Principal and Independent Components in Neural Networks - Recent Developments

E. Oja, J. Karhunen, L. Wang, and R. Vigario

Laboratory of Computer and Information Science
Helsinki University of Technology
Rakentajanaukio 2C, 02150 Espoo, Finland
Email: Erkki.Oja@hut.fi

Abstract

Nonlinear extensions of one-unit and multi-unit Principal Component Analysis (PCA) neural networks, introduced earlier by the authors, are reviewed. The networks and their nonlinear Hebbian learning rules are related to other signal expansions like Projection Pursuit (PP) and Independent Component Analysis (ICA). Separation results for mixtures of real world signals and images are given.

1 Introduction

Principal Component Analysis (PCA) is a widely used technique in data analysis. Mathematically, it is defined as follows: let $\mathbf{C} = \mathrm{E}\{\mathbf{x}\mathbf{x}^T\}$ be the covariance matrix of L-dimensional zero mean input data vectors $\mathbf{x}$. The ith principal component of $\mathbf{x}$ is $\mathbf{x}^T\mathbf{c}(i)$, where $\mathbf{c}(i)$ is the normalized eigenvector of $\mathbf{C}$ corresponding to the ith largest eigenvalue $\lambda(i)$. The subspace spanned by the principal eigenvectors $\mathbf{c}(1), \ldots, \mathbf{c}(M)$ $(M < L)$ is called the PCA subspace (of dimensionality M). PCA is used in many applications because of its optimality properties in data compression and information representation [19].

It is now well-known that PCA can be realized neurally in various ways. A Principal Component Analysis (PCA) network [26] is a one-layer feedforward neural network which is able to extract the principal components of the stream of input vectors. Typically Hebbian type learning rules are used based on the one-unit learning algorithm originally proposed by one of the authors in [24]. Many different versions and extensions of this basic algorithm have been proposed during the recent years; for reviews and introductions, see e.g. [29, 6, 13]. PCA networks are useful in signal characterization, optimal feature extraction, and data compression. PCA is also the basis of subspace classifiers that have been used e.g. in speech and texture classification [27].

In the field of neural networks, there has been growing interest in *nonlinear extensions* of the PCA. One of the present authors generalized the one-unit and multi-unit PCA learning rules to nonlinear versions in [28]. Recently, some authors have discussed the relation of these rules to *Projection Pursuit* (PP) [12, 36]. Another interesting development is the relation of neural networks to *Independent Component Analysis* (ICA) [15] and the *source separation problem.*

The source separation problem [15, 8], to be discussed in more detail in Section 4, is important in certain applications of signal processing and communications. The problem is to find a set of source signals, when only linear combinations or mixtures of them are given. A neural solution is a network, to be discussed in Section 5, which takes the mixture signals as inputs and produces approximations to the independent source signals as the outputs. A prerequisite for separating the sources is that the output signals must be mutually uncorrelated, a condition which already the principal components satisfy. However, in addition to the uncorrelatedness, the output signals must be mutually independent (or as independent as possible). This leads in a natural way from PCA to Independent Component Analysis. A precise definition of ICA together with a detailed analysis is given in a recent fundamental paper by Comon [8].

Generally, source separation and ICA require higher than second-order statistics, not provided by standard PCA. Such higher-order statistics can be incorporated into the computations either explicitly or by using suitable nonlinearities. We favor the latter approach because it suits better to a neural network environment. Recently, Karhunen et al [20] proposed the multilayer ICA network based on linear and nonlinear PCA layers as a neural realization of ICA.

These developments will be reviewed here. As the starting point, we use the nonlinear constrained Hebbian rules introduced by one the authors in Oja et al [28]. These will be reviewed in Section 2 for the one-unit case and Section 3 for the multi-unit case, and they are related to optimization and Exploratory Projection Pursuit. Also, the bigradient learning rule of Wang et al [35] is reviewed here. In Section 4, the source separation problem and the ICA expansion are reviewed and a two stage solution is analyzed in which the first stage normalizes the input data by whitening or sphering, and the second stage performs nonlinear PCA. This structure is closely related to both multidimensional PP and ICA. Section 5 is a review of the full ICA network, and applications to speech and image separation are given in Section 6.

2 Nonlinear Hebbian learning: the one-unit case

2.1 Learning rule for one neuron

Consider first a single artificial neuron receiving an L-dimensional input vector $\mathbf{x}$ [28]. The neuron is trying to adapt its weight vector $\mathbf{w}$ so that a function $E\{f(\mathbf{w}^T\mathbf{x})\}$ is maximized, where E is the expectation with respect to the (unknown) density of $\mathbf{x}$ and $f(.)$ is a continuous objective function. In order to prevent the objective function from getting arbitrarily large values, the problem must be constrained by setting the Euclidean norm of $\mathbf{w}$ constant.

The only available information on $\mathbf{x}$ is the sequence of training samples $\mathbf{x}_1, \mathbf{x}_2, \ldots$ during a learning period. A *constrained gradient ascent learning rule* based on sample functions is then

$$\tilde{\mathbf{w}}_{k+1} = \mathbf{w}_k + \alpha_k \frac{d}{d\mathbf{w}_k} f(\mathbf{w}_k^T \mathbf{x}_k), \ \mathbf{w}_{k+1} = \frac{\tilde{\mathbf{w}}_{k+1}}{||\tilde{\mathbf{w}}_{k+1}||} \tag{1}$$

where α_k is the learning rate, usually a sequence of positive numbers decreasing slowly to zero; for details on the learning rates, see [25]. The first equation in eq. (1) is a pure gradient ascent step, and the other is a normalizing step. Denoting the derivative of $f(y)$ by $g(y)$, the gradient with respect to $\mathbf{w}$ becomes

$$\frac{d}{d\mathbf{w}} f(\mathbf{w}^T \mathbf{x}) = g(\mathbf{w}^T \mathbf{x})\mathbf{x}. \tag{2}$$

For small values of α_k, eq. (1) can be approximated by

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k (I - \mathbf{w}_k \mathbf{w}_k^T)\mathbf{x}_k g(\mathbf{w}_k^T \mathbf{x}_k) \tag{3}$$

in which terms proportional to α_k^2 have been dropped [28].

Depending on the function $f(\mathbf{w}^T\mathbf{x})$, several cases are covered by this formalism. For instance, if the objective function is

$$f(y) = lncosh(y), \ y = \mathbf{w}^T \mathbf{x}, \tag{4}$$

then $g(y) = (d/dy)f(y) = tanh(y)$, the activation function commonly used in Perceptron networks. Several possible cost functions are listed in [28].

2.2 Connection to Exploratory Projection Pursuit

The Exploratory Projection Pursuit (PP) technique was introduced by Friedman and Tukey [10] for visualization purposes, especially clustering. The purpose is to find directions $\mathbf{w}'$ in the input space that are somehow interesting, for instance, such that projections $y = \mathbf{w}'^T\mathbf{x}$ of the data on the direction $\mathbf{w}'$ is multimodal or in general non-Gaussian. This is achieved by minimizing or maximizing some suitable criterion or quality function. An especially useful quality function is the *fourth cumulant*, also called *kurtosis*, of the projection, defined as $E\{y^4\} - 3(E\{y^2\})^2$. In general, the kurtosis of a random variable can attain positive or negative values, with the kurtosis of a Gaussian zero-mean variable being equal to zero. A high kurtosis (a positive value) signifies a peaked unimodal density, sharper than a Gaussian, while a low kurtosis (a high negative value) indicates multimodality. Therefore a direction that has low kurtosis is often interesting for visualization and clustering; see e.g. [36]. Good review papers on the PP technique are given by Huber [14] and by Friedman [11].

The constrained Hebbian learning rule of Section 2.1 was used by Fyfe et al [12] in exploring neural learning for exploratory projection pursuit. In case $f(y)$ is a polynomial, the neuron weight will find the direction in the input space that will maximize a higher moment: for $g(y) = y^2$ and $f(y) = 1/3y^3$, the third moment $E\{y^3\}$ will be maximized. For $g(y) = y^3$, the fourth moment $E\{y^4\}$ will be maximized.

As noted above, multimodality is revealed by a low kurtosis, and hence the objective function in maximization should rather be $f(y) = -y^4$. There is an interesting relationship between this and the objective function (4): expanding in Taylor series yields

$$lncosh(y) = 1/2y^2 - 1/12y^4 + 1/45y^6 + O(y^8), \tag{5}$$

hence the function that the neuron is maximizing is

$$\begin{aligned} E\{lncosh(y)\} &= 1/2E\{y^2\} - 1/12E\{y^4\} + 1/45E\{y^6\} + E\{O(y^8)\} && (6) \\ &= 1/2E\{(\mathbf{w}^T\mathbf{x})^2\} - 1/12E\{(\mathbf{w}^T\mathbf{x})^4\} && (7) \\ &+ 1/45E\{(\mathbf{w}^T\mathbf{x})^6\} + E\{O[(\mathbf{w}^T\mathbf{x})^8]\}. && (8) \end{aligned}$$

The term $1/2E\{(\mathbf{w}^T\mathbf{x})^2\}$ is dominating in the expression especially for small values of $y = \mathbf{w}^T\mathbf{x}$, and so the neuron with this activation function will have a tendency to learn the direction with maximal variance – i.e., the principal component of the input data. However, let us explicitly prevent this by *sphering* the data first (see [12]): by linear preprocessing, the covariance $\mathbf{C} = E\{\mathbf{x}\mathbf{x}^T\}$ can be made equal to the unit matrix. Let us assume that the input vector has been sphered: $\mathbf{C} = \mathbf{I}$. Then the first term in the above expression becomes $1/2E\{(\mathbf{w}^T\mathbf{x})^2\} = 1/2\mathbf{w}^T\mathbf{C}\mathbf{w} = 1/2\mathbf{w}^T\mathbf{w} = 1/2$, since the weight vector is normalized to unit length. Now this term has no influence in the minimization. Also note that due to sphering, $y = \mathbf{w}^T\mathbf{x}$ is effectively normalized because its variance is equal to one, and we can assume that y^4 is larger than the higher moments with a relatively high probability. Therefore, it is the second term, $-1/12E\{(\mathbf{w}^T\mathbf{x})^4\}$ that is dominating, and kurtosis is minimized at least approximately.

Example 2.1. As a simple example of these quality functions, consider a two-peaked Gaussian mixture. Let us assume that the two peaks (mean values of the two Gaussian components) are symmetrically located around the origin at points $\pm m$, and both components have equal variance σ^2. Then the random variable y with this density has mean value zero and variance equal to $\sigma^2 + m^2$. Sphering means now that the variance is constrained to unity: therefore, assume that $\sigma^2 + m^2 = 1$. When $m = 0$ and there is only one peak at the origin, then $\sigma = 1$ and the density is the standard $N(0, 1)$ Gaussian. When m grows, σ decreases until at $m = 1$ the density consists of two zero-width atoms at the points ± 1.

The kurtosis of the $N(0, 1)$ density is zero, and it decreases as the two peaks separate. The lower curve (denoted "1") in Fig. 1 is the kurtosis as the function of m. The upper curve (denoted "2") is the graph of $E\{lncosh(10y)\}$ (normalized so that the value at $m = 0$ is zero; any additive constant has no effect on the minimization problem). The derivative of this objective function is the sharp sigmoid $tanh(10y)$. It is seen that this objective function also measures deviation from normality much in the same way as the kurtosis.

Figure 1: The two quality functions: kurtosis (curve 1) and the expectation of function $lncosh(y)$ (curve 2) for a two-peaked density, as functions of the distance of the peaks from the origin

Sudjianto and Hassoun [32] made the interesting observation that the $tanh(\beta y)$ function, with a suitable parameter β, follows very closely the function $2\phi(y) - 1$ where $\phi(y)$ is the cumulative distribution function of the Gaussian density. Thus $z = tanh(\beta \mathbf{w}^T \mathbf{x})$ will have an almost *uniform density* if $\mathbf{w}^T\mathbf{x}$ is Gaussian. In Projection Pursuit, such a direction should be avoided. Friedman's index in PP (see Comon [8]) is based precisely on the deviation of z from the uniform density.

In the nonlinear Hebbian learning rule this same effect can now be obtained by using the objective function $f(\mathbf{w}^T\mathbf{x}) = tanh^2(\mathbf{w}^T\mathbf{x})$, which, when maximized, would force the output of the neuron z close to its saturation values ± 1 - hence, make the density of z very non-uniform. This was one of the criteria proposed in [28], and the derivative to be used in the learning rule (3) is then $g(y) = tanh(y)(1 - tanh^2(y))$. The function $tanh^2(y)$ also has the effect of minimizing kurtosis because $tanh^2(y) = y^2 - 2/3y^4 + 34/90y^6 - ...$, and in the expectation the fourth order term will again dominate assuming the second order term is constant. However, the domination is not as strong as in the $lncosh(y)$ function.

3 Nonlinear Hebbian learning: the multi-unit case

3.1 The learning rules

A straightforward generalization of eq. (3) to the case of many identical parallel units is possible. Consider a one-layer neural network of M parallel units. Each unit i has the same L-element input vextor $\mathbf{x}$ and its own L-dimensional weight vector $\mathbf{w}(i)$, which together comprise an $L \times M$ *weight matrix* $\mathbf{W} = [\mathbf{w}(1)...\mathbf{w}(M)]$. Following Oja et al [28], the following learning rules are generalizations of the linear PCA learning rule:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \alpha_k(\mathbf{I} - \mathbf{W}_k\mathbf{W}_k^T)\mathbf{x}_k g(\mathbf{x}_k^T\mathbf{W}_k), \tag{9}$$
$$\mathbf{W}_{k+1} = \mathbf{W}_k + \alpha_k[\mathbf{x}_k g(\mathbf{x}_k^T\mathbf{W}_k) - \mathbf{W}_k g(\mathbf{W}_k^T\mathbf{x}_k) g(\mathbf{x}_k^T\mathbf{W}_k)]. \tag{10}$$

The nonlinear vector function $g(\mathbf{x}_k^T\mathbf{W}_k)$ must be understood element by element. These learning rules were related to optimization criteria by Karhunen and Joutsensalo [17]. Let us consider each algorithm in turn, and also introduce a third possibility closely related to the above, the bigradient algorithm. In this section, the learning rules are discussed from a general optimization point of view, and they will be consequently applied to the Independent Component Analysis problem in Section 4.

3.1.1 Robust PCA learning rule

Consider first algorithm (9). It was shown by Karhunen and Joutsensalo [17] that it is a stochastic approximation algorithm for the minimization problems

$$J_0(\mathbf{w}(i)) = E\{f(\mathbf{x}^T\mathbf{w}(i))\} + \Sigma_{j=1}^{k}\lambda_{ij}(\mathbf{w}(i)^T\mathbf{w}(j) - \delta_{ij}),\ i = 1, ..., M, \tag{11}$$

where λ_{ij} are Lagrangian coefficients. Each neuron is maximizing its objective function under the constraint that the weight vectors are mutually orthonormal.

The use of a generalized *non-quadratic* criterion function $f(y)$ in (11) as a "robust Principal Components" criterion can be justified from two viewpoints. First, if $f(y)$ grows less than quadratically, (11) is more robust than standard variances against outliers and long-tailed noise. Examples of such robust cost functions are $f(y) = lncosh(y)$ and $f(y) = |y|$; see also [6, 17]. Second, if $f(y)$ is non-quadratic, (11) introduces higher-order statistics into computations, while the variance criterion $f(y) = y^2$ would measure second-order statistics (covariances) only. This is sufficient for Gaussian data, but usually not sufficient for inputs with non-Gaussian densities [19]. For example, if $f(y) = y^4$ (suitably scaled), the first term in (11) measures the kurtosis of the input data provided that the data vectors are whitened (sphered) prior to inputting them to the network, as explained above in Section 2.2.

The algorithm (9) is useful in clustering, exploratory projection pursuit, and in extraction of robust features from the input data. In principle, it could be applied to the respective generalized MCA problem, namely minimization of the criterion J_0 in (11) with the same constraints, simply by changing the gain parameter α_k negative. Regrettably, (9) is then not stable. Therefore, in the following we consider a different approach, where the orthonormality constraints are not taken into account via Lagrange multipliers, but by minimizing another criterion simultaneously.

3.1.2 The bigradient algorithm

In the context of PCA type neural networks, the basic idea of optimizing two criteria simultaneously has been introduced earlier in [5] for a somewhat different single neuron problem. First, the criterion

$$J_1(\mathbf{w}(i)) = E\{f(\mathbf{x}^T\mathbf{w}(i))\} \tag{12}$$

which is the first part in the criterion (11) must be maximized or minimized by the i-th neuron. Second, the orthonormality constraints can be satisfied by minimizing for the i-th weight vector the criterion or *penalty function*

$$J_2(\mathbf{w}(i)) = \frac{1}{2}(1-\mathbf{w}(i)^T\mathbf{w}(i))^2 + \frac{1}{2}\sum_{j=1,j\neq i}^{M}(\mathbf{w}(i)^T\mathbf{w}(j))^2. \tag{13}$$

Clearly, the minimum is achieved when all the squared quantities are zero, which happens if and only if the weight vector $\mathbf{w}(i)$ has unit norm and is orthogonal to the weight vectors of all the other neurons.

To proceed to a stochastic gradient ascent or descent algorithm that uses a sample sequence of inputs $\mathbf{x}_k$, the instantaneous gradients of the two criteria are needed. Omitting the expectation, the instantaneous gradient of (12) with respect to $\mathbf{w}(i)$ is $g(\mathbf{x}^T\mathbf{w}(i))\mathbf{x} = g(y(i))\mathbf{x}$. Similarly, the gradient of $J_2(\mathbf{w}(i))$ with respect to $\mathbf{w}(i)$, at the k-th learning step, is

$$\mathbf{u}_k(i) = \frac{\partial J_2(\mathbf{w}_k(i))}{\partial \mathbf{w}_k(i)} = -\mathbf{w}_k(i)(1-\mathbf{w}_k(i)^T\mathbf{w}_k(i)) + \sum_{j\neq i}^{M}(\mathbf{w}_k(j)^T\mathbf{w}_k(i))\mathbf{w}_k(j). \tag{14}$$

These gradients can be used for simultaneous optimization of $J_1(\mathbf{w}(i))$ and $J_2(\mathbf{w}(i))$ as follows:

1. For $i = 1,\ldots,M$, compute a new weight vector estimate for optimizing the criterion $J_1(\mathbf{w}(i))$:

   $$\mathbf{w}^*_{k+1}(i) = \mathbf{w}_k(i) + \mu_k g(y_k(i))\mathbf{x}_k. \tag{15}$$

   Here μ_k is chosen *positive in maximizing* $J_1(\mathbf{w}(i))$ and *negative in minimizing* it.

2. Normalize the vectors $\mathbf{w}^*_{k+1}(i)$, $i = 1,\ldots,M$, by trying to minimize the criterion $J_2(\mathbf{w}(i))$:

   $$\mathbf{w}_{k+1}(i) = \mathbf{w}^*_{k+1}(i) - \gamma_k \mathbf{u}^*_{k+1}(i). \tag{16}$$

   Here γ_k is another gain parameter that is always positive, and $\mathbf{u}^*_{k+1}(i)$ is the value of the gradient (14) computed using the vectors $\mathbf{w}^*_{k+1}(i)$ obtained from (15).

The above bigradient type algorithm is iterated until the weight vectors have converged with sufficient accuracy. If necessary, the sample vectors $\mathbf{x}_k$ can be used several times, cyclically or by random selection, for achieving convergence.

Usually the gain parameters μ_k and γ_k are relatively small for stability reasons. Therefore, we can form a simpler approximative bigradient algorithm by inserting the expression of $\mathbf{w}^*_{k+1}(i)$ from (15) into (16), and retaining only the terms that are first order with respect to μ_k and γ_k. This yields the algorithm

$$\mathbf{w}_{k+1}(i) = \mathbf{w}_k(i) + \mu_k g(y_k(i))\mathbf{x}_k - \gamma_k \mathbf{u}_k(i). \tag{17}$$

For the symmetric constraints (13), the learning rule (17) can be expressed conveniently in the matrix form

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k \mathbf{x}_k \mathbf{g}(\mathbf{y}_k^T) + \gamma_k \mathbf{W}_k(\mathbf{I} - \mathbf{W}_k^T \mathbf{W}_k). \tag{18}$$

This is our final bigradient algorithm. In practice, μ_k is a small constant or tends slowly towards zero with time k, and a suitable value of γ_k is in the range $0.5 \ldots 1$. This algorithm is suitable both for minimization and for maximization of the criterion J_1 in eq. (12) under the orthonormality constraints imposed by the penalty function J_2 in eq. (13).

3.1.3 The nonlinear PCA expansion

Finally, consider the learning rule in eq. (10). It was shown by Karhunen and Joutsensalo in [17] that it approximates a stochastic gradient descent algorithm for the minimization problem

$$J_3(\mathbf{W}) = E\{\|\mathbf{x} - \mathbf{W} g(\mathbf{W}^T \mathbf{x})\|^2\}. \tag{19}$$

This is a least mean square criterion in which the input vector $\mathbf{x}$ is represented by a nonlinear expansion of the weight vectors; for details, see [17].

Since this criterion is not explicitly minimizing or maximizing any quality criteria $E\{f(\mathbf{x}^T \mathbf{w}(i))\}$ for the neuron weights, it may be difficult to see whether it has any connections to our basic problems of independent component analysis or source separation. To illustrate that such a connection exists, consider the following example.

Example 3.1. Assume that $\mathbf{x}$ is 2-dimensional and has a uniform density in a square that is *not* aligned with the coordinate axes $x(1), x(2)$, according to Fig. 2. Then it is easily shown that the elements $x(1), x(2)$ are uncorrelated and have equal variances, thus, $\mathbf{x}$ is sphered. However, the elements are not independent. The problem is to find a rotation $\mathbf{y} = \mathbf{R}\mathbf{x}$ of $\mathbf{x}$ such that the elements of the rotated vector $\mathbf{y}$ are statistically independent. It is obvious from Fig. 2 that the elements of $\mathbf{y}$ must be aligned with the orientation of the square, because then and only then the density is separable into two marginal densities: $p(\mathbf{y}) = p_1(y(1))p_2(y(2))$, with both p_1 and p_2 one-dimensional uniform densities.

The minimal solution to criterion (19), with $\mathbf{W} = [\mathbf{w}(1)\mathbf{w}(2)]$ a 2×2 matrix and $g(.)$ a suitable nonlinearity, provides now such a rotation. This can be seen as follows. Assume that g is a very sharp sigmoid, e.g. $g(y) = tanh(10y)$. Then it can

Figure 2: The uniform density. Original coordinates: $x(1), x(2)$. Rotated independent coordinates: $y(1), y(2)$

be approximated by the *sign* function, and the term $\mathbf{W}g(\mathbf{W}^T\mathbf{x})$ in criterion (19) becomes

$$\begin{aligned}\mathbf{W}g(\mathbf{W}^T\mathbf{x}) &= \mathbf{w}(1)g(\mathbf{w}(1)^T\mathbf{x}) + \mathbf{w}(2)g(\mathbf{w}(2)^T\mathbf{x}) \qquad (20)\\ &\approx \mathbf{w}(1)sign(\mathbf{w}(1)^T\mathbf{x}) + \mathbf{w}(2)sign(\mathbf{w}(1)^T\mathbf{x}). \qquad (21)\end{aligned}$$

Thus according to (19), each $\mathbf{x}$ should be optimally represented by a point $\pm\mathbf{w}(1) \pm \mathbf{w}(2)$, with the signs depending on the angles between $\mathbf{x}$ and the basis vectors. Each choice of the two orthogonal basis vectors divides the square of Fig. 2 into four quadrants, and by the criterion (19), all the points in a given quadrant must be represented in the least mean square sense by just one point; e.g. in the first quadrant, where the angles between $\mathbf{x}$ and the basis vectors are positive, by the point $\mathbf{w}(1) + \mathbf{w}(2)$. From Fig. 2, it can be seen that the optimal fit is obtained when the basis vectors are aligned with the axes $y(1), y(2)$, and the point $\mathbf{w}(1) + \mathbf{w}(2)$ is the center of the smaller square bordered by the positive $y(1), y(2)$ axes. In the adaptive algorithm (10) minimizing (19), this is indeed the solution $\mathbf{W}$ as is shown by simulations using the density of Fig. 2.

In fact, the marginal densities of $x(1)$ and $x(2)$ in Fig. 2 have a triangular shape, having negative kurtosis; the kurtosis is minimized when the densities become uniform (as non-Gaussian as possible), and this happens when $x(1), x(2)$ are rotated to $y(1), y(2)$.

4 Relation of nonlinear PCA to the source separation problem and ICA

In source separation for linear memoryless channels and the related technique of Independent Component Analysis (ICA), the following basic data model is usually assumed (see e.g. [15, 8, 6]). There are M scalar-valued signals $s_k(1), ..., s_k(M)$ indexed by an index k; if k is one-dimensional, then it usually signifies discrete time. We assume that the signals have zero mean and they are mutually statistically independent. Examples of such signals are shown in Fig. 4. More concretely, the signals could be sampled speech waveforms, as is the case in one of the examples in Section 6; for different speakers the signals would be independent at least approximately. If the index k is two-dimensional, the signals are interpreted as images as in Fig. 5.

We assume that the original signals are *unobservable*, and all we have are a set of noisy linear mixtures $x_k(1), ..., x_k(L)$, with

$$x_k(j) = \sum_{i=1}^{M} s_k(i) a(ij) + n_k(j). \tag{22}$$

The coefficients $a(ij)$ are *unknown*. However, we assume that the mixtures are all different in the sense that the M vectors of mixture coefficients $\mathbf{a}(i) = (a(i1), ..., a(iL))^T$ are linearly independent. Such mixtures arise in several practical situations like speech separation or antenna array processing.

Denote by $\mathbf{x}_k = (x_k(1)....x_k(L))^T$ the L-dimensional data vector at discrete time (or point) k. By eq. (22), we can write

$$\mathbf{x}_k = \mathbf{A}\mathbf{s}_k + \mathbf{n}_k = \sum_{i=1}^{M} s_k(i)\mathbf{a}(i) + \mathbf{n}_k, \tag{23}$$

where in the M-vector $\mathbf{s}_k = (s_k(1), \ldots, s_k(M))^T$, the element $s_k(i)$ denotes the ith source signal (independent component) at time k, $\mathbf{A} = (\mathbf{a}(1), \ldots, \mathbf{a}(M))$ is the $L \times M$ 'mixing matrix' whose columns $\mathbf{a}(i)$ are the basis vectors of ICA, and $\mathbf{n}_k$ denotes the vector of noise components.

The following assumptions are typically made [4, 21]:

1. $\mathbf{A}$ is an unknown constant matrix with full column rank. Thus the number of sources $M \leq L$. Usually M is assumed to be known in advance.

2. The source signals (coefficients) $s_k(i)$ are mutually statistically independent at each time instant k, or as independent as possible.

3. Each source signal sequence $\{s_k(i)\}$ is a stationary zero-mean stochastic process. Only one source signal is allowed to be Gaussian. (In practice, the more non-Gaussian the signals are, the easier is the separation. Separation of several Gaussians is impossible because any linear mixture is just another Gaussian.)

Note that very little prior information is assumed on the matrix $\mathbf{A}$. Therefore, the strong independence assumptions are required to fix the ICA expansion (23).

Usually the basis vectors $\mathbf{a}(i)$ of ICA are not mutually orthogonal. The noise term $\mathbf{n}_k$ is often omitted from (23), because in fact it can be incorporated in the sum as one of the source signals.

The source separation problem [15, 4, 21] is now to find an $M \times L$ separating matrix $\mathbf{B}$ so that the M-vector

$$\mathbf{y}_k = \mathbf{B}\mathbf{x}_k \tag{24}$$

is an estimate $\mathbf{y}_k = \hat{\mathbf{s}}_k$ of the original independent source signals.

Under the assumptions made above, it turns out that the estimate $\hat{s}_k(i)$ of the ith source signal may appear in any component $y_k(j)$ of $\mathbf{y}_k$. It is also impossible to determine the amplitudes of the source signals $s_k(i)$ from the model (23) without additional assumptions.

Various more or less heuristic approaches have been proposed for achieving separation. The learning algorithms are constructed in such a way that they should satisfy some kind of independence condition after convergence. An example is the seminal Herault-Jutten (HJ) algorithm [15], which uses a neural-like network structure. This algorithm is simple and elegant, but may fail in separating more than two independent sources.

Most of the approaches dealing with signal separation and ICA are non-neural, and are based on some batch type or adaptive signal processing algorithm. However, some of these adaptive algorithms, such as the HJ algorithm and its modifications as well as the PFS/EASI algorithm proposed by Cardoso and Laheld [4, 21], can be interpreted as learning algorithms of a neural network. Quite recently, some authors [2, 9] have derived unsupervised "neural" learning rules from information-theoretic measures. The resulting algorithms show good separation performance, but are not truly realizable in neural networks because they are based on relatively complicated numerical operations, requiring for example the inversion of a matrix.

Several separation algorithms utilize the fact that if the data vectors $\mathbf{x}_k$ are first preprocessed by whitening (sphering) them, i.e., if $E\{\mathbf{x}_k\mathbf{x}_k^T\} = \mathbf{I}$, then the separating matrix $\mathbf{B}$ in (24) becomes orthogonal: $\mathbf{B}\mathbf{B}^T = \mathbf{I}$. This can be seen by writing $E\{\mathbf{y}_k\mathbf{y}_k^T\} = \mathbf{B}E\{\mathbf{x}_k\mathbf{x}_k^T\}\mathbf{B}^T = \mathbf{B}\mathbf{B}^T$, and this must be at least diagonal because we require that the elements of y_k are zero-mean and statistically independent. There is no way to find the energies of the individual source signals $s_k(i)$, as they are absorbed in the mixing coefficients, and therefore it usually suffices to find vectors y_k that have unit covariances. In this case, $\mathbf{B}$ is an orthogonal matrix.

It is not easy to verify the independence condition exactly in practice because the involved probability densities are unknown. Therefore, approximating *contrast functions* which are maximized for a separating matrix have been introduced [8]. Even these often lead to relatively intensive batch type computations. However, for prewhitened input vectors it can be shown [23] that the simpler contrast function, the sum of absolute values of the fourth order cumulants (kurtoses)

$$J_{kurt}(\mathbf{y}) = \sum_{i=1}^{M} \mid \mathrm{cum}(y(i)^4) \mid \; = \; \sum_{i=1}^{M} \mid E\{y(i)^4\} - 3E^2\{y(i)^2\} \mid \tag{25}$$

is maximized by a separating matrix $\mathbf{B}$ under certain conditions. This provides an interesting connection between the techiques of ICA and Projection Pursuit, in which kurtosis also can play an important role as shown in Section 2.

The following result (in a slightly different formulation) was proven by Moreau and Macchi [23]:

Theorem 1. Assume that the fourth order cumulants of all the original statistically independent source signals $s(i)$ have the same sign, and consider vectors $\mathbf{y} = \mathbf{Hs}$ with $\mathbf{s} = (s(1), ..., s(M))^T$ the source signal vector and $\mathbf{H}$ an invertible matrix. Assume that $\mathbf{y}$ has been sphered, i.e. $E\{\mathbf{y}\mathbf{y}^T\} = \mathbf{I}$. Then the criterion (25) satisfies:

1. $J_{kurt}(\mathbf{y})$ is invariant by scale changes in any of the elements $y(i)$;
2. $J_{kurt}(\mathbf{y}) = J_{kurt}(\mathbf{Hs}) \leq J_{kurt}(\mathbf{s})$;
3. $J_{kurt}(\mathbf{y}) = J_{kurt}(\mathbf{Hs}) = J_{kurt}(\mathbf{s})$ if and only if $\mathbf{H} = \mathbf{\Lambda P}$, for some permutation matrix $\mathbf{P}$ and some diagonal matrix $\mathbf{\Lambda}$ whose diagonal elements are ± 1.

The central conclusion 3 means that among vectors of the form $\mathbf{y} = \mathbf{Hs}$, with vector $\mathbf{s}$ having independent components, the criterion (25) is indeed maximized when $\mathbf{y}$ is either $\mathbf{s}$ itself or a permutation of its elements, possibly with changed signs.

For sphered vectors it holds $\mathrm{E}\{y(i)^2\} = 1$ and $\mathrm{cum}(y(i)^4) = E\{y(i)^4\} - 3$. Thus (25) is maximized by *minimizing* the sum of the fourth moments $\sum_{i=1}^{M} \mathrm{E}\{y(i)^4\}$ for *negatively kurtotic* sources, and by *maximizing* $\sum_{i=1}^{M} \mathrm{E}\{y(i)^4\}$ for *positively kurtotic* sources. We use this criterion in context with our nonlinear PCA type learning algorithms in a way outlined in detail in the next Section 5.

5 The two-layer ICA network

To recapitulate, we can achieve source signal separation by the following computational steps according to Section 4:

1. Starting point: we have a set of M statistically independent, zero-mean signals $s_k(i),\ i = 1, ..., M$ that are unobservable. By an unknown mixing process, a set of L observable linear mixtures $(M \leq L)\ x_k(j),\ j = 1, ..., L$ have been formed according to the model (22). For simplicity, let us assume that the additive noise is one of the M source signals and therefore it is omitted from the sum (22). Also, assume that the amplitudes of the source signals $s_k(i)$ have been absorbed in the mixing coefficients $a(ij)$ in such a way that we can assume that the variances of all $s_k(i)$ are equal to one. Thus we have from eq. (23):

$$\mathbf{x}_k = \mathbf{A}\mathbf{s}_k \tag{26}$$

with

$$E\{\mathbf{s}_k\mathbf{s}_k^T\} = \mathbf{I}. \tag{27}$$

2. Sphering (whitening): by a linear transformation $\mathbf{V}$, the signal vectors $\mathbf{x}_k$ are transformed to new signal vectors

$$\mathbf{v}_k = \mathbf{V}\mathbf{x}_k \tag{28}$$

such that

$$E\{\mathbf{v}_k\mathbf{v}_k^T\} = \mathbf{I}. \tag{29}$$

Matrix $\mathbf{V}$ could be easily estimated from a sample of the $\mathbf{x}_k$ vectors by first computing their covariance matrix in some way and then by normalizing using the eigenvalues, but it can also be realized neurally.

3. Separation: by a linear transformation $\mathbf{B}$, the whitened vectors $\mathbf{v}_k$ are transformed into output signal vectors

$$\mathbf{y}_k = \mathbf{B}\mathbf{v}_k \tag{30}$$

such that the elements of $\mathbf{y}_k$ are not only uncorrelated, but statistically independent. We constrain the matrix $\mathbf{B}$ to be an orthogonal matrix, which implies that

$$E\{\mathbf{y}_k\mathbf{y}_k^T\} = \mathbf{B}E\{\mathbf{v}_k\mathbf{v}_k^T\}\mathbf{B}^T = \mathbf{B}\mathbf{I}\mathbf{B}^T = \mathbf{I}. \tag{31}$$

Then the condition of Theorem 1 in Section 4 is satisfied. Now, eqs. (26), (28), and (30) yield for the output vector

$$\mathbf{y}_k = \mathbf{B}\mathbf{V}\mathbf{A}\mathbf{s}_k \tag{32}$$

and maximizing the criterion (25) for the elements $y_k(i)$ of the output vector could be used to find the original independent components $s_k(i)$. This is because according to the Condition 3 in Theorem 1, at any maximum point of (25) the matrix $\mathbf{BVA}$ attains the form $\mathbf{\Lambda P}$, and $\mathbf{y}_k$ becomes equal to a version of $\mathbf{s}_k$ with possibly permuted and sign-reversed elements. Note that with these steps, even if we obtain the matrix $\mathbf{BVA}$, we cannot yet estimate the reconstruction matrix $\mathbf{B}$ itself because $\mathbf{A}$ is unknown. This must be done separately as shown by Karhunen et al. [20].

Because both kurtosis maximization/minimization and the sphering can be performed adaptively in an on-line process receiving a sequence of mixture signals $\mathbf{x}_k$, a neural network is a possible implementation for the algorithms. Consider the two-layer neural network of Fig. 3 in which the first layer is linear and spheres the data. If the original L-element input vector is $\mathbf{x}$ and the output of the first layer is $\mathbf{v} = \mathbf{V}\mathbf{x}$, then $E\{\mathbf{v}\mathbf{v}^T\} = \mathbf{I}$. The elements of $\mathbf{v}$ have variances equal to 1 and are uncorrelated - but in general not independent. The sphering transformation $\mathbf{V}$ is implemented by the weights of the linear layer and can be learned in a neural learning algorithm introduced by Plumbley [31]:

$$\mathbf{V}_{k+1} = \mathbf{V}_k + \mu_k(\mathbf{V}_k\mathbf{x}_k\mathbf{x}_k^T\mathbf{V}_k^T - \mathbf{I})\mathbf{V}_k. \tag{33}$$

In the second layer, one of the nonlinear Hebbian learning rules (9), (10), or (18) is used, with the sphered vectors $\mathbf{v}_k$ taken as inputs instead of the original

Figure 3: The ICA network. Weight matrix $\mathbf{V}$: the whitening transformation. Weight matrix $\mathbf{W}$: the separation transformation

$\mathbf{x}_k$. The nonlinear gradient function g must be chosen so that kurtosis is minimized/maximized; Fig. 1 of Section 2 shows that the hyperbolic tangent, which is the derivative of the *lncosh* function, is a suitable function. Algorithms (10) and (18) seem to work better than (9). This is because usually the number of mixtures L is equal to the number of original signals M (in fact, Theorem 1 applies to this case only), in which case the weight matrix $\mathbf{W}_k$ will be square. It is also required to be an orthogonal matrix. Then the learning rule (9) cannot be used due to the form of the term $\mathbf{I} - \mathbf{W}_k \mathbf{W}_k^T$, which is zero for any orthogonal matrix $\mathbf{W}_k$. Even starting from a non-orthogonal initial weight matrix $\mathbf{W}_0$, the matrix will be orthogonalized in the algorithm (9) too quickly for the proper rotation to take place. This effect seems to be avoided in the bigradient algorithm (18) in which the normalizing term is additive. For algorithm (10), this problem does not occur.

After a proper learning period using a sample of the mixture signals $\mathbf{x}_k$, the output of the network $\mathbf{y}_k = \mathbf{W}_k \mathbf{v}_k$ tends to a vector that approximates the original signal vector $\mathbf{s}_k$, although the order of the signals and their amplitudes may have changed. After learning, the network can be used to separate additional samples from the same signals, not occurring in the training set.

6 Experimental results on source signal separation

We have simulated the above algorithms using various test cases of real-world data. In the following, we present two examples and conclusions from the simulations.

Example 6.1. In this experiment, real speech signals were used. Speech samples were collected from 10 speakers (7 males, 3 females) over a regular radio broadcast and sampled at a sampling rate of 8 kHz. The duration of the samples was 10 seconds each. Ten linear mixtures were made using a nonsingular mixing matrix. The mixtures sound totally unintelligible and very much like "cocktail party" noise. A time window of one second from the beginning of each mixture sample, containing 8000 samples, was collected and the 10-dimensional input vectors $\mathbf{x}_k$ were constructed. This set of 8000 vectors were used cyclically in training the ICA network of Fig. 3. The nonlinear PCA algorithm used here was the bigradient learning rule of eq. (18), with a constant learning rate $\mu_k = -0.01$. Now the signals have a sharply peaked density with positive kurtosis, and therefore kurtosis maximization was approximated by the algorithm. This was achieved by using a hyperbolic tangent nonlinearity and negative learning rates. After convergence, the 10 seconds long mixture signals were inputted to the network, and the outputs were equally long approximations to the original source signals. The qualitative accuracy of the approximations was quite good, up to easy recognition of both the words and the speakers.

Example 6.2. In this experiment, 3 digital images were used, shown in Fig. 5a. The image size was 100×140. Using row-by-row scanning, each image was converted to a scalar signal $s_k(i),\ i = 1, 2, 3,\ k = 1, ..., 14000$. Three mixtures were made, shown in Fig. 5b. The 14000 signal vectors $\mathbf{x}_k$ of dimension 3 obtained from the mixtures, were all used for training the ICA network. The batch was iterated 20 times, and the Nonlinear PCA rule (10) with $g(y) = tanh(y)$ nonlinearity was used for training the separation matrix $\mathbf{W}$. The sphered mixtures are shown in Fig. 5c. This is essentially the result that would be obtained by the usual linear PCA technique; it can be seen that already linear PCA has a weak separation ability. The outputs of the converged network, again arranged into 100×140 images, are shown in Fig. 5d. The signs and amplitudes of the reconstructed signals have been adjusted to the suitable range for visual display. As can be seen, the visual quality of the reconstructions is good.

7 Conclusions

The nonlinear Hebbian learning rules proposed by the first author in Oja et al (1991) were reviewed in the light of Projection Pursuit and Independent Component Analysis, following recent ideas of several authors. Previously, it has been shown that these rules are related to Principal Component Analysis in the sense that they

Figure 4: Separation of images from mixtures. a) original digital images. b) linear mixtures. c) sphered (whitened) mixtures, outputs of the first layer of the ICA network. d) approximations to the sources, outputs of the second layer of the ICA network

tend to find those directions in the input space that have *maximal variance* and are *uncorrelated*. However, provided that all directions in the input space have equal variance, the one-unit case with a $tanh(y)$ nonlinearity in the learning is approximately *minimizing the kurtosis* of the neuron output. This means that the weight vector of the unit will find a direction in the input space on which the projection of the input data is heavily clustered and deviates strongly from the Gaussian distribution, like is the case in the Projection Pursuit technique. This is useful in data clustering and visualization.

The multi-unit algorithms have the extra advantage that while minimizing or maximizing the kurtosis in several directions, they tend to find those directions along which the input data projections are *statistically independent*. Therefore these algorithms can be used for Independent Component Analysis and source separation. In both applications, it is essential to remove the influence of first and second order statistics, which can be done by first making the inputs zero mean and then removing different variances by data sphering. This can be done in a simple two-layer network, called here the ICA network.

Experiments with real signals showed that the ICA network is able to separate independent signals from their mixtures, e.g. ten speech signals from ten mixtures, with small approximation error.

References

[1] P. Baldi and K. Hornik, "Neural networks for principal component analysis: learning from examples without local minima," *Neural Networks*, vol. 2, pp. 53-58, 1989.

[2] A. Bell and T. Sejnowski, "Blind separation and blind deconvolution: an information-theoretic approach," in *Proc. 1995 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, Detroit, USA, May 1995, pp. 3415-3418.

[3] G. Burel, "Blind separation of sources: a nonlinear neural algorithm," *Neural Networks*, vol. 5, pp. 937-947, 1992.

[4] J.-F. Cardoso and B. Laheld, "Equivariant adaptive source separation," manuscript submitted to *IEEE Trans. on Signal Processing*, October 1994.

[5] Y. Chauvin, "Principal component analysis by gradient descent on a constrained linear Hebbian cell", in *Proc. Int. Joint Conf. on Neural Networks*, Washington DC, USA, June 1989, pp. 1373 - 1380.

[6] A. Cichocki and R. Unbehauen, *Neural Networks for Optimization and Signal Processing.* New York: John Wiley, 1993.

[7] P. Comon, "Separation of stochastic processes," in *Proc. of Workshop on Higher-Order Spectral Analysis*, Vail, Colorado, June 1989, pp. 174-179.

[8] P. Comon, "Independent component analysis - a new concept?," *Signal Processing*, vol. 36, pp. 287-314, 1994.

[9] G. Deco and W. Brauer, "Nonlinear higher-order statistical decorrelation by volume-conserving neural architectures," to appear in *Neural Networks*, 1995 (in press).

[10] Friedman, J. and Tukey, J., "A projection pursuit algorithm for exploratory data analysis," *IEEE Trans. Comput.* C-23, 1974, pp. 881 - 889.

[11] J. Friedman, "Exploratory projection pursuit," *J. Amer. Statistical Assoc.*, vol. 82, no. 397, pp. 249-266, March 1987.

[12] C. Fyfe, D.R. McGregor, and R. Baddeley, "Exploratory projection pursuit: an artificial neural network approach," Dep. Computer Sci., Univ. of Strathclyde (Glasgow, Scotland), Res. Rep. /94/160, Oct. 1994.

[13] S. Haykin, *Neural Networks: A Comprehensive Foundation.* New York: IEEE Computer Society Press and Macmillan, 1994.

[14] P. Huber, "Projection pursuit," *The Annals of Statistics*, vol. 13, no. 2, pp. 435-475, 1985.

[15] C. Jutten and J. Herault, "Blind separation of sources, part I: an adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, no. 1, pp. 1-10, July 1991.

[16] J. Karhunen, "Adaptive algorithms for estimating eigenvectors of correlation type matrices," in *Proc. 1984 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, San Diego, California, March 1984, pp. 14.6.1-14.6.4.

[17] J. Karhunen and J. Joutsensalo, "Representation and separation of signals using nonlinear PCA type learning," *Neural Networks*, vol. 7, no. 1, pp. 113-127, 1994.

[18] J. Karhunen, "Optimization criteria and nonlinear PCA neural networks," in *Proc. 1994 IEEE Int. Conf. on Neural Networks*, Orlando, Florida, June 1994, pp. 1241-1246.

[19] J. Karhunen and J. Joutsensalo, "Generalizations of principal component analysis, optimization problems, and neural networks," to appear in *Neural Networks* (in press).

[20] J. Karhunen, L. Wang, and J. Joutsensalo, "Neural estimation of basis vectors in independent component analysis," to appear.

[21] B. Laheld and J.-F. Cardoso, "Adaptive source separation with uniform performance," in *Signal Processing VII: Theories and Applications (Proc. EUSIPCO-94)*, M. Holt et al. (Eds.). Lausanne: EURASIP, 1994, vol. 2, pp. 183-186.

[22] K. Matsuoka, M. Ohya, and M. Kawamoto, "A neural net for blind separation of nonstationary signals," *Neural Networks*, vol. 8, no. 3, pp. 411-419, 1995.

[23] E. Moreau and O. Macchi, "New self-adaptive algorithms for source separation based on contrast functions," in *Proc. IEEE Signal Proc. Workshop on Higher Order Statistics*, Lake Tahoe, USA, June 1993, pp. 215-219.

[24] E. Oja, "A simplified neuron model as a Principal Components Analyzer," *J. Math. Biol.* 15, 1982, pp. 267 - 273.

[25] E. Oja and J. Karhunen, "On stochastic approximation of eigenvectors and eigenvalues of the expectation of a random matrix", *J. Math. Anal. Appl.* 106, 1985, pp. 69 - 84.

[26] E. Oja, "Neural networks, principal components, and subspaces". *Int. J. of Neural Systems* 1, 1989, pp. 61 - 68.

[27] E. Oja and T. Kohonen, "The subspace learning algorithm as a formalism for Pattern Recognition and Neural Networks," *Proc. IEEE 1988 Int. Conf. on Neural Networks*, San Diego, CA, July 1988, pp. 277-284.

[28] E. Oja, H. Ogawa, and J. Wangviwattana, "Learning in nonlinear constrained Hebbian networks," in *Artificial Neural Networks (Proc. ICANN-91)*, T. Kohonen et al. (Eds.). Amsterdam: North-Holland, 1991, pp. 385-390.

[29] E. Oja, "Principal components, minor components, and linear neural networks," *Neural Networks*, vol. 5, pp. 927-935, 1992.

[30] J. Platt and F. Faggin, "Networks for the separation of sources that are superimposed and delayed," in *Advances in Neural Processing Systems 4*, J. Moody et al. (Eds.). San Mateo, California: Morgan Kaufmann, 1991, pp. 730-737.

[31] M. Plumbley, "A Hebbian/anti-Hebbian network which optimizes information capacity by orthonormalizing the principal subspace, in Proc. IEE Conf. on Artificial Neural Networks, Brighton, UK, May 1993, pp. 86-90.

[32] A. Sudjianto and M. Hassoun, "Nonlinear Hebbian rule: a statistical interpretation", in *Proc. IEEE Int. Conf. on Neural Networks*, Orlando, Fla., July 1994, pp. 1247 - 1252.

[33] J. Taylor and M. Plumbley, "Information theory and neural networks," in *Mathematical Approaches to Neural Networks*, J. Taylor (Ed.). Amsterdam: Elsevier Science Publ., 1993, pp. 307-340.

[34] C. Therrien, *Discrete Random Signals and Statistical Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1992.

[35] L. Wang, J. Karhunen, and E. Oja, "A bigradient optimization approach for robust PCA, MCA, and source separation," submitted to *1995 IEEE Int. Conf. on Neural Networks*, Perth, Australia, November 1995.

[36] M. sterberg and R. Lenz, "Unsupervised parallel feature extraction from first principles, in *Advances in Neural Processing Systems 6*, J. Cowan et al. (Eds.). San Francisco: Morgan Kaufmann, 1994, pp. 136-143.