Pergamon

Neural Networks, Vol. 8, No. 4, pp. 549–562, 1995

0893-6080/95 $9.50 + .00

0893-6080(94)00098-0

CONTRIBUTED ARTICLE

Generalizations of Principal Component Analysis, Optimization Problems, and Neural Networks

JUHA KARHUNEN AND JYRKI JOUTSENSALO

Helsinki University of Technology

(*Received* 17 *March* 1994; *accepted* 7 *October* 1994)

Abstract—*We derive and discuss various generalizations of neural PCA (Principal Component Analysis)-type learning algorithms containing nonlinearities using optimization-based approach. Standard PCA arises as an optimal solution to several different information representation problems. We justify that this is essentially due to the fact that the solution is based on the second-order statistics only. If the respective optimization problems are generalized for nonquadratic criteria so that higher-order statistics are taken into account, their solutions will in general be different. The solutions define in a natural way several meaningful extensions of PCA and give a solid foundation for them. In this framework, we study more closely generalizations of the problems of variance maximization and mean-square error minimization. For these problems, we derive gradient-type neural learning algorithms both for symmetric and hierarchic PCA-type networks. As an important special case, the well-known Sanger's generalized Hebbian algorithm (GHA) is shown to emerge from natural optimization problems.*

Keywords—Principal components, Optimization, Neural network, Unsupervised learning, Nonlinearity, Robust statistics, Generalized Hebbian algorithm, Oja's rule.

1. INTRODUCTION

Principal component analysis (PCA) is a well-known, widely used statistical technique. Essentially, the same basic technique is utilized in several areas under different names such as Karhunen–Loeve transform or expansion, Hotelling transform, and signal subspace or eigenstructure approach. In pattern recognition, PCA is used in various forms for optimal feature extraction and data compression (Devijver & Kittler, 1982). In image processing, PCA defines the Hotelling or KL transform that is optimal in image data compression (Jain, 1989). In signal processing, a useful characterization of signals is to assume that they roughly lie in the signal subspace defined by PCA. Several modern methods of signal modeling, spectrum estimation, and array processing are based on this concept (Therrien, 1992).

Let $\mathbf{x}$ be an L-dimensional data vector coming from some statistical distribution centralized to zero: $E\{\mathbf{x}\} = \mathbf{0}$. The ith principal component $\mathbf{x}^T\mathbf{c}(i)$ of $\mathbf{x}$ is defined by the normalized eigenvector $\mathbf{c}(i)$ of the data covariance matrix $\mathbf{C} = E\{\mathbf{x}\mathbf{x}^T\}$ associated with 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 networks are neural realizations of PCA in which the weight vectors $\mathbf{w}(i)$ of the neurons or the weight matrix $\mathbf{W} = [\mathbf{w}(1), \ldots, \mathbf{w}(M)]$ converge to the principal eigenvectors $\mathbf{c}(i)$ or to the PCA subspace during the learning phase.

It is well known that standard PCA emerges as the optimal solution to several different information representation problems. These include:

1. maximization of linearly transformed variances $E\{[\mathbf{w}(i)^T\mathbf{x}]^2\}$ or outputs of a linear network under orthonormality constraints ($\mathbf{W}^T\mathbf{W} = \mathbf{I}$);
2. minimization of the mean-square representation error $E\{\|\mathbf{x} - \hat{\mathbf{x}}\|^2\}$, when the input data $\mathbf{x}$ are approximated using a lower dimensional linear subspace $\hat{\mathbf{x}} = \mathbf{W}\mathbf{W}^T\mathbf{x}$;
3. uncorrelatedness of outputs $\mathbf{w}(i)^T\mathbf{x}$ of different neurons after an orthonormal transform ($\mathbf{W}^T\mathbf{W} = \mathbf{I}$); and
4. minimization of representation entropy.

Derivations of the optimal PCA solutions with the required assumptions and constraint conditions can be found in several textbooks (see, e.g., Devijver & Kittler, 1982; Jain, 1989; Young & Calvert, 1974). The

Acknowledgements: The authors are grateful to Prof. Erkki Oja for useful comments and insightful discussions on the topic of the paper, and to a reviewer for his detailed comments.

Requests for preprints should be sent to Dr. Juha Karhunen, Helsinki University of Technology, Laboratory of Computer and Information Science, Rakentajanaukio 2 C, FIN-02150 Espoo, Finland.

criterion to be optimized can often be defined in slightly different forms, so that the solution is provided by either the PCA subspace or the principal eigenvectors themselves.

In the next section, we briefly consider the relative merits and shortcomings of linear and nonlinear PCA networks and algorithms. Various robust and nonlinear extensions of neural PCA are introduced by generalizing in the following sections each of the above-mentioned quadratic optimization criteria, which lead to standard PCA solution. Such an approach gives a sound mathematical foundation to the generalizations and helps to understand the properties of the corresponding learning algorithms. The main attention is devoted to the first two criterions, for which we derive several new learning algorithms and present experimental results.

Another typical approach to "nonlinear" PCA has been just to insert a nonlinearity somewhere in a PCA network and see what happens, or to propose some other heuristic modification. The results of such heuristic algorithms are more difficult to interpret. A third approach is to start from some fixed neural network structure and study what kind of algorithms can be realized using it. This viewpoint has been adopted in a recent review report (Oja & Karhunen, 1993). Sometimes these approaches lead to the same learning algorithms that are obtained from suitable optimization criteria. Such connections are pointed out in the paper.

In this paper, we extend and generalize in several ways the original ideas presented in Karhunen and Joutsensalo (1994). Preliminary results have been given in the conference papers (Karhunen & Joutsensalo, 1993a, b). In the following, we present these results in a unified and extended form in the general optimization framework described above. Perhaps the most important single result is the derivation of the well-known generalized Hebbian algorithm (GHA) (Sanger, 1989b) (as well as its robust and "nonlinear" counterparts) from the variance maximization and mean-square error minimization problems.

The rest of the paper is organized as follows. In the next section, we compare and discuss linear and nonlinear neural PCA on a general level. After this, we discuss in the following sections generalizations of each of the four information representation problems mentioned before. The main attention is on the first two problems, namely variance maximization and representation error minimization, for which we present new theoretical and experimental results. In the conclusions, we present some general comments on the results.

2. LINEAR AND NONLINEAR NEURAL PCA

It is now well known that relatively simple, neurobiologically justified Hebbian-type learning rules can provide PCA. This, together with the usefulness and many applications of PCA, has prompted a lot of interest in various neural realizations of PCA (see Cichocki & Unbehauen, 1993a; Haykin, 1994; Hertz, Krogh, & Palmer, 1991; Kung, 1993; Oja, 1992). However, PCA networks and learning algorithms have some limitations that diminish their attractiveness:

1. Standard PCA networks are able to realize only linear input-output mappings.
2. The eigenvectors needed in standard PCA can be computed efficiently using well-known numerical methods. Gradient-type neural PCA learning algorithms converge relatively slowly, and achieving a good accuracy requires an excessive number of iterations in large problems.
3. Principal components are defined solely by the data covariances (or correlations). These second-order statistics characterize completely only Gaussian data and stationary, linear processing operations.
4. PCA networks cannot usually separate independent subsignals from their linear mixture.

If a PCA-type network contains nonlinearities, the situation becomes much more favorable for a neural realization. First, the input–output mapping becomes generally nonlinear, which is a major argument for using neural networks. Nonlinear processing of the data is often more efficient, and the properties of standard linear methods have been explored thoroughly.

Second, neural algorithms become much more competitive or may be the only possibility for heuristic learning principles. In optimizing nonquadratic criteria, one must resort to iterative algorithms anyway, because efficient closed-form solutions are usually not available. The gradient-type neural learning algorithms are iterative by nature, and a suitably chosen nonlinearity (e.g., the sigmoid) may be implemented via analog hardware almost as easily as linear functions.

Our third motivation of using nonlinearities is that they introduce in an implicit way higher-order statistics into the computations. This can be seen by expanding the nonlinearities into their Taylor series. Higher-order statistics, defined by cumulants and higher than second moments (see, e.g., Nikias & Mendel, 1993), are needed for a good characterization of non-Gaussian data. There exist several important problems that cannot adequately be solved using merely second-order statistics. This has prompted a lot of recent research in higher-order statistics and spectra (e.g., in signal processing) (Nikias & Mendel, 1993).

Fourth, the outputs of standard PCA networks are usually at most mutually uncorrelated but not independent, which would be more desirable in many cases. We have demonstrated (Karhunen & Joutsensalo, 1994) that adding nonlinearities to a PCA network increases the independence of the outputs, so that the original signals can sometimes be roughly separated from their mixture. Recently, Independent Component Analysis (ICA) (Comon, 1994) has been introduced as an interesting extension of PCA in context with the

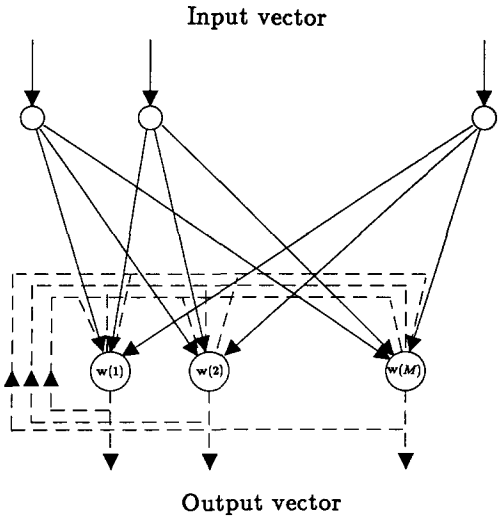

FIGURE 1. Architecture of the symmetric network. Feedback connections (dashed lines) are needed in the learning phase only.

signal separation problem (Jutten & Herault, 1991). This will be discussed later in this paper.

Naturally, nonlinear PCA-type networks have some drawbacks compared to the linear ones. The mathematical analysis of the learning algorithms is often inherently difficult, making the properties of the networks less well understood. The nonlinear learning algorithms are more complicated and may sometimes be caught more easily in local minima. Adding nonlinearities to a neural network does not help automatically or in all problems. For some nonquadratic criteria the final input–output mapping is still linear, because the nonlinearities appear in the learning rule only.

In the following presentation, nonlinear PCA-type networks and learning algorithms are divided into symmetric and hierarchic ones quite similar to those for standard PCA networks. In standard PCA learning algorithms, some kind of hierarchy or differentiation is necessary between the learning rules of different neurons to get the principal components or eigenvectors themselves. The completely symmetric algorithms yield PCA subspace and some linear combinations of principal components only. It seems that in nonlinear PCA networks hierarchy is not so important, because nonlinearities break the complete symmetry during learning, and the outputs of symmetric networks become more unique as in the linear case (Oja, Ogawa, & Wangviwattana, 1992; Karhunen & Joutsensalo, 1994; Xu, 1993).

The learning algorithms derived by considering generalizations of the optimization problems leading to standard PCA can be divided into two classes in another way. We distinguish between so-called *robust PCA* algorithms (Karhunen & Joutsensalo, 1993a, b; Cichocki & Unbehauen, 1993b; Xu & Yuille, 1993), and *nonlinear PCA* algorithms. We define robust PCA so that the criterion to be optimized grows less than quadratically, and the constraint conditions are the same as for the standard PCA solution, which emerges from the respective quadratic criterion. Typically, the weight vectors of the neurons (basis vectors of the expansion) are required to be mutually orthonormal. Robust PCA problems usually lead to mildly nonlinear algorithms, in which the nonlinearities appear at selected places only. More specifically, at least some of the outputs of the neurons are still their linear responses $y(i) = \mathbf{x}^T\mathbf{w}(i)$, where $\mathbf{w}(i)$ is the weight vector of ith neuron. In the nonlinear PCA algorithms all the outputs $g[y(i)]$ of the neurons are nonlinear functions of the response.

The structure of our nonlinear PCA network is shown in Figure 1 for the symmetric case, and in Figure 2 for the standard hierarchic arrangement. The network contains input and output layers only. After learning, the feedback connections between outputs and inputs shown by dashed lines in the figures are not needed, and the network becomes purely feedforward. The same structure can be used for all the algorithms, but details of the realization vary. Other structures have been proposed in the literature (e.g., recurrent generalizations and nonlinear PCA networks with lateral connections) (Palmieri, 1994).

Finally, we note that various nonneural, nonlinear extensions of PCA have been introduced in statistics (e.g., Bekker & de Leeuw, 1988; Gifi, 1990; Hastie & Stuetzle, 1989). These approaches are often more nonlinear than ours in the sense that also the basis functions of the expansion (corresponding to the weight vectors of the neurons) are nonlinear, not only the coefficients of the expansion.

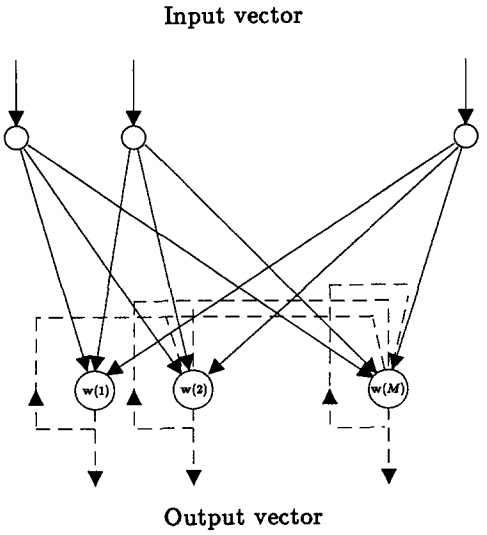

FIGURE 2. Architecture of the hierarchic network. Feedback connections (dashed lines) are needed in the learning phase only.

3. GENERALIZATION OF VARIANCE MAXIMIZATION

In this section, we study more closely generalizations of the first information representation problem mentioned in the Introduction, namely variance maximization.

The standard quadratic problem leading to a PCA solution is one of how to maximize the output variances $E\{y(i)^2\} = E\{[\mathbf{w}(i)^T\mathbf{x}]^2\} = \mathbf{w}(i)^T\mathbf{C}\mathbf{w}(i)$ of the linear network under orthonormality constraints. The number of neurons M is assumed to be less than or equal to the dimension L of the data vectors $\mathbf{x}$. The maximization problem is not well defined unless the nonrandom L-dimensional weight vectors $\mathbf{w}(i)$ of the neurons are constrained somehow. In lack of prior knowledge, orthonormality constraints are the most natural, because they measure the variances along maximally different directions. Normally, the ith weight vector $\mathbf{w}(i)$ is constrained so that it must have unit norm and be orthogonal to the weight vectors $\mathbf{w}(j)$, $j = 1, \ldots, i-1$ of the previous neurons. These constraints take the mathematical form $\mathbf{w}(i)^T\mathbf{w}(j) = \delta_{ij}$, $j \leq i$, where the Kronecker delta $\delta_{ij} = 1$ for $i = j$ and 0 for $i \neq j$. The optimal $\mathbf{w}(i)$ is then the ith principal eigenvector $\mathbf{c}(i)$ of $\mathbf{C}$, and the outputs of the PCA network become the principal components of the data vectors. The PCA networks and learning algorithms are in this case hierarchic. In the following, we refer to this constraint set and case as the *standard hierarchic case.*

The respective variance maximization problem can be solved for symmetric orthonormality constraints $\mathbf{w}(i)^T\mathbf{w}(j) = \delta_{ij}$, $j \neq i$, as well. It is convenient to define the $L \times M$ weight matrix $\mathbf{W} = [\mathbf{w}(1), \ldots, \mathbf{w}(M)]$, for which columns are the weight vectors of the M neurons. The symmetric orthonormality constraints then become $\mathbf{W}^T\mathbf{W} = \mathbf{I}$, where $\mathbf{I}$ is the unit matrix. Let $\mathbf{y} = \mathbf{W}^T\mathbf{x}$ denote the response vector, which is the output vector of a linear PCA network. The criterion to be maximized can then be represented compactly as $E\{\|\mathbf{y}\|^2\} = \mathrm{tr}(\mathbf{W}^T\mathbf{C}\mathbf{W})$, where $\mathrm{tr}(\cdot)$ denotes the trace of the matrix. The optimal solution is now given by any orthonormal basis spanning the PCA subspace, and is thus not unique. This version of the variance maximization problem leads to PCA subspace networks and learning rules. We refer to this case and constraint set as the *standard symmetric case* in the rest of the paper.

Consider now generalization of the variance maximization problem for robust PCA. Instead of using the standard mean-square value, we can maximize a more general expectation $E\{f[\mathbf{x}^T\mathbf{w}(i)]\}$ of the response $\mathbf{x}^T\mathbf{w}(i)$ of the ith neuron. The function $f(t)$ is assumed to be a valid cost function that grows less than quadratically, at least for large values of t. More specifically, we assume that $f(t)$ is even, nonnegative, continuously differentiable almost everywhere, and $f(t) \leq t^2/2$ for large values of $|t|$. Furthermore, its only minimum is attained at $t = 0$, and $f(t_1) \leq f(t_2)$ if $|t_1| < |t_2|$. Some of these assumptions are not absolutely necessary. Examples of such a function are $f(t) = \ln\cosh(t)$ and $f(t) = |t|$ (see, Cichocki & Unbehauen, 1993a; Karhunen & Joutsensalo, 1994).

The criterion to be maximized is then for each neuron weight vector $\mathbf{w}(i)$, $i = 1, \ldots, M$ of the form

$$J_1[\mathbf{w}(i)] = E\{f[\mathbf{x}^T\mathbf{w}(i)]\} + \sum_{j=1}^{I(i)} \lambda_{ij}[\mathbf{w}(i)^T\mathbf{w}(j) - \delta_{ij}]. \quad (1)$$

Here the summation imposes via the Lagrange multipliers $\lambda_{ij} = \lambda_{ji}$ the necessary orthonormality constraints $\mathbf{w}(i)^T\mathbf{w}(j) = \delta_{ij}$. Both the hierarchic and symmetric problems can be discussed under the same general criterion (1). In the standard symmetric case, the upper bound of the summation index is $I(i) = M$ for all $i = 1, \ldots, M$. In the standard hierarchic case $I(i) = i$; the optimal weight vector of the ith neuron defines then the robust counterpart of ith principal eigenvector $\mathbf{c}(i)$. It is possible to choose the constraints for each neuron in eqn (1) from an even more general set of indices $S(i)$, provided that the index i corresponding to the normalization constraint $\mathbf{w}(i)^T\mathbf{w}(i) = 1$ is included in $S(i)$. In particular, the order of the neurons could be permuted in hierarchic networks. However, the two basic cases described above are the most relevant ones, and we concentrate on them in the following.

The gradient of $J_1[\mathbf{w}(i)]$ with respect to $\mathbf{w}(i)$ is

$$\mathbf{h}(i) = \frac{\partial J_1(\mathbf{w}(i))}{\partial \mathbf{w}(i)} = E\{\mathbf{x}g[\mathbf{x}^T\mathbf{w}(i)]\} + 2\lambda_{ii}\mathbf{w}(i) + \sum_{j=1, j\neq i}^{I(i)} \lambda_{ij}\mathbf{w}(j), \quad (2)$$

where $g(t)$ is the derivative $df(t)/dt$ of $f(t)$. At the optimum, the gradients must vanish for $i = 1, \ldots, M$. Differentiation with respect to the Lagrange multipliers yields the orthonormality constraints

$$\mathbf{w}(i)^T\mathbf{w}(j) = \delta_{ij}, j = 1, \ldots, I(i), \quad (3)$$

which must also be satisfied at the optimum. The optimal values of the Lagrange multipliers can be determined by multiplying eqn (2) by $\mathbf{w}(j)^T$, $j = 1, \ldots, I(i)$, from the left, and equating the result to zero. Taking into account eqn (3), this yields $\lambda_{ij} = -\mathbf{w}(j)^T E\{\mathbf{x}g[\mathbf{x}^T\mathbf{w}(i)]\}$ for $i \neq j$, and $\lambda_{ii} = -0.5\mathbf{w}(i)^T E\{\mathbf{x}g[\mathbf{x}^T\mathbf{w}(i)]\}$. Inserting these values into eqn (2) we get

$$\mathbf{h}(i) = \left[\mathbf{I} - \sum_{j=1}^{I(i)} \mathbf{w}(j)\mathbf{w}(j)^T\right] E\{\mathbf{x}g[\mathbf{x}^T\mathbf{w}(i)]\}. \quad (4)$$

A practical stochastic gradient algorithm for maximizing eqn (1) is now obtained by inserting the estimate $\mathbf{h}_k(i)$ of the gradient vector (4) at step k into the update formula

$$\mathbf{w}_{k+1}(i) = \mathbf{w}_k(i) + \mu_k \mathbf{h}_k(i). \quad (5)$$

Here μ_k is the gain parameter. Throughout the paper, we use the standard instantaneous gradient estimates. They are obtained simply by omitting the expectations and using instead of them the instantaneous values of the quantities in question. The final algorithm thus becomes ($i = 1, \ldots, M$)

$$\mathbf{w}_{k+1}(i) = \mathbf{w}_k(i) + \mu_k \left[\mathbf{I} - \sum_{j=1}^{I(i)} \mathbf{w}_k(j)\mathbf{w}_k(j)^T \right] \mathbf{x}_k g[\mathbf{x}_k^T \mathbf{w}_k(i)]. \quad (6)$$

The assumptions made earlier on the cost function $f(t)$ imply that its derivative $g(t)$ appearing in eqn (6) and the other learning algorithms in this paper should be an odd nondecreasing (often monotonically growing) function of t. For stability reasons, it is at least necessary to assume that $g(t) \leq 0$ for $t < 0$ and $g(t) \geq 0$ for $t > 0$ (see Oja, Ogawa, & Wangviwattana, 1991).

Defining the instantaneous representation error vector

$$\mathbf{e}_k(i) = \mathbf{x}_k - \sum_{j=1}^{I(i)} [\mathbf{x}_k^T \mathbf{w}_k(j)]\mathbf{w}_k(j) = \mathbf{x}_k - \sum_{j=1}^{I(i)} y_k(j)\mathbf{w}_k(j), \quad (7)$$

the algorithm (6) can be written in a simpler form

$$\mathbf{w}_{k+1}(i) = \mathbf{w}_k(i) + \mu_k g[y_k(i)]\mathbf{e}_k(i). \quad (8)$$

From eqns (7) and (8), one can easily see that no matrix multiplications are needed in the actual realization. The representation error is discussed more closely in the next section.

In the symmetric case $I(i) = M$, $i = 1, \ldots, M$, the error vector $\mathbf{e}_k(i)$ becomes the same $\mathbf{e}_k$ for all the neurons. Then eqn (6) can be expressed compactly in the matrix form

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k [\mathbf{I} - \mathbf{W}_k \mathbf{W}_k^T] \mathbf{x}_k g(\mathbf{x}_k^T \mathbf{W}_k) = \mathbf{W}_k + \mu_k \mathbf{e}_k g(\mathbf{y}_k^T), \quad (9)$$

where $\mathbf{y}_k = \mathbf{W}_k^T \mathbf{x}_k$ is the instantaneous response vector. The function $g(t)$ is applied separately to each component of its argument vector. The algorithm (9) coincides with the well-known Oja's PCA subspace rule (Cichocki & Unbehauen, 1993a; Hertz et al., 1991; Kung, 1993; Oja, 1992) in the linear special case $g(t) = t$. Otherwise, (9) defines a robust generalization of Oja's rule that was first proposed quite heuristically at the end of the paper by Oja et al. (1991).

In the standard hierarchic case $I(i) = i$, eqn (6) can be written in the matrix form

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k \{ \mathbf{x}_k g(\mathbf{y}_k^T) - \mathbf{W}_k \mathrm{UT}[\mathbf{y}_k g(\mathbf{y}_k^T)] \} \quad (10)$$

where the upper triangular operator UT sets the elements of its argument matrix to zero below the diagonal. In the linear special case $g(t) = t$, eqn (10) coincides exactly with the well-known GHA algorithm (Cichocki & Unbehauen, 1993a; Haykin, 1994; Kung, 1993; Oja, 1992) proposed originally by Sanger (1989a, b). Otherwise, eqn (10) defines a robust generalization of the GHA algorithm.[1] Another, more practical formulation of eqn (10) is obtained by noting that the error vector (7) can be expressed in the standard hierarchic case recursively as $\mathbf{e}_k(i) = \mathbf{e}_k(i-1) - y_k(i)\mathbf{w}_k(i)$, $\mathbf{e}_k(0) = \mathbf{x}_k$. This shows that robust GHA can be implemented locally in a similar manner as standard GHA (Sanger, 1989b).

In the linear case, $g(t) = t$ corresponding to standard PCA learning the function $f(t) = t^2/2$, and the criterion to be maximized can be expressed in the natural closed form $E\{f[\mathbf{x}^T\mathbf{w}(i)]\} = \mathbf{w}(i)^T\mathbf{C}\mathbf{w}(i)/2$. It is clear that the optimal solution for the robust criterion (1) will in general not coincide with the respective PCA solution, though it can be close to it. As an example, consider the choice $f(t) = |t|$. The directions $\mathbf{w}(i)$ that maximize $E\{|\mathbf{x}^T\mathbf{w}(i)|\}$ are for some arbitrary nonsymmetric distribution different from those maximizing the variances $E\{[\mathbf{x}^T\mathbf{w}(i)]^2\}$ under orthonormality constraints.

A more nonlinear generalization of the variance maximization problem is not straightforward, because it is not easy to decide what constraints should be imposed on the weight vectors, and the respective algorithms become less practical for nonorthonormal constraints (Karhunen & Joutsensalo, 1994).

4. GENERALIZED REPRESENTATION ERROR

4.1. Standard PCA Solutions

Consider the linear approximation $\hat{\mathbf{x}}(i)$ of the data vectors $\mathbf{x}$ in terms of a set of vectors $\mathbf{w}(j)$, $j = 1, \ldots, I(i)$:

$$\hat{\mathbf{x}}(i) = \sum_{j=1}^{I(i)} [\mathbf{x}^T\mathbf{w}(j)]\mathbf{w}(j) = \sum_{j=1}^{I(i)} y(j)\mathbf{w}(j). \quad (11)$$

Because the number $I(i)$ of the basis vectors $\mathbf{w}(j)$ is usually smaller than the dimensionality L of the data vectors, there will be some error. The instantaneous representation (approximation) error $\mathbf{e}_k(i) = \mathbf{x}_k - \hat{\mathbf{x}}_k(i)$ for any data vector $\mathbf{x}_k$ is given by eqn (7). Here it is assumed that the $\mathbf{w}(j)$ vectors can be updated simultaneously.

Standard PCA-type solutions are obtained by minimizing the (quadratic) mean-square representation error $E\{\|\mathbf{e}(i)\|^2\} = E\{\|\mathbf{x} - \hat{\mathbf{x}}(i)\|^2\}$. Again, both the hierarchic and symmetric case can be considered. If the error must be minimized sequentially for any number

[1] In Karhunen and Joutsensalo (1993b) it was claimed that a similar derivation using different coefficients yields robust generalization of the SGA algorithm (Oja, 1992). However, this is not true, because any scalar coefficients of the Lagrange multipliers can always be absorbed into them.

of terms $I(i) = 1, I(i) = 2$, up to $I(i) = M$, the optimal basis vectors are the principal eigenvectors of the data covariance matrix: $\mathbf{w}(j) = \mathbf{c}(j)$. If the error must be minimal only for $I(i) = M$, the optimal solution $\hat{\mathbf{x}} = \mathbf{W}\mathbf{W}^T\mathbf{x}$ is given by any orthonormal ($\mathbf{W}^T\mathbf{W} = \mathbf{I}$) basis of the PCA subspace spanned by $\mathbf{c}(1), \ldots, \mathbf{c}(M)$ (Palmieri & Zhu, 1993). It is noteworthy that it is not necessary to impose any constraints on the weight vectors in the approximation (11), because minimization of the mean-square error will force the weight vectors mutually orthonormal.

Bourland and Kamp (1988) as well as Baldi and Hornik (1989) have shown that the same PCA subspace solution minimizes the mean-square error for the more general approximation $\hat{\mathbf{x}} = \mathbf{W}\mathbf{Q}^T\mathbf{x}$, where $\mathbf{W}$ and $\mathbf{Q}$ are $L \times M$ matrices. This holds even if

$$\hat{\mathbf{x}} = \mathbf{W}\, h(\mathbf{Q}^T\mathbf{x}), \tag{12}$$

where $h(t)$ is a smooth nonlinearity behaving linearly in the vicinity of the origin (Bourland & Kamp, 1988). These results show that PCA subspace provides the optimal solution for a linear MLP network, or if the output layer of a three-layer MLP (with one hidden layer) is linear. In these networks, the approximation (data compression) takes place in the hidden layer, where the number of units (M) is smaller than in the input and output layers (L). A more general MLP network does not usually lead to the PCA solution and could thus be regarded as a nonlinear extension of PCA. In particular, several authors have recently proposed and studied for data compression and representation a specific type of five-layer MLP network, where the input and output layers have the same number of units, and the network is trained using the back-propagation algorithm in the autoassociative mode. Usually the data compression achieved in the "bottleneck" middle layer in such a network is somewhat better than that provided by the respective PCA solution (see, e.g., Kambhatla & Leen, 1993).

Though useful, these approaches lead to multilayer structures requiring several weight matrices. The back-propagation learning algorithms are prone to local minima and often require excessive times for convergence. Our approaches are simpler: the network has two layers and requires only one weight matrix.

4.2. Robust PCA Algorithms for Linear Networks

We first consider robust generalizations of the mean-square representation error $E\{\|\mathbf{e}(i)\|^2\}$. Robust PCA-type algorithms can be obtained by minimizing the criterion

$$J_2[\mathbf{e}(i)] = \mathbf{1}^T E\{f[\mathbf{e}(i)]\} = \mathbf{1}^T E\{f[\mathbf{x} - \hat{\mathbf{x}}(i)]\}. \tag{13}$$

Here the L-vector $\mathbf{1}^T = [1, \ldots, 1]$, and $f(t)$ satisfies the assumptions specified before. The relationship of eqn (13) to the mean-square error becomes clearer, if we define a new function $h(t) = \sqrt{f(t)}$ and express $J_2[\mathbf{e}(i)]$ in the form $E\{\|h[\mathbf{e}(i)]\|^2\}$. This is always possible, because $f(t)$ is assumed to be nonnegative. The chosen notation is somewhat easier to handle mathematically. The criterion (13) coincides with the standard mean-square error if $f(t) = t^2$, and defines a robust generalization of it if $f(t)$ grows less than quadratically.

Minimizing eqn (13) with respect to the weight vector $\mathbf{w}(i)$ leads to the stochastic gradient algorithm ($i = 1, \ldots, M$)

$$\mathbf{w}_{k+1}(i) = \mathbf{w}_k(i) + \mu_k\{\mathbf{w}_k(i)^T g[\mathbf{e}_k(i)]\mathbf{x}_k + \mathbf{x}_k^T\mathbf{w}_k(i) g[\mathbf{e}_k(i)]\}, \tag{14}$$

$$\mathbf{e}_k(i) = \mathbf{x}_k - \sum_{j=1}^{I(i)} [\mathbf{x}_k^T\mathbf{w}_k(j)]\mathbf{w}_k(j) = \mathbf{x}_k - \sum_{j=1}^{I(i)} y_k(j)\mathbf{w}_k(j), \tag{15}$$

The instantaneous error vector (15) in eqn (14) has the same form as eqn (7) in the previous section, but is reproduced here for convenience and completeness. We have given the detailed derivation of this algorithm in the symmetric special case in Karhunen and Joutsensalo (1994). The derivation goes through quite similarly for the more general algorithm (14) provided that the ith error vector $\mathbf{e}_k(i)$ depends on the ith weight vector $\mathbf{w}_k(i)$ [i.e., the summation in eqn (15) contains the index i as was assumed in the previous section], and will not be repeated here.

The robust PCA algorithm (14)–(15) can again be applied both to the symmetric and hierarchic cases as in the previous section. Thus, in the symmetric case $I(i) = M$, the error vector (15) is the same

$$\mathbf{e}_k = \mathbf{x}_k - \mathbf{W}_k\mathbf{W}_k^T\mathbf{x}_k \tag{16}$$

for all the weight vectors $\mathbf{w}_k(i), i = 1, \ldots, M$, and eqn (14) can be written compactly

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k[\mathbf{x}_k g(\mathbf{e}_k^T)\mathbf{W}_k + g(\mathbf{e}_k)\mathbf{x}_k^T\mathbf{W}_k]. \tag{17}$$

In the hierarchic case $I(i) = i$, the error vectors (15) are different for each weight vector estimate $\mathbf{w}_k(i)$. In this case, eqn (14) estimates the robust counterparts of the principal eigenvectors $\mathbf{c}(i)$.

The first update term $\mathbf{w}_k(i)^T g[\mathbf{e}_k(i)]\mathbf{x}_k$ in the complete algorithm (14) is proportional to the same vector $\mathbf{x}_k$ for all the weight vectors $\mathbf{w}_k(i)$. Furthermore, we can assume that the average value of coefficient $\mathbf{w}_k(i)^T g[\mathbf{e}_k(i)]$ is close to zero, because the error vector $\mathbf{e}_k(i)$ should be relatively small after the initial convergence and its sign can be either positive or negative. Hence, this term can usually be neglected without committing much error. This approximation leads to a simpler algorithm

$$\mathbf{w}_{k+1}(i) = \mathbf{w}_k(i) + \mu_k \mathbf{x}_k^T \mathbf{w}_k(i) g[\mathbf{e}_k(i)]$$
$$= \mathbf{w}_k(i) + \mu_k y_k(i) g[\mathbf{e}_k(i)], \quad (18)$$

In the symmetric special case, eqn (18) becomes the approximation of eqn (17):

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k g(\mathbf{e}_k)\mathbf{x}_k^T \mathbf{W}_k = \mathbf{W}_k + \mu_k g(\mathbf{e}_k)\mathbf{y}_k^T. \quad (19)$$

It is interesting to compare the algorithms (18) and (8) derived from different optimization criteria. They closely resemble each other, the only difference being that in eqn (18) the nonlinearity $g(t)$ is applied to the error $\mathbf{e}_k(i)$, whereas in eqn (8) it is applied to the response $y_k(i) = \mathbf{x}_k^T \mathbf{w}_k(i)$. However, this has the important consequence that if the network is taught using the approximative algorithm (18) that tries to minimize robust representation error, the final input–output mapping is still linear. For the generalized variance maximization algorithm (8), the outputs of the corresponding PCA-type network are nonlinear $g[\mathbf{x}_k^T \mathbf{w}_k(i)]$. Therefore, these algorithms yield, in general, somewhat different weight vectors.

4.3. Relationship to Standard PCA Learning Rules

In this subsection, we study more closely the relationship of the algorithms derived thus far to the well-known neural PCA learning rules. This can be done by setting $g(t) = t$ in the algorithms, which leads to standard PCA learning.

In this special case, eqns (18) and (8) coincide and estimate the same standard PCA solution. In particular, the robust PCA subspace algorithms (9) and (19) become the same as the well-known Oja's PCA subspace rule:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k \mathbf{e}_k \mathbf{y}_k^T = \mathbf{W}_k + \mu_k [\mathbf{I} - \mathbf{W}_k \mathbf{W}_k^T] \mathbf{x}_k \mathbf{x}_k^T \mathbf{W}_k. \quad (20)$$

Similarly, in the standard hierarchic case $I(i) = i$ in both eqns (18) and (8) coincides with the well-known Sanger's GHA algorithm:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k \{\mathbf{x}_k \mathbf{y}_k^T - \mathbf{W}_k \mathrm{UT}[\mathbf{y}_k \mathbf{y}_k^T]\}, \quad (21)$$

$\mathbf{y}_k = \mathbf{W}_k^T \mathbf{x}_k$. These results imply that we have actually derived Oja's PCA subspace rule and Sanger's GHA in two different ways: from the variance maximization problem using orthonormality constraints taken into account via Lagrange multipliers, and by minimizing the mean-square representation error. The first derivation yields exactly Oja's PCA subspace rule and Sanger's GHA. However, it is somewhat inaccurate in the sense that the expressions of the Lagrange multipliers are exactly valid only in the optimum, but are then used everywhere in the respective stochastic gradient algorithm (8).

These derivations clearly show that Oja's PCA subspace rule and GHA are approximative algorithms. Their relationship to the variance maximization and mean-square error minimization problems is exactly the same. The only difference is that Oja's PCA subspace rule corresponds to a completely symmetric network structure and Sanger's GHA to the standard hierarchic structure.

For nonlinear $g(t)$, the "optimal" robust stochastic gradient algorithm is eqn (14), which takes the form of eqn (17) in the symmetric case. In the linear special case $g(t) = t$, eqn (17) reduces to the "optimal" standard PCA subspace estimation algorithm

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k [\mathbf{x}_k \mathbf{e}_k^T \mathbf{W}_k + \mathbf{e}_k \mathbf{x}_k^T \mathbf{W}_k] \quad (22)$$

that has been derived independently by several authors (see Karhunen & Joutsensalo, 1994; Palmieri & Zhu, 1993). Its theoretical properties have been explored in Xu (1993), and some experimental comparisons to other standard PCA subspace rules are given in Palmieri and Zhu (1993). However, the respective hierarchic optimal algorithm and its relationship to Sanger's GHA are obviously new results, even in the linear special case. In fact, we have, for the first time to our knowledge, derived GHA and its robust generalizations from natural optimization criteria in Karhunen and Joutsensalo (1993b).

4.4. Nonlinear PCA Algorithms

Consider now briefly the more nonlinear versions of PCA-type algorithms. A heuristic way to get them is to require that the outputs of the neurons are always nonlinear $g[y(i)] = g[\mathbf{x}^T \mathbf{w}(i)]$ in the algorithms. Applied to eqn (8), this yields the nonlinear PCA algorithm ($i = 1, \ldots, M$)

$$\mathbf{w}_{k+1}(i) = \mathbf{w}_k(i) + \mu_k g[y_k(i)] \mathbf{b}_k(i), \quad (23)$$

which is otherwise similar to algorithm (8), but now the error vector defined by

$$\mathbf{b}_k(i) = \mathbf{x}_k - \hat{\mathbf{x}}_k(i) = \mathbf{x}_k - \sum_{j=1}^{I(i)} g[y_k(j)] \mathbf{w}_k(j) \quad (24)$$

contains nonlinearities when compared to eqn (7). More formally, one can show that eqn (23) is an approximative stochastic gradient algorithm for minimizing the mean-square representation error $E\{\|\mathbf{b}(i)\|^2\}$. $\mathbf{b}_k(i)$ is the instantaneous estimate of the error vector $\mathbf{b}(i)$ at step k. We have presented a detailed derivation in the symmetric special case $I(i) = M$ in Karhunen and Joutsensalo (1994). The same algorithm is mentioned in passing in Xu (1993) as a special case of his more general LMSER algorithm. The derivation for the more general error vector (24) is quite similar and is therefore omitted here.

In Karhunen and Joutsensalo (1994) we have actually derived the respective optimal algorithm for the nonquadratic criterion (13) and the error vector (24) in the symmetric special case. The general algorithm is rather complex, and yields as its special cases the robust

PCA subspace algorithm (17) and the nonlinear algorithm (23) in the symmetric case. The latter algorithm can be written compactly

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k \mathbf{b}_k g(\mathbf{y}_k^T), \tag{25}$$

where the error vector $\mathbf{b}_k$ is the same for all the neurons

$$\mathbf{b}_k = \mathbf{x}_k - \mathbf{W}_k g(\mathbf{W}_k^T \mathbf{x}_k). \tag{26}$$

The nonlinear approximative subspace algorithm (25) can be regarded as a straightforward nonlinear generalization of Oja's PCA subspace rule. It was first proposed heuristically in Oja et al. (1991) and is studied experimentally in Karhunen and Joutsensalo (1994).

The respective hierarchic algorithm is a direct nonlinear generalization of GHA. It is obtained from eqns (23) and (24) by choosing $I(i) = i$, and can be written in matrix form as

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k \{\mathbf{x}_k g(\mathbf{y}_k^T) - \mathbf{W}_k \mathrm{UT}[g(\mathbf{y}_k) g(\mathbf{y}_k^T)]\}. \tag{27}$$

Sanger (1989a) proposed this algorithm as a heuristic nonlinear extension of GHA and presented experimental results for a very specific "nonlinearity" $g(t) = 0$, $t < 0$; $g(t) = t$, $t > 0$. Now our considerations relate eqn (27) clearly to the mean-square representation error. This helps greatly in interpreting the results of the experiments and in understanding the properties of the nonlinear GHA algorithm (27). Again, it is possible to implement eqn (27) locally, because in the standard hierarchic case the error vector (24) can be computed from the recursion $\mathbf{b}_k(i) = \mathbf{b}_k(i-1) - g[y_k(i)]\mathbf{w}_k(i)$, $\mathbf{b}_k(\mathbf{0}) = \mathbf{x}_k$.

In eqn (24), the approximation $\hat{\mathbf{x}}_k(i)$ is linear with respect to the basis vectors $\mathbf{w}_k(j)$ of the expansion (weight vectors of the neurons), but the coefficients $g[y_k(j)]$ of the expansion (outputs of the neurons) are generally nonlinear. This kind of expansion looks at first sight slightly cumbersome. Its main advantage seems to be that the nonlinear coefficients implicitly take higher-order statistical information into account, and the outputs of the neurons become more independent than in standard PCA networks after convergence.

It should be noted that these nonlinear PCA algorithms yield in general something else than the standard PCA solution. The equivalence results of Bourlard and Kamp (1988) described earlier are not applicable, because the forward weight matrix $\mathbf{Q}$ in eqn (12) is constrained to be the same as the feedback matrix $\mathbf{W}$. However, especially for mild nonlinearities, the results can still be close to the respective PCA solution. Similarly, the weight vectors of different neurons estimated using these nonlinear PCA algorithms are typically not exactly orthogonal, but not far from orthogonality. This property can be explained by considering the mean-square error criterion $E\{\|\mathbf{b}(i)\|^2\}$. If some of the weight vectors $\mathbf{w}_k(j)$ in eqn (24) were close to some other weight vector in direction, the corresponding term $g[y_k(j)]\mathbf{w}_k(j)$ would diminish only slightly the mean-square error.

There exist several possibilities for obtaining results that differ more from the standard PCA. The first is naturally to use a more nonlinear back-propagation algorithm in autoassociative mode. A second possibility is to impose some meaningful additional constraints, and the third one to use an additional learning rule that will "wrest" the estimated weight vectors farther away from the PCA subspace.

5. EXPERIMENTAL RESULTS

In the following, we present some experimental results on the performance of the algorithms derived in the previous sections and compare them with standard PCA approaches.

5.1. Robust PCA Algorithms

We first study various robust PCA subspace algorithms in a simple but illustrative case where it is possible to compare the results with the theoretical PCA solution. The data vectors $\mathbf{x}_k$ were five-dimensional, and their components were independent zero mean random variables with different variances $\sigma^2(1), \ldots, \sigma^2(5)$. Then the exact covariance matrix $\mathbf{C}$ becomes diagonal, and its eigenvalues are directly the diagonal elements (variances) $\sigma^2(i)$. The ith element of the principal eigenvector corresponding to the eigenvalue $\sigma^2(i)$ is $+1$ (or -1), and its other elements equal to zero.

In the first case, the random variables were purely Gaussian with variances $\sigma^2(1) = 5.0$, $\sigma^2(2) = 3.0$, $\sigma^2(3) = 1.0$, $\sigma^2(4) = 0.4$, and $\sigma^2(5) = 0.2$. We consider the estimation of a two-dimensional PCA subspace, which is now defined by the eigenvectors $\mathbf{c}(1) = [1, 0, 0, 0, 0]^T$ and $\mathbf{c}(2) = [0, 1, 0, 0, 0]^T$ corresponding to $\lambda(1) = 5.0$ and $\lambda(2) = 3.0$.

In the second case, each component of the data vectors came from the same Gaussian distribution as before with probability 0.9, but with probability 0.1 from a uniform distribution in the interval $[-10, 10]$. The uniform distribution can be used as a model for impulsive noise or outliers. The eigenvalues of $\mathbf{C}$ are now $0.9\lambda(i) + a$, where $a = 0.1(20)^2/12 \approx 3.33$ is the additional variance due to the uniform distribution. Because $\mathbf{C}$ is diagonal and the mutual order of its eigenvalues does not change, the theoretical principal eigenvectors and PCA subspace remain the same as in the first case.

In the simulations, we generated 300 data vectors in both the cases, and used the same data set several times in the gradient algorithms for achieving final convergence. The gain parameter μ_k was a small constant (0.015) in the beginning and then decreased slowly. The following methods were compared:

- The batch estimation method, in which the data covariance matrix $\mathbf{C}$ is first estimated from the available K (zero-mean) data vectors: $\hat{\mathbf{C}} = 1/K \sum_{k=1}^{K} \mathbf{x}_k \mathbf{x}_k^T$. Then the principal eigenvectors of $\hat{\mathbf{C}}$ are computed using standard techniques.
- Oja's PCA subspace rule [i.e., eqn (20)].
- Robust variance maximization algorithm (9) with the nonlinearity $g(t) = \tanh(t)$.
- Optimal robust error minimization algorithm (17) with the nonlinearity $g(t) = \log(1 + 5t)$. [This worked somewhat better than the $\tanh(t)$ nonlinearity.]
- Approximative robust error minimization algorithm (19) using $g(t) = \tanh(t)$.

We compared the estimated PCA subspaces to the theoretical one in terms of the SVD-based procedure given in Section 12.4 in Golub and Van Loan (1983). In this method, the closeness of two subspaces is measured by determining the principal angles between them. Table 1 shows the angles for purely Gaussian data (first case), and Table 2 for the second case of mixed Gaussian and uniformly distributed data. The estimated PCA subspace is the better the closer the respective angles are to zero. For increasing statistical reliability, the simulations were repeated for 100 independent realizations of the above-described data set. The tables represent averages of these simulations.

When the data were purely Gaussian (Table 1), all the methods performed well and yielded a good estimate for the theoretical PCA subspace. Table 2 shows that in the case of mixed Gaussian and uniformly distributed data (modeling the existence of impulsive noise), the approximative robust error minimization algorithm (19) was clearly superior compared to the others. However, a closer examination revealed that in most individual simulations the corresponding optimal algorithm (17) performed excellently: the second principal angle θ_2 is typically less than 2°. But in about 14% of the realizations $w(2)$ converged to a wrong (orthogonal) subspace, yielding $\theta_2 \approx 90°$. The approximation (19) is clearly more reliable, leading to better average results. Recently, Palmieri and Zhu (1993) have made similar observations about the sensitivity of the optimal subspace algorithm (17) to local minima in the linear special case.

TABLE 1
Principal Angles (in Degrees) Between Theoretical and Estimated PCA Subspace for Purely Gaussian Data

Algorithm	θ_1	θ_2
Batch method	1.5	3.8
Oja's PCA subspace rule	1.5	3.8
Alg. (9), $g(t) = \tanh(t)$	1.5	4.1
Alg. (17), $g(t) = \log(1 + 5t)$	1.1	8.5
Alg. (19), $g(t) = \tanh(t)$	1.1	3.9

TABLE 2
Principal Angles (in Degrees) Between Theoretical and Estimated PCA Subspace for Mixed Gaussian and Uniformly Distributed Data

Algorithm	θ_1	θ_2
Batch method	4.8	22.8
Oja's PCA subspace rule	4.6	24.1
Alg. (9), $g(t) = \tanh(t)$	5.8	21.8
Alg. (17), $g(t) = \log(1 + 5t)$	0.7	30.5
Alg. (19), $g(t) = \tanh(t)$	1.1	8.5

The robust variance maximization algorithm (9) performed only slightly better than the batch method or standard Oja's PCA subspace rule in impulsive noise. However, comparing it with the theoretical PCA subspace is not quite fair, because the optimal solution of the variance maximization criterion (1) is not necessarily the same. An indication of this is that changing the parameters in eqn (9) affected only slightly the results in our simulations. In general, scaling of the $\tanh(t)$ function should be done suitably so that it suppresses the outliers but not too much the original data.

The hierarchic versions of robust PCA algorithms behaved qualitatively similar to their subspace counterparts in the respective simulations. When the data vectors were purely Gaussian, robust GHA (10) with tanh and log-type nonlinearities $g(t)$ and the approximative hierarchic error minimization algorithm (18) with $I(i) = i$ yielded equally good estimates of the principal eigenvectors $\mathbf{c}(i)$ as the batch estimation method after convergence. For mixed Gaussian and uniformly distributed data, eqn (18) clearly performed best. The average normalized projections of the two first batch eigenvector estimates $\hat{\mathbf{c}}(1)$ and $\hat{\mathbf{c}}(2)$ onto the corresponding exact PCA eigenvectors $\mathbf{c}(1)$ and $\mathbf{c}(2)$ were 0.952 and 0.891, respectively. When eqn (18) was used, the respective projections were 0.976 and 0.965. The optimal algorithm (14) often performed very well but converged more easily than eqn (18) towards a wrong eigenvector. Robust GHA again yielded roughly equally good results as the batch method.

If robust counterparts of the principal eigenvectors are not necessarily needed, it is usually better to use the symmetric subspace algorithms. This is because they give more accurate estimates of the PCA subspace as a whole and have somewhat better stability properties. The stability issue is discussed in Karhunen (1994), where an exact stability bound is derived for Oja's PCA subspace rule and its robust generalization (9). In practice, robust and nonlinear PCA algorithms have better stability properties than the corresponding standard neural PCA algorithms if the (odd) nonlinearity $g(t)$ satisfies the condition $|g(t)| < |t|$, or grows less than linearly. On the other hand, nonlinearities growing faster than linearly easily cause stability problems in the algorithms and are not recommended.

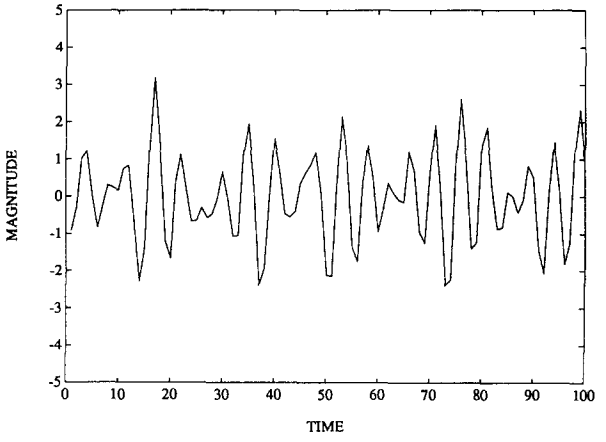

FIGURE 3. The test data: 100 samples of two sinusoids in colored Gaussian noise.

The superiority of the approximative robust algorithm (19) [or more generally eqn (18)] in the experiment described above seems at first surprising, because the batch procedure is usually regarded the best and is therefore used routinely in different applications. In Anderson (1958), the batch estimates of the eigenvalues and -vectors of the data covariance matrix $\mathbf{C}$ are proved to be optimal in the maximum likelihood sense, if the data vectors have a Gaussian distribution. However, the maximum likelihood estimates are distribution dependent (Sorenson, 1980), so that this result generally does not hold for other types of data. The batch estimates correspond to the quadratic mean-square error criterion, which weights heavily large errors. Therefore, their quality rapidly degrades in the presence of impulsive noise or outliers in the data.

The excellent performance of eqn (17) in the successful cases suggests that the optimal solution of the robust criterion eqn (13) coincides with the standard PCA solution on some specific conditions at least. The standard PCA solution is obtained by minimizing the mean-square error between $\mathbf{x}$ and its estimate $\hat{\mathbf{x}}$. There exist theoretical results stating that the mean-square estimates are in fact optimal for a larger class of error criteria provided that certain symmetry conditions are satisfied (see, e.g., Sorenson, 1980, pp. 158–165). It is obvious that these results can be applied to eqn (13), too, justifying our suggestion.

We have tested the derived robust algorithms in more realistic cases using higher-dimensional data. When we previously studied, in the context of sinusoidal frequency estimation, the performance of the nonlinear generalizations of Oja's PCA subspace rule suggested in Oja et al. (1991), we found that the symmetric variance maximization algorithm (9) with $g(t) = \tanh(t/\alpha)$ tolerated somewhat better strong or impulsive noise than standard Oja's PCA subspace rule (Karhunen & Joutsensalo, 1992).

A problem related to the estimation of sinusoidal frequencies is directions-of-arrival estimation in array processing. In both these problems, PCA subspace can be used for estimating the unknown directions or sinusoidal frequencies. The background theory of this rather specialized topic can be found in Orfanidis (1988) and Therrien (1992). In the simulations with different algorithms described in Joutsensalo and Karhunen (1993), the optimal robust algorithm (17) with a log-type nonlinearity performed somewhat better than its linear PCA counterpart. In another comparison with similar data containing some impulsive noise, it gave about 25% more accurate results than Oja's standard PCA subspace rule with otherwise the same parameters. In Karhunen and Joutsensalo (1993a), we have presented an example of tracking slowly changing spatial frequencies, in which eqn (17) with the nonlinearity $g(t) = \text{sgn}(t)\log(1 + 10|t|)$ performs better than Oja's PCA subspace rule.

5.2. Nonlinear PCA Algorithms

As mentioned earlier, the main advantage of the nonlinear PCA algorithms over the respective linear ones is that the outputs of the nonlinear PCA network usually are more independent than in the linear case. Such outputs are in many cases more meaningful than the mutually uncorrelated variance maximizing outputs of a linear PCA network. In the following, this property is demonstrated using sinusoidal signals.

In the test example, the training data consisted of 100 samples $x(0), \ldots, x(99)$ of two real sinusoids in additive colored noise. The samples were generated from the formula

$$x(k) = A_1\cos(2\pi f_1 k + \theta_1) + A_2\cos(2\pi f_2 k + \theta_2) + v(k), \quad (28)$$

where the parameters of the sinusoids unknown to the network were: amplitudes $A_1 = 0.8$ and $A_2 = 1.2$, and normalized frequencies $f_1 = 0.17$ and $f_2 = 0.22$. The phases θ_1 and θ_2 were randomly chosen fixed numbers. The colored noise process $v(n)$ was generated from the autoregressive (AR) model $v(n) = 1.058v(n-1) - 0.81v(n-2) + u(n)$, where $u(n)$ is white Gaussian noise. The signal-to-noise ratio (SNR) was 5 dB. The power spectrum of this process has a rather disturbing peak frequency 0.15 (the poles are $0.9\exp[\pm j2\pi 0.15]$). The 15-dimensional data vectors $\mathbf{x}_k = [x(k), x(k+1), \ldots, x(k+14)]^T$ were collected from successive samples and were used several times for achieving convergence. The PCA subspace dimension (number of neurons) was $M = 4$, which is correct for this sinusoidal model (Orfanidis, 1988; Therrien, 1992).

After learning, test data (Figure 3) generated from eqn (28) using different realization of the noise process

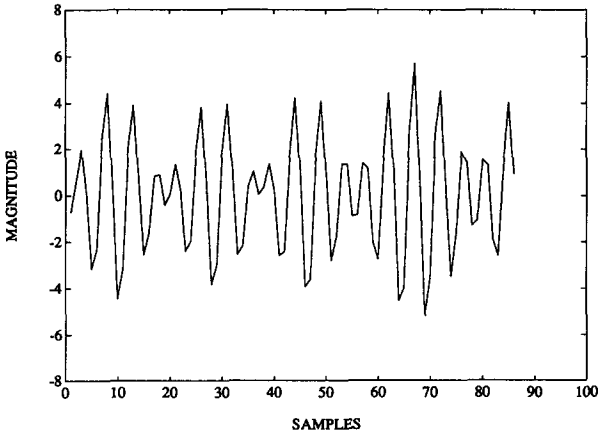

FIGURE 4. Output sequence of the first neuron of the linear PCA network trained by the standard GHA algorithm for the test data.

and phases of the sinusoids were presented to PCA-type networks. Figures 4 and 5 show the outputs $\mathbf{x}_k^T\mathbf{w}(1)$ and $\mathbf{x}_k^T\mathbf{w}(2)$ of the two first neurons for the linear hierarchic PCA network trained by the standard GHA algorithm. These outputs are almost similar linear combinations of the two sinusoids and AR noise as the test data, even though some of the noise has been filtered out. The outputs of the third and fourth neuron look similar and are not shown here. Figures 6 and 7 show the outputs $g[\mathbf{x}_k^T\mathbf{w}(1)]$ and $g[\mathbf{x}_k^T\mathbf{w}(2)]$ of the two first neurons for otherwise similar but nonlinear hierarchic PCA network trained by the nonlinear GHA algorithm (27) using $g(t) = \text{sign}(t)\log(5|t| + 1)$. Now the first neuron has clearly learned the stronger sinusoid corresponding to the frequency $f_2 = 0.22$, and the second neuron the weaker sinusoid with the frequency $f_1 = 0.17$. Due to the difficulty of the problem, the output sequences contain some AR noise. The outputs of the third and fourth neurons are almost similar to the first two ones with a phase shift of $\pi/4$. A tanh-type nonlinearity could be used, but it clips out the peaks of the sinusoids.

Quite recently, Sudjianto and Hassoun (1994) have given some theoretical justifications to this separation property. For other types than sinusoidal processes the separation properties of nonlinear PCA algorithms are not so good, because they try to minimize the mean-square error and not directly separate the subsignals. In the next section, we briefly discuss the more general signal separation problem.

In Karhunen and Joutsensalo (1994), we have presented experimental results for the respective subspace algorithm (25) in a similar but somewhat easier test case where the sinusoidal frequencies were not so closely spaced. Compared to the subspace algorithm (25), eqn (27) performs similarly but arranges the separated signals according to their power.

6. FROM UNCORRELATEDNESS TO INDEPENDENCE

In the following, we briefly consider generalization of the third information representation problem leading to the PCA solution mentioned in the Introduction, namely uncorrelatedness of coefficients after an orthonormal transform.

In a neural network environment, the requirement of uncorrelatedness usually means that the outputs $y(i) = \mathbf{w}(i)^T\mathbf{x}$, $i = 1, \ldots, M$, of a linear network must be mutually uncorrelated: $E\{y(i)y(j)\} = 0$, $i \neq j$. This immediately yields the general condition $\mathbf{w}(i)^T\mathbf{C}\mathbf{w}(j) = 0$ for zero-mean data vectors. It should be noted that if the number of neurons M is less than or equal to the dimension L of the data vectors $\mathbf{x}$, there usually exists an infinite number of possible linearly independent bases $\mathbf{w}(1), \ldots, \mathbf{w}(M)$ that satisfy the uncorrelatedness requirement (Jain, 1989; Palmieri, Zhu, & Chang, 1993). If the weight vectors $\mathbf{w}(i)$ are constrained to be mutually orthonormal, the uncorrelatedness requirement is satisfied if different weight vectors are linear combinations of mutually excluding sets of eigenvectors $\mathbf{c}(j)$ of $\mathbf{C}$. This is easy to see by representing the weight vectors in the basis of the eigenvectors $\mathbf{c}(j)$. If we finally impose a further constraint that the output powers (variances) $E\{y(i)^2\}$ of the M neurons must be maximal, the weight vectors $\mathbf{w}(i)$ become the principal eigenvectors $\mathbf{c}(1), \ldots, \mathbf{c}(M)$ of the data covariance matrix $\mathbf{C}$ (cf. Haykin, 1989, Section 11.2). Thus, the standard hierarchic PCA network yield uncorrelated outputs after convergence, but symmetric networks (estimating PCA subspace only) generally not.

A stronger requirement is that the outputs of the neural network should be statistically independent (or as independent as possible). If merely second-order statistics are taken into account, this reduces to the uncorrelatedness condition. Once again, Gaussian random

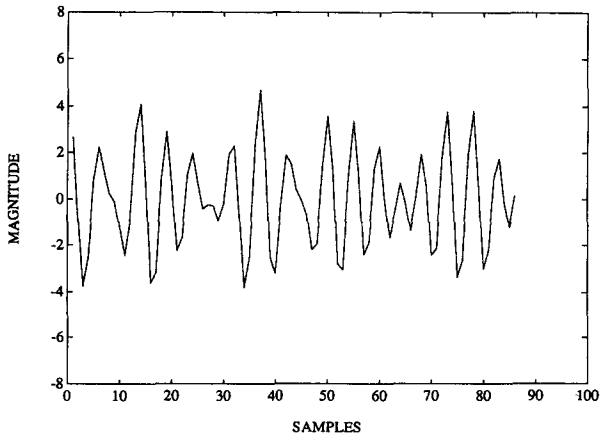

FIGURE 5. Output sequence of the second neuron of the linear PCA network trained by the standard GHA algorithm for the test data.

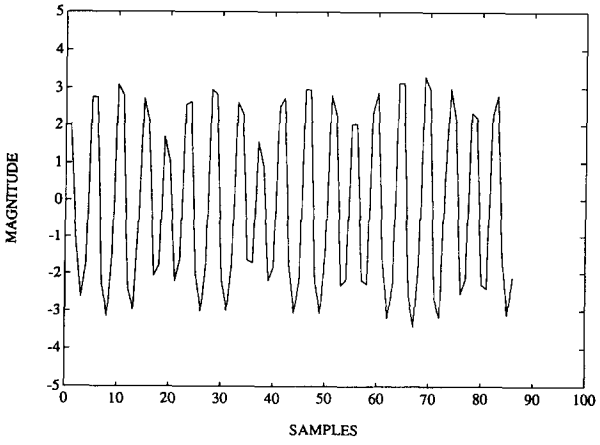

FIGURE 6. Output sequence of the first neuron for the nonlinear PCA network trained by the nonlinear GHA algorithm (27) for the test data.

variables are a special case, because for them independence is equivalent to uncorrelatedness.

Some neural algorithms for producing independent output signals have recently been proposed in context with the blind separation problem in signal processing. Jutten and Herault have introduced a somewhat heuristic neural algorithm (Jutten & Herault, 1991; Cichocki & Unbehauen, 1993a) for separating original source signals from their linear mixture, and Burel (1992) has considered a more general but difficult nonlinear problem. These neural approaches are related to an interesting and obvious extension of PCA called Independent Component Analysis (ICA or INCA). This concept is formally defined and its relationships to optimization criteria are discussed in Comon (1994). It is noteworthy that ICA does not require a nonlinear network for linear mixtures, but its basis vectors are usually nonorthogonal and the learning algorithm must contain some kind of nonlinearities to take into account higher-order statistics.

In spite of the success of Jutten and Herault's algorithm in practical examples, there are still several open research problems in extending neural PCA to ICA. For example, Jutten and Herault's algorithm requires in its basic form that there are available N different linearly independent linear mixtures (time series) of the original N source signals as an input to the network. Thus, this algorithm is not applicable if there is available only one linear combination (time series) of the original source signals as in our example of sinusoidal signals in colored noise.

The recent work of Österberg and Lenz (1994) is also related to these ideas. Starting from a determinant criterion leading to PCA subspace solution, they have developed in a neural network environment new optimization criteria that contain higher-order moments and yield more independent outputs than PCA.

7. ENTROPY-BASED CRITERIA

The fourth criterion yielding PCA as the optimal solution is based on the concept of entropy. Assume that the input signal $\mathbf{x}$ is Gaussian, and the output is linear, $\mathbf{y} = \mathbf{W}^T\mathbf{x}$ with probability density function $f(\mathbf{y})$, and that the weight vectors of the neurons are mutually orthonormal: $\mathbf{W}^T\mathbf{W} = \mathbf{I}$. The entropy defined by

$$H(\mathbf{y}) = -\int_{\mathbf{y}} f(\mathbf{y})\log[f(\mathbf{y})]\, d\mathbf{y} \tag{29}$$

is maximized when the weight vectors (columns of the weight matrix $\mathbf{W}$) are the principal eigenvectors $\mathbf{c}(1)$, ..., $\mathbf{c}(M)$ of the data covariance matrix $\mathbf{C}$ (Young & Calvert, 1974). Any orthonormal basis of the PCA subspace will also maximize $H(\mathbf{y})$.

Equation (29) is just one example of entropy-based criteria. There exist several somewhat different entropies and related information–theoretic criteria that can be used for measuring the effectiveness of information compression (see, e.g., Devijver & Kittler, 1982; Jain, 1989; Young & Calvert, 1974). PCA provides in many cases the optimal solution, but showing this typically requires specific assumptions such as Gaussianity of the data and orthonormality of the basis vectors as in eqn (29).

Linsker (1992, 1993) and Plumbley (1993a, b) have applied this kind of information–theoretic optimization principles mainly to linear PCA-type networks. If the specific assumptions are not satisfied, such linear networks can yield something else than a PCA solution. Recently, Linsker (1993) has extended his ideas to a mildly nonlinear network with interesting results. We will not discuss these information–theoretic optimization principles any more here, because excellent recent reviews (Haykin, 1994; Taylor & Plumbley, 1993) are available.

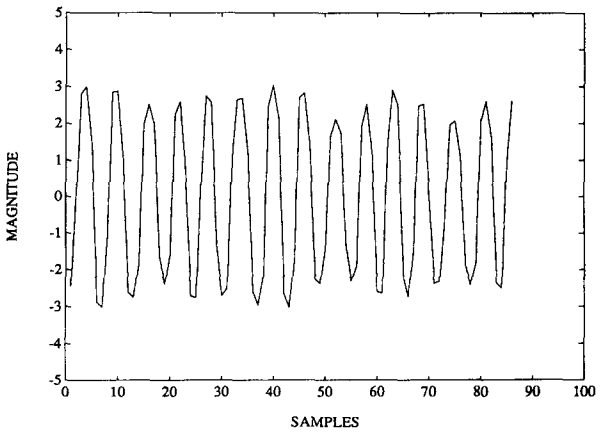

FIGURE 7. Output sequence of the second neuron for the nonlinear PCA network trained by the nonlinear GHA algorithm (27) for the test data.

8. CONCLUSIONS

It is by no means obvious in advance that the different information representation problems discussed in this paper have the same optimal solution. In a linear network their solution is essentially the same and is provided by PCA. This is mathematically a very nice result. However, it often requires, even in a linear network, some additional constraining assumptions. Furthermore, the preceding discussions have clearly shown that the optimality of PCA in these information representation problems results from taking into account second-order statistics (i.e., covariances) only. In a sense, an implicit assumption is then made that the data or outputs of neurons have roughly Gaussian distribution.

There is much more information in non-Gaussian data than just its second-order statistics (Nikias & Mendel, 1993). Ideally, this should be taken into account in information processing tasks for getting optimal results. A way to incorporate higher-order statistics, at least implicitly, into the computations is to use nonlinearities in PCA-type networks. Each of the information representation problems leading to standard PCA can be taken as a starting point of such a nonlinear generalization with its associated merits and drawbacks. One can then talk about nonlinear (or robust) PCA, or perhaps more appropriately about unsupervised learning beyond PCA. The solutions of these generalized information representation problems are usually different from each other and from PCA. Thus nonlinear PCA is a nonunique concept, unless the optimization problem or learning equations leading to it are defined appropriately. This viewpoint has been emphasized in a recent review paper by Xu (1994), too. From this viewpoint, linear PCA can be regarded as a degenerate case in which the optimal solutions coincide.

In this general framework, we have studied more closely generalization of two of the problems leading to standard PCA solutions. These are maximization of the output variances and minimization of the mean-square representation error in PCA-type networks. For the generalized problems, we have derived gradient-type learning algorithms both for symmetric and hierarchic networks. We have considered mildly nonlinear learning algorithms yielding robust PCA type solutions, and more nonlinear ones yielding what could be called nonlinear PCA. Several known PCA learning algorithms are obtained as special cases. In particular, well-known Sanger's generalized Hebbian algorithm and its robust and nonlinear generalizations are derived from natural optimization problems.

Even a linear PCA-type network can have a rather rich behavior (Palmieri & Zhu, 1993). There is already some evidence (e.g., Palmieri, 1994), that nonlinear PCA-type networks are able to yield interesting results that are qualitatively clearly different from standard PCA. This, together with the justifications presented in Section 2, motivates their further research.

REFERENCES

Anderson, T. W. (1958). *An introduction to multivariate statistical analysis.* New York: John Wiley.

Baldi, P., & Hornik, K. (1989). Neural networks for principal component analysis: Learning from examples without local minima. *Neural Networks,* **2,** 53–58.

Bekker, P., & de Leeuw, J. (1988). Relations between variants of non-linear principal components analysis. In J. L. A. van Rijckevorsel & J. de Leeuw (Eds.), *Component and correspondence analysis. Dimension reduction by function approximation,* Wiley Series in Probability and Mathematical Statistics (pp. 1–31). New York: John Wiley.

Bourlard, H., & Kamp, Y. (1988). Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics,* **59,** 291–294.

Burel, G. (1992). Blind separation of sources: A nonlinear neural algorithm. *Neural Networks,* **5,** 937–947.

Cichocki, A., & Unbehauen, R. (1993a). *Neural networks for optimization and signal processing.* New York: John Wiley.

Cichocki, A., & Unbehauen, R. (1993b). Robust estimation of principal components by using neural network learning algorithms. *Electronics Letters,* **29,** 1869–1870.

Comon, P. (1994). Independent component analysis—a new concept? *Signal Processing,* **36**(3), 287–314.

Devijver, P. A., & Kittler, J. (1982). *Pattern recognition: A statistical approach.* Englewood Cliffs, NJ: Prentice–Hall.

Gifi, A. (1990). *Nonlinear multivariate analysis.* Wiley Series in Probability and Mathematical Statistics. New York: John Wiley.

Golub, G. H., & Van Loan, C. F. (1983). *Matrix computations.* Baltimore, MD: Johns Hopkins Univ. Press.

Hastie, T., & Stuetzle, W. (1989). Principal Curves. *Journal of the American Statistical Association,* **84,** 502–516.

Haykin, S. (1989). *Modern filters.* New York: Macmillan.

Haykin, S. (1994). *Neural networks: A comprehensive foundation.* New York: IEEE Computer Society Press and Macmillan.

Hertz, J., Krogh, A., & Palmer, R. G. (1991). *Introduction to the theory of neural computation.* Reading, MA: Addison–Wesley.

Jain, A. K. (1989). *Fundamentals of digital image processing.* Englewood Cliffs, NJ: Prentice–Hall.

Joutsensalo, J., & Karhunen, J. (1993). Nonlinear multilayer principal component type subspace learning algorithms. In C. A. Kamm et al. (Eds.), *Neural networks for signal processing III* (pp. 68–77). New York: IEEE Press.

Jutten, C., & Herault, J. (1991). Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture. *Signal Processing,* **24,** 1–10.

Kambhatla, N., & Leen, T. K. (1993). A fast non-linear dimension reduction. *Proceedings of the* 1993 *IEEE International Conference on Neural Networks,* San Francisco, CA, March 1993 (pp. 1213–1218).

Karhunen, J. (1994). Stability of Oja's PCA subspace rule. *Neural Computation* **6,** 739–747.

Karhunen, J., & Joutsensalo, J. (1992). Learning of sinusoidal frequencies by nonlinear constrained Hebbian algorithms. In S. Y. Kung et al. (Eds.), *Neural networks for signal processing II* (pp. 39–48). New York: IEEE Press.

Karhunen, J., & Joutsensalo, J. (1993a). Learning of robust principal component subspace. *Proceedings of the International Joint Conference on Neural Networks,* Nagoya, Japan, October 1993 (pp. 2409–2412).

Karhunen, J., & Joutsensalo, J. (1993b). Nonlinear generalizations of principal component learning algorithms. *Proceedings of the International Joint Conference on Neural Networks,* Nagoya, Japan, October 1993 (pp. 2599–2602).

Karhunen, J., & Joutsensalo, J. (1994). Representation and separation of signals using nonlinear PCA type learning. *Neural Networks,* **7,** 113–127.

Kung, S. Y. (1993). *Digital neural networks.* Englewood Cliffs, NJ: Prentice–Hall.

Linsker, R. (1992). Local synaptic learning rules suffice to maximize mutual information in a linear network. *Neural Computation,* **4,** 691–702.

Linsker, R. (1993). Deriving receptive fields using an optimal encoding criterion. In S. J. Hanson, J. D. Cowan, & C. L. Giles (Eds.), *Neural information processing system* 5 (pp. 953–960). San Mateo, CA: Morgan Kaufmann.

Nikias, C. L., & Mendel, J. M. (1993). Signal processing with higher-order spectra. *IEEE Signal Processing Magazine,* **10,** 10–37.

Oja, E. (1992). Principal components, minor components, and linear neural networks. *Neural Networks,* **5,** 927–935.

Oja, E., & Karhunen, J. (1993). Nonlinear PCA: Algorithms and applications. *Report A18, September* 1993. Espoo, Finland: Helsinki University of Technology, Laboratory of Computer and Information Science.

Oja, E., Ogawa, H., & Wangviwattana, J. (1991). Learning in nonlinear constrained Hebbian networks. In T. Kohonen et al. (Eds.), *Artificial neural networks* (pp. 385–390). Amsterdam: North–Holland.

Oja, E., Ogawa, H., & Wangviwattana, J. (1992). Principal component analysis by homogeneous neural networks, part II: Analysis and extensions of the learning algorithms. *IEICE Transactions on Information and Systems* (Japan), E75-D, **3,** 376–382.

Orfanidis, S. J. (1988). *Optimum signal processing,* 2nd ed. New York: Macmillan.

Österberg, M., & Lenz, R. (1994). Unsupervised parallel feature extraction from first principles. In J. D. Cowan, G. Tesauro, & J. Alspector (Eds.), *Advances in neural information processing systems* 6 (pp. 136–143) San Francisco, CA: Morgan Kaufmann.

Palmieri, F. (1994). Hebbian learning and self-association in nonlinear neural networks. *Proceedings of the* 1994 *IEEE International Conference on Neural Networks,* Orlando, FL, June–July 1994 (pp. 1258–1263).

Palmieri, F., & Zhu, J. (1993). *Hebbian learning in linear neural networks: A review* (Tech. Rep. 5/93). Storrs, CT: University of Connecticut, Department of Electrical and Systems Engineering.

Palmieri, F., Zhu, J., & Chang, C. (1993). Anti-Hebbian learning in topologically constrained linear networks: A tutorial. *IEEE Transactions on Neural Networks,* **4,** 748–761.

Plumbley, M. D. (1993a). Efficient information transfer and anti-Hebbian neural networks. *Neural Networks,* **6,** 823–833.

Plumbley, M. D. (1993b). A Hebbian/anti-Hebbian network which optimizes information capacity by orthonormalizing the principal subspace. *Proceedings of the IEE Conference on Artificial Neural Networks,* Brighton, UK, May 1993 (pp. 86–90).

Sanger, T. D. (1989a). An optimality principle for unsupervised learning. In D. S. Touretzky (Ed.), *Advances in neural information processing systems* **1** (pp. 11–19). Palo Alto, CA: Morgan Kaufmann.

Sanger, T. D. (1989b). Optimal unsupervised learning in a single-layer linear feedforward network. *Neural Networks,* **2,** 459–473.

Sorenson, H. W. (1980). *Parameter estimation—principles and problems.* New York: Marcel Dekker.

Sudjianto, A., & Hassoun, M. (1994). Nonlinear Hebbian rule: A statistical interpretation. *Proceedings of the* 1994 *IEEE International Conference on Neural Networks,* Orlando, FL, June–July 1994 (pp. 1247–1252).

Taylor, J. G., & Plumbley, M. D. (1993). Information theory and neural networks. In J. G. Taylor (Ed.), *Mathematical approaches to neural networks* (pp. 307–340). Amsterdam: Elsevier Science Publishers.

Therrien, C. W. (1992). *Discrete random signals and statistical signal processing.* Englewood Cliffs, NJ: Prentice–Hall.

Xu, L. (1993). Least mean square error reconstruction principle for self-organizing neural-nets. *Neural Networks,* **6,** 627–648.

Xu, L. (1994). Theories for unsupervised learning: PCA and its nonlinear extensions. *Proceedings of the* 1994 *IEEE International Conference on Neural Networks,* Orlando, FL, June–July 1994 (pp. 1252–1257).

Xu, L., & Yuille, A. (1993). Self-organizing rules for robust principal component analysis. In S. J. Hanson, J. D. Cowan, & C. L. Giles (Eds.), *Advances in neural information processing systems* 5 (pp. 467–474). San Mateo, CA: Morgan Kaufmann.

Young, T. Y., & Calvert, T. W. (1974). *Classification, estimation, and pattern recognition.* New York: American Elsevier.