8.07f/W(Jul162004) +model NEUCOM : 10633 Prod.Type:FTP pp.1–11(col.fig.:NIL) ED:SreejaGA PAGN:Kaushik SCAN:

Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Neurocomputing ▮ (▮▮▮▮) ▮▮▮–▮▮▮
NEUROCOMPUTING
www.elsevier.com/locate/neucom

Extending ICA for finding jointly dependent components from two related data sets

Juha Karhunen*, Tomas Ukkonen

Helsinki University of Technology, Adaptive Informatics Research Centre, P.O. Box 5400, FIN-02015 HUT, Espoo, Finland

Received 26 August 2005; received in revised form 17 May 2006; accepted 20 October 2006
Communicated by M. Welling

Abstract

In this paper, we introduce some methods for finding mutually corresponding dependent components from two different but related data sets in an unsupervised (blind) manner. The basic idea is to generalize cross-correlation analysis by taking into account higher-order statistics. We propose independent component analysis (ICA) type extensions for the singular value decomposition of the cross-correlation matrix. They extend cross-correlation analysis in a similar manner as ICA extends standard principal component analysis for covariance matrices. We present experimental results demonstrating the usefulness of the proposed methods both for artificially generated data and for a cryptographic problem.

Keywords: Independent component analysis; Cross-correlations; Singular value decomposition; Dependence measures; Blind signal separation

1. Introduction

Principal component analysis (PCA) [7,5,20] and independent component analysis (ICA) [20,5] are well-known techniques for unsupervised (blind) extraction of useful information from vector-valued data $\mathbf{x}$. While PCA is a well-established, old statistical technique, ICA has gained a lot of popularity during the last decade because it often provides more meaningful results.

Standard linear PCA and ICA are both based on the same type of simple linear latent variable model for the observed data vector $\mathbf{x}(t)$:

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) = \sum_{i=1}^{n} s_i(t)\mathbf{a}_i. \quad (1)$$

In this model, the data vector $\mathbf{x}(t)$ is expressed as a linear combination of scalar coefficients $s_i(t)$, $i = 1, 2, \ldots, n$, which multiply the respective constant basis vectors $\mathbf{a}_i$, $i = 1, 2, \ldots, n$. The scalar coefficients $s_i(t)$, $i = 1, 2, \ldots, n$, are different for each data vector $\mathbf{x}(t)$, depending directly on it. They can be collectively presented as the coefficient vector $\mathbf{s}(t) = [s_1(t), s_2(t), \ldots, s_n(t)]^{\mathrm{T}}$. The constant basis vectors $\mathbf{a}_i$, $i = 1, 2, \ldots, n$, are usually estimated by some criterion from the entire data set $\mathbf{x}(t)$, $i = 1, 2, \ldots, T$, where T is the number of available sample vectors. Hence they also depend on the properties of the data, but once they have been estimated, they are the same for all the data vectors belonging to this data set. The basis vectors $\mathbf{a}_i$ can be collectively presented in terms of the basis matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n]$.

The scalar coefficients $s_i(t)$ are in different contexts called principal components, independent components, source signals, latent variables, (hidden) factors, or (hidden) causes, depending on the problem and application at hand. The index t may denote time, position, or just number of the sample vector, again depending on the context. For simplicity, we assume here that both the data vector $\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^{\mathrm{T}}$ and the source vector $\mathbf{s}(t)$ are zero mean n-vectors, and that the basis matrix $\mathbf{A}$ is a full-rank constant $n \times n$ matrix. The column vectors $\mathbf{a}_i$, $i = 1, 2, \ldots, n$ of the matrix $\mathbf{A}$ comprise the basis vectors of PCA or ICA, and the components $s_i(t)$ of the source vector

*Corresponding author. Tel.: +358 9 451 3270; fax: +358 9 451 3277.
E-mail addresses: Juha.Karhunen@hut.fi (J. Karhunen), tomas.ukkonen@iki.fi (T. Ukkonen).
URL: http://www.cis.hut.fi/juha/.

0925-2312/$ - see front matter © 2006 Published by Elsevier B.V.
doi:10.1016/j.neucom.2006.10.144

$\mathbf{s}(t)$ are, respectively, principal or independent components corresponding to the data vector $\mathbf{x}(t)$.

From now on, we leave the index t out, assuming that the order of the data vectors $\mathbf{x}(t)$ is not important. This assumption is made in standard PCA and ICA. It is valid if the data vectors are randomly taken samples from some statistical distribution that the data obeys. However, the data vectors $\mathbf{x}(t)$ can have significant temporal structure, if they are subsequent samples from a vector-valued time series which is temporally correlated (non-white). Alternative methods to ICA have been developed for extracting the source signals or independent components in such cases. They usually utilize either temporal autocorrelations or non-stationary of variance; see [20,4,28]. These methods may work in cases which standard ICA is not able to handle, for example when the source signals are Gaussian, but on the other hand, they fail if the data does not have any temporal structure. ICA can often be successfully applied to temporally correlated data sets, too, but it is then not the optimal technique in the sense that it neglects the temporal information contained in the data.

In PCA, it is required that the basis matrix is orthogonal: $\mathbf{A}^T\mathbf{A} = \mathbf{I}$, implying that the basis vectors $\mathbf{a}_i$ are mutually orthonormal. In ICA, there is no such requirement, and hence the basis matrix $\mathbf{A}$, called there the mixing matrix, and the basis vectors $\mathbf{a}_i$ of ICA are generally non-orthogonal. In both the expansions, the components s_i must be mutually uncorrelated: $E\{s_i s_j\} = 0, i \neq j$. To get the true principal components, the variances $E\{s_i^2\}$ are in addition sequentially maximized for $i = 1, 2, \ldots, n$ [7,21,20,5]. Alternatively, principal components emerge from minimization of a mean-square approximation error criterion; see [7,21,20] for details.

In ICA, the orthogonality condition of PCA is replaced by the strong but often realistic requirement that the components s_i of the source vector $\mathbf{s}$ should be statistically independent (or as independent as possible). Furthermore, at most one of the independent components is allowed to have a Gaussian distribution. This still leaves the sign, order, and scaling of the independent components s_i ambiguous [20]. Usually they are scaled so that their variances $E\{s_i^2\} = 1$.

Assuming zero mean, $E\{\mathbf{x}\} = \mathbf{0}$, the covariance matrix of the data $\mathbf{x}$ is for both PCA and ICA

$$\mathbf{C}_{xx} = E\{\mathbf{x}\mathbf{x}^T\} = \mathbf{A}E\{\mathbf{s}\mathbf{s}^T\}\mathbf{A}^T = \mathbf{A}\mathbf{C}_{ss}\mathbf{A}^T, \quad (2)$$

where the covariance matrix $\mathbf{C}_{ss} = E\{\mathbf{s}\mathbf{s}^T\}$ of the source vector $\mathbf{s}$ is a diagonal matrix due to the uncorrelatedness of the components s_i.

Because PCA considers second-order statistics (covariances) only, it can be easily computed using the eigendecomposition of the covariance matrix $\mathbf{C}_{xx}$. The ith basis vector $\mathbf{a}_i$ of the PCA expansion (1) is the ith principal eigenvector of the matrix $\mathbf{C}_{xx}$, corresponding to its ith largest eigenvalue. The ith coefficient $s_i(t)$ of the PCA expansion (1) is then the projection $\mathbf{a}_i^T\mathbf{x}(t)$ of the data vector $\mathbf{x}(t)$ onto this eigenvector. The PCA basis vectors can be computed very efficiently using standard numerical software developed for symmetric eigenproblems. An alternative but much less accurate and efficient way is to apply linear PCA neural networks taught by Hebbian (and possibly anti-Hebbian) learning rules [7,5]. Such stochastic gradient algorithms for estimating the PCA expansion were developed by the first author together with Prof. E. Oja in a somewhat different context already in early 1980s [23,27]. Neural or other adaptive PCA estimation algorithms [6,5] are mainly useful in situations where it is necessary to adapt the PCA expansion to new incoming samples or to track slow changes in the statistical properties of the data.

Just like PCA, one can arrive at ICA from several different viewpoints or criteria. The most important ones are maximization of non-Gaussianity, maximum likelihood estimation, minimization of mutual information, and nonlinear decorrelation [20]. The ICA expansion is somewhat more difficult to estimate than PCA, requiring higher-order statistics in a form or another except for the case of time-correlated signals mentioned above. However, several good batch or adaptive neural type algorithms now exists for estimating the ICA expansion, too [20,5]. The two most popular ICA algorithms used currently are batch type FastICA algorithm(s) [20,28] and adaptive neural natural gradient algorithm [5,17,20].

Both standard PCA and ICA have been generalized into many different directions. Generalizations of PCA are discussed for example in [7,21,16], and generalizations of ICA in [20,5,17,28]. In this paper, we consider a generalization in which one tries to find mutually dependent corresponding components from two different but related data sets $\mathbf{X} = \mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(T_x)$ and $\mathbf{Y} = \mathbf{y}(1), \mathbf{y}(2), \ldots, \mathbf{y}(T_y)$ having T_x and T_y data vectors, respectively. For simplicity, we assume in this paper that such dependences appear between transformed components of the vectors $\mathbf{x}$ and $\mathbf{y}$ pairwise, while their other component pairs are statistically fairly independent. Possible time dependences between subsequent sample vectors $\ldots, \mathbf{x}(t-1), \mathbf{x}(t), \mathbf{x}(t+1), \ldots$ and $\ldots, \mathbf{y}(t-1), \mathbf{y}(t), \mathbf{y}(t+1), \ldots$ are neglected, or we assume that the data sets $\mathbf{X}$ and $\mathbf{Y}$ consist of randomly taken sample vectors from the respective vector-valued data distributions.

A well-known related statistical technique is canonical correlation analysis [26]. There one tries to find linear combinations x^* and y^* of the components of the vectors $\mathbf{x}$ and $\mathbf{y}$, respectively, so that x^* and y^* have maximal correlations. Because canonical correlation analysis resorts to second-order statistics only, its solution can again be found using eigenanalysis and singular value decomposition of auto- and cross-covariance matrices of $\mathbf{x}$ and $\mathbf{y}$ [26]. Fyfe and Lai have considered a neural implementation of canonical correlation analysis in [25], and a nonlinear generalization of it using kernels in [10]. Furthermore, Koetsier et al. have presented in [24] an unsupervised neural algorithm called exploratory correlation analysis for the extraction of common features in multiple data sources.

This method is closely related with canonical correlation analysis.

In an interesting paper, Akaho and his co-authors [2] have considered an ICA style generalization of canonical correlation analysis which they call multimodal independent component analysis (MICA). In their method, standard linear ICA is first applied to both data sets $\mathbf{x}$ and $\mathbf{y}$ separately. Then the corresponding dependent components of the two ICA expansions are identified using a natural gradient type learning rule. The method may work appropriately in practice in most cases, but it has a theoretical weakness. If two scalar variables s_1 and s_2 are statistically independent and similarly t_1 and t_2, but s_1 and t_1 depend on each other and similarly s_2 and t_2, one cannot in general theoretically deduce anything on the dependence or independence of the variable pairs s_1 and t_2 or s_2 and t_1. For example, s_1 and t_2 may have a common part which does not appear in s_2 and t_1, which makes them statistically dependent.

2. Theoretical background

2.1. Removal of second-order dependencies

Consider two different but related data sets $\mathbf{X} = \mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(T_x)$ and $\mathbf{Y} = \mathbf{y}(1), \mathbf{y}(2), \ldots, \mathbf{y}(T_y)$. The dimension m of the vectors $\mathbf{y}$ belonging to the data set $\mathbf{Y}$ is in general different from the dimension n of the vectors $\mathbf{x}$ belonging to the data set $\mathbf{X}$. Assuming zero mean also for $\mathbf{y}$, the cross-covariance matrix of $\mathbf{x}$ and $\mathbf{y}$ is theoretically defined by [20,30]

$$\mathbf{C}_{xy} = \mathrm{E}\{\mathbf{x}\mathbf{y}^{\mathrm{T}}\}. \tag{3}$$

The elements $\mathrm{E}\{x_i y_j\}$ of this matrix are the cross-covariances between the components x_i and y_j of the vectors $\mathbf{x}$ and $\mathbf{y}$, and they are in general non-zero. In practice, the probability distributions of the vectors $\mathbf{x}$ and $\mathbf{y}$ are usually not known. The cross-covariance matrix $\mathbf{C}_{xy}$ must then be estimated from the available pairs of sample vectors:

$$\hat{\mathbf{C}}_{xy} = \frac{1}{T}\sum_{i=1}^{T} \mathbf{x}_i \mathbf{y}_i^{\mathrm{T}}, \tag{4}$$

where $T = \min(T_x, T_y)$ [20,30].

The cross-covariance matrix $\mathbf{C}_{xy}$ (or in practice the estimated cross-covariance matrix $\hat{\mathbf{C}}_{xy}$) can be diagonalized using its singular value decomposition (SVD) (see for example [15,7,30]):

$$\mathbf{C}_{xy} = \mathbf{U}\mathbf{D}_{st}\mathbf{V}^{\mathrm{T}}. \tag{5}$$

Here $\mathbf{U}$ and $\mathbf{V}$ are $n \times n$ and $m \times m$ orthogonal matrices, respectively, and

$$\mathbf{D}_{st} = \mathrm{E}\{\mathbf{s}\mathbf{t}^{\mathrm{T}}\} \tag{6}$$

is an $n \times m$ (pseudo)diagonal matrix (that is, a diagonal matrix appended with zeros if $m \neq n$ [15]). The matrices $\mathbf{U}$ and $\mathbf{V}$ and $\mathbf{D}_{st}$ are obtained from the eigendecompositions of the symmetric matrices $\mathbf{C}_{xy}\mathbf{C}_{xy}^{\mathrm{T}}$ and $\mathbf{C}_{xy}^{\mathrm{T}}\mathbf{C}_{xy}$, respectively [7,15]. Standard PCA is a special case of the SVD expansion (5) in which $\mathbf{x} = \mathbf{y}$, $\mathbf{U} = \mathbf{V}$, and $\mathbf{s} = \mathbf{t}$. SVD can be estimated using neural PCA type algorithms [5], too, but we have in this work used more efficient and accurate standard numerical algorithms for computing it.

We can think that the diagonalization (5) of the cross-covariance matrix $\mathbf{C}_{xy}$ is realized via two orthogonal linear transformations $\mathbf{U}$ and $\mathbf{V}$:

$$\mathbf{x} = \mathbf{U}\mathbf{s}, \quad \mathbf{y} = \mathbf{V}\mathbf{t}, \tag{7}$$

where the corresponding components s_i and t_i of the vectors $\mathbf{s}$ and $\mathbf{t}$ are correlated: $\mathrm{E}\{s_i t_i\} \neq 0$, but their different components are uncorrelated: $\mathrm{E}\{s_i t_j\} = 0$ for $i \neq j$. Later on in our experiments, to make the comparisons easier, the variances of the components of the vectors $\mathbf{x}$ and $\mathbf{y}$ are always normalized to unity.

The key idea in this work is to allow non-orthogonal square transformation matrices $\mathbf{A}$ and $\mathbf{B}$ instead of $\mathbf{U}$ and $\mathbf{V}$:

$$\mathbf{x} = \mathbf{A}\mathbf{s}, \quad \mathbf{y} = \mathbf{B}\mathbf{t}. \tag{8}$$

In a similar manner as in standard linear ICA for one data set $\mathbf{x}$, we require that the transformations $\mathbf{A}$ and $\mathbf{B}$ not only make the different components s_i and t_j, $i \neq j$, of the vectors $\mathbf{s}$ and $\mathbf{t}$ uncorrelated, but they should be as independent as possible. The goal is to concentrate the dependencies between the vectors $\mathbf{s}$ and $\mathbf{t}$ as far as possible to their corresponding components s_i and t_i, which are in turn required to be as dependent as possible.

Using the transformations (8), the cross-covariance matrix $\mathbf{C}_{xy}$ can be expressed as

$$\mathbf{C}_{xy} = \mathbf{A}\mathbf{D}_{st}\mathbf{B}^{\mathrm{T}}. \tag{9}$$

It should be noted that it is always possible to find orthogonal matrices $\mathbf{U}$ and $\mathbf{V}$ which provide the SVD (5), and make the different components of the vectors $\mathbf{x}$ and $\mathbf{y}$ uncorrelated. By finding suitable transformations (8) among the considerably more flexible class of non-orthogonal matrices $\mathbf{A}$ and $\mathbf{B}$, one should therefore in general be able to achieve more than just decorrelation.

2.2. Removal of higher-order dependencies

Our approach for computing the matrices $\mathbf{A}$ and $\mathbf{B}$ is based on nonlinear decorrelation and the FastICA algorithm [20]. The algorithm has converged to a good solution when

$$\mathrm{E}\{\mathbf{x}\mathbf{g}(\mathbf{x})^{\mathrm{T}}\} \tag{10}$$

is a diagonal matrix, and the data vectors $\mathbf{x}$ have been preprocessed to have zero mean and unit variance. The vector $\mathbf{g}(\mathbf{x}) = [g(x_1), g(x_2), \ldots, g(x_n)]^{\mathrm{T}}$ is a nonlinear transformation of the data vector $\mathbf{x}$. The nonlinearity $g(t)$ must be chosen carefully in order to get as independent signals as

Please cite this article as: J. Karhunen, T. Ukkonen, Extending ICA for finding jointly dependent components from two related data sets, Neurocomputing (2006), doi:10.1016/j.neucom.2006.10.144

possible. Good nonlinearities for wide classes of signals are $g(t) = \tanh(t)$ or $g(t) = t^3$.

Theoretically, statistical independence of the variables x_i and x_j requires that $\mathrm{E}\{f(x_i)g(x_j)\} = \mathrm{E}\{f(x_i)\}\mathrm{E}\{g(x_j)\}$ for all continuous functions f and g that are zero outside a finite interval. However, it can be justified (see [20, Section 12.1]) that the variables x_i and x_j, $i \neq j$, are usually statistically independent when their nonlinear correlations are zero:

$$\mathrm{E}\{x_i g(x_j)\} = 0 \quad \text{or} \quad \mathrm{E}\{x_j g(x_i)\} = 0. \tag{11}$$

Here it is assumed that x_i and x_j have zero mean and that g is an odd nonlinear function.

Therefore, to remove cross-dependencies between the zero mean vectors $\mathbf{x}$ and $\mathbf{y}$, we should diagonalize the matrices

$$\mathrm{E}\{\mathbf{x}\mathbf{y}^{\mathrm{T}}\}, \quad \mathrm{E}\{\mathbf{x}\mathbf{g}(\mathbf{y})^{\mathrm{T}}\}, \quad \mathrm{E}\{\mathbf{g}(\mathbf{x})\mathbf{y}^{\mathrm{T}}\}. \tag{12}$$

We can try to roughly diagonalize all these matrices by diagonalizing just their sum matrix

$$\mathrm{E}\{\mathbf{x}\mathbf{y}^{\mathrm{T}} + \mathbf{x}\mathbf{g}(\mathbf{y})^{\mathrm{T}} + \mathbf{g}(\mathbf{x})\mathbf{y}^{\mathrm{T}}\}. \tag{13}$$

In this respect, our method resembles ICA and blind source separation (BSS) methods based on lagged covariance matrices, where one also tries to simultaneously diagonalize several lagged covariance matrices approximately; see for example [20, Section 18.1.3].

The matrix (13) can be further generalized to

$$\mathrm{E}\{[\mathbf{x} + (\mathbf{g}(\mathbf{x}) - \mathrm{E}\{\mathbf{g}(\mathbf{x})\})][\mathbf{y} + (\mathbf{g}(\mathbf{y}) - \mathrm{E}\{\mathbf{g}(\mathbf{y})\})]^{\mathrm{T}}\}, \tag{14}$$

where the term $\mathrm{E}\{(\mathbf{g}(\mathbf{x}) - \mathrm{E}\{\mathbf{g}(\mathbf{x})\})(\mathbf{g}(\mathbf{y}) - \mathrm{E}\{\mathbf{g}(\mathbf{y})\})^{\mathrm{T}}\}$ vanishes when the vectors $\mathbf{x}$ and $\mathbf{y}$ are independent.

We can diagonalize this form by simple use of SVD. In general, we want to diagonalize the matrix

$$\mathrm{E}\{\mathbf{f}(\mathbf{x})\mathbf{g}(\mathbf{y})^{\mathrm{T}}\} = \mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}. \tag{15}$$

We can do this nonlinearly with transforms

$$\mathbf{x}' = \mathbf{f}^{-1}(\mathbf{U}^{\mathrm{T}}\mathbf{f}(\mathbf{x})), \quad \mathbf{y}' = \mathbf{g}^{-1}(\mathbf{V}^{\mathrm{T}}\mathbf{g}(\mathbf{y})) \tag{16}$$

provided that the inverse functions $\mathbf{f}^{-1}(\cdot)$ and $\mathbf{g}^{-1}(\cdot)$ exist.

Assume now that the data vectors $\mathbf{x}$ and $\mathbf{y}$ have been whitened and cross-decorrelated. For vector-valued functions $\mathbf{f}(\mathbf{x})$ which map their components independently, the optimal linear mapping in the mean-square error sense is then $\mathbf{f}(\mathbf{x}) = \mathbf{A}\mathbf{x} = a\mathbf{I}\mathbf{x}$, and similarly for $\mathbf{g}(\mathbf{y})$. This can be used to find linear approximations to the nonlinear diagonalizing transforms (16):

$$\mathbf{x}' = a^{-1}\mathbf{I}\mathbf{U}^{\mathrm{T}}a\mathbf{I}\mathbf{x} = \mathbf{U}^{\mathrm{T}}\mathbf{x}, \tag{17}$$

$$\mathbf{y}' = b^{-1}\mathbf{I}\mathbf{V}^{\mathrm{T}}b\mathbf{I}\mathbf{y} = \mathbf{V}^{\mathrm{T}}\mathbf{y}. \tag{18}$$

We have used these approximations in context with Method 2, which will be described in the next section. Although suboptimal, they turned out to provide good results in our experiments. However, some preprocessing involving higher-order statistics and/or nonlinearities is required before applying them, because otherwise (17) and (18) would give only orthogonal transformations of $\mathbf{x}$ and $\mathbf{y}$ which cannot find independence.

3. Methods

We have developed and tested several somewhat heuristic methods based on the above ideas. We restricted our testing to matrices which have a similar form as in Eq. (13). Eqs. (17) and (18) could also be used iteratively to totally or significantly reduce non-diagonal values of this form of matrix containing nonlinear correlations. In the following, we present the two methods which performed on average best in our experiments.

3.1. Method 1

In the first method (Method 1), we first estimate the independent components[1] of the vectors $\mathbf{x}$ and $\mathbf{y}$ using the FastICA method [20]. Let us denote the vectors containing these estimated independent components by $\hat{\mathbf{s}}_x$ and $\hat{\mathbf{s}}_y$:

$$\mathbf{x} = \mathbf{A}\hat{\mathbf{s}}_x, \quad \mathbf{y} = \mathbf{B}\hat{\mathbf{s}}_y. \tag{19}$$

Here $\mathbf{A}$ is an $n \times n$ matrix and $\hat{\mathbf{s}}_x$ an n-dimensional vector, and $\mathbf{B}$ is an $m \times m$ matrix and $\hat{\mathbf{s}}_y$ an m-dimensional vector. Furthermore, the variances of vectors $\hat{\mathbf{s}}_x$ and $\hat{\mathbf{s}}_y$ were normalized to unity for getting suitable starting vectors.

After this, the SVD of the matrix

$$\mathbf{F}_{xy} = \mathrm{E}\{\mathbf{x}\mathbf{y}^{\mathrm{T}} + \tanh(\mathbf{x})\mathbf{y}^{\mathrm{T}} + \mathbf{x}\tanh(\mathbf{y})^{\mathrm{T}}\} = \mathbf{U}_F\mathbf{D}_F\mathbf{V}_F^{\mathrm{T}} \tag{20}$$

containing nonlinear correlations of the vectors $\mathbf{x}$ and $\mathbf{y}$ is computed quite similarly as for the standard cross-correlation matrix $\mathbf{C}_{xy}$ in (5). On the right-hand side of Eq. (20), $\mathbf{U}_F$ and $\mathbf{V}_F$ denote the orthogonal left and right matrices of the SVD of the matrix $\mathbf{F}_{xy}$, and the diagonal matrix $\mathbf{D}_F$ contains the respective singular values. The nonlinearity, in (20) $\tanh(\cdot)$, is applied to each component of the vectors $\mathbf{x}$ and $\mathbf{y}$ separately.

Finally, the estimated source (independent component) vectors $\hat{\mathbf{s}}_x$ and $\hat{\mathbf{s}}_y$ in Eq. (19) are rotated using the singular vector matrices $\mathbf{U}_F$ and $\mathbf{V}_F$, yielding the final results

$$\mathbf{s}_x^* = \mathbf{U}_F^{\mathrm{T}}\hat{\mathbf{s}}_x, \quad \mathbf{s}_y^* = \mathbf{V}_F^{\mathrm{T}}\hat{\mathbf{s}}_y. \tag{21}$$

The basic idea behind this method is to include nonlinear correlations of the components of the vectors $\mathbf{x}$ and $\mathbf{y}$ into computation of the matrix $\mathbf{F}_{xy}$. In (20), the sigmoidal $\tanh(\cdot)$ nonlinearity is applied to $\mathbf{x}$ and $\mathbf{y}$ to achieve this goal.

This is a heuristic way to try to concentrate the dependencies between $\mathbf{x}$ and $\mathbf{y}$ into their corresponding components. That is, ideally there should exist one component in the vector $\mathbf{s}_y^*$ which is dependent on the selected component of the vector $\mathbf{s}_x^*$, while these two components are statistically independent of the other components of the vectors $\mathbf{s}_x^*$ and $\mathbf{s}_y^*$. But due to the averaged nature of the expectation defining the matrix $\mathbf{F}_{xy}$, this is in practice usually not achieved at least perfectly.

[1] Or the most independent components if strictly statistically independent components do not exist.

We tried several related methods in our experiments. In some of them, preprocessing took place instead of ICA using PCA whitening. We tried also the cubic nonlinearities $\mathbf{x}^3\mathbf{y}^{\mathrm{T}}$ and $\mathbf{x}(\mathbf{y}^{\mathrm{T}})^3$, but they seem to be sensitive to noise and did not provide as good results as Method 1. Method 1 was selected to this paper because it provided on average the best results and is computationally sufficiently efficient.

3.2. Method 2

The just described Method 1 tries to find one-dimensional signal pairs s_i, t_i where all the relevant information about the ith component t_i of the vector $\mathbf{t}$ has been concentrated onto the corresponding component s_i of the vector $\mathbf{s}$ and vise versa. These ideas can be also used to find a linear mapping between two sets of signals.

Method 2 extends the linear mean-square error minimization to a more generic linear method. The method relaxes assumptions about distributions of signals and errors. The idea is to solve signal pairs with one of the methods described above, and then find one-dimensional mappings minimizing the mean-square error between s_i and t_i pairs. These one-dimensional mappings are sufficient for cross-independent signal pairs, where the signals $s_j, j \neq i$ do not contain any information about the correct value of t_i. An optimum linear mapping minimizing the mean-square error changes only the sign and scaling of zero mean signals [14], and can be carried out without changing mutual information (or independencies) between the variables

$$t_i = \rho_{t_i s_i} s_i, \quad I(\rho_{t_i s_i} X, Y) = I(X, Y), \tag{22}$$

where $\rho_{t_i s_i}$ is correlation between t_i and s_i. Thus if the given data have been sphered to have zero mean and unit variance, and the cross-dependence matrix $\mathbf{G}_{xy}$ can be diagonalized with mappings $\mathbf{s} = \mathbf{U}^{\mathrm{T}}\mathbf{x}$ and $\mathbf{t} = \mathbf{V}^{\mathrm{T}}\mathbf{y}$ then the mapping from $\mathbf{x}$ to $\mathbf{y}$ is

$$\mathbf{W} = \mathbf{V}\,\mathrm{diag}(\rho_{t_1 s_1}, \rho_{t_2 s_2}, \ldots, \rho_{t_N s_N})\mathbf{U}^{\mathrm{T}}. \tag{23}$$

This method can be seen as an extension of linear mean-square error optimization which assumes Gaussian distributions. If a cross-correlation matrix $\mathbf{C}_{xy}$ is used as a cross-dependence matrix $\mathbf{G}_{xy}$, then the resulting mapping becomes

$$\mathbf{W} = \mathbf{V}\,\mathrm{diag}(\rho_{t_1 s_1}, \rho_{t_2 s_2}, \ldots, \rho_{t_N s_N})\mathbf{U}^{\mathrm{T}} = \mathbf{V}\mathbf{S}\mathbf{U}^{\mathrm{T}} = \mathbf{C}_{xy}^{\mathrm{T}}. \tag{24}$$

It can be seen that this is exactly the same as given by linear mean-square error minimization for sphered data. With a bit of calculus one can see that minimization of the mean-square error criterion

$$\mathrm{E}\left\{\tfrac{1}{2}\|\mathbf{W}\mathbf{x} - \mathbf{y}\|^2\right\} \tag{25}$$

yields the optimal solution

$$\mathbf{W} = \mathbf{C}_{yx}\mathbf{C}_x^{-1} = \mathbf{C}_{xy}^{\mathrm{T}}, \tag{26}$$

where the last step follows from the whitening (sphering) of the data $\mathbf{x}$: $\mathbf{C}_x = \mathbf{C}_x^{-1} = \mathbf{I}$. So if diagonalization of a cross-dependence matrix removes most of the correlations between different components of the involved vectors, then Method 2 minimizes the mean-square error and at the same time it tries to take non-Gaussian properties of distributions into account.

In our tests we preprocessed the data with PCA to have zero mean and unit variance, and then used the cross-dependence matrix

$$\mathbf{G}_{xy} = \mathrm{E}\{\tanh(\mathbf{x})\mathbf{y}^{\mathrm{T}} + \mathbf{x}\tanh(\mathbf{y})^{\mathrm{T}}\} \tag{27}$$

which was iteratively diagonalized with SVD. Brief tests indicated that this method has about same minimum mean-square error (25) when $\mathbf{y} = \mathbf{A}\mathbf{x} + \varepsilon$. But sometimes the method performed considerably better than the standard pseudoinverse based least-square error minimization (26) when the output vectors $\mathbf{y}$ were generated from $\mathbf{x}$ with two different matrices $\mathbf{y} = \mathbf{A}\mathbf{x}$ and $\mathbf{y} = \mathbf{B}\mathbf{x}$. A problem with Method 2 is that it suffers from the same theoretical weakness as Akaho's et al. method [2], mentioned at the end of Introduction.

4. Measuring the dependence

Theoretically, a suitable measure of the dependence between any two continuous scalar random variables x and y is their mutual information [16,20]

$$I_{xy} = \int_{-\infty}^{+\infty} p_x(x) \log \frac{p_x(x)}{p_y(y)}\,\mathrm{d}x\,\mathrm{d}y, \tag{28}$$

where $p_x(x)$ and $p_y(y)$ denote the probability density functions of x and y, respectively. The mutual information can easily be generalized for vector-valued random variables $\mathbf{x}$ and $\mathbf{y}$. It is actually the Kullback–Leibler divergence (information) between x and y, and measures the distance between the probability densities $p_x(x)$ and $p_y(y)$ [16,20].

Mutual information I_{xy} is strictly speaking not a proper distance measure because it is not symmetric for x and y. But it has the following important theoretical property: mutual information is always non-negative, and it is zero if and only if x and y are statistically independent. The more dependent they are the larger is their mutual information I_{xy}.

While mutual information is in some sense a theoretically ideal dependence measure, it cannot usually be applied in practice. The basic reason is that it is very difficult to reliably estimate the tails of the distributions $p_x(x)$ and $p_y(y)$ [11,20]. Therefore, one must resort to some kind of approximations (see for example [20,5]) or to other simpler dependence measures.

A review of dependence measures related to tests of independence in statistics can be found in the paper [31]. However, such tests are not necessarily most suitable in context with ICA, because they typically make specific assumptions on the distributions of the variables to be studied (for example, Gaussianity).

Please cite this article as: J. Karhunen, T. Ukkonen, Extending ICA for finding jointly dependent components from two related data sets, Neurocomputing (2006), doi:10.1016/j.neucom.2006.10.144

In ICA and BSS, measures of statistical dependence have been developed and studied in several papers. Bach and Jordan [3] have introduced contrast functions based on canonical correlations in a reproducing kernel Hilbert space. They have shown that these contrast functions are related to mutual information and have desirable mathematical properties as measures of statistical dependence. Their ideas have recently been developed further in [12], where two new kernel-based functionals are introduced for measuring the degree of independence of random variables.

Another way is to use characteristic functions for defining statistical independence and for measuring dependence. This approach has been studied in [8,9], leading to three criteria for ICA. Dependence measures can be based either on approximating mutual information using the characteristic function or on applying a moment generating function. Furthermore, simpler quadratic measures for estimating dependence have been developed in [1,32].

We made preliminary experiments with a few of these methods using simple test cases of three statistically independent source signals. We chose the method based on moment generating function because for it the difference between the cases of independent and more or less dependent signals was the largest. However, also the other tested methods gave qualitatively correct results. That is, more independent variables provided better values of the respective performance index than more dependent ones.

In the following, we explain the dependence measure derived from the method based on moment generating function [8,9] in more detail. The moment generating function method is based on estimation of the expectation

$$\mathrm{E}[\exp(\mathbf{w}^{\mathrm{T}}\mathbf{x})] = \mathrm{E}\left[\exp\left(\sum_{i=1}^{n} w_i x_i\right)\right] \tag{29}$$

over the components $x_1, x_2, \ldots, x_n$ of the data vector $\mathbf{x}$. Here $\mathbf{w}$ is the weight vector whose components $w_1, w_2, \ldots, w_n$ define some linear combination of the components of $\mathbf{x}$. Clearly, if $\mathbf{w}^{\mathrm{T}}$ is one of the row vectors of the inverse $\mathbf{A}^{-1}$ of the square mixing matrix $\mathbf{A}$ in the standard linear ICA model (1), $\mathbf{w}^{\mathrm{T}}\mathbf{x}$ becomes one of the independent components s_j [20]. On the other hand, if the components x_i of $\mathbf{x}$ in (29) are statistically independent, Eq. (29) decouples into

$$\mathrm{E}[\exp(\mathbf{w}^{\mathrm{T}}\mathbf{x})] = \prod_{i=1}^{n} \mathrm{E}[\exp(w_i x_i)]. \tag{30}$$

Based on this observation, one can estimate for two scalar random variables x_i and x_j the quantity

$$d_{x_i x_j}(w_i, w_j) = \{\mathrm{E}[\exp(w_i x_i + w_j x_j)] - \mathrm{E}[\exp(w_i x_i)]\mathrm{E}[\exp(w_j x_j)]\}^2. \tag{31}$$

This is always non-negative, and becomes zero when the variables x_i and x_j are independent. The moments and moment generating function do not uniquely define the variables x_i and x_j, but a large correspondence implies that the functions are similar.

In the experiments, we measured the independence of a two-dimensional random variable by computing the function [8,9]

$$I_{x_i x_j}[\mathbf{w}] = d_{x_i x_j}(w_i, w_j) d_{x_i x_j}(-w_i, -w_j) + d_{x_i x_j}(-w_i, w_j) d_{x_i x_j}(w_i, -w_j). \tag{32}$$

This is a positive, real-valued function measuring the dependence. We generated this function only at the point $\mathbf{w} = (1, 1)$. Finally, the quality of the found solution was assessed by computing the quantity

$$J(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n} I_{x_i y_i}(1, 1)}{\sum_{i=1}^{n} \sum_{j \neq i}^{n} I_{x_i y_j}(1, 1)}. \tag{33}$$

The higher the value of $J(\mathbf{x}, \mathbf{y})$ is, the more dependent $\mathbf{x}$ and $\mathbf{y}$ are. This is a measure of goodness which tries to take into account both independence and dependence between the targeted pairs x_i, y_i and non-pairs $x_i, y_j, j \neq i$ of the signals.

The above formulas have been derived and given for some general vectors $\mathbf{x}$ and $\mathbf{y}$ (which have the same dimensionality). In the experiments, they were replaced by the estimated source vectors $\mathbf{s}_x^*$ and $\mathbf{s}_y^*$. The targeted pairs are the corresponding components of these vectors. The expectations in (31) are estimated in the usual way by replacing them with the respective sample averages.

5. Experimental results

5.1. Artificially generated data

First, we present some experimental results for artificially generated data. Such data are useful in testing various methods, because the original source signals are known, enabling computation of performance or error measures and visual inspection of the quality of the results.

The original source signals were as follows:

$$\begin{aligned} s_1(t) &= n(t), \\ s_2(t) &= \sin(350t)\sin(60t), \\ s_3(t) &= \mathrm{triangular}(70t), \\ s_4(t) &= \sin(800t)\sin(80t), \\ s_5(t) &= \cos(400t) + 4\cos(60t). \end{aligned} \tag{34}$$

These five sources, comprising together the source vector $\mathbf{s}(t)$, have been depicted in the five subfigures on the left-hand side of Fig. 1. They have been adopted from Example 7.2 in [5]. Four of the source signals are actually deterministic for easier visual inspection of the results, while the first source signal $s_1(t) = n(t)$ is zero mean Gaussian white noise with variance 1.

The five subfigures on the right-hand side of Fig. 1 show the five related source signals $\mathbf{t}(t) = \mathbf{f}(\mathbf{s}(t))$. They were generated by applying the nonlinear transformation

$$f(s(t)) = [s(t)]^3 - 0.5[s(t)]^2 \tag{35}$$

Please cite this article as: J. Karhunen, T. Ukkonen, Extending ICA for finding jointly dependent components from two related data sets, Neurocomputing (2006), doi:10.1016/j.neucom.2006.10.144

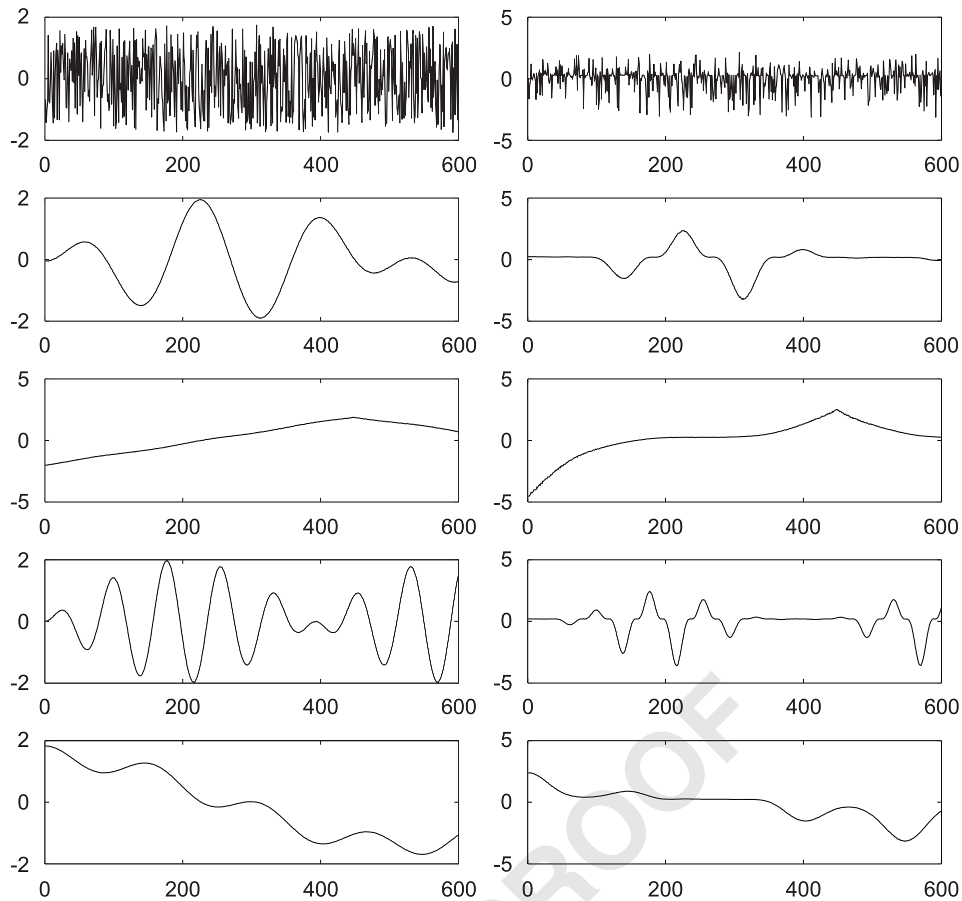


Fig. 1. The five source signals $\mathbf{s}(t)$ (left) and their nonlinear transformations $\mathbf{t}(t) = \mathbf{f}(\mathbf{s}(t))$ (right). The horizontal axis shows the sample (time) index t.

to the sources $s_i(t)$, that is, componentwise to the source vector $\mathbf{s}(t)$. The means of generated source signals $\mathbf{s}(t)$ and $\mathbf{t}(t)$ were set to zero and their variances were normalized to unity.

The first data set $\mathbf{x}(t)$ was obtained by mixing the original sources $\mathbf{s}(t)$ with a non-singular mixing matrix $\mathbf{A}$: $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$. The second, related data set $\mathbf{y}(t)$ was generated quite similarly by mixing the nonlinearly transformed sources $\mathbf{t}(t)$ with another non-singular mixing matrix $\mathbf{B}$, yielding $\mathbf{y}(t) = \mathbf{B}\mathbf{t}(t) = \mathbf{B}f(\mathbf{s}(t))$. The data vectors $\mathbf{x}(t)$ and $\mathbf{y}(t)$ have been depicted in Fig. 2.

Fig. 3 shows the jointly dependent sources estimated using the SVD (5) and (7): $\mathbf{s} = \mathbf{U}^{\mathrm{T}}\mathbf{x}$, $\mathbf{t} = \mathbf{V}^{\mathrm{T}}\mathbf{y}$. Fig. 4 depicts the sources provided by our first method (19)–(21). A visual inspection of the results suggests that the proposed Method 1 performs somewhat better than linear SVD in this example. In particular, it has managed to separate much better than SVD the fourth pair of signals in Fig. 1. This is confirmed by the values of the performance index (33). It is much higher, 33.6, for the Method 1 than the respective value 2.1 of the SVD-based basic method.

The results were qualitatively similar for the other nonlinearities and data sets tried in our simulations. A general conclusion of these experiments is that our Method 1 performs better than the SVD-based method. The difference in performance is small for 'easy' data sets of three source signals, but becomes significant for more difficult data sets have more sources. Method 2 did not perform in these experiments as well as Method 1, and therefore we have not shown the results for it.

5.2. Application to cryptographic data

In these experiments, we tried to find out the dependent corresponding components from texts and their encrypted versions. The texts were taken from the data sets made available by Project Gutenberg [13] in ASCII form. We picked up four books, with each ASCII letter at the same position in the books corresponding to one component of a four-dimensional vector. Thus, the first vector $\mathbf{x}(1)$ was the ASCII equivalent of the four letters which appeared first in each of the four books, the second vector $\mathbf{x}(1)$ contained the second letters of these books, and so on. There were 288,048 such vectors $\mathbf{x}(t)$ $(t = 1, 2, \ldots, 288, 048)$. The encrypted corresponding vectors $\mathbf{y}(t)$ $(t = 1, 2, \ldots, 288, 048)$ were ~~generating~~ by applying a 128-bit AES-ECB encoding [29] separately to each ASCII letter appended by zeros, so that each letter contained 128 bits. The encrypted 128 bit long numbers were transformed to floating point numbers having 96 bits, which we further

Please cite this article as: J. Karhunen, T. Ukkonen, Extending ICA for finding jointly dependent components from two related data sets, Neurocomputing (2006), doi:10.1016/j.neucom.2006.10.144

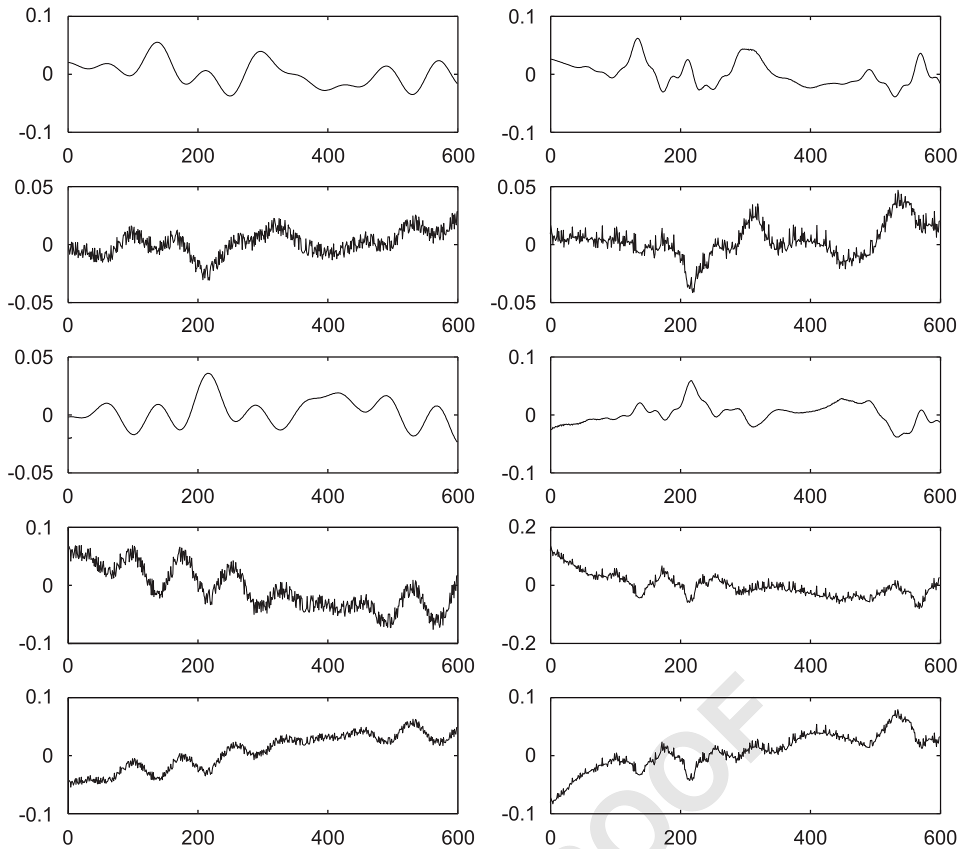


Fig. 2. The generated input data (mixtures) $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$ (left) and $\mathbf{y}(t) = \mathbf{B}\mathbf{t}(t)$ (right). Each subfigure shows one component of the vectors $\mathbf{x}(t)$ and $\mathbf{y}(t)$ as a function of time (sample index) t.

approximated by a 32-bit floating point number in our MATLAB experiments. The source signals were preprocessed so that their mean was zero and variance unity.

We tried to estimate the source vectors $\mathbf{s}$ and $\mathbf{t}$ as well as the mixing matrices $\mathbf{A}$ and $\mathbf{B}$ in the model (8) using the methods developed in Section 3. That is, we tried to determine these quantities so that the joint information between the corresponding components of $\mathbf{s}$ and $\mathbf{t}$ is maximized. After this, we computed the connectivity matrix $\mathbf{M}$, defined by

$$\mathbf{t} = \mathbf{M}\mathbf{s} = \mathbf{A}^{-1}\mathbf{B}\mathbf{s} \tag{36}$$

assuming that the mixing matrix $\mathbf{A}$ is square and of full rank and hence invertible. It is realistic to expect that the elements of the connectivity matrix $\mathbf{M}$ have higher absolute values when the encrypted message is strongly related to the original text.

Encryption aims at blurring or mixing the information contents of a message as much as possible, so that it cannot be identified any more from the encrypted version [29]. Thus, the goal of encryption is a kind of opposite to what ICA and BSS methods aim at. The nonlinearity used in AES encoding has been designed so that it is as far as possible from a linear function, making breaking of the encryption difficult, especially using customary linear statistical methods. Therefore the studied problem is difficult, and the results may highlight differences between various methods.

We tried several algorithms for this problem. A general conclusion on these experiments is that the performance of the suggested algorithms for estimating jointly dependent components gradually improves. Roughly speaking, they start to perform appropriately when the number of the elements in the vectors $\mathbf{x}(t)$ and $\mathbf{y}(t)$ nears $200{,}000$. As an example, consider Method 1. It could connect correctly two components of the vectors $\mathbf{x}(t)$ and $\mathbf{y}(t)$ when the number of sample vectors was $t = 180{,}000$, and all four components for $t = 262{,}440$ and $288{,}048$. Some other algorithms performed slightly better, being able to find out all the four dependent components already when $t = 180{,}000$. This holds especially for Method 2, which was on an average the best performing of the methods we tested in this problem. It was able to solve the problem faster and better than the SVD-based basic method.

For identifying the connected components, we used the following heuristics. Consider the absolute values $|m_{ij}|$ of the elements m_{ij} of the connectivity matrix $\mathbf{M}$. Find the element having the largest absolute value, and mark the corresponding components having the indices $i = i_{\max}$ and $j = j_{\max}$ connected. Continue the procedure by finding the element of the matrix $\mathbf{M}$ having the next largest absolute value and different indices $i \neq i_{\max}$ and $j \neq j_{\max}$, and connect

Please cite this article as: J. Karhunen, T. Ukkonen, Extending ICA for finding jointly dependent components from two related data sets, Neurocomputing (2006), doi:10.1016/j.neucom.2006.10.144

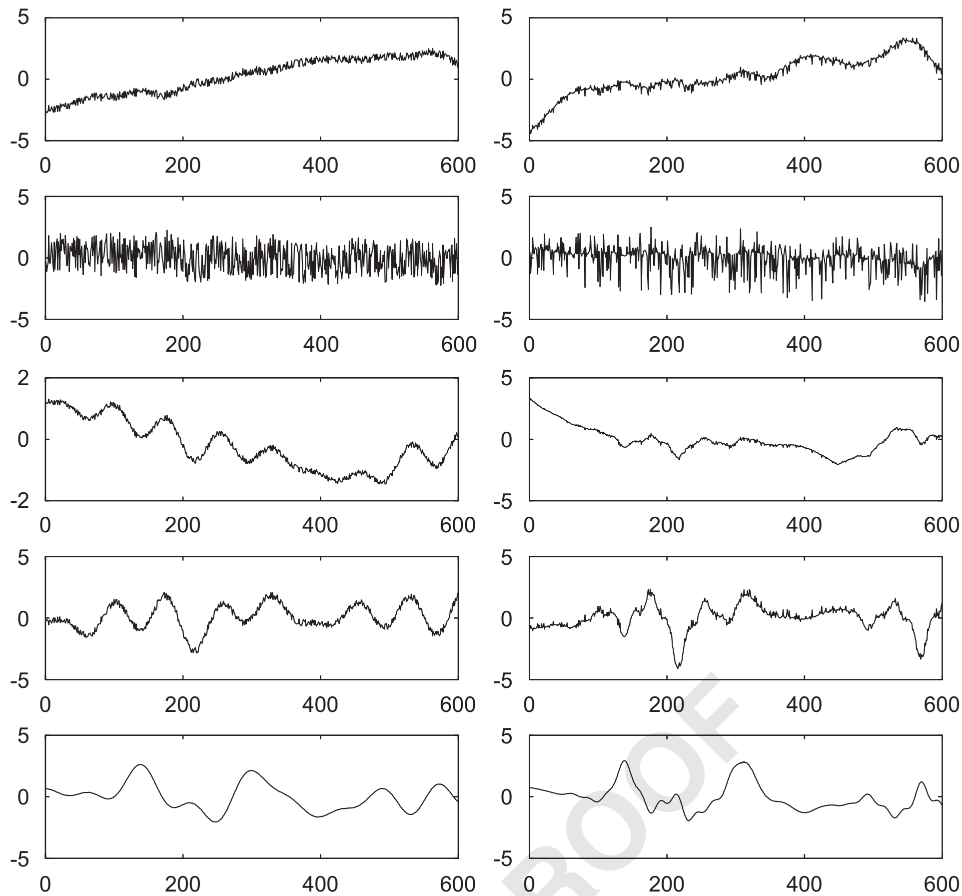


Fig. 3. The jointly dependent sources found using singular value decomposition.

the corresponding components. The procedure is continued until all the components of the vectors **s** and **t** have been connected. Of course, connecting takes place so that only one element on each row and column of the matrix **M** is selected. That is, each of the components of the source vector **s** is connected to one of the component of the source vector **t**.

When the number t of data vectors increased, not only the found connected components gradually became the correct ones. Also the absolute values of correct elements in the matrix **M** increased, and large erroneous values decreased. The methods using ICA for preprocessing were quite slow, because ICA was not usually able to find an independent group of source signals.

The final value of the matrix **M** for $t = 288,048$ data vectors is shown in (37) for the best performing Method 2. For clarity, we have omitted the common multiplying factor 10^{36} from the elements of **M**. From the results (37), one can without doubt deduce the correct jointly dependent corresponding source pairs. The corresponding largest elements of the matrix **M** on its each row and column have been boldfaced in (37).

$$\mathbf{M} = \begin{bmatrix} 0.138 & -0.225 & 0.939 & \mathbf{2.889} \\ -1.446 & 1.329 & \mathbf{2.269} & 1.039 \\ 0.330 & \mathbf{2.797} & -1.212 & -0.050 \\ \mathbf{2.719} & 0.428 & 1.410 & 0.514 \end{bmatrix}. \qquad (37)$$

6. Concluding remarks

In this paper, we have presented first result on some novel methods for finding mutually corresponding dependent components from two different but related data sets. Our methods generalize cross-correlation analysis based on SVD to take into account higher-order statistics in a similar manner as in ICA. The data model is rather simple, and could be generalized in several ways. A natural extension would be to allow a more flexible model than pairs of dependent components independent of other such pairs, see for example [18,19,22]. Experimental results demonstrating the usefulness of proposed methods have been presented both for artificially generated and realistic cryptography data.

Please cite this article as: J. Karhunen, T. Ukkonen, Extending ICA for finding jointly dependent components from two related data sets, Neurocomputing (2006), doi:10.1016/j.neucom.2006.10.144

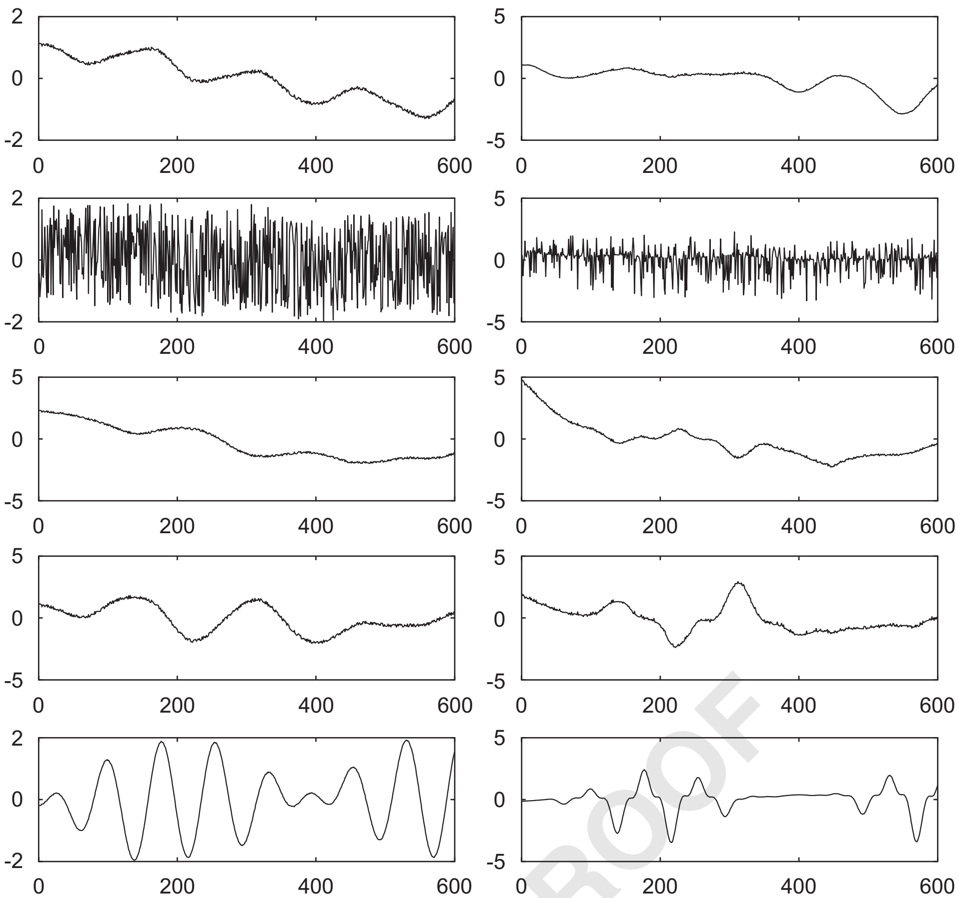


Fig. 4. The jointly dependent sources found using Method 1.

Acknowledgement

This research has been funded by the Academy of Finland under its Center of Excellence Programme.

References

[1] S. Achard, D. Pham, C. Jutten, Quadratic dependence measure for nonlinear blind sources separation, in: Proceedings of the Fourth International Symposium on Independent Component Analysis and Blind Signal Separation (ICA2003), Nara, Japan, April 2003, pp. 263–268.

[2] S. Akaho, Y. Kiuchi, S. Umeyama, MICA: multidimensional independent component analysis, in: Proceedings of the 1999 International Joint Conference on Neural Networks (IJCNN'99), Washington, DC, USA, July 1999, IEEE Press, pp. 927–932.

[3] F. Bach, M. Jordan, Kernel independent component analysis, J. Mach. Learn. Res. 3 (2002) 1–48.

[4] J.-F. Cardoso, The three easy routes to independent component analysis; contrasts and geometry, in: Proceedings of the Third International Conference on Independent Component Analysis and Blind Signal Separation (ICA2001), San Diego, CA, USA, December 2001, pp. 1–6.

[5] A. Cichocki, S.-I. Amari, Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications, Wiley, New York, 2002.

[6] P. Comon, G. Golub, Tracking a few extreme singular values and vectors in signal processing, Proc. IEEE 78 (1990) 1327–1343.

[7] K. Diamantaras, S.-Y. Kung, Principal Component Neural Networks: Theory and Applications, Wiley, New York, 1996.

[8] J. Eriksson, A. Kankainen, V. Koivunen, Novel characteristic function based criteria for ICA, in: Proceedings of the Third International Conference on Independent Component Analysis and Blind Signal Separation (ICA2001), San Diego, CA, USA, December 2001, pp. 108–113.

[9] J. Eriksson, V. Koivunen, Characteristic-function-based independent component analysis, Signal Process. 83 (2003) 2195–2208.

[10] C. Fyfe, P. Lai, ICA using kernel canonical correlation analysis, in: Proceedings of the Second International Workshop on Independent Component Analysis and Blind Signal Separation (ICA2000), Helsinki, Finland, June 2000, pp. 279–284.

[11] X. Giannakopoulos, J. Karhunen, E. Oja, Experimental comparison of neural algorithms for independent component analysis and blind separation, Int. J. Neural Syst. 9 (2) (1999) 651–656.

[12] A. Gretton, R. Herbrich, A. Smola, O. Bousquet, B. ~~Schlkopf,~~ Kernel methods for measuring independence, J. Mach. Learn. Res. 6 (2005) 2075–2129.

[13] Project Gutenberg, URL: ⟨http://www.promo.net/pg/index.html⟩.

[14] M. Hayes, Statistical Digital Signal Processing and Modeling, Wiley, New York, 1996.

[15] S. Haykin, Modern Filters, Macmillan, New York, 1989.

[16] S. Haykin, Neural Networks—A Comprehensive Foundation, second ed., Prentice-Hall, Englewood Cliffs, NJ, 1998.

[17] S. Haykin (Ed.), Unsupervised Adaptive Filtering, Vol. I: Blind Source Separation, Wiley, New York, 2000.

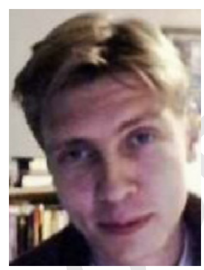
[18] A. Hyvärinen, P. Hoyer, Emergence of phase and shift invariant features by decomposition of natural images into independent feature subspaces, Neural Comput. 12 (7) (2000) 1705–1720.

Please cite this article as: J. Karhunen, T. Ukkonen, Extending ICA for finding jointly dependent components from two related data sets, Neurocomputing (2006), doi:10.1016/j.neucom.2006.10.144

[19] A. Hyvärinen, P. Hoyer, M. Inki, Topographic independent component analysis, Neural Comput. 13 (2001) 1527–1558.
[20] A. Hyvärinen, J. Karhunen, E. Oja, Independent Component Analysis, Wiley, New York, 2001.
[21] J. Karhunen, J. Joutsensalo, Generalizations of principal component analysis, optimization problems, and neural networks, Neural Networks 8 (4) (1995) 549–562.
[22] J. Karhunen, F. Meinecke, H. Valpola, A. Ziehe, Final technical report on BSS models and methods for non-independent BSS, Technical Report Deliverable D21, European Joint Project BLISS, 2003. URL: ⟨http://www.lis.inpg.fr/pages_perso/bliss/⟩.
[23] J. Karhunen, E. Oja, New methods for stochastic approximation of truncated Karhunen–Loeve expansions, in: Proceedings of the Sixth International Conference on Pattern Recognition, Munich, Germany, October 1982, pp. 550–553.
[24] J. Koetsier, D. MacDonald, D. Charles, C. Fyfe, Exploratory correlation analysis, in: Proceedings of the 10th European Symposium on Artificial Neural Networks (ESANN2002), Bruges, Belgium, April 2002, pp. 483–488.
[25] P. Lai, C. Fyfe, A neural implementation of canonical correlation analysis, Neural Networks 12 (1999) 1391–1397.
[26] K. Mardia, J. Kent, J. Bibby, Multivariate Analysis, Academic Press, London, 1979.
[27] E. Oja, Subspace Methods for Pattern Recognition, Research Studies Press, Letchworth, Hertfordshire, England, 1983.
[28] S. Roberts, R. Everson (Eds.), Independent Component Analysis: Principles and Practice, Wiley, New York, 2002.
[29] D. Stinson, Cryptography: Theory and Practice, Chapman & Hall, CRC Press, London, Boca Raton, FL, 2002.
[30] C. Therrien, Discrete Random Signals and Statistical Signal Processing, Prentice-Hall, Englewood Cliffs, NJ, 1992.
[31] D. Tjostheim, Measures of dependence and tests of independence, Statistics 28 (1996) 249–284.
[32] K. Waheed, F. Salam, A data-derived quadratic independence measure for adaptive blind source recovery in practical applications, in: Proceedings of the 45th IEEE International Midwest Symposium on Circuits and Systems, vol. 3, Tulsa, Oklahoma, USA, August 2002, pp. 473–476.

Juha Karhunen received the D.Sc. (Tech.) degree from Helsinki University of Technology in 1984. Since 1976, he has been in the Laboratory of Computer and Information Science at Helsinki University of Technology, Espoo, Finland, where he became a Professor in Computer Science in 1999 (specialization area: neural networks and signal processing). He belongs to the Adaptive Informatics Research Centre in the laboratory, which has been selected by the Academy of Finland as one of the Centers of excellence in research in Finland.

His current research interests include unsupervised variational Bayesian learning, independent component analysis, blind source separation, and their applications. Prof. Karhunen has published about 100 conference and journal papers on these topics, and given invited talks in international conferences. He is a co-author of the popular textbook and monograph A. Hyvärinen, J. Karhunen, and E. Oja, "Independent Component Analysis", Wiley 2000. He is a senior member of IEEE, and has earlier been a member of the editorial board in the journals Neurocomputing and Neural Processing Letters.



Tomas Ukkonen has received his M.Sc. (Tech.) degree in 2006 from the Department of Computer Science and Engineering, Helsinki University of Technology (HUT), Espoo, Finland. Before his graduation he worked in the Finnish Geodetic Institute at the Department of Geoinformatics and Cartography of HUT. His current interests include Bayesian methods, independent component analysis, analytical CRM, and spatial statistics. He is currently employed as Data Analysts in the Itella TGM Corp.

Please cite this article as: J. Karhunen, T. Ukkonen, Extending ICA for finding jointly dependent components from two related data sets, Neurocomputing (2006), doi:10.1016/j.neucom.2006.10.144