Pergamon

Neural Networks, Vol. 8, No. 4, pp. 525–535, 1995
Copyright © 1995 Elsevier Science Ltd
Printed in the USA. All rights reserved
0893-6080/95 $9.50 + .00

0893-6080(94)00108-1

CONTRIBUTED ARTICLE

Nonlinear Higher-Order Statistical Decorrelation by Volume-Conserving Neural Architectures

GUSTAVO DECO AND WILFRIED BRAUER*

Siemens AG

(*Received* 25 *February* 1994; *accepted* 10 *October* 1994)

Abstract—*A neural network learning paradigm based on information theory is proposed as a way to perform, in an unsupervised fashion, redundancy reduction among the elements of the output layer without loss of information from the sensory input. The model developed performs nonlinear decorrelation up to higher orders of the cumulant tensors and results in probabilistically independent components of the output layer. This means that we don't need to assume Gaussian distribution at either the input or the output. The theory presented is related to the unsupervised learning theory of Barlow, which proposes redundancy reduction as the goal of cognition. When nonlinear units are used (sigmoid or higher-order pi-neurons), nonlinear principal component analysis is obtained. In this case, nonlinear manifolds can be reduced to minimum dimension manifolds. If such units are used the network performs a generalized principal component analysis in the sense that non-Gaussian distributions can be linearly decorrelated and higher orders of the correlation tensors are also taken into account. The basic structure of the architecture involves a general transformation that is volume conserving and therefore the entropy, yielding a map without loss of information. Minimization of the mutual information among the output neurons eliminates the redundancy between the outputs and results in statistical decorrelation of the extracted features. This is known as factorial learning. To sum up, this paper presents a model of factorial learning for general nonlinear transformations of an arbitrary non-Gaussian (or Gaussian) environment with statistically nonlinearly correlated input. Simulations demonstrate the effectiveness of this method.*

Keywords—Nonlinear decorrelation, Volume-conserving architectures, Factorial learning.

1. INTRODUCTION

In the last few years there has been an increasing interest in theories and models for unsupervised detection of statistical correlations in a sensorial environment. One of the most important theories of feature extraction is the one proposed by Barlow (1959, 1989). Barlow describes the process of cognition as a preprocessing of the sensorial information performed by the nervous system to extract the statistically relevant and independent features of the inputs without losing information. This means that the brain should statistically decorrelate the extracted information. As a learning strategy, Barlow (1959, 1989) 2nd Barlow et al (1989) formulated the principle of *redundancy reduction.* The term redundancy refers to the statistical dependence between the components involved, and therefore the principle of learning by redundancy reduction tries to find a transformation such that the transformed components are as statistical independent as possible. This kind of learning is called *factorial learning.* In the binary case, factorial learning leads to *factorial codes* (i.e., codes with no redundancy). Recently, Atick and Redlich (1990, 1992) and Redlich (1993a,b) concentrated on the original idea of Barlow, yielding a very interesting formulation of early visual processing and factorial learning. Redlich (1993b) reduces redundancy at the input by using a network structure that is a reversible cellular automation and therefore guarantees the conservation of information in the transformation between input and output.

In the linear case, Obradovic and Deco (1993) apply Barlow's principle and derive a learning rule that performs principal component analysis (PCA). In this paper, PCA is derived as linear transformation that conserves and minimizes the mutual information between the outputs in order to decorrelate them. This point of

Acknowledgement: The authors would like to give very special thanks to one of the reviewers for his useful comments and careful examination of the manuscript.

* Technische Universität München, Institut für Informatik, Munich, Germany.

Requests for reprints should be sent to Gustavo Deco, Siemens AG, Corporate Research and Development, ZFE ST SN 41, Otto-Hahn-Ring 6, 81739 Munich, Germany.

view is an alternative to the "infomax" principle presented by Linsker (1988, 1989, 1992), who essentially proposes a learning rule that maximizes the mutual information between input and output layer. In the linear case, the infomax principle is related to PCA, given that deterministic networks are used (no noise on the output) and the covariance matrix of the input noise is diagonal (Földiak 1989, Barlow & Földiak, 1989). Rubner and Tavan (1989) proved that a network composed of linear neurons can be trained to perform PCA, if the synaptic adaptation is Hebbian in a vertical sense (i.e., from inputs to outputs), and anti-Hebbian for the inhibitory lateral connections between the output units. The Hebbian and anti-Hebbian forms of the learning rules can be derived from first principles by applying information theoretic concepts (Linsker, 1988; Kuehnel & Tavan, 1990). The Hebbian learning rule postulates that the strength of a synaptic connection can be adjusted, if its level of activity changes. An active synapse, which repeatedly triggers the activation of its postsynaptic neuron, will grow in strength, whereas others will gradually weaken. In other words, the plasticity of a synapse w with presynaptic stimulus x and postsynaptic activation y is modified (learns) according to $\Delta w = \eta x$. The modification is then proportional to the post- and presynaptic activation value. To introduce inhibitory processes during learning, sometimes an anti-Hebbian learning rule is introduced by $\Delta w = -\eta x$, which indicates the antiproportional relation between synapses strength increase and pre- and postsynaptic activation values. In most of the learning paradigms analyzed in the artificial neural network literature, identification of Hebbian and anti-Hebbian terms is of fundamental relevance due to the biologically motivation of these rules.

Some nonlinear extensions of PCA for decorrelation of sensorial input signals were recently introduced. These follow very closely Barlow's original ideas of unsupervised learning. Deco and Parra (1993) defined a stochastic neural network of Ising spins that performs statistical decorrelation of Boolean outputs by nonlinear transformations using information theoretic concepts. Atick and Redlich (1992) and especially the two excellent papers of Redlich (1993a, b) use similar information theoretic concepts and reversible cellular automata architectures to define how nonlinear decorrelation can be performed. Taylor and Coombes (1993) presented an extension of Oja's learning rule for higher-order neural networks.

The aim of our work is to formulate a neural network architecture and a novel learning paradigm that performs Barlow's unsupervised learning in the most general fashion. The basic idea is to define an architecture that assures perfect transmission without loss of information. Consequently, the nonlinear transformation defined by the neural architecture is always bijective. The architecture performs a volume-conserving transformation (i.e., determinant of the Jacobian matrix is equal to 1). As a particular case we can derive the reversible cellular automata architecture proposed by Redlich (1993b) and in the linear case an architecture similar to the one formulated by Rubner and Tavan (1989) for PCA. The learning paradigm is defined so that the components of the output signal are statistically decorrelated. Due to the fact that the output distribution is not necessarily Gaussian, even if the input is Gaussian, we perform a cumulant expansion of the output distribution and find the rules that should be satisfied by the higher-order correlation tensors in order to be decorrelated. This paper generalizes previous work on factorial learning in two senses: the architecture performs general nonlinear transformation without loss of information, and the decorrelation is performed without assuming Gaussian distributions.

In Section 2 we introduce the theoretical formalism. Section 3 introduces the architecture and learning rule for general activation functions. Section 4 defines the learning paradigm for sigmoid activation functions, for higher-order neural networks, and for linear networks. In Section 5 we present results and simulations. The last section is dedicated to the conclusions. The appendix presents the derivation of multidimensional cumulant expansions.

2. THEORETICAL FORMALISM

Let us consider an input vector $\vec{x} = (x_1, x_2, \ldots, x_d)$ of dimensionality d with components distributed according to the probability distribution $P(\vec{x})$, which is not factorial (i.e., the components of $\vec{x}$ are statistically correlated) and therefore

$$P(\vec{x}) \neq \prod_i^d P(x_i). \tag{1}$$

The goal of Barlow's unsupervised learning rule is to find a transformation

$$\vec{y} = \vec{F}(\vec{x}) \tag{2}$$

such that the components of the output vector d-dimensional $\vec{y} = (y_1, y_2, \ldots, y_d)$ are "*statistically*" decorrelated (i.e., independent).

This means that the probability distributions of the components y_i are independent and therefore we have

$$P(\vec{y}) = \prod_i^d P(y_i). \tag{3}$$

The objective of factorial learning is to find a neural network that performs the transformation $\vec{F}(\ )$ such that the joint probability distribution $P(\vec{y})$ of the output signals is factorized as in eqn (3). To implement factorial learning, the information contained in the input should be transferred to the output neurons without loss but, the probability distribution of the output neurons should be statistically decorrelated. Let us now define these facts from the information theory perspective. The first aspect is to assure the entropy is conserved, that is,

$$H(\vec{x}) = H(\vec{y}) \tag{4}$$

where the symbol $H(\vec{a})$ denotes the entropy of $\vec{a}$ and $H(\vec{a}/\vec{b})$ the conditional entropy of $\vec{a}$ given $\vec{b}$. One way to achieve this goal is to construct an architecture that independently of its synaptic parameters always satisfies eqn (4). Thus, the architecture will conserve information or entropy. The transmitted entropy satisfies

$$H(\vec{y}) \leq H(\vec{x}) + \int P(\vec{x}) \ln\left(\det\left(\frac{\partial \vec{F}}{\partial \vec{x}}\right)\right) d\vec{x} \tag{5}$$

where equality holds only if $\vec{F}$ is bijective (i.e., reversible). Conservation of information and bijectivity is assured if the neural transformation conserves the volume, which mathematically can be expressed by the fact that the Jacobian of the transformation should have determinant unity, that is,

$$\det\left(\frac{\partial \vec{F}}{\partial \vec{x}}\right) = 1 \tag{6}$$

where $(\partial \vec{F})/(\partial \vec{x})$ is the Jacobian matrix of the neural network transformation. We formulate in Section 3 an architecture that always conserves the volume, independent of the values of its weights. As a particular case, when the dimensionality of the input space is even, it is also possible to define a volume-conserving transformation that is at the same time "symplectic" (Abraham & Marsden, 1978). A symplectic transformation satisfies the following equation,

$$S_d = \left(\frac{\partial \vec{F}}{\partial \vec{x}}\right)^T S_d \left(\frac{\partial \vec{F}}{\partial \vec{x}}\right) \tag{7}$$

where

$$S_d = \begin{bmatrix} 0 & -I \\ I & 0 \end{bmatrix} \tag{8}$$

and I is the unity matrix. By using eqn (7) it is easy to prove that the composition of symplectic transformations is also symplectic. The Jacobian associated with such an architecture has the form

$$\left(\frac{\partial \vec{F}}{\partial \vec{x}}\right) = \begin{bmatrix} I & W \\ 0 & I \end{bmatrix}, \tag{9}$$

which satisfies eqn (7). The fact that this architecture conserves the information has its correspondence in statistical mechanics in the conservation of entropy for conservative systems.

Let us now concentrate on the main aspect of factorial learning, namely the decorrelation of the output components. Here the problem is to find a volume-conserving transformation that satisfies eqn (3). The major problem is that the distribution of the output signal will not necessarily be Gaussian. Therefore, it is impossible to use the technique of minimizing the mutual information between the components of the output as done by Redlich (1993b) or Deco and Parra (1993); then in the non-Gaussian case we do not have an analytical expression for the calculation of the entropies that are involved in the minimization of the mutual information between the output components. In fact, Redlich (1993b) and Deco and Parra (1993) minimize only an upper bound of the entropies given by the minimization of the sum of the output variances. The second theorem of Gibbs assures that the entropy of a distribution has an upper bound given by the entropy of a Gaussian distribution with the same variance as the original one. Using this theorem, Deco and Parra (1993) reduce the problem of statistical decorrelation to the problem of minimizing the upper bound of the entropies [i.e., the sum of the output variances (which is the entropy of a sum of Gaussian distributions)].

An alternative way to decorrelate non-Gaussian distributions is to expand the distribution in higher orders of the correlation matrix and impose the independence condition of eqn (3). To achieve this we propose to use a cumulative expansion (Papoulis, 1991) of the output distribution.

Let us define the Fourier transform of the output distribution,

$$\phi(\vec{K}) = \int d\vec{y}\, \exp[i(\vec{K} \cdot \vec{y})] P(\vec{y}) \tag{10}$$

$$\phi(K_i) = \int dy_i \exp[i(K_i \cdot y_i)] P(y_i) \tag{11}$$

where $\vec{K} = (K_1, \ldots, K_d)$ denote the coordinates in the Fourier transformed space and $i = \sqrt{-1}$. The cumulative expansion of a distribution is (Papoulis, 1991)

$$\phi(\vec{K}) = \exp\left[\sum_{n=1}^{\infty} \frac{i^n}{n!} \sum_{i_1, i_2 \ldots, i_n}^{n} \varkappa_{i_1, \ldots, i_n} K_{i_1} K_{i_2} \ldots K_{i_n}\right] \tag{12}$$

$$\phi(K_i) = \exp\left[\sum_{n=1}^{\infty} \frac{i^n}{n!} \varkappa_i^{(n)} K_i^n\right]. \tag{13}$$

The one-dimensional cumulants $\varkappa_i^{(n)}$ of eqn (13) are given in Reichl (1980). The first four one-dimensional cumulants are

$$\varkappa_i^{(1)} = \bar{y}_i \tag{14}$$

$$\varkappa_i^{(2)} = C_i^{(2)} - \bar{y}_i^2 \tag{15}$$

$$\varkappa_i^{(3)} = C_i^{(3)} - 3C_i^{(2)}\bar{y}_i + 2\bar{y}_i^3 \tag{16}$$

$$\varkappa_i^{(4)} = C_i^{(4)} - 3(C_i^{(2)})^2 - 4C_i^{(3)}\bar{y}_i + 12C_i^{(2)}\bar{y}_i^2 - 6\bar{y}_i^4 \tag{17}$$

where $C_i^{(n)}$ are the one-dimensional higher-order moments $C_i^{(n)}$ and are given by

$$C_i^{(n)} = \int dy_i' P(y_i')(y_i')^n. \quad (18)$$

Let us also define the multidimensional correlation tensors $C_{i\ldots j}$ as

$$C_{i\ldots j} = \int d\vec{y}\,' P(\vec{y}\,') y_i' \ldots y_j'. \quad (19)$$

The Appendix generalizes this to the case of multidimensional cumulants $\varkappa_{i_1,\ldots,i_n}$ as in eqn (12). It is clear from the definition that $\varkappa_i^{(n)} = \varkappa_{i_1,\ldots,i_n}$ when $i_1 = \ldots = i_n = i$.

In the Fourier space the independence condition is given by (Papoulis, 1991)

$$\phi(\vec{K}) = \prod_i \phi(K_i), \quad (20)$$

which is equivalent to

$$\ln(\phi(\vec{K})) = \ln\left(\prod_i \phi(K_i)\right) = \sum_i \ln(\phi(K_i)). \quad (21)$$

Substituting eqns (12) and (13) into eqn (21), we obtain that in the case of independence the following equality is satisfied

$$\sum_{n=1}^{\infty} \frac{i^n}{n!} \sum_{i_1,i_2\ldots,i_n}^{n} \varkappa_{i_1,\ldots,i_n} K_{i_1} K_{i_2} \ldots K_{i_n} = \sum_i^d \left(\sum_{n=1}^{\infty} \frac{i^n}{n!} \varkappa_i^{(n)} K_i^n \right). \quad (22)$$

In both expansions we will only consider the first four cumulants $\varkappa_{i_1}$, $\varkappa_{i_1,i_2}$, $\varkappa_{i_1,i_2,i_3}$, valid.$\varkappa_{i_1,i_2,i_3,i_4}$. Using the cumulants expression,

$$\varkappa_i = 0, \quad (23)$$

$$\varkappa_{ij} = C_{ij}, \quad (24)$$

$$\varkappa_{ijk} = C_{ijk}, \quad (25)$$

$$\varkappa_{ijkl} = C_{ijkl} - C_{ij}C_{kl} - C_{ik}C_{lj} - C_{il}C_{jk} \quad (26)$$

derived in the Appendix and the one-dimensional cumulants [eqns. (14)–(17)] after the transformation (A.7), eqn (22) becomes

$$-\tfrac{1}{2} \sum_{i,j} K_i K_j \{ C_{ij} - C_i^{(2)} \delta_{ij} \} - (\tfrac{1}{6}) i \sum_{i,j,k} K_i K_j K_k$$
$$\times \{ C_{ijk} - C_i^{(3)} \delta_{ijk} \} + \tfrac{1}{24} \sum_{i,j,k,l} K_i K_j K_k K_l$$
$$\times \{ (C_{ijkl} - 3C_{ij}C_{kl}) - (C_i^{(4)} - 3(C_i^{(2)})^2)\delta_{ijkl} \} = 0. \quad (27)$$

Equation (27) should be satisfied for all values of $\vec{K}$. The $\delta_{ij\ldots l}$ denotes Kroenecker's delta given by

$$\delta_{ij\ldots l} = 1 \quad \text{if } i = j = \ldots = l$$

$$\delta_{ij\ldots l} = 0 \quad \text{otherwise.} \quad (28)$$

Due to the fact that eqn (27) should be satisfied for all $\vec{K}$, all coefficients in each summation of eqn (27) must be zero. This means that

$$C_{ij} - C_i^{(2)} \delta_{ij} = 0, \quad \forall i, j \quad (29)$$

$$C_{ijk} - C_i^{(3)} \delta_{ijk} = 0, \quad \forall i, j, k \quad (30)$$

$$(C_{ijkl} - 3C_{ij}C_{kl}) - (C_i^{(4)} - 3(C_i^{(2)})^2)\delta_{ijkl} = 0, \quad \forall i, j, k, l \quad (31)$$

or equivalently,

$$C_{ij} = 0, \quad \text{if} \quad (i \neq j) \quad (32)$$

$$C_{ijk} = 0, \quad \text{if} \quad (i \neq j \vee i \neq k) \quad (33)$$

$$C_{ijkl} = 0, \quad \text{if} \quad (\{ i \neq j \vee i \neq k \vee i \neq l \} \wedge \neg L) \quad (34)$$

$$C_{iijj} - C_{ii}C_{jj} = 0, \quad \text{if} \quad (i \neq j). \quad (35)$$

In eqn (34) L is the logical expression

$$L = \{ (i = j \wedge k = l \wedge j \neq k) \vee (i = k \wedge j = l \wedge i \neq j) \vee (i = l \wedge j = k \wedge i \neq j) \}, \quad (36)$$

which excludes the cases considered in eqn (35). The conditions of independence given by eqns (32)–(35) can be achieved by minimization of the cost function

$$E = \alpha \sum_{i<j} C_{ij}^2 + \beta \sum_{i<j\leq k} C_{ijk}^2 + \gamma \sum_{(i<j\leq k\leq l)\wedge\neg L} C_{ijkl}^2 + \lambda \sum_{i<j} (C_{iijj} - C_{ii}C_{jj})^2 \quad (37)$$

where α, β, γ, λ are the inverse of the number of elements in each summation, respectively.

In conclusion, minimizing the cost given by eqn (37) with a volume conserving network, we achieve nonlinear decorrelation of non-Gaussian distributions.

It is very easy to test whether a factorized probability distribution [eqn (3)] satisfies eqns (32)–(35).

As a particular case, if only second-order terms are used in the cumulant expansion, the learning rule reduces to eqn (32), which expresses nothing more than the diagonalization of the second-order covariance matrix. In this case, by taking the inverse transform of the cumulant expansion of the Fourier transform of the distribution

$$P(y_i) = \left(\frac{1}{2\pi}\right) \int dK_i \exp[-i(K_i \cdot y_i)] \phi(K_i) \quad (38)$$

and substituting for $\phi(K_i)$ the first two terms of the cumulative expansion (13), then we obtain

$$P(y_i) = \int dK_i \exp[-i(K_i \cdot y_i)](\exp[i\varkappa_i^{(1)} K_i - (\varkappa_i^{(2)} K_i^2)])$$

$$= \frac{\exp\left(-\dfrac{(y_i - \varkappa_i^{(1)})^2}{2\varkappa_i^{(2)}}\right)}{\sqrt{2\pi\varkappa_i^{(2)}}}, \quad (39)$$

which is a Gaussian distribution with mean $\varkappa_i^{(1)}$ and variance $\varkappa_i^{(2)}$. Diagonalization of the covariance matrix decorrelates statistically the components of the output only if we assume a Gaussian distribution of the outputs. In general, the distribution of the output is not Gaussian and therefore higher orders of the cumulant expansion should be taken into account, yielding the learning rule conditions eqns (32)–(35) (up to fourth order, generalization to higher orders is straightforward). In the case of Gaussian distribution, minimization of the sum of the variances at each output leads to statistically decorrelation. This fact has a nice information theoretic background, namely the minimization of the mutual information between the output components. Statistical independence as expressed in eqn (3) is equivalent to (Atick & Redlich, 1992)

$$M = \sum_{i=1}^{d} I(x_1, \ldots, x_{i-1}; x_i) = \sum_{j} H(y_j) - H(\vec{y}) = 0 \quad (40)$$

where M is the mutual information between the output components and $I(x_1, \ldots, x_{i-1}; x_i)$ is the mutual information between the components $x_1, \ldots, x_{i-1}$ and x_i. This means that to minimize the redundancy at the output we minimize the mutual information between the different components of the output vector. Due to the fact that the volume-conserving structure of the neural network conserves the entropy [i.e., $H(\vec{y}) = H(\vec{x}) = \text{constant}$], the minimization of M reduces to the minimization of $\Sigma_j H(y_j)$. Using eqn (38) we can reduce the problem of statistical decorrelation in the Gaussian case to the problem of minimizing the sum of the output variances, that is,

$$\begin{aligned}\min(M) &\equiv \min\left(\sum_{i=1}^{d} \ln(\varkappa_i^{(2)})\right) \\ &\equiv \min(\mathrm{Tr}(\ln(\mathrm{diag}(\varkappa_{ij})))). \quad (41)\end{aligned}$$

It is interesting to note that eqn (41) implies minimization of the product of the diagonal elements of $\varkappa_{ij}$. Wegner's theorem gives us a lower bound for this product (Bodewig, 1956),

$$\det([\varkappa_{ij}]) \leq \prod_{j} \varkappa_{jj} \quad (42)$$

where equality holds iff the covariance matrix of the outputs is diagonal.

3. VOLUME-CONSERVING ARCHITECTURE AND LEARNING RULE

In this section we define a neural network architecture that is volume conserving and therefore can be used for the implementation of the learning rules described in the last section. Figure 1a shows the basic architecture of one layer. The dimensionality of input and output layers is the same and equal to d. A similar architecture was proposed by Redlich (1993b) using the theory of reversible cellular automata. The analytical definition of the transformation defined by this architecture can be written as

$$y_i = x_i + f_i(x_1, \ldots, x_{i-1}, \vec{\omega}_i), \quad (43)$$

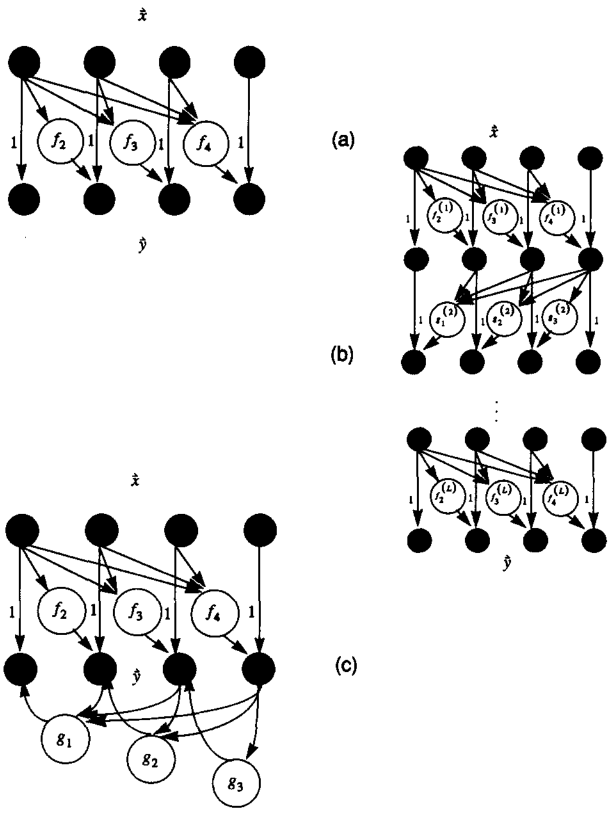

FIGURE 1. Volume-conserving neural architectures. (a) Single-layer volume-conserving network; (b) multilayer network, the Jacobians of the layers being lower and upper triangular matrices, respectively; (c) two-layer architecture. In this last case the second layer can be interpreted as asymmetric lateral connections.

where $\vec{\omega}_i$ represents a set of parameters of the function f_i. Note that independently of the functions f_i the network is always volume conserving and satisfies eqns (6) and (7). In particular, f_i can be calculated by another neural network, by a sigmoid neuron, by polynomials (higher-order neurons), etc. Due to the asymmetric dependence on the input variables and the direct connections with weights equal to 1 between corresponding components of input and output neurons, the Jacobian matrix of the transformation defined in eqn (43) is an upper triangular matrix with diagonal elements all equal to 1, yielding a determinant equal to 1. A network with inverted asymmetry also can be defined as

$$y_i = x_i + g_i(x_{i+1}, \ldots, x_d, \vec{\theta}_i) \quad (44)$$

corresponding to a lower triangular Jacobian matrix with diagonal elements all equal to 1, being therefore also volume conserving. The vectors $\vec{\theta}_i$ represent the parameters of functions g_i. To yield a general nonlinear transformation from inputs to outputs (without asymmetric dependences), it is possible to build a multilayer architecture like the one shown in Figure 1b, which involves mixed versions of networks described by eqns

(43) and (44), respectively. Due to the fact that successive application of volume-conserving transformation is also volume conserving, the multilayer architecture is also volume conserving. In the two-layer case (Figure 1c), the second layer can be interpreted as asymmetric lateral connections between the neurons of the first layer. If f_i is linear an architecture similar to the one presented by Rubner and Tavan for PCA (1989) is obtained. However, in our case the feedforward connections between input layer and first layer are also asymmetric.

As demonstrated in the last section, we minimize a cost function E to decorrelate nonlinearly correlated non-Gaussian inputs. Let us analyze for simplicity a two-layer architecture (Figure 1c) with the first layer given by eqn (43) with h_i in place of y_i and the second layer by eqn (44) with h_i in place of x_i. Let us denote the output of the hidden layer by $\vec{h}$ and use it as input

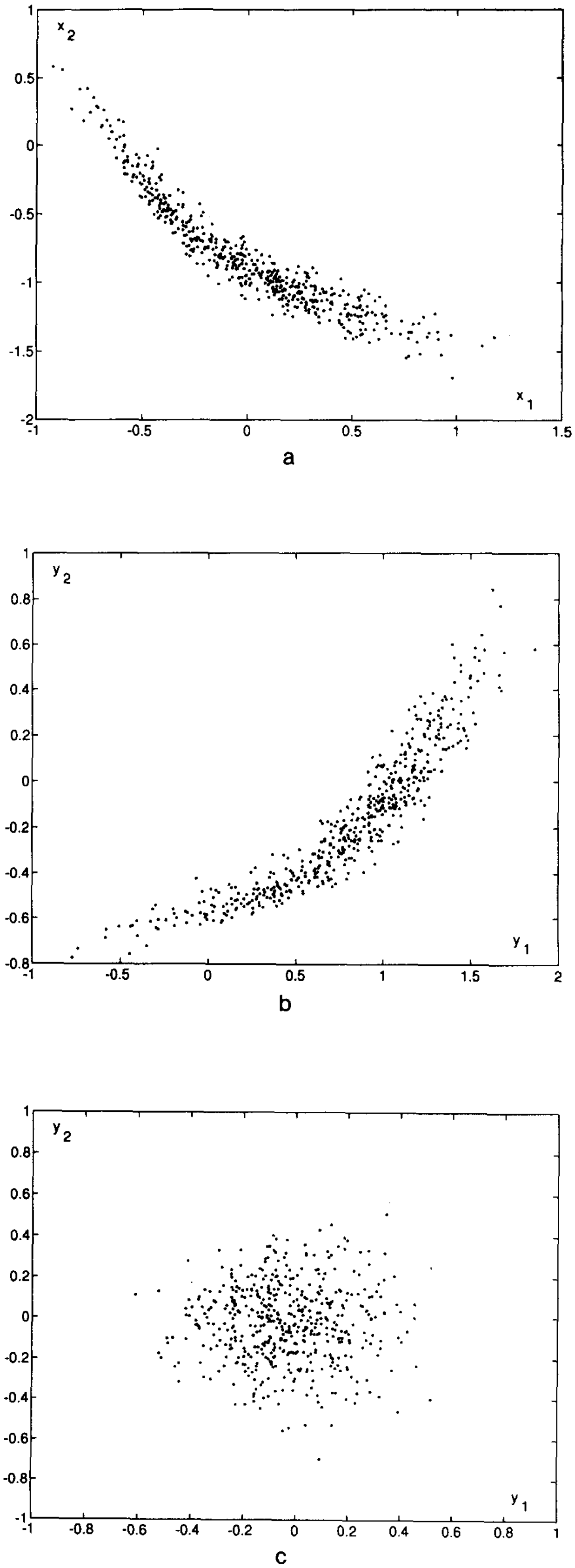

FIGURE 2. Input and output space distribution after training with PCA and with a sigmoidal volume-conserving network. (a) Input space; (b) output space after PCA; (c) output space of the volume-conserving network.

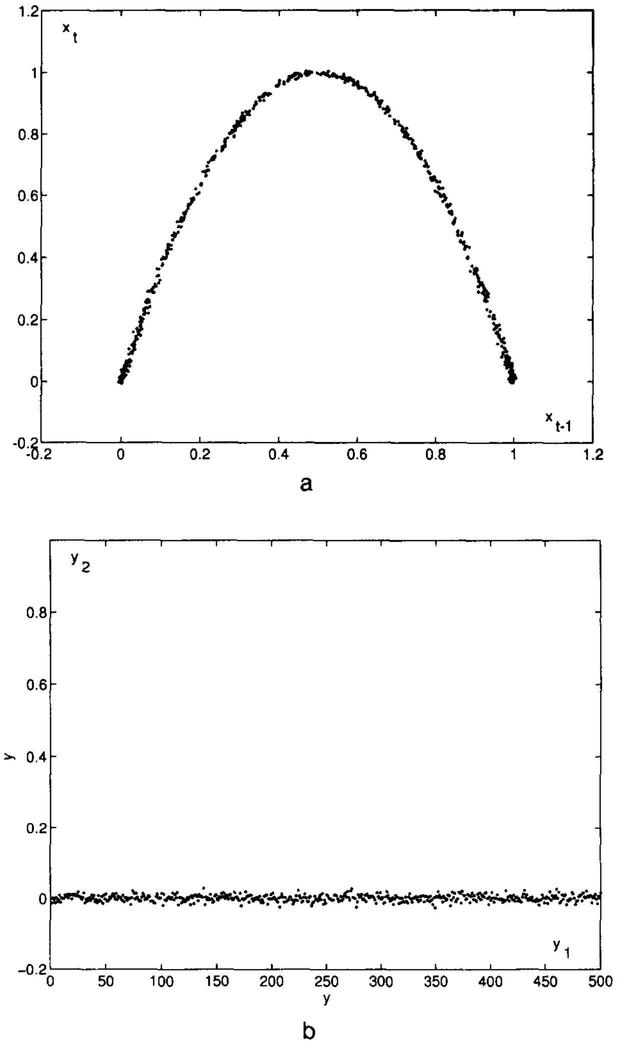

FIGURE 3. Input and output space distribution after training with a one-layer polynomial volume-conserving network of order $R = 2$ for the logistic map. (a) Input space; (b) output space of the volume-conserving network.

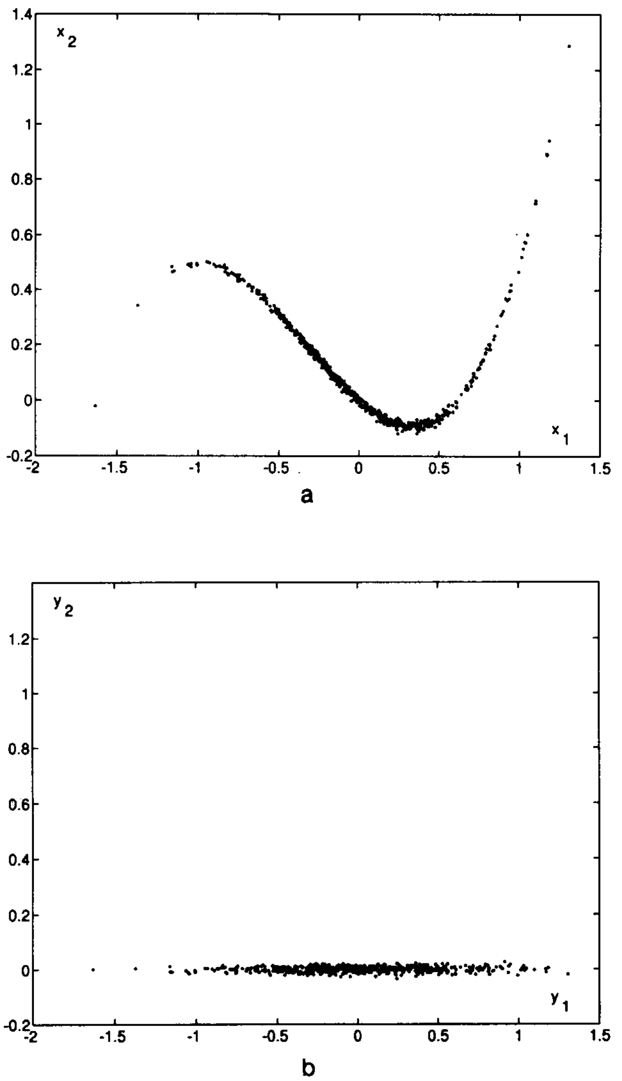

FIGURE 4. Input and output space distribution after training with a one-layer polynomial volume-conserving network of order $R = 3$ for the noisy curve of eqn (67). (a) Input space; (b) output space of the volume-conserving network.

of eqn (44) with output $\vec{y}$. The extension to multilayer architectures is straightforward.

The learning rule can be easily expressed by gradient descent method:

$$\vec{\theta}_i = \vec{\theta}_i - \eta \frac{\partial E}{\partial \vec{\theta}_i} \tag{45}$$

$$\vec{\omega}_i = \vec{\omega}_i - \eta \frac{\partial E}{\partial \vec{\omega}_i}. \tag{46}$$

To calculate the derivative of the cost functions, we need the derivative of eqn (19)

$$\frac{\partial C_{i_1 \ldots i_n}}{\partial \Theta} = \frac{1}{N} \sum_{p=1}^{N} \left\{ \frac{\partial}{\partial \Theta} (y_{i_1} - \bar{y}_{i_1}) \ldots (y_{i_n} - \bar{y}_{i_n}) + \ldots + (y_{i_1} - \bar{y}_{i_1}) \ldots \frac{\partial}{\partial \Theta} (y_{i_n} - \bar{y}_{i_n}) \right\} \tag{47}$$

where Θ represents the parameters $\vec{\theta}_i$ and $\vec{\omega}_i$. In eqn (47) the probability were replaced by the empirical one by taking instead of $\int dy P(y)$ the summation over the N training patterns $(1/N) \Sigma_{p=1}^{N}$. We also need the derivative of the mean values. Using the definition of the empirical mean values

$$\bar{y}_{i_n} = \frac{1}{N} \sum_{p=1}^{N} y_{i_n} \tag{48}$$

the derivative of the mean values is given by

$$\frac{\partial}{\partial \Theta} - \bar{y}_{i_n} = \frac{1}{N} \sum_{p=1}^{N} \left\{ \frac{\partial}{\partial \Theta} (y_{i_n}) \right\}. \tag{49}$$

The gradients of the different outputs are

$$\frac{\partial}{\partial \vec{\theta}_i} y_i = \frac{\partial}{\partial \vec{\theta}_i} g_i(\vec{\theta}_i, h_{i+1}, \ldots, h_d) \tag{50}$$

$$\frac{\partial}{\partial \vec{\omega}_i} y_k = \left(\frac{\partial}{\partial h_i} g_k(\vec{\theta}_k, h_{k+1}, \ldots, h_d) \right) \times \left(\frac{\partial}{\partial \vec{\omega}_i} f_i(\vec{\omega}_i, x_1, \ldots, x_{i-1}) \right) \delta_{i>k} + \left(\frac{\partial}{\partial \vec{\omega}_i} f_i(\vec{\omega}_i, x_1, \ldots, x_{i-1}) \right) \delta_{ik} \tag{51}$$

where $\delta_{i>k}$ is equal to 1 if $i > k$ and 0 otherwise. Equations (45)–(46) and (49)–(51) define the learning rules that we use in this paper.

4. NETWORKS WITH LINEAR, SIGMOIDAL, AND HIGHER-ORDER ACTIVATION FUNCTIONS

In this section we introduce three different architectures of volume-conserving networks corresponding to different choices for the functions f and g of the last section. The first model just replaces f and g by sigmoid neurons. In this case, a two-layer architecture can be defined by

$$h_i = x_i + \tanh\left(\sum_{j=0}^{i-1} \omega_{ij} x_j \right) \tag{52}$$

$$y_i = h_i + \tanh\left(\sum_{j=i+1}^{d} \theta_{ij} h_j \right). \tag{53}$$

For this architecture eqns (50) and (51) become

$$\frac{\partial}{\partial \theta_{ij}} y_k = \delta_{ki} \delta_{j>i} h_j (1 - y_k^2) \tag{54}$$

$$\frac{\partial}{\partial \omega_{ij}} y_k = ((1 - y_k^2) \theta_{ki} \delta_{i>k} + \delta_{ik})(1 - h_i^2) x_j \delta_{j<i}. \tag{55}$$

The second model involves higher-order neurons (Durbin & Rumelhart, 1989). In this case, each function f_i or g_i is a product of polynomial functions of the inputs. The update equations are given by

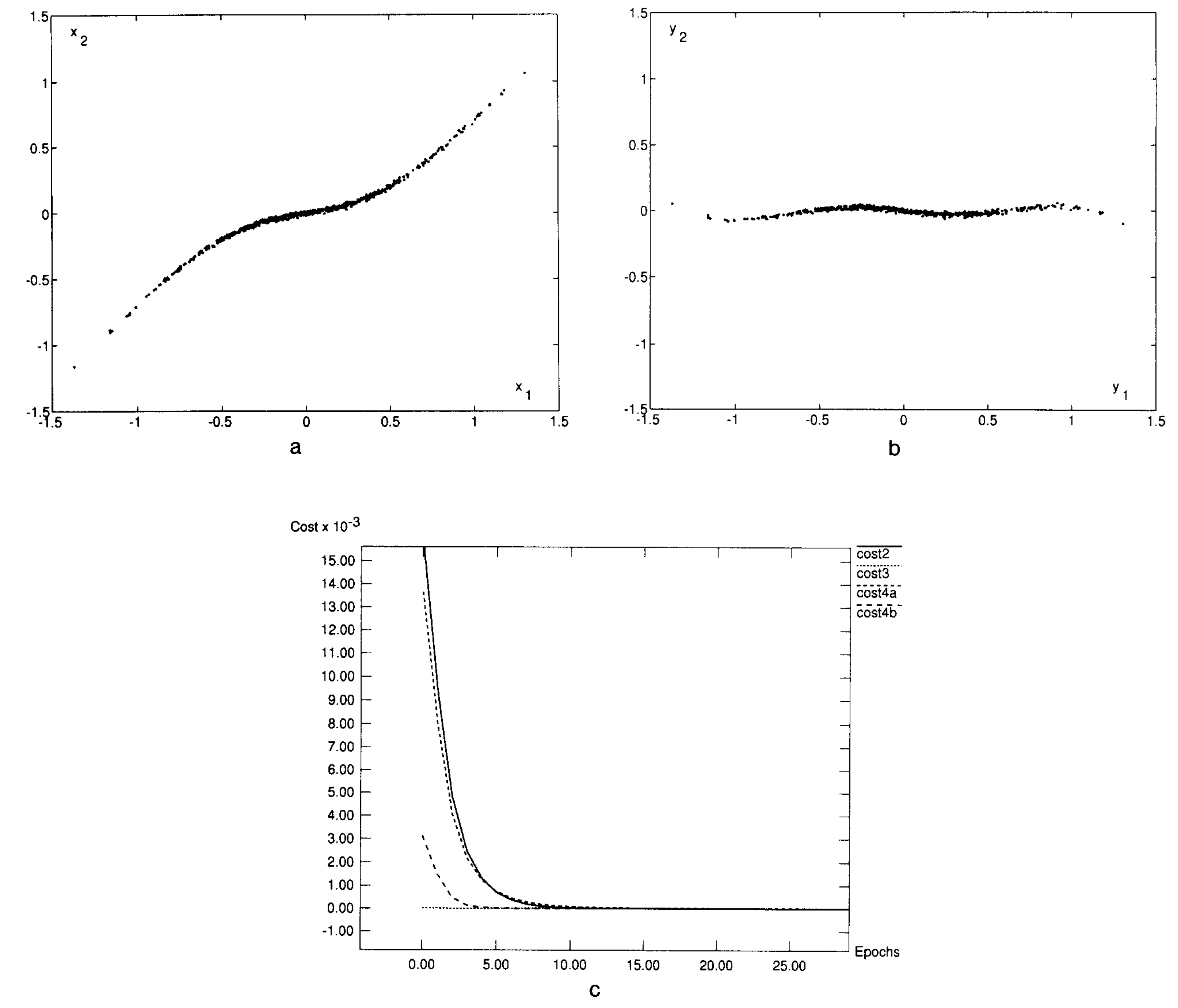

FIGURE 5. Input and output space distribution after training with a two-layer polynomial volume-conserving network of order $R = 4$ for the noisy curve of eqn (68). (a) Input space; (b) output space of the volume-conserving network. (c) Development of the four summands of the cost function E [eqn (37)] during learning: (cost 2) first summand ($E = \alpha \sum_{i<j} C_{ij}^2$); (cost 3) second summand ($\beta \sum_{l<j\leq k} C_{ijk}^2$); (cost 4a) third summand ($\gamma \sum_{(i<j\leq k\leq l)\wedge\neg L} C_{ijkl}^2$); (cost 4b) fourth summand ($\lambda \sum_{i<j} (C_{iijj} - C_{ii}C_{jj})^2$).

$$h_i = x_i + \prod_{j=1}^{i-1}\left(\sum_{r=0}^{R} \omega_{ijr} x_j^r\right) \tag{56}$$

$$y_i = h_i + \prod_{j=i+1}^{d}\left(\sum_{r=0}^{R} \theta_{ijr} h_j^r\right) \tag{57}$$

where R is the order of the polynomial used. In this case, the two-layer architecture is a higher-order network with a general volume-conserving structure. The derivatives involved in the learning rule are given by

$$\frac{\partial}{\partial\theta_{ijr}} y_k = \delta_{ki}\delta_{j>i} h_j^r \left(\frac{y_i}{\sum_{r=0}^{R} \theta_{ijr} h_j^r}\right) \tag{58}$$

$$\frac{\partial}{\partial\omega_{ijr}} y_k = \left(\left(\frac{y_k}{\sum_{r=0}^{R} \theta_{kir} h_i^r}\right)\left(\sum_{r=0}^{R} \theta_{kir} r h_i^{r-1}\right)\delta_{i>k} + \delta_{ik}\right) \times \delta_{j<i} x_j^r \left(\frac{h_i}{\sum_{r=0}^{R} \omega_{ijr} x_j^r}\right) \tag{59}$$

Another kind of higher-order networks (Taylor & Coombes, 1993) is given by the following set of update equations:

$$h_i = x_i + \sum_{j=1}^{i-1} \omega_{ij} x_j + \sum_{j,k=1}^{i-1} \omega_{ijk} x_k x_j + \ldots \tag{60}$$

$$y_i = h_i + \sum_{j=i+1}^{d} \theta_{ij} h_j + \sum_{j,k=i+1}^{d} \theta_{ijk} h_k h_j + \ldots \tag{61}$$

Finally, let us regard linear networks. Here the update equations are

$$h_i = x_i + \left(\sum_{j=1}^{i-1} \omega_{ij} x_j\right) \tag{62}$$

$$y_i = h_i + \left(\sum_{j=i+1}^{d} \theta_{ij} h_j\right), \tag{63}$$

which are a special case of eqns (60) and (61) when the polinomial is of first order. For this architecture eqns (50) and (51) result in

$$\frac{\partial}{\partial \theta_{ij}} y_k = \delta_{ki}\delta_{j>i}h_j \tag{64}$$

$$\frac{\partial}{\partial \omega_{ij}} y_k = (\theta_{ki}\delta_{i>k} + \delta_{ik})x_j\delta_{j<i}. \tag{65}$$

This last network is a generalization of PCA for the linear case, in the sense that non-Gaussian distributions can be linearly decorrelated, taking into account also higher orders of the correlation tensors. In spite of the fact that the architecture can be done identical to the architecture of Rubner and Tavan (1989), the learning rules are different, so in our case the goal is statistical decorrelation and therefore higher-order cumulants are taken into account in the learning rule. On the contrary, Rubner and Tavan (1989) only converges to PCA, which means decorrelation by taking into account only the second order (i.e., diagonalization of the covariance matrix).

5. RESULTS AND SIMULATIONS

We will present herein four different experiments using the three architectures defined in the last section and the learning rule introduced in Sections 2 and 3. The input space in all experiments is two-dimensional in order to show graphically the results and effects of the presented model.

In the first experiment we apply a volume-conserving architecture consisting of sigmoid neurons as defined in eqns (52) and (53). The input space was generated by feeding the inverse of a nonlinear volume-conserving sigmoidal teacher network with two decorrelated Gaussian variables. The result of this inversion (i.e., the input data) is plotted in Figure 2a. Obviously, the input is nonlinearly correlated. Figure 2b shows the output space obtained after applying singular value decomposition (i.e., PCA). Because the correlation is nonlinear and the distribution is non-Gaussian (nonlinear inversion of Gaussian distributions), PCA cannot decorrelate the outputs. PCA only diagonalizes the covariance matrix. In this case, it is not sufficient to diagonalize the covariance matrix to yield independence of the output components. Therefore, Figure 2b still shows a structure. Figure 2c depicts the distribution of the output of the two-layer volume-conserving sigmoidal network after training. It is obvious that in this case the outputs are decorrelated. In addition, the trained network converged to the inverted teacher network. In this case, not only was the covariance diagonalized but also the higher-order correlation tensors satisfied eqns (32)–(35) with accuracy 10^{-7}, which fulfills the condition of statistical independence.

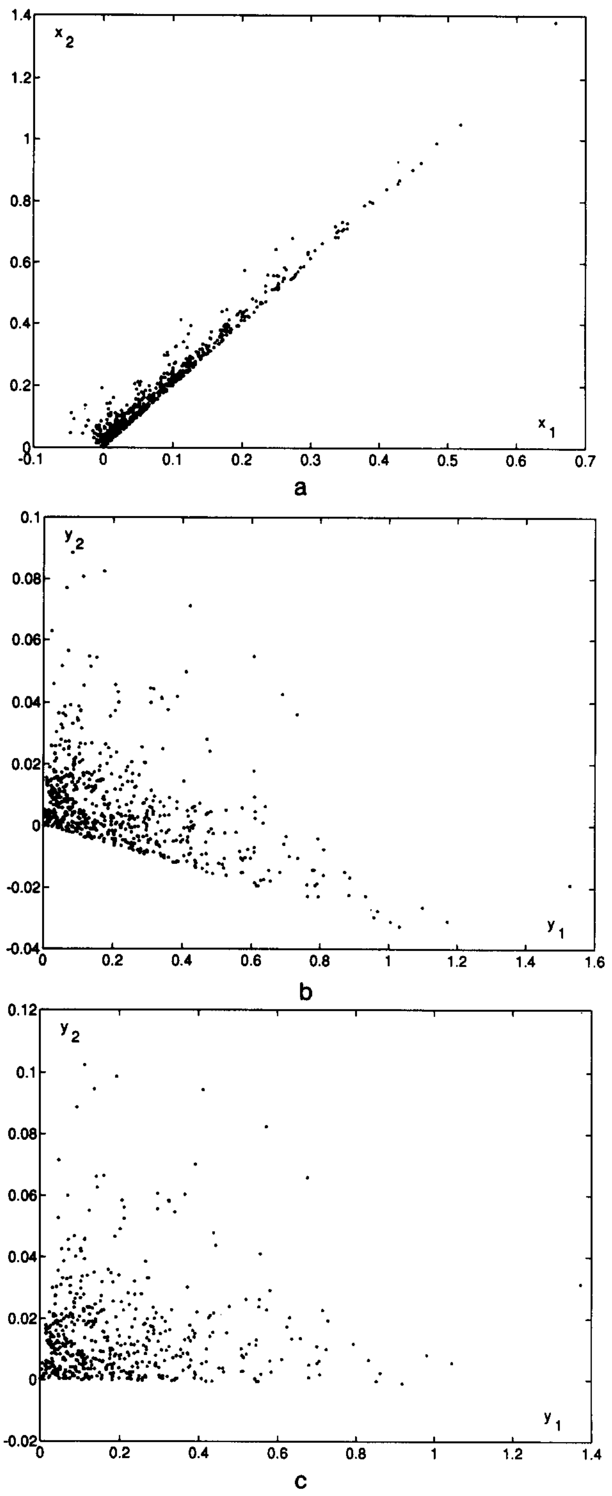

FIGURE 6. Input and output space distribution after training with PCA and with a two-layer linear volume-conserving network. The inputs are exponentially distributed and linearly correlated. (a) Input space; (b) output space after PCA; (c) output space of the volume-conserving network.

The second, third, and fourth experiments apply a more general architecture, namely higher-order volume-conserving neural network corresponding to eqns (56) and (57) described in the last section. The experiments aim at learning noisy nonlinear polynomial and rational curves.

Figure 3a and b plots the input and output space of the second experiment after training is finished, respectively. The first experiment concerns the chaotic time series generated by the logistic map. In this case, the noisy logistic map was used to generate the input by using the following recursive iteration

$$x_t = 4x_{t-1}(1 - x_{t-1}) + v \qquad (66)$$

where v introduces 1% Gaussian noise. The input space is defined by using x_t as x_2 and x_{t-1} as x_1. It is important to remark that the noise is not white but colored, due to its inclusion in the iterative mapping. A one-layer polynomial network with $R = 2$ was used. The learning constant was $\eta = 0.01$ and 20,000 iterations of training were performed. The result is shown in Figure 3b. The volume-conserving network decorrelated the output space extracting the strong nonlinear correlation that generated the curve in the input space. After training, only one coordinate (y_1) is dominant in describing the curve. The whole information was compressed into the first coordinate of the output. This means that the deterministic part that generates the chaotic time series was modeled, even in the presence of colored noise. This is the generalization of data compression normally performed by using linear PCA (also called Karhunen–Loewe transformation). It is important to remark that a traditional fit of polynomials (for example, of second order in this case) takes into account only second-order correlations. In the present example, due to the colored character of the noise, higher-order cumulants are need to factorize perfectly (Figure 3b) the output probability as is obtained with our model that implement decorrelation up to the fourth order. In fact, analyzing the weights of the trained network, the polynomial rhs of eqn (66) is recovered.

Figure 4a and b shows the input and output distribution for the third experiment, using a one-layer volume-conserving polynomial network of order $R = 3$. The input space was generated using the equation

$$x_2 = 0.5(x_1^2 - x_1 + x_1^3) + v \qquad (67)$$

where x_1 is a variable with standard normal distribution and v introduces 1% Gaussian noise. The training was performed during 20,000 iterations and the learning constant was $\eta = 0.01$. Statistical decorrelation leading to data compression is achieved. By analyzing the weights of the network the equation of the curves [eqn (67)] can be extracted.

The fourth experiment is similar, but in this case a two-layer network of order $R = 4$ was used. The input space is given by the rational function

$$x_2 = 0.2x_1 + \frac{x_1^3}{(1 + x_1^2)} + v \qquad (68)$$

where x_1 and v are as in the last case. The results are shown in Figure 5a (input space) and Figure 5b (output space). Figure 5c shows the evolution of the four summands of eqn (37) during learning. It is important to remark that at the beginning the tensors of second and third order are equally important. During learning all summands are simultaneously minimized, resulting in a statistically decorrelated output. The training was performed during 20,000 iterations and the learning constant was $\eta = 0.005$. This case is very interesting because the function that correlates the inputs is a rational function and the network is polynomial. This means that the learning algorithm tries to find the polynomial approximation that extracts the nonlinear relation (in this case a rational function) (i.e., the first orders of the Taylor expansion of the nonlinearity that correlate the inputs). Therefore, the results after training (shown in Figure 5b) are not as "perfect" as in the former cases, in the sense that the correlation was not absolutely extracted (ondulation around the x-axis) until the fourth order in a Taylor expansion of the real nonlinearity that correlates the data was extracted.

The last experiment was performed to show an example of non-Gaussian linearly correlated input. Figure 6a shows these two inputs, both distributed according to an exponential distribution. The result of PCA is shown in Figure 6b. Decorrelation is not total, although the covariance matrix is diagonal. Note the structured distribution. The result of a two-layer volume-conserving linear network trained with our learning rule is plotted in Figure 6c, which shows the decorrelated outputs. The structure in the negative region of y_2 disappeared.

6. CONCLUSIONS

In this paper we proposed a neural architecture and an unsupervised learning paradigm. This approach, which is based on Information Theory, is a means to perform redundancy reduction among the elements of the output layer without losing information, as the data is sent through the network. This is called factorial learning. The model developed performs a generalization of Barlow's unsupervised learning, which consists of nonlinear decorrelation up to higher orders of the cumulant tensors. After training, the components of the output layer are uncorrelated up to a certain desired order (in our experiments the fourth order). Due to the use of higher-order cumulant expansion, non-Gaussian distributions can be rigorously handled up to a desired order. When nonlinear units are used (sigmoid or higher-order pi-neurons), nonlinear principal component analysis is obtained. In this case, nonlinear manifolds can be reduced to a minimum dimension manifolds. When linear units are used, the network performs a generalized principal component analysis in the sense that non-Gaussian distribution can be linearly decorrelated.

The basic idea is to define an architecture that assures the perfect transmission of information (i.e., no information is lost). Thus, the nonlinear transformation defined by the neural architecture is always bijective. The architecture performs a volume-conserving transformation, which means that the determinant of the Jacobian matrix is equal to 1. As a particular case we can derive the reversible cellular automata architecture proposed by Redlich (1993b). In the linear case we obtain an architecture similar to Rubner and Tavan's (1989) for principal component analysis. This paper generalizes previous works on factorial learning in two ways: the architecture performs a general nonlinear transformation without loss of information, and the decorrelation is performed without assuming Gaussian distributions. Simulations performed with networks of sigmoid neurons and with higher-order neurons demonstrate the efficiency of this method in decorrelating nonlinearly correlated inputs and in regularizing curves.

REFERENCES

Abraham, R., & Marsden, J. (1978). *Foundations of mechanics.* London: The Benjamin–Cummings Publishing Company.

Atick, J., & Redlich, A. (1990). Towards a theory of early visual processing. *Neural Computation,* **2,** 308–320.

Atick, J., & Redlich, A. (1992). What does the retina know about natural scenes. *Neural Computation,* **4,** 196–210.

Barlow, H. (1959). Sensory mechanism, the reduction of redundancy, and intelligence. In *National Physical Laboratory Symposium N. 10, The Mechanization of Thought Processes.* London: Her Majesty's Stationery Office.

Barlow, H. (1989). Unsupervised learning. *Neural Computation,* **1,** 295–311.

Barlow, H., & Földiak, P. (1989). Adaptation and decorrelation in the cortex. In R. Durbin, C. Miall, & Mitchison (Eds.), *The computing neuron.* New York: Addison–Wesley.

Barlow, H., Kaushal, T., & Mitchison, G. (1989). Finding minimum entropy codes. *Neural Computation,* **1,** 412–423.

Bodewig, E. (1956). Matrix calculus. Amsterdam: North Holland.

Deco, G., & Parra, L. (1993). Nonlinear features extraction by redundancy reduction with stochastic neural networks. Submitted to *Biological Cybernetics.*

Durbin, R., & Rumelhart, D. (1989). "Product units: A computationally powerful and biologically plausible extension to backpropagation networks. *Neural Computation,* **1,** 133–142.

Földiak, P. (1989). Adaptive network for optimal linear feature extraction. *In Proc. IEEE/INNS Int. Joint Conf. Neural Networks, Washington, DC* (Vol. 1, pp. 401–405). New York: IEEE Press.

Kuehnel, H., & Tavan, P. (1990). The anti-Hebb rule derived from information theory. In R. Eckmiller, G. Hartmann, & G. Hauske (Eds.), *Parallel processing in neural systems and computers* (p. 187–190). North-Holland: Elsevier Science.

Linsker, R. (1988). Self-organization in a perceptual network. *Computer,* **21,** 105.

Linsker, R. (1989). How to generate ordered maps by maximizing the mutual information between input and output signals. *Neural Computation,* **1,** 402–411.

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

Obradovic, D., & Deco, G. (1993). Generalized linear features extraction: An information theory approach. Submitted to *Neural Computation.*

Papoulis, A. (1991). *Probability, random variables, and stochastic processes* (3rd ed.). New York: McGraw–Hill.

Redlich, A. N. (1993a). Redundancy reduction as a strategy for unsupervised learning. *Neural Computation,* **5,** 289–304.

Redlich, A. N. (1993b). *Supervised factorial learning. Neural Computation,* **5,** 750–766.

Reichl, L. E. (1980). *A modern course in statistical physics.* London: Ed. Edward Arnold, Hodder and Stoughton Ltd.

Rubner, J., & Tavan, P. (1989). A self-organization network for principal-component analysis. *Europhysics Letters,* **10,** 693–698.

Taylor, J. G., & Coombes, S. (1993). Learning higher order correlations. *Neural Networks,* **6,** 423–427.

APPENDIX

In this appendix we present the derivation of multidimensional cumulants. Let us first develop the exponential factor in the integrand of eqn (10). We obtain,

$$\phi(\vec{K}) = \int d\vec{y}\, \exp[i(\vec{K}\cdot\vec{y})]P(\vec{y})$$
$$= \left(\sum_{n=0}^{\infty} \frac{i^n}{n!} \sum_{i_1,i_2\ldots,i_n}^{n} C_{i_1,\ldots,i_n} K_{i_1} K_{i_2} \ldots K_{i_n} \right. \tag{A.1}$$

where $C_{i_1\ldots i_n}$ are the higher-order correlation tensors defined in eqn (19). The series expansion is meaningful only if the higher moments are small and thus the series converges. Convenient is the cumulant expansion given by eqn (12). To find the higher-order cumulants $\varkappa_{i_1,\ldots,i_n}$ we perform the Taylor expansion of the rhs of eqn (12). We obtain

$$\phi(\vec{K}) = \sum_{n=0}^{\infty} \left(\frac{i^n}{n!} \sum_{i_1,i_2\ldots,i_n}^{n} \frac{\partial^n}{\partial K_{i_1} K_{i_2} \ldots K_{i_n}} \right.$$
$$\left. \times \left(\exp\left[\sum_{n=1}^{\infty} \frac{i^n}{n!} \sum_{i_1,i_2\ldots,i_n}^{n} \varkappa_{i_1,\ldots,i_n} K_{i_1} K_{i_2} \ldots K_{i_n} \right] K_{i_1} K_{i_2} \ldots K_{i_n} \right) \right. . \tag{A.2}$$

After calculating the derivatives and equating terms of same order K in eqns (A.1) and (A.2), by simple but very tedious algebra, it is possible to obtain the cumulants. The first four cumulants are given by

$$\varkappa_i = \bar{y}_i, \tag{A.3}$$

$$\varkappa_{ij} = C_{ij} - \bar{y}_i\bar{y}_j, \tag{A.4}$$

$$\varkappa_{ijk} = C_{ijk} - C_{ij}\bar{y}_k - C_{jk}\bar{y}_i - C_{ik}\bar{y}_j + 2\bar{y}_i\bar{y}_j\bar{y}_k, \tag{A.5}$$

$$\begin{aligned}\varkappa_{ijkl} = {} & C_{ijkl} - C_{ijk}\bar{y}_l - C_{ijl}\bar{y}_k - C_{ilk}\bar{y}_j - C_{ljk}\bar{y}_i - C_{ij}C_{kl} - C_{ik}C_{lj} \\ & - C_{il}C_{jk} + C_{ij}\bar{y}_k\bar{y}_l + C_{ik}\bar{y}_j\bar{y}_l + C_{il}\bar{y}_j\bar{y}_k + C_{jk}\bar{y}_i\bar{y}_l + C_{jl}\bar{y}_k\bar{y}_i \\ & + C_{kl}\bar{y}_i\bar{y}_j - 6\bar{y}_i\bar{y}_j\bar{y}_k\bar{y}_l.\end{aligned} \tag{A.6}$$

After an extra transformation

$$\vec{y}\,' = \vec{y} - \overline{(\vec{y})} \tag{A.7}$$

to remove the bias $\overline{(\vec{y})}$, we can rewrite the cumulants as

$$\varkappa_i = 0, \tag{A.8}$$

$$\varkappa_{ij} = C_{ij}, \tag{A.9}$$

$$\varkappa_{ijk} = C_{ijk}, \tag{A.10}$$

$$\varkappa_{ijkl} = C_{ijkl} - C_{ij}C_{kl} - C_{ik}C_{lj} - C_{il}C_{jk}. \tag{A.11}$$