Neural Networks, Vol. 6, pp. 627–648, 1993
Printed in the USA. All rights reserved.

0893-6080/93 $6.00 + .00

ORIGINAL CONTRIBUTION

Least Mean Square Error Reconstruction Principle for Self-Organizing Neural-Nets

LEI XU

Peking University and Harvard University

(*Received* 29 *July* 1991; *revised and accepted* 16 *October* 1992)

Abstract—*We proposed a new self-organizing net based on the principle of Least Mean Square Error Reconstruction (LMSER) of an input pattern. With this principle, a local learning rule called LMSER is naturally obtained for training nets consisting of either one or several layers. We proved that for one layer with n_1 linear units, the LMSER rule lets their weights converge to rotations of the data's first n_1 principal components. These converged points are stable and corresponding to the global minimum in the Mean Square Error (MSE) landscape, which has many saddles but no local minimum. The results indirectly provided a picture about LMSER's global convergence, which is also suitable for Oja rule since we proved that the evolution direction of the Oja rule has a positive projection on that of LMSER. We have also revealed an interesting fact that slight modifications of the LMSER rule (also the Oja rule) can perform the true Principal Component Analysis (PCA) without externally designing for building asymmetrical circuits required by previous studies.*

Keywords—Self-organization, Least MSE reconstruction, PCA nets, Convergence analysis, Symmetry breaking.

1. INTRODUCTION

Self-organization has been studied intensively by many researchers for decades. Various models of self-organizing nets have been developed for various purposes. The well known results in literature include Grossberg's theories on competitive learning and adaptive resonance (Grossberg, 1969, 1972, 1976a, 1976b, 1987), von der Malsburg's models for orientative column formation and retinotopic map (von der Malsburg, 1973), Fukushima's cognitron and neocognitron for pattern recognition (Fukushima, 1975, 1980), Kohonen's topographic map for somatosensory map and vector quantization (Kohonen, 1982, 1988), as well as Carpenter and Grossberg's pattern recognition models ART1, 2, 3 (Carpenter & Grossberg, 1987a, 1987b, 1988, 1990). In recent years, many further developments have been made along the paths initiated by these classical results (Ahalt et al., 1990; Barlow, 1989; Desieno, 1988; Rumelhart & Zipser, 1985; Xu et al., 1992). Furthermore, a number of new roads have also been explored. Examples are the self-organizing net proposed by Becker and Hinton (1992) for discovering spatially coherent properties, the studies on combining self-organization into supervised learning (Casdagli, 1989; Hecht-Nielsen, 1987; Jacobs, 1991; Moody & Darken, 1989; Poggio & Girosi, 1990), and Principal Component Analysis (PCA) nets for feature extraction and data compression. Especially, the study on PCA nets, stemmed from Oja's constrained Hebbian rule (Oja, 1982), has grown quite rapidly and recently formed a notable research branch of self-organization with considerable developments (Baldi & Hornik, 1989, 1991; Chauvin, 1989; Foldiak, 1989; Hornik & Kuan, 1991; Kammen & Yuille, 1988; Kung, 1990; Linsker, 1986, 1988; Oja, 1989; Oja, Ogawa, & Wangviwattana, 1991; Rubner & Tavan, 1989; Sanger, 1989; Xu, 1991; Xu, Oja, & Suen, 1992; Yuille, Kammen, & Cohen, 1989).

However, some important issues about PCA nets have not been solved or need further exploration. First, as pointed out by Hornik and Kuan (1991), there are only results of local convergence analysis about the Oja subspace rule (Oja, 1989), as well as other related multiunit PCA learning models (e.g., Rubner & Tavan, 1989), the further global analysis seems to be extremely challenging although extensive experiments have illustrated that these PCA models do converge globally.

I am grateful to Prof. E. Oja, Lappeenranta University of Technology, Finland. During 1989.2–1990.5, I worked at his laboratory as a visiting senior researcher. It is this visit and the discussions with him that arose my interest and gave me many insights on self-organizing net, which laid the base for my present work. I would also like express my thanks to Prof. M. Hasselmo and Dr. T. S. Lee of Harvard University for correcting English and for their comments, as well as to the reviewers for their comments and suggestions which improved the early version of this manuscript considerably.

Requests for reprints should be sent to Lei Xu, G-14 Pierce Hall, Division of Applied Sciences, Harvard University, Cambridge, MA 02138, USA.

Second, the Oja subspace rule (Oja, 1989) and some other PCA models (e.g., Foldiak, 1989) do not perform true PCA, but a "collective version" of PCA, which can be called Principal Subspace Analysis (PSA) (i.e., projecting input patterns onto a subspace spanned by a number of the largest principal eigenvectors). As will be further discussed in Section 3.3, PSA is inferior to PCA in some aspects. Although some existing PCA nets (Rubner & Tavan, 1989; Sanger, 1989) can perform the true PCA, some external "hardwiring" has to be made on the net architecture so that it becomes an asymmetrical circuit. An interesting question arises: "Without any external "hardwiring," is there some way to make a symmetrical model (like the Oja subspace rule) break the symmetry so that true PCA can be performed?" Third, the existing PCA models, as well as most of the other self-organizing models mentioned in the above paragraph, are constructed in a *bottom up* way, i.e., one first has a local learning rule (e.g., Hebbian rule or some modified version) for updating the weights of one unit, then one specifically designs an architecture for multiple units and heuristically figures out a mechanism which can allocate or extend the local learning rule to act on multiple units appropriately. However, we are not clear whether these heuristical learning rules relate to some general principles such as minimization of some energy or error function, and whether one can start from some general principle through a *top down* manner to naturally obtain some local learning rule which can also perform PCA. Fourth, if such a general principle exists, it is interesting to explore further whether one can build a multilayer self-organizing net based on the general principle to get more functions than the existing PCA nets, which, like the Oja subspace rule, are of one layer architecture and can only perform feature extraction or data compression tasks.

This paper aims to explore the above issues. We propose a new self-organizing net which has a general architecture consisting of either one layer or multiple layers. The net is trained by a local learning rule called LMSER which is naturally derived from a general principle—Least Mean Square Error (MSE) Reconstruction of input patterns. The net of one layer architecture not only can perform true PCA, but also provide several important results for the problems arising in the first three issues above. This network is an example of a multilayer self-organizing system which is trained based on a general principle rather than some specifically and heuristically designed bottom up training algorithm. We also speculated that this multilayer net may possess a number of useful potential functions.

In section 2, the general architecture of our multilayer net is proposed in detail. The architecture is very similar to that of the ordinary fully connected (i.e., each unit on one layer is connected to all the units in the next layer) multilayer feedforward net trained by the back propagation technique, except that in our net each connection weight is bidirectional and symmetrical. Each unit will receive the bottom up signals from the lower layer and the top down signals from the upper layer. Each unit also emits a top down signal to its lower layer and a bottom up signal to its upper layer. The net works in two phases: *perception* and *learning*. In the perception phase, when a pattern is presented to the bottom layer (called input field), the net will pass the signals up and the net will experience a stable dynamic process which has been proved to converge into an equilibrium. Then, the top-down signal received by the input field is regarded as the reconstruction of the input pattern. In the learning phase, all the weights are modified based on a general principle that the Mean Square Error (MSE) between input patterns and their reconstructions is minimized.

In the subsequent sections, we will not thoroughly explore the proposed multilayer architecture. Some speculative discussions on a number of potential applications and extensions of this multiple layer architecture were given in a recent conference paper (Xu, 1991), while further experimental and theoretical investigations are left for a separate later study. In Section 3, we concentrate on the further theoretical and experimental studies of the one layer special case. It can be seen that this simple net contains rich contents that can provide several important results.

In Section 3.1, we show that a one layer net of linear units, trained by the LMSER rule, has the same function as Oja subspace rule has. More precisely, we have theoretically proven that for a one layer net consisting of n_1 linear units, the LMSER rule (which minimizes the MSE of reconstruction in the gradient descent way) will let the weight vectors of these units converge to some rotations of the first n_1 principal eigenvectors of the input data's covariance matrix, and that these converged points are stable and correspond to the global minimum in the landscape of MSE of reconstruction. We have also proved that just like the landscape discovered by Baldi and Hornik (1989) for the linear d-p-d architecture[1] of supervised learning nets, the landscape of MSE of reconstruction has no other local minimum points but many saddle points. For the Oja subspace rule, as well as some other PCA learning rules, although the existing theoretical result from some local convergence analysis indicates that the converged points described above are the only stable stationary points, there is no result about the global convergence for these rules. Actually, such a global convergence analysis is regarded to be extremely challenging (Hornik & Kuan, 1992). Here, for the first time, the results about the LMSER rule have connected these converged points

[1] i.e., This is a feedforward network consisting of one hidden layer with q units and one output layer with d units. The dimension of input $\vec{x}$ is also d. The net is trained by back propagation with the same input $\vec{x}$ being taken as the desired output.

with an energy landscape which has no other local minimums. This connection provides a clear picture about global convergence, although indirectly.

In Section 3.2, we show further that either the one-unit Oja rule (Oja, 1982) or the subspace rule (Oja, 1989) is a downhill rule for minimizing the MSE of reconstruction. More precisely, we have proven that on the average its evolution direction has a positive projection on the evolution direction of the LMSER rule. As pointed out in Baldi and Hornik's (1991) recent review paper, the Oja rule cannot do a gradient search of any energy function; but there may be the possibility that the Oja rule may have a positive scalar product with the gradient of some energy function and thus still perform a downhill search. Our result has confirmed this conjecture and built the connection between the Oja rule and the LMSER rule. This connection not only shows that the above results about global convergence of the LMSER rule is also suitable for the Oja rule since it makes a downhill (although not gradient descent) search of the same energy landscape, but also makes it possible for us to further modify the Oja subspace rule in Section 3.3 for performing true PCA.

Furthermore, in Section 3.3 we reveal an interesting and important fact that a slight modification of the LMSER rule, as well as the Oja subspace rule, will enable a symmetrically circuited net to perform the true PCA. Two kinds of modifications are proposed and the results of theoretical analysis, as well as experimental simulations, are provided to show how they perform PCA without external design for building asymmetrical circuits. Since these modifications perform the true PCA, they are as capable as Sanger's GHA and other asymmetrically circuited nets (e.g., Rubner & Schulten, 1990) for some practical tasks such as data compression, feature extraction, and the interpretation of the development of orientation cells in the cortical field (Bienenstock et al., 1982; Hubel & Wiesel, 1962; Sanger, 1989). The more interesting point here is that these tasks can be performed by a network without requiring any externally "hardwired" asymmetry required by Sanger (1989) and Rubner and Schulten (1990).

2. MULTIPLE LAYER SELF-ORGANIZING NETS BASED ON THE LEAST MSE RECONSTRUCTION PRINCIPLE

2.1. The Architecture of the Multiple Layer Self-Organizing Net

Let us consider a multiple layer net as shown in Figure 1(a) in which a unit at one layer is connected to all the units in the next layer. Assume that the connections are bidirectional and their weights are symmetrical, i.e., the architecture used here is the same as the ordinary multilayer forward net except that here each connection weight is bidirectional and symmetrical. If biologically the existence of such symmetrical weights is a question, we can divide each of these connections into two parts with each having the same weight and pointing in opposite directions, as shown in Figure 1(a) by the solid and dashed arrows, respectively. However, for convenience, in the sequel, we will equivalently consider that there is only one symmetrical weight between the two units.

In this architecture, the activity z_{jk} of the j-th unit on the k-th ($k > 0$) layer is activated by two signals. One is a bottom-up signal y_{jk}, the other is a top-down signal u_{jk}. That is,

$$z_{jk} = s(y_{jk} + u_{jk}) \quad \text{or} \quad \tau \frac{dz_{jk}}{dt} = -z_{jk} + s(y_{jk} + u_{jk})$$

$$y_{jk} = \sum_{i=1}^{n_{k-1}} w_{ijk} z_{i(k-1)}, \quad \text{and} \quad u_{jk} = \sum_{r=1}^{n_{k+1}} w_{jr(k+1)} z_{r(k+1)} \qquad (1)$$

where the differential equation for z_{jk} is just the conventional dynamic equation for a neural unit which can be found in many publications, e.g., in Section 3.3 of the book by Hertz, Krogh, and Palmer (1991) or in Kohonen's book (Kohonen, 1988). w_{ijk} is the weight connecting the i-th unit on the $(k-1)$-th layer and the j-th unit on the k-th layer, n_k is the number of units on the k layer, and $s(.)$ is a sigmoid function.

On the top layer, each unit receives only bottom up signals from its immediate lower layer, but no top down signals. It also does not emit any bottom up signal but just top down signals to its immediate lower layer. On the bottom layer, the bottom up signal received by each unit j is $y_{j0} = x_j$, i.e., the j-th component of input vector $\vec{x}$. The top down signal it received is u_{j0} given by eqn (1) with $k = 0$. This layer functions as the input field, each unit j is just a port and does not follow the dynamic given in eqn (1).

Equation (1) can also be written in the matrix form. Let $\vec{z}_k$, $\vec{y}_k$, $\vec{u}_k$ be $n_k \times 1$ vectors, we have

$$\vec{z}_k = S(\vec{y}_k + \vec{u}_k) \quad \text{or} \quad \tau \frac{d\vec{z}_k}{dt} = -\vec{z}_k + S(\vec{y}_k + \vec{u}_k)$$

$$\vec{y}_k = W_k \vec{z}_{(k-1)}, \quad \text{and} \quad \vec{u}_k = W'_{k+1} \vec{z}_{k+1} \qquad (2)$$

where W_k is a $n_k \times n_{k-1}$ matrix, and $S(\vec{\xi}) = [s(\xi_1), \ldots, s(\xi_n)]^t$ for a $n \times 1$ vector $\vec{\xi}$.

The net works in two phases: *perception* and *learning,* which will be described in Sections 2.2 and 2.3, respectively.

2.2. The Perception Phase: A Stable Dynamic Process

When an input pattern $\vec{x}$ comes as the bottom up signal, it propagates up to the first layer. Then the units in the first layer emit both bottom up signals to the second layer and top down signals back to the input field. Next, activated by the bottom up signals from the first layer, the units in the second layer emit both bottom up signals

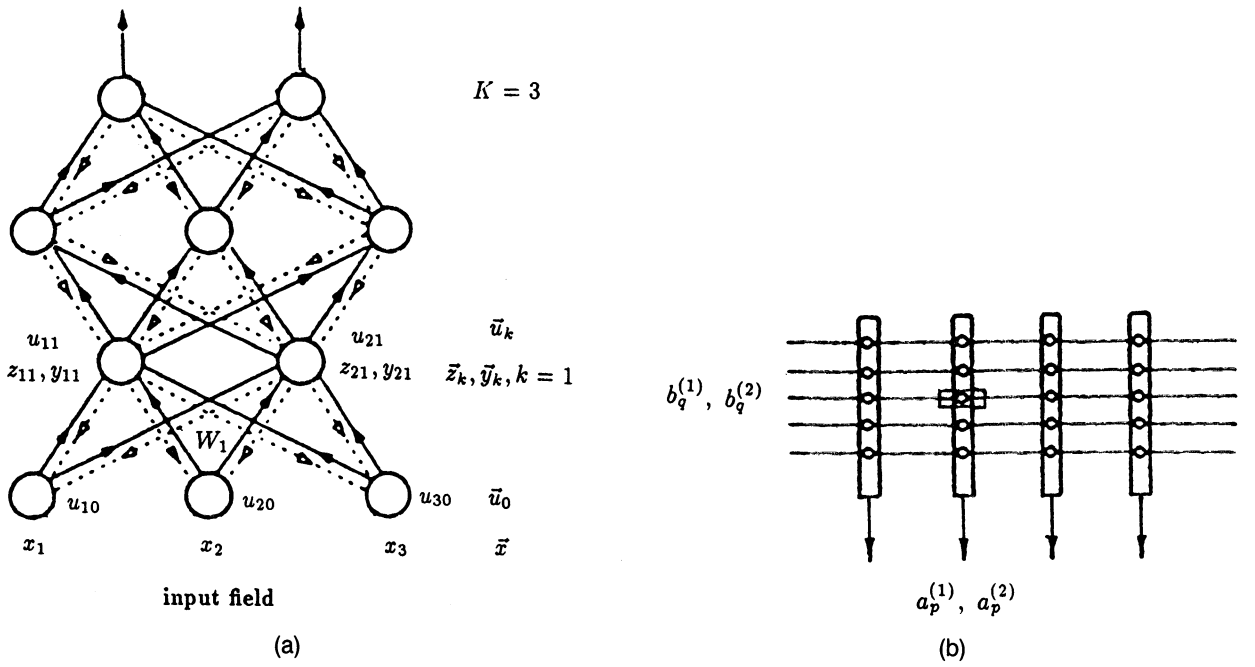


FIGURE 1. A new self-organizing net (a) The architecture of multi-layer net. Here, the connections are bidirectional and their weights are symmetrical. Except for the bottom and the top layers, the activity of each unit is activated by two types of signals: bottom up and top down signals which are indicated by the solid and dashed arrows, respectively. The bottom layer is called input field for the external input patterns. (b) The matrix-type representation of the connections between the two neighbor layers. Where $a_p^{(1)} = z_{p1}$, $b_q^{(1)} = \varepsilon_{q0}$, $a_p^{(2)} = s'_{p1}\varepsilon_{p1}$, $b_q^{(2)} = x_q$ for eqn (8a) (see text); and $a_p^{(1)} = z_{pk}$, $b_q^{(1)} = s'_{q(k-1)}\varepsilon_{q(k-1)}$, $a_p^{(2)} = s'_{pk}\varepsilon_{pk}$, $b_q^{(2)} = z_{q(k-1)}$ for eqn (8b). All of these signals are available locally at the two involved units, which causes only local modifications.

to activate the units on the third layer and the top down signals back to the first layer. These top down signals will make the activities of the units on the first layer change. These changes in turn cause the changes in the top down and bottom up signals emitted by these units. Furthermore, the changes in these bottom up signals will alter the activities of the units on the second layer, which further cause the changes in the activities of the units on both the first and third layers through top down and bottom up signals. Here, we can see that actually the presentation of an input $\vec{x}$ will drive the net into a quite complicated dynamic process. Each unit in any layer (except the input field) remains inactive for just a while in the beginning. Once the unit is activated by the bottom up signals from its immediate lower layer, its activity, as well as its received/emitted top down and bottom up signals, change. These changes are highly coupled with the changes corresponding to the rest of the units in the net. As a result, the top down signals received by the input field also change, namely, the reconstructed input pattern also changes. If these changes are ceaseless, one cannot get a stable reconstructed pattern. This is not what we want. We want these changes to eventually vanish so the activities (and thus all the top down and bottom up signals) can finally converge to some specific values. If so, we say that this dynamic process reached its stable state and the dynamic process is stable.

Fortunately, the dynamic process described in eqns (1) or (2) is stable, which is guaranteed by the following theorem.

THEOREM 1. *When $s(.)$ is a sigmoid-type function, the activities of all the units will finally converge to a set of specific values such that the minimum of the following global Lyapunov or energy function has been reached*

$$J_{\text{net}} = -\frac{1}{2}\sum_{k=2}^{K}\sum_{i=1}^{n_{k-1}}\sum_{j=1}^{n_k} w_{ijk}z_{i(k-1)}z_{jk} + \sum_{k=1}^{K}\sum_{j=1}^{n_k}\int_0^{z_{jk}} s^{-1}(z)\,dz + \sum_{j=1}^{n_1}\sum_{i=0}^{n_0} z_{i1}x_i w_{ij1}. \quad (3)$$

Proof. Let $\vec{z} = [\vec{z}_1^{\,t}, \ldots, \vec{z}_K^{\,t}]^t$ and for simplicity we redenote the components of $\vec{z}$ by z'_p, $p = 1, \ldots, N$ (i.e., $\vec{z} = [z'_1, \ldots, z'_N]^t$) with $N = \sum_{k=1}^{K} n_k$. It is not difficult to see that eqns (1) and (3) are the special case of the following equations:

$$\tau\frac{dz'_p}{dt} = -z'_p + s\left(\sum_q w'_{pq}z'_q + I_p\right)$$

$$J_{\text{net}} = -\frac{1}{2}\sum_{p=1}^{N}\sum_{q\neq p}^{N} w'_{pq}z'_q z'_p + \sum_{p=1}^{N}\int_0^{z'_p} s^{-1}(z)\,dz + \sum_{p=1}^{N} z'_p I_p. \quad (4)$$

That is, eqns (1) and (3) are obtained from eqn (4) by letting those $w_{pq} = w_{ijk}$ if $z'_p = z_{i(k-1)}$, $z'_q = z_{jk}$, $k > 1$ and $w_{pq} = 0$ if z'_p, z'_q are not the two units that one is located on a layer and the other is on the next layer, as well as by letting $I_p = y_{j1} = \sum_i z_{i1} x_i$ if $z'_p = z_{1j}$ and $I_p = 0$ if z'_p is not located on the 1st layer.

Equation (4) is just the global Lyapunov or energy function studied by Cohen and Grossberg (1983) and Hopfield (1984). It is well known that the dynamic process will converge to the minima of J_{net}. Q.E.D.

After the process reached its stable state, the top down signals $\vec{u}_0 = W_0^t \vec{z}_1$ from the first layer are also stabilized. If $\vec{u}_0 - \vec{x} = 0$, then we say that $\vec{u}_0$ is a complete reconstruction of the input pattern $\vec{x}$. In this case, no learning will take place and the activities of the units on the top layers or/and other upper layers can be regarded as the perception of the present input $\vec{x}$. If $\vec{u}_0 - \vec{x} \neq 0$, then we say that $\vec{u}_0$ is a partial reconstruction of $\vec{x}$. One can also regard $\vec{u}_0$ as the recalled pattern by key $\vec{x}$. The activities of the units on the top layers can be regarded as the partial perception of the input $\vec{x}$. Moreover, one can also trigger the following learning phase to reduce the difference $\vec{u}_0 - \vec{x}$.

2.3. The Learning Phase: Based on the Least MSE Reconstruction Principle

The task of learning is to adjust all the weights in the net in order to reduce the reconstruction error. These weights are usually initialized randomly.

Here, we let the learning be guided by the principle of minimizing the MSE, that is,

$$\min_{\{W_k, k=1,\ldots,K\}} J,$$

$$J = \tfrac{1}{2} E(\|\vec{x} - \vec{u}_0\|^2) = \tfrac{1}{2} E(\|\vec{x} - W_1^t \vec{z}_1\|^2). \quad (5a)$$

Using the gradient descent approach, we have the following rule to modify each weight w_{ijk}:

$$\tau^w \frac{\partial w_{ijk}}{\partial t} = -\frac{\partial J}{\partial w_{ijk}}. \quad (5b)$$

In the practical implementation of this gradient descent rule, there are usually two ways to obtain $\partial J/\partial w_{ijk}$. One is to collect a batch of data $\vec{x}(i)$, $i = 1, \ldots, N$ and use $\hat{J} = 1/2N \sum_{i=1}^{N} \|\vec{x}(i) - \vec{u}_0\|^2$ as an estimate of J given by eqn (5a). Then we try to obtain $\partial \hat{J}/\partial w_{ijk}$ for replacing $\partial J/\partial w_{ijk}$ in eqn (5b). This is usually called the *batch way*. The other way is to just use $J^0 = \frac{1}{2}\|\vec{x} - \vec{u}_0\|^2$ to replace the J given in eqn (5b), i.e., to let the following rule replace the rule eqn (5b):

$$\tau^w \frac{\partial w_{ijk}}{\partial t} = -\frac{\partial J^0}{\partial w_{ijk}}. \quad (5c)$$

This is usually called the *adaptive way* or *stochastic approximation*. In the cases that the input $\vec{x}$ comes randomly and stationary from a distribution (i.e., the input sequence about $\vec{x}$ is a stationary process), by taking expectation on both sides of eqn (5c) we can approximately[2] obtain

$$\tau^w \frac{\partial w_{ijk}}{\partial t} = -\frac{\partial \frac{1}{2} E\|\vec{x} - \vec{u}_0\|^2}{\partial w_{ijk}} = -\frac{\partial J}{\partial w_{ijk}}.$$

Therefore, on the average, the rule in eqn (5c) minimizes the J given in eqn (5a) in the gradient descent way.

The batch way needs a period of time and some extra storages for collecting data samples and thus this rule is not suitable for *on line* processing. The adaptive way or stochastic approximation solves such a problem. For a nonreal-time application, an optimization problem like that given in eqn (5a) can be implemented by either batch way or stochastic approximation. Many existing self-organizing algorithms, such as the Oja rule (Oja, 1982, 1989) and the Kohonen learning rule (Kohonen, 1988), use the stochastic approximation approach. In this paper we adopt the stochastic approximation approach for implementing the minimization of the MSE of the reconstruction.

We start from eqn (5c) to derive our local learning rule. First we rewrite J^0 into

$$J^0 = \frac{1}{2}\|\vec{x} - W_1^t \vec{z}_1\|^2 = \frac{1}{2}\sum_{i=1}^{n_0} x_i^2 - \sum_{i=1}^{n_0}\sum_{j=1}^{n_1} w_{ji1} z_{j1} x_i + \frac{1}{2}\sum_{i=1}^{n_0}\left(\sum_{j=1}^{n_1} w_{ij1} z_{j1}\right)^2. \quad (5d)$$

Noticing that $z_{jk} = s(y_{jk} + u_{jk})$ and u_{jk} is irrelevant to any weights below the k-th layer, we try to decouple the problem of eqn (5d) layer by layer from the bottom to the top of the net.

First, for the layer $k = 1$, we have

$$\varepsilon_{i1} = -\frac{\partial J^0}{\partial z_{i1}} = \sum_{j=1}^{n_0} w_{ij1} x_j - \sum_{j=1}^{n_0} w_{ij1}\left(\sum_{r=1}^{n_1} w_{rj1} z_{r1}\right) = y_{i1} - y'_{i1},$$

$$\varepsilon_{j0} = -\frac{\partial J^0}{\partial u_{j0}} = x_j - u_{j0} \quad (6a)$$

$$\frac{\partial z_{i1}}{\partial w_{pq1}} = \frac{\partial s(y_{i1} + u_{i1})}{\partial w_{pq1}} = s'_{i1} \frac{\partial}{\partial w_{pq1}} \sum_{j=1}^{n_0} w_{ij1} x_j = s'_{i1} \delta_{ip} x_q,$$

$$\frac{\partial u_{j0}}{\partial w_{pq1}} = \frac{\partial}{\partial w_{pq1}} \sum_{r=1}^{n_1} w_{rj1} z_{r1} = z_{p1} \delta_{jq} \quad (6b)$$

where $y'_{ik} = \sum_{r=1}^{n_{k-1}} w_{irk} u_{r(k-1)}$ or in the matrix form $\vec{y}'_k = W_k \vec{u}_{k-1}$. $\vec{y}'_k$ can be regarded as the reflected signals of the top down signals $\vec{u}_{(k-1)}$. Furthermore, we denote $s'_{ik} = s'(y_{ik} + u_{ik})$, and let $\delta_{ip} = 1$ if $i = q$; $\delta_{ip} = 0$ if $i \neq q$.

[2] Where we assume that w_{ijk} changes slowly relative to $\vec{x}$.

On the first layer it follows from eqns (6a) and (6b) that for $p = 1, \ldots, n_1$, $q = 1, \ldots, n_1$, we have

$$\tau^w \frac{\partial w_{pq1}}{\partial t} = -\frac{\partial J}{\partial w_{pq1}} = -\sum_{i=1}^{n_1} \frac{\partial J^0}{\partial z_{i1}} \frac{\partial z_{i1}}{\partial w_{pq1}} - \sum_{j=1}^{n_0} \frac{\partial J^0}{\partial u_{j0}} \frac{\partial u_{j0}}{\partial w_{pq1}} = s'_{p1}\varepsilon_{p1}x_q + \varepsilon_{q0}z_{p1}. \quad (6c)$$

Second, for the layers $k = 2, \ldots, K$, we have

$$\frac{\partial z_{j(k-1)}}{\partial z_{ik}} = \frac{\partial s(y_{j(k-1)} + u_{j(k-1)})}{\partial z_{ik}} = s'_{j(k-1)} \frac{\partial u_{j(k-1)}}{\partial z_{ik}}$$

$$= s'_{j(k-1)} \frac{\partial \sum_{r=1}^{n_k} w_{rjk} z_{rk}}{\partial z_{ik}} = s'_{j(k-1)} w_{ijk}$$

$$\varepsilon_{ik} = -\frac{\partial J^0}{\partial z_{ik}} = -\sum_{j=1}^{n_{(k-1)}} \frac{\partial J^0}{\partial z_{j(k-1)}} \frac{\partial z_{j(k-1)}}{\partial z_{ik}}$$

$$= \sum_{j=1}^{n_{(k-1)}} \varepsilon_{j(k-1)} \frac{\partial z_{j(k-1)}}{\partial z_{ik}} = \sum_{j=1}^{n_{(k-1)}} \varepsilon_{j(k-1)} w_{ijk} \quad (7a)$$

where ε_{ik}, $i = 1, \ldots, n_k$ can be obtained recursively from the bottom layer up to the top layer, with the initials given by eqn (6a).

We further calculate

$$\frac{\partial z_{ik}}{\partial w_{pqk}} = s'_{ik} \frac{\partial y_{ik}}{\partial w_{pqk}} = s'_{ik} \frac{\partial}{\partial w_{pqk}} \sum_{j=1}^{n_{k-1}} w_{ijk} z_{j(k-1)}$$

$$= s'_{ik}\delta_{ip} z_{q(k-1)} + s'_{ik} \sum_{j=1}^{n_{k-1}} w_{ijk} s'_{j(k-1)} \frac{\partial u_{j(k-1)}}{\partial w_{pqk}}$$

$$\frac{\partial z_{i(k-1)}}{\partial w_{pqk}} = s'_{i(k-1)} \frac{\partial y_{i(k-1)}}{\partial w_{pqk}} = s'_{i(k-1)} \frac{\partial}{\partial w_{pqk}} \sum_{j=1}^{n_k} w_{jik} z_{jk}$$

$$= s'_{i(k-1)}\delta_{iq} z_{pk} + s'_{i(k-1)} \sum_{j=1}^{n_k} w_{jik} s'_{jk} \frac{\partial y_{jk}}{\partial w_{pqk}}.$$

These are the highly coupled equation groups, the solutions of which need to be solved nonlocally. However, we have $s'(.) = s(1 - s) \leq \gamma = \frac{1}{4}$ for sigmoid function $s(.)$. The second terms in the both equations are of order γ^2, while the first terms are of order γ. Thus we can use the following approximations to replace the above coupled equations:

$$\frac{\partial z_{ik}}{\partial w_{pqk}} \approx s'_{ik}\delta_{ip} z_{q(k-1)}, \quad \frac{\partial z_{i(k-1)}}{\partial w_{pqk}} \approx s'_{i(k-1)}\delta_{iq} z_{pk}. \quad (7b)$$

Consequently, on the k-th layer it follows from eqns (7a) and (7b) that for $p = 1, \ldots, n_1$, $q = 1, \ldots, n_1$, we have the following learning rule:

$$\tau^w \frac{\partial w_{pqk}}{\partial t} = -\frac{\partial J}{\partial w_{pqk}} = \sum_{i=1}^{n_k} \varepsilon_{ik} \frac{\partial z_{ik}}{\partial w_{pqk}} + \sum_{i=1}^{n_{(k-1)}} \varepsilon_{i(k-1)} \frac{\partial z_{i(k-1)}}{\partial w_{pqk}}$$

$$\approx s'_{pk}\varepsilon_{ik} z_{q(k-1)} + s'_{q(k-1)}\varepsilon_{q(k-1)} z_{pk}. \quad (7c)$$

As a summary, we rewrite the main learning equations eqns (6a)(6c), (7a)(7c) together:

$$\varepsilon_{i0} = x_i - u_{i0}, \quad \varepsilon_{i1} = y_{i1} - y^r_{i1},$$

$$\tau^w \frac{\partial w_{pq1}}{\partial t} = \varepsilon_{q0} z_{p1} + s'_{p1}\varepsilon_{p1} x_q, \quad \text{for} \quad k = 1 \quad (8a)$$

$$\varepsilon_{ik} = \sum_{j=1}^{n_{(k-1)}} \varepsilon_{j(k-1)} w_{ijk}, \quad \tau^w \frac{\partial w_{pqk}}{\partial t} \approx s'_{q(k-1)}\varepsilon_{q(k-1)} z_{pk} + s'_{pk}\varepsilon_{ik} z_{q(k-1)}, \quad \text{for} \quad k \geq 2. \quad (8b)$$

To gain more insights on the obtained learning equations eqn (8a) and (8b), we unfold the connections between the k-th and $k + 1$-th layers into matrix-type as shown in Fig. 1(b), where for the first layer we have $a_p^{(1)} = z_{p1}$, $b_q^{(1)} = \varepsilon_{q0}$, $a_p^{(2)} = s'_{p1}\varepsilon_{p1}$, $b_q^{(2)} = x_q$, and for the k-th layer we have $a_p^{(1)} = z_{pk}$, $b_q^{(1)} = s'_{q(k-1)}\varepsilon_{q(k-1)}$, $a_p^{(2)} = s'_{pk}\varepsilon_{pk}$, $b_q^{(2)} = z_{q(k-1)}$. All of these signals are available locally at the two involved units, thus *all the computations are local.*[3] Furthermore, in the first term of eqn (8a), we can see that $z_{p1}x_q$ is just the classical Hebbian learning term. Because the rule in eqn (8) is obtained from the LMSER principle, for convenience, we call it the LMSER learning rule. Moreover, since the learning occurs bottom up through forwardly propagating ε_{ik} (which can also be regarded as errors), we can also call this bottom up implementation of the LMSER rule as a *Forward Error Propagation* method.

It should be noticed that the *perception* and the *learning* phases take place at the same time but at different time scales. As the changes of weights disturb the old dynamic equilibrium, the new equilibrium will be established quickly. Moreover, the smaller the difference $\bar{u}_0 - \bar{x}$ and the shorter the time that $\bar{x}$ is exposed on the input field, the less is learned about $\bar{x}$. In contrast, the larger the difference $\bar{u}_0 - \bar{x}$ and the longer the time that $\bar{x}$ is exposed on the input field, the more is learned about $\bar{x}$. This feature is somewhat similar to that of human learning.

3. FURTHER THEORETICAL AND EXPERIMENTAL STUDIES ON ONE LAYER SPECIAL CASE

3.1. Mathematical Analysis on Linear Units: PSA Emerges Automatically

When $K = 1$, i.e., the number of layers is reduced to 1, we get a special case as shown in Figure 2. In this case, we can simplify the notations as $\vec{z} = \vec{z}_1$, $\vec{y} = \vec{y}_1$, $\vec{u} = \vec{u}_0$, $W = W_1$, and rewrite eqns (2) and (5a) into

$$\vec{z} = S(\vec{y}), \quad \vec{y} = W\vec{x}, \quad \vec{u} = W^t\vec{z}$$
$$J = \tfrac{1}{2}E(\|\vec{x} - \vec{u}\|^2) \quad (9a)$$

[3] When calculating ε_{p1} in eqn (8a), one needs y^r_{q1} which, as mentioned earlier, is the reflection of the top down signal $\vec{u}_0$. y^r_{q1} is still a local signal although it costs a little more computation. Moreover, in eqn (8a), $s'_{p1}\varepsilon_{p1}x_q$ is of one order higher than $z_{p1}\varepsilon_{q0}$ with respect to γ, in the way similar to eqn (7b) we can also just omit the term $s'_{p1}\varepsilon_{p1}x_q$ for saving some computations. However, in the multilayer net this saving is not so critical. Here we would rather keep $s'_{p1}\varepsilon_{p1}x_q$ in eqn (8a).

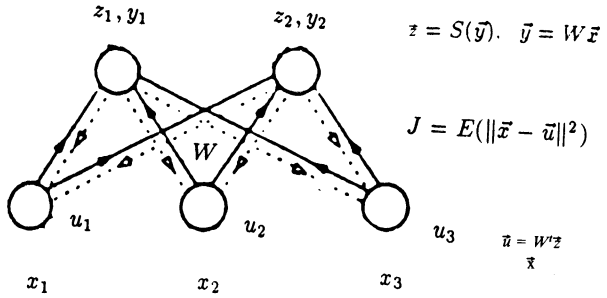


FIGURE 2. The architecture of one layer with n_1 units, each of which is connected to all the variables of the $n_0 \times 1$ input vector $\vec{x}$. Here, $\vec{z} = [z_1, \ldots, z_{n_1}]$ are the activity of n_1 units, W is the connection matrix. $\vec{y}$ is the bottom up signals from the input field and $\vec{u}$ the top down signal from the neural layer to the input layer. $\vec{u}$ is regarded as the reconstruction of $\vec{x}$ produced by the neural layer, and J is the MSE of this reconstruction.

where $S(\vec{\xi}) = [s(\xi_1), \ldots, s(\xi_n)]^t$ for a $n \times 1$ vector $\vec{\xi}$; $\vec{y}$ is the bottom up signals from the input field and $\vec{u}$ the top down signal from the neural layer to the input layer. $\vec{u}$ is regarded as the reconstruction of the input $\vec{x}$ by the neural layer. Thus, J is the MSE of this reconstruction.

Moreover, eqn (8b) has gone and eqn (8a) becomes

$$\vec{\varepsilon}_0 = \vec{x} - \vec{u}, \quad \vec{\varepsilon}_1 = \vec{y} - \vec{y}^r, \quad \vec{y}^r = W\vec{u}$$

$$\tau^w \frac{dW}{dt} = \vec{z}\vec{\varepsilon}_0^t + S'\vec{\varepsilon}_1\vec{x}^t$$

$$S' = \text{diag}[s'(y_1), \ldots, s'(y_{n_1})]. \tag{9b}$$

This is a stochastic approximation rule for minimizing J by gradient descent. Equation (9b) can also be obtained by directly solving dJ/dW through certain steps of derivations.

In this subsection, we study the case that $s(x) = x$. In this case, we have $\vec{z} = \vec{y}$, $S' = I$. It follows from eqns (9a) and (9b) that

$$\tau^w \frac{dW}{dt} = \vec{y}\vec{x}^t - \vec{y}\vec{u}^t + \vec{y}\vec{x}^t - \vec{y}^r\vec{x}^t = W\vec{x}\vec{x}^t - W\vec{x}\vec{x}^tW^tW + W\vec{x}\vec{x}^t - WW^tW\vec{x}\vec{x}^t \tag{10a}$$

Assume that $\vec{x}$ comes from a stationary process with $E(\vec{x}) = \vec{0}$, and let Σ denote the covariance matrix $E(\vec{x}\vec{x}^t)$. Let us further assume that W changes slowly relative to $\vec{x}$. Following the average analysis used by Oja (1982), we take expectation on both sides of eqn (10a), resulting in

$$\tau^w \frac{dW}{dt} = W\Sigma - W\Sigma W^tW + W\Sigma - WW^tW\Sigma. \tag{10b}$$

We will now prove that the learning rule of eqn (10a) performs automatically PSA. First, we will prove that all the possible converged points of the weight matrix W (i.e., all the critical points of eqn (10b)) are, up to an arbitrary rotation, the matrices consisting of n_1 eigenvectors of Σ as their row vectors (Theorem 2). Second, we will prove that all these possible converged points of W are saddle points of the energy landscape of J, except the only stable one which is, up to an arbitrary rotation, a matrix consisting of the first n_1 eigenvectors of Σ as its row vectors, and this stable point let J reach its global minimum value (Theorem 3).

THEOREM 2. *Assume $n_1 < n_0$ and Σ is nonsingular. Let $\Phi = [\vec{\phi}_1, \ldots, \vec{\phi}_{n_0}]$ and $\Lambda = \text{diag}[\lambda_1, \ldots, \lambda_{n_0}]$ be the matrices of eigenvectors and eigenvalues of Σ respectively, then the critical points of eqn (10b) are $W = RDP\Phi^t$, R is an arbitrary $n_1 \times n_1$ rotation matrix, i.e., $R^tR = I$. $D = [D_1 | 0]$ is $n_1 \times n_0$ matrix with D_1 being an $n_1 \times n_1$ diagonal matrix and its diagonal elements only taking value of $+1, -1, 0$. $P_{n_0 \times n_0}$ is an arbitrary permutation matrix.*

Proof. Let $dW/dt = 0$, we know that the critical points of eqn (10b) satisfy

$$W\Sigma - W\Sigma W^tW + W\Sigma - WW^tW\Sigma = 0. \tag{10c}$$

By singular value decomposition, we can write $W = R_{n_1 \times n_1} D_{n_1 \times n_0} \Psi^t_{n_0 \times n_0}$ such that $R^tR = I$, $\Psi^t\Psi = I$, $D = [D_1 | \vec{0}]$, D_1 is an $n_1 \times n_1$ diagonal matrix and some of its diagonal elements may be zero. We can also decompose $W\Sigma^{-1} = R'D'\Psi'$ with $R'^tR' = I$, $\Psi'^t\Psi' = I$ and $D' = [D'_1 | \vec{0}]$.

Since $\text{Rank}[W] = \text{Rank}[W\Sigma^{-1}]$, $\text{Rank}[D'] = \text{Rank}[D'_1] = \text{Rank}[D] = \text{Rank}[D_1]$, thus

$$\begin{aligned} W &= R'D'\Psi'^t\Sigma \\ &= R'D'\Psi'^t\Phi\Lambda\Phi^t = RD\Psi^t = W \Leftrightarrow R^tR'D'\Psi'^t\Phi\Lambda\Phi^t\Psi \\ &= R^tR'D'\Psi'^t\Phi P\Lambda P\Phi^t\Psi = D \Leftrightarrow R^tR' = I, \\ &\quad \Psi'^t\Phi P = I, \quad P\Phi^t\Psi = I, \quad D'\Lambda = D \Leftrightarrow R = R', \\ &\quad \Psi' = \Psi = \Phi P, \quad D'\Lambda = D \end{aligned}$$

where $P_{n_0 \times n_0}$ is an arbitrary permutation of the identity matrix $I_{n_0 \times n_0}$.

Therefore, we have $W = R_{n_1 \times n_1} D_{n_1 \times n_0} P\Phi^t$, put it into eqn (10c) and notice that $R^tR = I$, $\Phi^t\Phi = I$, $PP^t = I$ and $P^tP = I$, we have

$$DP\Lambda - DP\Lambda P^tD^tDP + DP\Lambda - DD^tDP\Lambda = 0$$

$$DP\Lambda P^t - DP\Lambda P^tD^tD + DP\Lambda P^t - DD^tDP\Lambda P^t = 0.$$

Then noticing that $P\Lambda P^t = \Lambda'$ is a diagonal matrix (its diagonal elements are permutation of the diagonal elements of Λ), we have

$$D_1\Lambda'_1 - D_1\Lambda'_1D_1^tD_1 + D_1\Lambda'_1 - D_1D_1^tD_1\Lambda'_1 = 0 \tag{10d}$$

where $\Lambda' = \text{diag}[\Lambda'_1, \Lambda'_2]$ with Λ'_1 being a $n_1 \times n_1$ matrix. It follows from eqn (10d) that $D_1\Lambda'_1 = D_1^3 \Leftrightarrow D_1 = D_1^3 \Leftrightarrow$ the diagonal matrix only has the diagonal elements with values $+1, -10$. Q.E.D.

THEOREM 3. *Assume $\lambda_1 \geq \lambda_2 \ldots \lambda_{n_1} > \lambda_{n_1+1} \geq \lambda_{n_0} > 0$ and $\Phi = [\vec{\phi}_1, \ldots, \vec{\phi}_{n_0}]$. Then all the critical points of eqn (10b) given by Theorem 2 are saddle points of the energy landscape of J, except those with $W = R'\Phi'^t$,*

$R''R' = I$, $\Phi''\Phi' = I$, where $\Phi' = D\Phi'$, and $D = [D_1 | \vec{0}]$ with D_1 being diagonal matrix and its diagonal elements being either $+1$ or -1. Furthermore, these $W = R'\Phi''$ let $J = \frac{1}{2}E(\|\vec{x} - \vec{u}\|^2) = \frac{1}{2}E(\|\vec{x} - W^tW\vec{x}\|^2)$ reach its only local (also global) minimum $J_{min} = \frac{1}{2}\sum_{i=n_1+1}^{n_0} \lambda_i$.

Proof. To notice $\mathrm{tr}(\vec{x}\vec{x}^t) = \mathrm{tr}(\vec{x}^t\vec{x}) = \vec{x}^t\vec{x}$ and $\mathrm{tr}(AB) = \mathrm{tr}(BA)$, we have

$$2J = \mathrm{tr}(E(\vec{x} - W^tW\vec{x})(\vec{x} - W^tW\vec{x})^t)$$
$$= \mathrm{tr}(\Sigma) - 2\,\mathrm{tr}(W\Sigma W^t) + \mathrm{tr}(W^tW\Sigma W^tW).$$

From Theorem 2 that any critical point is given by $W = R\Phi'^t$, $\Phi'^t = DP\Phi^t$, we have

$$2J = \mathrm{tr}(\Sigma) - 2\,\mathrm{tr}(\Phi'^t\Sigma\Phi') + \mathrm{tr}(\Phi'\Phi'^t\Sigma\Phi'\Phi). \quad (11)$$

First, we consider all of the critical points with $\mathrm{Rank}[D] = r < n_1$. In this case, only r rows of Φ'^t are eigenvectors of Σ and the other rows are zero vectors. Assume the j-th row is a zero vector, we slightly perturb the row by $\epsilon\vec{\phi}_k$ such that $\vec{\phi}_k$ is not among the r eigenvectors. Then from eqn (11), we can have $2\Delta J = -\epsilon^2\lambda_k + O(\epsilon^4)$, i.e., J decreases. Thus, these critical points are saddle points.

Second, we see the case that $\mathrm{Rank}[D] = n_1$, which means that the diagonal elements of D_1 are either $+1$ or -1 and thus the n_1 rows of Φ'^t are eigenvectors of Σ. Due to the permutation caused by $P \neq I$, there are two possible situations. One situation is that not each of the n_1 rows is among the first n_1 eigenvectors of Σ. Assume the j-th row is an eigenvector $\vec{\phi}_i$, $i > n_1$, we slightly perturb the row through replacing $\vec{\phi}_i$ by $(\vec{\phi}_i + \epsilon\vec{\phi}_k)/\sqrt{1+\epsilon^2}$ with $\vec{\phi}_k$ being among the first n_1 eigenvectors; Then from eqn (11), we can have $2\Delta J = -\epsilon^2(\lambda_k - \lambda_i)/(1+\epsilon^2)$, again J decreases. Thus, these critical points are also saddle points. The other situation is that the rows of Φ'^t are the first n_1 eigenvectors of Σ. Now, if one of the rows is perturbed through replacing $\vec{\phi}_i$ by $(\vec{\phi}_i + \epsilon\vec{\phi}_k)/\sqrt{1+\epsilon^2}$ with $k > n_1$, J will increase. That is, only those Φ' consisting of the first n_1 eigenvectors of Σ are local minimums. This means that $\Phi'^t = P'D\Phi^t$, or $W = RP'D\Phi^t$, where P' is a $n_1 \times n_1$ permutation matrix. Furthermore, let $R' = RP'$, we also have $R''R' = I$.

In addition, in this case, eqn (11) becomes

$$2J = \sum_{i=1}^{n_0} \lambda_i - \sum_{i=1}^{n_1} \lambda_i = \sum_{i=n_1+1}^{n_0} \lambda_i. \quad \text{Q.E.D.}$$

Theorem 2 and Theorem 3 have not only shown that the stable converging point of W is the rotation of the first n_1 principal component, but also that the landscape of J has only one unique minimum which has a flat bottom (the different rotations corresponds to the horizontally moving within the flat area). As shown in Figure 3, going up from this flat bottom there are in total $\sum_{i=1}^{n_1} C_{n_0}^i - 1$ plateaux which form all the other critical points of eqn (10b) (the first $C_{n_0}^{n_1} - 1$ plateaux due to all the other n_1 nonsingular combinations of eigenvectors, the next $C_{n_0}^{n_1-1}$ plateaux due to all the $n_1 - 1$ combinations, . . . , and so forth).

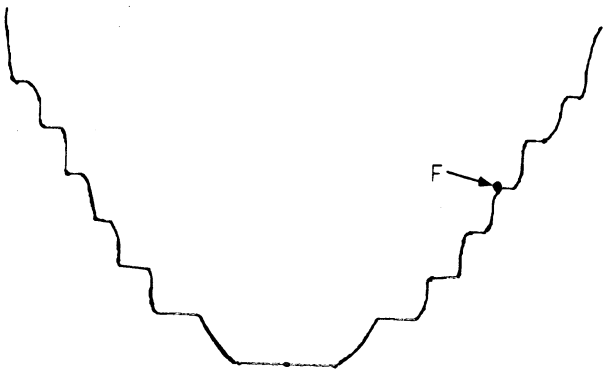


FIGURE 3. The landscape of J. There is only one unique minimum which is the flat bottom, and there are in total $\sum_{i=1}^{n_1} C_{n_0}^i - 1$ plateaux which are the saddle points.

Theoretically, the gradient descent search may get stuck in any of these plateaux. In general, however, the stochastic approximation learning rule of eqn (10a) does not exactly search along the gradient descent direction of J, but with some random fluctuations which will make W horizontally random walk in the flat area where the learning was stuck. As long as the probability that it moves to the unbounded margin (e.g., point F in Figure 3) is greater than zero (this is generally true when $\vec{x}$ is randomly sampled from a population), W will eventually fall down the plateau. The similar thing will happen when the learning is stuck in any one of other plateaux until it finally falls down to the bottom. This indicates that the LMSER rule will let W globally converge to its only stable points—the rotations of a matrix consisting of the first n_1 eigenvectors of Σ. That is, the row space of W is the subspace spanned by the first n_1 principal components. Thus, the LMSER rule automatically performs PSA.

In fact, the landscape of J is quite similar to the energy landscape of the linear d-p-d architecture of a two-layer forward net by back propagation (Baldi & Hornik, 1989). Thus, Theorems 2 and 3 have also built a connection between the one-linear-layer special case of our self-organizing net to Baldi and Hornik's linear d-p-d architecture of two-layer self-supervised net. The connection will explain, from the landscape viewpoint, why the two obviously different architectures, as well as the related learning rules, will lead to similar results.

3.2. By-Products: New Results About Oja's Subspace Rule

By generalizing his classical one-unit PCA rule given in 1982 to the cases of multiunits in one layer, Oja (1989) proposed his subspace rule for PCA given as follows:

$$\tau^w \frac{dW}{dt} = \vec{y}\vec{x}^t - \vec{y}\vec{y}^tW, \quad \vec{y} = W\vec{x}. \quad (12)$$

Several authors have investigated this rule (Hornik & Kuan, 1992; Krogh & Hertz, 1990). A good review about them and other PCA related methods is recently given by Baldi and Hornik (1991). Here, we prove a theorem which can build up a connection between the LMSER rule and the Oja subspace rule. Then based on this connection, we give several new results about the Oja subspace rule.

THEOREM 4. *On the average, the evolution direction of Oja's subspace rule in eqn* (12) *has a positive projection on the evolution direction of the LMSER rule given in eqn* (10*a*). *Precisely,* $E(vec[G_0])^t E(vec[G]) = 2E(vec[G_0])^t E(vec[G_0]) > 0$, *where* $G_0 = \vec{y}\vec{x}^t - \vec{y}\vec{y}^t W$, $G = \vec{y}(\vec{x} - \vec{u})^t + (\vec{y} - \vec{y}')\vec{x}^t$, *and "vec" transforms a matrix into a column vector by stacking the columns of the matrix one underneath the other.*

Proof. Since G_0 is the first term of G, we have

$$E(\text{vec}[G_0])^t E(\text{vec}[G]) = E(\text{vec}[G_0])^t E(\text{vec}[G_0]) + P_0, \; P_0 = E(\text{vec}[G_0])^t E(\text{vec}[(\vec{y} - \vec{y}')\vec{x}^t])$$

By $\text{vec}[A]^t\text{vec}[B] = \text{tr}(AB^t) = \text{tr}(B^tA)$, we further have

$$\begin{aligned} P_0 &= \text{tr}(E(G_0)E((\vec{y} - \vec{y}')\vec{x}^t)) \\ &= \text{tr}((W\Sigma - W\Sigma W^tW)(W\Sigma - WW^tW\Sigma)^t) \\ &= \text{tr}(W\Sigma(W\Sigma)^t) - \text{tr}(W\Sigma W^tW(W\Sigma)^t) \\ &\quad - \text{tr}(W\Sigma(WW^tW\Sigma)^t) + \text{tr}(W\Sigma W^tW(WW^tW\Sigma)^t). \end{aligned}$$

To Notice that

$$\begin{aligned} \text{tr}(W\Sigma W^tW(WW^tW\Sigma)^t) &= \text{tr}(W(WW^tW\Sigma)^tW\Sigma W^t) \\ &= \text{tr}(W\Sigma W^tW(W\Sigma W^tW)^t), \end{aligned}$$

Summing up the first, second and fourth term and an extra second term, the result is $\text{tr}(E(G_0)E(G_0)^t) > 0$. Thus we have $P_0 = \text{tr}(E(G_0)E(G_0)^t) + P_0'$ with P_0' given by

$$\begin{aligned} P_0' &= \text{tr}(W\Sigma W^tW(W\Sigma)^t) - (W\Sigma(WW^tW\Sigma)^t) \\ &= \text{tr}(\Sigma W^tW\Sigma W^tW) - \text{tr}(\Sigma^2 W^tWW^tW). \end{aligned}$$

Since W^tW is semipositively defined, and Σ is positively defined. It is well known that there should be a rotation matrix Φ, $\Phi^t\Phi = I$ such that $\Sigma = \Phi\Lambda\Phi^t$, $W^tW = \Phi D\Phi^t$, and Λ, D are diagonal matrix with their elements ≥ 0. Putting them into P_0' and noticing $\Phi^t\Phi = I$, we have

$$\begin{aligned} P_0' &= \text{tr}(\Phi^t\Lambda D\Lambda D\Phi) - \text{tr}(\Phi^t D\Lambda^2 D\Phi) \\ &= \text{tr}(\Phi^t\Lambda^2 D^2\Phi) - \text{tr}(\Phi^t\Lambda^2 D^2\Phi) = 0. \end{aligned}$$

Thus, in summary, we have $E(\text{vec}[G_0])^t E(\text{vec}[G]) = 2E(\text{vec}[G_0])^t E(\text{vec}[G_0]) > 0$. Q.E.D.

Based on this theorem, we obtain some new results about the Oja subspace rule in eqn (12).

First, for the Oja subspace rule, presently no global picture such as energy landscape has been obtained. In fact, Baldi and Hornik (1991) have shown that the rule in eqn (12) cannot be interpreted as a gradient descent search of any energy function. But they did not exclude the possibility that the rule may have a positive scalar product with the gradient of some energy function and thus still perform a down hill search. Theorem 4 confirms their conjecture. The LMSER function $J = \frac{1}{2}E(\|\vec{x} - \vec{u}\|^2)$ is such an energy function.

Second, for the Oja subspace rule, it is only known that the rotations of the first n_1 principal eigenvectors are the local stable points of eqn (12). There is no global picture about how the rule works. Here, Theorem 3 and Theorem 4 together give a global picture of the search process of Oja's subspace rule. The search process is quite similar to the search process of the LSMER rule described earlier. When the search is stuck at a plateau of J, the random walks produced by the stochastic approximation rule and the random sampling of input patterns will eventually make W fall down the plateau until it finally reaches the only bottom of the landscape of J. That is, like the LMSER rule, Oja's subspace rule has the global convergence. This is quite a strong result. Presently in the literature, as pointed out by Hornik and Kuan (1991), only local convergence analysis about the Oja subspace rule as well as other related PCA learning rule has been made and the global convergence analysis on these rules seem to be extremely challenging. Here, although our analysis is not extremely strict, i.e., we have not strictly shown that under what condition the random walks of W on each plateau has a positive probability to reach the unbound margin of the plateau (we believe this condition should be very mild for the stochastic approximation algorithm, even the independent random sampling of patterns alone may be enough), this analysis does supply a reasonable global analysis.

Third, the Oja subspace rule is regarded as a rule which needs nonlocal computations and thus is not so "biologically plausible" (Hornik & Kuan, 1992; Oja, 1989). However, from eqns (10a) and (9a) and eqn (8a) and eqn (2), we see that eqn (12) can be obtained by just dropping the second term in the LMSER rule for one layer net. As shown in Section 2.3 and Figure 1(b), the LMSER rule involves only local computations. Thus, Oja's subspace rule, which involves only a part of the computation needed by the LMSER rule, can also be reformulated and implemented as a local algorithm and should also be as "biologically plausible" as other local algorithms.

To obtain more insights about the Oja rule and the LMSER rule, we show some experimental examples.

We let $\vec{x}$ be a 3-D vector coming from a 3-D population of 400 samples with zero mean. These samples locate on a ring in R^3 space. Its projection on $x - y$, $y - z$, and $x - z$ planes are shown in Figure 4. Figure 5 and Figure 6 show the results of one-unit and two-unit case, respectively. These simulations demonstrate how weight vectors converge as learning proceeds.

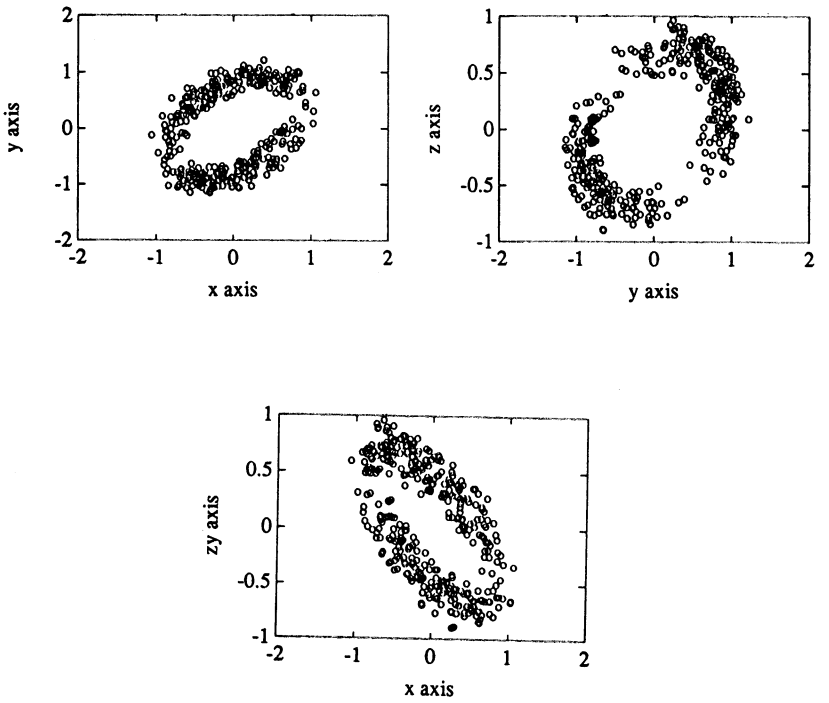


FIGURE 4. The projections of 400 three dimensional samples with zero mean.

For simplicity, we fixed the learning rate for both rules with $\alpha = 0.01$ for one unit case and $\alpha = 0.05$[4] for two units case and let both cases start from the same randomly chosen initial of weight vector.

The results of the two rules are quite similar except that the Oja rule fluctuates a little bigger but converges a little faster than the LMSER rule. We also tried different initial weights. For the one-unit case, the weight vector always converges to the first eigenvector for both rules. However, for the two-unit case, as shown in Figure 7, the weight matrix converged to different values for different rules, which the different values locate all at the bottom of energy landscape. This is because of the arbitrary rotation matrix R given in the above theorems.

Before closing this subsection, we would like to make a few remarks on some related work. As indicated in Section 2.1, Theorem 2 and Theorem 3 build a connection between the LMSER rule for one layer linear self-organizing net and Baldi and Hornik's (1989) work of using linear d-p-d architecture of two layer self-supervised net by back propagation. Theorem 4 has further linked such a connection to the Oja subspace rule. Recently, in the deep study of their d-p-d architecture which minimizes $E(\|\mathbf{x} - AB\mathbf{x}\|^2)$ by using two weight matrices: A for one d-p layer, B for other p-d layer, Baldi and Hornik (1991) realized the possibility of minimizing $E(\|\mathbf{x} - AA'\mathbf{x}\|^2)$ (which is equivalent to J) may behave similar to Oja's rule (called the Symmetric Algorithm). In fact, they have derived the gradient descent learning rule for minimizing this error function in the special case of only one unit, and roughly showed that it approximates Oja's one unit rule when the normal of weight vector is approaching 1. However, they found the rule "is not particularly simple and local."[5] This may be one reason that they did not further explore this possibility. Hrycej (1990) also intuitively connected the Oja rule to the error function J and thought that Oja's rule exactly performs the gradient descent search of J. Unfortunately, this thought is incorrect since Baldi and Hornik (1991) have shown that the Oja rule cannot do gradient descent search of any energy function.

3.3. Performing PCA by Symmetrically Circuited Nets

As shown above, with the simple activation $s(x) = x$, both the LMSER rule and the Oja rule let the weights of n_1 units of the symmetrically circuited net in Figure 2 to converge not to the first n_1 principal component but to an arbitrary rotation of the first n_1 principal components. That is, these weight vectors collectively tune to the n_1-D principal subspace spanned by the n_1 principal components of the input data. The output

[4] Where the learning rate α is involved due to the following discrete implementation of eqn (10a):

$$W(t+1) = W(t) + \alpha[\vec{y}(\vec{x} - \vec{u})' + (\vec{y} - \vec{y}')\vec{x}'].$$

[5] This is because of their formulation. In our formulation, even for the multiunits case, the LMSER rule is local and quite simple.

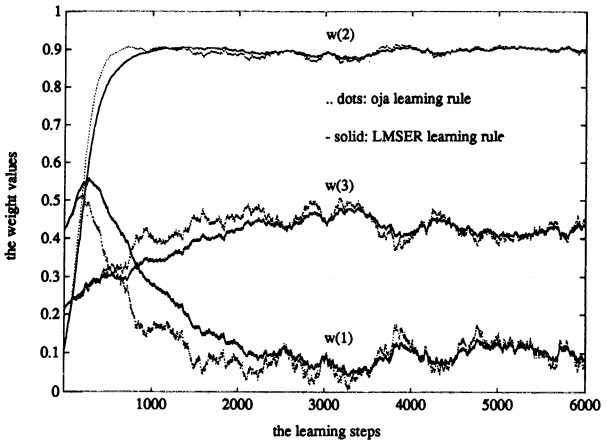
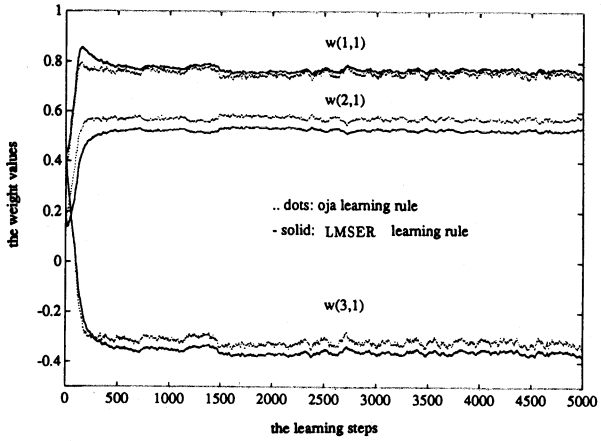


FIGURE 5. The comparison of the learned weights by the LMSER rule and the Oja rule in the one linear unit case. The results of two rules are quite similar. The weight vector $\vec{w}$ is converged to the first principal component of the data given in Figure 4.

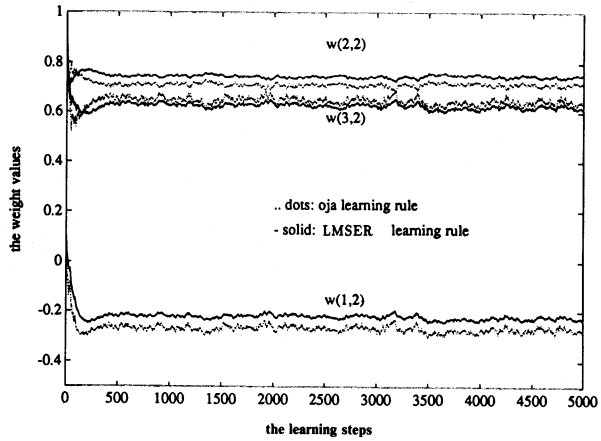
signals of n_1 units are not decorrelated, but be smeared with the equal variance on the average. Each unit has not been specialized for selecting one feature dimension. Instead, on the average, each unit equally shares all of the n_1 feature dimensions. Thus, the rules do not perform the true PCA, but a "collective version" of PCA, or PSA.[6] In comparison with a true PCA (i.e., each unit tunes to one component, e.g., made by Sanger's GHA, 1989), PSA may have disadvantages. First, the number n_1 is critical. It needs to be externally predefined. If it is too small, the net is incapable of capturing all the major features of complex patterns. If it is too large, the net will inefficiently use many units to represent simple patterns. While PCA can use a small number of units (even one unit) to extract the principal features of patterns, when the patterns are too complex and there remain features not extracted by the previously used units, more units will join in for extracting the extra features. Second, from the view of practical application, PCA has more information compression ability than the collective one (Sanger, 1989).

Presently there are two ways for realizing PCA by a neural net. One is due to Sanger's (1989) GHA or Rubner and Schulten's (1990)'s approach, which needs to design externally an asymmetrical circuit for neural units. The other way is to combine Oja's one unit rule and the competitive learning (Barrow, 1987). The competitive learning can be implemented by either the "hardcut" Winner-Take-All (WTA) or the lateral interaction of units, where the latter implements an approach more plausible for biological system. However, for the linear units, the combined use of Oja's one unit rule and the symmetrical lateral interactions still perform similarly to the LMSER and Oja subspace rules (Foldiak, 1989; Hornik & Kuan, 1992). Again, a particular asymmetrical circuit should be externally "hardwired" (Hornik & Kuan, 1992; Rubner & Tavan, 1989) in order to perform PCA. These results seem to give a feeling that some external hardwiring of an asymmetrical circuit seems necessary and the learning rules for these asymmetrical circuits seem superior to the learning rules (like the Oja subspace rule) for the symmetrical circuit shown in Figure 2 (Hornik, 1991).

However, in this subsection we will reveal an interesting fact that a slight modification of the Oja subspace rule in eqn (12), as well as the LMSER rule in eqn (10a) is able to let the symmetrically circuited net shown in Figure 2 to perform PCA. Two kinds of modifications will be proposed and the results of theoretical analysis, as well as experimental simulations, will be



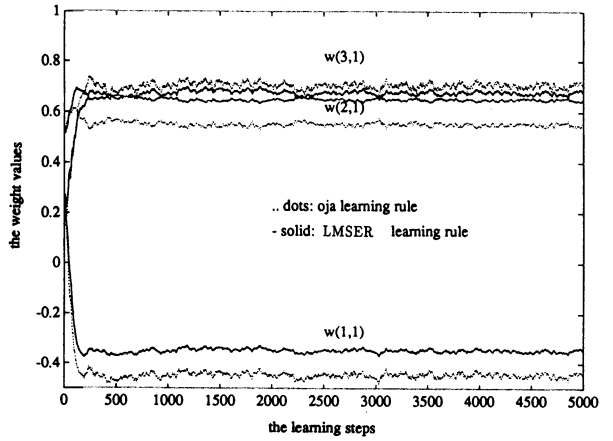
(a)



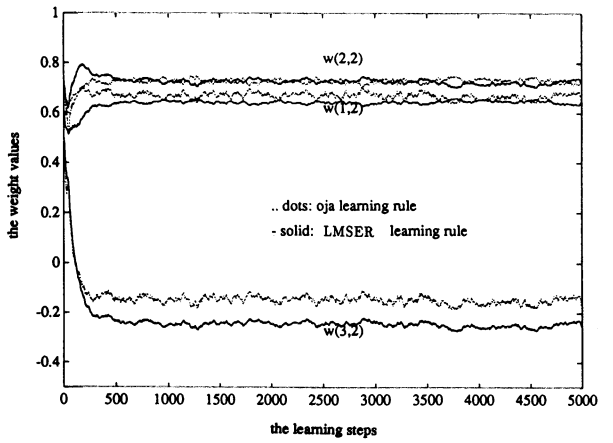
(b)

FIGURE 6. The comparison of the learned weights by the LMSER rule and the Oja rule in the two linear units case. Again, the similar behaviors as in Figure 5 are observed. (a) and (b) are, respectively, the learning curves of $\vec{w}_1$, and $\vec{w}_2$, which converged to a rotation of the first and second principal components of the data given in Figure 4.

[6] This is also the reason that Oja called his rule a subspace rule but not a PCA rule.



(a)



(b)

FIGURE 7. The results are obtained in the same condition as those given in Figure 6 for another set of initial weights. Now, the weights for two rules converged to the two different rotations of the first and second principal components of the data. This is because of the arbitrary rotation matrix R given in Theorems 2, 3, and 4.

provided to show how they perform PCA without externally hardwiring for forming an asymmetrical circuit.

The first modification is to slighly change eqn (12) into eqn (13) or eqn (10a) into eqn (14):

$$\tau^w \frac{dW}{dt} = \vec{z}\vec{x}^t - \vec{y}\vec{u}^t, \tag{13}$$

$$\tau^w \frac{dW}{dt} = \vec{z}\vec{x}^t - \vec{y}\vec{u}^t + \vec{z}\vec{x}^t - \vec{y}^r\vec{x}^t. \tag{14}$$

where $\vec{y}$, $\vec{u}$, $\vec{y}^r$ are still the same as they are defined in eqns (9a) and (9b). But here $\vec{z}$ is not the simplest case $\vec{z} = \vec{y}$, it is given by

$$\vec{z} = S(\vec{y}) = A_m\vec{y}, \quad A_m = \text{diag}[a_1, \ldots, a_{n_1}] \tag{15}$$

and $a_1 > a_2 > \ldots > a_{n_1}$ are all positive. That is, the activation function $S(.)$ is still linear, but for each unit the output y_j is amplified by a different factor a_j (i.e., $z_j = a_j y_j, j = 1, \ldots, n_1$).

By observing that $\vec{y}\vec{x}^t$ is the Hebbian term for the simplest activation $s(y) = y$ and $\vec{z}\vec{x}^t$ is the Hebbian term for the activation $z_j = a_j y_j$, $a_j > 0$, one can see that eqns (13) and (14) are obtained from eqns (12) and (10a) by slightly changing the Hebbian term in accordance with the change of activation function. One can also find that such changes do not involve any external hardwiring of an asymmetrical circuit. The symmetrical circuit shown in Figure 2 is still used. But in this case there does exist some external design of an asymmetry—asymmetrical amplifying factors for the neural units.

In the sequel, we will show that both eqns (13) and (14) can automatically perform PCA.

As in eqn (10), we take expectation on both sides of eqn (13) and eqn (14), resulting in

$$\tau^w \frac{dW}{dt} = A_m W\Sigma - W\Sigma W^t A_m W, \tag{15a}$$

$$\tau^w \frac{dW}{dt} = A_m W\Sigma - W\Sigma W^t A_m W + A_m W\Sigma - WW^t A_m W\Sigma. \tag{15b}$$

In a way similar to what we did in proving Theorem 2, we can also prove Theorem 5 which states that all of the possible converged points of the weight matrix W, i.e., all the critical points of eqns (15a) and (1b), are the matrices consisting of n_1 eigenvectors of Σ as their row vectors.

THEOREM 5. *Assume $n_1 < n_0$ and Σ is nonsingular. Let $\Phi = [\vec{\phi}_1, \ldots, \vec{\phi}_{n_0}]$ and $\Lambda = diag[\lambda_1, \ldots, \lambda_{n_0}]$ be the matrices of eigenvectors and eigenvalues of Σ, respectively, then the critical points of eqn (15a) and (15b) are $W = DP\Phi^t$, where $D = [D_1|0]$ is $n_1 \times n_0$ matrix with D_1 being an $n_1 \times n_1$ diagonal matrix and its diagonal elements only taking value of $+1$, -1, 0. $P_{n_0 \times n_0}$ is an arbitrary permutation matrix.*

Proof. Let $dW/dt = 0$, we know that the critical points of eqns (15a) and (15b) satisfy

$$A_m W\Sigma - W\Sigma W^t A_m W = 0, \tag{16a}$$

$$A_m W\Sigma - W\Sigma W^t A_m W + A_m W\Sigma - WW^t A_m W\Sigma = 0. \tag{16b}$$

By singular value decomposition and through a derivation as the first part of the proof of Theorem 2, we also have $W = R_{n_1 \times n_1} D_{n_1 \times n_0} P\Phi^t$; put it into eqns (16a) and (16b) and notice that $R^tR = I$, $\Phi^t\Phi = I$, $PP^t = I$, and $P^tP = I$, we have

$$A_m RDP\Lambda P^t - RDP\Lambda P^t D^t R^t A_m RD = 0,$$

$$A_m RDP^t\Lambda P^t - RDP\Lambda P^t D^t R^t A_m RD + A_m RDP\Lambda P^t - RDD^t R^t A_m RDP\Lambda P^t = 0.$$

The equalities hold only when every term is a diagonal matrix. Since A_m and $P\Lambda P^t$ are both diagonal, in order to ensure $A_m RDP\Lambda P^t$ be diagonal, it must be that R is diagonal. Further from $R^tR = I$, we see $R = I$.

Furthermore, by putting $R = I$ into the two equalities and following the same way as we did in the proof of Theorem 2, we can see that D_1 has only the diagonal elements with values $+1, -1, 0$. Q.E.D.

Next, we need to show that all the critical points are unstable, except of the one which consists of the first n_1 eigenvectors of Σ as its row vectors.

First we consider the rule in eqn (13). One may already noticed that eqn (15a) looks quite similar to Brockett's (1991) eqn (1)—the gradient flow equation on the space of $n \times n$ orthogonal matrices:

$$\dot{\Theta} = -\Theta N \Theta^t Q \Theta + Q \Theta N, \tag{17}$$

which has only one stable critical point that Θ consists of the n eigenvectors of the symmetrical matric Q as the column vectors and these eigenvectors are ordered in such a way that the order of eigenvalues $\lambda_1, \ldots, \lambda_n$ is the same as the order of the diagonal elements $n_{11}, \ldots, n_{nn}$,[7] where $N = \text{diag}[n_{11}, \ldots, n_{nn}]$ is a $n \times n$ positive diagonal matrix.

However, we shoud point out that eqn (15a) is different from Brockett's equation on two points. One is that in eqn (17), Θ is a $n \times n$ square matrix, while in eqn (15a), W is a $n_1 \times n_0$ nonsquare matrix. The other is that in eqn (17), Θ is constrained to always belonging to the set $SO(n)$ of $n \times n$ orthogonal matrices with a positive determinant, while in eqn (15a), the initial W can be of any $n_1 \times n_0$ matrix. Therefore, Brockett's results on eqn (17) cannot be directly brought to eqn (15a). Fortunately, we can use some of his results to reach our goal here.

Similar to the way that Brockett used to show that eqn (17) is the gradient ascent equation which maximizes $\text{tr}(\Theta^t Q \Theta N)$ in the constraint of Θ varying among $SO(n)$, we can also show that eqn (15a) is the gradient ascent equation which maximizes[8]

$$J_t = \text{tr}(W \Sigma W^t A_m) \tag{18}$$

in the constraint that W varies among $SO(n_0, n_1)$—a set consists of matrices which contain n_1 mutually orthogonal n_0 dimensional vectors as their row vectors.

Moreover, we know from Brockett's paper (1991) that if Θ starts at an element of $SO(n)$, eqn (17) will let Θ always remain in $SO(n)$. Similarly, here we also have that if W starts at an element of $SO(n_0, n_1)$, eqn (15a) will let W always remain in $SO(n_0, n_1)$. As described in Theorem 5 we know that from any initial values, W will finally reach one of critical points, i.e., W will finally reach one element of set $SO(n_0, n_1)$. Therefore, we see that eventually eqn (15a) will evolve only within $SO(n_0, n_1)$ to maximize J_t and thus the evolving process can be explored through the landscape of J_t which is described by the following theorem:

THEOREM 6. *Assume* $\lambda_1 > \lambda_2 \ldots \lambda_{n_1} > \lambda_{n_1+1} \geq \lambda_{n_0} > 0$ *and* $\Phi = [\vec{\phi}_1, \ldots, \vec{\phi}_{n_0}]$, *then all of the critical points of eq. (15a) given by Theorem 5 are saddle points of the landscape of* J_t, except the one with $W = D\Phi^t$, *and* $D = [D_1 | \vec{0}]$ *with* D_1 *being diagonal matrix and its diagonal elements being either* $+1$ *or* -1. *Furthermore, this* $W = D\Phi^t$ *lets* J_t *reach its only local (also global) maximum* $J_t^{max} = \sum_{i=1}^{n_1} a_i \lambda_i$.

Proof. Let us consider the critical points $W = DP\Phi^t$ given in Theorem 5 separately, according to the different cases of D, P:

1. Rank$[D] = r < n_1$, which means that there are only r rows of W are eigenvectors of Σ and the other rows are zero vectors. Assuming the j-th row is a zero vector, we slightly perturb the row by $\epsilon\vec{\phi}_k$ such that $\vec{\phi}_k$ is not among the r eigenvectors; then from eqn (18), we have $\Delta J_t = \epsilon^2 \lambda_k a_j$, i.e., J_t increases. Thus, these critical points are saddle points of J_t.
2. Rank$[D] = n_1$, which means that the diagonal elements of D_1 is either $+1$ or -1 and thus the n_1 rows of W are eigenvectors of Σ. Due to the permutation caused by $P \neq I$, there are three possible situations. First, not all the n_1 rows are the first n_1 eigenvectors of Σ. Assuming the j-th row is an eigenvector $\vec{\phi}_i$, $i > n_1$, we slightly perturb the row through replacing $\vec{\phi}_i$ by $(\vec{\phi}_i + \epsilon\vec{\phi}_k)/\sqrt{1+\epsilon^2}$ with $\vec{\phi}_k$ being among the first n_1 eigenvectors; Then from eqn (18), we have $\Delta J_t = \epsilon^2(\lambda_k - \lambda_i)a_j/(1+\epsilon^2)$, i.e., J_t increases. Thus, these critical points are saddle points of J_t. Second, the n_1 rows are the first n_1 eigenvectors of Σ, but they are not in the descent order. In this situation, there is at least a pair (i, j), $i < j$ such that the i-th row is $\vec{\phi}_r$ and the j-th row is $\vec{\phi}_k$ with $r > k$. We slightly perturb the i-th row through replacing $\vec{\phi}_r$ by $(\vec{\phi}_r + \epsilon\vec{\phi}_k)/\sqrt{1+\epsilon^2}$ and perturb the j-th row through replacing $\vec{\phi}_k$ by $(\vec{\phi}_k + \epsilon\vec{\phi}_r)/\sqrt{1+\epsilon^2}$; Then from eqn (18), we have $\Delta J_t = \epsilon^2(a_i - a_j)(\lambda_k - \lambda_r)/(1+\epsilon^2) > 0$ (since $a_i > a_j$, $\lambda_k > \lambda_r$), i.e., J_t increases. Thus, these critical points are also saddle points of J_t. Finally, the third situation is that $W = D\Phi^t$, i.e., $W = [\pm\vec{\phi}_1, \ldots, \pm\vec{\phi}_{n_1}]$. Now, for any pair (i, j), $i < j$, the perturbation of replacing $\vec{\phi}_i$ by $(\vec{\phi}_i + \epsilon\vec{\phi}_j)/\sqrt{1+\epsilon^2}$ and replacing $\vec{\phi}_j$ by $(\vec{\phi}_j + \epsilon\vec{\phi}_i)/\sqrt{1+\epsilon^2}$ will result in $\Delta J_t = \epsilon^2(a_i - a_j)(\lambda_j -$

[7] i.e., if λ_k is the j-th largest among $\lambda_1, \ldots, \lambda_n$, then n_{kk} is also the j-th largest among $n_{11}, \ldots, n_{nn}$.

[8] In fact, simply by letting $N = C^t A_m C$, $C = [I_{n_1 \times n_1} | 0_{n_1 \times (n_0 - n_1)}]$, and denoting $\Sigma = Q$, $W^t = \Theta C^t$ (i.e., W^t is the first n_1 column vectors of Θ), we can see that $\text{tr}(\Theta^t Q \Theta N)$ is identical to J_t and that eqn (17) becomes

$$\dot{\Theta} = -W^t A_m \Sigma \Theta + \Sigma W^t A_m C.$$

Furthermore, if we let both sides of the equation being multiplied by C^t from right and then let the resulted equation be transposed in the both sides, we can get exactly eqn (15a).

$\lambda_k)/(1+\epsilon^2) < 0$, i.e., J will decrease. Thus we see that $W = D\Phi^t$ is the only local (also global) maximum point. Furthermore, in this case, we have $J_t = \sum_{i=1}^{n_1} a_i\lambda_i$. Q.E.D.

Although Theorem 6 is for eqn (15a) and the rule in eqn (13), this theorem is also true for eqn (15b) and the rule in eqn (14). This is because of Theorem 7, which reveals that on the average, the evolution direction of eqn (13) has a positive projection on the evolution direction of eqn (14).

THEOREM 7.

$$E(\text{vec}[G_0])^t E(\text{vec}[G]) = 2E(\text{vec}[G_0])^t E(\text{vec}[G_0]) > 0,$$

where $G_0 = \vec{z}\vec{x}^t - \vec{y}\vec{u}^t$, $G = \vec{z}\vec{x}^t - \vec{y}\vec{u}^t + \vec{z}\vec{x}^t - \vec{y}^t\vec{x}^t$.

Proof.

$$E(\text{vec}[G_0])^t E(\text{vec}[G]) = 2E(\text{vec}[G_0])^t E(\text{vec}[G_0]) + \text{tr}(A),$$

and $A = (NW\Sigma - W\Sigma W^tNW)^t(W\Sigma W^tNW - WW^tNW\Sigma) = \Sigma W^tNW\Sigma W^tNW - \Sigma W^tNWW^tNW\Sigma - W^tNW\Sigma W^tW\Sigma W^tNW + W^tNW\Sigma W^tWW^tNW\Sigma$.

Since W^tNW is semipositive defined, Σ is positive defined, similar to that in Theorem 4, we have Φ, $\Phi^t\Phi = I$ such that $\Sigma = \Phi\Lambda\Phi^t$, $W^tNW = \Phi D\Phi^t$, and Λ, D are diagonal matrix with their elements ≥ 0. Put them into A and noticing $\Phi^t\Phi = I$, we have

$$A = \Phi^t\Lambda D\Lambda D\Phi - \Phi^t\Lambda DD\Lambda\Phi - \Phi^t D\Lambda\Lambda D\Phi + \Phi^t D\Lambda D\Lambda\Phi = 0.$$

Thus, we have $\text{tr}(A) = 0$ and $E(\text{vec}[G_0])^t E(\text{vec}[G]) = 2E(\text{vec}[G_0)^t E(\text{vec}[G_0]) > 0$. Q.E.D.

Theorems 2, 3, and 4 describe the evolving process of the LMSER rule and the Oja subspace rule. In a similar fashion, Theorems 6, 7, and 8 show that the learning process of rules in eqns (13) and (14) will let W climb the mountain of J_t and eventually will reach the top point $J_t^{\max} = \sum_{i=1}^{n_1} a_i\lambda_i$ for whatever initial values of W. Although the landscape has many plateaux which are the saddle points of J_t, some random fluctuations produced by stochastic approximation will drive the learning get rid of these plateaux.

Figures 8 and 9 show the simulation results of the modified Oja rule in eqn (13) and the modified LMSER rule in eqn (14) in comparison with the results of the original Oja rule in eqn (12) and LMSER rule in eqn (10a). The simulations were made on a two-units net shown in Figure 2 with the data still being that shown in Figure 4. In these simulations, the learning rates and the initial weight vectors are the same as those used in the simulation shown in Figure 7. The amplifying factor matrix used here is $A_m = \text{diag}[2, 1]$. In Figures 8 and 9, the curves show the developments of the angles between each weight vector to each of the three eigenvectors of the data given in Figure 4. The results of Figure 8 are obtained by the modified and original LMSER rules in eqns (14) and (10a). It is shown in Figure 8(a) that by the modified rule, weight vector $\vec{w}_1$ of the first unit has converged to an orientation which is nearly parallel (having an angle around 3 degrees) to that of the first eigenvector of the data, and this orientation is orthogonal (or nearly orthogonal) to both the second and third eigenvectors, respectively. In contrast, in Figure 8(b), by the original rule, $\vec{w}_1$ is only orthogonal to the third eigenvector, but parallel to neither the first or second eigenvector (having angles of 68 degrees and 160 degrees, respectively). Furthermore, in Figure 8(c), the modified rule has let the weight vector $\vec{w}_2$ of the second unit converge to an orientation which is nearly parallel (having an angle around 3 degrees) to that of the second eigenvector, and orthogonal (or nearly orthogonal) to both the first and third eigenvectors, respectively. Again, in contrast, in Figure 8(d), the original rule has let $\vec{w}_2$ be only orthogonal to the third eigenvector, but parallel to neither the first or second eigenvector (having angles of 20 degrees and 70 degrees, respectively). The results of Figure 9 are obtained by the modified and original Oja rules in eqns (13) and (12). The situations are very similar to those of Figure 8 except that again the results here fluctuate a little bigger but converge faster. So, we see that these simulation results have verified that the learning rule of eqns (13) and (14) do perform true PCA.

In the rest of this subsection, we introduce another kind of modification to the LMSER rule in eqn (10a) and the Oja subspace rule in eqn (12). This modification is even more interesting. As mentioned previously, the modifications in eqns (13) and (14) still need some external design of asymmetry on the amplifying factors $a_1, \ldots, a_{n_1}$, even though an externally hardwired asymmetrical circuit is unnecessary. Here the second kind of modifications do not need any kind of external design of asymmetry at all. In fact, one only need to include the sigmoid type nonlinearity $s(.)$ in each unit, and this nonlinearity $s(.)$ can be the same for all the units. In other words, the LMSER rule or the Oja subspace rule on the one layer net with sigmoid nonlinear units as shown in Figure 2 can perform PCA. More precisely, we can directly use the nonlinear LMSER rule in eqn (9b) and the following direct extension of the Oja subspace rule[9] to perform PCA:

$$\tau^w \frac{dW}{dt} = \vec{z}(\vec{x} - \vec{u})^t, \tag{19}$$

where $\vec{z}$ and $\vec{u}$ are the same as in eqns (9a) and (9b).

The theoretical analysis on "why the nonlinearity can cause the units to become selective automatically" seems quite difficult. A partial reason may be that the

[9] This rule is equivalent to the 4-th equation in his own generalization to nonlinear unit by Oja (1991), where he thought the equation is the most difficult one.

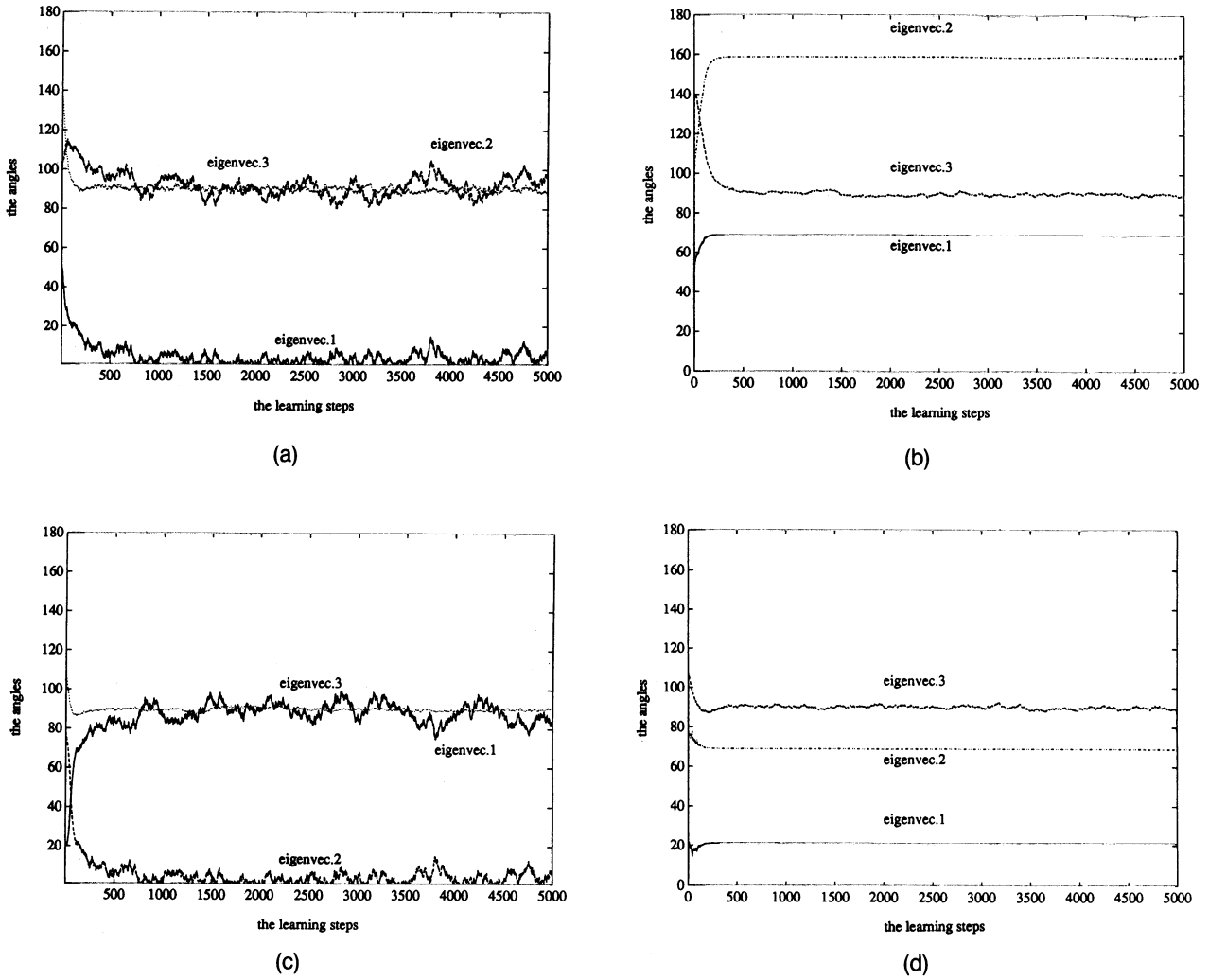


FIGURE 8. The developments of the angles between each learned weight vector to each of the three eigenvectors of the data given in Figure 4. These results are obtained by the modified LMSER rule in eqn (14) in comparison with the results of the original LMSER rule in eqn (10a). The learning rates and the initial weight vectors used here are the same as those used in the simulation shown in Figure 7. (a) With the modified LMSER rule, $\vec{w}_1$ has converged to an orientation which is nearly parallel (having an angle around 3 degrees) to that of the first eigenvector of the data, and is orthogonal (or nearly orthogonal) to both the second and third eigenvector, respectively. (b) With the original LMSER rule, $\vec{w}_1$ is only orthogonal to the third eigenvector, but not parallel to either the first or second eigenvector (having angles of 68 degrees and 160 degrees, respectively). (c) The modified LMSER rule has let $\vec{w}_2$ converge to an orientation which is nearly parallel (having an angle around 3 degrees) to that of the second eigenvector, and orthogonal (or nearly orthogonal) to both the first and third eigenvectors, respectively. (d) The original LMSER rule $\vec{w}_2$ is only orthogonal to the third eigenvector, but not parallel to either the first or second eigenvector (having angles of 20 degrees and 70 degrees, respectively).

amplifying factor $s'(y)$ changes as y takes different values, and that the y values of different units are usually different for the same input $\vec{x}$. As a result, the different $s'(y)$ values taken by these units together behave somewhat like that A_m did in eqn (15). Instead of exploring theoretical analysis, we demonstrate some experimental results in Figures 10–15. In these experiments we simply let the nonlinear sigmoid function be given by

$$s(x) = \tanh(\beta x) = \frac{e^{\beta x} - e^{-\beta x}}{e^{\beta x} + e^{-\beta x}} \qquad (20)$$

and use parameter $\beta = 3$. Figure 10 and Figure 11 are the counterparts of Figure 6 and Figure 7, respectively. They are obtained by using the same learning rate and initial weights as their counterparts. The LMSER rule used here is slightly simplified from eqn (9b) by letting $S' = I$ but now $\vec{y}' = W\vec{u}$, $\vec{u} = W'\vec{z}$, $\vec{z} = S(\vec{y})$, i.e., the sigmoid nonlinearity takes its main role in getting $\vec{z}$. From Figure 11 we can observe that unlike its counterpart Figure 7, the weights by both the LMSER rule and the Oja rule converge to the similar values. This is because the sigmoid nonlinearity has reduced the arbitrariness caused by the rotation matrix R in Theorem 4.

The results are also shown in Figures 12 and 13, which correspond to Figure 10, and in Figure 14 and

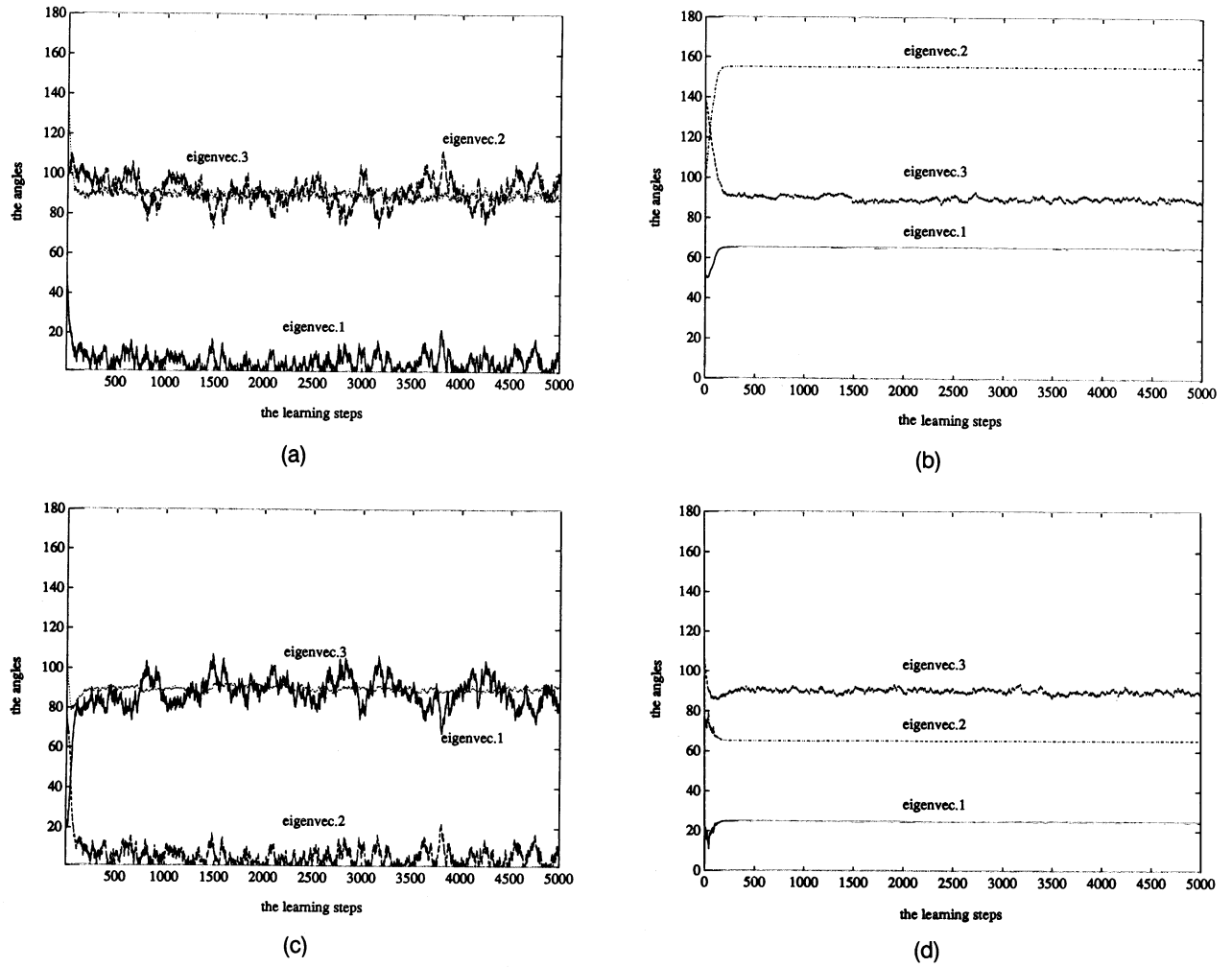
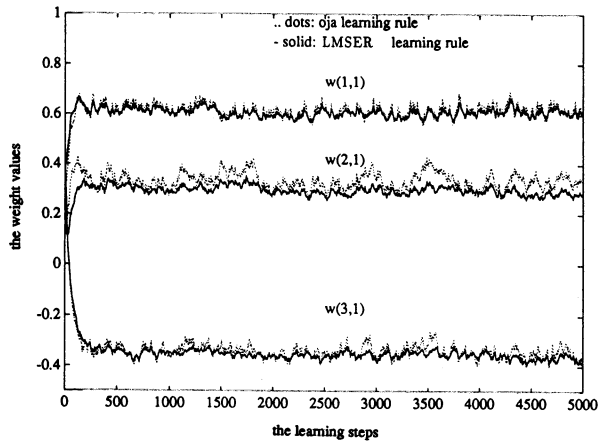


FIGURE 9. All are the same as in Figure 8 except that Figures 9(a) and (c) and Figures 9(b) and (d) are obtained by the modified Oja rule in eqn (13) and original Oja rule in eqn (12), respectively. The situations are very similar to those of Figure 8.

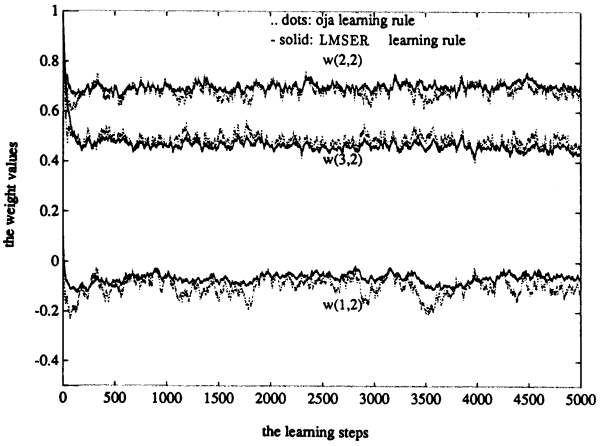
15, which correspond to Figure 11. In these figures, the curves show the developments of the angles between each weight vector to each of the three eigenvectors of the data given in Figure 4. The results of Figure 12 are obtained by the LMSER rule. It is shown in Figure 12(a) that with the nonlinear activation, the weight vector $\vec{w}_1$ of the first unit has converged to an orientation which is nearly parallel (but in the opposed direction, i.e., having an angle around 170 degrees) to that of the second eigenvector of the data, and this orientation is orthogonal (or nearly orthogonal) to the first and the third eigenvectors, respectively. In contrast, Figure 12(b), with the linear activation $\vec{w}_1$ is only orthogonal to the third eigenvector but parallel to neither the first nor second eigenvector (i.e., having angles of 30 degrees and 60 degrees, respectively). Furthermore, in Figure 12(c), the nonlinear activation has make the weight vector $\vec{w}_2$ of the second unit converge to an orientation which is nearly parallel (having an angle below 10 degrees) to that of the first eigenvector, and orthogonal (or nearly orthogonal) to the third and second eigenvectors, respectively. Again in contrast, in Figure 12(d), the linear activation $\vec{w}_1$ is only orthogonal to the third eigenvector but not parallel to either the first or second eigenvector (i.e., having angles of 60 degrees and 150 degrees, respectively). Thus, we see that the sigmoidal nonlinearity does make units become selective with each sensitive to one principal direction. That is, now the net performs true PCA.

For the results of Figure 13 obtained by the Oja rule, the situations are very similar to those of Figure 12 except that again the results here fluctuate a little bigger but converge faster than those in Figure 4. Furthermore, the similar things happen in Figures 14 and 15. By nonlinear LMSER and the Oja rule, $\vec{w}_1$ and $\vec{w}_2$ converge to the direction nearly in parallel to the first and second eigenvectors, respectively. That is, the net also performs PCA.

We also tried many other random initial values for weight vectors. For linear activation, the obtained re-



(a)



(b)

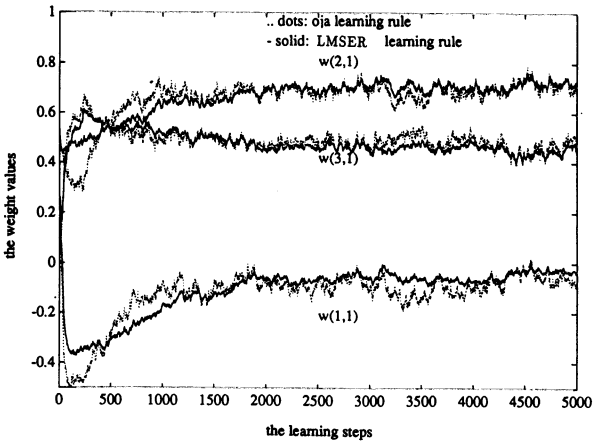
FIGURE 10. The comparison of the learned weights by the nonlinear LMSER rule and the nonlinear Oja rule in the two units case were obtained under the same initial weights and learning rate as those used in Figure 6. (a) and (b) are, respectively, the learning curves of $\vec{w}_1$ and $\vec{w}_2$, which now directly tend to the directions of second and first principal components of the data given in Figure 4.

sults may change considerably due to the arbitrary rotations. However, for nonlinear activation, the results remain nearly unchanged with different initial conditions subject to the reversed directions or the switch from that $\vec{w}_1$ tuned the first eigenvector and $\vec{w}_2$ tuned the second to that $\vec{w}_1$ tuned the second and $\vec{w}_2$ tuned the first. We varied β for $\tanh(\beta x)$ in eqn (20), and the similar results were obtained as well.[10] So, we have experimentally confirmed that nonlinear activation does let the LMSER and Oja rules perform PCA.

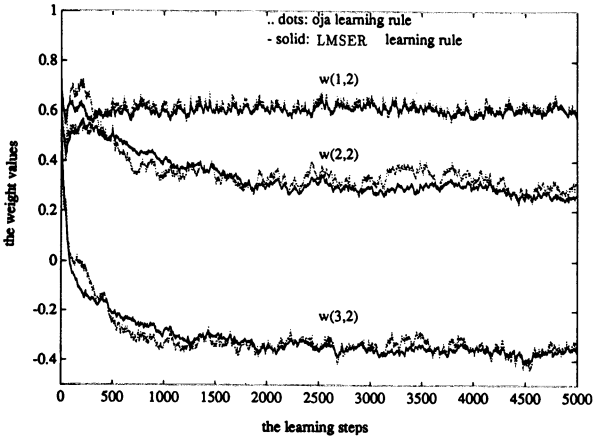
Finally, before ending this subsection we give some further remarks:

1. Since the nonlinear LMSER and Oja rules, as well as the modified Oja rule in eqn (13) and the LMSER rule in eqn (14) perform true PCA, they should be as capable as Sanger's GHA and other asymmetrically circuited nets for practical tasks such as data compression. For the same reason, they should also be able to give an interpretation similar to those given by Sanger (1989) and Rubner and Schulten (1990) for the emergence of the selective cells in the cortical receptive field. More interestingly, the nonlinear rules can perform PCA without any externally "hardwired" asymmetry as made by Sanger (1989) and Rubner and Schulten (1990). The symmetry is naturally broken by the nonlinearity of sigmoid function.
2. If we let $A_m = I$ in eqns (15) and (18), we will see that the rule in eqn (13) returns back to the Oja subspace rule in eqn (12), and thus we see that al-

[10] However, too small or too large β will influence the performances.



(a)



(b)

FIGURE 11. The comparison of the learned weights by the nonlinear LMSER rule and the nonlinear Oja rule in the two units case were obtained under the same initial weights and learning rate as those used in Figure 7. Being different from those in Figures 7 or 8, now due to the function of the sigmoid nonlinearity, $\vec{w}_1$ and $\vec{w}_2$ by both the LMSER rule and the Oja rule converge to the similar values, i.e., the directions of first and second principal components of data, respectively.

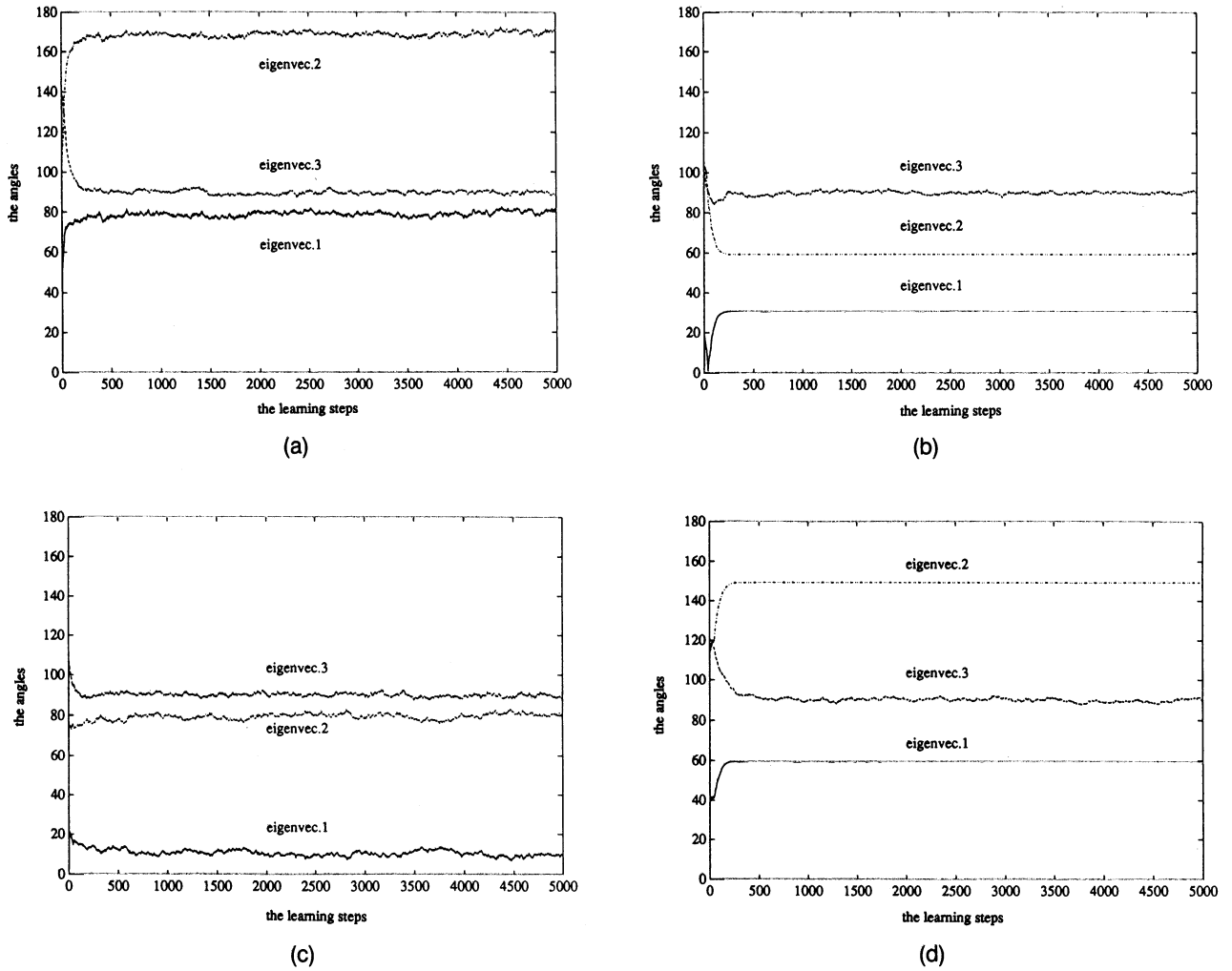


FIGURE 12. The developments of the angles between each learned weight vector to each of the three eigenvectors of the data given in Figure 4. These results are obtained by the LMSER rule and correspond to those in Figure 10. (a) With the nonlinear activation, $\vec{w}_1$ has converged to an orientation which is nearly parallel (having an angle around 170 degrees) to that of the second eigenvector of data, and is orthogonal (or nearly orthogonal) to the third (or the first) eigenvector, respectively. (b) With the linear activation, $\vec{w}_1$ is only orthogonal to the third eigenvector, but not parallel to either the first or second eigenvector (i.e., having angles of 30 degrees and 60 degrees, respectively). (c) The nonlinear activation has let $\vec{w}_2$ converge to an orientation which is nearly parallel (having an angle below 10 degrees) to that of the first eigenvector, and orthogonal (or nearly orthogonal) to the third (or the second) eigenvector, respectively. (d) With the linear activation, $\vec{w}_2$ is only orthogonal to the third eigenvector, but not parallel to either the first nor second eigenvector (having angles of 60 degrees and 150 degrees, respectively). We see that the sigmoid nonlinearity does make units become selective.

though the Oja rule cannot be interpreted as a unconstrained gradient descent search of any energy function (Baldi & Hornik, 1991), the rule can be interpreted as a gradient ascent search of $\mathrm{tr}(W\Sigma W^t)$ in the constraint that W always satisfies $WW^t = I$. Moreover, Theorem 5, plus a small modification of Theorem 6,[11] provides also a description of the global picture about the convergence of the Oja subspace rule.

3. From Figures 12–15, we can also observe that there is some bias between the desired eigenvectors and the converged vectors obtained by the nonlinear LMSER and Oja rules. For example, in Figure 12(a), $\vec{w}_1$ does not exactly approach the second eigenvector but with an angle around 170 degrees, also it is not exactly orthogonal to the first eigenvector but with an angle around 80 degrees. The reason is that due to the nonlinear activation, the rules may not perform exactly PCA as well as even PSA. It may be interesting to study further the impacts of these bias, which may be a reason that Oja

[11] i.e., using $W = PD\Phi^t$ to replace $W = D\Phi^t$, where P is a $n_1 \times n_1$ permutation matrix.

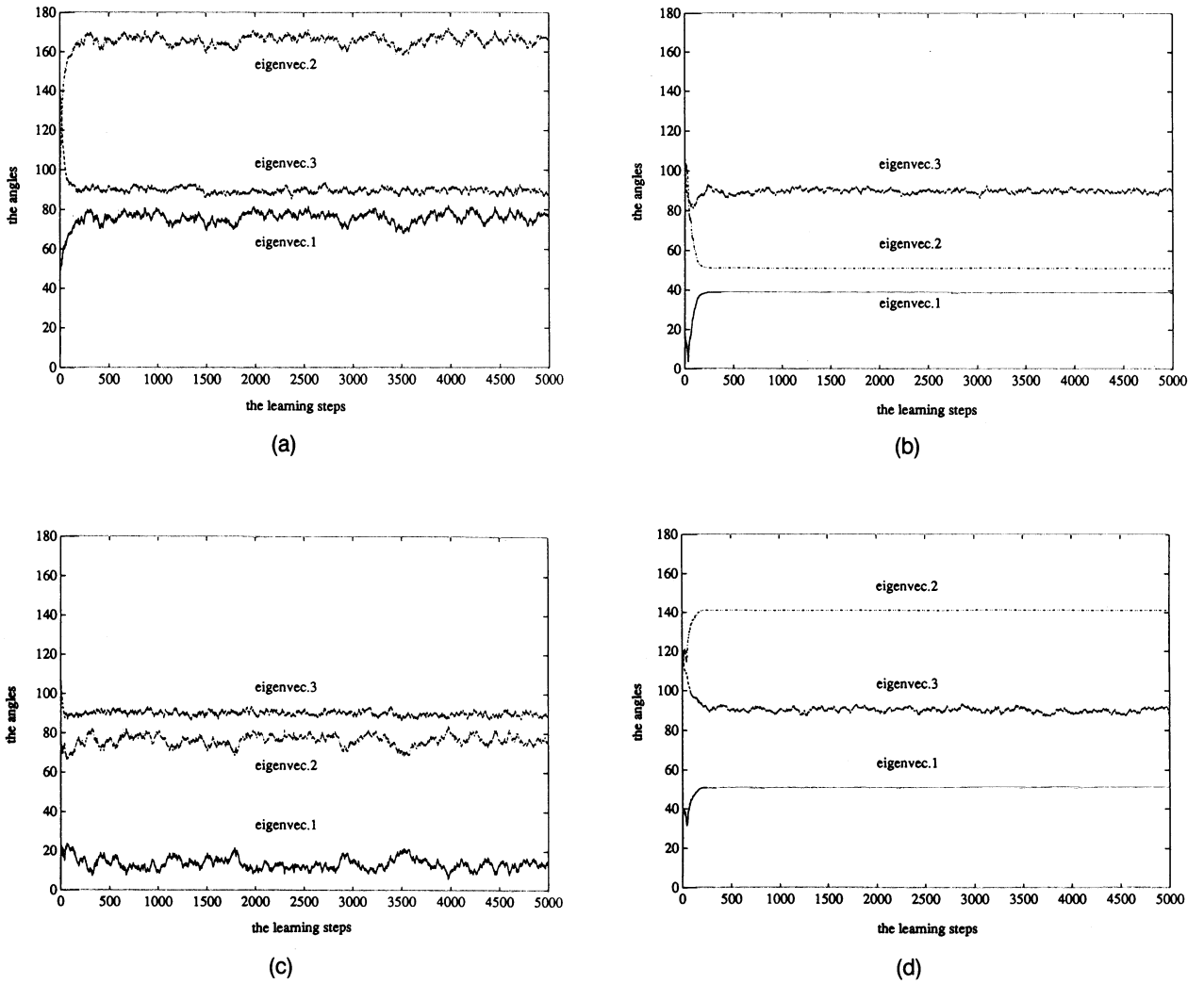


FIGURE 13. All are the same as in Figure 12 except that these results are obtained by the Oja rule. The situations are very similar to those of Figure 12.

(1991) recently observed that the sigmoid is able to reduce strong noise effectively.

4. We may also form conjectures about other symmetrical PCA nets such as that given by Foldiak (1989); they may also perform true PCA when the sigmoid nonlinearity is included. It is also interesting to study further the functions of the "magic sigmoid". Grossberg (1973) showed that in a laterally feedback competitive net, the sigmoid activation can suppress noise, contrast enhance suprathreshold activities, normalize total activity, and store the contrast enhanced and normalized pattern in short term memory. The sigmoid was also shown by Oja (1991) to be able to suppress strong outlier for PCA. It also takes a key role in back propagation and the fully connected associative memory model. In this subsection, we have shown that it can make the LMSER rule or Oja rule perform true PCA.

5. CONCLUSIONS

We proposed a new self-organizing net based on the LMSER principle. The net has a general architecture of either one layer or multilayer. We proved the stability of its dynamic process in the perception phase, and derived the local learning rule which performs gradient descent of the LMSER. Particularly for one layer with n_1 linear units, we have shown that the LMSER rule will let their weight vectors converge to some rotations of the first n_1 principal eigenvectors of the covariance matrix of input data. These converged points are stable and corresponding to the global minimum of the landscape of J. This landscape has no other local minimum points but many saddle points. Furthermore, we have also shown that on the average the evolution direction of the Oja subspace rule has a positive projection on the evolution direction of the LMSER rule. This con-

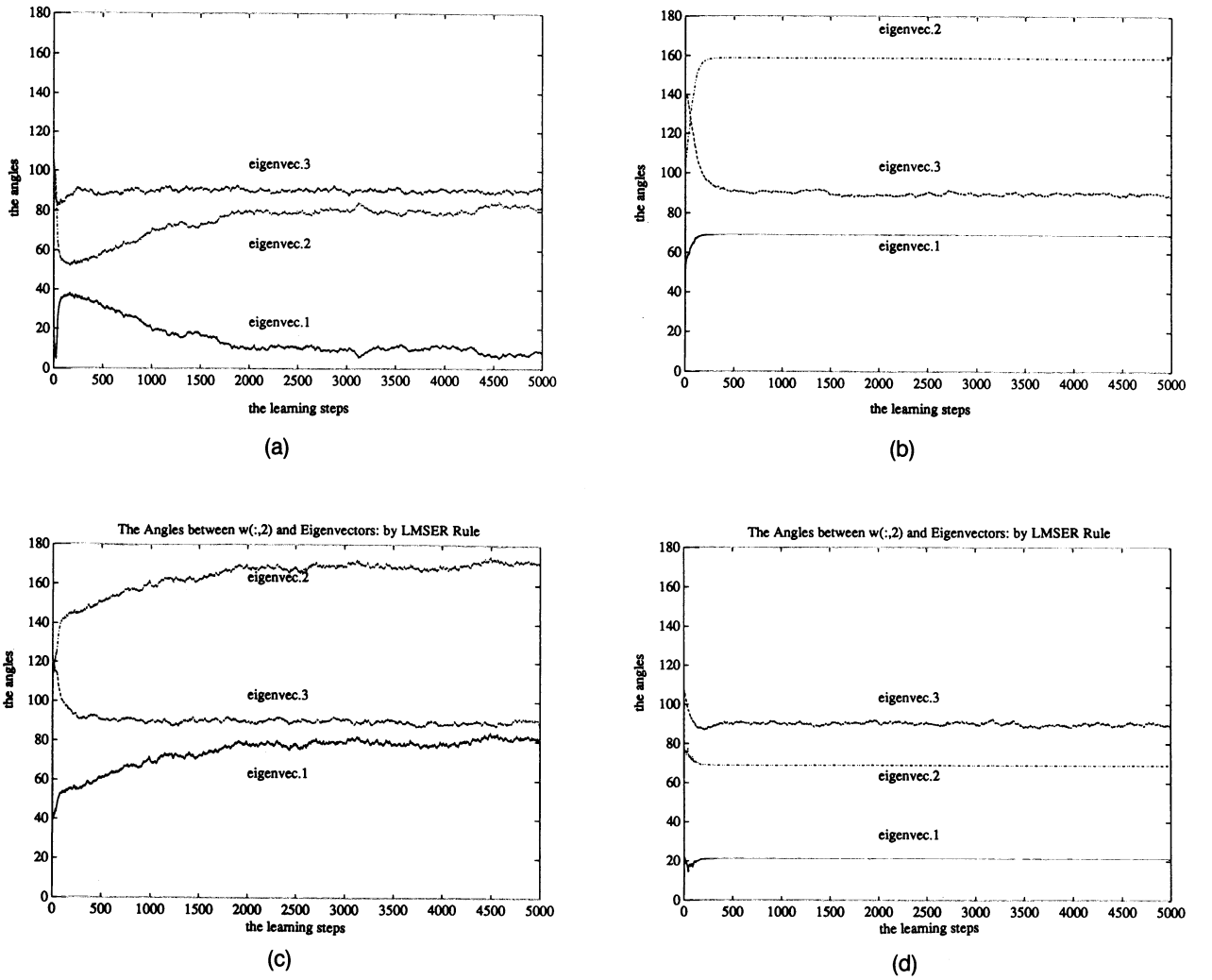


FIGURE 14. These results are obtained by the LMSER rule and correspond to those in Figure 11. (a) With the nonlinear activation, $\vec{w}_1$ has converged to an orientation which is nearly parallel (having an angle below 10 degrees) to that of the first eigenvector, and is orthogonal (or nearly orthogonal) to the third (or the second) eigenvector, respectively. (b) With the linear activation, $\vec{w}_1$ is only orthogonal to the third eigenvector, but not parallel to either the first or second eigenvector. (c) The nonlinear activation has let $\vec{w}_2$ converge to an orientation which is nearly parallel (having an angle over 170 degrees) to that of the second eigenvector, and orthogonal (or nearly orthogonal) to the third (or the first) eigenvector, respectively. (d) With the linear activation, $\vec{w}_2$ is only orthogonal to the third eigenvector, but not parallel to either the first or second eigenvector.

nection indicates that the Oja rule has the characteristics similar to those of LMSER. Interestingly, we have discovered that through the sigmoid nonlinearity or some slight linear modification, the LMSER rule, as well as the Oja rule, enables a symmetrically circuited one-layer-net to perform true PCA. Two kinds of modifications have been proposed and the results of theoretical analysis, as well as experimental simulations, have been provided to show how they perform PCA without the external design of an asymmetrical circuit which is necessarily required by the methods of Sanger (1989) and Rubner and Schulten (1990). These results indicate that with slight modifications, the LMSER rule, as well as the Oja rule, should be as capable as Sanger's GHA and other asymmetrically circuited nets (e.g., Rubner & Schulten, 1990) for practical tasks based on PCA. They may also interpret how the development of biological cortical field can break symmetry without any externally "hardwired" asymmetry.

Finally, we would like to also mention that the net proposed in this paper may have a number of potential applications, such as associative memory, feature extraction, data compression, unsupervised pattern clustering and recognition, attentional recognition, and for interpreting the development of orientation cells in the cortical field, as well as the emergence of mental imagery image in brain. Some speculative discussions were recently made (Xu, 1991) on these potential applica-

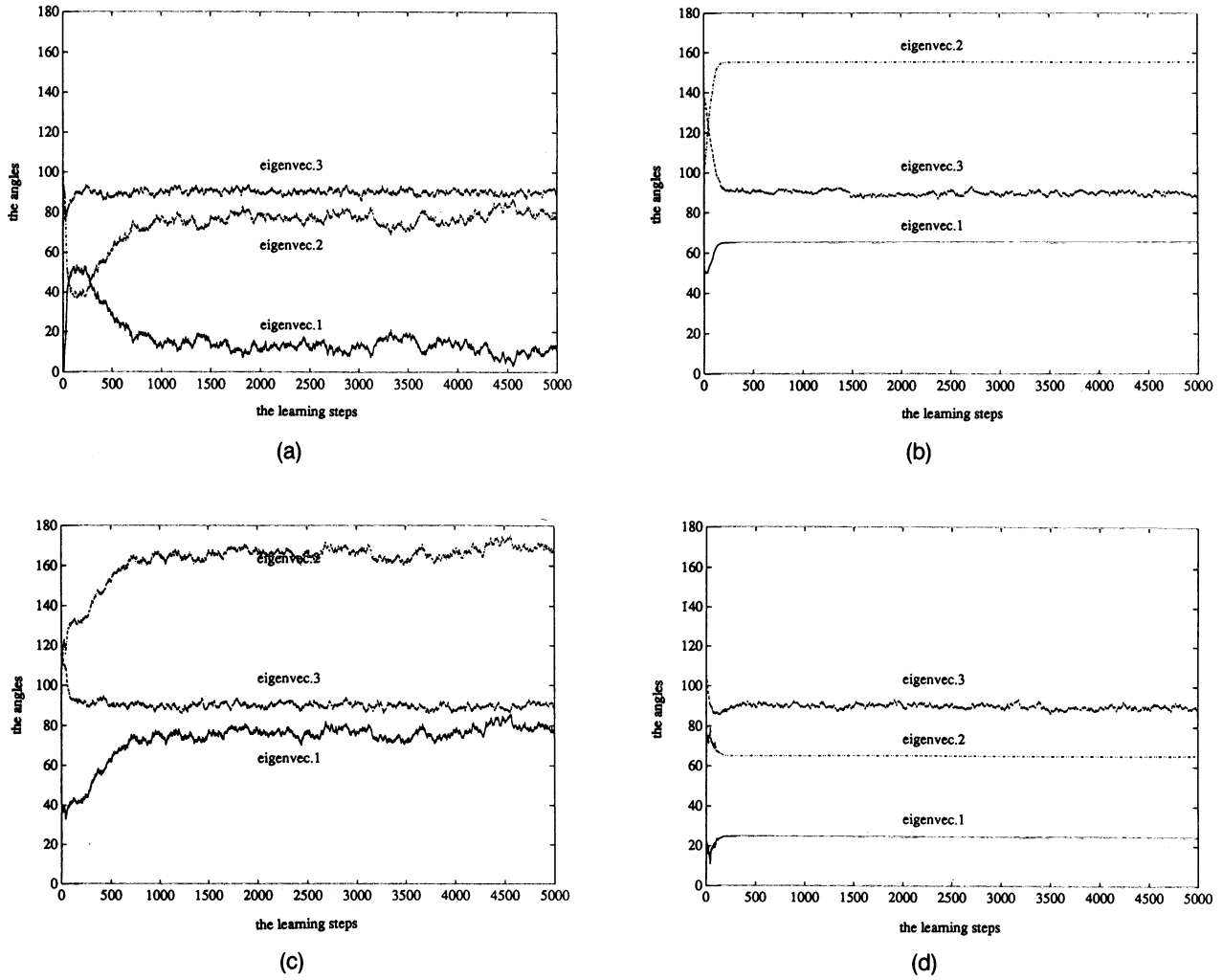


FIGURE 15. All are the same as in Figure 14 except that these results are obtained by the Oja rule. The situations are very similar to those of Figure 14.

tions, as well as on the possibilities of extending the net to the supervised learning net or to a generalized associative memory.

REFERENCES

Ahalt, S. C., Krishnamurty, A. K., Chen, P., & Melton, D. E. (1990). Competitive learning algorithms for vector quantization. *Neural Networks,* **3,** 277–291.

Baldi, P., & Hornik, K. (1989). Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks,* **2,** 52–58.

Baldi, P., & Hornik, K. (1991). Back-propagation and unsupervised learning in linear networks. In Y. Chauvin & D. E. Rumelhart (Eds.), *Back propagation: Theory, architectures and applications.* Hillsdale, NJ: Erlbaum Associates.

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

Barrow, H. G. (1987). Learning receptive fields. *Proceedings of the* 1987 *IEEE First Annual Conference on Neural Networks,* **4,** 115–121.

Becker, S., & Hinton, G. E. (1992). Self-organizing neural network that discover surfaces in random-dot stereogram. *Nature,* **355,** 161–163.

Bienenstock, E. L., Cooper, L. N., & Munro, P. W. (1982). Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex. *Journal of Neuroscience,* **2,** 32–48.

Brockett, R. W. (1991). Dynamical systems that sort lists, diagonalize matrices, and solve linear programming problems. *Linear Algebra and Its Applications,* **146,** 79–91.

Capenter, G. A., & Grossberg, S. (1987a). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing,* **37,** 54–115.

Capenter, G. A., & Grossberg, S. (1987b). ART2: Self-organization of stable category recognition codes fo analog input patterns. *Applied Optics,* **26,** 4919–4930.

Capenter, G. A., & Grossberg, S. (1988, March). The ART of adaptive pattern recognition by a self-organizing neural network. *IEEE Computer,* 77–88.

Capenter, G. A., & Grossberg, S. (1990). ART3: Hierarchical searching using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks,* **3,** 129–152.

Casdagli, M. (1989). Nonlinear prediction of chaotic time series, *Physica,* **35D,** 335–356.

Chauvin, Y. (1989). Principal component analysis by gradient descent on a constrained linear Hebbian cell. *Proceedings of the IEEE International Conference on Neural Networks, Washington D.C.,* **I,** 373–380.

Cohen, M. A., & Grossberg, G. (1983). Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Transactions on Systems, Man and Cybernetics,* **13,** 815–826.

Desieno, D. (1988). Adding a conscience to competitive learning. *Proceedings of the IEEE International Conference on Neural Networks,* **I,** 117–124.

Foldiak, P. (1989). Adaptive network for optimal linear feature extraction. *Proceedings of the IEEE International Conference on Neural Networks, Washington D.C.,* **I,** 401–405.

Fukushima, K. (1975). Cognitron: A self-organizing multilayered neural network. *Biological Cybernetics,* **20,** 121–136.

Fukushima, K. (1980). Neocognitron: A self-organizing neural model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics,* **36,** 193–202.

Grossberg, S. (1969). On learning and energy entropy dependence in recurrent and nonrecurrent signed networks. *Journal of Statistical Physics,* **1,** 319–350.

Grossberg, S. (1972). Neural expectation: Cerebellar and retinal analogs of cells fired by learnable or unlearned pattern classes. *Kybernetik,* **10,** 49–57.

Grossberg, S. (1973). Contour enhancement, short term memory, and constancies in reverberating neural networks. *Studies in Applied Mathematics,* **52,** 217–257.

Grossberg, S. (1976a). Adaptive pattern classification and universal recording: I. Parallel development and coding of neural feature detectors. *Biological Cybernetics,* **23,** 121–134.

Grossberg, S. (1976b). Adaptive pattern classification and universal recording: II. Feedback, expectation, olfaction, illusions. *Biological Cybernetics,* **23,** 187–202.

Grossberg, S. (1987). Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science,* **11,** 23–63.

Hecht-Nielsen, R. (1987). Counterpropagation networks. *Applied Optics,* **26,** 4979–4984.

Hertz, J., Krogh, J., & Palmer, R. G. *Introduction to the theory of neural computation.* Reading, MA: Addison-Wesley.

Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex. *Journal of Physiology (London),* **160,** 106–154.

Hopfield, J. J. (1982). Neural Networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Science, USA,* **79,** 2554–2558.

Hopfield, J. J. (1984). Neurons with graded responses have collective computational abilities like those of two-state neurons. *Proceedings of the National Academy of Science, USA,* **81,** 3088–3092.

Hornik, K., & Kuan, C. M. (1992). Convergence analysis of local feature extraction algorithms. *Neural Networks,* **5**(2), 229–260.

Hrycej, T. (1990). Self-organization by delta rule. *Proceedings of the* 1990 *International Joint, Conference on Neural Networks,* **2,** 307–312.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation,* **3,** 79–87.

Kammen, D. M., & Yuille, A. L. (1988). Spontaneous symmetry-breaking energy functions and the emergence of orientation selective cortical cells. *Biological Cybernetics,* **59,** 23–31.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics,* **43,** 59–69.

Kohonen, T. (1988). *Self-organization and associative memory.* Berlin: Springer-Verlag.

Krogh, A., & Hertz, J. A. (1990). Hebbian learning of principal components. In R. Eckmiller, G. Hartmann & G. Hauske (Eds.), *Parallel processing in neural systems and computers* (pp. 183–186). Amsterdam: Elsevier (North-Holland).

Kung, S. Y. (1990). Constrained principal component analysis via an orthogonal learning network. *Proceedings of the* 1990 *IEEE International Symposium on Circuits and Systems, New Orleans, LA* (pp. 138–140). New York: IEEE Press.

Linsker, E. (1986). From basic network principles to neural architecture. *Proceedings of the National Academy of Science, USA,* **83,** 7508–7512, 8390–8394, 8779–8783.

Linsker, E. (1988, March). Self-organization in a perceptual network. *IEEE Computer,* 105–117.

von der Malsburg, Ch. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik,* **14,** 85–100.

Moody, J., & Darken, C. (1989). Fast learning in networks of locally-tuned processing units. *Neural Computation,* **1,** 281–294.

Nowlan, S. J., & Hinton, G. E. (1990). From competitive learning to adaptive mixtures of experts. *Proceedings of the* 1990 *Neural Information Processing Systems, Denver.*

Oja, E. (1982). A simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology,* **16,** 267–273.

Oja, E. (1989). Neural networks, principal components, and subspaces. *International Journal of Neural Systems,* **1,** 61–68.

Oja, E., Ogawa, H., & Wangviwattana, J. (1991). Learning in nonlinear constrained Hebbian networks. *Proceedings of the International Conference on Artificial Neural Networks, Helsinki* (pp. 385–390). Amsterdam: North-Holland.

Poggio, T., & Girosi, F. (1990). Regularization algorithms for learning that are equivalent to multilayer networks. *Science,* **247,** 978–982.

Rubner, J., & Tavan, P. (1989). A self-organizing network for principal-component analysis. *Europhysics Letters,* **10,** 693–689.

Rubner, J., & Schulten, K. (1990). Development of feature detectors by self-organization. *Biological Cybernetics,* **62,** 193–199.

Rumelhart, D. E., & Zipser, D. (1985). Feature discovery by competitive learning. *Cognitive Science,* **9,** 75–112.

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

Sanger, T. D. (1989). Optimal unsupervised learning in feedforward neural networks (Tech. Rep. No. 1086). Cambridge, MA: MIT AI Lab.

Xu, L. (1991). Least MSE reconstruction for self-organization: (I) Multi-layer neural-nets and (II) further theoretical and experimental studies on one layer nets. *Proceedings of the International Joint Conference on Neural Networks-*1991*-Singapore,* 2363–2373.

Xu, L., Krzyzak, A., & Oja, E. Rival penalized competitive learning for clustering analysis, RBF net and curve detection. *IEEE Trans. on Neural Networks as a Regular Paper.* (in press).

Xu, L., Oja, E., & Suen, C. Y. (1992). Modified Hebbian learning for curve and surface fitting. *Neural Networks,* **5,** 393–407.

Yuille, A. L., Kammen, D. M., & Cohen, D. S. (1989). Quadrature and the development of orientation selective cortical cells by Hebb rules. *Biological Cybernetics,* **61,** 183–194.