Carlo-style approximation of the curvature of some performance measure at the solution. Simulations using artificial and real-world data validate our approach.

Robust and overcomplete ICA with inlier detection. Classical ICA algorithms are often sensitive to outliers. We present a new ICA algorithm for super-Gaussian sources that is based on an index for outlier detection that uses nearest neighbor methods. The outlier index is not employed to remove outliers but instead directly to find inliers—the data points in the most concentrated regions—which represent the ICA directions for super-Gaussian source signals. Our inlier-based approach is by construction robust against outliers and can be naturally applied to the overcomplete ICA problem, in which there are more sources than sensors. A comparison of our new method with classical algorithms—in terms of robustness—and a comprehensive empirical analysis of its performance—with respect to dimensionality, number of sources and number of data points—underlines its key advantages.

Nonlinear ICA with kernel methods. We present a kernel-based algorithm for nonlinear ICA that uses kernel feature spaces to approximate nonlinearities. Applying linear ICA based on time structure in the resulting high-dimensional spaces can unmix strongly nonlinear mixtures. The key is to use some dimensionality reduction technique to make the application of ICA methods computationally and numerically tractable. Experiments demonstrate the excellent performance and efficiency of our algorithm for several problems of nonlinear ICA.

Keywords: ICA, reliability assessment, robust ICA, overcomplete ICA, outlier detection, nonlinear ICA, kernel PCA, kernel methods.

Zusammenfassung

"Independent component analysis" (ICA) ist ein Werkzeug der statistischen Datenanalyse und Signalverarbeitung, welches multivariate Signale in ihre Quellkomponenten zerlegen kann. Obwohl das klassische ICA Modell sehr nützlich ist, gibt es viele Anwendungen, die Erweiterungen von ICA erfordern. In dieser Dissertation präsentieren wir neue Verfahren, die die Funktionalität von ICA erweitern.

Zuverlässigkeit und Gruppierung von ICA Komponenten durch Hinzufügung von Rauschen ("noise injection"). Normalerweise, ist Rauschen in Signalen von Nachteil. Wir präsentieren eine neues Verfahren, das Rauschen konstruktiv zum Signal addiert, um die Zuverlässigkeit und Gruppenstruktur von empirischen ICA Komponenten zu bestimmen. Hierbei verallgemeinern wir frühere, auf "bootstrap" basierende Arbeiten zur Zuverlässigkeitsbestimmung auf beliebige ICA Algorithmen. Unsere Methode kann man als Monte-Carlo-Approximation der Krümmung gewisser Evaluierungsfunktionen an der Lösung auffassen. Simulationen mit künstlichen und echten Daten validieren unseren Ansatz.

Robuste und überbestimmte ("over-complete") ICA durch Ausreissererkennung. Klassische ICA Algorithmen sind oft Ausreisser-anfällig. Wir präsentieren einen neuen ICA Algorithmus für supergausssche Quellsignale, der auf Ausreisserindizes basiert, welche "nearest-neighbor" Methoden verwenden. Der Ausreisserindex wird jedoch nicht dazu benutzt, um Ausreisser zu entfernen, sondern direkt um "Inlier" (das Gegenteil von "Outlier", d.h. von Ausreissern) zu finden, welche dann die ICA Richtungen für supergausssche Quellsignale darstellen. Unser auf "inlier" basierende Ansatz ist nach Konstruktion robust gegenüber Ausreissern und kann auf Probleme der überbestimmten ICA angewandt werden, welche mehr Quellen als Sensoren haben. Ein Vergleich unserer Methode mit anderen klassischen ICA-Verfahren zeigt empirisch die Robustheit unseres Verfahrens gegenüber Ausreissern und zusätzlichem Rauschen.

Nichtlineare ICA mit Kernmethoden. Wir präsentieren ein kernbasiertes Verfahren zur nichtlinearen ICA, das mithilfe von Kernmerkmalsräumen die Nichtlinearitäten approximiert. Die Anwendung von linearer ICA, die auf Zeitstruktur basiert, in den resultierenden hochdimensionalen Räumen entmischt stark nichtlineare Mischungen. Der Schlüssel ist der Einsatz von Methoden zur Dimensionsreduktion, um die Anwendung von ICA Verfahren überhaupt möglich zu machen. Experimente demonstrieren die Leistungsfähigkeit und Effizienz unseres Verfahrens für verschiedene Probleme nichtlinearer ICA.

Acknowledgements

First of all, I would like to express my thanks to Prof. Dr. Klaus-Robert Müller for supervising my research and supporting me in many respects. Furthermore, I would like to thank Prof. Dr. Luis Almeida and Dr. Aapo Hyvärinen for agreeing to be "Gutachter" of my dissertation.

The work of this dissertation has been done at the Fraunhofer institute FIRST (former GMD FIRST) in Berlin. I would like to thank all current and former members of this group for creating an open research atmosphere and for many productive discussions, including Dr. Gilles Blanchard, Dr. Benjamin Blankertz, Mikio Braun, Guido Dornhege, Julian Laub, Dr. Motoaki Kawanabe, Dr. Jens Kohlmorgen, Matthias Krauledat, Roman Krepki, Dr. Pavel Laskov, Steven Lemm, Frank Meinecke, Dr. Sebastian Mika, Dr. Gunnar Rätsch, Christin Schäfer, Rolf Schulz, Dr. Anton Schwaighofer, Sören Sonnenburg, Dr. Masashi Sugiyama, Dr. Olaf Weiss, and Andreas Ziehe. In particular, I thank my roommate Dr. Pavel Laskov who helped me often in many ways and the students Paul von Bünau and Carolina Pizano.

Furthermore, I would like to thank Christin Schäfer for making the distances between Adlershof, Schöneberg and Golm much shorter than they might appear by looking at the schedules of public transportation.

Notably, I would like to thank my colleagues that have proof-read parts, Dr. Pavel Laskov, Steven Lemm, Frank Meinecke, Dr. Sebastian Mika, Dr. Motoaki Kawanabe.

Several other people contributed to this thesis in one way or another. In particular, I thank the co-authors of my publications, Dr. Benjamin Blankertz, Guido Dornhege, Dr. Motoaki Kawanabe, Frank Meinecke, Prof. Dr. Klaus-Robert Müller, Dr. David Tax and Andreas Ziehe, for fruitful collaborations and allowing me use parts of our articles as a basis for this thesis. Chapter 3 is based on [37, 36], Chapter 4 contains material of [35, 69, 68], and Chapter 5 is based on [41, 40].

I gratefully acknowledge partial support from DFG grants (DFG SFB 618-B4, JA 379/9-2, MU 987/1-1), from EU-project BLISS (IST-1999-14190) and from the EU-PASCAL network of excellence (IST-2002-506778).

Finally, I would like to thank my parents and Dr. Simone Wissing.

Contents

1 Introduction

Suppose we are having a conversation at a crowded cocktail party. It is usually no problem to focus on the person you are talking to, although our two ears are receiving a wild mixture of different sounds originating from various sources: for example, the conversation of the people next to us or the stereo system playing background music. Despite all the background noise the brain enables us to understand the person we are trying to listen to.

Replace the two ears with microphones and the brain with a computer. Can we program a computer such that it separates the microphone recordings into the different sound sources of the cocktail party? Can it single out the words of the person in front of us? This is the cocktail-party problem which is quite difficult to solve, but it illustrates the goal of blind source separation (BSS): decompose signals that have been recorded by an array of sensors (i.e. multichannel recordings) into the underlying sources. This source separation problem is called blind because neither the mixing process nor the characteristics of the source signals are known.

The BSS problem in the real world

The BSS problem arises in diverse situations: for example, a doctor records the electrocardiogram (ECG) of a pregnant woman with several electrodes located at her abdomen and her thorax in order to examine the heart rhythm of the fetus. Besides other sources the recorded signals contain the heartbeat of the mother and also—with much smaller amplitude—the heartbeat of the fetus. The BSS problem in this situation is to separate the signal generated by the fetus from the heartbeat of the mother.

In another medical context, neurologists monitor the electrical activity of the cortex with an electroencephalogram (EEG) in order to study the brain patterns evoked by different stimuli. Current EEG systems record simultaneously up to hundreds of electrodes. The obtained signals are a mixture of the activity of the different areas in the brain but also of artifacts such as the heartbeat or movements of the eyeballs. The BSS problem is to remove the artifacts and to decompose the EEG signals into signals originating from specific regions of the brain.

In a chemical plant, numerous sensors monitor the production process. For quality control and much more importantly, for early warning systems, these sensor recordings can be combined, which leads to the BSS problem of finding a clear representation of the recorded data by identifying the relevant factors.

Principal component analysis (PCA) belongs to the standard techniques of statistical data analysis. By making use of the correlations between simultaneously recorded signals, PCA is able to obtain a representation with less redundancy. However, it can

only identify an orthogonal basis of the subspace that contains the signals but it can not determine the directions of the sources inside this subspace. Thus PCA does not solve the BSS problem.

Independent component analysis (ICA) solves several instances of the BSS problem by taking into account higher-order statistics which are ignored by PCA that relies only on second-order statistics. ICA has been proposed in the 1980's ([42, 43, 7]; see [55] for a detailed history of ICA). Since its introduction it has become an indispensable tool for statistical data analysis and processing of multi-channel data.

The classical ICA setting

Suppose we have recorded n signals $x_1[t], \ldots, x_n[t]$ for $t = 1, \ldots, T$. In the simplest setting for ICA, these n signals are modeled as linear combinations of n unknown source signals $s_1[t], \ldots, s_n[t]$,

$$x_i[t] = \sum_{j=1}^{n} a_{ij}\, s_j[t] \qquad \text{for } i = 1, \ldots, n. \tag{1.1}$$

The coefficients a_{ij} determine what proportions of the sources $s_j[t]$ appear in which observed signal $x_i[t]$. These coefficients form the so-called mixing matrix A, which is assumed to be invertible and square. Viewing the recorded signals and the source signals as multivariate time-series,

$$x[t] = \begin{bmatrix} x_1[t] \\ \vdots \\ x_n[t] \end{bmatrix} \quad \text{and} \quad s[t] = \begin{bmatrix} s_1[t] \\ \vdots \\ s_n[t] \end{bmatrix}, \tag{1.2}$$

the ICA model can be succinctly written as

$$x[t] = A\, s[t]. \tag{1.3}$$

Because neither the mixing matrix A nor the true sources $s[t]$ are known or observable, ICA is called a blind technique. Prima facie, it seems to be impossible that we can recover the sources $s[t]$ only by analyzing the observed signals $x[t]$. However, the key is to assume that the source signals $s_1[t], \ldots, s_n[t]$ are spatially statistically independent, i.e. knowing the time course of one component, for example of $s_1[t]$, does not provide any information about the time course of the other components $s_2[t], \ldots, s_n[t]$. This is a quite plausible assumption in the initial real-world examples. Note that spatial independence is different from temporal independence which refers to independence in time[1], which would imply that $s[t]$ is independent of $s[t+1]$.

ICA tries to find a separating matrix B such that the resulting signals

$$y[t] = B\, x[t] \tag{1.4}$$

[1] For simplicity, we view t as a time index. However, for unordered data, t enumerates the data in an arbitrary order.

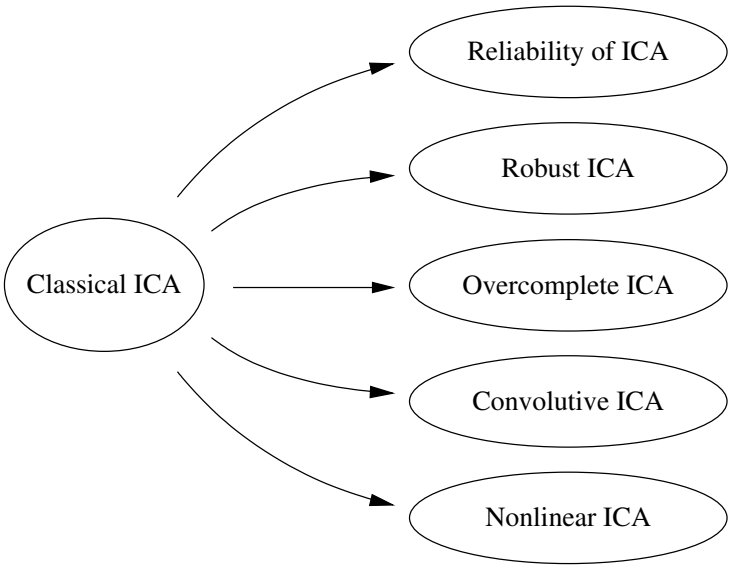


Figure 1.1: Main extensions of classical ICA.

are spatially as independent as possible. The n components of $y[t]$ are called independent components. If either (i) at most one source has a Gaussian distribution, or (ii) the source signals have different spectra, or (iii) they have different variance profiles, a separating matrix can be found such that the true sources are recovered by the demixed signals, $y[t] = Bx[t]$, up to permutation and scaling. These three assumptions are the basis for most ICA algorithms. In Chapter 2 we study these assumptions with tools from information geometry and use them to derive the objective functions of most classical ICA algorithms from the maximum likelihood principle in a uniform way.

Despite its great success in many real-world applications (see [50] and references therein), the classical ICA model has several limitations which has motivated new ICA paradigms that are beyond the classical setting. This thesis proposes new approaches for some of these extensions of classical ICA.

Beyond classical ICA

What is beyond classical ICA? Figure 1.1 shows the five main directions into which we can depart from the classical setting.

Reliability analysis of independent components. In classical ICA the mixing model is assumed to be linear, which is often correct in biomedical data analysis. The examples mentioned above on extracting the ECG of a fetus or on decomposing the EEG of a human brain have approximately a linear and instantaneous mixture because electric fields propagate with the speed of light. Thus

such signals are especially suited for the application of classical ICA methods [100, 99, 101, 75]. However, after having applied an off-the-shelf ICA toolbox, the next step is to let a medical expert interpret the hundreds of alleged sources that were extracted from the multi-channel EEG. This is time-consuming and, more importantly, there is the danger of over-interpretation, because hundreds of sources can show a wide variety of shapes, some of which might match the controlled stimuli by chance. To avoid this problem, we have to estimate the error bars of the ICA directions. This allows us to sort out the directions with large variance, because they were probably chosen by chance, and to keep the direction with small variance, because those directions most likely reflect the inherent statistical structure of the underlying data.

Bootstrap is a tool to assess the variance of arbitrary estimated quantities (invented by Bradley Efron, see [29] for a detailed introduction to bootstrap or [102] for a brief exposition). Bootstrap approximates the variance of possibly complicated estimators by creating several surrogate samples which are resampled from the given data with respect to the empirical distribution. This provides several estimates of the unknown quantity and thus by the law of large numbers an approximation of its variance via the sample variance.

Meinecke et al. [75, 70, 71, 67] apply bootstrap to estimate the error bars of ICA directions. This is not straightforward because each run of ICA can result in different mixing matrices which nonetheless represent the same ICA solution due to the permutation and scaling invariance of ICA. The trick they propose is to unmix the signals initially before doing the bootstrap-resampling. However, the classical bootstrap-resampling is limited to ICA algorithms that ignore the time structure of the signals. Meinecke et al. overcome this difficulty for ICA algorithms that rely on time-shifted covariance matrices (such as TDSEP/SOBI, [112, 12]) by carefully taking into account the structure of these matrices. This approach can also be extended to define resampling schemes for other ICA algorithms. However, different ICA algorithms require different resampling schemes dependent on the statistical structure that guide the chosen ICA methods. This makes the bootstrap-resampling approach difficult to apply for practitioners that often prefer to view their ICA tool as a black box. Thus, we propose a new approach to reliability assessment which we introduce in Chapter 3 that can be naturally applied to any ICA algorithm. Our approach is well motivated by the fundamental assumptions of classical ICA, which we discuss in detail in Chapter 2. Instead of resampling from the empirical distribution we show in Chapter 3 that carefully adding Gaussian noise to the estimated independent components allows us to partially fade out their statistical structure in a controlled manner. Applying the chosen ICA method (as a black box) again to the noise-corrupted data, reveals the reliability of the initially estimated independent components. We expect that reliable independent components reflecting pronounced underlying statistical structure of the sample are less affected by the noise, than unreliable components which were probably chosen by chance. Besides providing estimates of the reliability of the independent components, this analysis shows

also the grouping structure of the unreliable components. Note further that our approach is not limited to deterministic algorithms which transform small perturbations of the data into small perturbations of the mixing matrix. Instead we introduce in Chapter 3 performance measures for assessing the reliability that are by definition invariant to the usual invariances of ICA algorithms.

Himberg et al. [44] distinguish between statistical reliability, which is closely related to our notion of reliability described above, and algorithmic reliability, which reflects the fact that the results of stochastic ICA algorithm can depend very much on the initial starting points. Running a stochastic ICA algorithm several times with slightly different initial conditions on the same data set yields a large number of possible ICA directions, that can be analyzed by clustering methods. ICA directions that are algorithmically reliable form a dense cluster and are easily identifiable. Algorithmic and statistical reliability differ mainly in the way they are estimated, i.e. whether the initial conditions of the algorithms are perturbed or the data itself. However, both algorithmic and statistical reliability quantify how pronounced the statistical structure of the ICA directions is, that is exploited by the chosen ICA algorithm.

The problem of multidimensional ICA (MICA) is closely related to reliability assessment as pointed out by Meinecke et al. [67, 71, 75]. In MICA the assumption of having statistically independent sources (one-dimensional subspaces) is relaxed. MICA allows the sources to be multidimensional, so the assumption is that there are statistically independent subspaces containing signals that could be dependent on each other. In a situation where the estimated independent components are unreliable, there might be still reliable multidimensional independent subspaces. Several methods exist to describe such subspaces, for examples, by means of grouping matrices, that shows which estimated independent components form a common subspace, or dependency graphs (see [9]). Meinecke et al. (for example [71]) are able to estimate such a grouping matrix (they call it separating matrix) as a result of their resampling approach. Similarly, we propose in Chapter 3 a grouping matrix that integrates into our framework of reliability assessment with noise-injection and demonstrate in experiments its validity.

Robust ICA. The key innovation that allowed classical ICA algorithms to solve the BSS problem for the linear model was its use of higher-order statistics. Therewith ICA was able to resolve the signal subspace into independent components which were invisible for PCA. However, higher-order methods are often extremely sensitive to outliers. For example, ICA algorithms based on kurtosis might choose a direction only because in that direction happens to be one single data point with a very large norm (see also [46] for a statistical analysis of the robustness of FastICA). ICA methods based on time structure are usually more robust because they mostly rely on second-order statistics. But also the estimation of covariance matrices can be easily corrupted by outliers because the influence of a single data point still grows quadratically with its norm.

Real-world data is often contaminated by outliers. Thus there is the need to design ICA methods that are particularly robust. In [35] we proposed an outlier detection method based on nearest neighbors that allows to sort a data set from very typical points to very untypical points. Despite the simplicity of these sorting indices we were able to demonstrate empirically in [35] that they are competitive with much more sophisticated outlier detection methods (see for example [10]) and we showed that our outlier indices can robustify methods for clustering and nonlinear dimensionality reduction. Thus, a simple strategy to obtain robust ICA methods would be to robustify existing ICA algorithms by employing these indices as a preprocessing step for outlier rejection.

We will approach the problem of robust ICA along a different path: instead of using outlier indices to eliminate extreme points from the data set, we employ these indices to identify the data points in the most concentrated regions. For super-Gaussian sources, the points of these regions—which we call inliers—are promising candidates for the columns of the mixing matrix. We use this idea in Chapter 4 to construct a new ICA algorithm called IBICA for super-Gaussian sources that is by definition robust against outliers.

Overcomplete ICA. A particularly difficult variant of the classical ICA is obtained by assuming more sources than sensors. In that case the sources can not be uniquely recovered even if the mixing matrix is known. Usually, approaches for overcomplete ICA perform two two steps: (i) identification of the the mixing matrix, and (ii) approximate reconstruction of the sources.

The approaches for the first step often assume that the source signals are super-Gaussian (or even sparse). Intuitively speaking, this assumption reduces the overlap between the data that is scattered along different ICA directions. The overcomplete ICA problem, for example, has been solved by assuming a probabilistic model [61, 81] or by performing k-means clustering of lines that go through the origin. The lines corresponding to the estimated means represent the columns of the mixing matrix [77, 78]. See also [49, 50] for an overview. Note that the assumption of super-Gaussianity can often be enforced by sparsification methods (for example [80, 50, 15]).

In the experiments presented in Chapter 4 we demonstrate that the IBICA algorithm, conceived as an algorithm for robust ICA, also solves the overcomplete ICA for super-Gaussian source signals. By construction it can effortlessly process high-dimensional data sets as we will see later.

The second step, the reconstruction of the sources from the mixed signals given the mixing matrix, is not trivial in the overcomplete setting. All single data points have to be assigned to the different ICA directions, i.e. to the columns of the mixing matrix. However, if the source signals are not extremely sparse this assignment is not unique. There are different ways to approach this problem. For example, the winner-takes-all strategy assigns a data point to the closest ICA direction. [14] present a more sophisticated method that represents each data point in the mixed signal as a linear combination of the n closest source directions

(in the two-dimensional case of the two closest directions). This method can be formulated as an linear optimization problem and it can be solved efficiently (see [14] and also [50] for other approaches to recover the sources).

Nonlinear ICA. In the chemical plant example discussed earlier, the sensors record various physical quantities. As shown in [97, 50], industrial process data can be sometimes represented more compactly by a nonlinear model,

$$x[t] = f(s[t]), \tag{1.5}$$

that generalizes classical ICA by replacing the mixing matrix by an arbitrary invertible function $f : \Re^n \to \Re^n$. Similarly to classical ICA, the goal is to recover the sources $s[t]$ from the observed signals $x[t]$ without knowing the mixing function f. An important property of nonlinear ICA, that emphasizes its difficulty, is that the solution to nonlinear ICA is non-unique and infinitely many solutions exist, as pointed out by [52], if the space of possible mixing functions is unlimited.

Nonlinear ICA is a challenging research task, and several methods have been proposed in the literature. Existing algorithmic approaches of the general nonlinear ICA problem have used, for example, self-organizing maps [82, 62], extensions of generative topographic mapping (GTM [83]), neural networks [16, 65], or Bayesian ensemble learning [97, 54, 58] to unfold the nonlinearity f (and many others, for example [60, 106, 50, 56]). Also a kernel-based method was tried on simplistic toy signals [32]. A neural net approach that can be seen as a nonlinear generalization of the infomax principle [11] is MISEP which is presented in [2].

An important special case of the general nonlinear model is obtained by restricting the mixing function f in the mixing model to operate component-wise on a linear mixture of $s[t]$,

$$x[t] = f(As[t]). \tag{1.6}$$

This is the so-called post-nonlinear model which has nice uniqueness properties as discussed in [95]. This problem can be solved by simultaneously estimating the inverse of the one-dimensional nonlinear functions and the mixing matrix (see [95, 1] and also [50]). In [110] we presented another approach to post-nonlinear mixtures that linearizes the post-nonlinearities via componentwise Gaussianization.

In this thesis we consider the general nonlinear ICA problem. In Chapter 5 we propose an new algorithm for nonlinear ICA that finds solutions under the assumption that the source signals $s[t]$ have some time structure. Assuming this additional structure in the signals allows us to unmix nonlinearly mixed signals. This is related to slow feature analysis (SFA, [105]) which extracts slowly varying signals from a nonlinear mixture. However, SFA uses fixed nonlinearities to transform the data explicitly in some feature space. Our method, which we explain in detail in Chapter 5, uses the kernel trick to employ a high-dimensional (possibly infinite-dimensional) function space without explicitly working in that space.

2 Objective functions of classical ICA

Typically, a method for ICA consists of two parts: an objective function and an optimization algorithm (see [50]). In this chapter, we will focus on the objective functions, because we are interested in the question, what statistical structures in the signals are relevant for ICA. We will derive the objective functions for different ICA algorithms from the maximum likelihood principle, hereby clearly stating the underlying assumptions of the statistical model. We note that this presentation is based on and strongly influenced by Cardoso's three easy routes [21] but also on and by other works [50, 19, 22, 84, 86, 87]. Our exposition is not aimed to be exhaustive. A more detailed and more complete overview can be found in the book of Hyvärinen, Karhunen and Oja [50].

While trying to avoid letting the formulas fall out of the blue, we decided to transfer most of the proofs to the appendix in order to keep a steady flow of ideas.

Chapter outline

After fixing some notations in Section 2.1, we introduce in Section 2.2 some basic concepts from information geometry [4] which will help to keep the presentation brief. We carefully define some manifolds in distribution space that characterize special sets of distributions relevant for ICA. For clarity, we explicitly work out the projections onto these manifolds. In Section 2.3 we formulate the non-property assumptions which will be the key for the discussion of the maximum likelihood approach that follows in the remaining sections: we will demonstrate that the objective functions of most ICA algorithms are based on non-Gaussianity, non-stationarity or non-flatness. These findings will motivate in Chapter 3 the noise-injection approach for estimating the reliability of ICA components.

2.1 Basic notations

The methods considered in this thesis process data which is given by a real-valued $n \times T$-dimensional matrix,

$$X = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1T} \\ X_{21} & X_{22} & & X_{2T} \\ \vdots & & \ddots & \vdots \\ X_{n1} & X_{n2} & \cdots & X_{nT} \end{bmatrix} \in \Re^{n \times T}. \tag{2.1}$$

with $\Re$ being the set of real numbers. We denote single rows or columns using the colon notation (see [34]), i.e. we refer to the j-th row of X by

$$X_{j:} = [X_{j1}\ X_{j2}\ \cdots\ X_{jT}] \tag{2.2}$$

and to the t-th column of X by

$$X_{:t} = \begin{bmatrix} X_{1t} \\ X_{2t} \\ \vdots \\ X_{nt} \end{bmatrix}. \tag{2.3}$$

Alternatively, we use for columns $x[t] := X_{:t}$ and for column entries $x_j[t] := X_{jt}$. The rows of X are usually interpreted as instances of one-dimensional signals and thus X itself as an instance of a multivariate signal.

The notations for the corresponding random variables match the notations of the data instances as summarized in the following table:

	matrix	rows	columns	entries
data	X	$X_{j:}$	$X_{:t}$	X_{jt}
(alternative)			$x[t]$	$x_j[t]$
random variable	X	$X_{j:}$	$X_{:t}$	X_{jt}

The probability density function (PDF) of the random variable X is denoted as p_X. The marginal distribution $p_{X_{j:}}$, $p_{X_{:t}}$ and $p_{X_{jt}}$ corresponding to a row, a column and an entry of X are obtained from p_X by integrating out all but the indicated variables. A column vector related to X is written as x (with PDF p_x), its entries as x_j (with PDF p_{x_j}). p_X refers to the true PDF of X and q_X to a different PDF, usually the PDF that is implied by the assumed model. We call q_X the modeled PDF.

2.2 Tools from information geometry

Information geometry employs the sophisticated methods of differential geometry to analyze spaces of probability distributions (see [4]). In the following discussion, some basic notions of information geometry help us to be succinct and clear. Our explanations are similar to (and motivated by) [21, 22]. We consider the space of all distributions (represented by their PDFs) of $n \times T$-matrices. Distances in that space are measured by the Kullback-Leibler divergence (KLD), which is defined as

$$D(p\|q) := \int p(X)\ \log \frac{p(X)}{q(X)}\, dX. \tag{2.4}$$

Note that the KLD is non-negative and equal to zero only if $p = q$, but not symmetric.

2.2.1 Manifolds in distribution space

We define some submanifolds in distribution space that have special characteristics which are important for ICA. We will call submanifolds for brevity simply manifold, since the distinction is not important for our discussion. Let Z be an $n \times T$-matrix with PDF p_Z:

- The manifold $\mathcal{I}$ contains all distributions that are statistically independent along the columns, i.e. $p_Z \in \mathcal{I}$ if and only if p_Z can be factorized in the following way:

$$p_Z(Z) = \prod_{j=1}^{n} p_{Z_{j:}}(Z_{j:}). \tag{2.5}$$

  The distributions in $\mathcal{I}$ are also called spatially independent.

- The manifold $\mathcal{G}$ contains all distributions that are independent identically distributed (IID) in time, i.e. $p_Z \in \mathcal{G}$ if and only if there exists a density p_z such that

$$p_Z(Z) = \prod_{t=1}^{T} p_z(Z_{:t}) = \prod_{t=1}^{T} p_z(z[t]). \tag{2.6}$$

  The distributions in $\mathcal{G}$ are also called temporally independent.

- The manifold $\mathcal{S}$ contains all distributions that are at each point in time multivariate Gaussian distributed with zero mean, i.e. $p_Z \in \mathcal{S}$ if and only if there exist covariance matrices $\Sigma_1^Z, \ldots, \Sigma_T^Z$ such that

$$p_Z(Z) = \prod_{t=1}^{T} \phi(Z_{:t}; \Sigma_t^Z) = \prod_{t=1}^{T} \phi(z[t]; \Sigma_t^Z), \tag{2.7}$$

  with $\phi(v; \Sigma) := \exp(-v^\top \Sigma^{-1} v/2)/\sqrt{(2\pi)^n \det \Sigma}$ being the multivariate Gaussian density with zero mean and covariance matrix Σ (note that for complex variables v the transpose $\top$ does a conjugate-complex transpose). In the intersection of $\mathcal{S}$ and $\mathcal{I}$ the covariance matrices $\Sigma_1^Z, \ldots, \Sigma_T^Z$ are diagonal.

- The coefficients of the discrete Fourier transform (DFT) of the rows of Z are denoted by

$$\tilde{z}[l] = \frac{1}{\sqrt{T}} \sum_{t=1}^{T} z[t] \exp(-2\sqrt{-1}\pi l t/T) \quad \text{for } l \in \{1, \ldots, T\}. \tag{2.8}$$

  The manifold $\mathcal{F}$ contains all distributions such that the DFT coefficients of its rows are independent and have a complex Gaussian distribution, i.e. $p_Z \in \mathcal{F}$ if and only if there exist covariance matrices $\Gamma_1^Z, \ldots, \Gamma_T^Z$ (which could be complex-valued, but must be hermitian) such that

$$p_Z(Z) = \prod_{l=1}^{T} \phi(\tilde{z}[l]; \Gamma_l^Z). \tag{2.9}$$

  In the intersection of $\mathcal{F}$ and $\mathcal{I}$ the covariance matrices $\Gamma_1^Z, \ldots, \Gamma_T^Z$ are diagonal.

2.2.2 Pythagoras in distribution space

Let $\mathcal{M}$ be a manifold in distribution space. The projection of p onto $\mathcal{M}$, denoted by $p^{\mathcal{M}}$, is the distribution in $\mathcal{M}$ that is closest to p (in the KLD sense), i.e.

$$p^{\mathcal{M}} := \arg\min_{q \in \mathcal{M}} D(p\|q). \tag{2.10}$$

These projections are especially useful for exponential (also called e-flat) manifolds that we define next.

A manifold $\mathcal{M}$ in distribution space is called exponential (or e-flat) if and only if it contains the exponential segment between any two of its members, i.e. the implication

$$p(X), q(X) \in \mathcal{M} \implies p(X)^{(1-\alpha)} q(X)^{\alpha} \exp(-\psi(\alpha)) \in \mathcal{M} \tag{2.11}$$

must hold for all $0 \leq \alpha \leq 1$ with $\psi(\alpha)$ being an appropriate normalizing factor. It can be easily seen that $\mathcal{I}, \mathcal{G}, \mathcal{S}, \mathcal{F}$ and their intersections (such as $\mathcal{G} \cap \mathcal{I}$) are exponential manifolds (for example, for $p, q \in \mathcal{I}$ write out the expression $p^{(1-\alpha)} q^{\alpha} \exp(-\psi(\alpha))$ to see that it is actually in $\mathcal{I}$ itself, similar for the other manifolds). This is quite convenient because for an exponential manifold $\mathcal{M}$ we have Pythagoras' theorem:

$$D(p\|q) = D(p\|p^{\mathcal{M}}) + D(p^{\mathcal{M}}\|q) \quad \text{for all } q \in \mathcal{M} \tag{2.12}$$

(for a proof see [4]). Informally speaking, the KLD behaves (somewhat) like the squared Euclidean distance. Suppose we have a hyperplane $\mathcal{M}$ in $\Re^n$. Let q be a point on that plane and p another point. Denote by $p^{\mathcal{M}}$ the orthogonal projection of p onto $\mathcal{M}$, then p, $p^{\mathcal{M}}$ and q form a right-angled triangle and the squared lengths of its sides respect Pythagoras' theorem,

$$\|p - q\|^2 = \|p - p^{\mathcal{M}}\|^2 + \|p^{\mathcal{M}} - q\|^2 \tag{2.13}$$

(see Figure 2.1).

2.2.3 Projections in distribution space

For the previously defined manifolds $\mathcal{I}, \mathcal{G}, \mathcal{F}, \mathcal{S}$ it is instructive to write out the projections explicitly:

- Projection on $\mathcal{I}$: the projected PDF $p_Z^{\mathcal{G}}$ is the product of the marginal PDFs $p_{Z_{j:}}$ of p_Z,

$$p_Z^{\mathcal{I}}(Z) = \prod_{j=1}^{n} p_{Z_{j:}}(Z_{j:}). \tag{2.14}$$

- Projection on $\mathcal{G}$: let p_z be the n-dimensional PDF obtained by averaging the marginal PDFs $p_{Z_{:t}}$, i.e.

$$p_z(z) := \frac{1}{T} \sum_{t=1}^{T} p_{Z_{:t}}(z). \tag{2.15}$$

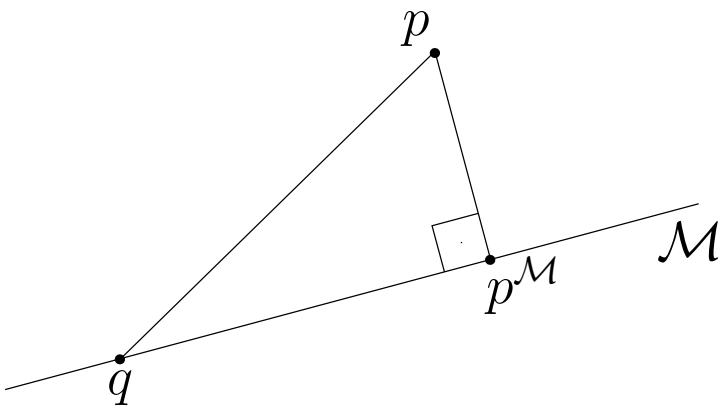


Figure 2.1: Pythagoras' theorem in distribution space: the squared distance of p to $q \in \mathcal{M}$ is equal to the sum of the squared distances between p and $p^{\mathcal{M}}$ and between $p^{\mathcal{M}}$ and q.

We call p_z the average marginal PDF of Z. Then the projection of p_Z onto $\mathcal{G}$ is

$$p_Z^{\mathcal{G}}(Z) = \prod_{t=1}^{T} p_z(Z_{:t}) = \prod_{t=1}^{T} p_z(z[t]). \tag{2.16}$$

- Projection on $\mathcal{S}$: let Σ_t^Z be the covariance matrix of $Z_{:t}$ (or $z[t]$) according to its marginal density $p_{Z_{:t}}$. The projection of p_Z onto $\mathcal{S}$ is the product of Gaussian distributions with those covariance matrices,

$$p_Z^{\mathcal{S}}(Z) = \prod_{t=1}^{T} \phi(Z_{:t}; \Sigma_t^Z) = \prod_{t=1}^{T} \phi(z[t]; \Sigma_t^Z). \tag{2.17}$$

- Projection on $\mathcal{F}$: let Γ_l^Z be the covariance matrix of the l-th coefficient vector $\tilde{z}[l]$ of the DFT of the rows of Z (note that Γ_l^Z is a hermitian matrix, see also the definition of $\mathcal{F}$ above). The projection of p_Z onto $\mathcal{F}$ is the product of Gaussian distributions of $\tilde{z}[l]$ with those covariance matrices,

$$p_Z^{\mathcal{F}}(Z) = \prod_{l=1}^{T} \phi(\tilde{z}[l]; \Gamma_l^Z). \tag{2.18}$$

  Note that for example the sequence of the upper-left entries of the matrices $\Gamma_1^Z, \ldots, \Gamma_T^Z$ represent the power spectrum of the first signal. Similarly, other entries of those matrices are power spectra of the other signals or cross power spectra respectively.

Note that the projection $p^{\mathcal{M}}$ is the only distribution in $\mathcal{M}$ that allows the decomposition by Pythagoras' theorem. Thus to prove that the given expressions are the correct

projections for $\mathcal{I}, \mathcal{G}, \mathcal{F}, \mathcal{S}$, it suffices to show that they fulfill Pythagoras' theorem, which can be easily seen by writing out the involved terms and remembering Fubini's theorem.

2.3 Non-property assumptions

Most ICA algorithms are based on certain assumptions. For example algorithms that search for non-Gaussian directions require that there is at most one Gaussian direction, otherwise the source signals can not be recovered. Other ICA algorithms exploit the time structure of the sources. They need some diversity in the spectra to separate the signals, so their spectra should not be flat and different to each other. Alternatively, the sources could be non-stationary with different variance profiles. Let us look at the various assumptions in more detail.

In practical situations the observed signals are usually given as an $n \times T$ matrix,

$$X = [x[1] \cdots x[T]] \tag{2.19}$$

In matrix form the ICA model (introduced in Equation (1.3) can be written as

$$X = AS \tag{2.20}$$

with S being an $n \times T$ matrix that contains the source signals along its rows. We assume that all signals have zero mean. The assumption that the sources are statistically independent means that the modeled PDF q_S of S factorizes along the columns,

$$q_S(S) = \prod_{j=1}^{n} q_{S_{j:}}(S_{j:}) = \prod_{j=1}^{n} q_{S_{j:}}(s_j), \tag{2.21}$$

which is equivalent to assuming $q_S \in \mathcal{I}$ with $\mathcal{I}$ being one of the manifolds defined above. This basic premise is included in the following three assumptions about the non-properties of the source signals: absence of Gaussianity, absence of flatness and absence of stationarity:

Non-Gaussianity. Assume that at most one of the sources is Gaussian distributed and that the sources are independent identically distributed (IID) (independent in time). Formally this is noted by $q_S \in \mathcal{G} \cap \mathcal{I}$ or $q_S = q_S^{\mathcal{G} \cap \mathcal{I}}$, i.e. the (modeled) PDF of S can be written as a double product of some one-dimensional PDFs $q_{s_1}, \ldots, q_{s_n}$,

$$q_S(S) = \prod_{t=1}^{T} \prod_{j=1}^{n} q_{s_j}(S_{jt}) = \prod_{t=1}^{T} \prod_{j=1}^{n} q_{s_j}(s_j[t]) \tag{2.22}$$

and that at most one of the involved one-dimensional distributions $q_{s_1} \ldots, q_{s_n}$ is Gaussian.

Non-stationarity. The PDF of the sources S is assumed to be the product in time of Gaussian distributions with varying diagonal covariance matrices. Formally this

is noted by $q_S \in \mathcal{S} \cap \mathcal{I}$ or $q_S = q_S^{\mathcal{S}\cap\mathcal{I}}$, i.e. the (modeled) PDF of S can be written as

$$q_S(S) = \prod_{t=1}^{T} \phi(S_{:t}; \Sigma_t^S) = \prod_{t=1}^{T} \phi(s[t]; \Sigma_t^S) \tag{2.23}$$

with $\Sigma_t^S := \mathrm{diag}(\sigma_{1t}^2, \ldots, \sigma_{nt}^2)$ being a diagonal $n \times n$ matrix with local variances $\sigma_{1t}^2, \ldots, \sigma_{nt}^2$ along the diagonal. We call $\sigma_{j1}^2, \ldots, \sigma_{jT}^2$ the variance profile of the j-th source signal. We assume furthermore that the variance profiles of the sources are pairwise not proportional to each other.

Non-flatness. The PDF of the sources S is assumed to be the product of the Gaussian distributions of the DFT coefficient vectors of the rows of S. Formally this is noted by $q_S \in \mathcal{F} \cap \mathcal{I}$ or $q_S = q_S^{\mathcal{F}\cap\mathcal{I}}$, i.e. the (modeled) PDF of S can be written as

$$p_S(S) = \prod_{l=1}^{T} \phi(\tilde{s}[l]; \Gamma_l^S). \tag{2.24}$$

with $\tilde{s}[l]$ being the aforementioned DFT coefficient vectors and $\Gamma_l^S := \mathrm{diag}(g_{1l}, \ldots, g_{nl})$ being the diagonal $n \times n$ matrix with the entries $g_{1l}, \ldots, g_{nl}$ along the diagonal. Note that $g_{j1}, \ldots, g_{jT}$ is the power spectrum of the j-th signal. Because of the independence of the sources the cross power spectra vanish, so $\Gamma_1^S, \ldots, \Gamma_T^S$ are diagonal. We assume furthermore that the power spectra of the sources are pairwise not proportional to each other.

These assumption might seem to be very restrictive. Obviously, real-world sources seldom fulfill one of these assumptions completely. However, they are merely a technical vehicle. For example, assuming IID signals implies that only properties of the marginal distribution are used and that time structure (non-flatness) and non-stationarity is ignored. Or, assuming Gaussian distributions implies that only second-order information is used.

The modeled PDF q_S is characterized by certain parameters θ, which differ dependent on the chosen assumption:

- Under the non-Gaussianity assumption, θ consists of the n one-dimensional PDFs $q_{s_1}, \ldots, q_{s_n}$.
- Under the non-stationarity assumption, θ consists of the T diagonal covariance matrices $\Sigma_1^S, \ldots, \Sigma_T^S$.
- Under the non-flatness assumption, θ consists of the T diagonal covariance matrices $\Gamma_1^S, \ldots, \Gamma_T^S$.

Modeling the source signals S induces also a model for the observed signals $X = AS$. In particular, the modeled PDF for X is:

$$q_X(X; A, \theta) = \frac{q_S(A^{-1}X)}{|\det A|^T} \tag{2.25}$$

which depends on the unknown mixing matrix A and θ. This equation is proven in Appendix A.2. Note that T is the number of columns of X and not the transpose operator $\top$.

2.4 Maximizing the likelihood

In order to solve the ICA problem, we have to find the mixing matrix A. We will use the following strategy which is common in statistics: find a matrix A and parameters θ such that the modeled PDF of X is maximized. Hereby we hope to get the solution that best fits the observed data. This approach is called Maximum Likelihood (ML) estimation.

What is the likelihood of the data? Suppose X is fixed (which is the case in practice, because we observe X only once), then $q_X(X; A, \theta)$ can be viewed as a function of A and θ. This function is called the likelihood of X for the parameters A and θ, denoted by

$$\mathcal{L}(A, \theta) := q_X(X; A, \theta) = \frac{q_S(A^{-1}X)}{|\det A|^T}. \tag{2.26}$$

Note that the likelihood is not a function of X but of the unknown mixing matrix A, the parameter of interest, and θ, the nuisance parameter.

The expectation with respect to the true PDF of X of the negative logarithm of the likelihood can be decomposed into two terms:

$$-E_X \log \mathcal{L}(A, \theta) = D(p_X \| q_X) + h(p_X) \tag{2.27}$$

(see Appendix A.3 for the proof). The first term is the KLD between the true PDF of X and its modeled PDF. The second term is the entropy $h(p_X) := -E_X \log p_X(X) = -\int p_X(X) \log p_X(X)\, dX$ of X (expectation with respect to its true density). Because the entropy of X does not depend on the parameters A and θ, we see that maximizing the likelihood (for an empirical distribution determined by X) leads to a minimization of the mismatch between p_X and q_X, i.e. between the true and the modeled PDF. With other word, maximizing the likelihood looks for a parameter setting such that the model best fits the observed data. This is a general property of ML estimation.

Denote by $Y = A^{-1}X$ the demixed signals which should match the unknown source S as close as possible (up to permutation and scaling). The KLD between p_X and q_X can be written in terms of the demixed signals Y and the sources S,

$$D(p_X \| q_X) = D(p_Y \| q_S) \tag{2.28}$$

(see Appendix A.3 for the proof). Therefore, the mismatch between p_X and q_X can be measured as the KLD between the true PDF of Y and the modeled PDF of S. Concluding, we have shown that maximizing the likelihood $\mathcal{L}(A, \theta)$ corresponds to minimizing the KLD between p_Y and q_S which will be analyzed further under the three non-property assumptions.

2.5 Objective functions assuming non-Gaussianity

In this section we assume non-Gaussianity, i.e. the PDF of the sources is modeled as $q_S \in \mathcal{G} \cap \mathcal{I}$. This implies for the likelihood (Equation (2.26)) that it factorizes (by Equation (2.22),

$$\mathcal{L}(A, \theta) = \frac{\prod_{t=1}^{T} \prod_{j=1}^{n} q_{s_j}((A^{-1})_{:j}^{\top} x[t])}{|\det A|^T} \tag{2.29}$$

with $(A^{-1})_{:j}$ being the j-th column of the inverse of A, or equivalently for the logarithm of the likelihood (log-likelihood)

$$\log \mathcal{L}(A, \theta) = \sum_{t=1}^{T} \sum_{j=1}^{n} \log q_{s_j}((A^{-1})_{:j}^{\top} x[t]) - T|\det A|. \tag{2.30}$$

Many ICA algorithms use this expression as their objective function and maximize it by gradient ascent (see [33, 88, 88]), natural gradient (see [6, 5]) or fixed-point algorithms (FastICA, see [47]). The nuisance parameters, $q_{s_1}, \ldots, q_{s_n}$, are either previously fixed (by choosing nonlinearities in form of score functions of the assumed densities) or optimized simultaneously with the mixing matrix A (for example in FastICA). ICA based on the infomax principle (by Bell and Sejnowski, see [11]) corresponds to maximizing the likelihood with a gradient ascent approach (as noted by several authors, for example [18, 76]). The book of Hyvärinen, Karhunen and Oja [50] contains a more detailed summary of the ramifications between these different approaches.

More objective functions for ICA can be obtained by decomposing the KLD between p_Y and q_S (the right-hand side expression of Equation (2.28)) using the information geometric machinery defined above,

$$\begin{aligned} D(p_Y \| q_S) &= D(p_Y \| p_Y^{\mathcal{G}}) + D(p_Y^{\mathcal{G}} \| q_S) \\ &= D(p_Y \| p_Y^{\mathcal{G}}) + D(p_Y^{\mathcal{G}} \| p_Y^{\mathcal{G} \cap \mathcal{I}}) + D(p_Y^{\mathcal{G} \cap \mathcal{I}} \| q_S), \end{aligned} \tag{2.31}$$

(see Figure 2.2). The first equality is based on Pythagoras' theorem for manifold $\mathcal{G}$, the second on Pythagoras' theorem for manifold $\mathcal{I}$. Let us examine the resulting terms. The first term, $D(p_Y \| p_Y^{\mathcal{G}})$, is equal to $D(p_X \| p_X^{\mathcal{G}})$, because the KLD is invariant under transformations (prove see Appendix A.4). Because $\mathcal{G}$ is closed under linear transformations, $D(p_X \| p_X^{\mathcal{G}})$ does not depend on A (and also not on θ). So we can ignore the first term for the minimization of the initial expression $D(p_Y \| q_S)$. Note that the nuisance parameter θ has been isolated to the third term, $D(p_Y^{\mathcal{G} \cap \mathcal{I}} \| q_S)$. By choosing $q_S = p_Y^{\mathcal{G} \cap \mathcal{I}}$ the non-Gaussianity assumption is fulfilled (because $q_S \in \mathcal{G} \cap \mathcal{I}$) and the third term vanishes. Hereby we minimized $D(p_Y \| q_S)$ (because the KLD is non-negative) and we eliminated the nuisance parameter θ. So, we are left with the middle term $D(p_Y^{\mathcal{G}} \| p_Y^{\mathcal{G} \cap \mathcal{I}})$ which we inspect further. Writing out the involved PDFs,

$$\frac{1}{T} D(p_Y^{\mathcal{G}} \| p_Y^{\mathcal{G} \cap \mathcal{I}}) = \frac{1}{T} D(\prod_{t=1}^{T} p_y \| \prod_{t=1}^{T} \prod_{j=1}^{n} p_{y_j}) = D(p_y \| \prod_{j=1}^{n} p_{y_j}) = I(y_1, \ldots, y_n), \tag{2.32}$$

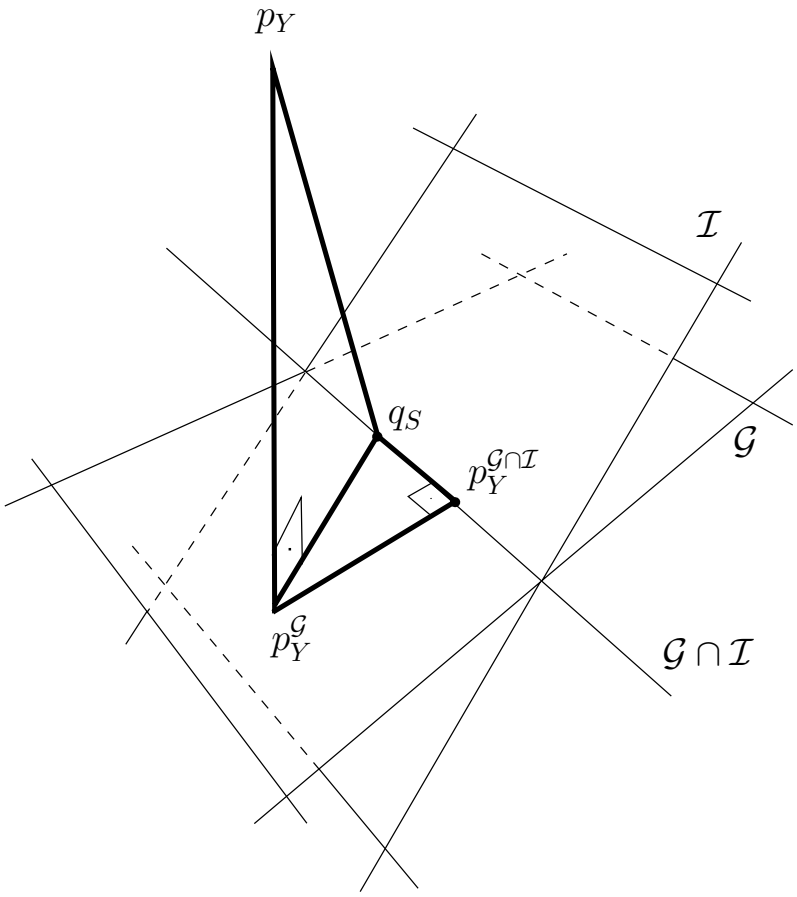


Figure 2.2: Applying twice Pythagoras' theorem in distribution space for the manifolds $\mathcal{G}$ and $\mathcal{G}\cap\mathcal{I}$: the divergence between p_Y and $q_S \in \mathcal{G}\cap\mathcal{I}$ is decomposed. First p_Y is projected onto $\mathcal{G}$ and from there onto $\mathcal{G}\cap\mathcal{I}$. For the objective function under the non-Gaussianity assumption only the divergence between $p_Y^{\mathcal{G}}$ and $p_Y^{\mathcal{G}\cap\mathcal{I}}$ is relevant. Replacing $\mathcal{G}$ with $\mathcal{F}$ or $\mathcal{S}$ we get the corresponding picture for the non-flatness and non-stationarity assumptions.

reveals that the middle term is the mutual information. So, we have seen that maximizing the likelihood corresponds to minimizing the mutual information,

$$\min_{A,\theta} -E_X \log \mathcal{L}(A,\theta) = \min_A T\, I(y_1,\ldots,y_n). \tag{2.33}$$

The mutual information can be minimized directly using a kernel approach (kernel ICA, see [8]). In order to derive more objective functions for ICA we decompose it further,

$$I(y_1,\ldots,y_n) = \left(\sum_{j=1}^{n} h(p_{y_j})\right) - h(p_x) + \log|\det A| \tag{2.34}$$

(for the proof see Appendix A.5) with $h(p_x)$ being the entropy of the average marginal density of X (see Equation (2.15)). The entropy of p_x does not depend on A so we can omit it. $\log|\det A|$ can be eliminated under an additional assumption: suppose that the observed data X has been whitened (see Appendix A.1), so the demixing matrix A^{-1} (and also A itself) has to be a rotation matrix (see Appendix A.1 for a proof). The logarithm of the determinant of a rotation matrix is zero. That means that for whitened data the mutual information is equal to the sum of the entropies of the single demixed signals,

$$I(y_1,\ldots,y_n) = \left(\sum_{j=1}^{n} h(p_{y_j})\right). \tag{2.35}$$

So instead of minimizing the mutual information, we can equivalently minimize the sum of single entropies of the demixed signals.

For fixed variance, the maximum entropy distribution is the Gaussian distribution. Therefore minimizing entropy can be interpreted as non-Gaussianization. This idea is implemented in several ICA algorithms, for example by maximizing the so-called neg-entropy, which is the difference between the entropy of a Gaussian distribution and the entropy of a demixed signal (for example [47]). Maximizing non-Gaussianity can also be motivated by the central limit theorem: sums of random variables tend to be Gaussian distributed, i.e. the mixing matrix makes the distributions more Gaussian. There are other ways to exploit non-Gaussianity based on kurtosis or other higher-order moments (for example approximating negentropy with cumulants [26]; see [50] for a detailed summary). One of these algorithms, called JADE (joint-approximate diagonalization of eigenmatrices, see [24]), will be used in subsequent chapters: it whitens the data as a preprocessing step and then simultaneously diagonalizes several matrices that contains some of the fourth order cross-cumulants.

2.6 Objective functions assuming non-stationarity

In this section we assume non-stationarity, i.e. the PDF of the sources is modeled as $q_S \in \mathcal{S} \cap \mathcal{I}$. Analogously, to the previous section, we decompose the KLD between p_Y

and q_S (the right-hand side expression of Equation (2.28)),

$$\begin{aligned} D(p_Y \| q_S) &= D(p_Y \| p_Y^{\mathcal{S}}) + D(p_Y^{\mathcal{S}} \| q_S) \\ &= D(p_Y \| p_Y^{\mathcal{S}}) + D(p_Y^{\mathcal{S}} \| p_Y^{\mathcal{S}\cap\mathcal{I}}) + D(p_Y^{\mathcal{S}\cap\mathcal{I}} \| q_S). \end{aligned} \tag{2.36}$$

Again, both equalities are based on Pythagoras' theorem for exponential manifolds (for $\mathcal{S}$ and $\mathcal{I}$). Since the first term $D(p_Y \| p_Y^{\mathcal{S}})$ equals $D(p_X \| p_X^{\mathcal{S}})$ and $\mathcal{S}$ is closed under linear transformations, it can be ignored. Again, the nuisance parameter θ has been isolated to the third term, $D(p_Y^{\mathcal{S}\cap\mathcal{I}} \| q_S)$. By choosing $q_S = p_Y^{\mathcal{S}\cap\mathcal{I}}$ the non-stationarity assumption is fulfilled and the third term vanishes, hereby eliminating the nuisance parameter θ. We are left with the middle term $D(p_Y^{\mathcal{S}} \| p_Y^{\mathcal{S}\cap\mathcal{I}})$ which we inspect further.

First note that the KLD between two zero mean Gaussian can be expressed in terms of their covariance matrices,

$$D(\Sigma \| \Gamma) := D(\phi(x, \Sigma) \| \phi(x, \Gamma)) = \frac{1}{2}(\mathrm{tr}(\Gamma^{-1}\Sigma) - \log\det(\Gamma^{-1}\Sigma) - n). \tag{2.37}$$

This suggests a natural measure for diagonality,

$$\mathrm{off}(\Sigma) := D(\Sigma \| \operatorname{diag}(\Sigma)). \tag{2.38}$$

Let $\Sigma_1^Y, \ldots, \Sigma_T^Y$ be the covariance matrices for the different points in time for Y and similarly $\Sigma_1^X, \ldots, \Sigma_T^X$ for X (see the paragraph before Equation (2.17)). Then we have

$$\Sigma_t^Y = A^{-1}\Sigma_t^X A^{-\top}. \tag{2.39}$$

Using these expressions we write out the KLD between $p_Y^{\mathcal{S}}$ and $p_Y^{\mathcal{S}\cap\mathcal{I}}$,

$$D(p_Y^{\mathcal{S}} \| p_Y^{\mathcal{S}\cap\mathcal{I}}) = \sum_{t=1}^{T} D(\Sigma_t^Y \| \operatorname{diag}(\Sigma_t^Y)) = \sum_{t=1}^{T} \mathrm{off}(\Sigma_t^Y) = \sum_{t=1}^{T} \mathrm{off}(A^{-1}\Sigma_t^X A^{-\top}). \tag{2.40}$$

So, we have seen that minimizing the objective function based on ML estimation under the non-stationarity assumption corresponds to a simultaneous diagonalization problem of time-dependent covariance matrices of X. In practice these matrices can be estimated by assuming block-wise stationarity and calculating their sample versions (see [86]; also [66, 23, 48] for other methods exploiting non-stationarity). Besides using the criterion in Equation (2.38) for simultaneous diagonalization (as done in [85]), other diagonalization methods can be used as well ([57] or FFDIAG [111] and references therein). In subsequent chapters we will use Pham's implementation SEPAGAUS as an example of an ICA algorithm working under the non-stationarity assumption (because it can be easily restricted to take into account only non-stationarity).

2.7 Objective functions assuming non-flatness

In this section we assume non-flatness, i.e. the PDF of the sources is modeled as $q_S \in \mathcal{F} \cap \mathcal{I}$. Analogously to the previous sections we decompose the KLD between p_Y

and q_S (the right-hand side expression of Equation (2.28)) using Pythagoras' theorem for $\mathcal{F}$ and $\mathcal{I}$,

$$\begin{aligned} D(p_Y \| q_S) &= D(p_Y \| p_Y^{\mathcal{F}}) + D(p_Y^{\mathcal{F}} \| q_S) \\ &= D(p_Y \| p_Y^{\mathcal{F}}) + D(p_Y^{\mathcal{F}} \| p_Y^{\mathcal{F}\cap\mathcal{I}}) + D(p_Y^{\mathcal{F}\cap\mathcal{I}} \| q_S). \end{aligned} \tag{2.41}$$

Again the first term $D(p_Y \| p_Y^{\mathcal{F}})$ (being equal $D(p_X \| p_X^{\mathcal{F}})$ and noting that $\mathcal{F}$ is closed under linear transformations) can be ignored, and the third term $D(p_Y^{\mathcal{F}\cap\mathcal{I}} \| q_S)$ vanishes by setting $q_S = p_Y^{\mathcal{F}\cap\mathcal{I}}$, hereby eliminating the nuisance parameter θ. Again, the middle term $D(p_Y^{\mathcal{F}} \| p_Y^{\mathcal{F}\cap\mathcal{I}})$ remains and, similarly to the previous section, we rewrite it as a diagonalization problem.

Let $\Gamma_1^Y, \ldots, \Gamma_T^Y$ be the covariance matrices of the coefficient vectors $\tilde{z}[1], \ldots, \tilde{z}[T]$ of the DFT of the rows of Z and similarly $\Gamma_1^X, \ldots, \Gamma_T^X$ for X. Then we have

$$\Gamma_t^Y = A^{-1} \Gamma_t^X A^{-\top}. \tag{2.42}$$

Using these expressions we write out the KLD between $p_Y^{\mathcal{F}}$ and $p_Y^{\mathcal{F}\cap\mathcal{I}}$,

$$D(p_Y^{\mathcal{F}} \| p_Y^{\mathcal{F}\cap\mathcal{I}}) = \sum_{t=1}^{T} D(\Gamma_t^Y \| \operatorname{diag}(\Gamma_t^Y)) = \sum_{t=1}^{T} \operatorname{off}(\Gamma_t^Y) = \sum_{t=1}^{T} \operatorname{off}(A^{-1} \Gamma_t^X A^{-\top}). \tag{2.43}$$

Analogously to the previous section, we see that minimizing the objective function based on ML estimation under the non-flatness assumption corresponds to a simultaneous diagonalization problem of (possibly complex) covariance matrices (having followed [84, 87]). There are different ways to solve this problem in practice. For example, these matrices can be estimated as the Fourier transforms of time-shifted covariance matrices, which are defined as

$$C_\tau^X := \sum_{t=1}^{T-\tau} x[t] x[t+\tau]^\top \quad \text{for } \tau \in \{0, \ldots, T-1\}. \tag{2.44}$$

However, noting that the matrices $\Gamma_1^X, \ldots, \Gamma_T^X$ are diagonal if and only if the time-shifted covariance matrices are diagonal, we see that it is sufficient to simultaneously diagonalize the latter. In practice, it is enough that only some of these matrices are jointly diagonalized (often their symmetrized versions). This strategy is pursued by several algorithms (for example by TDSEP [112] which is equivalent to SOBI [12]; see also [57]). Early work along these lines was done in [96, 73]. Regarding the choice of the joint diagonalization method care has to be taken, because in general the time-shifted covariance matrices are not positive-definite. Therefore, algorithms like FFDIAG [111] or ACDC [107] should be used. Alternatively, the observed data X can be preprocessed by whitening. This restricts the diagonalizing matrix to the set of rotation matrices. In that case, a fast method based on Jacobi angles is the method of choice (see [25]). In subsequent chapters we will use TDSEP (with whitening as preprocessing) as a representative for methods based on non-flatness.

2.8 Summary

Starting from the maximum likelihood principle we derived most of the objective functions for classical ICA employing basic tools from information geometry. In particular, we showed under the non-Gaussianity assumption that maximizing the likelihood corresponds to minimizing mutual information. For whitened data, we further proved that minimizing the mutual information is equivalent to maximizing the sum of the marginal negentropies. Furthermore, we have seen that under the non-flatness and non-stationarity assumptions, maximizing the likelihood can be phrased as a simultaneous diagonalization problem of certain covariance matrices that express the time structure of the signals.

Our discussion covered the objective functions of most classical ICA algorithms, so we conclude that the statistical structure relevant for most ICA algorithms is captured by the non-property assumptions, which are non-Gaussianity, non-stationarity and non-flatness. These findings will motivate in the next chapter a general scheme for reliability estimation of independent components that can be applied to any ICA algorithm.

3 Reliability and grouping of independent components with noise-injection

In order to apply successfully unsupervised learning algorithms such as ICA to real-world problems, it is of fundamental importance to determine how trustworthy their results are. For example, Figure 3.1 shows the results of applying ICA in two different situations: the left panel shows a scatterplot of the mixture of sound signals. The directions obtained by the ICA algorithm accurately express the structure of the underlying density, so the resulting decomposition accurately separates the sound signals. This is in contrast to the right panel, which contains the scatterplot of two-dimensional random noise. The chosen directions do not reflect any property of the data, because the distribution of the noise is rotation-invariant. Thus, the ICA decomposition obtained from the sample of the data is meaningless.

Meinecke et al. [71, 70, 67] proposed a bootstrap resampling method that can distinguish these two situations by estimating the reliability and grouping of independent components found by ICA algorithms. It has been successfully applied to various real-world problems, for example in biomedical data analysis to magnetoencephalograms (MEG; see [67, 71, 75]). Their method profits from the well-developed theory of boot-

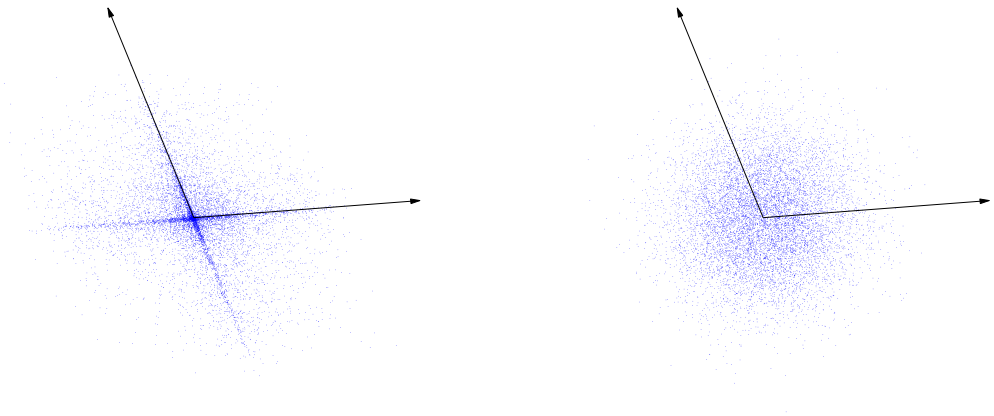


Figure 3.1: Can you trust your independent components? Left panel: the chosen directions reflect the underlying statistical structure of the data; right panel: the data is rotation-invariant, so any other directions could have been chosen.

strap (see [30]). However, their approach is tailored to specific ICA algorithms and it is sometimes not straightforward to define a resampling strategy that preserves the statistical structure relevant to the considered ICA algorithm.

We will propose a different approach that views the chosen ICA algorithm as a black box. Any ICA algorithm that refers to the inherent ideas of ICA can be plugged into our procedure. As we have seen in Chapter 2 the statistical structure relevant for ICA algorithms is non-Gaussianity, non-flatness and non-stationarity. Our method partially destroys this structure by corrupting the data with stationary white Gaussian noise. The motivation for this strategy is, that we expect reliable components to be extracted even if they have forfeited some of their structure, whereas unreliable projections (which were chosen by chance) will be lost in the process. The general idea is to measure reliability as stability with respect to noise, i.e. to a fading-out of the marginal non-properties: non-Gaussianity, non-flatness and non-stationarity.

Chapter outline

In Sections 3.1 and 3.2 we present our approach based on noise-injection: Section 3.1 introduces new performance functions for angle deviations and groupings that are by definition invariant to permuted and scaled solutions of the applied ICA black box. Furthermore, we expound some interesting theoretical interpretations of the noise-injection method. For the implementation, care has to be taken with respect to normalizations along the different steps of the algorithm. Section 3.2 explains step by step our procedure. Finally, we validate the method in Section 3.3 on artificially generated signals and real-world data from a fetal ECG.

3.1 Theoretical motivation

We first introduce some further notations that facilitate the following theoretical discussion. Also we define some performance measures that will be useful to evaluate the reliability and grouping structure of independent components.

3.1.1 Preliminaries

Like in Chapter 2, we represent multivariate time-series $x[t]$ consisting of n components each of length T as an $n \times T$ matrix,

$$X = [x[1] \cdots x[T]]. \tag{3.1}$$

We assume that all signals, i.e. the rows of X, have zero mean. An ICA algorithm that estimates the mixing matrix A from X determines implicitly also the demixing matrix $W = A^{-1}$. Thus we can view more formally the ICA algorithm under consideration as a function

$$\begin{array}{rccc} \Theta & : & \Re^{n \times T} & \rightarrow & \Re^{n \times n} \\ & & X & \mapsto & W \end{array} \tag{3.2}$$

that maps a data matrix X to a demixing matrix W. With this notation, the matrix containing the demixed signals can be written as

$$Y = \Theta(X)X. \tag{3.3}$$

For simplicity, we often ignore in the following discussion the scaling and permutation invariance of ICA algorithms.

The fact that Θ is an ICA algorithm implies some simple properties of this mapping, which we will utilize below: the demixing matrix of already demixed signals, $\Theta(X)X$, is the identity matrix,

$$\Theta(\Theta(X)X) = I. \tag{3.4}$$

Remixing the given data before demixing changes the demixing matrix by right-multiplying it by the inverse of the remixing,

$$\Theta(BX) = \Theta(X)B^{-1}. \tag{3.5}$$

Both equalities hold up to permutation and scaling, the invariances of any ICA solution.

3.1.2 Injecting noise

In Chapter 2 we have seen that the statistical structures that most algorithms for ICA exploit are non-Gaussianity, non-stationarity and non-flatness. In order to analyze the reliability of the extracted components $Y = \Theta(X)X$, we partially destroy their statistical structure by adding noise N which is stationary Gaussian distributed and independent in time, i.e. we look at the signals $Y + \sigma N$ with σ being a small constant (that we discuss later). As a result, the latter signals are more Gaussian, more stationary and spectrally more flat. Inspired by the neural networks community, where noise is injected to the data in order to achieve regularization effects (see [13]), we call this idea noise-injection.

The next step examines how the noise-injection influences the demixing matrix,

$$C = \Theta(Y + \sigma N), \tag{3.6}$$

which we get by applying again the ICA algorithm to the noise-injected signals. Since C would be the identity matrix for the noiseless case (for $\sigma = 0$, see Equation (3.4)), we consider measures of performance,

$$\begin{array}{rccc} \mu & : & \Re^{n\times n} & \rightarrow & \Re \\ & & C & \mapsto & \mu(C), \end{array} \tag{3.7}$$

that value (some aspect of) the deviation of C from the identity matrix:

Amari index. To measure how close the matrix C as a whole is to a scaled permutation matrix, we define a slightly modified Amari-index (replacing $|\cdot|$ used in [6] by $(\cdot)^2$ to ensure differentiability):

$$\mu(C) = \sum_{i=1}^{n}\left(\sum_{j=1}^{n}\frac{C_{ij}^2}{\max_k C_{ik}^2} - 1\right) + \sum_{j=1}^{n}\left(\sum_{i=1}^{n}\frac{C_{ij}^2}{\max_k C_{kj}^2} - 1\right). \tag{3.8}$$

Note that C has to be row-wise normalized by left-multiplication with a diagonal matrix, i.e. the norms of the rows of C are one. The first sum of the Amari index is small if each column of C has exactly one dominating element. The second sum becomes small if each row of C has exactly one dominating element. If both sums are small, C is close to a permutation matrix. The Amari index is invariant to left-multiplication by a permutation matrix.

Angle distance. Instead of calculating a single real number that considers the total demixing matrix C at once (like the Amari index does), we can also focus on the j-th column of C to obtain information solely about the j-th component of Y, written as $Y_{j:}$. Assume that the rows of C have been normalized to unit norm and that the components of Y are normalized to unit variance. Further, let $Z = C(Y + \sigma N)$ be the result of demixing the noise-injected signals. The absolute values of the i-th row of C determine with what proportions of the components $Y_{1:}, \ldots, Y_{n:}$ the signal $Z_{i:}$ is assembled. These proportions can be interpreted as the cosines of certain angles given by $\arccos(|C_{ij}|)$. These angles characterize the relation between $Z_{i:}$ and $Y_{j:}$. If $|C_{ij}|$ is one, we can infer due to the row-normalization of C that only $Y_{j:}$ has contributed to $Z_{i:}$, i.e. $Z_{i:} = \pm Y_{j:}$. Accordingly, the corresponding angle is zero, $\arccos(1) = 0$. That means that $Y_{j:}$ has been recovered without any distortions of other components. Thus we say in that case that the angle deviation of $Y_{j:}$ is zero.

Note, that the j-th column could have several entries equal to one, implying that $Y_{j:}$ is represented in Z by more than one signal with angle zero. If the angle is $\pi/2 = \arccos(0) = \arccos(|C_{ij}|)$, the component $Y_{j:}$ does not appear in the i-th component of Z.

In general, the j-th column of C characterizes to which components of Z the signal $Y_{j:}$ contributes. The largest among the absolute values of the entries in the j-th column of C determines the smallest angle that $Y_{j:}$ has with any of the components of $Z_{i:}$. This suggests the following definition: we call the smallest angle between $Y_{j:}$ and the components of Z,

$$\mu_j(C) = \min_i \arccos |C_{ij}|, \tag{3.9}$$

the angle distance of the j-th component of Y (to the closest component of Z). If this angle distance is small even after injecting noise, the corresponding component was reliably estimated in the initial demixing, otherwise it is not reliable. Note, that taking the absolute value of each entry ensures that the orientation is ignored for the calculation of the angle.

Grouping matrix. The discussion on the angle distance in the previous item can be extended to motivate the definition of the grouping matrix. Initially, we replace all entries of C by their absolute values, since we are only interested in proportions and the orientations should have no influence. If we assume further that the norms of the rows of C are normalized to one, the column vectors indicate with what proportions the components of Y represent the components of Z as explained in the previous item. If each component of Y is mapped by C to exactly

one component of Z, the columns of C are orthogonal to each other, i.e. their inner products vanish. If one of Z's components is a mixture of two components of Y, the inner product of the corresponding columns does not vanish, because both columns would contain non-zero entries in the corresponding row. We see that the inner products between different columns of C quantify whether two components of Y are part of a common subspace. The entries of the grouping matrix collects the inner products between all column vectors of C,

$$\mu_{jk}(C) = |C_{:j}|\,|C_{:k}| = \sum_{i=1}^{n} |C_{ij}|\,|C_{ik}|. \tag{3.10}$$

If $\mu_{jk}(C)$ is large, there is at least one component of Z in which both signals $Y_{j:}$ and $Y_{k:}$ coincide with large proportions. If that is the case, $Y_{j:}$ and $Y_{k:}$ contribute to the same subspace and are grouped together. A small value indicates that the two corresponding components $Y_{j:}$ and $Y_{k:}$ are mapped to different subspaces, so they are not grouped together.

Angle distance for subspaces. Simple methods from linear algebra, that calculate angles between spaces, enable us to generalize the angles between vectors to angles between subspaces allowing us to assess the reliability of subspaces.

Other performance measures being tailored to specific situations can be used in our framework as well, as long as they have the following properties:

- μ is greater or equal to zero with equality for the identity matrix (and for permutation matrices).
- μ is continuous and differentiable in a neighborhood around its minimum, i.e. around a permutation matrix.

In the following theoretical discussion, we consider μ to be the Amari index. However, all results are equally valid for other measures that fulfill the properties mentioned above, i.e. also for the angle distance of a single component and for the entries of the grouping matrix, because they have those properties.

Repeating the noise-injection for different instances of noise and averaging the resulting indices, we are estimating the expected value of the Amari-index with respect to the Gaussian noise, i.e. we are calculating

$$E_N \mu(\Theta(\Theta(X)X + \sigma N)). \tag{3.11}$$

This expression has a nice interpretation: define[1] $\varphi(X) = \mu(\Theta(X))$. Then writing out the Taylor expansion[2] of φ around $\Theta(X)X$, the expected Amari-index with respect to

[1] This seemingly unnecessary definition helps us to write down the matrix differentials as unambiguously as possible.

[2] In the appendix, we briefly review the necessary definitions of matrix differentials.

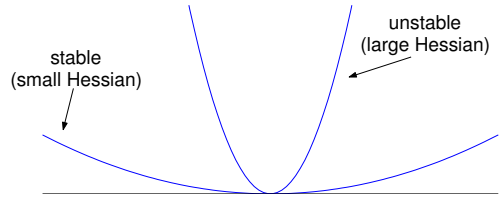


Figure 3.2: The Hessian of φ reflects the curvature of φ. If it is at the solution $\Theta(X)X$ small, the solution is stable; if it is large, the solution is unstable.

the noise,

$$\begin{aligned} & E_N \mu(\Theta(\Theta(X)X + \sigma N)) \\ = \; & E_N \varphi(\Theta(X)X + \sigma N) \\ \approx \; & \varphi(\Theta(X)X) \\ & + \sigma E_N \mathsf{D}\varphi(\Theta(X)X) \operatorname{vec} N \\ & + \sigma^2 E_N (\operatorname{vec} N)^\top \mathsf{H}\varphi(\Theta(X)X) \operatorname{vec} N \\ = \; & \sigma^2 \operatorname{tr} \mathsf{H}\varphi(\Theta(X)X), \end{aligned} \tag{3.12}$$

surprisingly simplifies to the trace of the Hessian of φ at $\Theta(X)X$. We assumed σ to be small, such that the higher-order terms could be omitted. For the last equality, we utilized:

- that $\varphi(\Theta(X)X) = \mu(\Theta(\Theta(X)X)) = \mu(I) = 0$,
- that $E_N \operatorname{vec} N$ is the null vector (or alternatively, that $\mathsf{D}\varphi(\Theta(X)X)$ is the null matrix, because $\Theta(X)X$ is a minimum of φ and φ is continuous and differentiable in some neighborhood around $\Theta(X)X$), and
- that $E_N (\operatorname{vec} N)^\top G \operatorname{vec} N = \operatorname{tr}(G)$ for any squared matrix G, because the entries of N are i.i.d. normally distributed.

Slightly rewriting this,

$$\frac{E_N \mu(\Theta(\Theta(X)X + \sigma N))}{\sigma^2} \approx \operatorname{tr} \mathsf{H}\varphi(\Theta(X)X), \tag{3.13}$$

we see that the result of our noise-injection method can be interpreted as a stochastic derivative of φ at $\Theta(X)X$, or with other words, as a Monte-Carlo-style approximation of the curvature of the Hessian of φ at $\Theta(X)X$, see Figure 3.2. Note, that for this theoretical result it is not important how we choose $\sigma > 0$. Even for comparing different algorithms (which result into different φ), this method is useful, since the estimate of our method does not depend on σ (as long as σ is small enough to allow us to neglect higher-order terms, and σ is large enough to get a measurable variation). However, the next section will show that the right-hand side expression of Equation (3.13) does not directly estimate the reliability measure based on the true distribution (see

Equation (3.14)). Though we observe a monotonic relationship in that situation, the estimated value depends on σ. Later in Section 3.3.1, we will present an experiment that shows that σ can be chosen such that this relationship is even linear, and in Section 3.3.4 we will use σ to fade in and out of the grouping matrix.

3.1.3 Measuring reliability based on the true distribution

The data matrix X represents one data point sampled from some unknown underlying probability distribution $P : \Re^{n \times T} \to [0, 1]$. In practical situations, we are only given this single data point X (yet this is usually enough to estimate a demixing matrix, since the distribution of X is already determined by much lower-dimensional distributions; for example, the marginal distributions for algorithms based on non-Gaussianity). A measure of reliability based on the true distribution P has to refer to other data points X', drawn from the same P (obviously, no other data sets are available in practice). In order to compare the demixing matrix estimated from X, i.e. $\Theta(X)$, to some other estimate $\Theta(X')$, we simply invert the later and multiply it to the former and measure how close the result is to a permutation matrix. This, as before, can be achieved by the Amari-index. Simple transformations (for the first equality use Equation (3.5)) imply for the mathematical description of this idea,

$$\begin{aligned} & E_{X'}\mu(\Theta(X)\Theta(X')^{-1}) \\ = \; & E_{X'}\mu(\Theta(\Theta(X')X)) \\ = \; & E_{X'}\mu(\Theta(\Theta(X)X + (\Theta(X') - \Theta(X))X)) \end{aligned} \tag{3.14}$$

where the expectation is taken with respect to $P(X')$. If the data sampled from P contains some statistical information that can be exploited by the ICA algorithm, the difference between the two estimated demixing matrices will be small, i.e. $(\Theta(X') - \Theta(X))X$ is also small compared to $\Theta(X)X$. Furthermore, its rows are approximately independent and, again compared to $\Theta(X)X$, it does not contain much statistical information (i.e. non-Gaussianity, non-flatness, non-stationarity, see [21]). Inspired by ideas from perturbation theory, it is therefore plausible to replace $(\Theta(X') - \Theta(X))X$ by a small noise term and exchange the expectation over X' by an expectation over the noise N, i.e.

$$E_N\mu(\Theta(\Theta(X)X + \sigma N)), \tag{3.15}$$

which is exactly our approach of noise injection. Note, that though we exchanged the expectation over the unaccessible distribution of P by an expectation over noise, which can easily be simulated, we did not replace X' by $X + \sigma N$, but instead terms in which X' has already been processed to a demixing matrix.

However, despite this interesting connection between our method and the truth, Expressions (3.14) and (3.15) are generally not equal. The reason is that $\Theta(X') - \Theta(X)$ depends on the reliability as well, which is ignored after substituting the noise. Yet, our experience shows that there is a monotonic relationship between those two expressions, which we can currently not describe in full mathematical rigor. In Section 3.3.1 however, we perform experiments that show that for the right choice of σ this relationship can even become linear.

3.1.4 Relation to bootstrap resampling

Standard bootstrap resampling (see [29]) requires that the column vectors of the data matrix X are independent and identically distributed, which is not the case for non-white or non-stationary signals. The latter need more sophisticated resampling techniques (see[71]) that are specially tailored to the ICA algorithm under consideration. However, the design of an appropriate resampling strategy is not always straightforward.

These bootstrap-type methods have in common that they approximate the true underlying distribution P by some empirical estimate. How is our method approaching the true distribution? Observing that

$$\begin{aligned} & E_N \mu(\Theta(\Theta(X)X + \sigma N)) \\ = \quad & E_N \mu(\Theta(X + \sigma\Theta(X)^{-1}N)\Theta(X)^{-1}), \end{aligned} \tag{3.16}$$

(again, applying Equation (3.5)), we see that our method replaces the true unknown distribution P by a kernel density estimate that is based solely on the single data point X, and hereby mimicking the expression[3]

$$E_{X'} \mu(\Theta(X')\Theta(X)^{-1}). \tag{3.17}$$

However, in our situation this coarse density estimate is sufficient, since as mentioned above, we do not analyze the resampled data points themselves. For our reliability analysis, we are only interested in the corresponding demixing matrices $\Theta(X + \sigma\Theta(X)^{-1}N)$. Note, that in Expression (3.16) the noise is added to the mixed signals X (and not to the unmixed signals $\Theta(X)X$, as in Expression (3.16)), and our formalism (here in the shape of the correct application of Equation (3.5)) ensures that the added noise is appropriately spatially colored by $\Theta(X)^{-1}$.

3.2 Algorithmic details

Instead of simply using the Amari-index to judge the whole demixing matrix at once, in this section we will estimate the stability for each component separately (using the angle distances) and try to detect potential block structures (using the grouping matrix).

Step 1: Fixing the scaling indeterminacy

Bearing in mind the usual indeterminacies of ICA solutions, i.e. arbitrary scaling and permutation, we can assume without loss of generality that the mixing matrix A, the inverse of $W = \Theta(X)$, has unit-length columns, i.e.

$$A_{:j}^{\top} A_{:j} = 1 \text{ for } j \in \{1, \ldots, n\}. \tag{3.18}$$

[3] Note, that in general Expressions (3.17) and (3.14) are not perfectly equal, yet for most μ they are closely related.

This ensures that the energy of each component $Y_{j:}$ is equal to the sum of the energies of the proportions of $Y_{j:}$ in the components of X, which can be written as $A_{:j}Y_{j:}$, mathematically speaking

$$\begin{aligned}\frac{1}{T}\operatorname{tr}(A_{:j}Y_{j:}Y_{j:}^\top A_{:j}^\top) &= \frac{1}{T}\operatorname{tr}(A_{:j}^\top A_{:j}Y_{j:}Y_{j:}^\top)\\ &= \frac{1}{T}\operatorname{tr}(Y_{j:}Y_{j:}^\top)\\ &= \frac{1}{T}Y_{j:}Y_{j:}^\top,\end{aligned} \tag{3.19}$$

using for the first equality $\operatorname{tr}(CD) = \operatorname{tr}(DC)$, for the second the fact that the columns of A have unit length and for the third that $Y_{j:}Y_{j:}^\top$ is a scalar.

Step 2: Adjusting the noise levels to the signal energies

Note, that the noise level has to be adjusted for each component separately: otherwise some components—the weak ones—might loose all their statistical structure while others—the strong ones—are not affected at all, which would be undesirable since such a procedure would favor strong components over weak components.[4]

Let E be the matrix that contains the square roots of the energies of the extracted components on the diagonal or, equivalently speaking, their standard deviations,

$$E = \begin{bmatrix} \sqrt{\frac{1}{T}Y_{1:}Y_{1:}^\top} & & 0 \\ & \ddots & \\ 0 & & \sqrt{\frac{1}{T}Y_{n:}Y_{n:}^\top} \end{bmatrix}. \tag{3.20}$$

Being aware of the fact that this matrix contains not exactly energies, we call it nonetheless for simplicity "energy matrix".

Adding R instances of Gaussian white noise, written as $n \times T$ matrices $N^{(1)}$, …, $N^{(R)}$, that have been adjusted by the energy matrix E, to the extracted components Y provides us with R versions of Y, the statistical structure of which has been partially destroyed:

$$\begin{aligned}\tilde{Y}^{(1)} &= \cos(\chi)Y + \sin(\chi)EN^{(1)}\\ &\vdots\\ \tilde{Y}^{(R)} &= \cos(\chi)Y + \sin(\chi)EN^{(R)}.\end{aligned} \tag{3.21}$$

$0 \le \chi \le \pi/2$ is a parameter that can be visualized as a turning knob: $\chi = 0$ adds no noise, i.e. all statistical structure is preserved, $\chi = \pi/2$ produces only noise, i.e. all

[4] Equivalently, we could normalize the extracted signals to variance one and add noise with the same noise level for each of the signals. We pursue the slightly more complicated way that preserves the true variances in order to avoid further alterations of the statistical structure.

statistical structure is destroyed and in between some structure is kept, some is destroyed. Since $\cos^2(\chi) + \sin^2(\chi) = 1$ it is guaranteed that the noisy versions have in each component the same energy as their colleagues in Y.

Note, that choosing χ corresponds to choosing σ which was used in the theoretical motivation. In the experiments, we fix $\chi = \pi/8$ using empirical evidence: a simulation in Section 3.3.1 will show that for this particular value the estimated index corresponds to the true index. Additionally, we use in Section 3.3.4 the full range of χ to fade into the grouping matrix.

Step 3: Demixing again

The versions with the damaged statistical structure are mixed by randomly generated mixing matrices $B^{(1)}, \ldots, B^{(R)}$, which have unit-length columns. Hereby, we obtain mixtures in which the initially extracted components keep their energy. The additional mixing is important for algorithms, the performance of which might depend on the starting conditions. Since the remixing matrices are known, they can be taking into account in the later analysis.[5]

By applying the chosen ICA algorithm to the remixed versions $B^{(1)}\tilde{Y}^{(1)}, \ldots, B^{(R)}\tilde{Y}^{(R)}$, we obtain demixing matrices $V^{(1)}, \ldots, V^{(R)}$ and hereby demixed signals,

$$\begin{aligned} Z^{(1)} &= V^{(1)}B^{(1)}\tilde{Y}^{(1)} \\ &\vdots \\ Z^{(R)} &= V^{(R)}B^{(R)}\tilde{Y}^{(R)}. \end{aligned} \tag{3.22}$$

Step 4: Constructing the relevant transformation

Transforming the remixed noisy versions $B^{(r)}\tilde{Y}^{(r)}$ to $Z^{(r)}$ can be seen as demixing the remixed, initially extracted components $B^{(r)}Y$, but ignoring the part of the statistical structure, that has been destroyed by injecting the noise.

Instead of computing the Amari-index, as we did for simplicity in the theoretical discussion of Section 3.1, for the angle distances and the grouping matrix we have to calculate the angle between a new component $Z^{(r)}_{i:}$ and an initially extracted component $Y_{j:}$. These angles allow us to evaluate each component separately and to find groupings, as we will see below. For this purpose, two things have to be ensured:

1. We need to consider the transformation with respect to the normalized signals

$$Y^{normalized} = E^{-1}Y \tag{3.23}$$

each having variance one. Due to

$$\begin{aligned} Z &= V^{(r)}B^{(r)}Y \\ &= V^{(r)}B^{(r)}EY^{normalized} \end{aligned} \tag{3.24}$$

[5] Note, that this remixing has no effect for equivariant algorithms.

the transformation to proceed with is

$$V^{(r)}B^{(r)}E. \tag{3.25}$$

2. The transformed signals must be normalized as well, i.e. we have to left-multiply $V^{(r)}B^{(r)}E$ by a diagonal matrix $D^{(r)}$ such that the rows of

$$U^{(r)} = D^{(r)}V^{(r)}B^{(r)}E \tag{3.26}$$

have unit-norm. We refer to this matrix as $U^{(r)}$.

The latter matrix describes the relevant transformation: the angle between $Z_{i:}^{(r)}$ and $Y_{j:}$ is the arcus cosine of the absolute value of the ij-th entry of that matrix,

$$\alpha_{ij}^{(r)} = \arccos(|U^{(r)}|_{ij}), \tag{3.27}$$

which is some number between 0 and $\pi/2$. Note, that we have to take the absolute value of each matrix entry, since orientation should not influence the calculation of the angle.

Step 5: Estimating reliability and grouping structure

Using these angles, we compute the statistics regarding the initially extracted components Y. To begin with, we calculate for each component the root mean-squared angle distance (RMSAD) to $Y_{j:}$:

$$\begin{aligned} v_j &= \sqrt{\frac{1}{R}\sum_{r=1}^{R}(\min_i \alpha_{ij}^{(r)})^2} \\ &= \sqrt{\frac{1}{R}\sum_{r=1}^{R}(\mu_j(U^{(r)}))^2}, \end{aligned} \tag{3.28}$$

using for the expression in the second line the notation from Section 3.1. These values estimate the uncertainty of the extracted components Y, as we will see in the experiments section. A large RMSAD means unreliable, a small RMSAD means the corresponding component is reliable.

Furthermore, we calculate a matrix that displays the grouping structure of the extracted signals, which is called the mean grouping matrix (or simply grouping matrix):

$$\begin{aligned} S_{jk} &= \frac{1}{R}\sum_{r=1}^{R}|U_{:j}^{(r)}|^\top|U_{:k}^{(r)}| \\ &= \frac{1}{R}\sum_{r=1}^{R}\sum_{i=1}^{n}|U_{ij}^{(r)}||U_{ik}^{(r)}| \\ &= \frac{1}{R}\sum_{r=1}^{R}\mu_{jk}(U^{(r)}) \end{aligned} \tag{3.29}$$

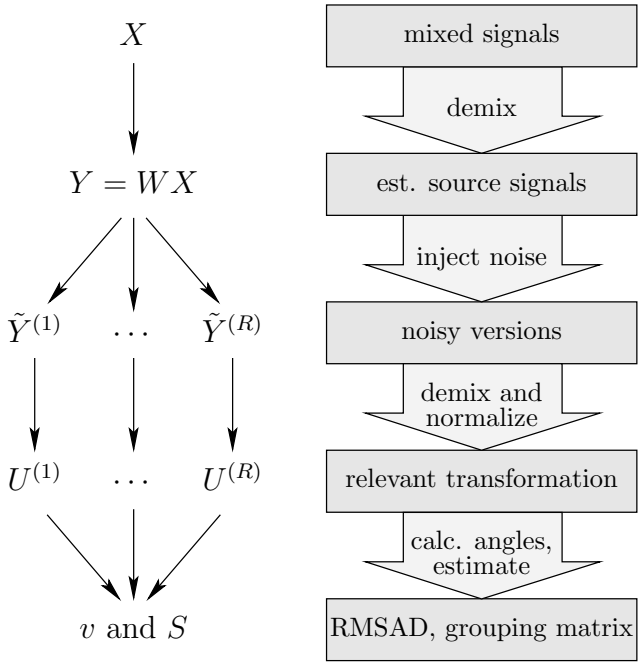


Figure 3.3: This schematic view shows our method at a glance.

employing for the expression in the third line the notation of Section 3.1. Note, that the possible block-structure of S can be automatically obtained by repeatedly sorting its submatrices.

3.3 Experiments

In order to further validate our approach, we show empirically that for a particular choice of χ our noise-injection method approximates the true angle deviations. Next, we examine a toy data set, where we are able to compare the results to the ground truth. After that, we explore some biomedical data. Finally, we compare the grouping matrices resulting from different amounts of noise, i.e. different values of χ.

For all experiments, we use the same three algorithms, each going exclusively one of the easy routes to ICA: as an example of an algorithm that exploits only non-Gaussianity we use JADE (see [24]), for non-flatness TDSEP (see [112] and also [12]) and for non-stationarity a simplistic variant of SEPAGAUS (see [84]), which we forced to ignore all statistical information but non-stationarity[6].

[6]The input parameters of SEPAGAUS allow us to specify that only one positive frequency channel will be used.

3.3.1 True versus estimated RMSAD

Transferring the results of Section 3.1 to the RMSAD, we see that our method approximates the trace of the Hessian of the angle distance, independent of the choice of $\chi > 0$. This is already enough to compare different algorithms. However, fine-tuning this parameter allows us to obtain an estimate of the true RMSAD.

For the simulation in this section, we use two sound sequences, that represent two processes. Adding noise creates processes that contain different amounts of statistical information. By taking parts of the whole sequence we can get different instances of the same process, that we randomly mix with a known matrix. To calculate the true RMSAD, we estimate the projection directions (rows of the demixing matrix) for 100 different mixed instances. To obtain our estimated RMSAD, we considered one instance. We apply noise-injection to this instance to get different noisy versions from which we can calculate our estimate of the RMSAD (using Equation (3.28)). Note, that we have used noise in two ways:

1. In order to create signals with differently strong pronounced statistical structures, we added noise to some given sound signals. This noise influences both their true and their estimated RMSAD. However, it has nothing to do with the noise-injection method.

2. The other noise process is essential for our noise-injection method to produce the noisy version of one instance of the process. This enables us to estimate the true RMSAD.

The coordinates of each point in Figure 3.4 show the true and estimated RMSAD for one particular process (i.e. one level of noise on the initial sound sequences). In all three plots we observe that these values are correlated. This means that our estimate of the RMSAD for one particular instance is with high probability close to the true one. Therefore for this setting of χ the estimated RMSAD approximates the true RMSAD. Note, that this finding depends very much on the choice of χ which controls the signal to noise ratio. In all experiments reported in this section, χ has been fixed to $\pi/8$.

3.3.2 Toy data

Running our method in a completely controlled environment enables us to easily evaluate our results. We consider seven signals that show different combinations of statistical structure (see [71]):

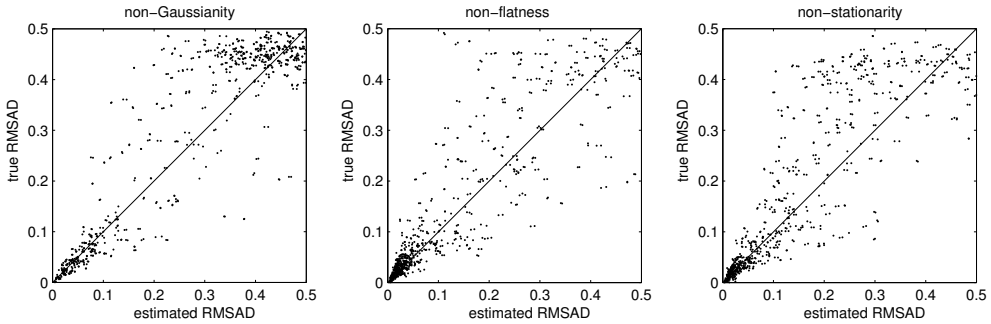


Figure 3.4: The estimated RMSAD correlates in all three cases with the true RMSAD. Note, that these scatterplots depend on the choice of the noise parameter χ. All experiments have been carried out with $\chi = \pi/8$.

7 signals	non-Gaussianity	non-flatness	non-stationarity
Speech	+	+	+
Music	+	+	+
Cosine	+	+	-
Sine	+	+	-
Uniform noise	+	-	-
Gaussian noise	-	-	-
Gaussian noise	-	-	-

These signals are mixed by a randomly chosen matrix and analyzed with the three ICA algorithms mentioned above. The results are visualized in Figure 3.5:

- The RMSAD—depicted in the right-most column—reveals which components have been recovered reliably: all three algorithms are able to find the two sound sources (RMSAD ≤ 0.0002), since these signals have a rich statistical structure (a combination of non-Gaussianity, non-flatness and non-stationarity). Because the signal consisting of uniform noise (signal 5) has neither non-flatness nor non-stationarity (time structure), only the algorithm exploiting non-Gaussianity (first row) detects that source.

- However, the grouping structure can not be directly infered from the RMSAD: the RMSAD bar-plot for the non-flatness-based algorithm (second row) resembles the bar-plot for the non-stationarity-based algorithm (third row). But the corresponding grouping matrices exhibit the underlying grouping structure, which is very different: signals 3 to 7 are for the non-stationarity-based algorithm one five-dimensional subspace, but for the non-flatness-based algorithm there are two independent subspaces: one two-dimensional space for the sine and cosine (which contain time structure, i.e. non-flatness), and a three-dimensional space for the noise signals that have no time structure. This matches exactly what we expect from the statistical properties of the signals (see the table above).

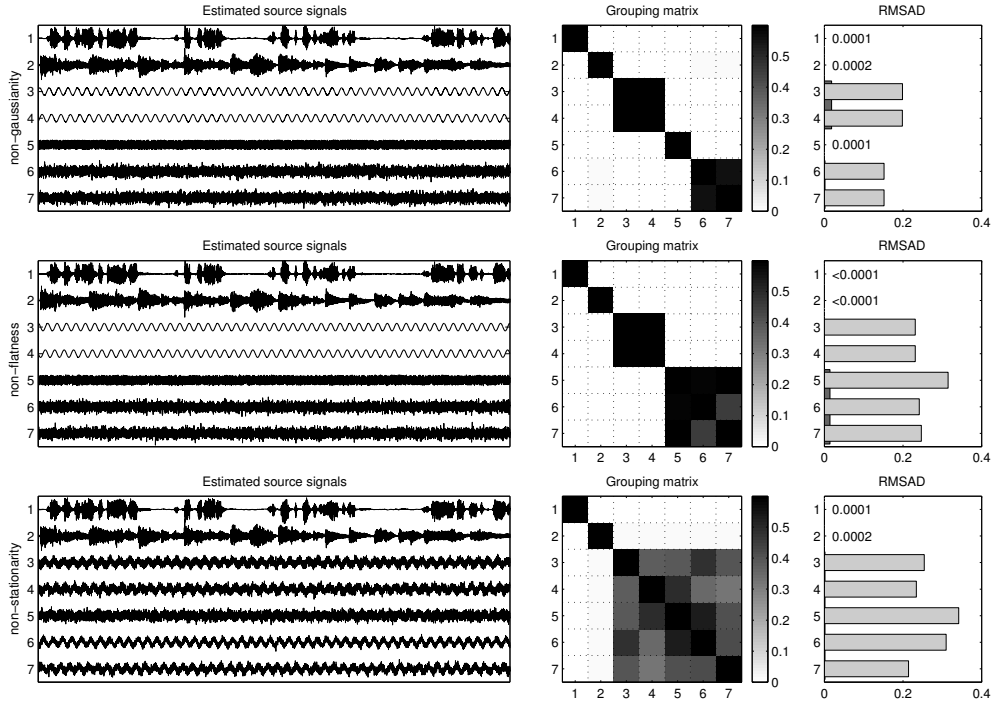


Figure 3.5: Shown are the results for the toy data. The structure revealed by the grouping matrix and by the RMSADs fits the table in the explaining text. For the first row we used JADE, for the second row we used TDSEP with time-lags $0, \ldots, 20$, for the third we used SEPAGAUS being restricted to one positive frequency channel.

- The grouping matrix for non-Gaussianity shows five blocks that correspond to five subspaces. The two-dimensional subspaces—sine versus cosine and Gaussian noise versus Gaussian noise—are examples with rotation-invariant distributions, which have no preferred ICA basis (that is the reason why none of the three algorithms was able to find one, see their grouping matrices).

- For subspaces, we can also calculate angles to the rest of the signals (RMSAD for subspaces). For the algorithm based on non-Gaussianity, we plotted the RMSAD for the sine/cosine subspace (signals 3 and 4) and for the Gaussian noise subspace (signals 6 and 7) as dark bars behind the corresponding light bars. We see that the RMSAD for the subspaces is much smaller than the RMSAD for the corresponding signals (for signals 6 and 7 the bar is too small to be visible). Similar, for the non-flatness-based algorithm the RMSAD of the subspace spanned by signals 3 and 4 and the space spanned by signals 5, 6 and 7 is much smaller than the RMSADs of the corresponding signals. The same holds for the subspace spanned by signals 3, 4, 5, 6 and 7 in the third row. We see, that unreliable components might be part of a very reliable subspace.

Note, that this finding does not rank the three non-properties: it reflects only the statistical properties of the investigated signals, which is exactly what our method is intended to do.

3.3.3 Fetal ECG

As an example of a real-world data set, we present results on fetal ECG data (from [59]) which contains 2500 data points sampled at 500Hz with 8 electrodes located at abdomen and thorax of a pregnant woman. The goal of applying ICA algorithms to this data is to separate the fetal heartbeat from the heartbeat of the mother. However, the practitioner might ask: which ICA algorithm should be used? The uncertainty, shown as the RMSAD in the right-most column of Figure 3.6, reveals that the non-Gaussianity based algorithm (JADE, see [24]) is the method of choice for this data set, which is in agreement with Meinecke et al. [71]. The grouping matrices underline this, because they clearly show that six independent signals have been found. Finally, looking at the estimated waveforms in the first row of the figure, we see that channels 1, 2, 3 and 4 contain the mother's heartbeat and channels 7 and 8 the fetal's heartbeat. The other algorithms did not extract those signals reliably.

3.3.4 Fading into the grouping structure

In order to illustrate the influence of the amount of added noise, we run the noise-injection procedure for varying noise parameters χ. Figure 3.7 shows the grouping matrices (left columns for the earlier used toy data; right columns for the fetal ECG data) as a function of the amount of noise (top to bottom).

Without adding noise (first row) we would expect diagonal grouping matrices. However, for the chosen ICA algorithm based on non-stationarity (SEPAGAUS), we obtain by repeated application different solutions for entries of the stationary signals which is

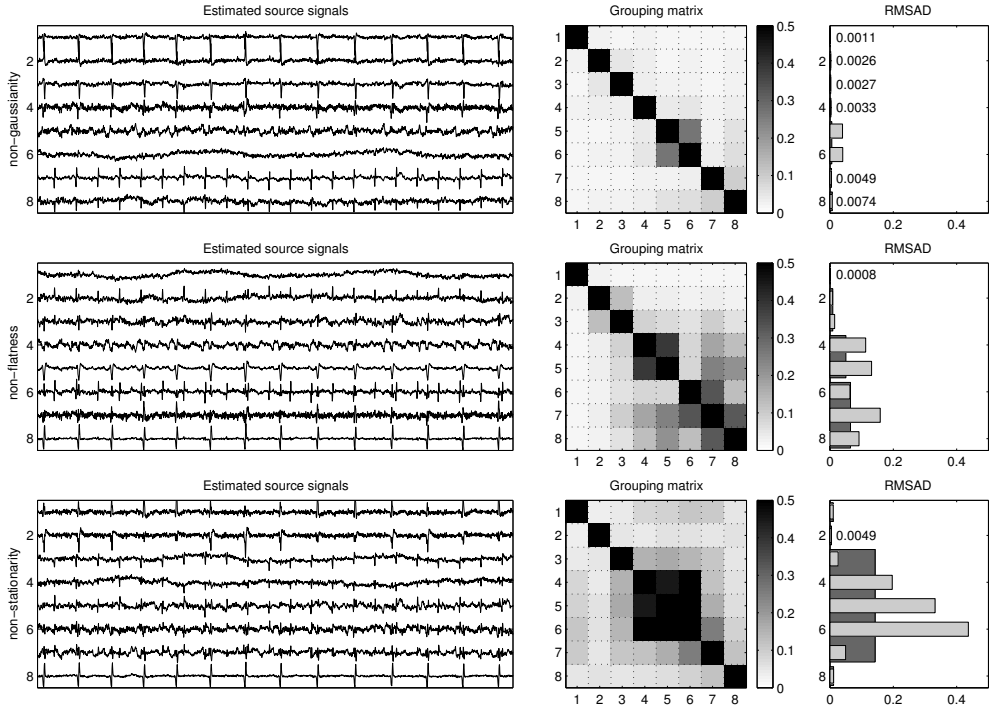


Figure 3.6: Shown are the results for the fetal ECG. As explained in the text, the extracted components from the ICA algorithm based on non-Gaussianity are the most reliable. For the first row we used JADE, for the second row we used TDSEP with time-lags $0, \ldots, 20$, for the third we used SEPAGAUS being restricted to one positive frequency channel.

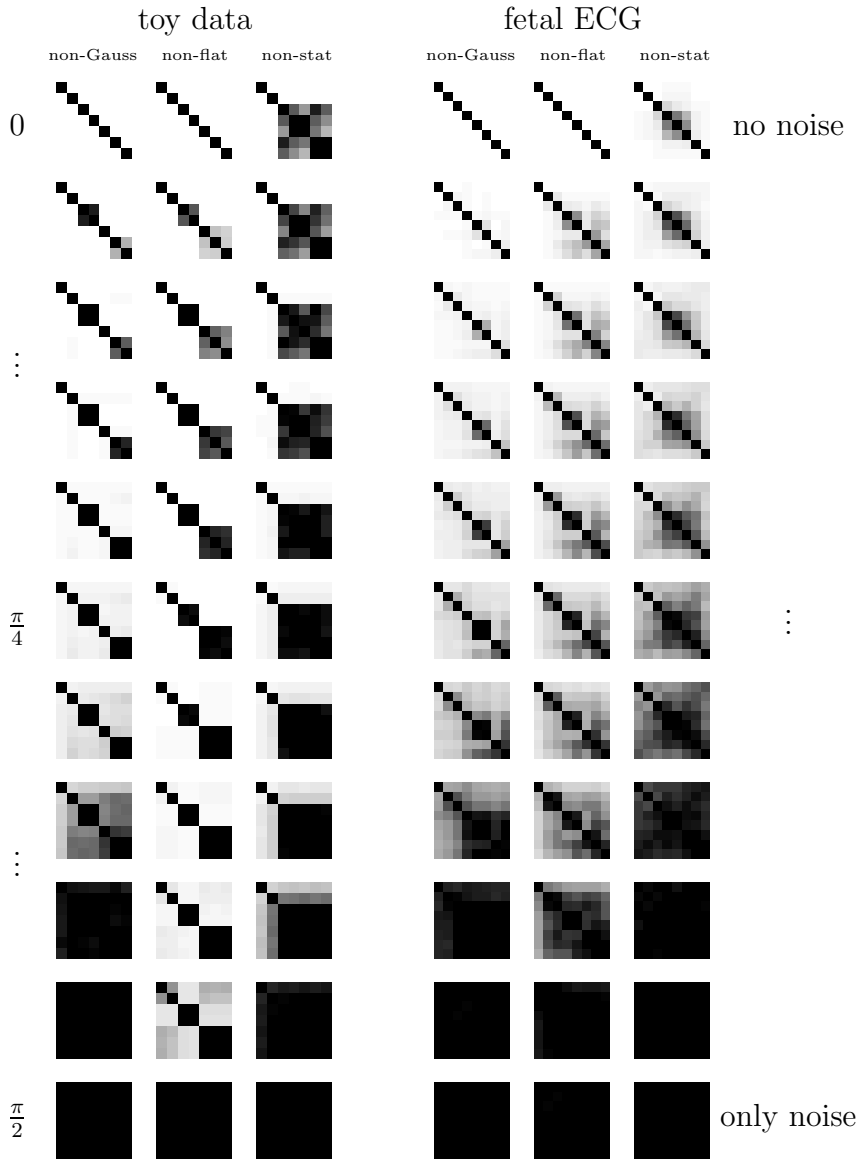


Figure 3.7: Fading into the grouping structure: top to bottom rows increases the amount of noise-injection from no noise (first row) to only noise (bottom row).

due to the fact that SEPAGAUS uses random initial matrices. Destroying the whole signal and keeping only the noise (last row), we observe as expected black matrices, i.e. the no structure can be found by the ICA algorithms. From top to bottom we see how the grouping structures (shown for $\chi = \pi/8$ earlier in Figures 3.5 and 3.6) fades in for smaller $\chi < \pi/8$ and fades out for larger $\chi > \pi/8$.

It is important to note that all entries of the grouping matrix increase monotonically from top to bottom. We see further that the grouping structure is visible for a large range of χ, suggesting that the choice of χ is not critical for the structure of the grouping matrix.

3.4 Summary

In this chapter we presented a new method to assess the reliability of independent components that can be easily applied to any ICA algorithm. Our approach has some nice theoretical interpretations and is thus well motivated. Furthermore, we showed empirically that our estimate approximates the true RMSAD. The noise-injection procedure can be seen as an alternative to the bootstrap resampling approach described in [71] with a similar complexity. However, our new method is easier to apply because it does not have to be adjusted to the ICA algorithm under consideration. Controlled toy experiments and experiments with fetal ECG data underline the usefulness of this approach.

4 Robust and overcomplete ICA with inlier detection

The objective functions of typical ICA algorithms (see Chapter 2) are often sensitive to outliers (especially algorithms based on kurtosis). A simple strategy to robustify existing algorithms is to apply outlier detection as a preprocessing step. In this chapter, we will follow a different idea: we show that a simple outlier index can be used directly to solve the ICA problem for super-Gaussian (or sparse) source signals. The key idea is that data points in very dense regions characterize the mixing matrix. We use outlier indices to find the points in the most concentrated areas. We call these points inliers. Those inliers directly determine the sought-after ICA directions.

For example, the left panel of Figure 4.1 shows a scatterplot of a two-dimensional mixture of four super-Gaussian source signals. The true directions are indicated by small lines on the circle. In the scatterplot they are also visible as the directions with higher density. This is in unison with the circular histogram (with 180 bins) of the distribution of the angles of the data points shown in the right panel. The directions of highest density correspond to the columns of the mixing matrix. To automatically find these directions, we apply simple outlier indices to the data points (after projecting them onto the unit sphere) in order to sort the data from points in dense regions—the inliers—to points in sparse regions—the outliers. The columns of the mixing matrix are identified by simply selecting points among the inliers which are the points of highest density. We call this new ICA algorithm inlier-based ICA (IBICA).

IBICA is not restricted to the classical setting with equal number of sources signals

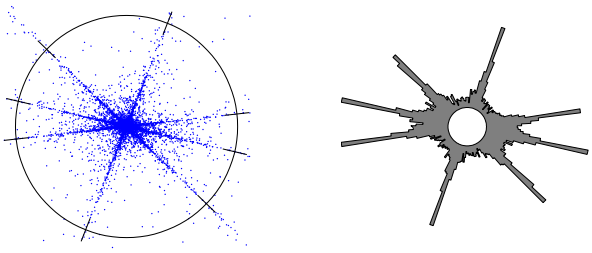


Figure 4.1: The left panel shows a scatterplot of a two-dimensional mixture of four super-Gaussian source signals. The little lines on the circle indicate the true ICA directions. The right panel shows a circular histogram (with 180 bins) of the distribution of the angles of the data. The directions of highest density correspond to the columns of the mixing matrix.

and mixed signals. Instead, IBICA applies naturally to the overcomplete setting and it is able to unmix strongly overcomplete mixtures (such as the mixture in Figure 4.1) as we will study in detail in the experiment section.

Since we use in this chapter outlier indices only to find inliers, we call them for simplicity inlier indices.

Chapter outline

In Section 4.1 we introduce simple, but powerful inlier indices based on nearest neighbors methods. In Section 4.2 we explain step by step how these inlier indices are employed by IBICA. In Section 4.3 we compare IBICA in the classical setting with other standard ICA algorithms with respect to robustness against kurtotic noise and outliers. Next, we empirically study IBICA for the overcomplete case and examine its performance with respect to the number of sources, the number of data points, the dimensionality of the data and the number of nearest neighbors (the only parameter of IBICA). Finally, we apply IBICA to an artificial image separation task for complete and overcomplete mixtures.

4.1 Inlier indices

Let $\text{dist} : \Re^n \times \Re^n \to \Re$ be a distance measure for the data points $z[1], z[2], \ldots, z[T] \in \Re^n$. We denote the k nearest neighbors of a data point $z \in \Re^n$ among $z[1], z[2], \ldots, z[T]$ by

$$\text{nn}_1(z), \ldots, \text{nn}_k(z) \in \{z[1], z[2], \ldots, z[T]\} \subset \Re^n. \tag{4.1}$$

so we have:

$$\text{dist}(z, \text{nn}_1(z)) \leq \text{dist}(z, \text{nn}_2(z)) \leq \ldots \leq \text{dist}(z, \text{nn}_k(z)). \tag{4.2}$$

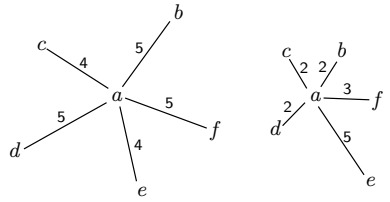
If there are several points that have the same distance to z (with respect to dist), we order them arbitrarily, to have the nearest neighbors well-defined. Using the nearest neighbors, we define two indices for each point $z \in \Re^n$. These will be the essential ingredients for the IBICA algorithm.

4.1.1 Kappa

The k-nearest neighbor density estimator assesses the density at a particular point by calculating the volume of the smallest ball centered at that point which contains its k nearest neighbors and relating it to the quotient k/T. It can be proven that this density estimator is L_2-consistent (see [63]). The first index represents the essence of the k nearest neighbor density estimator: $\kappa(z)$ is the radius of the smallest ball centered at z containing its k nearest neighbors, i.e. the distance between z and its k-th nearest neighbor,

$$\kappa(z) = \text{dist}(z, \text{nn}_k(z)). \tag{4.3}$$

In dense regions κ is small and in sparse regions κ is large.



$\kappa(a) = 5$ $\gamma(a) = 4.6$ $\kappa(a) = 5$ $\gamma(a) = 2.8$

Figure 4.2: Because γ takes into account all k nearest neighbors and not only the k-th like κ does, γ can distinguish better between sparse and dense regions than κ for $k = 5$.

4.1.2 Gamma

The index κ is somewhat wasteful: it considers the distance to the k-th nearest neighbor, but it ignores the distances to the closer neighbors. This suggests a refined index that considers all k nearest neighbors: $\gamma(z)$ is z's average distance to its k nearest neighbors,

$$\gamma(z) = \frac{1}{k} \sum_{j=1}^{k} \text{dist}(z, \text{nn}_j(z)). \tag{4.4}$$

This index enables us to distinguish the two situations depicted in Figure 4.2: the value of κ is in both situations the same, because the 5th nearest neighbor of a has both times the same distance to a, although the neighborhood on the right is denser. By exploiting all distances, γ can distinguish both situations.

4.1.3 Simple properties of κ and γ

Observing that the average of k distances $d_1, \ldots, d_k$ is bounded by their maximum, i.e. $(d_1 + \ldots + d_k)/k \leq \max(d_1, \ldots, d_k)$, we see that γ is bounded by κ,

$$0 \leq \gamma(z) \leq \kappa(z), \tag{4.5}$$

This means that if $\gamma(z)$ is large (implying that z is probably an outlier) also $\kappa(z)$ is large. On the contrary, if $\gamma(z)$ is small, then $\kappa(z)$ needs not to be small, since it might have ignored some relevant information. κ might misjudge a point from a denser region to be an outlier because it considers only the k-th nearest neighbor which might be far. With other words, κ might miss an inlier—a point in a dense region—that γ might be able to find. Note that the scales of these indices depend on the total number of data points that are considered for the k nearest neighbors. The more points are taken into account, the closer will be the neighbors and thus the smaller will be the indices.

We have extensively studied these indices for outlier detection and for robustification of algorithms (see [35]). Along these lines, we could calculate κ and γ for the given data

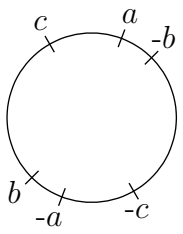


Figure 4.3: Since $-b$ is closer to a than c and $-c$, we have $\mathrm{dist}(a, b) < \mathrm{dist}(a, c)$.

$x[1], \ldots, x[T]$ with the Euclidean distance. This would allow us to remove outliers as a preprocessing step, hereby robustifying standard ICA algorithms that are sensitive to outliers. This idea will not be discussed in this chapter. Instead we used κ and γ directly to solve the ICA problem: the columns of the mixing matrix A are for super-Gaussian sources the inliers among the given data points with respect to some distance function specifically taylored to the ICA setting.

4.2 Inlier-based ICA

Due to ICA's scaling invariance, the orientations and norms of the column vectors of the mixing matrix A are irrelevant: a column vector of A is solely characterized by the one-dimensional subspace that it spans. Since we want to choose the columns of A among the one-dimensional subspaces spanned by the given data points $x[1], \ldots, x[T]$, the distance function, that we want to plug into the inlier indices, should not depend on their orientations and norms.

Step 1: project the data points onto the unit sphere

To eliminate the influence of the norms, we project all data points $x[1], \ldots, x[T]$ onto the unit sphere,

$$z[t] = \frac{x[t]}{\sqrt{x[t]^\top x[t]}} = \frac{x[t]}{|x[t]|} \tag{4.6}$$

(we assume zero mean). Prior to this normalization step we remove points that are very close to the origin, because in noisy settings the directions of these points do not contain much information about the true ICA directions. Furthermore, we avoid division by zero for points located exactly at the origin.

Step 2: calculate the inlier index for all data points

The Euclidean distance between the normalized points depends only on the directions of the original points $x[1], \ldots, x[T]$. For the ICA setting we further require an invariance with respect to their orientations: for example, the distance between a and $-a$

should be zero. This suggests to define the distance of two points a and b on the unit sphere as,

$$\mathrm{dist}(a,b) := \min(|a-b|, |a-(-b)|) = \min(|a-b|, |a+b|), \tag{4.7}$$

see Figure 4.3. This distance function does not depend on the orientation of the involved vectors, i.e.

$$\mathrm{dist}(a,b) = \mathrm{dist}(-a,b) = \mathrm{dist}(a,-b) = \mathrm{dist}(-a,-b), \tag{4.8}$$

thus being the natural distance for the ICA problem[1].

Having established the distance function, we apply the inlier indices introduced in the previous section. We define the κ and γ values of $z \in \mathcal{Z} := \{z[1], \ldots, z[T]\}$ to be

$$\kappa(z) = \mathrm{dist}(z, \mathrm{nn}_k(z)) \tag{4.9}$$

$$\gamma(z) = \frac{1}{k}\sum_{j=1}^{k} \mathrm{dist}(z, \mathrm{nn}_j(z)). \tag{4.10}$$

Intuitively speaking, $\gamma(z)$ is large if there are only a few other points close to the subspace spanned by z. With other words, z lies on the unit sphere in a sparse region, implying that z might be an outlier. If $\gamma(z)$ is small, then there are many points scattered along the subspace spanned by z. With other words, z lies in a dense region, implying that z is an inlier. The data points with the small γ are good candidates for the columns of A. A similar reasoning holds for κ with the differences pointed out in Section 4.1.

Step 3: pick the columns of the mixing matrix

In order to form the mixing matrix A, we could pick those m data points with the smallest γ values (respectively κ values) and stack them together. This approach will correctly provide only the first direction. However, the remaining $m-1$ columns of A might originate all from the same direction, which by chance happened to be located in a very dense cluster on the unit sphere. This problem is solved by iteratively defining several indices $\zeta^{(1)}, \zeta^{(2)}, \ldots$ based on γ (respectively based on κ) that avoid picking directions similar to directions already chosen.

Denote the columns of A by $a_1, \ldots, a_m \in \Re^n$. The first column of A is the vector $z \in \mathcal{Z}$ with the smallest γ. Thus defining for $z \in \mathcal{Z}$

$$\zeta^{(1)}(z) := \gamma(z), \tag{4.11}$$

[1] Note that dist is based on the Euclidean distance which is closely related to the geodesic distance: for two points a and b on the unit sphere, the geodesic distance is the angle between those vectors, i.e. $\arccos(a^\top b)$. For small angles this distance is proportional to the Euclidean distance and in general the relationship is monotonic, i.e. $\arccos(a^\top b) < \arccos(a^\top c)$ if and only if $|a-b| < |a-c|$ with c being another point on the unit sphere. We could also base dist on the geodesic distance. However, since for the inlier indices we are particularly interested in small distances, we would obtain similar results.

the first column vector of A is

$$a_1 := \arg\min_{z \in \mathcal{Z}} \zeta^{(1)}(z). \tag{4.12}$$

To avoid choosing a direction similar to a_1, we define a new index $\zeta^{(2)}$ that penalizes directions close to a_1,

$$\zeta^{(2)}(z) := \frac{\gamma(z)}{\operatorname{dist}(a_1, z)}. \tag{4.13}$$

Accordingly, the second column of A is

$$a_2 := \arg\min_{z \in \mathcal{Z}} \zeta^{(2)}(z). \tag{4.14}$$

Next, we want to avoid the directions of a_1 and a_2 simultaneously, so we define

$$\zeta^{(j)}(z) := \frac{\gamma(z)}{\min_{i<j} \operatorname{dist}(a_i, z)} \tag{4.15}$$

which is the general definition of $\zeta^{(j)}$ for $j > 1$. Similarly, we define the j-th column of A to be

$$a_j := \arg\min_{z \in \mathcal{Z}} \zeta^{(j)}(z). \tag{4.16}$$

Following this recipe, we iteratively determine the columns of A. Note that for $j > 1$ we have

$$\zeta^{(j)}(z) \leq \max\left(\zeta^{(j)}(z), \frac{\gamma(z)}{\operatorname{dist}(a_j, z)}\right) = \zeta^{(j+1)}(z) \tag{4.17}$$

and thus

$$\min_{z \in \mathcal{Z}} \zeta^{(j)}(z) \leq \min_{z \in \mathcal{Z}} \zeta^{(j+1)}(z). \tag{4.18}$$

Denoting the smallest value of each iteration by

$$\zeta_j := \min_{z \in \mathcal{Z}} \zeta^{(j)}(z) = \zeta^{(j)}(a_j), \tag{4.19}$$

we see that for $j > 1$ these values increase monotonically[2],

$$\zeta_2 \leq \zeta_3 \leq \dots \tag{4.20}$$

If all true source directions have been picked—say m directions—we can expect that ζ_{m+1} is much larger than ζ_m, because for all data points with small γ (respectively small κ) a direction close to them has been already chosen. Therefore, a large step in $\zeta_2, \zeta_3, \dots$ determines the number of sources. In practice this works well if there are enough data points as we will see in the experiments section.

[2]Note that ζ_1 is not necessarily smaller than ζ_2: suppose for some $z \in \mathcal{Z}$ we have $\zeta_1 = \gamma(a_1) < \rho\,\gamma(a_1) = \gamma(z)$ with $1 < \rho < \sqrt{2}$. Assume further that z has maximal distance to a_1 on the unit sphere with respect to dist, i.e. $\operatorname{dist}(a_1, z) = \sqrt{2}$. Then we have $\zeta^{(2)}(z) = \rho\,\gamma(a_1)/\sqrt{2} < \gamma(a_1) = \zeta_1$ and thus $\zeta_2 < \zeta_1$.

Computational costs

Step 1 of IBICA—the normalization step—costs $O(nT)$ with n being the dimensionality of the observed data and T the number of data points. For step 2, the whole distance matrix needs to be calculated, costing $O(nT^2)$. Finding the k-th nearest neighbor of one point can be done in linear time (selection in expected linear time, see [27]). For κ we have to do this for each point, i.e. $O(T^2)$. So, the total time complexity for κ is $O(T^2 n)$. γ requires all k nearest neighbors, which can be found (using k times selection in expected linear time) in $O(kT)$ for each point, i.e. in total $O(T^2 n + T^2 k)$ for all points. For large k, we can also sort all neighbors of a point (in $O(T \log T)$), i.e. in total $O(T^2 n + T^2 \log T)$ for all points. Summarizing, step 2 with κ requires $O(T^2 n)$ and with γ we need $O(T^2 n + T^2 \max(k, \log T))$. To simplify, we can say that the time complexity of step 2 is $O(T^2 \log T)$. Picking one direction in step 3 requires to sort the current index (in $O(T \log T)$), to calculate the distances to the last chosen direction (in $O(nT)$) and to update the indices (in $O(T)$). So, in total IBICA requires $O(T^2 \log T)$.

The computational costs of step 2 can be dramatically reduced by randomly partitioning the data into smaller disjoint subsets of approximately equal size and calculating for each subset the requested index. The scale of the indices changes with the number of data points, i.e. limiting the calculation to a subset, increases the scale compared to considering all data points. However, since all subsets have equal size, the resulting indices are comparable. Note that the smaller the subsets are chosen, the coarser the indices will be.

4.3 Experiments

The following experiments systematically study the properties of IBICA. First we compare IBICA in the classical setting to the standard ICA algorithms JADE [24] and FastICA [51] with respect to kurtotic noise and outliers. After that we examine the performance of IBICA in the classical and the overcomplete case with respect to the number of sources, its dimensionality, the choice of k, the number of data points.

Throughout the experiments we partitioned data sets with more than 1000 points into partitions of size approximately 1000 in order to speed-up the simulations. For simplicity, we fixed k, number of nearest neighbors, to 50 for all experiments (not in the study on the choice of k).

If not stated otherwise, all simulations use m-dimensional artificially generated sources $s[t]$, which are super-Gaussian signals that are obtained in three steps: (i) start with an $m \times T$ matrix of Gaussian noise; (ii) take its entries to the power of five; (iii) normalize such that each row has unit variance. The $n \times m$-dimensional mixing matrices A are also randomly generated in two steps: (i) start with an $n \times m$ matrix of Gaussian noise; (ii) normalize such that each column has unit variance. All curves show the median over 100 repetitions because the actual performance can depend on the actual realization of the involved data. In all comparisons all algorithms were applied to the same randomly created signals.

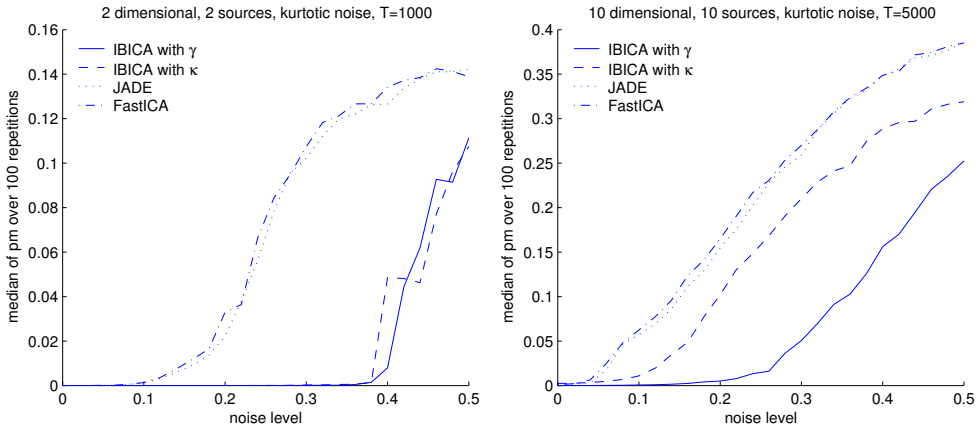


Figure 4.4: IBICA is more robust with respect to kurtotic noise than JADE and FastICA (left panel: two dimensions; right panel: 10 dimensions).

4.3.1 Performance measures

To investigate the performance of IBICA, we define a performance measure that compares the directions of the columns of two mixing matrices that are not necessarily square. Let A and B be two column-normalized $n \times m$ matrices, i.e. their columns have unit norms. We define a performance measure that compares the directions of the columns of A and B,

$$\mathrm{pm}(A, B) = 1 - \left(\frac{1}{2n} \sum_{i=1}^{n} \max_j |A^\top B|_{ij} + \frac{1}{2n} \sum_{j=1}^{n} \max_i |A^\top B|_{ij} \right) \tag{4.21}$$

with $|A^\top B|_{ij}$ being the ij-th entry of the matrix $A^\top B$. The absolute value ensures that the orientation of the columns is ignored. The performance measure is symmetric and bounded between zero and one. $\mathrm{pm}(A, B) = 0$ implies that the columns of A span the same one-dimensional subspaces as B and vice versa, or more precisely, for each column in A there is exactly one column in B that spans the same one-dimensional subspace. Thus, A and B perform the same mixing up to permutation of the source signals.

4.3.2 Robustness against kurtotic noise and outliers

In the classical ICA setting (i.e. $m = n$) we compare IBICA with JADE and FastICA. To show the robustness of IBICA we first consider the case where the mixed signals have been contaminated with kurtotic noise,

$$x[t] = As[t] + \sigma\eta[t], \tag{4.22}$$

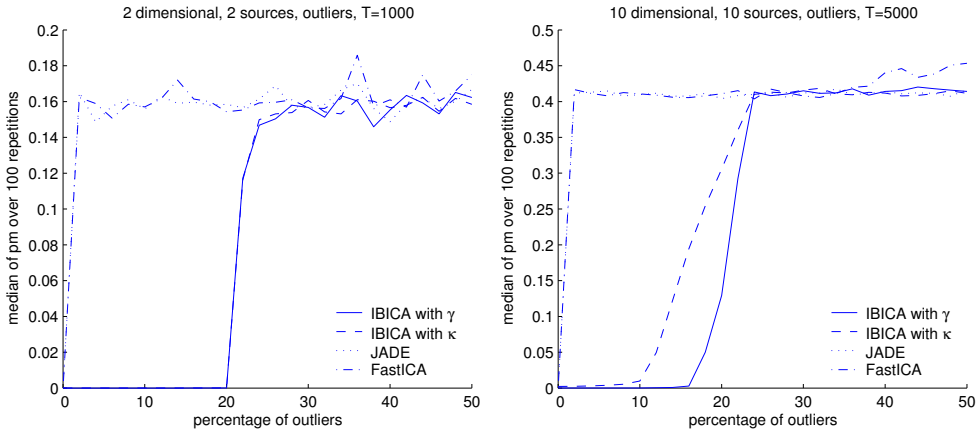


Figure 4.5: IBICA is very robust with respect to outliers: replacing a significant amount of the data with outliers does not change the result of IBICA (left panel: two dimensions; right panel: 10 dimensions).

with A being a square matrix. The kurtotic noise $\eta[t]$ is created in three steps: (i) start with an $n \times T$ matrix of Gaussian noise; (ii) transform its entries such that the column norms are taken to the power of nine; (iii) normalize such that each row has unit variance. We plot the performance measure pm as a function of the noise level σ for the kurtotic noise. Figure 4.4 shows that JADE and FastICA fail for smaller noise levels than IBICA for two-dimensional (left panel) and 10-dimensional data (right panel). In the two-dimensional case this difference is more pronounced than in 10 dimensions. We can also observe that IBICA based on γ is superior to IBICA based on κ for the 10-dimensional data.

To demonstrate the robustness of IBICA with respect to outliers, we replaced a certain percentage of the data points of the mixed signals, $x[t] = As[t]$, with A being a square matrix, by Gaussian noise with standard deviation 100. Figure 4.5 shows the performance measure pm as a function of the percentage of outliers. Adding only a few outliers spoils the solutions of JADE and FastICA. This is in contrast to the performance of IBICA: in two dimensions one fifth of the data points (20 percent) can be replaced by outliers without any performance loss. This shows the exceptional stability of IBICA which is due to the fact, that the columns of the mixing matrix are chosen among the data points in dense regions. Hereby, outliers are ignored and are not relevant for the process of determining A. Also in 10 dimension, up to 15 percent outliers do not influence the result of IBICA based on γ. Though IBICA with κ is stable up to 10 percent outliers, we see again that it is inferior to IBICA with γ.

4.3.3 Overcomplete mixtures in two dimensions

Figure 4.6 shows several results of applying IBICA based on γ with $k = 50$ to 15 randomly generated overcomplete mixtures in two dimensions with varying number

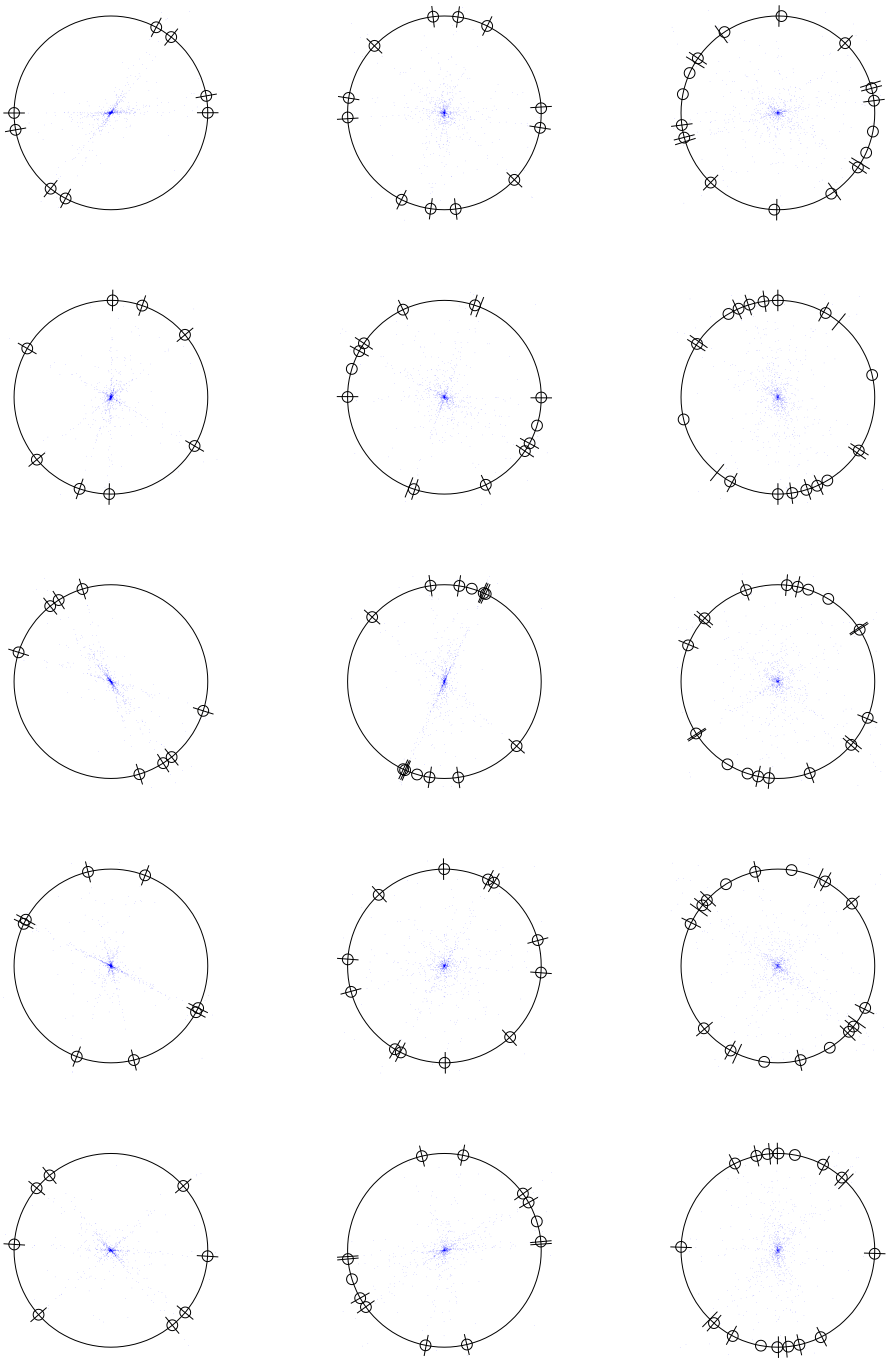


Figure 4.6: IBICA based on γ with $k = 50$ applied to 15 randomly generated overcomplete mixtures in two dimensions with varying number of sources. Each example consists of only 1000 data points. The short lines indicate the true directions, the little circle the estimated directions.

of sources (left column four sources, middle column six sources, right column eight sources). Each large circle contains the center of the scatterplot of the mixed data. The short lines on the large circle indicate the true signal directions, the little circles on the large circle the directions chosen by IBICA. We plot as many directions of IBICA as there are true directions.

The mixtures with four sources (left column) are perfectly identified by IBICA which is a good result especially by considering the examples in the fourth and fifth rows which contain directions that are very close to each other.

The mixtures with six sources (middle column) are sometimes recovered perfectly (first and fourth row). However, in the second row IBICA misses one source direction, probably because two true direction are very close to each other. Though in the third row there are three true directions close together, IBICA missed only one. In the fifth example, IBICA did not find one of the directions that was very close to a direction already chosen.

In the mixtures with eight sources (right column) usually one or two of the true directions are missed and the other directions are sometimes not precisely found. However, the setting for these randomly generated experiments is chosen such that we can see how the performance of IBICA breaks down. Since all examples in Figure 4.6 consist only of 1000 data points, the results are quite remarkable. Note further, that increasing the number of data points improves the accuracy (see Figure 4.8).

Suppose next that the correct number of sources is not known. Figure 4.7 shows stairs plots of $\zeta_1, \ldots, \zeta_{12}$ for the examples in Figure 4.6. The arrows indicate the true number of sources (left column four sources, middle column six sources, right column eight sources). In the left column (four sources) a gap is clearly visible in all stairs plots indicating the number of sources correctly to be four.

In the middle column (six columns) the gap is correct only for the first row. In this example also the directions were chosen correctly by inspecting Figure 4.6. The performance was also good for identifying six directions in the fourth row. However, the corresponding stairs plot shows a gap at the fifth step (and not at the sixth). This can be understood by looking at the corresponding scatterplot in Figure 4.6 which shows that there are two close directions. One of them is probably the sixth chosen direction. However, since already a similar direction has been chosen earlier, the indices do not show a large gap.

In the right column (eight sources) there are no clear gaps indicating the correct number of sources. However, increasing the number of data points, the gaps become more pronounced as we see in Figure 4.8. For the example with 5000 data points (right panels) we did not partition the data, because otherwise we would not profit from the larger data set for the determination of the nearest neighbors. Also note that $\zeta_1, \ldots, \zeta_{12}$ is on a smaller scale for 5000 data points (lower right panel) than for 1000 data points (lower left panel). γ scales down for larger data sets, because the nearest neighbors are closer. Note that we used for both examples $k = 50$. As the circle plots show (upper panels), IBICA determined also the true directions more precisely with more data points.

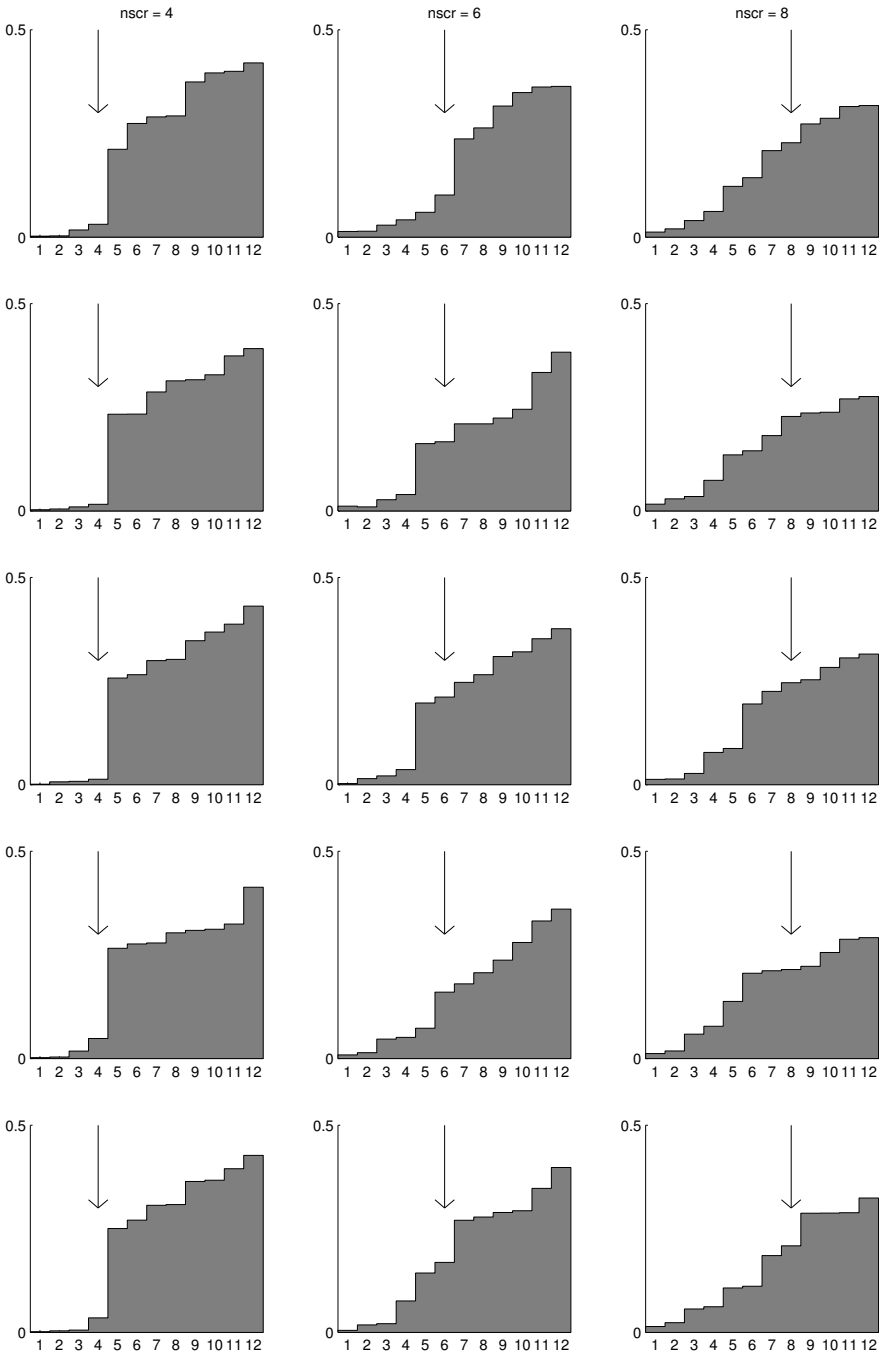


Figure 4.7: If the number of sources is not known, large gaps in the stairs plots of $\zeta_1, \ldots, \zeta_{12}$ can help. Shown are the stairs plot for the examples in Figure 4.6. Up to four sources the true number of sources can be identified, see also Figure 4.8.

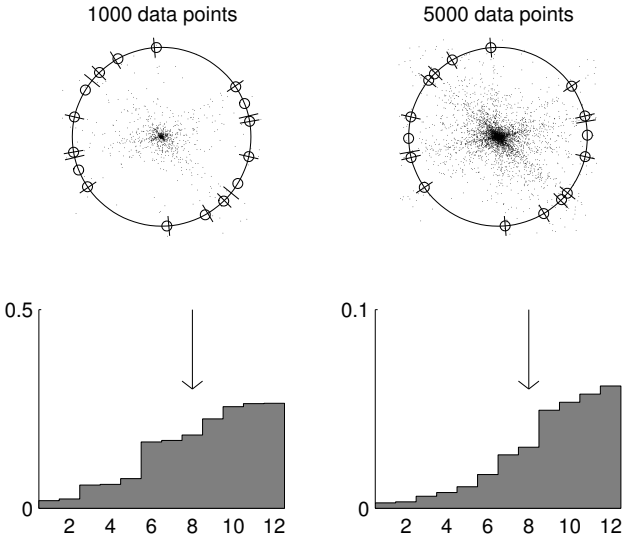


Figure 4.8: The gap in the stairs plot of $\zeta_1, \ldots, \zeta_{12}$ becomes more pronounced if we increase the number of data points.

4.3.4 Performance as a function of the number of sources

In another series of simulations we studied the performance of IBICA as a function of the number of sources as shown in Figure 4.9 for four-dimensional data. We increase the number of sources up to 40 sources. Since the performance also depends on the number of data points we run the experiments for 1000 and 3000 data points (left and right panels). As a baseline we also show the performance for a random solution (lower panel). Note that the performance function decreases with the number of sources. This effect is due to the fact, that the more random directions we pick in four dimensions the more might match by chance. Note that the performance of the random solution does not depend on the number of data points.

First of all, we see that the performance measure for less than 15 sources is much smaller than the baseline (lower panel). IBICA with γ degrades later than IBICA with κ for both situations (left and right panels). In general, we see that with more data points more sources can be found (left versus right panel). For 1000 data points, it is curious that the performance of IBICA with κ seems to decrease for more than 20 sources (left panel). However, in the lower panel we see a similar effect as noted above. So, the reason for the performance decrease of IBICA with κ is that it is not able to find a reasonable mixing matrix for more than 20 sources, and so, the more sources we have, more columns are correctly found by chance. We also observe, that the performance measure of IBICA with γ for 40 sources in four dimensions is still well below the baseline.

How is the performance of IBICA affected by the dimensionality of the mixed sig-

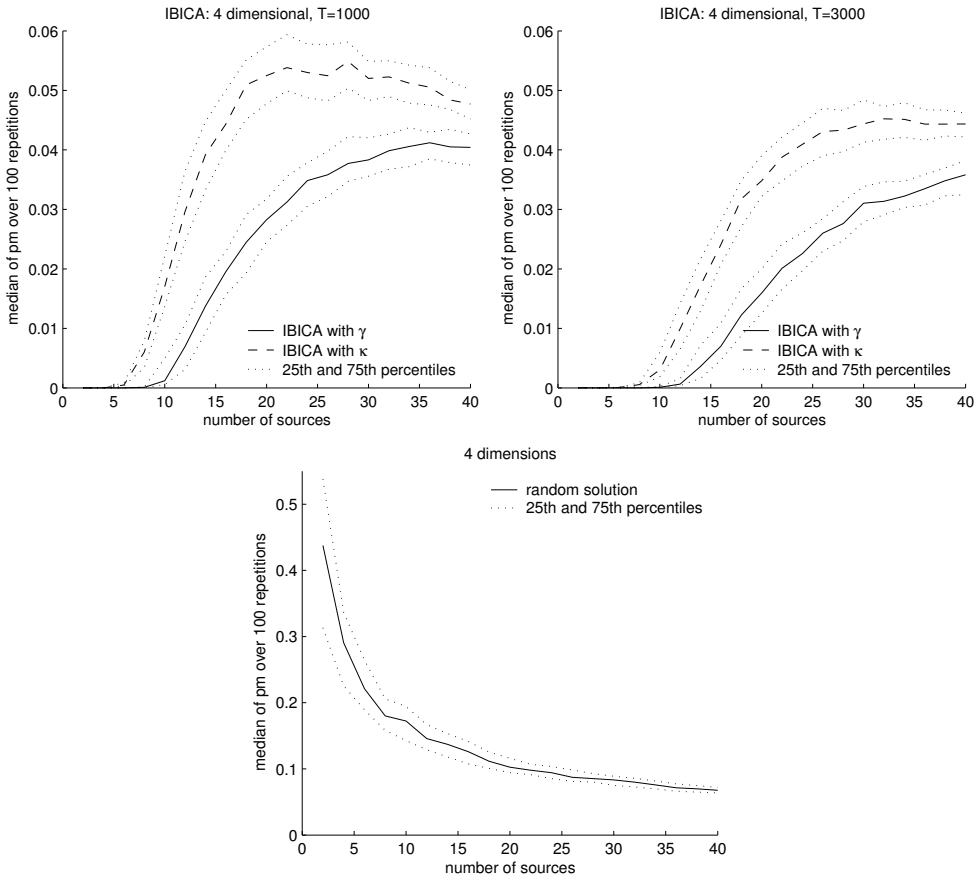


Figure 4.9: How is the performance of IBICA affected by the number of sources and the number of data points (left versus right panel)? As a baseline, the lower panel shows the performance measure applied to a random solution.

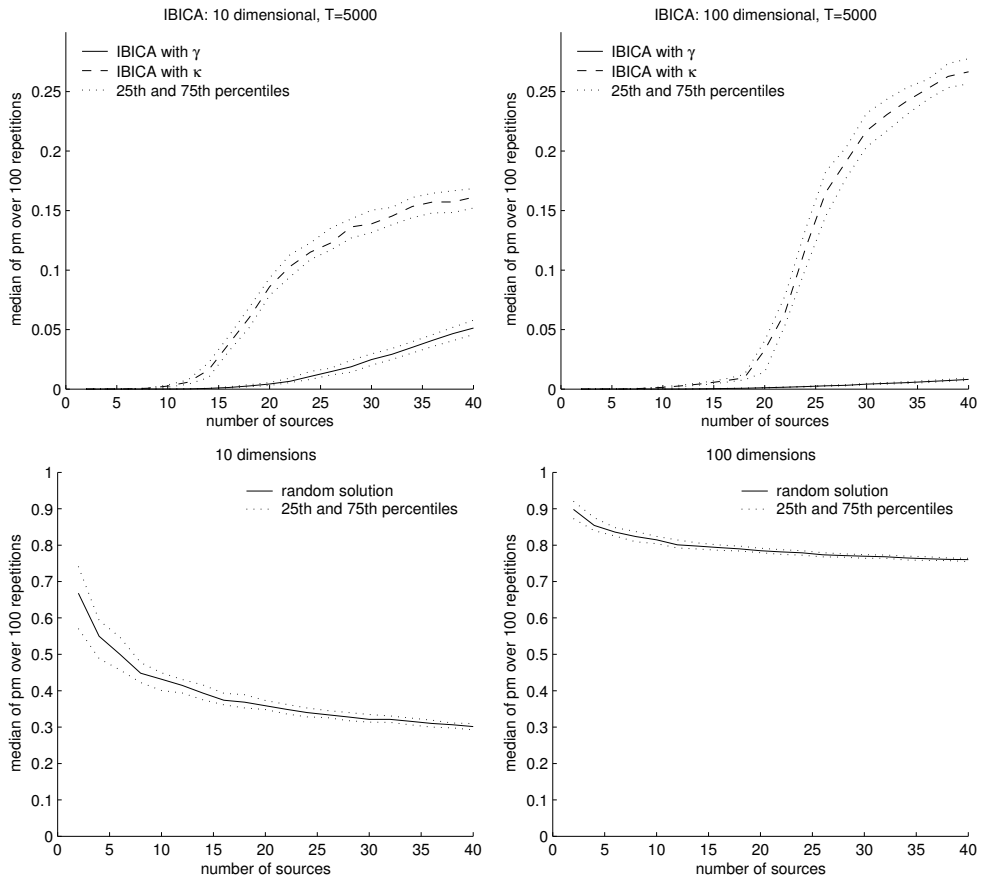


Figure 4.10: How is IBICA affected by the number of dimensions? The pm scales are not comparable between different dimensionalities (lower panels).

nals? Figure 4.10 shows the performance measure as a function of the number of sources for 10 dimensions (left panels) and 100 dimensions (right panels) using 5000 data points. First of all, we note that it is difficult to compare the performance between different dimensionalities as we see in the lower panels, that show the performance values for randomly generated solutions in 10 and 100 dimensions (similar to the lower panel in Figure 4.9). The performance measure has larger values for higher dimensions, which is also consistent with the lower panel in Figure 4.9 that shows performance values for four dimensions that are smaller than the values for 10 dimensions (lower left panel). The reason for these differences is that in higher dimensions, two randomly chosen vectors are more likely to be orthogonal to each other. Also we observe that the performance values decrease with increasing number of sources, which is in agreement with the lower panel of Figure 4.9 and due to the fact that with more source directions more randomly chosen directions can match as discussed already above.

For almost up to 10 sources we do not see a difference between the curves in the upper panels of Figure 4.10 for the different dimensions. With more then 10 sources the mixture becomes overcomplete in 10 dimensions and the performance of IBICA with κ gradually decreases getting closer to the baseline (lower left panel). The performance values of IBICA with γ stay longer very small and only increase slightly. In 100 dimensions, there is more space for an increasing number of sources, so the performance values (upper right panel) do not comes close to the baseline shown in the lower right panel. However, again we see that IBICA with γ is superior to IBICA with κ.

It is interesting to note, that by construction, very large dimensions are no problem for IBICA, because the algorithm considers only the distance matrix of the data which does not change if we embed the data into a higher-dimensional space. Therefore, running similar experiments in 1000 dimensions will lead to a similar performance as shown in the upper right panel.

4.3.5 Performance as a function of the number of nearest neighbors

All experiments presented so far set the number of nearest neighbors k to be 5 percent of the size of the largest possible partition which is 1000. So, k was always fixed to 50. Figure 4.11 plots the performance measure as a function of k for two situations: (i) in the left panel for 16 sources mixed in four dimensions with 1000 data points, (ii) in the right panel for 20 sources mixed in 10 dimensions also with 1000 data points. As a baseline we recorded simultaneously to running IBICA with γ and κ the performance measure for random solutions. Both panels show that the performance of IBICA depends on the choice of k. However, IBICA with γ depends less on k than IBICA with κ. We conclude that it would be promising to optimize k in the former simulations instead of fixing it to be 50. However, in order to cancel out the effect of k we have fixed it to one value.

4.3.6 Toy problems with images

In order to visually judge the performance of IBICA we mixed three images of size 640 by 480 pixels. In a first experiment we created three mixed images with a square

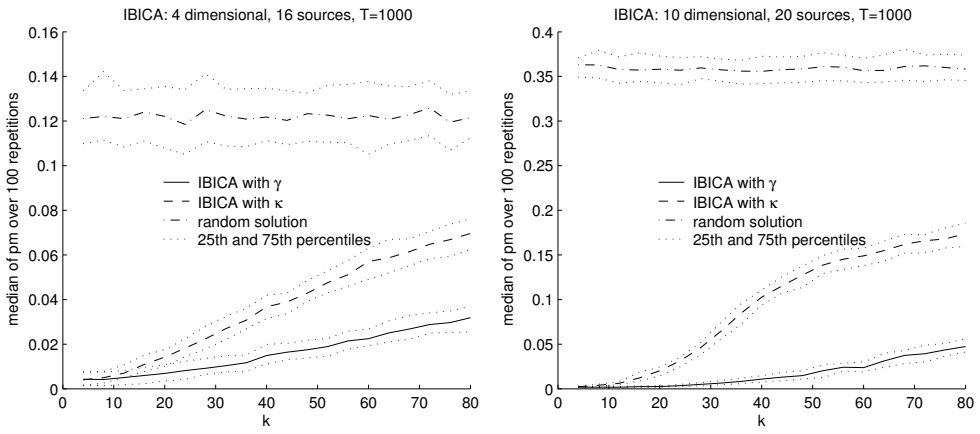


Figure 4.11: How does IBICA depend on the choice of k?

mixing matrix (see left column of Figure 4.12). To test the robustness of IBICA (with γ) we replaced 200 pixels by outliers (Gaussian noise with 10 times greater standard deviation than the original points). These outliers can be seen as the black pixels in the images in the left column.

Because the signals of the source images are neither sparse nor super-Gaussian, we initially sparsify the mixed data by a Haar wavelet transformation. The ICA problem is solved for the wavelet coefficients and the inverse wavelet transformation yields the sources. The second column of Figure 4.12 shows that IBICA is able to recover the correct source images. JADE and Fast ICA, right columns, are not able to unmix the images due to the outliers.

The sparsification step of the previous experiment allows also to separate overcomplete mixtures of images (three images in two mixtures). The left column of Figure 4.13 shows the mixed images and a scatterplot of the coefficients obtained by a Haar wavelet transformation. Applying IBICA (with γ) to these coefficients yields the 2×3-dimensional mixing matrix.

Given the mixing matrix, the reconstruction of the sources from the mixed signals is not trivial in the overcomplete setting as discussed in the introduction. The wavelet coefficients, shown as a scatterplot in the lower left panel of Figure 4.13 are assigned to the different ICA directions, i.e. to the columns of the mixing matrix using the method described in [14]. The reconstructed sources are shown in the middle column of Figure 4.13. While the images are well separated, the reconstruction loss is visible as artifacts due to miss-assigned wavelet coefficients.

4.4 Summary

Robustness is an important feature for algorithms intended for the real world. We showed how to employ simple outlier indices to obtain the inliers of the given data.

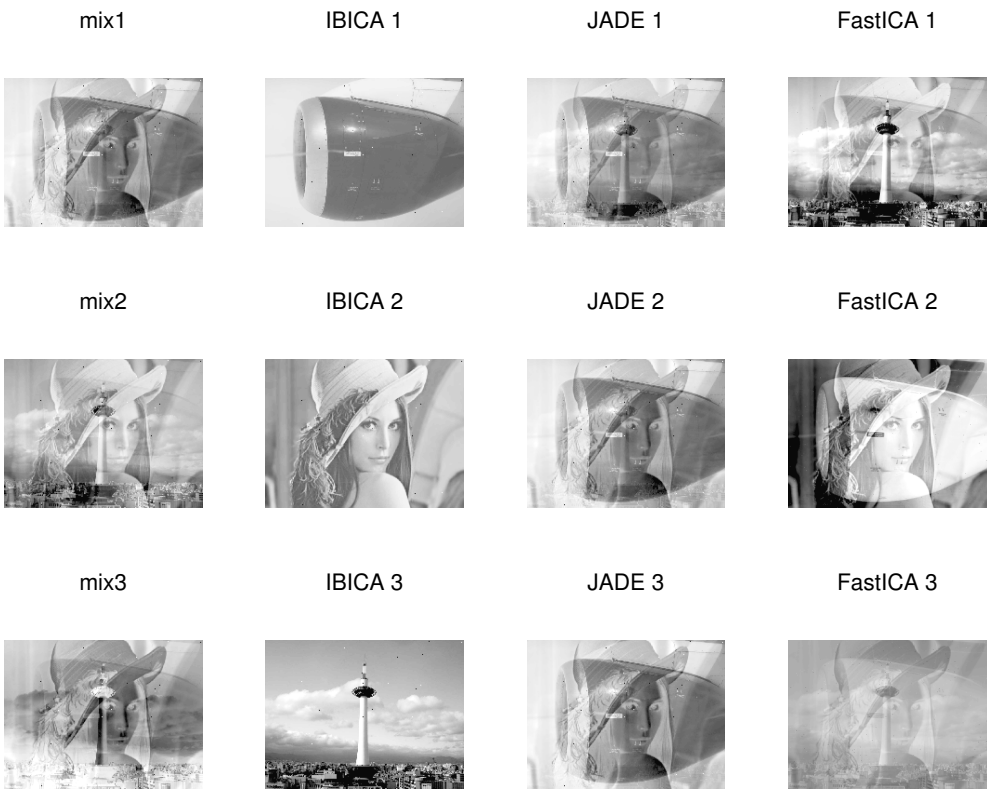


Figure 4.12: IBICA is able to separate mixed images (first column) that have been contaminated with outliers (black pixels in the images of the first columns) after sparsifying the data with a wavelet transformation as the images in the middle column show. JADE and FastICA fail (third and fourth column).

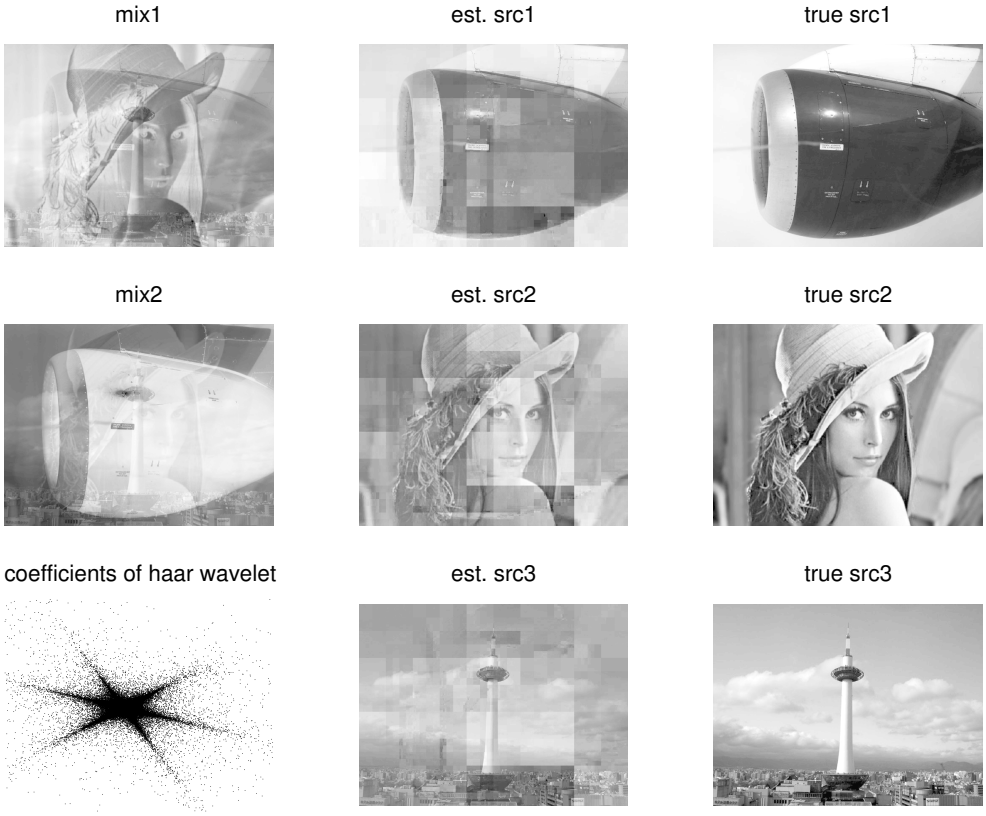


Figure 4.13: Three images have been linearly mixed into two images. Note that the signals of the source images are not super-Gaussian. After sparsifying the mixed signals (lower left scatterplot), IBICA is able to approximately recover the original images (middle versus right column). The block artifacts are due to wavelet coefficients that have been assigned to the wrong source.

These inliers directly characterize the columns of the mixing matrix. Hereby, we obtain a particularly robust approach to ICA for super-Gaussian sources that also solves the overcomplete ICA problem as we have seen in the experiments.

Note that some of the presented ideas appeared earlier in other geometrical algorithms (for example [14, 89]). The main difference of IBICA to these other methods is its usage of outlier indices which makes it particularly robust and which allows its use even in high dimensions because the used outlier indices are based solely on distances.

Furthermore, IBICA does not minimize any objective function. Instead, it uniquely defines the columns of the mixing matrix in terms of indices derived from the outlier indices. This has the advantage that by construction IBICA can not get stuck in local minima.

5 Nonlinear ICA with kernel methods

In the nonlinear ICA model, $x[t] = f(s[t])$, the mixing matrix of the linear ICA model, $x[t] = As[t]$, is replaced by a general nonlinear invertible function $f : \Re^n \to \Re^n$. For example, the components x_1 and x_2 of the mixture

$$\begin{aligned} x_1[t] &= -(s_2[t] + 1)\cos(\pi s_1[t]) \\ x_2[t] &= (s_2[t] + 1)\sin(\pi s_1[t]), \end{aligned} \tag{5.1}$$

are nonlinear mixtures of the sources s_1 and s_2. Figure 5.1 shows a scatterplot of x_1 and x_2 (right panel) next to a scatterplot of linearly mixed data (left panel). In the linear mixture the data is scattered along straight lines which are the ICA directions. In the nonlinear mixing the data is scattered along curves. Kernel-based learning maps the data shown in the scatterplots from the input space to some higher-dimensional feature space, such that the curves become straight lines. This idea is illustrated in Figure 5.2. Thus we transformed the nonlinear problem to a linear one which can be solved in principle with linear methods.

Kernel-based learning has become a popular technique recently (for example [98, 28, 92, 17, 93, 74]). The idea of kernelizing [93] allows to construct powerful nonlinear variants of existing linear algorithms—based on the scalar product—by mapping the data $x[1], \ldots, x[T] \in \Re^n$ implicitly into some kernel feature space $\mathcal{F}$ through some mapping $\Phi : \Re^n \to \mathcal{F}$. Performing a linear algorithm in $\mathcal{F}$ corresponds to a nonlinear algorithm in input space: in other words, linear BSS in $\mathcal{F}$ would give rise to a nonlinear BSS algorithm in input space. This can be done efficiently and never directly

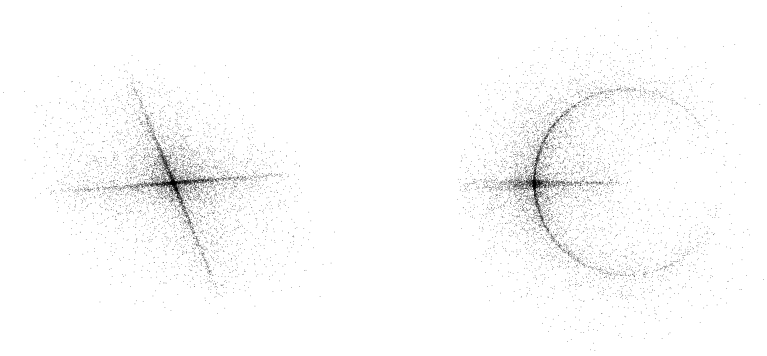


Figure 5.1: In a linear mixture (left panel), the signals are scattered along straight lines, which represent the ICA directions. In a nonlinear mixture (right panel), the signals are scattered along curves.

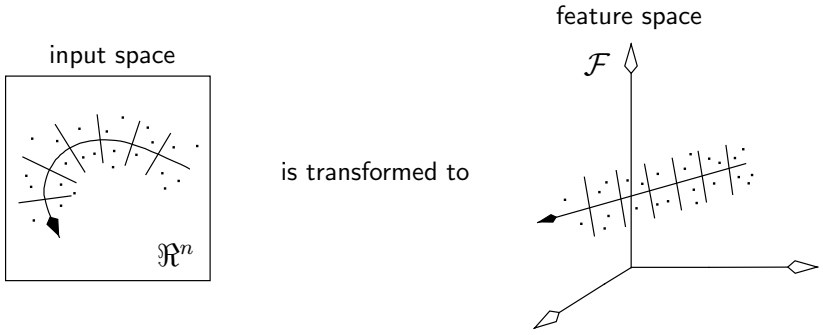


Figure 5.2: Kernel methods transform curves in input space (left) to straight lines in a higher-dimensional feature space (right).

but only implicitly in $\mathcal{F}$ by using the kernel trick $k(a, b) = \Phi(a) \cdot \Phi(b)$. However, a straight forward application of the kernel trick to ICA has so far failed for two reasons: (i) applying a linear ICA algorithm in feature space will not necessarily identify the sought-after signals, since there are very likely directions that are also independent but higher-order versions of the original signals, and (ii), in principle, the ICA algorithm has to be applied, after kernelizing, to a T-dimensional[1] problem which is numerically neither stable nor tractable.

We solve these problems by applying a dimension reduction step in feature space before using ICA. This is possible because typically the data forms a lower-dimensional subspace in $\mathcal{F}$, even much lower than T-dimensional. We therefore propose a mathematical construction—very much inspired by reduced set methods [91]—that allows us to adapt to the intrinsic data dimension. In the next step an orthonormal basis of this low-dimensional submanifold is constructed which eventually makes the computations of a subsequent ICA algorithm tractable. The subtle difference to reduced set techniques is that we do not aim at constructing a low-dimensional basis for a good classification, rather we aim for an efficient, i.e. low-dimensional description of the data in $\mathcal{F}$.

As pointed out in [52], in general there are no unique solutions to the nonlinear ICA problem. However, employing kernel feature spaces and dimension reduction we are able to restrict the space of possible functions used to invert the nonlinearity. Furthermore, exploiting the time structure of the unknown sources allows us to recover the sources s. Figure 5.3 gives an overview of the proposed method.

Chapter outline

In Section 5.1 we explain how to reduce the dimension of the kernel feature space that contains the mapped input space data. In Section 5.2 we apply a linear ICA algorithm based on time structure to the signals in the reduced space. This results in several

[1] Note that even though $\mathcal{F}$ might be infinite-dimensional the subspace of $\mathcal{F}$ where the data lies is at most T-dimensional.

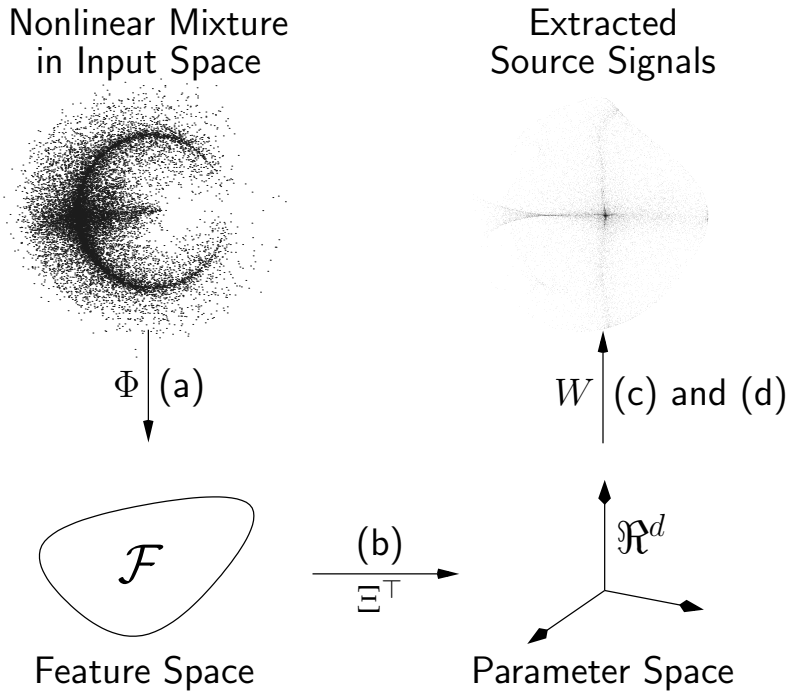


Figure 5.3: The nonlinear ICA problem is solved in four steps: (a) The data is mapped from input space to feature space, (b) the dimensionality is reduced, (c) second order temporal decorrelation ICA and (d) an automatic selection procedure is applied.

nonlinearly transformed signals (usually more than input dimensions) from which we select automatically the sources, see Section 5.3. In Section 5.4 we validate our method on nonlinearly mixtures of artificially generated signals and various sound signals.

5.1 Constructing kernel feature spaces of reduced dimension

In order to establish a linear problem in feature space that corresponds to some nonlinear problem in input space we need to specify how to map inputs $x[1], \ldots, x[T] \in \Re^n$ into feature space $\mathcal{F}$ and how to handle its possibly high-dimensionality. Note that $x[t]$ is scaled down such that its absolute maximum is one. Hereby we force the signals between -1 and 1 before mapping to $\mathcal{F}$. This will become important for the selection procedure (see Section 5.2).

In the following, we describe two methods that obtain an orthogonal basis in feature space with reduced dimension and we explain how to project the data onto this finite-dimensional basis such that ICA techniques can be applied.

There exist a variety of other dimensionality reduction methods for the kernel setting: In [94] the kernel matrix is approximated by iteratively picking columns of that matrix in a greedy manner. [31] uses the Sherman-Morrison-Woodbury method and the product-form Cholesky factorization to obtain low-rank kernel representations. [104] employs the Nyström method to a randomly sampled subset of the data which is very similar to the first of our proposed methods (see Section 5.1.1). However, we perform the resampling several times and use additionally the condition numbers of the corresponding kernel matrices to pick a particular subset.

5.1.1 Finding a basis via random sampling/clustering

In addition to the input points, consider some further points $v_1, \ldots, v_d \in \Re^n$ from the same space, that will later generate a basis in $\mathcal{F}$. Let us denote the mapped points[2] by $\Phi_x := [\Phi(x[1]) \cdots \Phi(x[T])]$ and $\Phi_v := [\Phi(v_1) \cdots \Phi(v_d)]$. We assume that the columns of Φ_v constitute a basis of the column space[3] of Φ_x, formally expressed as

$$\operatorname{span}(\Phi_v) = \operatorname{span}(\Phi_x) \;\text{ and }\; \operatorname{rank}(\Phi_v) = d. \tag{5.2}$$

Below, we will explain how and to what degree this assumption can be fulfilled. Moreover, Φ_v being a basis implies that the matrix[4] $\Phi_v^\top \Phi_v$ has full rank and its inverse exists. So, now we can define an orthonormal basis (called empirical kernel map in [91])

$$\Xi := \Phi_v (\Phi_v^\top \Phi_v)^{-1/2} \tag{5.3}$$

[2] We denote the points of the time series with square brackets, for example $x[t]$, and other points of the input space with subscripts, for example v_d.

[3] The column space of Φ_x is the space that is spanned by the column vectors of Φ_x, written $\operatorname{span}(\Phi_x)$.

[4] The ij-th entry of the matrix $\Phi_v^\top \Phi_v$ is $\Phi(v_i)^\top \Phi(v_j)$.

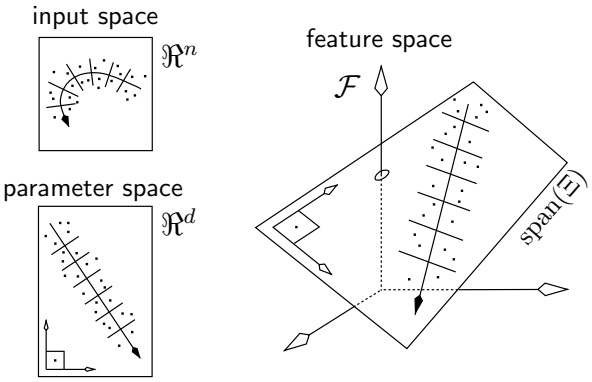


Figure 5.4: Input data are mapped to some submanifold of $\mathcal{F}$ which is in the span of a d-dimensional orthonormal basis Ξ. Therefore these mapped points can be parameterized in $\Re^d$. The linear directions in parameter space correspond to nonlinear directions in input space.

the column space of which is identical to the column space of Φ_v. Consequently, this basis Ξ enables us to parameterize all vectors that lie in the column space of Φ_x by some vectors in $\Re^d$. For instance for vectors $\sum_{t=1}^{T} \alpha_{\Phi i} \Phi(x[t])$, which we write more compactly as $\Phi_x \alpha_\Phi$, and $\Phi_x \beta_\Phi$ in the column space of Φ_x with α_Φ and β_Φ in $\Re^T$ there exist α_Ξ and β_Ξ in $\Re^d$ such that $\Phi_x \alpha_\Phi = \Xi \alpha_\Xi$ and $\Phi_x \beta_\Phi = \Xi \beta_\Xi$. The orthonormality implies

$$\alpha_\Phi^\top \Phi_x^\top \Phi_x \beta_\Phi = \alpha_\Xi^\top \Xi^\top \Xi \beta_\Xi = \alpha_\Xi^\top \beta_\Xi \tag{5.4}$$

which states the remarkable property that the dot product of two linear combinations of the columns of Φ_x in $\mathcal{F}$ coincides with the dot product in $\Re^d$. By construction of Ξ (see Equation (5.3)) the column space of Φ_x is naturally isomorphic (as a vector space) to the $\Re^d$. Moreover, this isomorphism is compatible with the two involved dot products as was shown in Equation (5.4). This implies that all properties regarding angles and lengths can be taken back and fourth between the column space of Φ_x and $\Re^d$. The space that is spanned by Ξ is called parameter space. Figure 5.4 pictures our intuition: usually kernel methods parameterize the column space of Φ_x in terms of the mapped patterns $\{\Phi(x[i])\}$ which effectively corresponds to vectors in $\Re^T$. The orthonormal basis from Equation (5.3), however enables us to work in $\Re^d$ i.e. in the span of Ξ.

Projecting the input data onto an orthonormal basis

By employing the kernel trick we can directly map the input data onto the subspace of feature space that is spanned by the orthonormal basis. The expressions

$$(\Phi_v^\top \Phi_v)_{ij} = \Phi(v_i)^\top \Phi(v_j) = k(v_i, v_j) \quad \text{with} \quad i, j = 1 \ldots d \tag{5.5}$$

are the entries of a real valued $d \times d$ matrix $\Phi_v^\top \Phi_v$ that can be effectively calculated using the kernel trick. By construction of $v_1, \ldots, v_d$, it has full rank and is thus invertible. Similarly we get

$$(\Phi_v^\top \Phi_x)_{ij} = \Phi(v_i)^\top \Phi(x[j]) \; = k(v_i, x[j]) \quad \text{with} \quad i = 1 \ldots d, \quad j = 1 \ldots T, \tag{5.6}$$

which are the entries of the real valued $d \times T$ matrix $\Phi_v^\top \Phi_x$. Using both matrices we compute finally

$$\begin{aligned} \Psi_x[t] \;&:=\; \Xi^\top \Phi(x[t]) = (\Phi_v^\top \Phi_v)^{-1/2} \Phi_v^\top \Phi(x[t]) \\ &= \begin{bmatrix} k(v_1, v_1) & \cdots & k(v_1, v_d) \\ \vdots & & \vdots \\ k(v_d, v_1) & \cdots & k(v_d, v_d) \end{bmatrix}^{-\frac{1}{2}} \begin{bmatrix} k(v_1, x[t]) \\ \vdots \\ k(v_d, x[t]) \end{bmatrix} \end{aligned} \tag{5.7}$$

which is a real valued $d \times 1$ vector representing a projected data point. Note that $(\Phi_v^\top \Phi_v)^{-1/2}$ can be omitted if the subsequent ICA procedure contains a whitening step.

Regarding the computational costs of this projection, we have to evaluate the kernel function $\mathrm{O}(d^2 n) + \mathrm{O}(dTn)$ times and Equation (5.7) requires $\mathrm{O}(d^3)$ multiplications where n denotes the dimension of the input space. Again, note that d is much smaller than T. Furthermore, after projection the storage requirements are reduced since we do not have to hold the full $T \times T$ kernel matrix but only a $d \times T$ matrix.

Choosing vectors for the basis in $\mathcal{F}$

So far, we have assumed to be given some points $v_1, \ldots, v_d$ that fulfill Equation (5.2) and we presented the beneficial properties of our construction. In fact, the vectors $v_1, \ldots, v_d$ are roughly analogous to a reduced set in the support vector world [91]. Note however that often we can only approximately fulfill Equation (5.2), i.e.

$$\mathrm{span}(\Phi_v) \approx \mathrm{span}(\Phi_x), \tag{5.8}$$

for example for an RBF kernel $\mathrm{span}(\Phi_x)$ is T-dimensional, but $\mathrm{span}(\Phi_v)$ is by definition d-dimensional (see Appendix A.7; for further discussion [103, 8]). Several options exist to achieve this approximation.

The points $v_1, \ldots, v_d$ have to be chosen, such that in Equation (5.7) the inversion of the kernel matrix K_v, whose entries are

$$(K_v)_{ij} := (\Phi_v^\top \Phi_v)_{ij} = k(v_i, v_j), \tag{5.9}$$

is numerically stable. We try to find a set of points such that the condition number[5] of its corresponding kernel matrix is below a certain threshold and if we add one more point the condition number is above. Since we can not check all possible combinations, we randomly sample d points (for fixed d) repeatedly, say r, for example, 100

[5] The condition number of a matrix is the ratio between the largest and the smallest singular value.

times. Roughly speaking, we perform this sampling for different d until we find d points, $v_1, \ldots, v_d$, that are linearly independent in feature space (more precisely: the corresponding condition number is below the threshold), but we can not find $d+1$ points with the same property (i.e. the condition number for those points is above the threshold). This is done by computing the kernel matrix in $\mathrm{O}(d^2 n)$ time, yielding an overall cost of $\mathrm{O}(dr)[\mathrm{O}(d) + \mathrm{O}(d^2 n)] = \mathrm{O}(d^3 rn)$ where n denotes the dimension of the input space.

This procedure determines d and points $v_1, \ldots, v_d$. Running k-means clustering (with $k = d$), costing $\mathrm{O}(Tdn)$, in input space is another way to pick such points. Our experience shows that both approaches work well as long as d is chosen large enough.

5.1.2 Finding a basis via kernel PCA

Another more direct method to obtain the low-dimensional subspace is Kernel PCA [93]. Such a subspace is optimal with respect to the reconstruction error in feature space, however computational costs are slightly increased (see Figure 5.15 in Section 5.4.5). For simplicity, we assume that the data is centered in feature space[6]. To perform kernel PCA, we need to find eigenvectors and eigenvalues of the covariance matrix $\frac{1}{T}\Phi_X \Phi_X^\top$. Denoting the diagonal matrix with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_T$ along the diagonal as Λ, the eigenvectors $E = [e_1 \cdots e_T]$ of the covariance matrix $\frac{1}{T}\Phi_X \Phi_X^\top$ fulfill

$$(\frac{1}{T}\Phi_X^\top \Phi_X)E = E\Lambda \tag{5.10}$$

which immediately implies

$$(\frac{1}{T}\Phi_X \Phi_X^\top)(\Phi_X E) = (\Phi_X E)\Lambda. \tag{5.11}$$

So, $\lambda_1, \ldots, \lambda_T$ are the eigenvalues of $\frac{1}{T}\Phi_X \Phi_X^\top$ with corresponding eigenvectors $\Phi_X E$. Normalizing the first d eigenvectors yields a d-dimensional orthonormal basis,

$$\Xi := \Phi_X E_d (T\Lambda_d)^{-1/2} \tag{5.12}$$

with $E_d := [e_1 \cdots e_d]$, Λ_d being the diagonal matrix with $\lambda_1, \ldots, \lambda_d$ along the diagonal and $(T\Lambda_d)^{-1/2}$ ensuring orthonormality. This basis enables us to parameterize the signals $\Phi(x[t])$ in feature space as real valued d-dimensional signals

$$\begin{aligned} \Psi_x[t] &:= \Xi^\top \Phi(x[t]) = (T\Lambda_d)^{-1/2} E_d^\top \Phi_X^\top \Phi(x[t]), \\ &= \frac{1}{\sqrt{T}} \begin{bmatrix} \frac{1}{\sqrt{\lambda_1}} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{\sqrt{\lambda_d}} \end{bmatrix} \begin{bmatrix} e_1^\top \\ \vdots \\ e_d^\top \end{bmatrix} \begin{bmatrix} k(x[1], x[t]) \\ \vdots \\ k(x[T], x[t]) \end{bmatrix} \end{aligned} \tag{5.13}$$

that are calculated conveniently using the kernel trick.

[6] The kernel matrix K with entries $k(x[i], x[j])$ can be easily centered (see also [93]) by $K \mapsto K - 1_T K - K 1_T + 1_T K 1_T$ with 1_T being the $T \times T$ matrix with all entries equal to $1/T$.

Since kernel PCA involves solving the eigenvalue problem for a large matrix, whose size depends on the amount of data, $O(T^3)$, we typically apply kernel PCA to a subset of the original data set if T becomes large (similar to [72, 91]).

5.2 Nonlinear ICA with time structure

Clearly, ICA algorithms can not be applied directly in full feature space without the proposed reduction step: they would need to solve a T-dimensional ICA problem which is intractable. A further problem is that manipulating such a $T \times T$ matrix can easily become numerically unstable and even overfitting might occur [53]. In the previous section we have mapped the signals $x[t]$ from input space onto signals $\Psi_x[t]$ in a d ($\ll T$)-dimensional parameter space (see Figure 5.3). This was done either by random sampling or k-means clustering using Equation (5.7) or by applying kernel PCA together with Equation (5.13). Now we are in a situation in which the nonlinear problem in input space has been transformed to a linear problem in parameter space where we can apply linear ICA methods. In particular, we propose to use TDSEP, a second order ICA technique that relies on time-shifted covariance matrices of the mapped signals $\Psi_x[t]$, hereby exploiting the assumed time structure of the unknown sources (in the appendix of [41] we discuss why not to use kurtosis-based techniques).

We briefly describe the TDSEP algorithm (details can be found in [112], see also[12]). For the signals in parameter space

$$\Psi_x[t] := \Xi^\top \Phi(x[t]) \in \Re^d \tag{5.14}$$

we define symmetrized time-shifted covariance matrices:

$$R_\tau := \frac{1}{2(T-\tau)} \sum_{t=1}^{T-\tau} ((\Psi_x[t] - \mu_\Psi)(\Psi_x[t+\tau] - \mu_\Psi)^\top + (\Psi_x[t+\tau] - \mu_\Psi)(\Psi_x[t] - \mu_\Psi)^\top), \tag{5.15}$$

with $\mu_\Psi := \frac{1}{T} \sum_{t=1}^{T} \Psi_x[t]$. Then we find a matrix W that simultaneously diagonalizes[7] several of these matrices R_{τ_i}, i.e. the matrices

$$W R_{\tau_i} W^\top \quad i = 1, \ldots, m \tag{5.16}$$

should become approximately diagonal. W is the sought-after demixing matrix. The extracted d nonlinear components are

$$y[t] := W \Psi_x[t] \in \Re^d. \tag{5.17}$$

5.3 Selecting from the extracted components

Among the extracted components $y[t]$ are besides the sought-after sources, also signals that we are not interested in. Empirically, these other signals can be explained by

[7] We use the algorithm described in [25], see also [20].

higher-order monomials of the sources as we will see next (with ideas from [39]). These monomials are well-motivated for polynomial kernels but are also useful to analyze signals that have been extracted using a Gaussian kernel.

5.3.1 Reconstructing the extracted components

For two source signals s_1 and s_2, we call the monomials of these sources up to a certain degree "quasi sources". For example, the quasi sources up to degree 2, i.e. where each variable appears up to degree 2, are

$$q_2 := (s_1^2 s_2^2,\ s_1^2 s_2,\ s_1^2,\ s_1 s_2^2,\ s_1 s_2,\ s_1,\ s_2^2,\ s_2)^\top. \tag{5.18}$$

Note that for brevity, we write s_1 instead of $s_1[t]$, i.e. s_1 is a signal. In general, the quasi sources up to degree m are all monomials of the form $s_1^{m_1} s_2^{m_2}$ for $0 \leq m_1, m_2 < m$. Accordingly, q_m is the vector containing all those monomials.

Most quasi sources are pairwise correlated: for two independent signals s_1 and s_2 the correlation between arbitrary monomials in s_1 and s_2 is

$$\begin{aligned} \operatorname{corr}(s_1^{k_1} s_2^{m_1}, s_1^{k_2} s_2^{m_2}) &= \frac{\operatorname{cov}(s_1^{k_1} s_2^{m_1}, s_1^{k_2} s_2^{m_2})}{\prod_{i=1,2} \sqrt{\operatorname{var}(s_1^{k_i} s_2^{m_i})}} \\ &= \frac{\mathrm{E}\{s_1^{k_1+k_2}\}\,\mathrm{E}\{s_2^{m_1+m_2}\} - \mathrm{E}\{s_1^{k_1}\}\,\mathrm{E}\{s_1^{k_2}\}\,\mathrm{E}\{s_2^{m_1}\}\,\mathrm{E}\{s_2^{m_2}\}}{\prod_{i=1,2} \sqrt{\mathrm{E}\{s_1^{2k_i}\}\,\mathrm{E}\{s_2^{2m_i}\} - (\mathrm{E}\{s_1^{k_i}\}\,\mathrm{E}\{s_2^{m_i}\})^2}}. \end{aligned} \tag{5.19}$$

Since for symmetrically distributed signals s_1 and s_2 (with zero mean and variance one) the odd moments are zero,

$$\mathrm{E}\{s_1^k\} = 0 \quad \text{if } k \text{ is odd}, \tag{5.20}$$

two quasi sources $s_1^{k_1} s_2^{m_1}$ and $s_1^{k_2} s_2^{m_2}$ are uncorrelated in most cases:

$$\operatorname{corr}(s_1^{k_1} s_2^{m_1}, s_1^{k_2} s_2^{m_2}) = 0 \quad \text{if } k_1 + k_2 \text{ is odd or } m_1 + m_2 \text{ is odd}. \tag{5.21}$$

This is easily implied from above equation using the fact, that if the sum of two integers is odd then one of the summands must be odd as well. Therefore the quasi sources for two signals can be divided into four groups with no correlations between the groups; for example, for the quasi sources up to degree 2 the four groups are (see Figure 5.5),

$$\{s_1^2 s_2^2, s_1^2, s_2^2\},\ \{s_1^2 s_2, s_2\},\ \{s_1 s_2^2, s_1\},\ \{s_1 s_2\}. \tag{5.22}$$

Now we will use these findings to reconstruct the extracted components for an easy example. Consider two sinusoidal source signals $s[t] = [s_1[t], s_2[t]]^\top$ that are nonlinearly mixed by

$$x[t] = A(s_1[t], s_2[t])^\top + c s_1[t] s_2[t] \tag{5.23}$$

with

$$A = \begin{bmatrix} -1.2173 & -1.1283 \\ -0.0412 & -1.3493 \end{bmatrix} \text{ and } c = \begin{bmatrix} -0.2611 \\ 0.9535 \end{bmatrix} \tag{5.24}$$

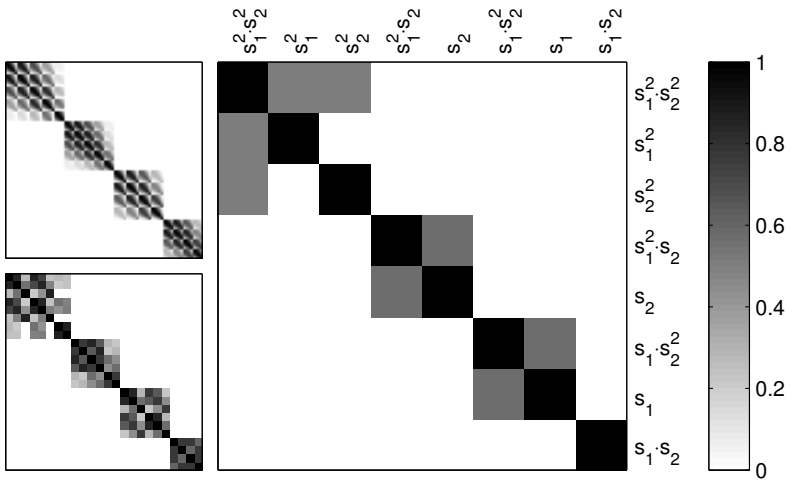


Figure 5.5: Most quasi sources are pairwise correlated; the right panel shows the covariance matrix of the quasi sources up to degree 2, the lower left panel up to degree degree 4 and the upper left panel up to degree 8. Note, that the quasi sources can always be divided into four groups.

(mixture taken from [73]). Running our algorithm with a polynomial kernel of degree 4,

$$k(a, b) = (a^\top b + 1)^4, \tag{5.25}$$

we have to consider the quasi sources up to degree 4: all possible products of s_1, s_1^2, s_1^3, s_1^4 and their counterparts in s_2. Using Equation (5.21) these quasi sources can also be arranged into four groups with no correlations between the groups. As examples, we explain four of the extracted signals using these quasi sources, i.e. y_7, y_4, y_1, y_9, shown in the left panels of Figure 5.6. The middle panels show the best matching quasi sources. Note, that the true sources, s_1 and s_2, have a very high correlation to their left neighbors, y_7 and y_4, respectively. The other extracted signals, y_1 and y_9, do not have a very high correlation to any of the quasi source signals: the best fits, $s_1^4 s_2$ and $s_1^4 s_2^4$, are plotted in the two lower middle panels. The extracted signals can better be explained with linear combinations of subsets of mutually correlated quasi sources. Therefore, we combined all quasi sources that are correlated with $s_1^4 s_2^4$ to reconstruct y_9. The result is shown in the lower right panel which reaches a good fit (corr = 0.960), similarly for y_1 and the other not shown extracted signals. Note, that for y_7 and y_4 that matched s_1 and s_2 already reasonably well, more quasi sources do not improve the result notably.

Empirically, we have seen that the extracted components can be explained by linear combinations of higher-order monomials of the sources. Using this knowledge several options to select the signals of interest suggest themselves: the signals built by higher-order monomials are very peaky and therefore have after proper normalization a lower variance and also a lower description length [90] than the signals of interest. However,

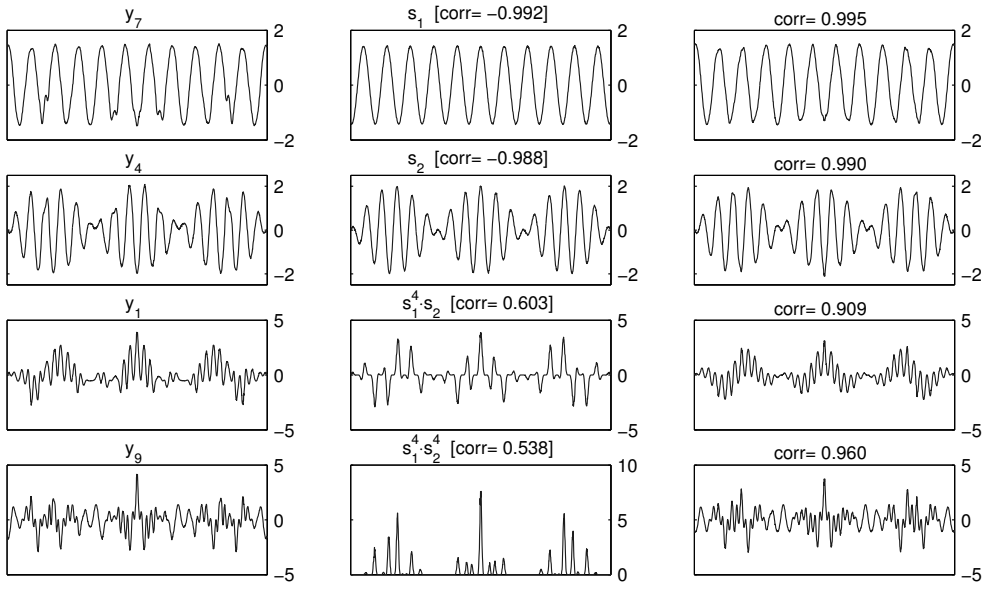


Figure 5.6: The extracted signals in the left panels (only four shown) are tried to be matched with single quasi sources in the middle panels and combinations of subgroups of quasi sources (right panels).

whether methods based on these intuitions work in practice depends very much on the considered signals. The goal is to identify the sought-after signals s_1 and s_2 among the other extracted signals. If, for example, s_1 has a much higher description length than s_2, there might be a problem: probably s_1^2 will have a large description length as well, and therefore it is more likely that a method based on these principles will prefer s_1^2, instead of s_2. This explains why these selection procedures can fail easily, which is in accordance with our experience of running a number of experiments with different signals.

5.3.2 Selection by rerunning the algorithm

An algorithmic trick that worked well in all our experiments is to apply the algorithm twice. This trick is motivated by work on assessing reliability [70, 71, 75]. Intuitively, the idea is to look for the most reliable components among the extracted signals, i.e. components that appear again after re-iteration of the algorithm. For this we repeat the algorithm with the same parameters (kernel choice, d, τ, ...), but instead of sending $x[t]$ into the feature space we start with the d-dimensional demixed results $y[t]$, map them to the feature space, reduce the dimensionality[8] and demix with TDSEP, which yields $y'[t]$. The sought-after components of $y[t]$ are the ones that are matched

[8]The kernel function can be used with signals of arbitrary dimension.

best by the components $y'[t]$ of the second run of TDSEP in feature space.

Why does this selection process find the right signals? As we have seen in the previous section most of the undesired signals are linear combinations of peaky higher-order monomials of the source signals, which can have very large values. Before the signals get mapped to feature space they are scaled, such that their absolute maximum is one (see first paragraph of Section 5.1). This is done by dividing each signal by its absolute maximum. The effect of this rescaling is that very large peaky signals are penalized, i.e. their variance is decreased more than the variance of signals that are less peaky. By doing so, we bias the desired signals to appear again with high correlations after another nonlinear demixing. This method works very well. All experiments documented in this article successfully used this selection method. Our experience shows that this selection fails only in cases where the sources are not recovered at all, i.e. where the demixing failed, which means that actually there is nothing to select from.

5.4 Experiments

Nonlinearities appear in different contexts; for example, amplifier saturation results in difficult nonlinearities. Also sensors can have nonlinearities which have a disadvantageous influence on the recorded signals. However, real-world signals have the drawback that the ground truth—i.e. the true source signals—is not known. Since we want to demonstrate the performance of our algorithm, we consider in this chapter well-defined, controlled situations where we can compare the results of the algorithm to the true sources.

5.4.1 Deterministic artificial data

In the first experiment we generate 2,000 data points from two sinusoidal signals $s[t] = [s_1[t], s_2[t]]^\top$ that have different frequencies ($s_1[t] = \sin(0.05\pi t)$, $s_2[t] = \sin(0.021\pi t)$ with $t = 1, \ldots, 2000$). These source signals are nonlinearly mixed (see left panel in the first row of Figure 5.7) by

$$\begin{aligned} x_1[t] &= e^{s_1[t]} - e^{s_2[t]} \\ x_2[t] &= e^{-s_1[t]} + e^{-s_2[t]}. \end{aligned} \tag{5.26}$$

We use a polynomial kernel of degree 9,

$$k(a, b) = (a^\top b + 1)^9, \tag{5.27}$$

which induces a feature space of all monomials up to degree 9. Applying k-means clustering to 500 randomly chosen input vectors we determine vectors $v_1, \ldots, v_{20}$ in input space, shown as "+" in the left panel in the first row of Figure 5.7. Projecting onto the feature space images of these vectors reduces the dimension to 20. As the second step we apply TDSEP (with time-shifts $\tau = 0 \ldots 7$) to those 20-dimensional mapped signals $\Psi_x[t]$. We obtain 20 components among which we select two components as described

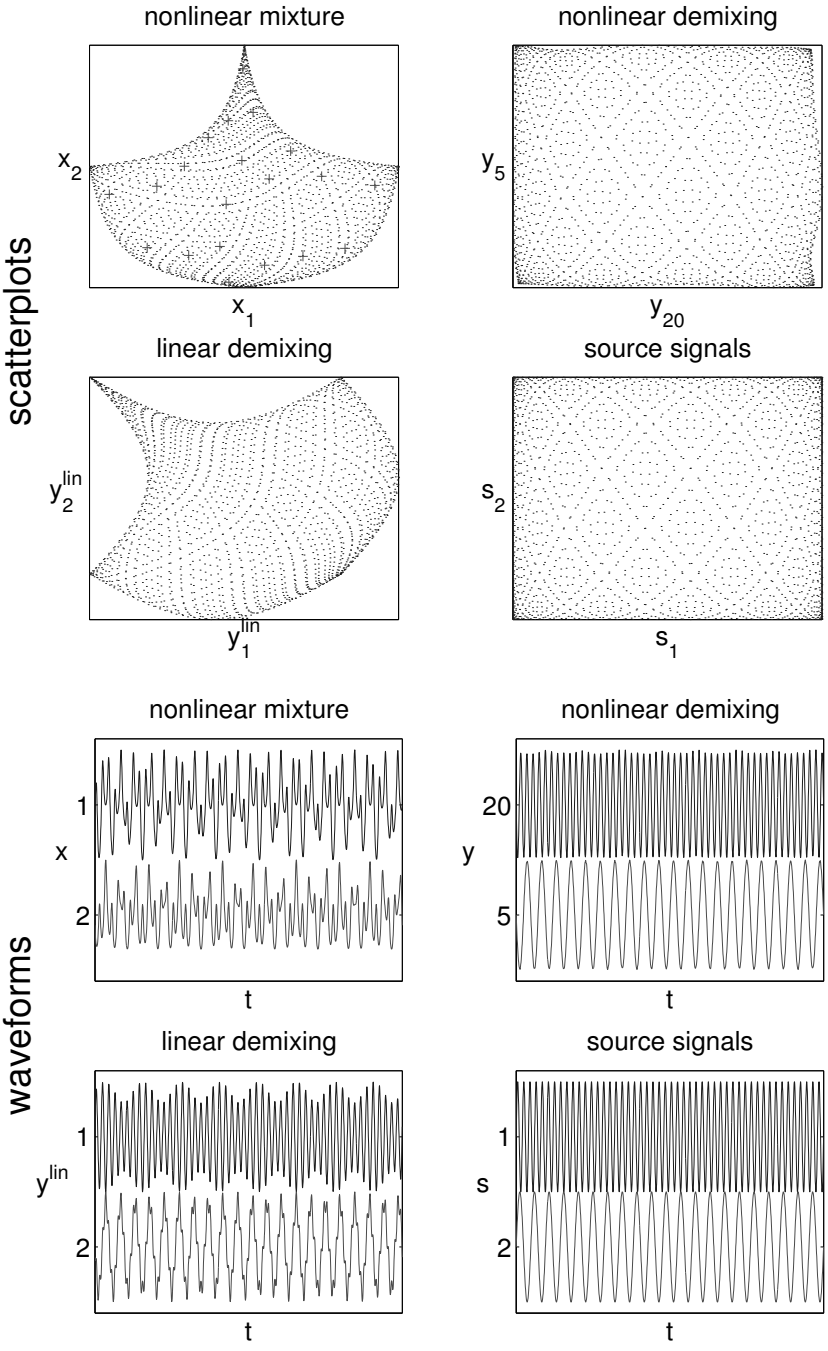


Figure 5.7: Deterministic artificial data: scatterplots and waveforms of the nonlinear mixture and the nonlinear demixing (first and third row) and of linear demixing and the true source signals (second and fourth row).

in Section 5.3.2. Their scatterplots and their waveforms are shown in the right panels in the first and third row of Figure 5.7. For comparison, we plot in the left panels of the second and fourth row the results of applying linear TDSEP directly to the nonlinearly mixed signals $x[t]$. In this simple example linear TDSEP reaches already a high correlation ($\mathrm{corr}(y_1^{lin}, s_1) = 0.9716, \mathrm{corr}(y_2^{lin}, s_2) = 0.9716$) to the true sources, but as we can see in the scatterplots shown in Figure 5.7, linear ICA fails to recover the right shape, in contrast to our nonlinear method which recovers the shape of the scatterplot almost perfectly ($\mathrm{corr}(y_{20}, s_1) = 0.9998, \mathrm{corr}(y_5, s_2) = 0.9999$). We study the same mixture with two sinusoidal signals that have almost the same frequencies ($s_1[t] = \sin(0.0045\pi t)$, $s_2[t] = \sin(0.005\pi t)$ with $t = 1, \ldots, 2000$). Figure 5.8 shows the results after running our algorithm with the same parameters as in the previous case. We see that even two signals that have almost the same frequencies are separated.

5.4.2 Speech data—bended

In another experiment we nonlinearly mix two speech signals $s[t] = [s_1[t], s_2[t]]^\top$ (each with 20,000 data points, sampling rate 8 kHz, each ranging between -1 and $+1$) by

$$\begin{aligned} x_1[t] &= -(s_2[t] + 1)\cos(\pi s_1[t]) \\ x_2[t] &= 1.5\,(s_2[t] + 1)\sin(\pi s_1[t]). \end{aligned} \tag{5.28}$$

We employ a Gaussian RBF kernel,

$$k(a, b) = e^{-\frac{\|a-b\|^2}{2\sigma^2}}, \tag{5.29}$$

which induces a feature space where each direction measures the similarity to one of the training points. We can set $\sigma^2 = \frac{1}{2}$ and use

$$k(a, b) = e^{-\|a-b\|^2}, \tag{5.30}$$

without loss of generality, if signals are scaled in an appropriate way. Projecting onto feature space images of the vectors $v_1, \ldots, v_{20} \in \Re^2$ (depicted as "+" in the left panel in the first row of Figure 5.9) that are determined by repeated random sampling, reduces the dimensionality to $d = 20$. Among the 20 signals that we obtain by TDSEP (with time-shifts $\tau = 0 \ldots 7$) we automatically choose with the selection method described in Section 5.3.2 two signals that turn out to reach very high correlations ($\mathrm{corr}(y_9, s_1) = 0.9768, \mathrm{corr}(y_4, s_2) = 0.9923$) with the original source signals. Since the linear method can only shear and rotate the data it fails to recover the two signals ($\mathrm{corr}(y_2^{lin}, s_1) = 0.8811, \mathrm{corr}(y_1^{lin}, s_2) = 0.4091$).

5.4.3 Speech data—twisted

For an even more difficult experiment we mix the two sound signals from the previous example by

$$\begin{aligned} x_1[t] &= (s_2[t] + 3s_1[t] + 6)\cos(1.5\pi s_1[t]) \\ x_2[t] &= (s_2[t] + 3s_1[t] + 6)\sin(1.5\pi s_1[t]), \end{aligned} \tag{5.31}$$

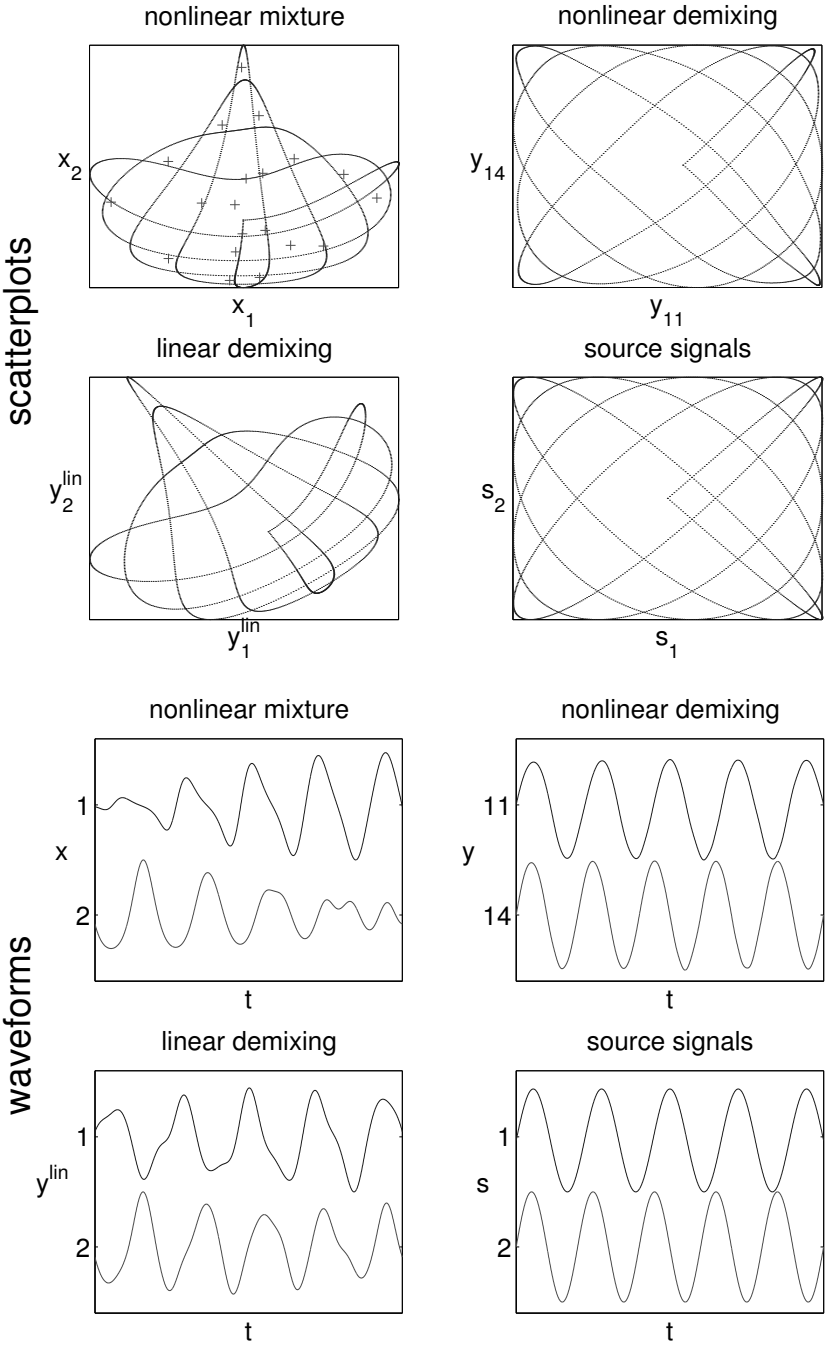


Figure 5.8: Deterministic artificial data with very close frequencies: Scatterplots and waveforms of the nonlinear mixture and the nonlinear demixing (first and third row) and of linear demixing and the true source signals (second and fourth row).

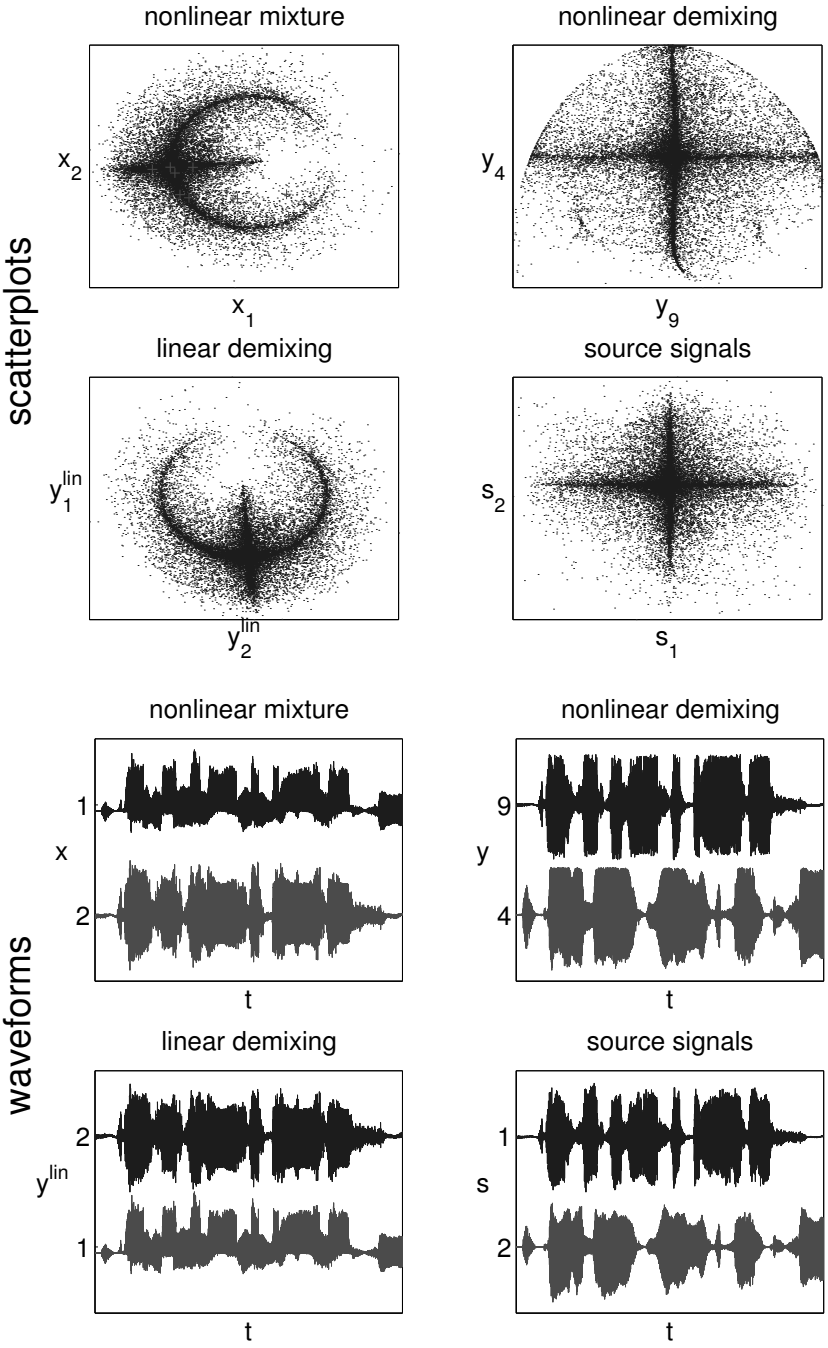


Figure 5.9: Speech data—bended: Scatterplots and waveforms of the nonlinear mixture and the nonlinear demixing (first and third row) and of linear demixing and the true source signals (second and fourth row).

which twists the sources. The first source is mapped along a spiral around the center. The second controls the deviation from that spiral. Note, that the second source contributes much less to the mixture than the first source. We map the data to a feature space induced by a Gaussian RBF kernel, $k(a,b) = e^{-\|a-b\|^2}$, and apply kernel PCA to 500 randomly chosen input vectors. We obtain a 25-dimensional subspace of feature space that approximates the high-dimensional manifold in feature space very well. Projecting the mixed signals into that space we obtain $\Psi_x[t]$ and finally applying TDSEP (with time-shifts $\tau = 0 \dots 7$) we recover 25 signals among which we select the signals of interest automatically (see Section 5.3.2). Again, these signals have very high correlations with the true sources ($\text{corr}(y_2, s_1) = 0.9900, \text{corr}(y_9, s_2) = 0.9466$), whereas the linear ones do not ($\text{corr}(y_2^{lin}, s_1) = 0.5703, \text{corr}(y_1^{lin}, s_2) = 0.0483$). Also their scatterplot and their waveforms represented in the right panels in the first and third row of Figure 5.10 show how well even the second source is found that is hidden as the amplitude of the spiral.

5.4.4 Analysis of the cross correlations through time

To analyze the found signals y more careful, we calculated the cross correlations with quasi sources for different time-lagged versions of the signals.

Figure 5.11 shows the cross correlations of the quasi sources up to degree 3 (see Section 5.3.1) and $y_2[t-\tau]$ and $s_1[t-\tau]$ for different τ (with $\tau = 0 \dots 7$). On the right panel, we see that s_1 is correlated to $s_1 s_2^2$, s_1^3, and $s_1^3 s_2^2$ as was already discussed in Section 5.3.1. Furthermore, there are also correlations with the time-shifted versions of those quasi-sources. Exactly the same holds for y_2 as we see in the left panel, i.e. y_2 recovers s_1 including its time structure. Corresponding results apply to y_9 and s_2 (see Figure 5.12).

To give some clue what the other signals that TDSEP extracted are, we analyzed y_1, $\dots$, y_{25} in a similar way. The upper panel of Figure 5.13 shows the cross correlations of these signals with the quasi sources. For example, we see that y_1 is strongly correlated with s_1^2 or that y_{16} is correlated with $s_1 s_2$, $s_1 s_2^3$, $s_1^3 s_2$ and $s_1^3 s_2^3$. Most signals have close connections to certain quasi sources. The lower panel of the same figure shows the corresponding cross correlations for time-shifted signals $y[t-\tau]$. Through time the correlations are less pronounced.

5.4.5 Kernel PCA versus random sampling versus clustering

In this section we compare the three proposed dimensionality reduction methods and discuss the trade-off in choosing the dimensionality of the subspace. We repeat the experiment of the previous section ("Speech Data—twisted") using different methods for dimensionality reduction (Kernel PCA versus Random Sampling versus Clustering) and for different subspace dimensionalities d. The results are shown in Figure 5.14: overall, it turns out that it does not matter too much for the separation result (measured here as the correlation to the true sources) which of the three reduction methods is used. Kernel PCA has slightly more difficulties to find the second source for small d. This might look surprising because Kernel PCA is optimal in finding a subspace

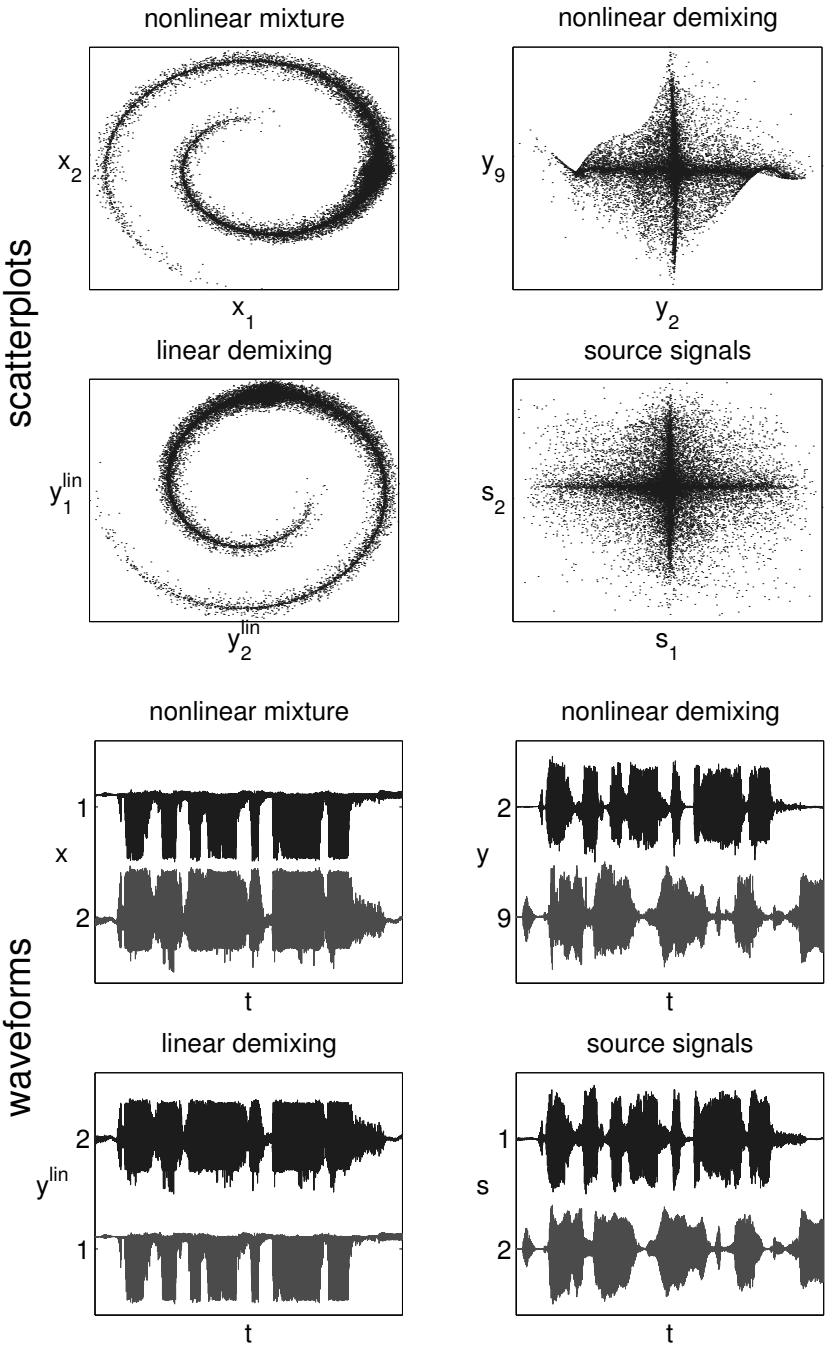


Figure 5.10: Speech data—twisted: Scatterplots and waveforms of the nonlinear mixture and the nonlinear demixing (first and third row) and of linear demixing and the true source signals (second and fourth row).

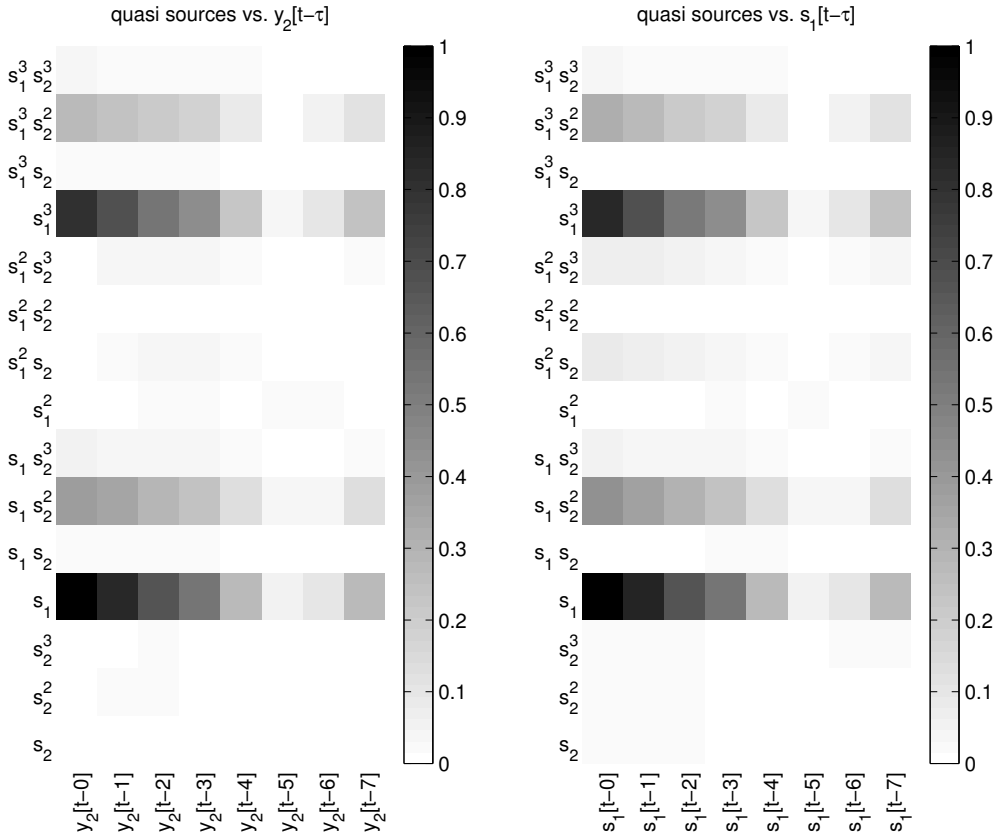


Figure 5.11: Cross correlations between (left) the quasi sources and $y_2[t-\tau]$ and between (right) the quasi sources and $s_1[t-\tau]$ for different τ.

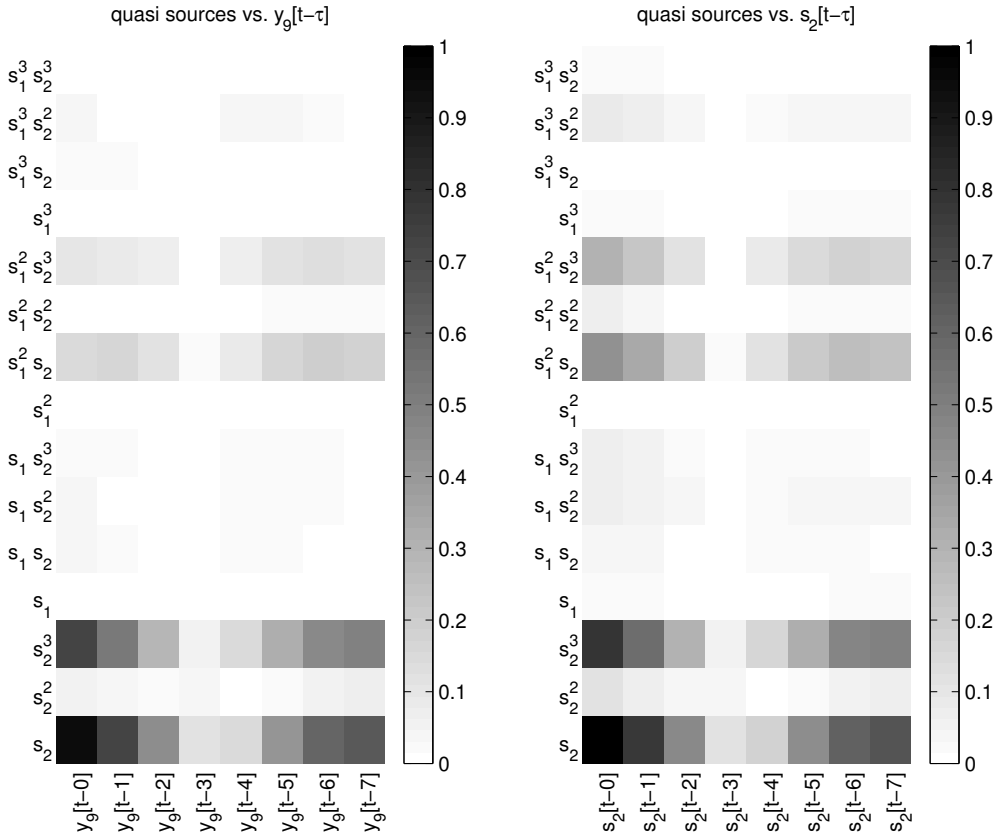


Figure 5.12: Cross correlations between (left) the quasi sources and $y_9[t-\tau]$ and between (right) the quasi sources and $s_2[t-\tau]$ for different τ.

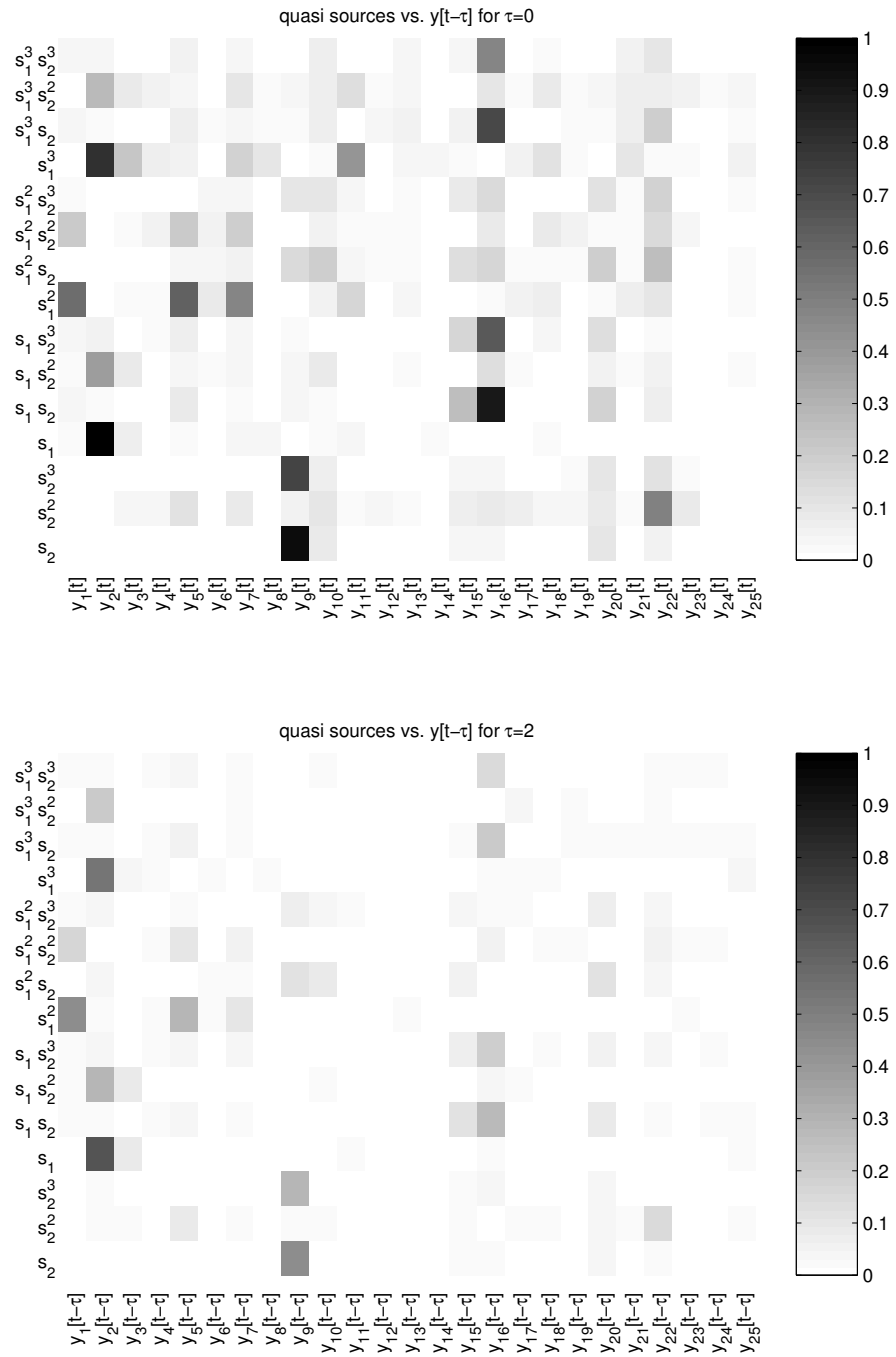


Figure 5.13: Cross correlations between the quasi sources and $y[t-\tau]$ for (top) $\tau = 0$ and (bottom) $\tau = 2$ for different τ.

of the feature space that contains most of the variance of the projected data, so one would expect better performance. The reason is, of course, that the PCA criterion does not necessarily optimize for good separation performance.

Furthermore, we see from the plots that increasing d generally improves the separation performance. But since the running time of TDSEP increases[9] for larger and larger d (see Figure 5.15), one should try to choose d such that the running time of TDSEP is still tolerable while the subspace is complex enough to demix.

The lower panel shows some interesting behavior: for small d the second signal can not be reconstructed very well. But increasing d to 10 and larger the subspace has enough complexity to unfold and thus to recover the second source.

The third column of Figure 5.16 validates this finding: shown are the scatterplots for the same experiment and we see that the biggest improvement happens between $d = 9$ and $d = 12$. The other columns show scatterplots for the two other experiments. Interestingly, the second mixture (second column) does not require a very large d. Already $d = 6$ is enough to recover the sources reasonably well.

5.4.6 Stochastic artificial data

For completeness, we test our method also for stochastic data with short correlation length: we generate 2,000 data points from two auto-regressive processes of order 3 and mix them using the twisting mixture from the previous example:

$$\begin{aligned} x_1[t] &= (s_2[t] + 3s_1[t] + 6)\cos(1.5\pi s_1[t]) \\ x_2[t] &= (s_2[t] + 3s_1[t] + 6)\sin(1.5\pi s_1[t]). \end{aligned} \tag{5.32}$$

We applied kTDSEP with the same parameters as in the previous experiment (kernel PCA applied to 500 randomly chosen input vectors, subspace dimensionality $d = 25$). As in the experiments before the linear method is not able to uncover the sources ($\mathrm{corr}(y_2^{lin}, s_1) = 0.7893, \mathrm{corr}(y_1^{lin}, s_2) = 0.0080$), but our nonlinear method is ($\mathrm{corr}(y_1, s_1) = 0.9917, \mathrm{corr}(y_{10}, s_2) = 0.9370$), as can be also seen in Figure 5.17.

5.4.7 More than two sources

In the experiments so far, we confined ourselves to mixtures of two sources because they can be nicely visualized. The next experiment demonstrates that our algorithm works also well with more than two sources:

We nonlinearly mix 7 audio sources $s[t] = [s_1[t], \ldots, s_7[t]]^\top$ (piano music, scientific utterance, cembalo music, street noise, cello music, funk music, political speech, each with 20,000 data points, sampling rate 8 kHz) by two steps:

1. Scale the signals between -1 and 1, i.e. they are contained inside the centered hypercube with side length 2. Rotate that cube such that its main diagonal (which has length $2\sqrt{7}$) is aligned with the first axis. This operation can be done by some orthogonal 7×7 matrix A.

[9] TDSEP involves simultaneous diagonalization of several $d \times d$ matrices, i.e. $\mathrm{O}(d^3)$.

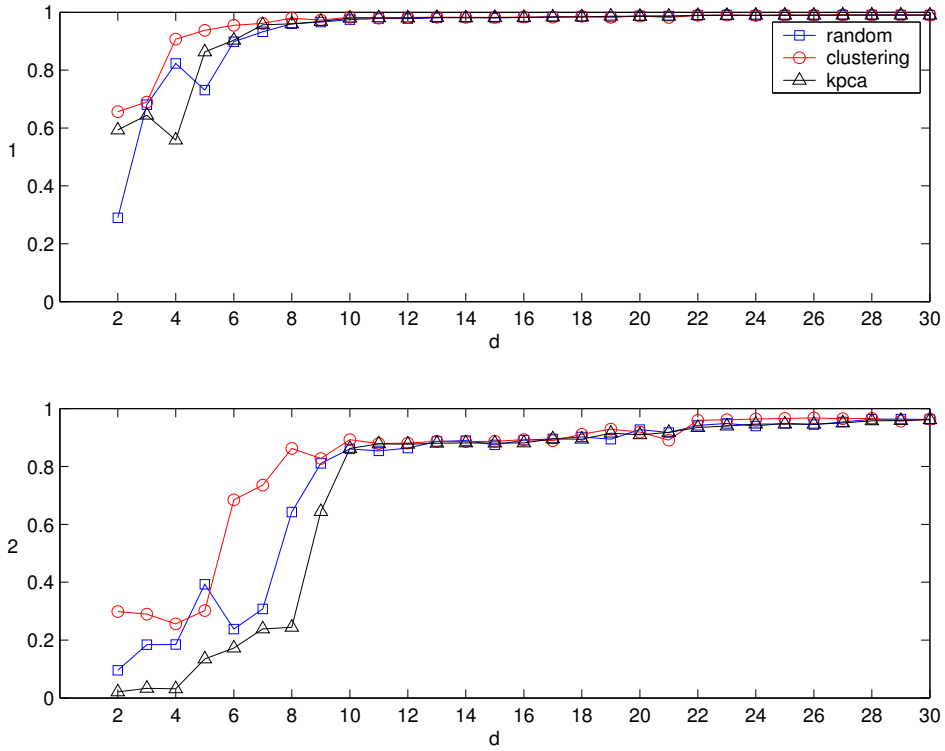


Figure 5.14: Correlations of the two source signals (upper and lower panel) to the best-matching extracted signals for Kernel PCA versus random sampling versus clustering and for different subspace dimensionalities d.

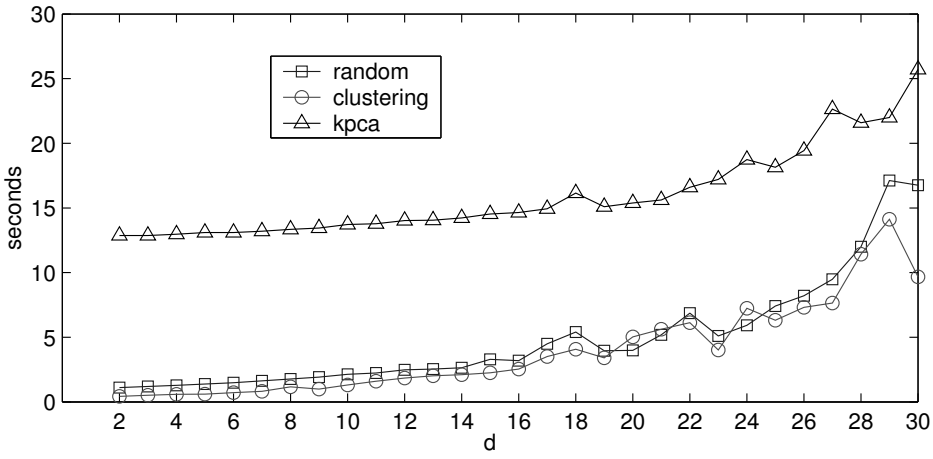


Figure 5.15: Computing times in seconds for the various runs of the experiment shown in Figure 5.14.

2. Rotate around different planes by an angle that depends on the first component of the vector $\bar{s}[t] = As[t]$, which we denote by $\bar{s}_1[t]$. More precisely, for a vector $\bar{s}[t]$ we define a 7×7 matrix

$$B(\bar{s}[t]) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \bar{s}_1[t] & 0 & 0 & 0 & 0 \\ 0 & -\bar{s}_1[t] & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \bar{s}_1[t] & 0 & 0 \\ 0 & 0 & 0 & -\bar{s}_1[t] & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \bar{s}_1[t] \\ 0 & 0 & 0 & 0 & 0 & -\bar{s}_1[t] & 0 \end{bmatrix} \tag{5.33}$$

and use it to define a rotation matrix (employing the matrix exponential)

$$C(\bar{s}[t]) = e^{\pi B(\bar{s}[t])} \tag{5.34}$$

that rotates along the 2nd and 3rd plane, along the 4th and 5th plane and along the 6th and 7th plane by the angle $\pi \bar{s}_1[t]$. Note that $C(\bar{s}[t])$ is an orthogonal matrix that is continuous in $\bar{s}[t]$.

The complete mixture reads then

$$x[t] = C(As[t])As[t]. \tag{5.35}$$

This nonlinear mixture is invertible because of the orthogonality and continuity of the matrices involved. Linear TDSEP is not able to demix $x[t]$: listening to the linearly demixed signals reveals that each component contains at least contributions of two

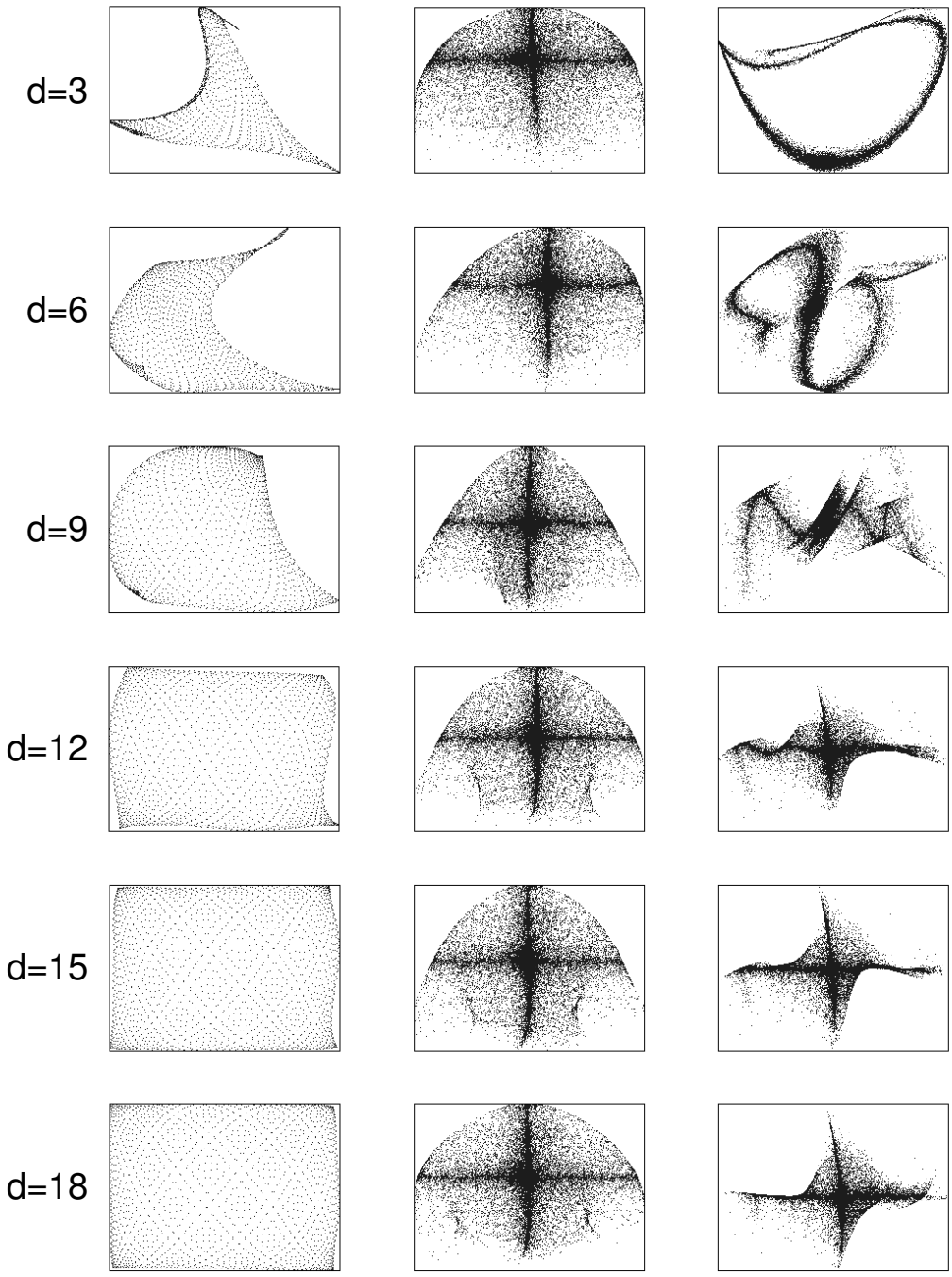


Figure 5.16: Scatterplots for different values of d: (left) artificial data, (middle) speech data—bended", (right) "speech data—twisted".

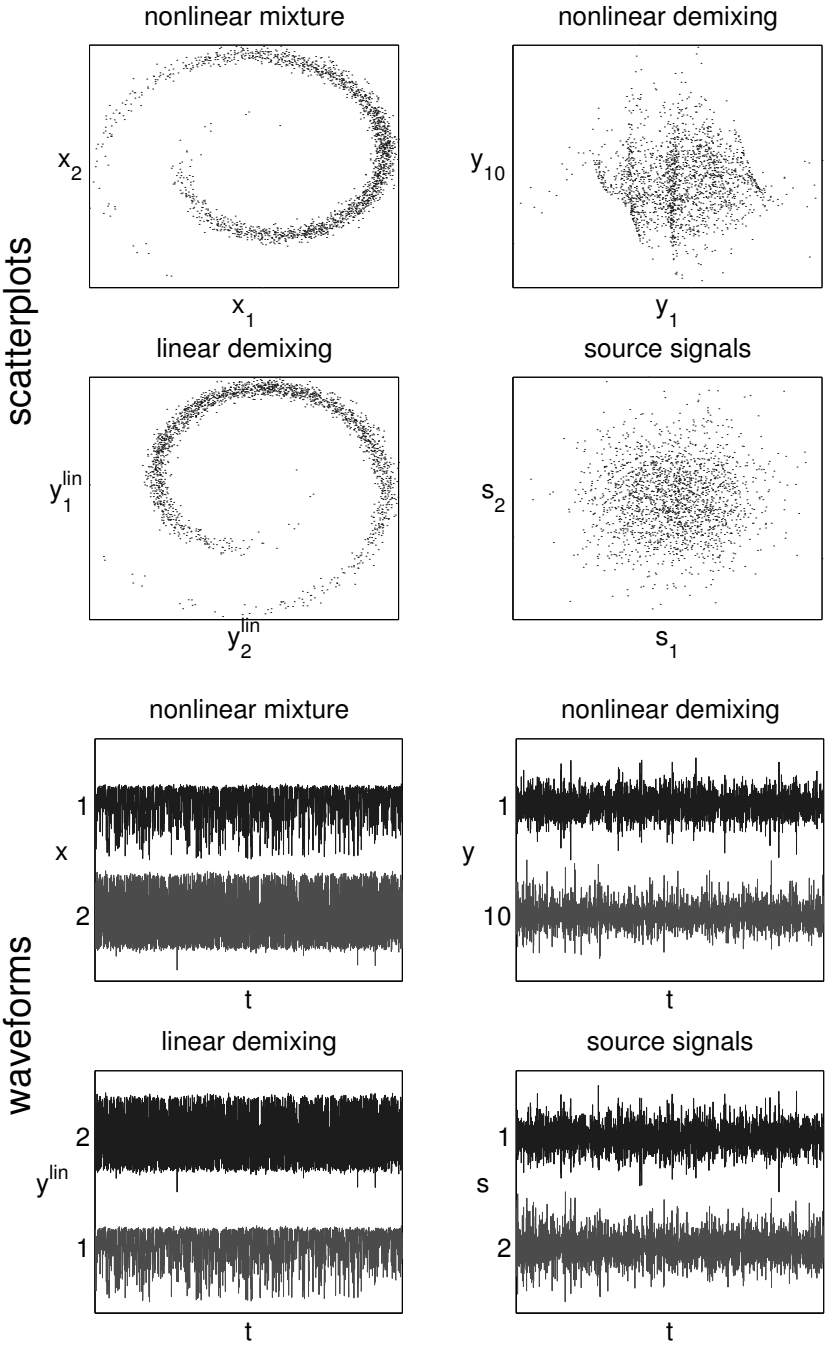


Figure 5.17: Stochastic artificial data: Scatterplots and waveforms of the nonlinear mixture and the nonlinear demixing (first and third row) and of linear demixing and the true source signals (second and fourth row).

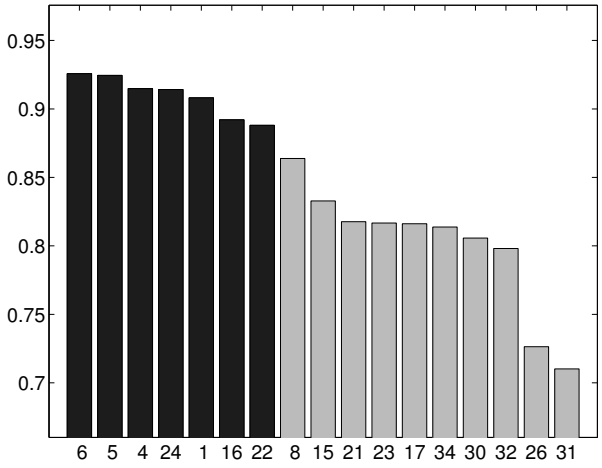


Figure 5.18: More than two sources: The 17 largest correlations between the demixed signals $y[t]$ and the signals that we obtain after applying kTDSEP again to $y[t]$. The numbers at the foot of the bars are the indices of the corresponding signals in $y[t]$. The seven left-most bars indicate the sought-after sources.

sources (and they are distorted as well). Their correlations with the true sources are below 0.82 ($\text{corr}(y_2^{lin}, s_1) = 0.7805$, $\text{corr}(y_1^{lin}, s_2) = 0.7415$, $\text{corr}(y_3^{lin}, s_3) = 0.6663$, $\text{corr}(y_4^{lin}, s_4) = 0.7481$, $\text{corr}(y_5^{lin}, s_5) = 0.7550$, $\text{corr}(y_6^{lin}, s_6) = 0.5929$, $\text{corr}(y_7^{lin}, s_7) = 0.8130$).

For kTDSEP, we apply k-means clustering to 500 randomly chosen input vectors and obtain $d = 40$ vectors $v_1, \ldots, v_{40}$. The mapped signals $\Phi(x[t])$ in the feature space (induced by a Gaussian RBF kernel $k(a, b) = \exp(-\|a - b\|^2)$) are projected onto the images of those 40 vectors. Applying TDSEP (with time-shifts $\tau = 0 \ldots 30$) to those resulting 40 signals $\Psi_x[t]$ and using the selection procedure described in Section 5.3.2 (see Figure 5.18), we find the seven sought-after sources. Those seven nonlinearly demixed signals not only have high correlations with the true sources ($\text{corr}(y_1, s_1) = 0.9432$, $\text{corr}(y_6, s_2) = 0.9496$, $\text{corr}(y_{24}, s_1) = 0.9394$, $\text{corr}(y_{16}, s_2) = 0.9218$, $\text{corr}(y_5, s_1) = 0.9508$, $\text{corr}(y_{22}, s_2) = 0.9142$, $\text{corr}(y_4, s_1) = 0.9402$), also their waveforms match the true sources very well (see right panels of Figure 5.19).

Note that the complexity of the algorithm depends mostly on the choice of d. I.e. even to demix seven nonlinearly mixed sources the most time-consuming part of the algorithm is to simultaneously diagonalize 31 time-shifted covariance matrices of size 40×40 which can be done very fast [25]. On a 600 MHz Pentium Laptop the Matlab calculation for this experiment with seven sources took less than 6 minutes.

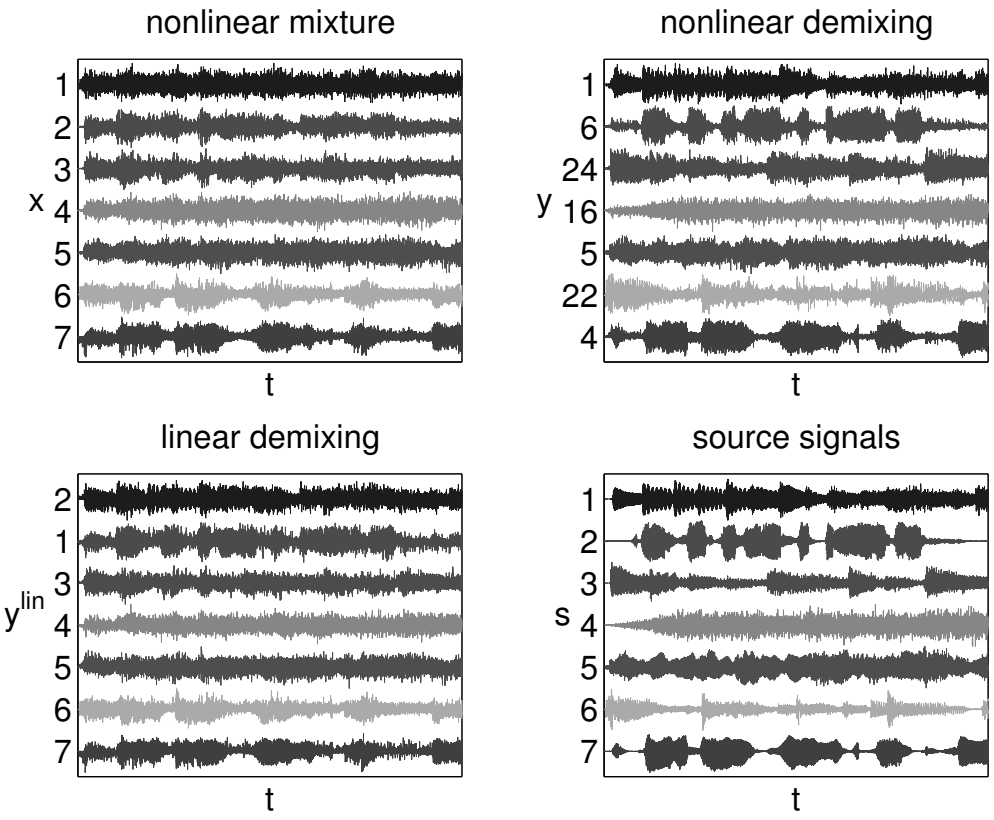


Figure 5.19: More than two sources: Waveforms of (upper left and right) the nonlinear mixture and the nonlinear demixing and of (lower left and right) linear demixing and the true source signals.

5.5 Summary

Our approach to the nonlinear ICA problem presented in this chapter combines three interesting ideas: kernel feature spaces, techniques for dimensionality reduction and blind source separation. The first two enable us to construct an orthonormal basis of the low-dimensional subspace in kernel feature space $\mathcal{F}$ where the data lies. This technique establishes a isomorphism (that preserves the scalar product) between the image of the data points in $\mathcal{F}$ and a d-dimensional space $\Re^d$. Moreover, we can acquire knowledge about the intrinsic dimension of the data manifold in $\mathcal{F}$ from the learning process.

In our formulation we approach the problem of nonlinear ICA from the viewpoint of kernel-based learning. The proposed kTDSEP algorithm allows to unmix arbitrary invertible nonlinear mixtures at low computational costs. The key to unmix difficult nonlinearities are the time correlations exploited by our algorithm via TDSEP. Intuitively speaking, the time structure provides the glue that yields the coherence for the separated signals.

[8] considered recently a further interesting application of kernel-based methods. However, in contrast to our algorithm that provides a nonlinear separation, Bach and Jordan's work is dealing with linear ICA and use the kernel trick in order to obtain a clever approximation of the mutual information.

Experiments on artificially generated signals and various audio signals, also with more than two sources, show the validity of our approach.

6 Synopsis

In this thesis we present new algorithms for extensions of classical ICA: for reliability assessment and grouping of independent components, for robust and overcomplete ICA, and for nonlinear ICA.

Reliability assessment of independent components is a concept that helps practitioners to interpret their results obtained from off-the-shelf ICA toolboxes. The important question is whether the calculated independent components reflect some meaningful underlying statistical structure of the data set or whether they are purely random. For certain ICA implementations this question has been answered by methods for reliability assessment based on bootstrap-resampling [71, 70, 67]. We have generalized these concepts based on the fundamental assumptions of classical ICA (Chapter 2 and designed a new method that applies to any ICA algorithm, because we view the chosen ICA method simply as a black box that is plugged into our noise-injection scheme (Chapter 3). The key idea of noise injection is to partially destroy the statistical structure of the extracted independent components by carefully adding Gaussian noise. Hereby we fade out the statistical structure in a controlled manner that allows us to analyze how the initial independent components are affected. This way we can estimate their reliability and also their grouping behavior.

Classical ICA algorithms are often sensitive to outliers. We presented a new ICA algorithm for super-Gaussian sources that is based on an index for outlier detection that uses nearest neighbor methods. The crucial idea is that the outlier index is not employed to remove outliers but instead directly to find inliers—the data points in the most concentrated regions—which represent the ICA directions for super-Gaussian source signals. Our inlier-based approach is by construction robust against outliers and applies naturally to the overcomplete ICA problem, in which there are more sources than sensors. In particular, our algorithm has the advantage that it does not optimize some cost function but instead it defines the columns of the mixing matrix in terms of some derived indices. Thus it can not get stuck into local minima and it has a fixed running time. A comparison of our new method with classical ICA algorithms in terms of robustness and a comprehensive empirical analysis of its performance with respect to dimensionality, number of sources and number of data points—underlines its key advantages.

Nonlinear ICA tries to unmix nonlinearly mixed signals, hereby finding a simple representation. We presented a kernel-based algorithm for nonlinear ICA that uses kernel feature spaces to approximate the nonlinearities. We empirically showed that applying linear ICA based on time structure implicitly in the resulting high-dimensional spaces can unmix strongly nonlinear mixtures. The main ingredient was a dimensionality reduction technique that made the application of ICA methods computationally and numerically tractable. Experiments demonstrated the excellent performance and

efficiency of our algorithm for several problems of nonlinear ICA.

Besides the representation of complicated time-series like industrial process data (see [97, 50]) there are more applications of nonlinear ICA conceivable, for example in the fields of telecommunications, array processing, vision and biomedical data analysis if the linear model does not hold. A particular interesting application of nonlinear ICA is the separation of nonlinearly mixed images which is a practical problem since it can appear in modern copy machines with thin paper [3].

An important direction for future research is to extend the kernel-approach to nonlinear ICA that is applicable to data without time structure. Such a method would allow us to perform density estimation in high dimensions because it would reduce the high-dimensional density estimation problem to several one-dimensional density estimation problems.

A Appendix: omitted proofs and lemmata

A.1 Whitening (also called sphering)

Let x be an n-dimensional random vector with PDF p_x and zero mean,

$$E_x\, x = \int x\, p(x)\, dx = 0. \tag{A.1}$$

If the entries of x are uncorrelated and have variance one, the covariance matrix of x equals the identity matrix I,

$$C_x := E_x\, xx^\top = \int xx^\top p(x)\; dx = I. \tag{A.2}$$

In that case we call x white.

Suppose x is not white. The problem of whitening is to find a matrix B, such that the linearly transformed random variable Bx is white. In practice we are given T data points $x[1], \ldots, x[T]$ that are IID according to p_x. View this data as an $n \times T$ matrix $X = [x[1], \ldots, x[T]]$. The sample covariance matrix is obtained from the covariance matrix above by replacing the integration by a finite sum,

$$\hat{C}_x := \frac{1}{T-1} \sum_{t=1}^{T} x[t]\, x[t]^\top = \frac{1}{T-1} XX^\top. \tag{A.3}$$

For non-white data, this matrix is not the identity, usually not even diagonal. In order to whiten we have to find a matrix B such the transformed data is white, i.e. its covariance matrix is the identity,

$$\hat{C}_{Bx} = \frac{1}{T-1} BX\, (BX)^\top = B\hat{C}_x B^\top = I. \tag{A.4}$$

This problem can be solved with the eigendecomposition of $\hat{C}_x$,

$$\hat{C}_x V = V\Lambda, \tag{A.5}$$

with V being an orthonormal matrix (i.e. $V^\top V = VV^\top = I$) of the eigenvectors (as columns) and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ being a diagonal matrix with the corresponding eigenvalues along the diagonal. Then we can show that the matrix

$$B := \Lambda^{-1/2} V^\top, \tag{A.6}$$

whitens x,

$$\hat{C}_{Bx} = B\,\hat{C}_x B^\top = \Lambda^{-1/2} V^\top \hat{C}_x\, V \Lambda^{-1/2} = \Lambda^{-1/2}\, V^\top V \Lambda\, \Lambda^{-1/2} = I \tag{A.7}$$

(for the third equality use Equation (A.5)). This whitening method is called principal component analysis (PCA). However, there are many other whitening matrices: let R be an orthonormal matrix (also called a rotation matrix; as before $RR^\top = R^\top R = I$). Then also the matrix RB whitens the data,

$$\hat{C}_{RBx} = R\,B\,\hat{C}_x B^\top R^\top = R\,I R^\top = I. \tag{A.8}$$

We see that whitened data stays white under orthonormal transformations. Also the converse is true: suppose x is white and R is any matrix, such that also Rx is white,

$$R\,\hat{C}_x R^\top = R\,I R^\top = R\,R^\top = I = R^\top R \tag{A.9}$$

(for the last equation recall that the right-inverse of squared matrices is also the left-inverse). We see that R must be a rotation (an orthonormal matrix). This is important for the ICA problem: the sources S are uncorrelated since they are statistically independent. Because of the scaling und permutation invariance of ICA we can assume without loss of generality that the sources are white. So we see that after whitening the observed data X, the remaining transformation to recover the sources remains a rotation. This fact is used by many ICA algorithms that use whitening as a preprocessing step.

Note that whitening uses only properties of the PDF p_X of the data matrix that are also present in the average marginal PDF p_x. For example time structure is not relevant for whitening. Nonetheless we call X white if the corresponding average marginal x is white.

A.2 Density of a transformation

Let x be a n-dimensional vector distributed according to p_x and A be an $n \times n$-matrix. The linearly transformed vector $y = Ax$ is then distributed according to

$$p_y(y) = |\det A|\, p_x(Ax). \tag{A.10}$$

This can be proved using the substitution rule for integrals (see for example [102]).

In the case of X and Y being $n \times T$ matrices the situation is slightly different. Assume X is distributed according to p_X. Let $\operatorname{vec} X$ be the nT-dimensional vector that contains the stacked columns of X, analogously $\operatorname{vec} Y$. For the vectorized matrices the relation $X = AY$ can be written as

$$\operatorname{vec} X = \begin{bmatrix} A & & 0 \\ & \ddots & \\ 0 & & A \end{bmatrix} \operatorname{vec} Y. \tag{A.11}$$

Noting $p_X(X) = p_{\operatorname{vec} X}(\operatorname{vec} X)$, analogously for Y, Equation (A.10) implies

$$p_Y(Y) = p_{\operatorname{vec} Y}(\operatorname{vec} Y) = |\det A|^T \, p_{\operatorname{vec} X}(\operatorname{vec} AY) = |\det A|^T \, p_Y(AY). \tag{A.12}$$

Note that T is the number of columns of X and Y and not the transpose operator, which is $\top$.

A.3 Connection between likelihood and KLD

The connection between the likelihood $\mathcal{L}(A, \theta) = q_X(X; A, \theta)$ and the KLD between the true PDF of X and the PDF implied by the model can be seen by taking the expectation with respect to the true PDF of X, denoted by p_X, of the negative logarithm of the likelihood:

$$\begin{aligned} -E_X \log \mathcal{L}(A, \theta) &= \int p_X(X) \, \log \frac{1}{q_X(X; A, \theta)} \, dX \\ &= \int p_X(X) \, \log \frac{p_X(X)}{q_X(X; A, \theta) p_X(X)} \, dX \\ &= \int p_X(X) \, \log \frac{p_X(X)}{q_X(X; A, \theta)} \, dX + h(p_X) \\ &= D(p_X \| q_X) + h(p_X). \end{aligned} \tag{A.13}$$

The third equality uses the differential entropy of X with respect to its true density function p_X, defined as $h(p_X) := -\int p_X(X) \, \log p_X(X) \, dX$. Finally, the fourth equality uses the definition of the KLD, $D(p\|q) := \int p(X) \, \log(p(X)/q(X)) \, dX$. We see that maximizing the likelihood is equivalent to minimizing the mismatch between the true and the modeled PDF.

The KLD between p_X and q_X can be rewritten as the KLD between the true PDF of $Y = A^{-1}X$ and q_S:

$$\begin{aligned} D(p_X \| q_X) &= \int p_X(X) \, \log \frac{p_X(X)}{q_X(X; A, \theta)} \, dX \\ &= \int p_X(AY) \, \log \frac{p_X(AY)}{q_X(AY; A, \theta)} \, |\det A|^T \, dY \\ &= \int \frac{p_Y(Y)}{|\det A|^T} \, \log \frac{p_Y(Y) \, |\det A|^T}{|\det A|^T \, q_S(Y)} \, |\det A|^T \, dY \\ &= \int p_Y(Y) \, \log \frac{p_Y(Y)}{q_S(Y)} \, dY \\ &= D(p_Y \| q_S). \end{aligned} \tag{A.14}$$

The second equality substitutes $X = AY$ using the substitution rule for integrals. The third equality employs $p_X(X) = p_Y(Y)/|\det A|^T$ (see Equation (A.12)) and the definition of the likelihood.

A.4 Invariance of the KLD under invertible transformations

Let p_X and q_X be two (possibly different) distributions of X. Let T be some linear transformation, $X = T(Y) = AY$, then we have $p_Y(Y) = |\det A|^T p_X(AY)$ and $q_Y(Y) = |\det A|^T q_X(AY)$. These identities imply

$$\begin{aligned} D(p_X \| q_X) &= \int p_X(X) \log \frac{p_X(X)}{q_X(X)} \, dX \\ &= \int p_X(AY) \log \frac{p_X(AY)}{q_X(AY)} |\det A|^T \, dX \\ &= \int \frac{p_Y(Y)}{|\det A|^T} \log \frac{p_Y(Y)|\det A|^T}{|\det A|^T q_Y(Y)} |\det A|^T \, dY \\ &= \int p_Y(Y) \log \frac{p_Y(Y)}{q_Y(Y)} \, dY \\ &= D(p_Y \| q_Y). \end{aligned} \tag{A.15}$$

which is almost identical to the second derivation in the last section. For general invertible transformation $X = T(Y)$ replace $|\det A|^T$ with the determinant of the Jacobian of T at $T^{-1}(X)$.

A.5 Decomposing the mutual information

The mutual information can be decomposed as follows:

$$\begin{aligned} I(y_1, \ldots, y_n) &= \frac{1}{T} D(p_Y^{\mathcal{G}} \| p_Y^{\mathcal{G} \cap \mathcal{I}}) \\ &= \frac{1}{T} \int p_Y^{\mathcal{G}}(Y) \log \frac{p_Y^{\mathcal{G}}(Y)}{p_Y^{\mathcal{G} \cap \mathcal{I}}(Y)} \, dY \\ &= -\frac{1}{T} \int p_Y^{\mathcal{G}}(Y) \log p_Y^{\mathcal{G} \cap \mathcal{I}}(Y) \, dY + \frac{1}{T} \int p_Y^{\mathcal{G}}(Y) \log p_Y^{\mathcal{G}}(Y) \, dY \\ &= -\frac{1}{T} \int \prod_{t'=1}^{T} p_y(Y_{:t'}) \log \prod_{t=1}^{T} \prod_{j=1}^{n} p_{y_j}(Y_{jt}) \, dY - \frac{1}{T} h(p_Y^{\mathcal{G}}) \\ &= -\frac{1}{T} \sum_{t=1}^{T} \sum_{j=1}^{n} \int \prod_{t'=1}^{T} p_y(Y_{:t'}) \log p_{y_j}(Y_{jt}) \, dY - \frac{1}{T} \left(h(p_X^{\mathcal{G}}) - \log |\det A|^T \right) \\ &= -\sum_{j=1}^{n} \int p_y(Y_{:t}) \log p_{y_j}(Y_{jt}) \, dY - \frac{1}{T} h\left(\prod_{t=1}^{T} p_x \right) + \log |\det A| \\ &= \left(\sum_{j=1}^{n} h(p_{y_j}) \right) - h(p_x) + \log |\det A| \end{aligned} \tag{A.16}$$

with $h(p_x)$ being the entropy of the PDF of the columns of X (averaged along the rows, see Equation (2.15)).

A.6 Jacobian and Hessian matrices of a matrix-valued matrix function

We follow the notation of [64]: for an $m \times p$ matrix-valued function that takes an $n \times q$ matrix argument,

$$\phi : \Re^{n \times q} \to \Re^{m \times p} \tag{A.17}$$

the Jacobian matrix of ϕ at X is the $mp \times nq$ matrix

$$\mathsf{D}\phi(X) = \frac{\partial \operatorname{vec} \phi(X)}{\partial (\operatorname{vec} X)^\top}, \tag{A.18}$$

with $\operatorname{vec} X$ being the vectorized matrix X, i.e. an $nq \times 1$ column vector containing the stacked columns of X. The Hessian matrix of ϕ at X is the $mnpq \times nq$ matrix

$$\mathsf{H}\phi(X) = \mathsf{D}(\mathsf{D}\phi(X))^\top = \frac{\partial \operatorname{vec} \mathsf{D}\phi(X)}{\partial (\operatorname{vec} X)^\top}. \tag{A.19}$$

Applying these notions to a real-valued function that takes an $n \times T$ matrix as its argument,

$$\phi : \Re^{n \times T} \to \Re, \tag{A.20}$$

the Taylor expansion (assuming that Z has small entries, and therefore omitting higher-order terms) reads

$$\phi(X + Z) \approx \phi(X) + \mathsf{D}\phi(X) \operatorname{vec}(Z) + (\operatorname{vec}(Z))^\top \mathsf{H}\phi(X) \operatorname{vec}(Z), \tag{A.21}$$

where the Jacobian matrix $\mathsf{D}\phi(X)$ is an $1 \times nT$ matrix (i.e. a row vector) and the Hessian matrix $\mathsf{H}\phi(X)$ is an $nT \times nT$ matrix.

A.7 Approximating the data manifold in feature space

For a polynomial kernel,

$$k(a, b) = (a^\top b + c)^p, \tag{A.22}$$

the feature space is finite-dimensional. For example, for homogeneous kernels, i.e. $c = 0$, the dimensionality can be calculated by the formula

$$\frac{(n + p - 1)!}{p!(n - 1)!} \tag{A.23}$$

with n being the dimensionality of the mixed signals (taken from [72]). For instance at $n = 3$ and for a polynomial kernel of degree $p = 5$ the feature space is 21-dimensional which can also be seen by plotting the largest eigenvalues of the corresponding kernel

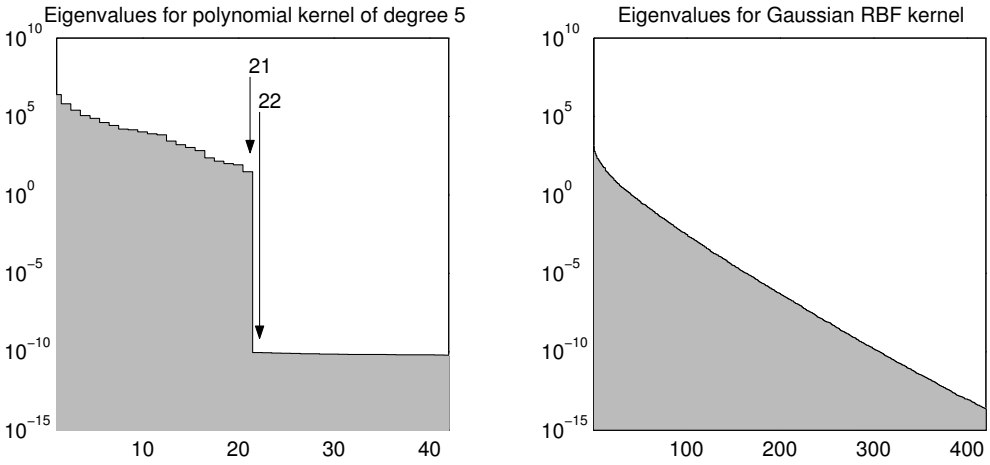


Figure A.1: The largest eigenvalues of the kernel matrix on a logarithmic scale: on the left for polynomial kernel of degree 5, on the right for Gaussian RBF kernel.

matrix (see Figure A.1, left panel). We clearly see the gap between the 21st and the remaining eigenvalues which should actually be zero[1]. Obviously, in this case we can fulfill Equation (5.2) with $d = 21$. For a Gaussian RBF kernel,

$$k(a, b) = e^{-\|a-b\|^2}, \tag{A.24}$$

the feature space is infinite-dimensional. However, T data points are always contained in a T-dimensional subspace—the one spanned by the mapped points themselves—but since the corresponding eigenvalues are decaying exponentially fast (see Figure A.1, right panel), the data in feature space can be approximated very well with a much lower-dimensional subspace (for more detailed discussions on this issue see [103, 8]). Therefore, we can approximate Equation (5.2) for suitable d.

[1] Those eigenvalues occur non-zero for numerical reasons.

Bibliography

[1] S. Achard, D.T. Pham, and C. Jutten. Blind source separation in post nonlinear mixtures. In T.-W. Lee, editor, *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2001)*, pages 295–300, 2001.

[2] L. Almeida. Linear and nonlinear ICA based on mutual information—the MISEP method. *Signal Processing*, 84:231–245, 2004.

[3] L. Almeida and M. Faria. Separating a real-life nonlinear mixture of images. In *Proc. Int. Workshop on Independent Component Analysis and Signal Separation (ICA 2004)*, Granada, Spain, 1999.

[4] S. Amari. *Differential-Geometrical Methods in Statistics*. Springer-Verlag, Berlin, 1985.

[5] S. Amari. Neural learning in structured parameter spaces—natural Riemannian gradient. In *Advances in Neural Information Processing Systems 9*, pages 127–133. MIT Press, 1997.

[6] S. 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.

[7] B. Ans, J. Hérault, and C. Jutten. Adaptive neural architectures: detection of primitives. In *Proceedings of COGNITIVA*, pages 593–597, Paris, France, 1985.

[8] F.R. Bach and M.I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

[9] F.R. Bach and M.I. Jordan. Beyond independent components: trees and clusters. *Journal of Machine Learning Research*, 4:1205–1233, 2003.

[10] V. Barnett and T. Lewis. *Outliers in statistical data*. Wiley series in probability and mathematical statistics. John Wiley & Sons Ltd., 2nd edition, 1978.

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

[12] A. Belouchrani, K. Abed Meraim, J.-F. Cardoso, and E. Moulines. A blind source separation technique based on second order statistics. *IEEE Trans. on Signal Processing*, 45(2):434–444, 1997.

[13] C.M. Bishop. Training with noise is equivalent to Tikohonov regularization. *Neural Computation*, 7:108–116, 1995.

[14] P. Bofill and M. Zibulevsky. Underdetermined blind source separation using sparse representations. *Signal Processing*, 81:2353–2362, 2001.

[15] A.M. Bronstein, M.M. Bronstein, M. Zibulevsky, and Y.Y. Zeevi. Blind deconvolution of images using optimal sparse representations. *IEEE Trans. on Image Processing*, 2004. to appear.

[16] G. Burel. Blind separation of sources: a nonlinear neural algorithm. *Neural Networks*, 5(6):937–947, 1992.

[17] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Knowledge Discovery and Data Mining*, 2(2):121–167, 1998.

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

[19] J.-F. Cardoso. Blind signal separation: statistical principles. *Proceedings of the IEEE*, 9(10):2009–2025, 1998.

[20] J.-F. Cardoso. High-order contrasts for independent component analysis. *Neural Computation*, 11(1):157–192, 1999.

[21] J.-F. Cardoso. The three easy routes to independent component analysis: contrasts and geometry. In T.-W. Lee, editor, *Proc. of the ICA 2001 workshop*, 2001.

[22] J.-F. Cardoso. Dependence, correlation and gaussianity in independent component analysis. Submitted, 2004.

[23] J.-F. Cardoso and D.-T. Pham. Separation of non-stationary sources: algorithms and performance. In S. Roberts and R. Everson, editors, *Independent Component Analysis: Principles and Practice*, pages 158–180. Cambridge University Press, 2001.

[24] J.-F. Cardoso and A. Souloumiac. Blind beamforming for non Gaussian signals. *IEE Proceedings-F*, 140(6):362–370, 1993.

[25] J.-F. Cardoso and A. Souloumiac. Jacobi angles for simultaneous diagonalization. *SIAM J.Mat.Anal.Appl.*, 17(1):161 ff., 1996.

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

[27] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*. MIT Press, 1989.

[28] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, UK, 2000.

[29] B. Efron and R.J. Tibshirani. *An Introduction to the Bootstrap*. Chapman and Hall, New York, 1994.

[30] B. Efron and R.J. Tibshirani. Improvements on cross-validation: the .632+ bootstrap method. *J. Amer. Statist. Assoc*, 92:548–560, 1997.

[31] S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

[32] C. Fyfe and P.L. Lai. ICA using kernel canonical correlation analysis. In *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2000)*, pages 279–284, Helsinki, Finland, 2000.

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

[34] G. Golub and C. van Loan. *Matrix Computations*. The Johns Hopkins University Press, 3rd edition, 1996.

[35] S. Harmeling, G. Dornhege, D. Tax, F. Meinecke, and K.-R. Müller. From outliers to prototypes: ordering data. Submitted, available from `http://ida.first.fhg.de/~harmeli/outliers.pdf`, 2004.

[36] S. Harmeling, F. Meinecke, and K.-R. Müller. Analysing ICA components by injecting noise. In *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2003)*, 2003.

[37] S. Harmeling, F. Meinecke, and K.-R. Müller. Injecting noise for analysing the stability of ICA components. *Signal Processing*, 84:255–266, 2004.

[38] S. Harmeling, P. von Bünau, A. Ziehe, and D.-T. Pham. Technical report on implementation of linear methods and validation on acoustic sources. Technical report, Deliverable D29 of the European joint project BLISS (Blind Source Separation and Applications, IST-1999-14190), 2003. Available from `http://www.lis.inpg.fr/pages_perso/bliss/`.

[39] S. Harmeling, A. Ziehe, M. Kawanabe, B. Blankertz, and K.-R. Müller. Nonlinear blind source separation using kernel feature spaces. In T.-W. Lee, editor, *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2001)*, pages 102–107, 2001.

[40] S. Harmeling, A. Ziehe, M. Kawanabe, and K.-R. Müller. Kernel feature spaces and nonlinear blind source separation. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002.

[41] S. Harmeling, A. Ziehe, M. Kawanabe, and K.-R. Müller. Kernel-based nonlinear blind source separation. *Neural Computation*, 15:1089–1124, 2003.

[42] J. Hérault and B. Ans. Circuits neuronaux à synapses modifiables: décodage de messages composites par apprentissage non supervisé. *C.-R. de l'Académie des Sciences*, 299(III-13):525–528, 1984.

[43] J. Hérault, C. Jutten, and B. Ans. Détection de grandeurs primitives dans un message composite par une architecture de calcul neuromimétique en apprentissage non supervisé. In *Actes du Xeme colloque GRETSI*, pages 1017–1022, Nice, France, 1985.

[44] J. Himberg, A. Hyvärinen, and F. Esposito. Validating the independent components of neuroimaging time-series via clustering and visualization. *NeuroImage*, 22(3):1214–1222, 2004.

[45] A. Honkela, S. Harmeling, L. Lundqvist, and H. Valpola. Using kernel PCA for initialisation of variational bayesian nonlinear blind source separation method. In C.G. Puntonet and A. Prieto, editors, *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2004)*, volume 3195 of *LNCS*, pages 790–797. Springer, 2004.

[46] A. Hyvärinen. One-unit contrast functions for independent component analysis: A statistical analysis. In *Neural Networks for Signal Processing VII (Proc. IEEE Workshop on Neural Networks for Signal Processing)*, pages 388–397, Amelia Island, Florida, 1997.

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

[48] A. Hyvärinen. Blind source separation by nonstationarity of variance: A cumulant-based approach. *IEEE Transactions on Neural Networks*, 12(6):1471–1474, 2001.

[49] A. Hyvärinen and M. Inki. Estimating overcomplete independent component bases for image windows. *Journal of Mathematical Imaging and Vision*, 17:139–152, 2002.

[50] A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley, 2001.

[51] A. Hyvärinen and E. Oja. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9(7):1483–1492, 1997.

[52] A. Hyvärinen and P. Pajunen. Nonlinear independent component analysis: Existence and uniqueness results. *Neural Networks*, 12(3):429–439, 1999.

[53] A. Hyvärinen, J. Särelä, and R. Vigário. Spikes and bumps: Artefacts generated by independent component analysis with insufficient sample size. In *Proc. Int. Workshop on Independent Component Analysis and Signal Separation (ICA'99)*, pages 425–429, Aussois, France, 1999.

[54] A. Iline, H. Valpola, and E. Oja. Detecting process state changes by nonlinear blind source separation. In T.-W. Lee, editor, *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2001)*, pages 704–709, 2001.

[55] C. Jutten. Source separation: from dusk till dawn. In *Proc. 2nd Int. Workshop on Independent Component Analysis and Blind Source Separation (ICA'2000)*, pages 15–26, Helsinki, Finland, 2000.

[56] C. Jutten, J. Karhunen, L. Almeida, and S. Harmeling. Technical report on separation methods of nonlinear mixtures. Technical report, Deliverable D29 of the European joint project BLISS (Blind Source Separation and Applications, IST-1999-14190), 2003. Available from `http://www.lis.inpg.fr/pages_perso/bliss/`.

[57] M. Kawamoto, K. Matsuoka, and M. Oya. Blind separation of sources using temporal correlation of the observed signals. *IEICE Trans. Fundamentals*, E80-A(4):695–704, 1997.

[58] H. Lappalainen and A. Honkela. Bayesian nonlinear independent component analysis by multi-layer perceptrons. In M. Girolami, editor, *Advances in Independent Component Analysis*, pages 93–121. Springer-Verlag, 2000.

[59] L.D. Lathauwer, B.D. Moor, and J. Vandewalle. Fetal electrocardiogram extraction by source subspace separation. In *Proceedings of HOS'95*, Aiguabla, Spain, 1995.

[60] T.-W. Lee, B.U. Koehler, and R. Orglmeister. Blind source separation of nonlinear mixing models. In *Neural Networks for Signal Processing VII*, pages 406–415. IEEE Press, 1997.

[61] M. Lewicki and T.J. Sejnowski. Learning overcomplete representations. *Neural Computation*, 12:337–365, 2000.

[62] J.K. Lin, D.G. Grier, and J.D. Cowan. Faithful representation of separable distributions. *Neural Computation*, 9(6):1305–1320, 1997.

[63] D. Loftsgaarden and C. Quesenberry. A nonparametric estimate of a multivariate density function. *Annals of Mathematical Statistics*, 36:1049–1051, 1965.

[64] J.R. Magnus and H. Neudecker. *Matrix differential calculus*. Wiley Series in Probability and Statistics. John Wiley & Sons, 1999.

[65] G. Marques and L. Almeida. Separation of nonlinear mixtures using pattern repulsion. In *Proc. Int. Workshop on Independent Component Analysis and Signal Separation (ICA'99)*, pages 277–282, Aussois, France, 1999.

[66] K. Matsuoka, M. Ohya, and M. Kawamoto. A neural net for blind separation of nonstationary signals. *Neural Networks*, 8(3):411–419, 1995.

[67] F. Meinecke. Resampling-techniken für ICA und ihre anwendungen in der biomedizinischen datenanalyse. Diplomarbeit, Institut für Physik, Universität Potsdam, 2003.

[68] F. Meinecke, S. Harmeling, and K.-R. Müller. Inlier-based ICA (IBICA) with an application to super-imposed images. *IJIST*, 2004. Submitted.

[69] F. Meinecke, S. Harmeling, and K.-R. Müller. Robust ICA for super-Gaussian sources. In C.G. Puntonet and A. Prieto, editors, *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2004)*, volume 3195 of *LNCS*, pages 217–224. Springer, 2004.

[70] F. Meinecke, A. Ziehe, M. Kawanabe, and K.-R. Müller. Estimating the reliability of ICA projections. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002.

[71] F. Meinecke, A. Ziehe, M. Kawanabe, and K.-R. Müller. A resampling approach to estimate the stability of one-dimensional or multidimensional independent components. *IEEE Transactions on Biomedical Engineering*, 49:1514–1525, Dez 2002.

[72] S. Mika. Kernel algorithms for nonlinear signal processing in feature spaces. Master's thesis, Technical University of Berlin, November 1998.

[73] L. Molgedey and H.G. Schuster. Separation of a mixture of independent signals using time delayed correlations. *Physical Review Letters*, 72:3634–3636, 1994.

[74] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.

[75] K.-R. Müller, R. Vigario, F. Meinecke, and A. Ziehe. Blind source separation techniques for decomposing event related brain signals. *International Journal of Bifurcation and Chaos*, 14(2):773–791, 2004.

[76] D. Obradovic and G. Deco. Information maximization and independent component analysis: Is there a difference? *Neural Computation*, 10(8):2085–2101, 1998.

[77] P.D. O'Grady and B.A. Pearlmutter. Hard-lost: Modified k-means for oriented lines. *Proceedings of the Irish Signals and Systems Conference 2004*, pages 247–252, 2004.

[78] P.D. O'Grady and B.A. Pearlmutter. Soft-lost: Em on a mixture of oriented lines. *ICA-2004 to appear*, 2004.

[79] E. Oja, S. Harmeling, and L. Almeida. Editorial for the special section on ICA. *Signal Processing*, 84:215–216, 2004.

[80] B.A. Olshausen and D.J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37:3311–3325, 1997.

[81] B.A. Olshausen and K.J. Millman. Learning sparse codes with a mixture-of-gaussians prior. *Advances in Neural Information Processing Systems, 12, MIT Press*, pages 841–847, 2000.

[82] P. Pajunen, A. Hyvärinen, and J. Karhunen. Nonlinear blind source separation by self-organizing maps. In *Proc. Int. Conf. on Neural Information Processing*, pages 1207–1210, Hong Kong, 1996.

[83] P. Pajunen and J. Karhunen. A maximum likelihood approach to nonlinear blind source separation. In *Proceedings of the 1997 Int. Conf. on Artificial Neural Networks (ICANN'97)*, pages 541–546, Lausanne, Switzerland, 1997.

[84] D.-T. Pham. Blind separation of instantaneous mixture of sources via the gaussian mutual information criterion. *Signal Processing*, 81:855–870, 2001.

[85] D.-T. Pham. Joint approximate diagonalization of positive definite matrices. *SIAM J. on Matrix Anal. and Appl.*, 22(4):1136–1152, 2001.

[86] D.-T. Pham and J.-F. Cardoso. Blind separation of instantaneous mixtures of non-stationary sources. In *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2000)*, pages 187–193, Helsinki, Finland, 2000.

[87] D.-T. Pham and P. Garrat. Blind separation of mixture of independent sources through a quasi-maximum likelihood approach. *IEEE Trans. on Signal Processing*, 45(7):1712–1725, 1997.

[88] 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.

[89] C.G. Puntonet, A. Prieto, C. Jutten, M. Rodriguez-Alvarez, and J. Ortega. Separation of sources: A geometry-based procedure for reconstruction of n-valued signals. *Signal Processing*, 46:267–284, 1995.

[90] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[91] B. Schölkopf, S. Mika, C.J.C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A.J. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, September 1999.

[92] B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[93] B. Schölkopf, A.J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[94] A.J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In P. Langley, editor, *Proc. ICML'00*, pages 911–918, San Francisco, 2000. Morgan Kaufmann.

[95] A. Taleb and C. Jutten. Source separation in post-nonlinear mixtures. *IEEE Trans. on Signal Processing*, 47(10):2807–2820, 1999.

[96] L. Tong, R.-W. Liu, V.C. Soon, and Y.-F. Huang. Indeterminacy and identifiability of blind identification. *IEEE Trans. on Circuits and Systems*, 38:499–509, 1991.

[97] H. Valpola, 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 (ICA2000)*, pages 351–356, Helsinki, Finland, 2000.

[98] V.N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag, New York, 1995.

[99] R. Vigário. Extraction of ocular artifacts from EEG using independent component analysis. *Electroenceph. Clin. Neurophysiol.*, 103(3):395–404, 1997.

[100] R. Vigário, A. Hyvärinen, and E. Oja. ICA fixed-point algorithm in extraction of artifacts from EEG. In *Proc. NORSIG'96*, pages 383–386, Espoo, Finland, 1996.

[101] R. Vigário, J. Särelä, V. Jousmäki, M. Hämäläinen, and E. Oja. Independent component approach to the analysis of EEG and MEG recordings. *IEEE Trans. Biomedical Engineering*, 47(5):589–593, 2000.

[102] L. Wasserman. *All of statistics*. Springer-Verlag, 2004.

[103] C.K.I. Williams and M. Seeger. The effect of the input density distribution on kernel-based classifiers. In *Proceedings of the Seventeenth International Conference on Machine Learning*. Morgan Kaufmann, 2000.

[104] C.K.I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In V. Tresp T.K. Leen, T.G. Dietrich, editor, *NIPS*, volume 13, pages 682–688. MIT Press, 2001.

[105] L. Wiskott and T.J. Sejnowski. Slow feature analysis: unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.

[106] H.H. Yang, S. Amari, and A. Cichocki. Information-theoretic approach to blind separation of sources in non-linear mixture. *Signal Processing*, 64(3):291–300, 1998.

[107] A. Yeredor. Non-orthogonal joint diagonalization in the least-squares sense with application in blind source separation. *IEEE Trans. Signal Processing*, 50(7):1545–1553, 2002.

[108] A. Ziehe, M. Kawanabe, S. Harmeling, and K.-R. Müller. Separation of post-nonlinear mixtures using ACE and temporal decorrelation. In T.-W. Lee, editor, *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2001)*, pages 433–438, 2001.

[109] A. Ziehe, M. Kawanabe, S. Harmeling, and K.-R. Müller. Blind separation of post-nonlinear mixtures using gaussianizing transformations and temporal decorrelation. In *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2003)*, pages 269–274, 2003.

[110] 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*, 4:1319–1338, Dec 2003.

[111] 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*, 5:777–800, 2004.

[112] A. Ziehe and K.-R. Müller. TDSEP - an efficient algorithm for blind separation using time structure. In L. Niklasson, M. Bodén, and T. Ziemke, editors, *Proceedings of the 8th International Conference on Artificial Neural Networks*, Perspectives in Neural Computing, pages 675–680, Berlin, 1998. Springer-Verlag.