Learning in Linear Neural Networks: A Survey

Pierre F. Baldi and Kurt Hornik, *Member, IEEE*

***Abstract*— Networks of linear units are the simplest kind of networks, where the basic questions related to learning, generalization, and self-organization can sometimes be answered analytically. We survey most of the known results on linear networks, including: 1) backpropagation learning and the structure of the error function landscape, 2) the temporal evolution of generalization, and 3) unsupervised learning algorithms and their properties. The connections to classical statistical ideas, such as principal component analysis (PCA), are emphasized as well as several simple but challenging open questions. A few new results are also spread across the paper, including an analysis of the effect of noise on backpropagation networks and a unified view of all unsupervised algorithms.**

I. Introduction

THIS paper addresses the problems of supervised and unsupervised learning in layered networks of linear units and, together with a few new results, reviews most of the recent literature on the subject. One may expect the topic to be fairly restricted, yet it is in fact quite rich and far from being exhausted. Since the first approximations of biological neurons using threshold gates [1], the nonlinear aspects of neural computations and hardware have often been emphasized and linear networks dismissed as uninteresting for being able to express linear input–output maps only. Furthermore, multiple layers of linear units can always be collapsed by multiplying the corresponding weight matrices. So why bother? Nonlinear computations are obviously extremely important, but these arguments should be considered as very suspicious; by stressing the input–output relations only, they miss the subtle problems of dynamics, structure, and organization that normally arise during learning and plasticity, even in simple linear systems. There are other reasons why linear networks deserve careful attention. General results in the nonlinear case are often absent or difficult to derive analytically, whereas the linear case can often be analyzed in mathematical detail. As in the theory of differential equations, the linear setting should be regarded as the first simple case to be studied. More complex situations can often be investigated by linearization, although this has not been attempted systematically in neural networks, for instance in the analysis of backpropagation learning. In backpropagation, learning is often started with zero or small random initial weights and biases. Thus, at least during the initial phase of training, the network is operating in its linear regime. Even when training is completed, one often finds several units in the network which are operating in their linear range. From the standpoint of theoretical biology, it has been argued that certain classes of neurons may be operating most of the time in a linear or quasi-linear regime and linear input–output relations seem to hold for certain specific biological circuits (see [2] for an example). Finally, the study of linear networks leads to new interesting questions, insights, and paradigms which could not have been guessed in advance and to new ways of looking at certain classical statistical techniques.

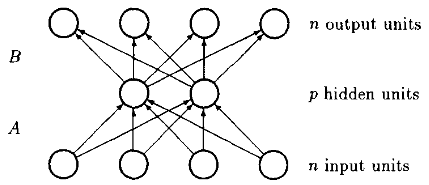


Fig. 1. The basic network in the autoassociative case ($m = n$).

To begin with, we shall consider a linear network with an n–p–m architecture comprising one input layer, one hidden layer, and one output layer with n, p, and m units, respectively (Fig. 1). The more general case, with, for instance, multiple hidden layers, can be reduced to this simple setting as we shall see. A will usually denote the $p \times n$ matrix connecting the input to the middle layer and B the $m \times p$ matrix of connection weights from the middle layer to the output. Thus, for instance, b_{ij} represents the strength of the coupling between the jth hidden unit and the ith output unit (double indexes are always in the post-presynaptic order). The network therefore computes the linear function $y = BAx$. In the usual learning from examples setting, we assume that a set of n-dimensional input patterns x_t $(1 \leq t \leq T)$ is given together with a corresponding set of m-dimensional target output patterns y_t $(1 \leq t \leq T)$ (all vectors are assumed to be column vectors). $X = [x_1, \cdots, x_T]$ and $Y = [y_1, \cdots, y_T]$ are the $n \times T$ and $m \times T$ matrices having the patterns as their columns. Because of the need for target outputs, this form of learning will also be called supervised. For simplicity, unless otherwise stated, all the patterns are assumed to be centered (i.e., $\langle x \rangle = \langle y \rangle = 0$). The symbol "$\langle \cdot \rangle$" will be used for averages over the set of patterns or sometimes over the pattern distribution, depending on the context. The approximation of one by the other is a central problem in statistics, but is not our main concern here. The environment is supposed to be stationary but the results could be extended to a slowly varying environment to deal with plasticity issues. Throughout this paper, learning will often be based on the minimization of an error function E depending

Manuscript received September 25, 1992; revised June 19, 1994. This work was supported in part by grants from NSF, AFOSR, and ONR.

P. F. Baldi is with the Jet Propulsion Laboratory, and Division of Biology, California Institute of Technology, Pasadena, CA 91109 USA.

K. Hornik is with the Institut für Statistik und Wahrscheinlichkeitstheorie, Technische Universität Wien, A-1040 Vienna, Austria.

IEEE Log Number 9409158.

on the synaptic weights. In the main case of backpropagation, the error function is

$$E(A,B) = \langle \|y - BAx\|^2 \rangle \tag{1}$$

where $\|u\|$ represents the Euclidean norm of the vector u. When no target outputs are provided, the learning (which then must be based on criteria to be specified, such as the maximization of the output variance) is unsupervised. An important special case of unsupervised learning is the case of autoassociation, when the input is used as a teacher (i.e., $y_t = x_t$). This is also called autoencoding or identity mapping in the literature.

Learning rules are algorithms for slowly altering the connection weights to achieve a desirable goal such as the minimization of an error function. Often, three different versions of the same rule have been given: the "on-line" version where the modification is calculated after the presentation of each pattern, the "off-line" version where the previous modifications are averaged over the cycle of all patterns, and the "continuous" version where the discrete changes induced by the "off-line" algorithm are approximated continuously by a differential equation governing the evolution of the weights in time. In some cases, the three formulations can be shown to lead to essentially the same results.

It will be convenient to use the notation $\Sigma_{uv} = \langle uv' \rangle$ where the prime denotes transposition of matrices. If both $\langle u \rangle$ and $\langle v \rangle$ are zero, Σ_{uv} is the covariance matrix of u and v. Σ_{xx}, for instance, is a real $n \times n$ symmetric nonnegative definite matrix. Hence, its eigenvalues can be ordered as $\lambda_1 \geq \cdots \geq \lambda_n \geq 0$. For mathematical simplicity, we shall often assume that in fact $\lambda_1 > \cdots > \lambda_n > 0$. This should not be regarded as a very restrictive assumption, since this condition can always be enforced by, at worst, perturbing the data by infinitesimal amounts and attributing these perturbations to "noise." Many conclusions are only slightly different when some eigenvalues coincide.

A good familiarity with linear algebra and basic calculus on the part of the reader should be sufficient to follow the paper. All the statistical techniques required to understand some of the results are briefly reviewed in the second section. These include least-squares regression, principal component analysis, and discriminant analysis. In Section III, we treat the case of supervised learning with backpropagation and the corresponding autoassociative special case. We study the landscape of the error function E of (1), its connections to the previously mentioned statistical techniques, and several consequences and generalizations, including noisy and deep networks. In Section IV, we study the problems of validation, generalization, and overfitting in a simple one-layer network trained to learn the identity map. Under some assumptions, we give a complete description of the evolution of the validation error as a function of training time. Section V covers a variety of unsupervised learning algorithms, based on variance maximization/minimization by Hebbian or anti-Hebbian learning or other error functions. Some of the more technical proofs are deferred to the Appendix.

II. Mathematical Background

A. Optimization of Quadratic Forms over Spheres

Let S be a symmetric $n \times n$ matrix. Then all the eigenvalues λ_i of S are real and can be ordered in the form $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ with corresponding normalized eigenvectors $u_1, \cdots, u_n$. Consider the problem of maximizing the quadratic form $E(a) = a'Sa$ over the sphere of radius ρ and centered at the origin ($\|a\| \leq \rho$). In geometry, it is well known (see, for instance, [3]) that the maximum of E is then reached on the surface of the sphere in the direction of the first eigenvector, that is, at the points $\pm\rho u_1$ where $E(\pm\rho u_1) = \lambda_1 \rho^2$. If $\lambda_1 > \lambda_2$, $\pm\rho u_1$ are the only two solutions. Similarly, the maximum of E over the intersection of the sphere with the linear space orthogonal to u_1 is reached at $\pm\rho u_2$, and so forth. Finally, the minimum of E over the entire sphere is obtained at $\pm\rho u_n$. All these properties are easily derived by decomposing a as $a = \Sigma_i \alpha_i u_i$ and noticing that $E(a) = \sum_i \lambda_i \alpha_i^2$.

B. Singular Value Decomposition

Let Z be an arbitrary $k \times l$ matrix with rank r. Then there exist numbers $\sigma_1 \geq \cdots \geq \sigma_r > 0$, the singular values of Z, an orthogonal $k \times k$ matrix U, and an orthogonal $l \times l$ matrix V such that $S = U'ZV$ is a $k \times l$ diagonal matrix of the form

$$S = \begin{bmatrix} D & O \\ O & O \end{bmatrix}$$

where $D = \mathrm{diag}(\sigma_1, \cdots, \sigma_r)$ is the diagonal matrix with entries $\sigma_1, \cdots, \sigma_r$. The decomposition

$$Z = USV'$$

is called the singular value decomposition (SVD) of Z (it is not necessarily unique). The matrices U and V in the SVD have the following meaning. As $Z'ZV = VS'U'USV'V = VS'S = V\mathrm{diag}(\sigma_1^2, \cdots, \sigma_r^2, 0, \cdots, 0)$, the columns of V are unit-length, mutually perpendicular eigenvectors of $Z'Z$, and $\sigma_1^2, \cdots, \sigma_r^2$ are the nonzero eigenvalues of $Z'Z$. Similarly, the columns of U are unit-length, mutually perpendicular eigenvectors of ZZ'. With the aid of the SVD, the pseudoinverse of Z can easily be given explicitly. If we write

$$S^+ = \begin{bmatrix} D^{-1} & O \\ O & O \end{bmatrix}$$

then $Z^+ = VS^+U'$ is the pseudoinverse of Z (see, for instance, [4] and [5] for more details on SVD and pseudoinverses).

C. Orthogonal Projections

If $\mathcal{L}$ is a linear subspace, we denote by $P_{\mathcal{L}}x$ the orthogonal projection of a vector x onto $\mathcal{L}$ and by $P_{\mathcal{L}^\perp}x = Q_{\mathcal{L}}x = x - P_{\mathcal{L}}x$ its projection onto the orthogonal complement of $\mathcal{L}$. If $\mathcal{L}$ has dimension l and is spanned by the linearly independent vectors $z_1, \cdots, z_l$, then $P_{\mathcal{L}}x = P_Z x$, where $Z = [z_1, \cdots, z_l]$ and $P_Z = Z(Z'Z)^{-1}Z'$. In particular, if the vectors z_i are mutually perpendicular unit vectors, the projection of x simply is $ZZ'x = z_1 z_1' x + \cdots + z_l z_l' x$. If the matrix Z is not full rank, P_Z can still be written as

$$P_Z = ZZ^+ = u_1 u_1' + \cdots + u_r u_r'$$

where $u_1, \cdots, u_r$ are the first r columns of the U matrix in the SVD of Z (and r is the rank of Z).

Consider now the problem of finding a vector w which minimizes $F(w) = \|c - Mw\|^2$ for a given vector c and a matrix M. In other words, we are looking for the vector in the image space of M (the space spanned by the columns of M) which is closest to c. Clearly, by the projection theorem, this is the orthogonal projection of c onto the image of M. In particular, if M is of full rank, then at the optimum w we must have $M(M'M)^{-1}M'c = Mw$ or, equivalently, $w = (M'M)^{-1}M'c$. The Hessian of F is $2M'M$; hence, if M is full rank, the Hessian is positive definite and the problem is strictly convex without any other critical points.

D. Least-Squares Regression

The problem of linear regression is the following. Given a set of n-dimensional input vectors $x_1, \cdots, x_T$ and a set of m-dimensional target vectors $y_1, \cdots, y_T$, find an $m \times n$ matrix A which minimizes $E(A) = \langle \|y - Ax\|^2 \rangle$. In other words, linear regression is exactly the usual learning problem in a linear network without any hidden units. Since the output units are completely uncoupled, the connection weights for each of them can be synthesized separately and therefore one needs only to consider the case $m = 1$, where we write $A = a'$. In this case, the problem has a simple geometrical interpretation: find a hyperplane through the origin in $(n+1)$-dimensional space which best fits (in the least-squares sense) a cloud of T points with coordinates $(x_1', y_1)', \cdots, (x_T', y_T)'$. Now

$$E(a) = \langle (y - a'x)^2 \rangle = a'\Sigma_{xx}a - 2\Sigma_{yx}a + \langle y^2 \rangle$$

and the gradient of E with respect to a is

$$\nabla E = 2\Sigma_{xx}a - 2\Sigma_{xy}.$$

E is continuous, differentiable, and bounded below by zero and therefore it must reach its minimum for a vector a satisfying $\Sigma_{xx}a = \Sigma_{xy}$. If Σ_{xx} is positive definite, then there is a unique solution given by

$$a = \Sigma_{xx}^{-1}\Sigma_{xy} \qquad (2)$$

and, in addition, E is strictly convex (with Hessian $2\Sigma_{xx}$) and so without any local minima (or even without any other critical point). The landscape is therefore as simple as possible, and this remains true even if some of the connections are forced in advance to take some fixed values, typically zero in the case of "local" connectivity (this introduces linear, thus convex, restrictions on the set of possible weights). When $m > 1$, everything goes through mutatis mutandis. In the case where Σ_{xx} is positive definite, the unique optimal A is called the slope matrix of the regression of y on x and is given by

$$A = \Sigma_{yx}\Sigma_{xx}^{-1}$$

which generalizes (2), taking into account that $A = a'$ in one dimension. (Formally, to reduce the m-dimensional case it is sufficient to notice that E can be rewritten as $E(A) = \|\text{vec}(Y) - (X' \otimes I)\text{vec}(A)\|^2/T$, where $\otimes$ denotes the Kronecker product of two matrices and "vec" is an operator which transforms a matrix into a vector by stacking its columns one above the other. See [6] for details.) In particular, even if the connectivity between the input and the output is local, the problem remains convex and without local minima and therefore, in principle, easy to learn by a gradient-descent type of mechanism. Finally, notice that if for an input x_t we approximate the corresponding output pattern y_t by its linear estimate $\hat{y}_t = \Sigma_{yx}\Sigma_{xx}^{-1}x_t$, then the covariance matrix of the estimates is given by $\Sigma = \langle \hat{y}\hat{y}' \rangle = \Sigma_{yx}\Sigma_{xx}^{-1}\Sigma_{xy}$.

E. Principal Comonent Analysis

Suppose we are given a collection of T objects. For each object x_t, the measurements of the same n characteristics $x_{1,t}, \cdots, x_{n,t}$ are available. Assume it is desired to extract some "structure" or "main features" from this collection of data. For efficient classification, it is obviously useful to compress the high-dimensional input data into something low dimensional without discarding too much relevant information. Of course, there are several different techniques for feature extraction and data compression. One of the simplest and most general-purpose ones is a statistical method known as principal component analysis (PCA).

By possibly subtracting the average $\langle x \rangle$, we can think of the data set $x_{i,t} (1 \le i \le n, 1 \le t \le T)$ as a cloud of T points in n-dimensional Euclidean space centered around the origin. To capture the main features of the data set, PCA is looking for directions along which the dispersion or variance of the point cloud is maximal, that is, looking for a subspace $\mathcal{L}$ such that the projection of the points x_t onto $\mathcal{L}$ has maximal variance. If $\mathcal{L}$ is the line spanned by the unit vector a, the projection $P_{\mathcal{L}}x$ is given by $P_a x = aa'x$ with squared length $\|P_a x\|^2 = (a'x)^2 = a'xx'a$. Hence, the average dispersion of the data set in the direction of the line is $\langle \|P_a x\|^2 \rangle = \langle a'xx'a \rangle = a'\langle xx' \rangle a = a'\Sigma_{xx}a$, where $\Sigma_{xx} = \langle xx' \rangle$ is the data covariance matrix. PCA looks for a unit vector a^* which maximizes $a'\Sigma_{xx}a$ over the set of all unit vectors. If $\lambda_1 > \cdots > \lambda_n > 0$ are the eigenvalues of Σ_{xx} with eigenvectors $u_1, \cdots, u_n$, then, by the previous result on quadratic forms in Section II-A, we know that $a^* = u_1$ (or equivalently $-u_1$) is the answer.

To sum up, PCA starts by finding the direction in which the dispersion of the cloud is maximal, which is the direction u_1 of the first eigenvector of the data covariance matrix. The first "feature" which is extracted is the first principal component $u_1'x_t$. The component of the data "explained" by the first principal component is the projection onto the line spanned by u_1. What remains unexplained is the dispersion of the residual $x_t - P_{u_1}x_t$ which is just the projection $Q_{u_1}x_t$ of x_t onto the orthogonal complement of u_1. In a second step, we proceed as before, but with the points x_t replaced by $Q_{u_1}x_t$. That is we look for straight lines $\mathcal{L}$ perpendicular to the line spanned by u_1 such that the projections of the points $Q_{u_1}x_t$ have maximal variance. This amounts to finding a unit vector b^*, perpendicular to u_1, which maximizes $b'\Sigma_{xx}b$ over all unit vectors perpendicular to u_1. Again, by the previous result, we know the answer is $b^* = \pm u_2$, and so forth. At the kth step, we look for lines $\mathcal{L}_k$ perpendicular to the space spanned by $u_1, \cdots, u_{k-1}$ such that the projections of the points x_t along $\mathcal{L}_k$ have maximal dispersion. This is achieved by choosing $\mathcal{L}_k$ as the line spanned by u_k.

After the completion of p steps, we extract the first p principal components $u_1'x_t, \cdots, u_p'x_t$ and reduce x_t to its projection onto the hyperplane spanned by the first p eigenvectors. One may be interested in asking whether this is the best possible data reduction of the kind under consideration, that is, the best possible projection of the data onto a p-dimensional hyperplane $\mathcal{H}$ in the sense that the projections of the data onto $\mathcal{H}$ have maximal variance. After all, a better result might have been achieved by choosing the hyperplane in a single step. This, however, is not the case.

Among all p-dimensional hyperplanes $\mathcal{H}$, the one spanned by the first p principal vectors $u_1, \cdots, u_p$ is the hyperplane such that $\langle \|P_{\mathcal{H}}x\|^2 \rangle$ is maximal. Equivalently, it is the hyperplane $\mathcal{H}$ which minimizes the average projection error $\langle \|x - P_{\mathcal{H}}x\|^2 \rangle$.

It is therefore possible to incrementally build the PCA feature extractor. Since $\mathcal{H}$ is the best p-dimensional hyperplane we can fit to the n-dimensional point cloud, the "flatter" the cloud the better the fit. It is worth investigating how good the fit is, that is, how much of the variance in the data set actually is explained by the first p principal components. This is easily computed, for the variance of the ith component is given by

$$\langle \|P_{u_i}x\|^2 \rangle = \langle (u_i'x)^2 \rangle = u_i'\Sigma_{xx}u_i = u_i'\lambda_i u_i = \lambda_i.$$

The total variance being equal to the sum of all the eigenvalues of Σ_{xx}, the proportion of total variance explained by the first p principal components equals $(\lambda_1 + \cdots + \lambda_p)/(\lambda_1 + \cdots + \lambda_n)$.

In fact, PCA performs "best data compression" among a wider class of methods. Let us write $U_p = [u_1, \cdots, u_p]$ for the matrix having the first p normalized eigenvectors of Σ_{xx} as its columns and let us stack the first p features $u_1'x_t, \cdots, u_p'x_t$ extracted by PCA into a column vector z_t. Then $z_t = U_p'x_t$ and $P_{U_p}x_t = U_pU_p'x_t = U_pz_t$. Hence, PCA is one method that linearly compresses n-dimensional inputs x_t into p-dimensional vectors z_t for some $p < n$, that is, $z = Ax$ for a suitable $p \times n$ matrix A. Linear reconstruction of the data can then be achieved by approximating x_t by $Bz_t = BAx_t$ for some suitable $n \times p$ matrix B.

Among all $p \times n$ matrices A and $n \times p$ matrices B, optimal linear data compression (in the sense that the average reconstruction error $\langle \|x - BAx\|^2 \rangle$ is minimized) is achieved if and only if the global map $W = BA$ equals the orthogonal projection P_{U_p} onto the hyperplane spanned by the first p eigenvectors of Σ_{xx}.

Finally, computing the covariance of two principal components gives that for $i \neq j$

$$\langle (u_i'x)(u_j'x) \rangle = \langle u_i'xx'u_j \rangle = u_i'\langle xx' \rangle u_j = u_i'\lambda_j u_j = 0.$$

Thus different components are uncorrelated, and we can think of the transformation of x_t into the vector of n principal components $[u_1'x_t, \cdots, u_n'x_t]'$ as an orthogonal transformation of the Euclidean space, such that in the new system of coordinates, the components of the points in the cloud are uncorrelated and with decreasing variance. Again, if only the first few coordinates in the new system vary significantly, we may approximately locate points by giving only these few coordinates.

PCA can also be examined from an information-theoretic standpoint and shown to be optimal, under simple assumptions, for a different measure. More precisely, consider a transmission channel (in our case, one can think of the network connecting the input units to the hidden units) with n-dimensional centered input vectors having a Gaussian distribution with covariance matrix $\Sigma_{xx} = \langle xx' \rangle$. The outputs of the channel are constrained to be p-dimensional vectors of the form $y = Lx$, for some $p \times n$ matrix L (and, without any loss of generality, we can assume that L has rank $p, p < n$. Hence, y is also Gaussian with covariance matrix $L\Sigma_{xx}L'$. Classically, the differential entropy of x is given by (see, for instance, [7] for more details)

$$H(x) = -\int p(x) \log p(x)\, dx = \tfrac{1}{2} \log [(2\pi e)^n \det(\Sigma_{xx})]$$

where $p(x)$ is the Gaussian density function, and similarly

$$H(y) = \tfrac{1}{2} \log [(2\pi e)^p \det(L\Sigma_{xx}L')].$$

The conditional distribution of x given y (see, for instance, [8]) is normal with mean

$$\mu_{x.y} = \Sigma_{xy}\Sigma_{yy}^{-1}y$$

and covariance matrix

$$\Sigma_{xx.y} = \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}.$$

It can be shown that $H(x|y)$, the conditional entropy of x given y (i.e., the entropy of the conditional distribution) is given by

$$H(x|y) = \tfrac{1}{2} \log ((2\pi e)^{n-p}\gamma_1 \cdots \gamma_{n-p})$$

where $\gamma_1 \geq \cdots \geq \gamma_{n-p} > 0$ are the nonzero eigenvalues of $\Sigma_{xx.y}$. As the entropy is one way of measuring our uncertainty, it is desirable to choose L so as to minimize $H(x|y)$. One can show that the optimal L is of the form $L = CU_p'$ where C is an invertible $p \times p$ matrix and $U_p = [u_1, \cdots, u_p]$. In particular, this choice also maximizes the information that y conveys about x measured by the mutual information $I(x, y)$ defined to be

$$I(x, y) = H(x) - H(x|y)$$

with value

$$I_{\text{PCA}}(x, y) = \tfrac{1}{2} \log ((2\pi e)^p \lambda_1 \cdots \lambda_p).$$

Thus, at least in the Gaussian setting, up to trivial transformations, the optimal linear map maximizing the mutual information is the principal component analyzer. Finally, PCA can also be connected to optimal inference methods (see [9]).

To illustrate the PCA feature extraction technique, consider the "open/closed book" data set in Mardia *et al.* [10, Table 1.2.1, p. 3f]. The data consist of the scores of $T = 88$ students on examinations in mechanics, vectors, algebra, analysis, and statistics (i.e., $n = 5$, where the first two exams were closed book and the other three were open book). For each exam, the best possible score was 100. It is found that the average score $\langle x \rangle$ equals (39.0, 50.6, 50.6, 46.7, 42.3)$'$ and that the eigenvalues of the covariance matrix Σ_{xx} are given by $\lambda_1 = 679.2$, $\lambda_2 = 199.8$, $\lambda_3 = 102.6$, $\lambda_4 = 83.7$ and $\lambda_5 = 31.8$. Hence, the first two principal components already explain 80% of the variance in the data (and 91% is achieved with the first three). The first two eigenvectors are $u_1 = (0.51, 0.37, 0.35, 0.45, 20.53)'$ and $u_2 = (0.75, 0.21, -0.08, -0.30, -0.55)'$. These findings can easily be interpreted. The authors conclude that "...the first principal component gives positive weight to all the variables and thus represents an average grade. On the other hand, the second principal component represents a contrast between the open-book and closed-book examinations..." For example, the scores and first two principal components of the two best students are (77, 82, 67, 67, 81), 66.4, and 6.4 and (63, 78, 80, 70, 81), 63.7, and -6.4. Even without looking at the individual test scores, by considering only the first two principal components one would conclude that the overall performances of the two students are very similar, but the first student did better on closed book and the second one better on open-book exams.

In conclusion, PCA is optimal in the least-mean-square sense and can serve two purposes: data compression by projecting high-dimensional data into a lower-dimensional space and feature extraction by revealing, through the principal components, relevant but unexpected structure hidden in the data (although an interpretation of these features in terms of the original variables may not always be straightforward).

F. Mean Square Classifier and Discriminant Analysis

Consider now the problem where the patterns x_t must be classified into m classes $C_1, \cdots, C_m$, with, in general, $m \ll n$. Thus for every input pattern x_t, there is a binary target output pattern $y_t = (0, \cdots, 1, \cdots, 0)'$ where $y_{i,t} = 1$ if and only if x_t belongs to C_i. One possible classification method consists in finding an $m \times n$ matrix L such that $\langle \|y - Lx\|^2 \rangle$ is minimal. Needless to say, this is a special case of least-squares regression, and, as we have seen, under the usual assumptions the optimal L is given by $L = \Sigma_{yx}\Sigma_{xx}^{-1}$ and is called the mean-square classifier.

In many applications n is very large compared to m, and therefore it becomes useful to first reduce the dimensionality of the input data. One is thus led to find a linear subspace of dimension p such that, when projected onto this subspace, the patterns x_t fall as much as possible into well-defined separated clusters facilitating the classification. This problem of finding an optimal projection is similar to the one encountered in PCA. Because of the clustering, however, a new measure must be introduced to compare different projections. Consider a projection $z = C'x$, where C is an $n \times p$ matrix. The total dispersion (variation) in the x-sample can be decomposed into the sum of within-class dispersions and between-class dispersions. When the x's are centered, the total dispersion is Σ_{xx}, and the dispersion between classes can be shown to be $\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}$. Upon projecting the patterns, the corresponding total and between classes dispersions of the z_t patterns become $C'\Sigma_{xx}C$ and $C'\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}C$. A projection is optimal if the between classes variation of the z's is as large as possible relative to the total variation. Different cost functions can be introduced at this stage. If the size of a variation matrix is measured by its determinant (the determinant of a matrix measures the volume of the image of a unit cube under the corresponding linear map), then we are led to the problem of finding an $n \times p$ matrix C maximizing the ratio

$$E(C) = \frac{\det(C'\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}C)}{\det(C'\Sigma_{xx}C)}. \tag{3}$$

The solution is well known.

All optimal matrices, the so-called discriminant analysis (DA) matrices, are of the form H_pR, where R is an arbitrary $p \times p$ invertible matrix and H_p has the first p eigenvectors of $\Sigma_{xx}^{-1}\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}$ as its columns.

It is not easy to see what the solutions look like in general. There is one case, however, where all optimal solutions can easily be described.

When $p = r = \text{rank}(\Sigma_{xy})$, an $n \times p$ matrix C is a DA matrix if and only if the space spanned by the columns of C coincides with the space spanned by the rows of the mean-square classifier $\Sigma_{yx}\Sigma_{xx}^{-1}$.

See, for instance, Kshirsager [8] for more details on DA.

III. Backpropagation

A. The Landscape Properties of E

We now consider the setting described in the Introduction where the learning procedure is based on the minimization of the cost function $E(A, B)$. A complete description of the landscape properties of E is given in Baldi and Hornik [6]. We shall briefly review the most salient features. E is best described in terms of its critical points, that is, the points where $\partial E/\partial a_{ij} = \partial E/\partial b_{ij} = 0$. It is first important to observe that if C is any $p \times p$ invertible matrix, then $E(A, B) = E(CA, BC^{-1})$. Therefore, at any point E really depends on the global map $W = BA$ rather than on A and B. For instance, there is an infinite family of pairs of matrices (A, B) corresponding to any critical point. Unlike the simple case of linear regression, however, W cannot be chosen arbitrarily: the network architecture constrains W to have at most rank p.

The remarkable property of the landscape of E is the absence of local minima in spite of the fact that E is not convex (nor is the set of all matrices of rank at most p). E is characterized by a unique global minimum (up to multiplication by a matrix C). All other critical points are saddle points. The structure of the critical points can be described completely. More precisely, assume for simplicity that $p \leq m \leq n$ and that $\Sigma = \Sigma_{yx}\Sigma_{xx}^{-1}\Sigma_{xy}$, the covariance matrix of the linear estimates $\hat{y}_t$ (see Section II-D), is full rank with m distinct eigenvalues $\lambda_1 > \cdots > \lambda_m$ and corresponding orthonormal eigenvectors $u_1, \cdots, u_m$. If $\mathcal{I} = \{i_1, \cdots, i_p\}$ with $1 \leq i_1 < \cdots < i_p \leq m$ is any ordered p-index set, let $U_{\mathcal{I}}$ denote the matrix $[u_{i_1}, \cdots, u_{i_p}]$. Then two full-rank matrices

A and B define a critical point of E if and only if there exist an ordered p-index set $\mathcal{I}$ and an invertible $p \times p$ matrix C such that

$$A = CU'_{\mathcal{I}}\Sigma_{yx}\Sigma_{xx}^{-1}, \tag{4}$$

$$B = U_{\mathcal{I}}C^{-1}. \tag{5}$$

For such a critical point we have

$$W = P_{U_{\mathcal{I}}}\Sigma_{yx}\Sigma_{xx}^{-1} \tag{6}$$

and

$$E(A, B) = \mathrm{trace}(\Sigma_{yy}) - \sum_{i \in \mathcal{I}} \lambda_i.$$

Therefore, a critical W of rank p is always the product of the ordinary least-squares regression matrix followed by an orthogonal projection onto the subspace spanned by p eigenvectors of Σ. The critical map W associated with the index set $\{1, \cdots, p\}$ is the unique local and global minimum of E. The remaining $\binom{m}{p} - 1$ p-index sets correspond to saddle points. All additional critical points defined by matrices A and B which are not of full rank are also saddle points and can be characterized in terms of orthogonal projections onto subspaces spanned by q eigenvectors, with $q < p$.

In the autoassociative case, (4)–(6) become

$$A = CU'_{\mathcal{I}} \tag{7}$$

$$B = U_{\mathcal{I}}C^{-1} \tag{8}$$

$$W = P_{U_{\mathcal{I}}} \tag{9}$$

and therefore the unique locally and globally optimal map W is the orthogonal projection onto the space spanned by the first p eigenvectors of Σ_{xx}.

This analysis links backpropagation in linear networks to several classical statistical techniques. In particular, at the global minimum of E, if $C = I_p$ then the activities in the hidden layer are given by $u'_1\hat{y}_t, \cdots, u'_p\hat{y}_t$, the principal components of the least-squares estimators $\hat{y}_t$ (see, for instance, [8]). In the autoassociative mode, these activities are given by $u'_1x_t, \cdots, u'_px_t$, and correspond to the coordinates of the vector x_t along the first p eigenvectors of Σ_{xx} as in the usual PCA. In general, if the initial conditions are random, one should not expect the backpropagation algorithm to converge to an optimum satisfying $C = I_p$. In the autoassociative case, this means that the rows of the final A and $u_1, \cdots, u_p$ will span the same space but $A' \neq [u_1, \cdots, u_p]$. Although at first sight this may seem a drawback, it must be regarded as a property leading to more robust networks. Indeed, in a physical implementation where the compressed version of the data in the hidden layer is to be sent to further processing layers, it may not be desirable that one of the units, extracting the principal component, has a variance much larger than the other units (it is known, for instance, that in the case of random symmetric matrixes, $\lambda_2 \ll \lambda_1$ almost always; see [11]). A more balanced strategy, where all the variances in the hidden layer are comparable, is by far preferable and is commonly observed in simulations.

Since the optimal solution can be expressed analytically, it can also be obtained effectively with numerical analysis techniques without resorting to any descent procedure. As pointed out in the Introduction, however, this is not the most relevant point of view here where the emphasis is on the learning behavior and emergent organizational principles of simple adaptive networks.

One of the central issues in learning from examples is the problem of generalization, that is, how does the network perform when exposed to a pattern never seen previously? In this setting, a precise quantitative answer can be given to this question. For instance, in the autoassociative case, the distortion of a new pattern is given by its distance to the subspace generated by the first p eigenvectors of Σ_{xx}.

In the special case where $\mathrm{rank}(\Sigma_{xy}) = r = p$, Gallinari *et al.* [12] have shown that if an $n - p - m$ architecture is trained to classify n-dimensional inputs into m $(m < n)$ classes, then the corresponding network performs discriminant analysis in the sense that, for an optimal $W = BA, A'$ is a DA matrix. In other words, under these assumptions, the projection realized by A' maximizes the ratio given in (3). In this context, however, either $p = r = m$, in which case the architecture is $n - m - m$ and there is no bottleneck, or $r < m$ and then full classification into m categories is not supported by the available data and there is no proper data compression (only filtering out of linear dependencies). In any case, all the network ever learns is to be a mean-square classifier, and this can be achieved without any hidden layer.

B. Deep Networks, Local Connectivity, Nonlinearities, and Bias

In Baldi [13], the case of deep networks with multiple hidden layers is briefly examined. It is easy to see that, in this case, the main constraint on the network comes from its bottleneck, that is, from the hidden layer with smallest size p (clearly, p could be attained in more than one hidden layer). Although the expression for the critical points may now become more involved, the main features of the landscape are unchanged: a multiplicity of saddle points, an absence of local minima, and a unique optimal input/output map satisfying (6) with $\mathcal{I} = \{1, \cdots, p\}$.

The bottleneck layer imposes a rank restriction on the map computed by the network. Additional important constraints can be introduced on the geometry of the connections. Often connections are assumed to be local, in the sense that a unit in one layer receives projections only from a restricted subset of elements in the previous layer, for instance according to a Gaussian distribution. These geometrical constraints play an essential role in self-organizing maps and in several models of "linear" cortical development; see for instance Linsker [14]–[16] and Miller *et al.* [17]. These topics deserve separate treatment and will not be addressed here. As mentioned in the previous section, however, in the case of a locally connected linear network without any hidden layer the landscape of the usual quadratic error is again completely devoid of local minima. Learning by descent methods should then be efficient. The landscape properties of the LMS (least mean square) error function of a linear locally connected multilayer network have not been carefully studied yet, and the previous results only give lower bounds. In particular, the question whether the error function has any local minimum remains open despite its disarming simplicity.

In the case of nonlinear units, few analytical results are known, but certainly local minima do appear. An important remark, however, has been made by Bourlard and Kamp [18]. In the autoassociative mode, it is natural to use linear units in the output layer. Under these conditions, nothing is to be gained by using nonlinear elements in the hidden layer. This is basically because the network is trying to approximate a linear map: the identity function. This result can be extended to any linear map. That is, if the set of pairs (x_t, y_t) of examples is such that $y_t = F(x_t)$ for every t with linear F, then nonlinear units in the hidden layer can lead to an approximation of F which is at best equivalent to the approximation obtainable by using linear units exclusively. Reports of simulations in the literature confirm this point and sometimes seem to indicate that the solution found using nonlinear elements is "close" to PCA (Cottrell *et al.* [19]).

Finally, if it is not desirable to assume the existence of a preprocessing stage where the data are centered, then the theory can easily be extended to the case of linear units with bias (see, for instance, [18] and [20] for more details).

C. Noise Analysis

How robust are the previous results against the effects of noise? Different sorts of noise can be introduced, for instance at the level of the synaptic weights or of the activation functions. To fix the ideas, assume in our case that the activation functions in both the hidden layer and the output layer are "noisy." Hence for an input x, the output of the hidden layer is $w = Ax + n$ and the activity in the output units is $z = Bw + e = BAx + Bn + e$. Assume that the noise terms n and e have mean zero, covariance matrices Σ_{nn} and Σ_{ee}, and that they are uncorrelated with each other and with the patterns x and y. It is also reasonable to assume for simplicity that Σ_{nn} is full rank. We are now interested in the problem of minimizing

$$\begin{aligned}\tilde{E}(A,B) &= \langle \|y - (BAx + Bn + e)\|^2 \rangle \\ &= E(A,B) + \mathrm{trace}(B\Sigma_{nn}B') + \mathrm{trace}(\Sigma_{ee}). \quad (10)\end{aligned}$$

Observe that the term $\mathrm{trace}(\Sigma_{ee})$ is just an additive constant and has no influence on the variation of $\tilde{E}$ with A and B. For any positive μ, $\tilde{E}(\mu A, B/\mu) - \tilde{E}(A,B) = \mathrm{trace}(B\Sigma_{nn}B')(1-\mu^2)/\mu^2$. Thus, if $B \neq 0$ and $\mu > 1$, then $\tilde{E}(\mu A, B/\mu) < \tilde{E}(A,B)$. As a result, without any additional constraints, there is no pair (A, B) which minimizes $\tilde{E}$. This is intuitively clear, as the network will try to make A as large as possible and/or B as small as possible so that the signal dominates the noise. It is therefore necessary to restrict the power of the signal Ax. One way of accomplishing this is to introduce "soft constraints" by adding penalty terms like $\langle \|Ax\|^2 \rangle$ or $\mathrm{trace}(AA')$ to $\tilde{E}$. Some results in this direction have been obtained in Plumbley [21].

The other possibility, which we shall consider here in more detail, is to explicitly restrict A to some compact subset $\mathcal{A}$ of the set of all $p \times n$ matrices, for instance, a sphere centered at zero (the case of "hard constraints"). This leads to the problem of minimizing (10) with $A \in \mathcal{A}$ and B arbitrary, which clearly has a well-defined solution. An optimal A must lie on the boundary $\partial\mathcal{A}$ of $\mathcal{A}$ (if not, we could find a $\mu > 1$ such that $\mu A \in \mathcal{A}$).

Let us write $\Sigma_{nn} = \sigma R$, where $\sigma > 0$ measures the noise level and R is some structure matrix (the simplest case is $R = I$, but if the units are physically close it may be unnatural to assume that the individual components of the noise are uncorrelated). The explicit dependence of $\tilde{E}$ on σ can be taken into account by writing

$$\begin{aligned}\tilde{E}(A,B) &= \tilde{E}_\sigma(A,B) \\ &= E(A,B) + \sigma\,\mathrm{trace}(BRB') + \mathrm{trace}(\Sigma_{ee}). \quad (11)\end{aligned}$$

As soon as $\sigma \leq 1$ (for example), it is straightforward to see that the solutions of the problem of minimizing $\tilde{E}$ with $A \in \mathcal{A}$ are identical to the solutions of minimizing $\tilde{E}$ with $A \in \mathcal{A}$ and $B \in \mathcal{B}$, where $\mathcal{B}$ is some fixed compact set independent of σ. By (11), as $\sigma \to 0$, $\tilde{E}(A,B)$ converges uniformly to $E(A,B) + \mathrm{trace}(\Sigma_{ee})$ over the compact set $\mathcal{A} \times \mathcal{B}$. Since these two functions differ only by an additive constant, the solutions of the noisy constrained problem approach the set of all pairs of matrices (A,B) satisfying (4) and (5) with in addition $A \in \mathcal{A}$ (this automatically forces B to be in $\mathcal{B}$, by restricting the matrices C). In other words, if $\mathcal{A}_\sigma$ is the set of all matrices $A \in \mathcal{A}$ which are optimal for noise level σ, then

$$\lim_{\sigma \to 0} \mathcal{A}_\sigma = \{A \in \partial\mathcal{A} : A = CU_p'\Sigma_{yx}\Sigma_{xx}^{-1} \text{ with invertible } C\}$$

(provided of course that the latter set is nonempty). That is, as is intuitively clear, if σ is very small, the solutions of the constrained noisy problem are essentially the same as the solutions of the nonnoisy problem.

In the Appendix we show that as $\sigma \to \infty$

$$\begin{aligned}\min_B \tilde{E}_\sigma(A,B) = {} & \mathrm{trace}(\Sigma_{yy} + \Sigma_{ee}) \\ & - \sigma^{-1}\mathrm{trace}(MA'R^{-1}A) + O(\sigma^{-2})\end{aligned}$$

uniformly over $\mathcal{A}$, where $M = \Sigma_{xy}\Sigma_{yx}$. Hence, if σ is very large, minimizing $\tilde{E}_\sigma$ over $\mathcal{A}$ is essentially equivalent to maximizing $\Phi(A) = \mathrm{trace}(MA'R^{-1}A)$ over $\mathcal{A}$. The solution set $\mathcal{A}_\Phi$ to this "asymptotic problem" depends significantly on the choice of the constraint set $\mathcal{A}$. For instance, if $\mathcal{A} = \{A : \|A\|_F^2 = \mathrm{trace}(AA') \leq \rho\}$($\|A\|_F$) is the so-called Frobenius norm of A), then $\mathcal{A}_\Phi$ consists of all A of the form $\sqrt{\rho}vu'$, where u and v are normalized principal eigenvectors of M and R^{-1}, respectively. On the other hand, if $\mathcal{A} = \{A : AA' = I_p\}$ (i.e., the rows of A are orthonormal) and $R = I$, then the rows of the optimal A span the space of the first p principal eigenvectors of M (for details, see the Appendix). Now notice that $AA' = I_p$ implies $\mathrm{trace}(AA') = p$. Hence in the high-noise case, full PCA of M is inferior to extraction of the first principal component of M only. Or in other words, it is better not to force the rows of A to orthonormality, but allow them to cooperate (build "balanced" representations) instead. In this sense, if σ is very large and $\mathcal{A}$ is "rich" enough, the solutions of the constrained noisy problem are of maximum redundancy where all the hidden units try to do the same thing.

Significantly refined results for the autoassociative case (where $M = \Sigma_{xx}^2$) have been given in Diamantaras and Hornik [22]. They show that for arbitrary invertible Σ_{nn} and orthogonally right-invariant $\mathcal{A}$ (i.e., $AY \in \mathcal{A}$ if $A \in \mathcal{A}$ and Y is orthogonal), $\tilde{E}$ is minimized for matrices A of the form CU' for suitable C (as usual, the columns of U are the eigenvectors of Σ_{xx}). Under the constraint $AA' = I_p$, the minima are attained at $A = \sum_{i=1}^p v_i u_i'$, where $v_1, \cdots, v_p$ are normalized eigenvectors of R^{-1} with the corresponding eigenvalues arranged in decreasing order. Under the Frobenius norm constraint $\mathrm{trace}(AA') \leq \rho$, the minima are attained at A of the form $\sum_{i=1}^r \sqrt{\rho}\gamma_i v_i u_i'$, where the γ_i and the rank r depend on the eigenvalues of Σ_{xx} and Σ_{nn}. In particular, if $\Sigma_{nn} = \sigma R$ as before, then $r = r(\sigma)$ is nonincreasing with $r(\sigma) = p$ for all σ sufficiently small and $r(\sigma) = 1$ for all σ sufficiently large. This result formalizes the intuition that the units should increase "cooperation" along with the noise level.

Further generalizations are possible by considering nonMSE measures of the "size" of the linear reconstruction errors $w = y-(BAx+Bn+e)$; see Hornik [23]. In particular, in the Gaussian case, the determinant of $\langle ww' \rangle$ measures the amount of transmitted information, and its constrained maximization is intimately related to the INFOMAX principle of Linsker [9].

IV. Generalization

This section is written with Y. Chauvin and is a modified version of the article "Temporal Evolution of Generalization During Learning in Linear Networks" [24]. The material, copyrighted by MIT Press, was included here with permission from the publisher.

A. Formal Setting

In practice, the question to be answered is how should one allocate limited resources and parameters, such as network size and architecture, initial conditions, training time, and available examples, to optimize generalization performance? One conventional approach is to consider the problem of learning as a surface fitting problem. Accordingly, neural networks should be very constrained, with a minimal number of parameters, to avoid the classical overfitting problem. In practice, however, not too much is known about overfitting and its onset, both as a function of network parameters and training time. Furthermore, the conventional view can be challenged. It may be the case, for instance, that a suitable strategy consists rather in using networks with a few extra parameters. These larger networks must be used in conjunction with nontrivial priors in a Bayesian framework and/or trained for shorter times, based on a careful monitoring of the validation error ("early-stopping").

Partial initial results on generalization problems have been obtained in recent years in terms of VC (Vapnik–Chervonenkis) dimension and statistical mechanics (see, for instance, [25]–[27]). Here, we propose a different and complementary approach consisting of a detailed analysis of generalization in simple feedforward linear networks. Even in this simple framework, the questions are far from trivial. Thus we have restricted the problem even further: learning the identity map in a single-layer feedforward linear network. With suitable assumptions on the noise, this setting turns out to be insightful and to yield analytical results which are relevant to what one observes in more complicated situations. Here, we first define our framework and derive the basic equations, first in the noiseless case and then in the case of noisy data. The basic point is to derive an expression for the validation function in terms of the statistical properties of the population and the training and validation samples. We then examine the main results which consist of an analysis of the landscape of the validation error as a function of training time. Simple simulation results are also presented, and several interesting phenomena are described. The results are discussed in the conclusion, and some possible extensions are briefly mentioned. Mathematical proofs are deferred to the Appendix.

We consider a simple feedforward network with n input units connected by a weight matrix A to n linear output units.[1] The network is trained to learn the identity function (autoassociation) from a set of centered training patterns $x_1, \cdots, x_T$. The connection weights are adjusted by gradient descent on the usual LMS error function

$$E(A) = \langle \|x - Ax\|^2 \rangle.$$

The gradient of E with respect to the weights A is

$$\nabla E = (A - I)\Sigma$$

where $\Sigma = \Sigma_{xx}$ is the covariance matrix of the training set. Thus, the gradient descent learning rule can be expressed as

$$A(k+1) = A(k) - \eta(A(k) - I)\Sigma$$

where η is the constant learning rate. Simple induction shows that

$$A(k) = (I - (I - \eta\Sigma)^k) + A(0)(I - \eta\Sigma)^k.$$

Hence if u_i and λ_i ($\lambda_1 \geq \cdots \geq \lambda_n > 0$) denote the eigenvectors and eigenvalues of Σ, then

$$A(k+1)u_i = (1 - (1 - \eta\lambda_i)^k)u_i + (1 - \eta\lambda_i)^k A(0)u_i. \tag{12}$$

The behavior of (12) is clear: provided the learning rate is less than the inverse of the largest eigenvalue ($\eta < 1/\lambda_1$), $A(k)$ approaches the identity exponentially fast. This holds for any starting matrix $A(0)$. The eigenvectors of Σ tend to become eigenvectors of $A(k)$, and the corresponding eigenvalues approach one at different rates depending on λ_i (larger eigenvalues are learned much faster). As a result, it is not very restrictive to assume, for ease of exposition, that the starting matrix $A(0)$ is diagonal in the u_i basis, that is, $A(0) = U\,\mathrm{diag}(\alpha_i(0))U'$, where as usual, $U = [u_1, \cdots, u_n]$.

[1] Krogh and Hertz [28] have independently analyzed the evolution of generalization in the case of one single linear unit. It can be shown that the evolution curve can assume one of several possible shapes, depending on a number of parameters. Although in the absence of any hidden layer there is no coupling between the output units of an $n-n$ network, it is still necessary to study the $n-n$ case since the corresponding evolution function results from the summation of the evolution curves of each output unit, each such curve being capable of assuming a different shape with different characteristics.

(In fact, learning is often started with the zero matrix.) In this case, (12) becomes

$$A(k)u_i = [1-(1-\eta\lambda_i)^k(1-\alpha_i(0))]u_i = \alpha_i(k)u_i.$$

A simple calculation shows that the corresponding error can be written as

$$E(A(k)) = \sum_{i=1}^{n} \lambda_i(\alpha_i(k)-1)^2.$$

We now modify the setting so as to introduce noise. To fix the ideas, the reader may think, for instance, that we are dealing with handwritten realizations of single-digit numbers. In this case, there are 10 possible patterns but numerous possible noisy realizations. In general, we assume that there is a population of patterns of the form $x+n$, where x denotes the signal and n denotes the noise, characterized by the covariance matrices $\overline{\Sigma} = \overline{\Sigma}_{xx}$, $\overline{\Sigma}_{nn}$, and $\overline{\Sigma}_{xn}$. Here, as everywhere else, we assume that the signal and the noise are centered. A sample $x_t + n_t$ $(1 \le t \le T)$ from this population is used as a training set. The training sample is characterized by the covariance matrices $\Sigma = \Sigma_{xx}$, Σ_{nn}, and Σ_{xn} calculated over the sample. Similarly, a different sample $x_v + n_v$ from the population is used as a validation set. The validation sample is characterized by the covariance matrices $\tilde{\Sigma} = \tilde{\Sigma}_{xx}$, $\tilde{\Sigma}_{nn}$, and $\tilde{\Sigma}_{xn}$. To make the calculations tractable, we shall make, when necessary, several assumptions. First, $\overline{\Sigma} = \Sigma = \tilde{\Sigma}$; thus there is a common basis of unit length eigenvectors u_i and corresponding eigenvalues λ_i for the signal in the population and in the training and validation samples. Then, with respect to this basis of eigenvectors, the noise covariance matrices are diagonal, that is, $\Sigma_{nn} = U\mathrm{diag}(\nu_i)U'$ and $\tilde{\Sigma}_{nn} = U\mathrm{diag}(\tilde{\nu}_i)U'$. Finally, the signal and the noise are always uncorrelated, that is, $\Sigma_{xn} = \tilde{\Sigma}_{xn} = 0$. (Obviously, it also makes sense to assume that $\overline{\Sigma}_{nn} = U\mathrm{diag}(\overline{\nu}_i)U'$ and $\overline{\Sigma}_{xn} = 0$, although these assumptions are not needed in the main calculation.) Thus we make the simplifying assumptions that both on the training and validation patterns the covariance matrix of the signal is identical to the covariance of the signal over the entire population, that the signal and the noise are uncorrelated, and that the components of the noise are uncorrelated in the eigenbase of the signal. Yet we allow the estimates ν_i and $\tilde{\nu}_i$ of the variance of the components of the noise to be different in the training and validation sets.

For a given A, the LMS error function over the training patterns is now

$$E(A) = \frac{1}{T}\sum_t \|x_t - A(x_t+n_t)\|^2.$$

As

$$\Sigma_{xn} = \Sigma_{nx} = 0,$$
$$E(A) = \mathrm{trace}((A-I)\Sigma(A-I)' + A\Sigma_{nn}A').$$

Hence, the gradient of E is

$$\nabla E = (A-I)\Sigma + A\Sigma_{nn}.$$

To compute the image of any eigenvector u_i during training, we have

$$A(k+1)u_i = \eta\lambda_i u_i + (1-\eta\lambda_i-\eta\nu_i)A(k)u_i.$$

Thus by induction

$$A(k) = A(0)M^k - \Sigma(\Sigma+\Sigma_{nn})^{-1}(M^k - I)$$

where $M = I - \eta(\Sigma + \Sigma_{nn})$, and

$$A(k)u_i = \frac{\lambda_i}{\lambda_i+\nu_i}[1-(1-\eta\lambda_i-\eta\nu_i)^k]u_i + (1-\eta\lambda_i-\eta\nu_i)^k A(0)u_i.$$

If again we assume, as in the rest of the section, that the learning rate satisfies $\eta < \min(1/(\lambda_i+\nu_i))$, the eigenvectors of Σ tend to become eigenvectors of $A(k)$ and $A(k)$ approaches the diagonal matrix $\mathrm{diag}(\lambda_i/(\lambda_i+\nu_i))$ exponentially fast. Assuming that $A(0)$ is $\mathrm{diag}(\alpha_i(0))$ in the u_i basis, we get

$$A(k)u_i = \frac{\lambda_i}{\lambda_i+\nu_i}(1-b_i a_i^k)u_i = \alpha_i(k)u$$

where $b_i = 1-\alpha_i(0)(\lambda_i+\nu_i)/\lambda_i$ and $a_i = 1-\eta\lambda_i-\eta\nu_i$. Notice that $0 < a_i < 1$. Using the fact that Σ_{nn} is $\mathrm{diag}(\nu_i)$ and $A(k)$ is $\mathrm{diag}(\alpha_i(k))$ in the u_i basis, we obtain

$$E(A(k)) = \sum_{i=1}^{n} \lambda_i(1-\alpha_i(k))^2 + \nu_i\alpha_i(k)^2. \qquad (13)$$

It is easy to see that $E(A(k))$ is a monotonically decreasing function of k which approaches an asymptotic residual error value given by

$$E(A(\infty)) = \sum_{i=1}^{n} \frac{\lambda_i\nu_i}{\lambda_i+\nu_i}.$$

For any matrix A, we can define the validation error to be

$$E^V(A) = \frac{1}{V}\sum_v \|x_v - A(x_v+n_v)\|^2.$$

Using the fact that $\tilde{\Sigma}_{xn} = 0$ and $\tilde{\Sigma}_{nn} = U\mathrm{diag}(\tilde{\nu}_i)U'$, a derivation similar to (13) shows that the validation error $E^V(A(k))$ is

$$E^V(A(k)) = \sum_{i=1}^{n} \lambda_i(1-\alpha_i(k))^2 + \tilde{\nu}_i\alpha_i(k)^2. \qquad (14)$$

Clearly, as $k \to \infty$, $E^V(A(k))$ approaches its horizontal asymptote, given by

$$E^V(A(\infty)) = \sum_{i=1}^{n} \frac{\lambda_i(\nu_i^2 + \tilde{\nu}_i\lambda_i)}{(\lambda_i+\nu_i)^2}.$$

It is the behavior of E^V before it reaches its asymptotic value, however, which is of most interest to us. This behavior, as we shall see, can be fairly complicated.

B. Validation Analysis

Obviously, $d\alpha_i(k)/dk = -(\lambda_i b_i a_i^k \log a_i)/(\lambda_i + \nu_i)$. Equation (14) and collecting terms yield

$$\frac{dE^V}{dk} = \sum_{i=1}^{n} \frac{2\lambda_i^2 b_i \log a_i}{(\lambda_i + \nu_i)^2} a_i^k (\nu_i - \tilde{\nu}_i + b_i a_i^k (\lambda_i + \tilde{\nu}_i))$$

or, in more compact form

$$\frac{dE^V}{dk} = \sum_{i=1}^{n} \beta_i a_i^k + \gamma_i a_i^{2k}$$

with

$$\beta_i = \frac{2\lambda_i^2 b_i}{(\lambda_i + \nu_i)^2} (\nu_i - \tilde{\nu}_i) \log a_i$$
$$\gamma_i = \frac{2\lambda_i^2 b_i^2}{(\lambda_i + \nu_i)^2} (\lambda_i + \tilde{\nu}_i) \log a_i.$$

The behavior of E^V depends on the relative size of ν_i and $\tilde{\nu}_i$ and the initial conditions $\alpha_i(0)$, which together determine the signs of b_i, β_i, and γ_i. The main result we can prove is as follows.

Assume that learning is started with the zero matrix or with a matrix with sufficiently small weights satisfying, for every i

$$\alpha_i(0) \le \min\left(\frac{\lambda_i}{\lambda_i + \nu_i}, \frac{\lambda_i}{\lambda_i + \tilde{\nu}_i}\right). \tag{15}$$

1) If for every $i, \tilde{\nu}_i \le \nu_i$, then the validation function E^V decreases monotonically to its asymptotic value and training should be continued as long as possible.

2) If for every $i, \tilde{\nu}_i > \nu_i$, then the validation function E^V decreases monotonically to a unique minimum and then increases monotonically to its asymptotic value. The derivatives of all orders of E^V also have a unique zero crossing and a unique extremum. For optimal generalization, E^V should be monitored and training stopped as soon as E^V begins to increase. A simple bound on the optimal training time k^{opt} is

$$\min_i \frac{1}{\log a_i} \log \frac{-\beta_i}{\gamma_i} \le k^{\text{opt}} \le \max_i \frac{1}{\log a_i} \log \frac{-\beta_i}{\gamma_i}.$$

In the most general case of arbitrary initial conditions and noise, the validation function E^V can have several local minima of variable depth before converging to its asymptotic value. The number of local minima is always at most n.

The main result is a consequence of the following statements, which are proved in the Appendix.

Case 1: For every i, $\tilde{\nu}_i \ge \nu_i$, that is, the validation noise is bigger than the training noise.

a) If for every i, $\alpha_i(0) \ge \lambda_i/(\lambda_i + \nu_i)$, then E^V decreases monotonically to its asymptotic value.

b) If for every i, $\lambda_i/(\lambda_i + \tilde{\nu}_i) \le \alpha_i(0) \le \lambda_i/(\lambda_i + \nu_i)$, then E^V increases monotonically to its asymptotic value.

c) If for every i, $\alpha_i(0) \le \lambda_i/(\lambda_i + \tilde{\nu}_i)$ and $\nu_i \ne \tilde{\nu}_i$, then E^V decreases monotonically to a unique global minimum and then increases monotonically to its asymptotic value. The derivatives of all orders of E^V have a unique zero crossing and a unique extremum.

Case 2: For every i, $\tilde{\nu}_i \le \nu_i$, that is, the validation noise is smaller than the training noise.

a) If for every i, $\alpha_i(0) \ge \lambda_i/(\lambda_i + \tilde{\nu}_i)$ and $\nu_i \ne \tilde{\nu}_i$, then E^V decreases monotonically to a unique global minimum and then increases monotonically to its asymptotic value. The derivatives of all orders of E^V have a unique zero crossing and a unique extremum.

b) If for every i, $\lambda_i/(\lambda_i + \nu_i) \le \alpha_i(0) \le \lambda_i/(\lambda_i + \tilde{\nu}_i)$, then E^V increases monotonically to its asymptotic value.

c) If for every i, $\alpha_i(0) \le \lambda_i/(\lambda_i + \nu_i)$, then E^V decreases monotonically to its asymptotic value.

Several remarks can be made on the previous statements. First, notice that in both b) cases, E^V increases because the initial $A(0)$ is already too good for the given noise levels. The monotonicity properties of the validation function are not always strict in the sense that, for instance, at the common boundary of some of the cases E^V can be flat. These degenerate cases can be easily checked directly. The statement of the main result assumes that the initial matrix be the zero matrix or a matrix with a diagonal form in the basis of the eigenvectors u_i. A random initial nonzero matrix, however, will not satisfy these conditions. E^V is continuous and even infinitely differentiable in all of its parameters. Therefore, the results are also true for sufficiently small random matrices. If we use, for instance, an induced l^2 norm for the matrices, then the norm of a starting matrix is the same in the original, or in the orthonormal, u_i basis. Equation (15) yields a trivial upper bound of $n^{1/2}$ for the initial norm which roughly corresponds to having random initial weights of order at most $n^{-1/2}$ in the original basis. Thus, heuristically, the variance of the initial random weights should be a function of the size of the network. This condition is not satisfied in many of the usual simulations found in the literature where inital weights are generated randomly and independently using, for instance, a centered Gaussian distribution with fixed standard deviation.

When more arbitrary conditions are considered, in the initial weights or in the noise, multiple local minima can appear in the validation function. As can be seen in one of the curves of the example given in Fig. 2, there exist even cases where the first minimum is not the deepest one, although these may be rare. Also in this figure, better validation results seem to be obtained with smaller initial conditions. This can easily be understood, in this small-dimensional example, from some of the arguments given in the Appendix.

Another potentially interesting and relevant phenomenon is illustrated in Fig. 3. It is possible to have a situation where, after a certain number of training cycles, both the LMS and the validation functions appear to be flat and to have converged to their asymptotic values. If training is continued, however, one observes that these plateaus can come to an end.

Finally, we have made an implicit distinction between validation and generalization throughout most of the previous sections. If generalization performance is measured by the LMS error calculated over the entire population, it is clear that our main result can be applied to the generalization error by assuming that $\overline{\Sigma}_{nn} = U\text{diag}(\overline{\nu}_i)U'$, and $\tilde{\nu}_i = \overline{\nu}_i$ for every i. In particular, in the second statement of the main

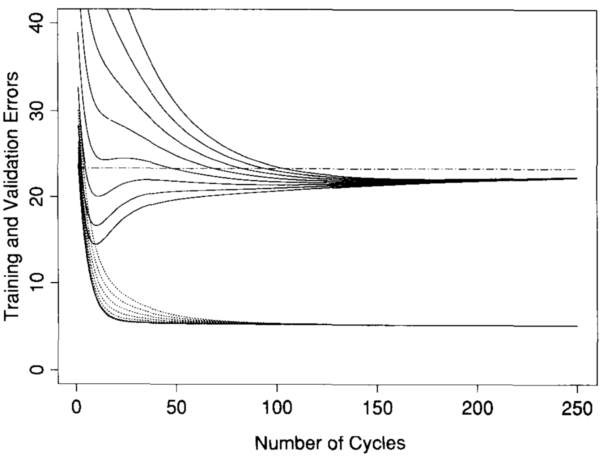
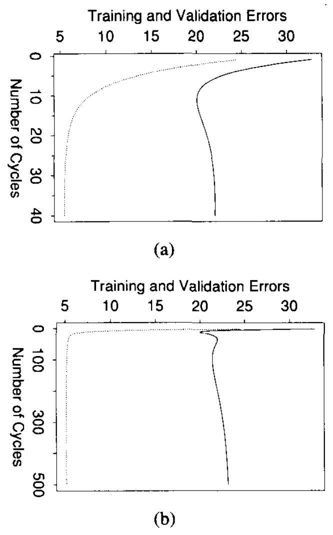


Fig. 2. LMS error functions (lower curves) and corresponding validation error functions (upper curves). The parameters are n =3, λ_1 = 22, .7, 2.5, ν_i = 4, 1, 3, $\bar{\nu}_i$ = 20, 20, 20, $\alpha_1(0)\alpha_2(0) = 0$. From top to bottom, the third initial weight corresponding to $\alpha_3(0)$ takes the values 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5. Notice in particular the fourth validation curve ($\alpha_3(0) = 0.9$), which has two local minima, the second being deeper than the first. At the first minimum, the LMS function is still far from its horizontal asymptote. Also in this case, the validation improves as the initial conditions become closer to zero.



(a)

(b)

Fig. 3. LMS error function (lower curves) and corresponding validation error functions (upper curves). The parameters are +n =3, λ_i = 22, .7, 2.5, ν_i = 4, 1, 4, $\bar{\nu}_i$ = 20, 20, 20, $\alpha_1(0) = \alpha_2(0) = 0$ and $\alpha_3(0) = .7$. Notice, on the first two curves, that after 40 cycles both the LMS and the validation function appear to be flat and would suggest to stop the training. The second set of curves corresponds to 500 training cycles. Notice the existence of a second (although shallow) minimum, undetectable after 40 cycles.

result, if for every i, $\bar{\nu}_i > \nu_i$, then the generalization curve has a unique minimum. Now, if a validation sample is used as a predictor of generalization performance and the ν_i's are close to the $\bar{\nu}_i$'s, then by continuity the validation and the generalization curves are close to each other. Thus, in this case, the strategy of stopping in a neighborhood of the minimum of the validation function should also lead to near-optimal generalization performance (see also [29]).

C. Conclusion

In the framework constructed above, based on linear single-layer feedforward networks, it has been possible to analytically derive interesting results on generalization. In particular, under simple noise assumption, we have given a complete description of the validation error E^V as a function of training time. It is rather remarkable that all the complex phenomena related to generalization which are observed in simulations of nonlinear networks are already present in the linear case. Although our framework is simplistic, it is already quite rich and leads to many nontrivial and perhaps mathematically tractable questions. This analysis is only a first step in this direction, and many questions remain unanswered. For instance, it seems to us that in the case of general noise and arbitrary initial conditions, the upper bound on the number of local minima is rather weak in the sense that, at least on the average, there are far fewer. It seems also that in general the first local minima of E^V is also the deepest. In the analysis conducted here, we have used uniform assumptions on the noise. In general, we can expect this not to be the case, and properties of the noise cannot be fixed *a priori*. Therefore one needs to develop a theory of E^V over different possible noise and/or sample realizations, that is, to find the average curve $\bar{E^V}$. It would also be of interest to study whether some of the assumptions made on the noise in the training and validation sample can be relaxed. Finally, other possible directions of investigation include the extension to multilayer networks and to general input/output associations.

V. Other Algorithms: Unsupervised Learning

The distinction between supervised and unsupervised is sometimes blurry. This is particularly obvious in the autoassociative case. With this in mind, we shall review in this section the main results of unsupervised learning in linear networks and contrast them with the results described in Section III. Since here the learning is unsupervised, the most relevant comparisons will be with the autoassociative mode. We begin with a brief discussion of anti-Hebbian learning for a single linear unit. Next, we look at variance maximization, Hebbian learning, and some of its variations. We start with the single-unit case and then examine the general case with multiple units, show that several "symmetric" algorithms which have been proposed are identical, discuss networks with lateral plastic interactions, and present a unified framework for analyzing these algorithms. Finally, we discuss gradient descent algorithms based on error functions other than (1). As we shall see, many results in the supervised and unsupervised mode are very similar for linear networks, especially the shape of the landscapes and the relation to PCA.

A. Anti-Hebbian Learning

Consider a linear network described by the input–output relation $y = Ax + z$. Suppose it is desired to minimize the average output power $E(A) = \langle \|y\|^2 \rangle = \mathrm{trace}(A\Sigma_{xx}A' + 2A\Sigma_{xz} + \Sigma_{zz})$. Clearly

$$\frac{\partial E}{\partial\, \mathrm{vec}(A)'} = 2\,\mathrm{vec}(A\Sigma_{xx} + \Sigma_{zx}) = 2\,\mathrm{vec}(\langle yx' \rangle).$$

Hence, the optimal A equals $-\Sigma_{zx}\Sigma_{xx}^{-1}$ (as was already known from the results in Section II-D), and the gradient descent algorithm for minimizing $E(A)$ is given by

$$A(k+1) = A(k) - \eta\langle yx' \rangle$$

with corresponding on-line version

$$\Delta A = -\eta y x'.$$

This rule is anti-Hebbian because each weight a_{ij} is modified proportionally to the negative of the product of pre- and postsynaptic activations. Clearly, this rule is equivalent to the "multiple adaptive linear elements" (MADALINE) algorithm of Widrow and Hoff [30] for minimizing the mean-square error of the linear prediction of $-z$ from x by gradient descent.

Palmieri *et al.* [31] have recently reviewed anti-Hebbian learning in linear networks. We refer the reader to their paper for more details.

B. Hebbian Learning: Single Unit Algorithms

Assume, for now, that there is only one linear unit described by the input–output relation $y = a'x$, where a is the vector of weights to be trained, as usual, from a set of centered inputs x.

Suppose, as is often the case, that a desirable goal for the unit is to find a set of weights which differentiates the inputs as much as possible. This can be formalized by requiring that the output variance (power) of the unit be as large as possible. In other words, we want to minimize the cost function

$$E(a) = -\langle (a'x)^2 \rangle = -a'\Sigma_{xx}a.$$

Obviously

$$\frac{\partial E}{\partial a_i} = -2\langle x_i y \rangle.$$

In the corresponding on-line learning rule, upon presentation of a pattern x, a is modified by an amount Δa given by

$$\Delta a_i = \eta x_i y \tag{16}$$

which is exactly Hebb's rule in its simplest form. It is clear, however, that without any other restrictions, E has no minimum and by taking coefficients a_i of arbitrarily large magnitude we can easily have $E \to -\infty$. If we modify the problem so that the variance is to be maximized under the restriction that $\|a\| \leq 1$, then we already know the optimal solution by applying the general result of Section II-A on quadratic forms. The optimal a is equal to $\pm u_1$, where u_1 is the normalized eigenvector of Σ_{xx} corresponding to the largest eigenvalue. At the optimum, the network computes the principal component of the input. In addition, the problem has no local minima. It is instructive to remark that if we try to maximize the variance under the constraint that, for every $i, |a_i| \leq 1$ (i.e., if we constrain the weights to belong to the inside of an n-dimensional cube rather than a sphere), then by a convexity argument it is easy to see that the optimum must be reached at one of the corners of the hypercube where $|a_i| = 1$ for every i (and strictly so if Σ_{xx} is positive definite). The determination of which corner of the hypercube realizes the maximum, however, is an NP-complete problem (see, for instance, the matrix cover problem in [32, p. 282]) and thus probably computationally intractable.

This discussion shows that it is useful to try to modify the simple Hebbian rule of (16) so as to attempt to maximize the output variance while keeping the norm of the weight vector bounded.

Oja [33] suggested keeping the weights normalized by having, upon presentation of pattern x

$$a_i \leftarrow \frac{a_i + \eta y x_i}{\sqrt{\sum_i (a_i + \eta y x_i)^2}}.$$

For η small

$$\Delta a_i = \eta y (x_i - y a_i) + O(\eta^2)$$

which yields, in more compact notation, the learning rule

$$\Delta a = \eta (xy - ay^2) \tag{17}$$

comprising the usual Hebbian term and a very simple normalizing term. By summing the right-hand side of (17) over all patterns, the corresponding off-line version can be expressed in the form

$$a(k+1) = a(k) + \eta(1 - a(k)a(k)')\Sigma_{xx}a(k). \tag{18}$$

Oja's rule cannot be interpreted in terms of a gradient of some error function E because $\partial(x_i y - a_i y^2)/\partial a_j = x_i x_j - 2a_i y x_j - \delta_{i,j} y^2$, and this expression is not symmetric in i and j (in other words, the integrability conditions $\partial^2 E/\partial a_i \partial a_j = \partial^2 E/\partial a_j \partial a_i$ are violated).

What can we say about the convergence of Oja's algorithm? If the on-line or off-line version of the algorithm converges, then by (18) (or equivalently by (17) applied to each pattern)

$$\Sigma_{xx}a = aa'\Sigma_{xx}a$$

must be satisfied at any equilibrium point a. Reasonably $a \neq 0$, and therefore a is an eigenvector of Σ_{xx} with eigenvalue $a'\Sigma_{xx}a$ and, by multiplying the above equation by a' on the left we find that $a'a = 1$. So all the possible limits of the off-line version of the algorithm are the normalized eigenvectors of Σ_{xx}. According to Oja [33] and Oja and Karhunen [34], if the distribution of the patterns x satisfies some reasonable assumptions and if $\eta \to 0$ at a suitable rate, then (17) and (18) can be approximated by the differential equation

$$\dot{a} = \frac{da}{dt} = \Sigma_{xx}a - aa'\Sigma_{xx}a \tag{19}$$

and the solution of (19) will approach with probability one a uniformly asymptotically stable equilibrium of the differential equation. In addition, if Σ_{xx} is positive definite with the largest eigenvalue λ_1 with multiplicity one (and normalized eigenvector u_1) and $a(0)$ is not perpendicular to u_1, then

$a(t) \to \pm u_1$ as $t \to \infty$ and $\pm u_1$ is uniformly asymptotically stable.

The case of Oja's algorithm is typical of what is usually found concerning the relations between the three versions of a given learning algorithm. If the weight changes induced by each pattern presentation are very small, then the on-line version can be approximated by the off-line version (or vice versa). By setting the weight changes to zero in the off-line version, all the possible limit points of the algorithm are derived. The actual limits are in general a strict subset of these possible solutions. In the stochastic approximation framework, where the learning rate tends to zero at a suitable rate and the input environment satisfies certain assumptions, such as stationarity, then the paths of the on- or off-line version asymptotically approach the solution paths of the ordinary differential equation corresponding to the continuous version. In particular, the actual limits must be asymptotically stable equilibria of the differential equation. In general, this requirement is sufficient to find that the learning process converges to the desired value. For more details on the relation between the on-line version and the continuous version, the so-called "associated ODE," see, in particular, Hornik and Kuan [35] and Kuan and Hornik [36].

In Linsker [9], layered networks of units with linear biased output of the form

$$y = a'x + b \tag{20}$$

are considered together with the class of learning rules

$$\Delta a_i = \eta_1 x_i y + \eta_2 x_i + \eta_3 y + \eta_4 \tag{21}$$

that is, Hebbian rules with additional linear or constant terms. The a_i are constrained to be in an interval $[-c, c]$ and the patterns are not necessarily centered. By averaging (21) over all patterns and taking its continuous approximation with the proper units, one can easily derive the system of differential equations

$$\dot{a}_i = \sum_j \sigma_{ij} a_j + k_1 + k_2 \sum_j a_j$$

where σ_{ij} is the (i, j)th element of the covariance matrix Σ_{xx} of the input patterns, and k_1 and k_2 are two constants which can easily be calculated from (20) and (21). In vector notation

$$\dot{a} = \Sigma_{xx} a + k_1 J_{n,1} + k_2 J_{n,n} a \tag{22}$$

where $J_{p,q}$ denotes the $p \times q$ matrix with all entries equal to one. If we let

$$E(a) = -\tfrac{1}{2}(a'\Sigma_{xx}a - 2k_1 a' J_{n,1} - k_2 a' J_{n,1} J'_{n,1} a)$$

then $\partial E / \partial a = -\dot{a}$. Therefore, the learning rule in (22) tends to minimize E. Depending on the values of the constants k_1 and k_2 and the covariance matrix Σ_{xx}, different mature states can be reached. Linsker shows how in layered systems of units satisfying (20), with the proper range of parameters and where successive layers evolve in time according to (22), particular feature detector cells such as center-surround or orientation selective can emerge in different layers, even with completely random external inputs to the first layer. For a theoretical analysis of his simulations, see Miller and MacKay [37]. Linsker also observes that, empirically, the learning process "does not get stuck in high lying local minima." This can be understood in several particular but important situations. Consider, for instance, a unit submitted to random inputs such that $\Sigma_{xx} = I$ and k_1 and k_2 are positive (this is automatically satisfied if all the learning rates in (21) and the averages $\langle x_i \rangle$ are positive). Then the matrix $M = I + k_2 J_{n,n}$ is positive definite. By convexity, the minimum of E must therefore occur at the boundary of the cube $[-c, c]^n$. Recall that, in general, this point on the boundary may be very difficult to determine. Here by inspection, however, the optimum is reached at the vertex $a' = (c, \cdots, c)$. If we consider the system

$$\dot{a} = Ma + k_1 J_{n,1} \tag{23}$$

with $a(0) = a_0$, it is clear that if the initial a_0 has all its components identical, this property will be preserved under the evolution described by (23). As a result, the system will converge to its constrained minimum. In particular, if the initial weights are zero (or random but small), the mature a will essentially be the global minimum. We can also calculate how long it takes for the learning process to converge. If we assume for simplicity that $a_0 = 0$, then the solution of (23) is

$$a(t) = e^{tM} \int_0^t e^{-sM} k_1 J_{n,1} \, ds.$$

A simple induction on n shows that the eigenvalues of M are of two types:

i) $\lambda_1 = 1 + nk_2$, with multiplicity 1 and normalized eigenvector $v_1 = n^{-1/2} J_{n,1}$

ii) $\lambda_2 = \cdots = \lambda_n = 1$ with multiplicity $n - 1$ and normalized eigenvectors $v_2, \cdots, v_n$ orthogonal to v_1 (their explicit form is not required to finish the calculation).

Let V be the matrix $[v_1, \cdots, v_n]$ and Λ the diagonal matrix $\operatorname{diag}(\lambda_1, \cdots, \lambda_n)$ so that $VV' = I$ and $MV = V\Lambda$. Then the matrix exponentials can be computed by using the relation

$$e^{sM} = V \operatorname{diag}(e^{\lambda_1 s}, \cdots, e^{\lambda_n s}) V'.$$

Some additional manipulations finally yield the solution

$$a(t) = k_1 \frac{e^{\lambda_1 t} - 1}{\lambda_1} J_{n,1}$$

and the cell reaches its mature state at time $t = t_c$ where

$$t_c = \frac{1}{\lambda_1} \log\left(1 + \frac{c\lambda_1}{k_1}\right).$$

C. Hebbian Learning: Multi-Unit Algorithms

Let us now consider several linear units simultaneously as described by the input–output relation $y = Ax$. (Notice the discrepancy in notation with respect to the single unit case, where we wrote $y = a'x$.)

Baldi [13] remarks that in the autoassociative case, if we let $C = I$ in (7) and (8), then at the optimum the matrices A and B are transposes of each other. This in turn suggests a possibly faster algorithm, where at each step a gradient correction is applied only to one of the connection matrices, while the

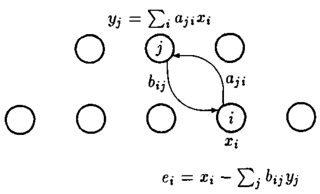


Fig. 4. The SEC (symmetric error correction) network.

other is modified in a symmetric fashion, thus avoiding the backpropagation of errors in one of the layers. One possible version of this idea can be expressed as

$$A(k+1) = A(k) - \eta\frac{\partial E}{\partial \text{vec}(A)'},$$
$$B(k+1) = A(k+1)'$$

with $A(0)$ random and, for instance, $B(0) = A(0)'$. In the averaged form, this gives

$$A(k+1) = A(k) + \eta A(k)\Sigma_{xx}(I - A(k)'A(k)). \quad (24)$$

It can be seen that there exist exceptional starting points which can in theory prevent the algorithm from converging to the optimal solution by incorporating a projection onto a nonprincipal eigenvector. Moreover, it can be seen that a necessary condition for convergence to the optimum is $\eta < 1/2\lambda_1$ (at least in the final stages of the learning process). If we specialize the evolution of the matrix A to the case of a single unit, we find for $a(k) = A(k)'$ that

$$a(k+1) = a(k) + \eta(\Sigma_{xx}a(k) - a(k)a(k)'\Sigma_{xx}a(k))$$

which is identical to (18). In other words, Oja's algorithm is the one-dimensional version of (24).

In Williams [20], the following SEC (symmetric error correction) learning algorithm is described. Consider a linear network, comprising only two layers, n input units connected to p hidden units with a connection matrix A, and feedback connections from the p hidden units back to the input units through a connection matrix B (see Fig. 4). A pattern x is presented and propagated forward to give $y = Ax$ and then backwards to allow the computation of an error $e = x - By$. The weight matrix A is then corrected according to

$$\Delta a_{ij} = \eta y_i e_j \quad (25)$$

and B is updated symmetrically, in the sense that $\Delta b_{ji} = \Delta a_{ij}$ or $\Delta B = (\Delta A)'$. In matrix notation, (25) is easily rewritten as

$$\Delta A = \eta Ax(x - BAx)'$$

or, in averaged form

$$A(k+1) = A(k) + \eta(A(k)\Sigma_{xx}(I - A(k)'B(k)'))$$

which is (24), provided the algorithm is started with symmetric initial conditions, i.e., $B(0) = A(0)'$.

We could also start from Oja's one-unit algorithm and try to generalize it to the case of many units. Hence, we are looking for learning rules

$$\Delta A = \eta(yx' + R)$$

where the first term is the usual Hebbian term to maximize the sum of the output variances, and R is a correction term to keep the connection matrix A (sub)orthogonal ($AA' = I$). That is, we should have $(A + \Delta A)(A + \Delta A)' \approx I$. Now for η small and $AA' = I$ we have

$$(A + \Delta A)(A + \Delta A)' = I + \eta(2yy' + AR' + A'R) + O(\eta^2).$$

Thus, R should sati fy the first-order condition $AR' + RA' = -2yy'$. If we choose R of the form $R = SA$, we obtain $S + S' = -2yy'$. Hence, we can either take S (or S') as $-yy'$ or as $-\text{diag}(yy') - 2\,\text{subdiag}(yy')$. (The "subdiag" operator sets all entries on and above the main diagonal to zero.)

The former choice, $S = -yy'$, gives

$$\Delta A = \eta(yx' - yy'A) \quad (26)$$

which is called the "subspace" algorithm in Oja [38]. In its averaged form

$$A(k+1) = A(k) + \eta(\Sigma_{xx}A(k) - A(k)A(k)'\Sigma_{xx}A(k))$$

which is identical to (24). In conclusion, we see that, quite remarkably, several algorithms proposed in the literature in Oja [33] and its generalizations in Baldi [13], Oja [38], and Williams [20] are in fact completely identical, although they were derived using different heuristics. Robustified versions of this rule have recently been given in Xu and Yuille [39].

The latter choice for S gives the "stochastic gradient ascent" (SGA) algorithm

$$\Delta A = \eta(yx' - \text{diag}(yy')A - 2\,\text{subdiag}(yy')A)$$

of Oja and Karhunen [34]. In fact, this rule can also be obtained as a first-order approximation to the Gram–Schmidt orthonormalization of the rows of A after a Hebbian step $\tilde{A} \leftarrow A + \eta yx'$. This is the most "natural" generalization of the derivation of Oja's rule to the multi-unit case. We notice that SGA is very similar to the "generalized Hebbian algorithm" (GHA)

$$\Delta A = \eta(yx' - \text{lower}(yy')A)$$

proposed by Sanger [40]. (The "lower" operator sets all entries above the main diagonal to zero.) In fact, since $\text{lower}(M) = \text{diag}(M) + \text{subdiag}(M)$, the rules differ by a $\text{subdiag}(yy')A$ term; see, e.g., Oja [41].

Several of the above algorithms were constructed with the objective to maximize the average output power $\langle\|y\|^2\rangle = \sum_i\langle y_i^2\rangle = \text{trace}(A\Sigma_{xx}A')$ over $\mathcal{A} = \{A : AA' = I\}$. (Observe that by our results in Section II-A, we already know that all such A are of the form CU_p' with C orthogonal, and hence result in PCA analyzers.) As proposed in Brockett [42], this goal can also be accomplished by constrained gradient ascent on the average output power. More generally, consider the weighted average power $E(A) = \sum_i \theta_i\langle y_i^2\rangle$. Notice that if $\theta_1 > \cdots > \theta_p > 0$, then $E(A)$ is maximized over $\mathcal{A}$ iff $A = [\pm u_1, \cdots, \pm u_p]'$. By this choice for the weights, the rows of A are forced to be the first p eigenvectors of Σ_{xx} rather than an orthonormal set of linear combinations of them. Writing $\Theta = \text{diag}(\theta_1, \cdots, \theta_p)$, we have

$$E(A) = \langle\text{trace}(\Theta yy')\rangle = \text{trace}(\Theta A\Sigma_{xx}A').$$

Brockett [42] shows that the gradient flow of E on $\mathcal{A}$ (with respect to the canonical Frobenius-norm based metric on $\mathcal{A}$) is given by

$$\dot{A} = \Theta A \Sigma_{xx} - A\Sigma_{xx} A' \Theta A. \tag{27}$$

(In fact, Brockett only considers the special case $p = n$, but the analysis for the case $p < n$ is virtually identical, cf. also the remarks in [43].) This can be seen as follows. The neighborhood of A in $\mathcal{A}$ can be parameterized as $A(\Omega) = Ae^{\Omega} = A(I + \Omega + \Omega^2/2 + \cdots)$ with skew-symmetric Ω (i.e., $\Omega' = -\Omega$). Hence for Ω small

$$\begin{aligned} &E(A(\Omega)) - E(A) \\ &\quad = \mathrm{trace}(\Theta A(I+\Omega)\Sigma_{xx}(I-\Omega)A') + \cdots - E(A) \\ &\quad = \mathrm{vec}(\Omega)'\mathrm{vec}(A'\Theta A\Sigma_{xx} - \Sigma_{xx}A'\Theta A) + \cdots \end{aligned}$$

and thus the gradient flow is $\dot{A} = A\dot{\Omega} = A(A'\Theta A \Sigma_{xx} - \Sigma_{xx}A'\Theta A) = \Theta A\Sigma_{xx} - A\Sigma_{xx}A'\Theta A$.

We notice that for $\Theta = I$, Brockett's algorithm simplifies to $\dot{A} = A\Sigma_{xx} - A\Sigma_{xx}A'A$, which is just the continuous version of the subspace algorithm. Therefore, this rule performs constrained gradient descent over $\mathcal{A}$. For unconstrained A, however, it does not perform gradient descent (see the discussion of the Oja one-unit algorithm further above). Nevertheless, as recently shown in Xu [43], it still goes "downhill" for the error function $\langle \|x - A'Ax\|^2 \rangle$. More precisely, if $\dot{A} = G(A)$ and $\dot{A} = G_0(A)$ are the continuous versions of gradient descent on this error function and the subspace algorithm, respectively, then

$$\mathrm{vec}(G_0(A))'\mathrm{vec}(G(A)) = 2\,\mathrm{vec}(G_0(A))'\mathrm{vec}(G_0(A))$$

which is strictly positive unless $G_0(A) = O$, i.e., A is a (possible) limit point of the subspace algorithm.

Finally, if we write $z = \Theta y = \Theta Ax$ (i.e., the activation function of output unit i is amplification by θ_i), we can rewrite the online version of Brockett's algorithm as

$$\Delta A = \eta(zx' - \Theta^{-1}zz'A).$$

This is formally equivalent to the weighted subspace algorithm of Oja [41].

D. Hebbian Learning: Lateral Interaction Algorithms

Up to this point, no lateral connections within the layer of hidden units were allowed. Clearly, if we use Hebbian learning with row-wise normalization or equivalently, if we apply Oja's one-unit algorithm to each of the rows of A, then all output units end up doing the same, namely extract the first principal component. An additional mechanism which introduces competition or some hierarchical order between the output units, however, might force the network to perform full PCA.

For example, as in Földiák [44], we can consider a linear architecture where the outputs are updated according to

$$y \leftarrow Ax + Wy. \tag{28}$$

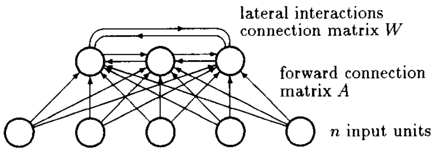


Fig. 5. Network with lateral interactions.

Here, y is the vector of activities in the p output units (there are no hidden units), A is the connection matrix from the inputs x to the outputs y, and W is the zero-diagonal matrix of lateral inhibitory connections among the y units. Földiák suggests to first keep applying (28) until the network settles to a stable state for which $y = Ax + Wy$ or, equivalently, $y = (I - W)^{-1}Ax$. The A matrix is then updated using Oja's algorithm for each row, i.e.,

$$\Delta A = \eta(yx' - \mathrm{diag}(yy')A). \tag{29}$$

The matrix W is initialized as O and adapted using the simple anti-Hebbian rule

$$\Delta W = -\mu\, \mathrm{offdiag}(yy') \tag{30}$$

(the "offdiag" operator sets the diagonal entries to zero).

Unfortunately, there are some serious problems with this rule. Let $y(k)$ denote the network output after k updating cycles (28) with fixed input x and initial output $y(0)$. Clearly

$$y(k) = W^k y(0) + (I + \cdots + W^{k-1})Ax.$$

To ensure convergence of $y(k)$ to $(I-W)^{-1}Ax$ as $k \to \infty$, we thus need that all eigenvalues of W be less than one in absolute value. This is not guaranteed by the algorithm. Even if this condition is satisfied, we note that as (30) keeps W symmetric, W^k is always nonzero unless $W = O$. Thus, it really requires infinitely many cycles to converge to the stable state, which is computationally infeasible. (Of course, the linear system $(I - W)y = Ax$ could be solved explicitly in finite time; but then the architecture is no longer self-contained, and the particularly attractive feature of performing only simple local computations is lost.) Hence, real-time implementations of Földiák's algorithm should only use a finite number of cycles (28), that is, one first updates y according to

$$y \leftarrow (I + \cdots + W^{k-1})Ax + W^k y$$

for some $k \geq 1$, and then updates A and W according to (29) and (30).

If it is really desired to stabilize the network outputs before updating the weights, one should have $W^k = O$ for some k. As already pointed out in an earlier version of this paper and in Hornik and Kuan [35], this can be accomplished by keeping W subdiagonal, rather than symmetric with zero diagonal, that is, by replacing (30) with the asymmetric anti-Hebbian rule

$$\Delta W = -\mu\, \mathrm{subdiag}(yy'). \tag{31}$$

In this case, $W^p = O$, and $y = (I - W)^{-1}Ax$ after p cycles. Clearly, this rule introduces a strict hierarchy between the output units, whereas the former forces symmetric competition.

A similar approach was taken in Rubner and Tavan [45]. There, the network is defined by the relation

$$y = (I + W)Ax, \qquad \mathrm{diag}(W) = 0 \tag{32}$$

i.e., the lateral connections W act on Ax directly rather than y, corresponding to a different interpretation of the feedback mechanism. (We can also interpret (32) as the input–output relation of a linear n–p–p network with input-to-hidden weights A for feature extraction and hidden-to-output weights Q for decorrelation, with all diagonal entries of Q hardwired to one.) In Rubner and Tavan's algorithm, A is updated using Hebbian learning with subsequent row normalization and W according to (31); but of course, one could also use the more local rule (29) for A and/or the symmetric anti-Hebbian rule (30) for W.

Two different Hebbian algorithms with lateral inhibition are proposed in Leen [46]. He starts with the "potential" (energy function)

$$U = -\frac{1}{2}\sum_i \langle y_i^2\rangle + \frac{\delta}{2}\sum_{j\neq k}(\langle y_j y_k\rangle)^2$$

for the simple architecture $y = Ax$. The first term is the usual average output power, and the second one is an interaction potential which penalizes correlations between the output unit activations. The trade-off between the two terms is measured by the coupling constant δ. As U is clearly unbounded, gradient descent on U has to be stabilized by an additional weight decay term. If the usual local Oja term is used, one obtains the continuous version in the form

$$\dot{A} = \langle yx'\rangle - \delta\,\mathrm{offdiag}(\langle yy'\rangle)\langle yx'\rangle - \mathrm{diag}(\langle yy'\rangle)A. \qquad (33)$$

This is not well suited for direct on-line implementation, however, as usually $\mathrm{offdiag}(\langle yy'\rangle)\langle yx'\rangle \neq \langle \mathrm{offdiag}(yy')yx'\rangle$. This problem can be overcome by introducing an additional weight matrix W which keeps track of the covariances $\langle y_j y_k\rangle$ between different output units according to

$$\dot{W} = -\kappa(W + \delta\,\mathrm{offdiag}(\langle yy'\rangle)). \qquad (34)$$

If $\kappa \gg 1, W \approx -\delta\,\mathrm{offdiag}(\langle yy'\rangle)$ which can be substituted into (33) to give

$$\dot{A} = (I - W)\langle yx'\rangle - \mathrm{diag}(\langle yy'\rangle)A. \qquad (35)$$

The analysis in Leen [46] shows (see also Section V-E) that the "desired" limits $A = U_p'$, $W = O$ are asymptotically stable iff

$$\delta > \max_{i\neq j}\frac{1}{\lambda_i + \lambda_j}, \qquad \kappa > \max_{i\neq j}\frac{(\lambda_i - \lambda_j)^2(\lambda_i + \lambda_j)}{\lambda_i^2 + \lambda_j^2}.$$

Hence, the minimal coupling and relaxation constants δ and κ depend on the size of the eigenvalues of Σ_{xx} and do not scale well with n (the larger n, the more likely small eigenvalues become). This problem can be overcome by replacing (34) with

$$\dot{w}_{ij} = -\langle y_i^2 + y_j^2\rangle w_{ij} - \delta\langle y_i y_j\rangle, \qquad i \neq j. \qquad (36)$$

With this modification, the stability requirement becomes $\delta > 1$.

The second algorithm proposed by Leen uses full lateral coupling between the outputs as specified by

$$y \leftarrow Ax + Wy.$$

In equilibrium, $y = (I - W)^{-1}Ax$, and A and W are updated according to (29), respectively

$$\dot{W} = \alpha W - \delta\,\mathrm{offdiag}(\langle yy'\rangle)$$

or preferably

$$\dot{w}_{ij} = \langle y_i^2 + y_j^2\rangle w_{ij} - \delta\langle y_i y_j\rangle, \qquad i \neq j.$$

We observe that as W is kept symmetric by this algorithm, the same problems with the stabilization of the outputs into $y = (I - W)^{-1}Ax$ exist as in Földiák's algorithm. Again, in a real-time implementation, we either have to keep W subdiagonal or perform a finite number of output updating cycles only. In fact, the first-order approximation $(I - W)^{-1}$ by $I + W$ which changes the input–output relation to $y = (I + W)Ax$ as in the architecture of Rubner and Tavan can already deliver the desired behavior for suitably chosen δ.

Another lateral inhibition algorithm is the "Adaptive Principal Component Extractor" (APEX) of Kung and Diamantaras [47]. APEX is based on the network $y = (I + W)Ax$ with W subdiagonal and uses the updating equations

$$\begin{aligned}\Delta A &= \eta(yx' - \mathrm{diag}(yy')A),\\ \Delta W &= -\eta(\mathrm{subdiag}(yy') + \mathrm{diag}(yy')W).\end{aligned}$$

We notice that this algorithm is similar to that of Rubner and Tavan [45] (it uses an additional decay term for W) and in fact identical to the variant of Leen's algorithm with full coupling obtained by keeping W subdiagonal and approximating $(I - W)^{-1}$ by $I + W$.

E. Hebbian Learning: A Unifying Framework

In the preceding two subsections we have presented a variety of constrained Hebbian learning algorithms in linear networks. We have seen many similarities between these algorithms, which is not too surprising as they were all constructed with the same goal in mind, namely maximizing average output variance under suitable constraints. The most important properties of these algorithms can satisfactorily be analyzed and understood within the following general framework.

Consider an adaptive linear system with updating rule

$$y \leftarrow P(W)y + Q(W)Ax \qquad (37)$$

where

$$P(W) = \sigma W + \cdots, \qquad Q(W) = (I + \tau W + \cdots)\Theta$$

with the dots indicating terms containing higher powers of W. As already discussed, in on-line implementations P and Q are finite-order polynomials in W, but the above also contains the case of full output stabilization where $Q(W) = (I - W)^{-1} = I + W + W^2 + \cdots$. The parameters of the system are adapted according to

$$\begin{aligned}\Delta A &= \eta(yx' - \Phi(yy')A) \qquad (38)\\ \Delta W &= \eta(\alpha W + \beta\,\mathrm{diag}(yy')W\\ &\quad + \gamma\, W\mathrm{diag}(yy') + \delta\,\Omega(yy')). \qquad (39)\end{aligned}$$

Here, Φ and Ω are suitable operators linear in the entries of yy', typically selection operators. This class of adaptive systems generalizes the framework of Hornik and Kuan [35] to contain all previously considered cases. (The class considered in their paper has $\Theta = I$ and $\alpha = \beta = \gamma = 0$.) We notice that in the networks without lateral interactions considered in Section V-C, $W \equiv O$, and the above simplifies to

$$\Delta A = \eta(yx' - \Phi(yy')A), \qquad y = \Theta Ax.$$

Brockett's algorithm is obtained with $\Phi(M) = \Theta^{-1}M$, and the subspace algorithm with the additional choice $\Theta = I$. Sanger's GHA corresponds to Φ = lower and $\Theta = I$, and the Oja and Karhunen SGA to $\Phi(M) = \mathrm{diag}(M) + 2\,\mathrm{offdiag}(M)$ and $\Theta = I$.

For the local algorithms in networks with lateral interactions discussed in Section V-D, Φ = diag. Initializing W with O and using Ω = offdiag keeps W symmetric with zero diagonal; choosing Ω = subdiag keeps W subdiagonal. Independently of the other choices, both selections for Ω are always possible; i.e., the algorithm can be run in symmetric or asymmetric mode. We have already discussed that these selections implement a competitive and a strictly hierarchical decorrelation mechanism, respectively.

The above general class of constrained Hebbian algorithms can most conveniently be analyzed in terms of the corresponding continuous versions. As explained in Hornik and Kuan [35], the "correct" averages are

$$\begin{aligned}\langle yx' \rangle &= Q(W)A\Sigma_{xx}\\ \langle yy' \rangle &= \sum_{i=0}^{\infty} P(W)^i Q(W)A\Sigma_{xx}A'Q(W')P(W')^i\\ &:= R(A, W)\end{aligned}$$

hence, the associated ODE's (ordinary differential equations) are

$$\dot{A} = Q(W)A\Sigma_{xx} - \Phi(R)A \tag{40}$$

$$\dot{W} = \alpha W + \beta\,\mathrm{diag}(R)W + \gamma\, W\mathrm{diag}(R) + \delta\,\Omega(R) \tag{41}$$

with $R = R(A, W)$, $A(0)$ "random" (but clearly nonzero), and $W(0) = O$. If the mature network is to act as a fast principal component analyzer, the desired equilibria are those for which $A = CU_p'$, for suitable invertible C, and $W = O$ (such that the network output can be computed from a new input in a single cycle). Hence ideally, all (asymptotically) stable equilibria of the system given by (40) and (41) should be of the desired form, and all other equilibria should be unstable. We notice, however, that if the A equilibria are not isolated (as, e.g., in the case of the subspace algorithm), they cannot be asymptotically stable. In these cases, the requirements should really be that the space spanned by the rows of A, i.e., the projection onto this space, be asymptotically stable in the induced dynamics.

In an equilibrium of (40)

$$Q(W)A\Sigma_{xx} = \Phi(R)A.$$

If $Q(W)$ is invertible, we can rewrite this as

$$A\Sigma_{xx} = MA, \qquad M = Q(W)^{-1}\Phi(R).$$

Hence, if u is an eigenvector with associated eigenvalue λ

$$\lambda Au = A\Sigma_{xx}u = MAu$$

i.e., Au is either zero or an eigenvector of M with associated eigenvalue λ. In this sense, the row space of A is spanned by the eigenvectors of Σ_{xx}. More precisely, we can find matrices U_0 and $U_\perp$ having mutually perpendicular unit length eigenvectors of Σ_{xx} as their columns, such that $A = CU_0'$ and $AU_\perp = O$. Of course, the conditions $\alpha W + \beta\,\mathrm{diag}(R)W + \gamma\, W\mathrm{diag}(R) + \delta\,\Omega(R) = O$ and $R = R(A, W)$ place further restrictions on A. For a complete description of the equilibria of the Brockett-type algorithms, see, e.g., Xu [43], for those of Sanger's GHA, see, e.g., Hornik and Kuan [35]. For the local algorithms with Φ = diag and $\Theta = I$, equilibria with $W = O$ satisfy

$$A\Sigma_{xx} = \mathrm{diag}(R)A, \qquad R = A\Sigma_{xx}A'$$

and either $\mathrm{subdiag}(R) = O$ or $\mathrm{offdiag}(R) = O$ which in either case forces R to be diagonal. Hence, the nonzero rows of A are unit length eigenvectors of Σ_{xx}. Unfortunately, a complete description of the equilibria of the local algorithms with arbitrary α, β, and γ has not been given yet and appears to be particularly challenging in the symmetric case. For the asymmetric mode with $\alpha = \beta = \gamma = 0$, Hornik and Kuan [35] have shown that if $\tau \neq 0$, the only equilibria with subdiagonal W and full rank R are those for which the rows of A are mutually perpendicular unit length eigenvectors of Σ_{xx} and $W = O$. Equilibria with rank deficient R are unstable.

Let us briefly indicate how the stability properties of the equilibria can be analyzed. For simplicity, assume $W = O$ and $\Theta = I$. Let E and H denote small perturbations of A and W, respectively. After linearization we obtain

$$\begin{aligned}\dot{E} &= (E + \tau H)A\Sigma_{xx}\\ &\quad - \Phi(R)E - \Phi(dR(E, H; A, O))A,\\ \dot{H} &= \alpha H + \beta\,\mathrm{diag}(R)H + \gamma\, H\mathrm{diag}(R)\\ &\quad + \delta\,\Omega(dR(E, H; A, O))\end{aligned}$$

where dR denotes the Fréchet differential of R at (A, O). Let U_0 and $U_\perp$ be as constructed further above, let Λ_0 and $\Lambda_\perp$ be the diagonal matrices of the associated eigenvalues of Σ_{xx}, and let $E_0 = EU_0$ and $E_\perp = EU_\perp$. Then it is readily verified that with $C = AU_0$, $dR(E, H; A, O) = (\tau)(HR + RH') + E_0\Lambda_0 C' + C\Lambda_0 E_0'$, i.e., dR is of the form $\Upsilon(E_0, H)$. The above sensitivity system thus gives

$$\dot{E}_\perp = E_\perp\Lambda_\perp - \Phi(R)E_\perp \tag{42}$$

$$\begin{aligned}\dot{E}_0 &= (E_0 + \tau HC)\Lambda_0\\ &\quad - \Phi(R)E_0 - \Phi(\Upsilon(E_0, H))C,\end{aligned} \tag{43}$$

$$\begin{aligned}\dot{H} &= \alpha H + \beta\,\mathrm{diag}(R)H + \gamma\, H\mathrm{diag}(R)\\ &\quad + \delta\,\Omega(\Upsilon(E_0, H)).\end{aligned} \tag{44}$$

The key insight is that (42), which describes the evolution of perturbations of A perpendicular to the rows of A, is completely decoupled from (43) and (44) which describe the evolution of the perturbations of W and the component of the perturbations of A along the row space of A. Thus, (42) can be analyzed separately and typically be used to show that equilibria A are unstable, unless their rows span the same space as the first p eigenvectors of Σ_{xx}.

In particular, for the lateral inhibition algorithms, this implies that A is unstable unless $A = U_p'$ (up to signs); hence we can take $C = I$ and the remaining system described by (43) and (44) decomposes into one-dimensional systems for the diagonal components of E_0 and three-dimensional systems for the off-diagonal entries of EU_0 and the corresponding entries of W, e.g., for $[E_0]_{ij}$, $[E_0]_{ji}$, and $[H]_{ij}$ (observe that H is either subdiagonal or symmetric with zero diagonal). Proceeding along these lines, the stability results of Leen described above and those in Hornik and Kuan [35] can be obtained. As of yet, however, no complete stability results for the class of lateral inhibition algorithms have been given; this issue is currently under investigation. In any case, the results of Leen show that the conclusion of Hornik and Kuan that "hierarchical decorrelation should always be preferred over more competitive, symmetric decorrelation mechanisms, for reasons of superior performance of the algorithms" [35, p. 235], cannot be maintained within the more general framework considered here.

Which of the algorithms presented above should we really employ for PCA learning? This question cannot be answered completely, as in addition to the stability properties of the associated ODE, issues of computational complexity and storage requirements need to be considered. This is beyond the scope of this paper. Nevertheless, we still maintain as a minimal requirement that the asymptotically stable equilibria be desired limit points and, conversely, that the desired limit points "allowed" by the algorithm (i.e., the equilibria of the associated ODE) be asymptotically stable. This rules out the subspace algorithm in favor of general Brockett-type rules where Θ has distinct positive entries. For the lateral inhibition algorithms, this implies that in symmetric mode, the simple anti-Hebbian decorrelation mechanism as, e.g., proposed by Barlow and Földiák [48] and used in Földiák [44] has to be combined with an additional weight decay W term as proposed in Leen [46]. Finally, we notice that if these algorithms are used in asymmetric mode, additional units can be added without retraining the already mature part of the network, i.e., one can incrementally build the principal component extractor. This property could be extremely attractive in some engineering applications.

F. Gradient-Based Learning

Chauvin [49] proposed an approach based on the construction of a cost function comprising two terms: a variance term to be maximized and a term penalizing large weight vectors. More explicitly, one wants to minimize

$$E(a) = -\alpha a'\Sigma_{xx}a + \beta(a'a-1)^2 \tag{45}$$

where α and β are two positive real coefficients that can be varied to adjust the relative importance of the two factors. The derivatives of this cost function corresponding to the presentation of one pattern x are

$$\frac{\partial E}{\partial a_i} = -2\alpha x_i y + 4\beta(a'a-1)a_i$$

with the corresponding learning rule

$$\Delta a_i = \eta(2\alpha x_i y - 4\beta(a'a-1)a_i) \tag{46}$$

and its averaged vector version

$$a(k+1) = a(k) + \eta(2\alpha\Sigma_{xx}a(k) - 4\beta(a(k)'a(k)-1)a(k)). \tag{47}$$

Notice that, in addition to the usual Hebbian part, (46) contains a normalizing term which is not very local in the sense that it depends on all the weights a_i. Because of the competition between the two terms, it is clear that E has a minimum which is attained for some optimal a_{opt}. If we consider E restricted to the surface $\|a\| = \|a_{\text{opt}}\| = \rho$, the second term on the right-hand side of (45) remains constant. Hence, by applying again the result on the optimization of quadratic forms over spheres, we have that a_{opt} is collinear to u_1, that is $\Sigma_{xx}a_{\text{opt}} = \lambda_1 a_{\text{opt}}$. Since we know the direction of a_{opt}, it is now sufficient to determine its length ρ. Clearly, $E(a_{\text{opt}}) = -\alpha\lambda_1\rho^2 + \beta(\rho^2-1)^2$ and this is a quartic polynomial which is minimized when $dE/d\rho = 0$, i.e., for $\rho = \pm(1+\alpha\lambda_1/2\beta)^{1/2}$. So finally

$$a_{\text{opt}} = \pm\left(1+\frac{\alpha\lambda_1}{2\beta}\right)^{1/2} u_1.$$

Notice that this analysis is to a certain extent independent of the detailed form of the term used to constrain the length of a in $E(a)$ (the term needs only to be some function of $\|a\|$). What can be said about the rest of the landscape of E? Clearly, at any equilibrium point of the algorithm, the relation

$$2\alpha\Sigma_{xx}a = 4\beta(a'a-1)a$$

must be satisfied and therefore either $a = 0$ or a is an eigenvector of Σ_{xx}. It can be shown (see [49] for details) that for $a = 0, E(0) = \beta$, and this corresponds to a local maximum of E. If Σ_{xx} is positive definite with all eigenvalues of multiplicity one, then all the critical points of E are of the form $a = \pm(1+\alpha\lambda_i/2\beta)^{1/2}u_i$ with associated cost $E(a) = -\alpha\lambda_i(1+\alpha\lambda_i/4\beta)$ $(i = 1,\cdots,n)$. All these critical points are saddle points with the exception of a_{opt} (corresponding to $i = 1$) which, as we have already seen, realizes the global minimum of E. In particular, this landscape has no local minima. In the off-line version, if the starting weight vector is not orthogonal to u_1 (and this can always be assumed for practical purposes) then, provided that the learning rate satisfies

$$\eta \ll \frac{1}{2\alpha\lambda_1} \quad \text{and} \quad \eta < \frac{1}{4(2\beta+\alpha\lambda_1)}$$

(47) always leads to a decrease of the cost function E and therefore the algorithm must converge.

We already pointed out in Section V-C that in the auto-associative case, if we let $C = I$ in (7) and (8), then at the optimum the matrices A and B are transposes of each other. This might suggest training A by gradient descent on the error function

$$E(A) = \langle\|x - A'Ax\|^2\rangle. \tag{48}$$

This possibility and its close relation to the subspace algorithm was first mentioned in a previous version of this paper and recently analyzed in great detail in Xu [43]. In particular, Xu shows that the full rank critical points of E are $A = CU_{\mathcal{I}}'$ with $CC' = I$ and are saddle points unless $\mathcal{I} = \{1, \cdots, p\}$, as to be expected from (7) and (8).

After some computation

$$\frac{\partial E}{\partial \operatorname{vec}(A)'} = \operatorname{vec}(2\,(A\Sigma_{xx}A'A + AA'A\Sigma_{xx}) - 4\Sigma_{xx}A)$$

and thus the on-line version of gradient descent on E is

$$\Delta A = \eta(2yx' - yy'A - AA'yx'). \tag{49}$$

We note that this rule is not particularly simple or local and that if $AA' = I$, the right-hand side is the same as in the subspace algorithm. If we follow Xu, however, and introduce additional units with activations $\hat{x} = A'y$ and $\hat{y} = A\hat{x} = AA'y$, we can rewrite (49) as

$$\Delta A = \eta(y(x - \hat{x})' + (y - \hat{y})x').$$

This can also be interpreted as a "doubly symmetric" modification of Williams' SEC algorithm, where the additional symmetry incurs from equal treatment of the reconstruction errors $x - \hat{x}$ at the input and $y - \hat{y}$ at the output layer. In terms of the full set of activations $x, y, \hat{x}$, and $\hat{y}$, the algorithm is trivially local, illustrating how locality can be achieved at the expense of additional storage and internal computations.

Similar to the subspace algorithm, the optimal A are not isolated and hence are not asymptotically stable equilibria of the continuous version of the algorithm. As a remedy, Xu introduces an additional amplification $z = \Theta y$ at the outputs; the algorithm is then modified as

$$\Delta A = \eta(z(x - \hat{x})' + (z - \hat{z})x') \tag{50}$$

where $\hat{x} = A'z$ and $\hat{z} = A\hat{x} = AA'z$. (The relation between the original and the modified version is the same as the one between the subspace algorithm and Brockett's algorithm.) If the $\theta_1 > \cdots > \theta_p > 0$, the asymptotically stable equilibria of the continuous version of (50) are exactly $A = U_p'$ (up to sign), as desired.

Of course, other error functions are possible, too. Without giving details, we mention the higher-order algorithm of Lenz and Österberg [50] which performs gradient descent on the ratio of the information transmitted by the network $y = Ax$ and the concentration $\sum_i \langle y_i^2\rangle(1 - \langle y_i^2\rangle)$ of the outputs, and constrained Lagrangian approach of Cichocki and Unbehauen [51].

Appendix
Some Mathematical Proofs

A. Noise Analysis: Some Remarks and the Case of High Levels of Noise

Consider the setting of Section III-C. As in Baldi and Hornik [6], one has the following two facts for $\tilde{E}$.

Fact 1: For any fixed B, the function $\tilde{E}(A, B)$ is convex in the coefficients of A and attains its minimum for any A satisfying

$$B'BA\Sigma_{xx} = B'\Sigma_{yx}.$$

If Σ_{xx} is invertible and B is full rank p, then $\tilde{E}$ is strictly convex and has a unique minimum reached when

$$A = \tilde{A}(B) = (B'B)^{-1}B'\Sigma_{yx}\Sigma_{xx}^{-1}.$$

This follows immediately since, for fixed B, the additional term $\operatorname{trace}(B\Sigma_{nn}B') + \operatorname{trace}(\Sigma_{ee})$ is just an additive constant.

Fact 2: For any fixed A, the function $\tilde{E}(A, B)$ is convex in the coefficients of B and attains its minimum for any B satisfying

$$B(A\Sigma_{xx}A' + \Sigma_{nn}) = \Sigma_{yx}A'.$$

In particular, if Σ_{nn} is invertible, then $\tilde{E}$ is strictly convex and has a unique minimum reached when

$$B = \tilde{B}(A) = \Sigma_{yx}A'(A\Sigma_{xx}A' + \Sigma_{nn})^{-1}.$$

For the proof, it can be checked that the gradient of $\tilde{E}$ with respect to B is

$$\frac{\partial \tilde{E}(A, B)}{\partial \operatorname{vec}(B)'} = 2[((A\Sigma_{xx}A' + \Sigma_{nn}) \otimes I)\operatorname{vec}(B) - \operatorname{vec}(\Sigma_{yx}A')]$$

and the corresponding Hessian is

$$\frac{\partial^2 \tilde{E}(A, B)}{\partial \operatorname{vec}(B)\partial \operatorname{vec}(B)'} = 2((A\Sigma_{xx}A' + \Sigma_{nn}) \otimes I)$$

from which the fact follows in the usual way.

Now, if $\Sigma_{nn} = \sigma R$, then for fixed A

$$\tilde{B}_\sigma(A) = \Sigma_{yx}A'(A\Sigma_{xx}A' + \sigma R)^{-1}$$

and by direct calculation

$$\begin{aligned}\min_B \tilde{E}_\sigma(A, B) &= \tilde{E}_\sigma(A, \tilde{B}_\sigma(A)) \\ &= \operatorname{trace}(\Sigma_{yy} + \Sigma_{ee}) \\ &\quad - \operatorname{trace}(\Sigma_{yx}A'(A\Sigma_{xx}A' + \sigma R)^{-1}A\Sigma_{xy}).\end{aligned}$$

Thus, if $\sigma \gg 1$, we find that

$$\begin{aligned}\min_B \tilde{E}_\sigma(A, B) &= \operatorname{trace}(\Sigma_{yy} + \Sigma_{ee}) \\ &- \sigma^{-1}\operatorname{trace}(\Sigma_{yx}A'R^{-1}A\Sigma_{xy}) + O(\sigma^{-2})\end{aligned}$$

uniformly over $\mathcal{A}$. Let

$$M = \Sigma_{xy}\Sigma_{yx}, \qquad \Phi(A) = \operatorname{trace}(MA'R^{-1}A).$$

If σ is very large, we might expect from the above that the solutions to the constrained noisy problem are "very close" to the set of elements of $\mathcal{A}$ which maximize $\Phi(A)$ over $\mathcal{A}$. More precisely, one has the following proposition.

Proposition 1: Suppose we choose, for all $\sigma > 0$, matrices $A_\sigma \in \mathcal{A}_\sigma$. Then

$$\lim_{\sigma \to \infty} \Phi(A_\sigma) = \phi = \max_{\mathcal{A}} \Phi(A).$$

Proof: Obviously, $\limsup_{\sigma\to\infty}\Phi(A_\sigma)\leq\phi$. Suppose this inequality were strict. Then there exist $\gamma>0$ and a subsequence $\sigma_k\to\infty$ as $k\to\infty$ such that, for all $k, \Phi(A_{\sigma_k})\leq\phi-\gamma$. Pick $A\in\mathcal{A}$ such that $\Phi(A)=\phi$. Then

$$\begin{aligned}\tilde{E}_{\sigma_k}(A_{\sigma_k},\tilde{B}_{\sigma_k}(A_{\sigma_k}))-\tilde{E}_{\sigma_k}(A,\tilde{B}_{\sigma_k}(A))\\ =\sigma_k^{-1}(\Phi(A)-\Phi(A_{\sigma_k}))+O(\sigma_k^{-2})\\ \geq\sigma_k^{-1}\gamma+O(\sigma_k^{-2})\end{aligned}$$

which would imply that $\liminf_k\sigma_k(\tilde{E}_{\sigma_k}(A_{\sigma_k},\ \tilde{B}_{\sigma_k}\ (A_{\sigma_k}))\ \tilde{E}_{\sigma_k}(A,\tilde{B}_{\sigma_k}(A)))>0$, which is impossible.

Hence, if we write $\mathcal{A}_\Phi=\{A\in\partial\mathcal{A}:\Phi(A)=\phi\}$, then in the foregoing sense, $\lim_{\sigma\to\infty}\mathcal{A}_\sigma=\mathcal{A}_\Phi$.

Proposition 2: Suppose that $\mathcal{A}=\{A:\operatorname{trace}(ASA')\leq\rho\}$ where S is symmetric and positive definite. Then

$$\mathcal{A}_\Phi=\{A:A=\sqrt{\rho}vw'S^{-1/2}\}$$

where v is a normalized principal eigenvector of R^{-1} and w is a normalized principal eigenvector of $S^{-1/2}MS^{-1/2}$. In particular, all such A are rank 1 matrices.

Proof: Let $C=AS^{1/2}\rho^{-1/2}$ such that $A=\sqrt{\rho}CS^{-1/2}$ and let $c=\operatorname{vec}(C)$. Then, by a simple calculation

$$\Phi(A)=\rho c'(S^{-1/2}MS^{-1/2}\otimes R^{-1})c$$

and

$$c'c=\operatorname{trace}(CC')=\rho^{-1}\operatorname{trace}(ASA')=1.$$

As a result

$$\mathcal{A}_\Phi=\{A:A=\sqrt{\rho}CS^{-1/2}\}$$

where $c=\operatorname{vec}(C)$ maximizes $c'(S^{-1/2}MS^{-1/2}\otimes R^{-1})c$ over $c'c=1$. But again, by the result on quadratic forms reviewed in Section II-A, we know that c is a normalized principal eigenvector of $S^{-1/2}MS^{-1/2}\otimes R^{-1}$. It can be seen that all such c's are given by $c=\operatorname{vec}(vw')$, where w is a normalized principal eigenvector of $S^{-1/2}MS^{-1/2}$ and v is a normalized principal eigenvector of R^{-1}, whence the proposition.

Apart from multiplicative constants, all rows of optimal A matrices in the above proposition are identical, and the network provides maximal redundancy. Several corollaries can be derived upon making more specific assumptions about the matrices M, R, and S. If $R\neq I_p$, the structure of the noise at the hidden layer is taken into account by suitable scaling of the rows of A. If $R=I_p$, we find that all p-dimensional unit-length vectors are normalized principal eigenvectors of R^{-1}, so in particular there is one optimal A with identical rows. If $S=I_n$, the corresponding w is a normalized principal eigenvector of M; in particular, in the autoassociative case, we find that $M=\Sigma_{xx}^2$ and therefore w is a principal eigenvector of Σ_{xx}. Finally, if the maximal eigenvalues of both $S^{-1/2}MS^{-1/2}$ and R^{-1} are simple, then the optimal A is uniquely determined.

A different class of sets $\mathcal{A}$ is considered in Linsker [9], namely

$$\mathcal{A}=\{A=[a_1,\cdots,a_p]':\|a_i\|\leq 1\}.$$

In this case, maximizing $\Phi(A)$ over $\mathcal{A}$ can be accomplished by solving the first order conditions of the Lagrangian

$$\Phi(A)+\sum_{i=1}^{p}\kappa_i(a_i'a_i-1).$$

This yields that any constrained optimal A satisfies the equation $R^{-1}AM=KA$, where $K=\operatorname{diag}(\kappa_1,\cdots,\kappa_p)$ is a diagonal matrix of Lagrange multipliers. If $R=I_p$, this simplifies to $AM=KA$ which means that the rows a_i' of A are left eigenvectors of M with eigenvalue κ_i. Hence, the optimal A matrices are the ones which have normalized principal eigenvectors of M as their rows.

Finally, if $R=I_p$ and $\mathcal{A}=\{A:AA'=I_p\}$, then maximizing $\Phi(A)$ over $\mathcal{A}$ amounts to maximizing $\operatorname{trace}(AMA')$ over $AA'=I_p$. By a consequence to the Poincaré separation theorem (cf. [4, p. 211, Theorem 13]), this is achieved if the rows of A are mutually orthogonal and span the space of the first p principal eigenvectors of M.

B. Analysis of the Landscape of the Validation Function

Let us study E^V under uniform conditions. We deal only with the case $\tilde{\nu}_i\geq\nu_i$ for every i (the case $\tilde{\nu}_i\leq\nu_i$ is similar).

If for every $i, \alpha_i(0)\geq\lambda_i/(\lambda_i+\nu_i)$ (case a)), then $b_i\leq 0, \beta_i\leq 0$ and $\gamma_i\leq 0$. Therefore, $dE^V/dk\leq 0$ and E^V decreases to its asymptotic value.

If for every $i, \lambda_i/(\lambda_i+\tilde{\nu}_i)\leq\alpha_i(0)\leq\lambda_i/(\lambda_i+\nu_i)$ (case b)), then $0\leq b_i\leq(\tilde{\nu}_i-\nu_i)/(\lambda_i+\tilde{\nu}_i), \beta_i\geq 0$, $\gamma_i\leq 0$ and $\beta_i+\gamma_i\geq 0$. Since a_i^{2k} decays to zero faster than a_i^k, $dE^V/dk\geq 0$ and E^V increases to its asymptotic value.

The most interesting case is c), where for every i, $\alpha_i(0)\leq\lambda_i/(\lambda_i+\tilde{\nu}_i)$, or equivalently, $b_i\geq(\tilde{\nu}_i-\nu_i)/(\lambda_i+\tilde{\nu}_i)$. Then $\beta_i\geq 0$, $\gamma_i\leq 0$, and $\beta_i+\gamma_i\leq 0$, so dE^V/dk is negative at the beginning and approaches zero from the positive side as $k\to\infty$. Strictly speaking, this is not satisfied if $\beta_i=0$. But this can occur only if $b_i=0$ or $\lambda_i=0$ (but then $\gamma_i=0$ also) or if $\nu_i=\tilde{\nu}_i$. For simplicity, let us add the assumption that $\nu_i\neq\tilde{\nu}_i$. A function which first increases (respectively, decreases) and then decreases (respectively, increases) with a unique maximum (respectively, minimum) is called unimodal. We need to show that E^V is unimodal. For this, we use induction on n combined with an analysis of the unimodality properties of the derivatives of any order of E^V. We actually prove the stronger result that the derivatives of all orders of E^V are unimodal and have a unique zero crossing.

For $p=1,2,\cdots$, define

$$F^{(p)}(k)=\frac{d^pE^V}{dk^p}.$$

Then

$$F^{(p)}(k)=\sum_i f_i^{(p)}(k)=\sum_i\beta_{i,p}a_i^k+\gamma_{i,p}a_i^{2k}$$

with $\beta_{i,p}=\beta_i(\log a_i)^{p-1}$ and $\gamma_{i,p}=\gamma_i(2\log a_i)^{p-1}$. Clearly, for any $p\geq 1$, we have $\operatorname{sign}(\beta_{i,p})=(-1)^{p+1}$, $\operatorname{sign}(\gamma_{i,p})=(-1)^p$, and $\operatorname{sign}(f_i^{(p)})(0)=\operatorname{sign}(\beta_{i,p}+\gamma_{i,p})=(-1)^p$. Therefore, $\operatorname{sign}(F^{(p)}(0))=(-1)^p$ and, as $k\to\infty$, $F^{(p)}(k)\to 0$ as $\sum_i\beta_{i,p}a_i^k$, thus with the sign of $\beta_{i,p}$ which is $(-1)^{p+1}$.

As a result, all the continuous functions $F^{(p)}$ must have at least one zero crossing. If $F^{(p)}$ is unimodal, then $F^{(p)}$ has a unique zero crossing. If $F^{(p+1)}$ has a unique zero crossing, then $F^{(p)}$ is unimodal. Thus if for some p_0, $F^{(p_0)}$ has a unique zero crossing, then all the functions $F^{(p)}$ $(1 \leq p < p_0)$ are unimodal and have a unique zero crossing. Therefore, E^V has a unique minimum if and only if there exists an index p such that $F^{(p)}$ has a unique zero crossing. By using induction on n, we are going to see that for p large enough this is always the case.

Before we start the induction, for any continuously differentiable function f defined over $[0, \infty)$, let

$$\text{zero}(f) = \inf\{x : f(x) = 0\}$$

and

$$\text{ext}(f) = \inf\left\{x : \frac{df}{dx}(x) = 0\right\}.$$

Most of the time, zero and ext will be applied to functions which have a unique zero or extremum. In particular, for any i and p, it is trivial to see that the functions $f_i^{(p)}$ are unimodal and with a unique zero crossing. A simple calculation gives

$$\text{zero}(f_i^{(p)}) = \frac{1}{\log a_i} \log \frac{-\beta_i}{2^{p-1}\gamma_i} \tag{51}$$

and

$$\text{ext}(f_i^{(p)}) = \text{zero}(f_i^{(p+1)}) = \frac{1}{\log a_i} \log \frac{-\beta_i}{2^p \gamma_i}. \tag{52}$$

Also notice that for any $p \geq 1$

$$\min_i \text{zero}(f_i^{(p)}) \leq \text{zero}(F^{(p)}) \leq \max_i \text{zero}(f_i^{(p)}) \tag{53}$$

$$\min_i \text{ext}(f_i^{(p)}) \leq \text{ext}(F^{(p)}) \leq \max_i \text{ext}(f_i^{(p)}). \tag{54}$$

(In fact, (53) and (54) are true for any zero crossing or extremum of $F^{(p)}$.)

We can now begin the induction. For $n = 1$, E^V trivially has a unique minimum and all its derivatives are unimodal with a unique zero crossing. Let us suppose that this is also true of any validation error function of $n - 1$ variables. Let $\lambda_1 \geq \cdots \geq \lambda_n > 0$ and consider the corresponding ordering induced on the variables $a_i = 1 - \eta\lambda_i - \eta\nu_i$, $1 > a_{i_1} \geq \cdots \geq a_{i_n} \geq 0$. Let i_j be a fixed index such that $a_{i_1} \geq a_{i_j} \geq a_{i_n}$ and write, for any $p \geq 1$, $F^{(p)}(k) = G^{(p)}(k) + f_{i_j}^{(p)}(k)$ with $G^{(p)}(k) = \sum_{i \neq i_j} f_i^{(p)}(k)$. The function $f_{i_j}^{(p)}$ is unimodal with a unique zero crossing and so is $G^{(p)}$ by the induction hypothesis. Now it is easy to see that $F^{(p)}$ will have a unique zero crossing if

$$\text{zero}(G^{(p)}) \leq \text{zero}(f_{i_j}^{(p)}) \leq \text{ext}(G^{(p)}).$$

By applying (53) and (54) to $G^{(p)}$, we see that $F^{(p)}$ has a unique zero crossing if

$$\max_{i \neq i_j} \text{zero}(f_i^{(p)}) \leq \text{zero}(f_{i_j}^{(p)}) \leq \min_{i \neq i_j} \text{ext}(f_i^{(p)}).$$

Substituting the values given by (51) and (52), we see that for large p, the above is equivalent to

$$\max_{i \neq i_j} -(p-1)\frac{\log 2}{\log a_i} \leq -(p-1)\frac{\log 2}{\log a_{i_j}} \leq \min_{i \neq i_j} -p\frac{\log 2}{\log a_i}$$

which is satisfied since $a_{i_1} \geq \cdots \geq a_{i_n}$. Therefore, using the induction hypothesis, we see that there exists an integer p_0 such that, for any $p > p_0$, $F^{(p)}$ has a unique zero crossing. But, as we have seen, this implies that $F^{(p)}$ has a unique zero crossing also for $1 \leq p \leq p_0$. Therefore E^V is unimodal with a unique minimum and its derivatives of all orders are unimodal with a unique zero crossing.

Notice that $F(k)$ cannot be zero if all the functions $f_i(k)$ are simultaneously negative or positive. Therefore, a simple bound on the position of the unique minimum k^{opt} is given by

$$\min_i \text{zero}(f_i) \leq \text{zero}(F) \leq \max_i \text{zero}(f_i)$$

or

$$\min_i \frac{1}{\log a_i} \log \frac{-\beta_i}{\gamma_i} \leq k^{\text{opt}} \leq \max_i \frac{1}{\log a_i} \log \frac{-\beta_i}{\gamma_i}.$$

(It is also possible, for instance, to study the effect of the initial $\alpha_i(0)$ on the position or the value of the local minima. By differentiating the relation $F^{(1)}(k) = 0$, one gets immediately

$$F^{(2)}(k)\,dk = \sum_i \left(\frac{\lambda_i + \nu_i}{\lambda_i b_i}\right)(\beta_i - i a_i^k + 2\gamma_i a_i^{2k})\,d\alpha_i(0)$$

see Fig. 2.)

To find an upper bound on the number of local minima of E^V in the general case of arbitrary noise and initial conditions, we first order the $2n$ numbers a_i and a_i^2 into an increasing sequence c_i, $i = 1, \cdots, 2n$. This induces a corresponding ordering on the $2n$ numbers β_i and γ_i yielding a second sequence C_i, $i = 1, \cdots, 2n$. Now the derivative of E^V can be written in the form

$$\frac{dE^V}{dk} = F^{(1)}(k) = \int C(a) a^k \, d\mu(a)$$

where μ is the finite positive measure concentrated at the points a_i and a_i^2. The kernel a^k in the integral is totally positive. Thus (see, for instance, [52, Theorem 3.1]), the number of sign changes of $F^{(1)}(k)$ is bounded by the number of sign changes in the sequence C. Therefore, the number of sign changes in $F^{(1)}$ is at most $2n - 1$ and the number of zeros of $F^{(1)}$ is at most $2n - 1$ and hence the number of local minima of E^V is at most n.

References

[1] W. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin Math. Biophysics*, vol. 7, pp. 115–133, 1943.

[2] D. A. Robinson, "The use of control systems analysis in the neurophysiology of eye movement," *Annu. Rev. Neurosci.*, vol. 4, pp. 463–503, 1981.

[3] T. Apostol, *Calculus* vol. II, 2nd ed. New York: Wiley, 1967.

[4] J. R. Magnus and H. Neudecker, *Matrix Differential Calculus with Applications in Statistics and Econometrics*. New York: Wiley, 1988.

[5] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, UK: Cambridge Univ. Press, 1985.

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

[7] R. E. Blahut, *Principles and Practice of Information Theory*. Reading, MA: Addison-Wesley, 1987.

[8] A. N. Kshirsagar, *Multivariate Analysis*. New York: Marcel Dekker, 1972.

[9] R. Linsker, "Self-organization in a perceptual network," *IEEE Comput. Mag.*, vol. 21, pp. 105–117, 1988.

[10] K. V. Mardia, J. T. Kent, and M. Bibby, *Multivariate Analysis*. New York: Academic, 1979.

[11] Z. Füredi and J. Komlós, "The eigenvalues of random symmetric matrices," *Combinatorica*, vol. 1, pp. 233–241, 1981.

[12] P. Gallinari, S. Thiria, and F. Fogelman Soulie, "Multilayer perceptrons and data analysis," in *Proc. 1988 ICNN Conf.* 1988, pp. 391–399.

[13] P. Baldi, "Linear learning: Landscapes and algorithms," in *Advances in Neural Information Processing Systems 1*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1989.

[14] R. Linsker, "From basic network principles to neural architecture: Emergence of spatial opponent cells," *Proc. National Academy Sci. USA*, vol. 38, pp. 7508–7512, 1986a.

[15] ——, "From basic network principles to neural architecture: Emergence of orientation selective cells," *Proc. Nat. Academy Sci. USA*, vol. 83, pp. 8390–8394, 1986b.

[16] ——, "From basic network principle to neural architecture: Emergence of orientation columns," *Proc. Nat. Academy Sci. USA*, vol. 83, pp. 8779–8783, 1986c.

[17] K. D. Miller, J. B. Keller, and M. P. Striker, "Ocular dominance column development: Analysis and simulation," *Sci.*, vol. 245, pp. 605–615, 1989.

[18] H. Bourlard and Y. Kamp, "Auto-association by the multilayer perceptrons and singular value decomposition," *Biol. Cybern.*, vol. 59, pp. 291–294, 1988.

[19] G. W. Cottrell, P. W. Munro, and D. Zipser, "Image compression by backpropagation: A demonstration of extensional programming," in *Advances in Cognitive Science 2*, N. E. Sharkey, Ed. Norwood, NJ: Abbex, 1988.

[20] R. J. Williams, "Feature discovery through error-correction learning," Institute for Cognitive Science, Univ. California, San Diego Tech. Rep. 8501, 1985.

[21] M. D. Plumbley, "Efficient information transfer and anti-Hebbian neural networks," *Neural Networks*, vol. 6, pp. 823–833, 1993.

[22] K. Diamantaras and K. Hornik, "Noisy principal component analysis," in *Proc. MEASUREMENT 93*, 1993.

[23] K. Hornik, "Noisy linear networks," in *Artificial Neural Networks for Speech and Vision*, R. Mammone, Ed. London: Chapman and Hall, pp. 37–44, 1993.

[24] P. Baldi and Y. Chauvin, "Temporal evolution of generalization during learning in linear networks," *Neural Computation*, vol. 3, pp. 589–603, 1991.

[25] E. B. Baum and D. Haussler, "What size net gives valid generalization?" *Neural Computation*, vol. 1, pp. 151–160, 1989.

[26] E. Levin, N. Tishby, and S. A. Solla, "A statistical approach to learning and generalization in layered neural networks." *Proc. IEEE*, vol. 78, pp. 1568–1574, 1990.

[27] H. Sompolinsky, N. Tishby, and H. S. Seung, "Learning from examples in large neural networks," *Physical Rev. Lett.*, vol. 65, pp. 1683–1686, 1990.

[28] A. Krogh and J. A. Hertz, "Generalization in a linear perceptron in the presence of noise," *J. Physics A*, vol. 25, pp. 1135–1147, 1992.

[29] C. Wang, S. Judd, and S. S. Venkatesh, "On optimal stopping time and effective machine size in learning," in *Advances in Neural Information Processing Systems 6*, J. D. Cowan, G. Tesauro and J. Alspector, Eds. San Francisco, CA: Morgan Kaufmann, 1994.

[30] B. Widrow and M. Hoff, "Adaptive switching circuits," in *Proc. 1960 WESCON Conv. Rec.: Part 4* 1960, pp. 96–104.

[31] F. Palmieri, J. Zhu, and C. Chang, "Anti-Hebbian learning in topologically constrained linear networks: a tutorial," *IEEE Trans. Neural Networks*, vol. 4, pp. 748–761, 1993.

[32] M. R. Garey and D. S. Johnson, *Computers and Intractability*. New York: H. Freeman and Company, 1979.

[33] E. Oja, "A simplified neuron model as a principal component analyzer," *J. Math. Biology*, vol. 15, pp. 267–273, 1982.

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

[35] K. Hornik and C.-M. Kuan, "Convergence analysis of local feature extraction algorithms," *Neural Networks*, vol. 5, pp. 229–240, 1992.

[36] C.-M. Kuan and K. Hornik, "Convergence of learning algorithms with constant learning rates," *IEEE Trans. Neural Networks*, vol. 2, pp. 484–488, 1991.

[37] D. J. C. MacKay and K. D. Miller, "Analysis of Linsker's simulation of Hebbian rules," *Neural Computation*, vol. 2, pp. 173–187, 1990.

[38] E. Oja, "Neural networks, principal components and subspaces," *Int. J. Neural Systems*, vol. 1, pp. 61–68, 1989.

[39] L. Xu and A. Yuille, "Self-extracting rules for robust PCA," *Advances in Neural Information Processing Systems 5*, S. J. Hanson, J. D. Cowan, and C. L. Giles, Eds. San Mateo, CA: Morgan Kaufmann, 1993, pp. 467–474.

[40] T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network," *Neural Networks*, vol. 2, pp. 459–473, 1989.

[41] E. Oja, "Principal components, minor components, and linear neural networks," *Neural Networks*, vol. 5, pp. 927–936, 1992.

[42] R. W. Brockett, "Dynamical systems that sort lists, diagonalize matrices and solve linear programming problems," *Linear Algebra Applicat.*, vol. 146, pp. 79–91, 1991.

[43] L. Xu, "Least mean square error reconstruction principle for self-organizing neural nets," *Neural Networks*, vol. 6, pp. 627–648, 1993.

[44] P. Földiák, "Adaptive network for optimal linear feature extraction," in *Proc. Joint Int. Conf. Neural Networks*, San Diego, CA, 1989, pp. 401–405.

[45] J. Rubner and P. Tavan, "A self-organizing network for principal component analysis," *Europhysics Lett.*, vol. 10, pp. 693–698, 1989.

[46] T. Leen, "Dynamics of learning in linear feature-discovery networks," *Network*, vol. 2, pp. 85–105, 1991.

[47] S. Y. Kung and K. Diamantaras, "A neural network for adaptive principal component extraction (APEX)," in *Proc. IEEE Int. Conf. Acoustics, Speech Signal Process.*, 1990, pp. 861–864.

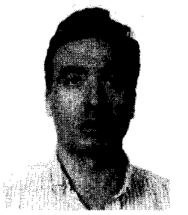
[48] H. B. Barlow and P. Földiák, "Adaptation and decorrelation in the cortex," in *The Computing Neuron*, C. Miall, R. M. Durbin, and G. J. Mitchison, Eds. New York: Addison-Wesley, 1989.

[49] Y. Chauvin, "Principal component analysis by gradient descent on a constrained linear Hebbian cell," in *Proc. Joint Int. Conf. Neural Networks*. San Diego, CA: 1989, pp. I: 373–380.

[50] R. Lenz and M. Osterberg, "Computing the Karhunen-Loève expansion with a parallel, unsupervised filter system," *Neural Computation*, vol. 4, pp. 382–392, 1992.

[51] A. Cichocki and R. Unbehauen, "Neural networks for computing eigenvalues and eigenvectors," *Biol. Cybern.*, vol. 68, pp. 155–164, 1992.

[52] S. Karlin, *Total Positivity*. Stanford, CA: Stanford Univ. Press, 1968.



Pierre F. Baldi was born in Rome, Italy. He received the Ph.D. degree from California Institute of Technology, Pasadena, in 1986.

He spent two years as a visiting lecturer at University of California-San Diego. Since 1988, Dr. Baldi has been a Member of the Technical Staff at the Jet Propulsion Laboratory at California Institute of Technology and a Visiting Research Associate at the Division of Biology at California Institute of Technology. His main research interests are in the theory and applications of intelligent computing, natural and artificial neural systems, and machine learning approaches in computational molecular biology.

Dr. Baldi is the co-founder of Net-ID, Inc., a neural network startup.

Kurt Hornik (M '92) was born in Vienna, Austria. He received the M.Sc. and Ph.D. degrees in applied mathematics in 1985 and 1987, respectively, both from the Technische Universität Wien of Vienna, Austria.

In the academic year 1987–88 he was Visiting Assistant Professor with the Department of Mathematics at the University of California, San Diego. Currently, he is Associate Professor with the Department of Statistics and Probability Theory of the Technische Universität Wien, Vienna, Austria. His current research interests include computational intelligence, time series analysis, statistics, econometrics, adaptive feature extraction algorithms, and pattern recognition using neural networks.

Dr. Hornik is Letters Editor for IEEE TRANSACTIONS ON NEURAL NETWORKS.