On Self-Organizing Algorithms and Networks for Class-Separability Features

Chanchal Chatterjee and Vwani P. Roychowdhury

***Abstract*—We describe self-organizing learning algorithms and associated neural networks to extract features that are effective for preserving *class separability*. As a first step, an adaptive algorithm for the computation of $Q^{-1/2}$ (where Q is the correlation or covariance matrix of a random vector sequence) is described. Convergence of this algorithm with probability one is proven by using stochastic approximation theory, and a single-layer linear network architecture for this algorithm is described, which we call the $Q^{-1/2}$ network. Using this network, we describe feature extraction architectures for: 1) unimodal and multicluster Gaussian data in the multiclass case; 2) multivariate linear discriminant analysis (LDA) in the multiclass case; and 3) Bhattacharyya distance measure for the two-class case. The LDA and Bhattacharyya distance features are extracted by concatenating the $Q^{-1/2}$ network with a principal component analysis (PCA) network, and the two-layer network is proven to converge with probability one. Every network discussed in the study considers a flow or sequence of inputs for training, thereby eliminating the need for a pooled data for training, and making the networks useful for online applications. Furthermore, the training of all layers of the networks can proceed simultaneously. Numerical studies on the performance of the networks for multiclass random data are presented.**

***Index Terms*—Adaptive learning, feature extraction networks, linear discriminant analysis networks.**

I. Introduction

In this study, we present self-organizing learning algorithms and associated neural networks to extract features that are effective for preserving *class separability*. In particular, we describe algorithms and networks for the following three features that are commonly studied in the pattern recognition literature [10]: 1) features for unimodal and multicluster Gaussian data in the multiclass case; 2) multivariate linear discriminant analysis in the multiclass case; and 3) features from the Bhattacharyya distance measure for the two-class case.

The features obtained from the above networks are entirely different from features that are used for data representation [9], [10]. In the literature for feature extraction with self-organizing neural networks, there are numerous algorithms that are optimal with respect to the representation of data. These include the principal component analysis (PCA) algorithms [8], [16], [22], [23], [27], [28], [30] that preserve the variance of the data, and Sammon's algorithm [15] that preserves the interpattern distances. Unlike the criteria for data representation, class-separability criteria are independent of coordinate systems, and depend on the class distributions, and the classifier used. Fig. 1 shows a two-class two-dimensional (2-D) classification problem in which the best feature for data representation is the projection on $\mathbf{e}_1$. This results in poor classification compared to the best feature for class separability which is the projection on $\mathbf{e}_2$.

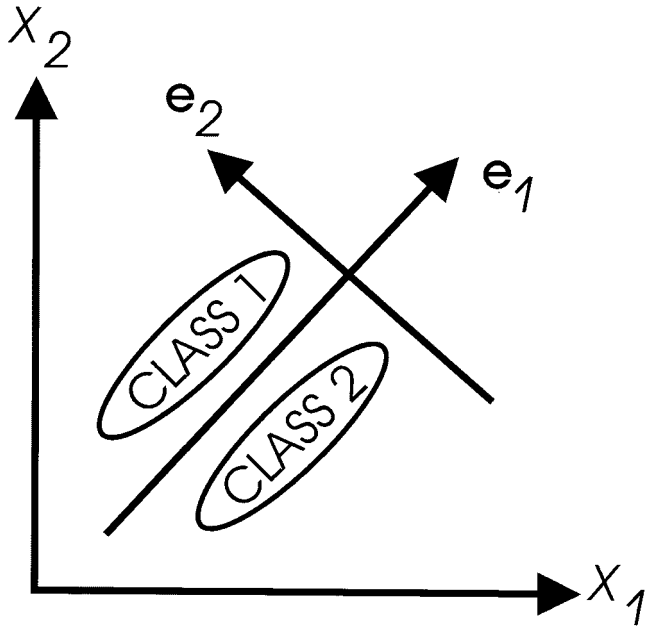

Fig. 1. Illustration of data representation feature $\mathbf{e}_1$ and class-separability feature $\mathbf{e}_2$.

A review of the state of the art feature extraction networks is given in Section I-B.

A. Algorithms and Architectures Discussed in this Study

In Section II, we show that all feature extraction tasks considered in this study involve the computation of $Q^{-1/2}$, where Q is the positive definite correlation matrix of a random vector sequence. Since we are training the networks with a sequence of data, we require an online (i.e., adaptive) algorithm for this computation. Hence, we first present a novel algorithm and network for the adaptive computation of $Q^{-1/2}$ from a sequence of data, and then we outline the three feature extraction networks discussed in this study.

Manuscript received December 8, 1995; revised July 12, 1996 and December 1, 1996. This work was supported in part by NSF Grants ECS-9308814 and ECS-9523423.

C. Chatterjee is with Newport Corporation, Irvine, CA 92606 USA.

V. P. Roychowdhury was with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-1285 USA. He is now with the Electrical Engineering Department, University of California, Los Angeles, CA 90095 USA.

Publisher Item Identifier S 1045-9227(97)02759-8.

$Q^{-1/2}$ Algorithm and Network: Given a sequence $\{\mathbf{x}(k)\}$ of random vectors, where $\lim_{k\to\infty} E[\mathbf{x}(k)\mathbf{x}(k)^T] = Q$, we provide a stochastic approximation algorithm to compute $Q^{-1/2}$. We prove that starting from an initial random set of weights, this algorithm converges with probability one to $Q^{-1/2}$. We also describe a single-layer linear network implementation of this algorithm, which is we call the $Q^{-1/2}$ *network*. Numerical simulations for this network as well as its applications are described in Section V.

Feature Extraction Algorithms and Networks: Using the $Q^{-1/2}$ algorithm and network discussed above, we derive three feature extraction networks and their training algorithms. First, the optimal features for *unimodal and multicluster Gaussian data* are extracted using a two-layer feedforward network. The first layer is a linear network trained by the $Q^{-1/2}$ algorithm, and the second layer is a quadratic network with constant weights, and does not require training.

Next, the LDA and *Bhattacharyya distance* features are extracted by a two-layer feedforward network where both layers are linear and require training. The first layer is trained by the $Q^{-1/2}$ algorithm, and the second layer by an adaptive eigenvector computation algorithm. We have used the so called "Sanger's rule" [13], [30] for this algorithm. The combined training procedure is proven to converge with probability one to the desired features.

An added advantage of these training algorithms is that they are adaptive, and thus: 1) they are well-suited for online applications and 2) they provide good architectures which are relatively easily implemented using current VLSI technologies. Since each algorithm uses a flow or sequence of inputs, we do not require a pooled data for network training. Besides the computational advantages, the methods have the added benefit of requiring less storage. Furthermore, all layers of the networks are trained *simultaneously*.

Note that, in this study, we emphasize the methods of feature extraction, and not the particulars of classification after these features are extracted. Since it is well known [26], [29], [31] that neural networks are useful in classification, a separate network may be used for this task. For example, classification of a unimodal Gaussian feature may be obtained by using a proper threshold and a minimum selector. The LDA and Bhattacharyya distance features may be classified by a linear classifier [10], and, hence, can be implemented by a single-layer feedforward network. In this study, the classifiers are used only to demonstrate that the networks estimate the features accurately, and preserve the classifying abilities of the actual features. We, therefore, use simple classifiers that are commonly considered in pattern recognition [10] to test our feature extraction methods.

B. State-of-the-Art Feature Extraction Networks

Self-Organized Networks for LDA: Self-organized methods to train feature extraction networks with data from known classes are recent. Mao and Jain [20] have proposed a two-layer network for LDA, where each layer is a PCA network due to Rubner *et al.* [27], [28]. Although the method is useful, there are a few problems. The first layer of the network has to be trained fully, till convergence is obtained, before the training of the second layer can begin. Thus, in its current formulation, the training of the two layers can not proceed simultaneously, which is impractical when the training input is a flow or sequence of data. The method requires a pooled data for network training, which may be unrealistic in many real-time environments. Furthermore, the class and mixture means of the data are assumed to be known in advance.

The new method overcomes these limitations. Due to the adaptive nature of this algorithm, we train both layers simultaneously, and do not require *a priori* knowledge of the class means. Instead, we estimate the class means during the learning process. In summary, we suggest a more direct solution to the LDA problem, that is adaptive in nature, and applicable to a sequence of inputs.

LVQ and ART Networks: Since class separation is the main criterion, unlike traditional self-organized methods, the networks discussed in this study require training patterns from known classes. This is different from competitive learning methods such as learning vector quantization (LVQ) and self-organizing map [13], [14], and the adaptive resonance theory (ART) models [13], [14]. Although LVQ also uses data from known classes, both LVQ and ART models are primarily designed for clustering and data compression. The updating procedures in these algorithms have been related to partitional clustering approaches in pattern recognition [21], [25], which are very different from the feature extraction methods described here. For example, LVQ has been linked to the sequential k-means algorithm [25], and the ART models to the sequential leader clustering algorithm [21].

Supervised Networks for Feature Extraction: The feature extraction ability of a supervised multilayer feedforward network is well studied [11], [26], [29], [31], [32]. For example, Gallinari *et al.* [11] studied a linear multilayer perceptron performing a heteroassociative mapping. This work shows that, for a linear network performing a one-of-m (one output unity, all others zero) classification, the solution of the weights which minimized the total mean square output error also maximized a criterion for linear discriminant analysis. Webb and Lowe [32] extended this result to a multilayer feedforward network that performs an arbitrary nonlinear transformation to a space spanned by the hidden units, and finally executes a linear transform in the output layer. It is apparent from this study that a nonlinear discriminant analysis is performed in the space of the hidden units by the minimization of the output error. Studies [2], [5] in the autoassociative case shows that the outputs from the hidden layer are the principal components of the input vector. The Gaussian feature extraction network has also been studied [35], and shown to be isomorphic to a one-layer sigma-pi back propagation network having an increasing activation function.

In spite of these useful results for multilayer feedforward networks, we note that they are valid only if proper convergence is obtained, which is not guaranteed because the energy surface may have multiple local minima. Furthermore, the minimization may be inaccurate due to incorrect network size. Using the gradient descent training procedure for minimization

has the added problem that it is often slow to converge, and it does not impose any particular structure on the network architecture or on the solution. Moreover, a feedforward network, trained as above, may not satisfy the optimality criterion for the LDA problem as described in Section II-B. A nonsingular transformation of the outputs will, however, satisfy this criterion.

Motivated by the preceding issues, several researchers have designed networks where each layer is trained only with data from the previous layer [13], [14], [30]. Such networks are usually studied under the general category of unsupervised networks. In these networks, the weights in each layer are trained not by comparing the final outputs with a known reference, but by the statistical properties of the outputs of that layer. The networks discussed in this study are in this category. The training algorithm for each layer is proven to converge with probability one to the desired feature. Furthermore, the optimality criterion (see Section II-B) for the LDA network is satisfied.

In Section II we provide the pattern recognition theory for the three feature extraction methods discussed above. In Section III, we present the training algorithms used in the feature extraction networks, including the $Q^{-1/2}$ algorithm and its proof of convergence. Section IV discusses the network models for the $Q^{-1/2}$ algorithm, and the three feature extraction methods. Section V has the numerical simulation results on multiclass multivariate random data. Section VI has the concluding remarks.

II. Feature Extraction Methods

In this section, we shall review the pattern recognition theory for the three feature extraction methods implemented in this study. Readers familiar with pattern recognition theory related to these methods can skip this section. Since, in general, feature extraction criteria depend on the classifier to be used, we shall simplify the problem by assuming that we seek the optimum feature set with reference to the Bayes classifier.

Let us consider a finite m-set of pattern classes $\{\omega_1, \cdots, \omega_m\}$ with a priori probabilities $P(\omega_i), i = 1, \cdots, m$, conditional probability density $p(\mathbf{x}/\omega_i), i = 1, \cdots, m$, and *a posteriori* probabilities $p(\omega_i/\mathbf{x}), \ i = 1, \cdots, m$. Let $\mathbf{x} \in R^d$ be a pattern vector whose mixture distribution is given by $p(\mathbf{x})$. In this study, we assume that $m \leq d$. The m *a posteriori* probability functions, mentioned above, are sufficient statistic, and carries all information for classification in the Bayes sense. Since the *a posteriori* probabilities sum to one, only $m - 1$ of these m functions are linearly independent, and are the ideal features for classification. The Bayes classifier in this feature space is a piecewise bisector classifier [10] which is its simplest form.

A. Optimal Features from Gaussian Data

Consider the feature $\ln p(\omega_i/\mathbf{x}) = \ln p(\mathbf{x}/\omega_i) + \ln P(\omega_i) - \ln p(\mathbf{x})$ for class $\omega_i, \ i = 1, \cdots, m$. Since in feature extraction, additive and multiplicative constants do not alter the subspace onto which distributions are mapped [10], and since $\ln p(\mathbf{x})$ is common for all classes, $\ln p(\mathbf{x}/\omega_i)$ is the relevant feature for ω_i. When the distribution is unimodal Gaussian, this feature reduces to the following quadratic function:

$$f_i(\mathbf{x}) = (\mathbf{x} - \mathbf{m}_i)^T Q_i^{-1} (\mathbf{x} - \mathbf{m}_i) \quad \text{for } i = 1, \cdots, m \tag{1}$$

where $\mathbf{m}_i$ is the class mean and Q_i is the class covariance for ω_i. From (1), we obtain the optimum feature for ω_i as

$$f_i(\mathbf{x}) = \left\| Q_i^{-1/2} (\mathbf{x} - \mathbf{m}_i) \right\|^2 \quad \text{for } i = 1, \cdots, m. \tag{2}$$

Note that for Gaussian data, $f_i(\mathbf{x})$ is the sufficient statistic for classification with minimum Bayes error. In addition to Gaussian distributions, $f_i(\mathbf{x})$ is a sufficient statistic for a wide class of unimodal symmetric distributions, as indicated in the Proposition below.

Proposition: For a two-class problem, define a feature $h(\mathbf{x}) = f_1(\mathbf{x}) - f_2(\mathbf{x})$. Then regardless of distribution of $\mathbf{x}$ we have

$$E[h(\mathbf{x}) \,|\, \omega_1] \leq \ln \frac{\det[Q_2]}{\det[Q_1]} \quad \text{and}$$
$$E[h(\mathbf{x}) \,|\, \omega_2] \geq \ln \frac{\det[Q_2]}{\det[Q_1]}. \tag{3}$$

The proof of the Proposition can be derived from a result in [6]. Equation (3) shows that the class means in the feature space may be separable by a threshold $t = \ln(\det[Q_2]/\det[Q_1])$. Thus, regardless of distribution of $\mathbf{x}$, $f_i(\mathbf{x})$ can have a significant classifying ability. Our simulations with uniform distributions corroborate this fact (see Section V).

If the data is not Gaussian, we may be able to convert it to a Gaussian-like distribution by using nonlinear transforms. Fukunaga [10] uses the *power transform* x^ν for $0 < \nu < 1$ on radar signals from a gamma distribution, and converts it to a Gaussian-like one.

If the data is multicluster Gaussian with r_i clusters in ω_i, then the feature f_{ij} for cluster j in ω_i is defined as

$$f_{ij}(\mathbf{x}) = \exp(-h_{ij}(\mathbf{x}))$$

where

$$h_{ij}(\mathbf{x}) = 1/2 \left\| Q_{ij}^{-1/2} (\mathbf{x} - \mathbf{m}_{ij}) \right\|^2 \quad \text{for } j = 1, \cdots, r_i, \ i = 1, \cdots, m. \tag{4}$$

Here $\mathbf{m}_{ij}$ and Q_{ij} are the cluster mean and covariance, respectively.

B. Linear Discriminant Analysis

The linear discriminant analysis criteria, although related to the Bayesian risk analysis [7], are mainly based upon a family of functions of scatter matrices. A within-class scatter matrix is the scatter of the samples around their respective class means $\mathbf{m}_i$, and given by

$$S_w = \sum_{i=1}^{m} P(\omega_i) E\left[(\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T \mid \omega_i \right] = \sum_{i=1}^{m} P(\omega_i) Q_i. \tag{5}$$

The between-class scatter matrix is the scatter of the class means $\mathbf{m}_i$ around the mixture mean $\mathbf{m}_0$, and is given by

$$S_b = \sum_{i=1}^{m} P(\omega_i)(\mathbf{m}_i - \mathbf{m}_0)(\mathbf{m}_i - \mathbf{m}_0)^T. \tag{6}$$

Finally, the mixture scatter matrix is the covariance of all samples regardless of class assignments

$$S_m = E\big[(\mathbf{x}-\mathbf{m}_0)(\mathbf{x}-\mathbf{m}_0)^T\big] = S_w + S_b. \tag{7}$$

There are several criteria [10] for feature extraction as functions of these scatter matrices. Some examples are $\mathrm{tr}[S_w^{-1}S_b]$ and $\det[S_b]/\det[S_w]$ that we wish to maximize. In the linear discriminant analysis case, the optimum linear transform Φ is composed of the $p(\leq d)$ eigenvectors of $S_w^{-1}S_b$ corresponding to its p largest eigenvalues. Since S_b is not full-rank and not a covariance matrix, we shall use S_m instead. A simple analysis shows that both $S_w^{-1}S_b$ and $S_w^{-1}S_m$ have the same eigenvector matrix Φ, and eigenvalue matrices Λ and $\Lambda+I$, respectively. Thus, $S_w^{-1}S_b$ and $S_w^{-1}S_m$ extract the same features, and the order of significance of the features are the same.

The computation of the eigenvector matrix Φ of $S_w^{-1}S_m$ is equivalent to the solution of the generalized eigenvalue problem $S_m\Phi = S_w\Phi\Lambda$ where Λ is the generalized eigenvalue matrix. Under the assumption of a positive definite matrix S_w, there exists a symmetric $S_w^{-1/2}$ such that the problem can be reduced to a symmetric eigenvalue problem $S_w^{-1/2}S_mS_w^{-1/2}\Psi = \Psi\Lambda$, where $\Psi = S_w^{1/2}\Phi$ is real, and $S_w^{-1/2}S_mS_w^{-1/2}$ is symmetric and real. If Ψ is orthonormal, then $\Psi^T\Psi = \Phi^TS_w\Phi = I$. Thus, Φ is real and orthonormal with respect to S_w. Furthermore, $\Phi^TS_m\Phi = \Lambda$ which is diagonal, real and positive definite. This solution is also known as the simultaneous diagonalization [10] of matrices S_w and S_m.

Optimality Criterion for the LDA Network: Since the LDA problem consists of finding the $p(\leq d)$ generalized eigenvectors of S_m with respect to S_w corresponding to its p largest generalized eigenvalues, we need to find network weights that satisfy the following two conditions: 1) they are the generalized eigenvectors of S_m and S_w and 2) they are arranged in the decreasing order of significance. We shall refer to this as the "*optimality criterion*" for the LDA network. Once this criterion is satisfied by a network, the significant components (for class separability) of the inputs are represented by the first few components of the outputs.

C. Bhattacharyya Distance Measure

By the two previous criteria, we obtain a maximum of $m-1$ features for an m-class problem. However, unless the *a posteriori* probability functions are selected as features, $m-1$ features are suboptimal in the Bayes sense. Several approaches [6], [9], [10], [24] have been proposed to solve this problem. One approach is to break up the data into more classes, thereby artificially increasing the rank of S_b. In another approach, Foley and Sammon [9] used the generalized Fisher criterion to generate an orthonormal set of feature vectors in a two-class case. Later Okada and Tomita [24] extended this method to a multiclass case.

In alternative approaches we consider that the scatter criterion extracts features in an $m-1$ dimensional subspace containing all classification information due to the scatter of mean vectors. The remaining $d-m+1$ dimensional subspace contains the information due to covariance differences. To select features in this subspace, the scatter criterion is no longer useful, and the Bhattacharyya distance criterion may be used. This is a convenient measure of class separability since it gives upper bound of the Bayes error. However, this criterion is only useful for a two-class Gaussian data.

For Gaussian distributions with means $\mathbf{m}_1$ and $\mathbf{m}_2$ and covariances Q_1 and Q_2, for the two classes, respectively, the Bhattacharyya distance μ is [10]

$$\mu = \frac{1}{8}(\mathbf{m}_1-\mathbf{m}_2)^T\left(\frac{Q_1+Q_2}{2}\right)^{-1}(\mathbf{m}_1-\mathbf{m}_2) + \frac{1}{2}\ln\frac{\det[(Q_1+Q_2)/2]}{\sqrt{\det[Q_1]}\sqrt{\det[Q_2]}}. \tag{8}$$

Only the covariance part; i.e., the second term of (8) is used. We need to maximize μ_2 defined as

$$\mu_2 = \ln\frac{\det[(Q_1+Q_2)/2]}{\sqrt{\det[Q_1]}\sqrt{\det[Q_2]}} = \frac{1}{2}\big[\ln\big(\det\big[Q_1^{-1}Q_2+Q_2^{-1}Q_1+2I\big]\big) - d\ln 4\big] \tag{9}$$

where d is the dimension of Q_1 and Q_2. The optimum linear transform [10] Φ is composed of the $p(\leq d)$ eigenvectors of $Q_1^{-1}Q_2$ (assuming Q_1 is positive definite) corresponding to the p largest $\theta_j = \lambda_j + (1/\lambda_j)$, where $\lambda_j,\ j=1,\cdots,d$, are the eigenvalues of $Q_1^{-1}Q_2$. As before, the eigenvectors of $Q_1^{-1}Q_2$ are obtained from a symmetric eigenvalue problem $Q_1^{-1/2}Q_2Q_1^{-1/2}\Psi = \Psi\Lambda$, where $\Psi = Q_1^{1/2}\Phi$ and $\Lambda = \mathrm{diag}(\lambda_1,\cdots,\lambda_d)$. This method is very effective in picking features along which the variances of two classes are different.

III. Training Algorithms for the Feature Extraction Networks

There are two major training algorithms used in the feature extraction networks. They are: 1) the new algorithm for the computation of $Q^{-1/2}$, where Q is the positive definite correlation matrix of a random sequence $\{\mathbf{x}(k)\in R^d\}$ and 2) an algorithm for the computation of the eigenvectors of Q.

Note that there is no unique solution for $Q^{-1/2}$. Let Φ and $\Lambda = \mathrm{diag}(\lambda_1,\cdots,\lambda_d)$ be the eigenvector and eigenvalue matrices, respectively, of Q. Then a solution for $Q^{-1/2}$ is ΦD, where $D = \mathrm{diag}(\pm\lambda_1^{-1/2},\cdots,\pm\lambda_d^{-1/2})$. However, in general this is not a symmetric solution, and for any orthonormal matrix R, ΦDR is also a solution. One can show that $Q^{-1/2}$ is symmetric if and only if it is of the form $\Phi D\Phi^T$, and there are 2^d symmetric solutions for $Q^{-1/2}$. When D is positive definite, we obtain the unique symmetric positive definite solution for $Q^{-1/2}$ as $\Phi\Lambda^{-1/2}\Phi^T$, where $\Lambda^{-1/2} \triangleq \mathrm{diag}(\lambda_1^{-1/2},\cdots,\lambda_d^{-1/2})$.

A. Adaptive Computation of $Q^{-1/2}$ and a Stochastic Approximation Proof

The algorithm for the adaptive computation of $Q^{-1/2}$ is

$$W(k+1) = W(k) + \eta(k)(I - W(k)\mathbf{x}(k)\mathbf{x}(k)^TW(k)) \tag{10}$$

where $W(0)\in R^{d\times d}$ is symmetric and nonnegative definite, and $\{\eta(k)\}$ is a scalar gain sequence. We shall prove that

$W(k) \rightarrow Q^{-1/2}$ with probability one (w.p.1) as $k \rightarrow \infty$, where $Q^{-1/2} = \Phi\Lambda^{-1/2}\Phi^T$ is the unique symmetric positive definite solution discussed above. The algorithm for $W(0)$ nonpositive definite is considered later.

In order to prove the convergence of (10), we shall use the stochastic approximation theory due to Ljung [3], [18], [19]. An alternative proof by a similar approach due to Kushner and Clark [17] can also be given. In a somewhat looser language, stochastic approximation theory states the following: 1) $W(k)$ can converge only to stable stationary points of the ordinary differential equation (ODE) $dW/dt = \lim_{k\rightarrow\infty} E[I - W\mathbf{x}(k)\mathbf{x}(k)^T W]$; 2) if $W(k)$ belongs to the domain of attraction of a stable stationary point W^* of the ODE infinitely often (i.o.) w.p.1, then $W(k)$ converges w.p.1 to W^* as $k \rightarrow \infty$; and 3) the trajectories of the ODE are the "asymptotic paths" of $W(k)$ generated by (10). The proof requires the following steps: 1) generating a set of assumptions; 2) finding the stable stationary points of the ODE; and 3) an analysis showing that $W(k)$ visits the domain of attraction of a stable stationary point infinitely often.

Assumptions and Formulation of the ODE: In order to prove the convergence of (10), we shall use Theorem 1 of Ljung [18]. The following is a general set of assumptions for the convergence proof of algorithm (10).

Assumption (A1): The training sequence $\{\mathbf{x}(k) \in R^d\}$ consists of random vectors, where each $\mathbf{x}(k)$ is uniformly bounded; i.e., $\|\mathbf{x}(k)\| < \beta < \infty$. Moreover, $\lim_{k\rightarrow\infty} E[\mathbf{x}(k)\mathbf{x}(k)^T] = Q$ where Q is positive definite.

Assumption (A2): $\{\eta(k) \in R^+\}$ is a decreasing sequence such that $\sum_{k=0}^{\infty} \eta(k) = \infty$, $\sum_{k=0}^{\infty} \eta(k)^r < \infty$ for some $r > 1$, and $\lim_{k\rightarrow\infty} \sup(\eta(k)^{-1} - \eta(k-1)^{-1}) < \infty$.

Assumption A1 is reasonable for most practical implementations, where $\{\mathbf{x}(k)\}$ are kept bounded either by deliberate measures or automatically. Methods to keep $\{\mathbf{x}(k)\}$ bounded are discussed in Ljung [18]. The physical meaning of A2 can be described as follows. Condition $\eta(k) \rightarrow 0$ allows the process to settle down in the limit, whereas $\Sigma\eta(k) = \infty$ insures that there is enough corrective action to avoid stopping short of the solution. Conditions $\Sigma\eta(k)^p < \infty$ and $\lim_{k\rightarrow\infty} \sup(\eta(k)^{-1} - \eta(k-1)^{-1}) < \infty$ guarantee that the variance of the accumulated noise is finite so that we can correct for the effect of noise. Assumption A2 holds with $\eta(k) \propto k^{-\delta}$ for $0 < \delta \leq 1$. The choice of $\delta = 1$ is a leading case.

In the literature for stochastic approximation proofs, there are many assumptions that are usually made on the statistical properties of $\{\mathbf{x}(k)\}$, such as statistically independent and i.i.d. However, Ljung [3], [18] permits far less restrictive choices for $\{\mathbf{x}(k)\}$. Specifically, we can assume that $\{\mathbf{x}(k)\}$ is generated by a linear structure

$$\begin{aligned} \tilde{\mathbf{x}}(k) &= A(W(k))\tilde{\mathbf{x}}(k-1) + B(W(k))\mathbf{e}(k) \\ \mathbf{x}(k) &= C(W(k))\tilde{\mathbf{x}}(k), \end{aligned} \tag{11}$$

or a nonlinear variant

$$\begin{aligned} \tilde{\mathbf{x}}(k) &= g(k, W(k),\ \tilde{\mathbf{x}}(k-1),\ \mathbf{e}(k)) \\ \mathbf{x}(k) &= h(k, W(k), \tilde{\mathbf{x}}(k)) \end{aligned} \tag{12}$$

can also be postulated [18]. Here $\{\mathbf{e}(k)\}$ is a uniformly bounded sequence of independent (not necessarily stationary or with zero means) random vectors. These structures are treated at length in [3], [18], and [19]. In light of these models, we state the following general assumption for $\{\mathbf{x}(k)\}$.

Assumption (A3): Sequence $\{\mathbf{x}(k)\}$ is generated by (11) or (12) satisfying the stability and regularity conditions of Ljung [18].

The main assumptions of Theorem 1 of Ljung [18] are:

L1. The function $h(W, \mathbf{x}) \triangleq I - W\mathbf{x}\mathbf{x}^T W$ is continuously differentiable with respect to W and $\mathbf{x}$. The derivatives are, for fixed W and $\mathbf{x}$, bounded in k.

L2. $\bar{h}(W) \triangleq \lim_{k\rightarrow\infty} E[h(W, \mathbf{x}(k))]$ exists. The expectation is over the distribution of $\mathbf{e}(.)$.

We modify the result given by Ljung [18] in Theorem 1 to suit the present algorithm in the following Lemma.

Lemma 1: Let A1–A3 hold. Let W^* be a locally asymptotically stable (in the sense of Lyapunov) solution for the ODE

$$\frac{dW}{dt} = I - WQW \tag{13}$$

with domain of attraction $D(W^*)$. If there is a compact subset $A \subset D(W^*)$ such that $W(k) \in A$ infinitely often, then we have $W(k) \rightarrow W^*$ with probability one as $k \rightarrow \infty$.

Proof: We shall show that we satisfy conditions L1 and L2 of Ljung. Clearly $h(W, \mathbf{x}) = I - W\mathbf{x}\mathbf{x}^T W$ is continuously differentiable with respect to W and $\mathbf{x}$ satisfying $L1$. Sequence $\{\mathbf{x}(k)\}$ is bounded by A1. The boundedness of $W(k)$ is proven in Lemmas 4 and 5 later. Thus, the derivatives are bounded in k. Condition L2 follows from A1. The result is a direct application of Theorem 1 of Ljung [18]. □

Solution of the ODE:

Lemma 2: There exists a unique solution $W(t, W_0)$ of the ODE (13) which satisfies a given initial condition $W(0, W_0) = W_0 \in R^{d\times d}$. Moreover, the solution depends continuously on t and W_0, and for any $W_1 \in R^{d\times d}$ if $W_0 \rightarrow W_1$ then $W(t, W_0) \rightarrow W(t, W_1)$ uniformly over t.

Proof: Let $B(W_0; r) \triangleq \{W : \|W - W_0\|_2 < r\}$ be a open ball with center W_0 and constant but arbitrary radius $r > 0$. Let $\bar{h}(W) = I - WQW$. We claim that $\bar{h}(W)$ satisfies a Lipschitz condition[1] in $B(W_0; r)$. For any $W' \in B(W_0; r)$ and $W'' \in B(W_0; r)$, we have

$$\begin{aligned} &\|\bar{h}(W') - \bar{h}(W'')\|_2 \\ &\quad = \|(W' - W'')Q(W' - W'') + W''Q(W' - W'') \\ &\qquad + (W' - W'')QW''\|_2 \\ &\quad \leq \|W' - W''\|_2[\|Q\|_2(\|W'\|_2 + 3\|W''\|_2)] \\ &\quad \leq L\|W' - W''\|_2 \end{aligned}$$

where $L = 4r\lambda_{\max}(Q)$, and $\lambda_{\max}(Q)$ is the largest eigenvalue of Q, which exists due to A1. This proves the claim.

By the uniqueness theorem [4] for the ODE (13), there is at most one solution $W(t, W_0)$ which satisfies a given initial condition $W(0, W_0) = W_0$. By the Continuity Theorem [4],

[1] A function h defined on a domain B is said to satisfy a Lipschitz condition, if there is a constant L such that $\|h(\mathbf{x}) - h(\mathbf{y})\|_2 \leq L\|\mathbf{x} - \mathbf{y}\|_2$ for all $\mathbf{x}, \mathbf{y} \in B$.

for any $W_1 \in B(W_0; r)$ if $\|W_1 - W_0\|_2 \leq r' < r$, then if $W_0 \to W_1$ then $W(t, W_0) \to W(t, W_1)$ uniformly over t. □

In the following analyses, we shall denote $\lambda_1(W)$ as the largest eigenvalue of $W, \lambda_d(W)$ as the smallest eigenvalue of W, and $(\lambda_1(Q), \cdots, \lambda_d(Q))$ as the eigenvalues of Q in decreasing order.

Lemma 3: Let Φ and Λ, respectively, be the eigenvector and eigenvalue matrices of Q, and let $\Lambda^{-1/2}$ be positive definite, then the point $W_+^* = \Phi\Lambda^{-1/2}\Phi^T$ is (*uniformly*) asymptotically stable. The domain of attraction of W_+^* includes $D(W_+^*) = \{W \in R^{d\times d} : W = W^T$ and $\lambda_d(W) > -\lambda_1(Q)^{-1/2}\}$.

Proof: Note that for $W(0)$ symmetric, $W(k)$ is symmetric for all k. Then the ODE (13) is a Riccati differential equation [1] with Hamiltonian matrix H shown below. An eigen decomposition of H is

$$H = \begin{bmatrix} 0 & Q \\ I & 0 \end{bmatrix} = \begin{bmatrix} \Phi\Lambda^{1/2} & \Phi\Lambda^{1/2} \\ \Phi & -\Phi \end{bmatrix} \begin{bmatrix} \Lambda^{1/2} & 0 \\ 0 & -\Lambda^{1/2} \end{bmatrix} \begin{bmatrix} 1/2\Lambda^{-1/2}\Phi^T & 1/2\Phi^T \\ 1/2\Lambda^{-1/2}\Phi^T & -1/2\Phi^T \end{bmatrix}. \quad (14)$$

Define $R = [\Phi + W_0\Phi\Lambda^{1/2}]^{-1}[\Phi - W_0\Phi\Lambda^{1/2}]$, where W_0 is the initial condition of the ODE at $t = 0$. Clearly, R exists if $[\Phi + W_0\Phi\Lambda^{1/2}]$ is nonsingular. Using the standard solution [1] for the Riccati differential equation (13), we obtain a trajectory $W(t, W_0)$ as

$$\begin{aligned} &W(t, W_0) \\ &\quad = \Phi[I - \exp(-\Lambda^{1/2}t)R\exp(-\Lambda^{1/2}t)] \\ &\qquad \times [I + \exp(-\Lambda^{1/2}t)R\exp(-\Lambda^{1/2}t)]^{-1}\Lambda^{-1/2}\Phi^T. \end{aligned} \quad (15)$$

In (15), $W(t, W_0)$ exists if $[I + \exp(-\Lambda^{-1/2}t)R\exp(-\Lambda^{-1/2}t)]$ is nonsingular for all $t \geq 0$. By Lemma 2, this solution, if it exists, is the unique solution for the ODE in (13).

The asymptotically stable solution is $W_+^* = \lim_{t\to\infty} W(t, W_0) = \Phi\Lambda^{-1/2}\Phi^T$ where $\Lambda^{-1/2}$ is positive definite. Note that this is the unique symmetric positive definite solution for $Q^{-1/2}$. An analysis of the conditions for nonsingualrity of the above matrices for all $t \geq 0$, gives us a domain of attraction for W_+^* that includes $D(W_+^*) = \{W \in R^{d\times d} : W = W^T$ and $\lambda_d(W) > -\lambda_1(Q)^{-1/2}\}$. □

It is now clear that (10) will converge to an asymptotically stable solution $W_+^* = \Phi\Lambda^{-1/2}\Phi^T$ for $W(0)$ symmetric and nonnegative definite, provided $W(k) \in A$ infinitely often, where A is a compact subset of $D(W_+^*)$.

$W(k) \in A$ Infinitely Often: Since A is compact, we need to establish a uniform upper bound for $\|W(k)\|$ for all k. In practical implementation, we can hard-limit the entries of $W(k)$ so that their magnitudes remain below a certain limit ρ and thus within a compact region A. However, this may automatically happen in (10) due to the following fact. We see from (10), that $W(k+1)$ grows in each recursion by $\eta(k)I$, but the growth is controlled by a "forgetting" term of $\eta(k)W(k)\mathbf{x}(k)\mathbf{x}(k)^TW(k)$. As $\|W(k)\|$ gets larger so does $\|W(k)\mathbf{x}(k)\|$. Thus, we can intuitively see that there exits a bound ρ for $\|W(k)\|$ such that if $\|W(k)\| > \rho$, then $\|W(k+1)\| < \|W(k)\|$ w.p.1. To prove this fact, we need the following definitions and assumption.

Definitions: A sequence $\{\mathbf{x}(k)\}$ is said to be in *general position* if every matrix $\tilde{A} = [\mathbf{x}(k), \cdots, \mathbf{x}(k+d-1)]$ of d consecutive vectors is of rank d. This sequence is in *uniform general position* if the smallest singular value of $\tilde{A}$ is uniformly bounded away from zero.

We assume the following.

Assumption (A4): The sequence $\{\mathbf{x}(k)\}$ is in uniform general position.

Lemma 4: Let A1–A4 hold. Then there exists a uniform upper bound for $\lambda_1(W(k))$ for all k.

Proof: Proof is given in the Appendix. □

The Lemma also gives us a uniform upper bound for $\lambda_1(W(0))$ that we need to satisfy at the start of the algorithm. Finally, in order to satisfy $W(k) \in A$ infinitely often, where $A \subset D(W_+^*)$, we establish a uniform lower bound for $\lambda_d(W(k))$. For this, we have the following Lemma.

Lemma 5: Let A1–A4 hold. Then there exists a uniform upper bound for $\eta(k)$ such that $\lambda_d(W(k)) \geq 0$ uniformly for all k.

Proof: Proof is given in the Appendix. □

Theorem 1: Let A1–A4 hold. Further assume that $W(0)$ and $\eta(0)$ are within their uniform upper bounds stated in Lemmas 4 and 5, respectively. If $W(0)$ is assigned random weights such that $W(0)$ is symmetric and nonnegative definite, then with probability one, algorithm (10) will converge, and $W(k) \to W_+^*$ as $k \to \infty$, where $W_+^* = \Phi\Lambda^{-1/2}\Phi^T$ is the unique symmetric positive definite solution for $Q^{-1/2}$.

Proof: By Lemma 5, $\lambda_d(W(k)) \geq 0$ uniformly for all k. By Lemma 4, $\lambda_1(W(k))$ is w.p.1 uniformly bounded above for all k. Then there exists a compact subset A of $D(W_+^*)$ such that $W(k) \in A$ infinitely often. The theorem is now implied by Lemma 1. □

Further Remarks on Algorithm (10): In algorithm (10) we assume $W(0)$ to be nonnegative definite. However, if it is necessary to have $W(0)$ nonpositive definite, then we modify (10) as follows:

$$W(k+1) = W(k) + \eta(k)(W(k)\mathbf{x}(k)\mathbf{x}(k)^TW(k) - I) \quad (16)$$

with $W(0)$ symmetric and nonpositive definite. The above lemmas and theorem can be extended to (16). The ODE is $dW/dt = WQW - I$, whose asymptotically stable solution is $W_-^* = -\Phi\Lambda^{-1/2}\Phi^T$, which is the unique symmetric negative definite solution for $Q^{-1/2}$. The domain of attraction of W_-^* includes $D(W_-^*) = \{W \in R^{d\times d} : W = W^T$ and $\lambda_1(W) < \lambda_1(Q)^{-1/2}\}$. Lemma 4 is modified to state that there exists a uniform lower bound for $\lambda_d(W(k))$ for all k. The uniform upper bound for $\eta(k)$ in Lemma 5 remains the same such that $\lambda_1(W(k)) \leq 0$ uniformly for all k.

B. Adaptive Computation of the Eigenvectors of Q

There are many algorithms to compute eigenvectors of the correlation matrix Q of a random sequence $\{\mathbf{x}(k) \in R^d\}$. These include algorithms due to Oja [22], [23], Sanger [30], Rubner *et al.* [27], [28], Foldiak [8], and Kung and Diamantaras [16] to name a few. Among these, we have chosen the Sanger's algorithm, for the following reasons: 1) this algorithm computes the eigenvectors of Q ordered by

decreasing eigenvalue; 2) the convergence of the algorithm is guaranteed for an initial random sets of weights, and for a wide choice of learning parameters $\gamma(k)$ [see (17) below]; and 3) the algorithm can be implemented in a network with local operations. The modified Sanger's algorithm is

$$V(k+1) = V(k) + \gamma(k)(\mathbf{x}(k)\mathbf{x}(k)^T V(k) - V(k)\mathrm{UT}[V(k)^T\mathbf{x}(k)\mathbf{x}(k)^T V(k)]). \quad (17)$$

Here $V(k)$ is the weight matrix, $\{\gamma(k)\}$ satisfy A2, and $\mathrm{UT}[\cdot]$ sets all entries of its matrix argument below the diagonal to zero, thereby making the matrix upper triangular. Sanger's algorithm is [30]

$$C(k+1) = C(k) + \gamma(k)(C(k)\mathbf{x}(k)\mathbf{x}(k)^T - \mathrm{LT}[C(k)\mathbf{x}(k)\mathbf{x}(k)^T C(k)^T]C(k)) \quad (18)$$

where $\mathrm{LT}[\cdot]$ makes the matrix lower triangular. Algorithm (17) is obtained from (18) by taking its transpose and replacing $C(k)^T$ by $V(k)$. Due to the convergence of Sanger's algorithm [30], if $V(0)$ is assigned random weights, $V(k)$ will converge w.p.1 to a matrix whose *columns* are the eigenvectors of Q, ordered by decreasing eigenvalue.

C. Adaptive Estimation of the Mean $\mathbf{m}$ and Correlation Q of $\{\mathbf{x}(k)\}$

An algorithm for the adaptive estimation of the mean vector $\mathbf{m}$ of a random sequence $\{\mathbf{x}(k)\}$ is

$$\mathbf{w}(k) = \mathbf{w}(k-1) + \delta(k)(\mathbf{x}(k) - \mathbf{w}(k-1)). \quad (19)$$

Here $\mathbf{w}(k)$ is the current estimate of the mean, and $\{\delta(k)\}$ satisfies A2. In order to prove its convergence, we need the following assumption.

Assumption (A5): $\{\mathbf{x}(k) \in R^d\}$ are uniformly bounded random vectors with $\lim_{k\to\infty} E[\mathbf{x}(k)] = \mathbf{m}$.

In order to prove the w.p.1 convergence of $\mathbf{w}(k)$ to $\mathbf{m}$, we use assumptions A5, A2, and A3. The ODE is $\frac{d\mathbf{w}}{dt} = \mathbf{m} - \mathbf{w}$. The only asymptotically stable solution is $\mathbf{m}$, whose domain of attraction is R^d.

An algorithm for the adaptive estimation of the correlation matrix Q of $\{\mathbf{x}(k)\}$ is

$$C(k) = C(k-1) + \delta(k)(\mathbf{x}(k)\mathbf{x}(k)^T - C(k-1)). \quad (20)$$

Using A1–A3, the ODE is $\frac{dC}{dt} = Q - C$. The only asymptotically stable solution is Q, whose domain of attraction is $R^{d\times d}$.

IV. Networks for Feature Extraction

In this section, we shall use the algorithms in Section III to train networks for feature extraction by the three criteria discussed in Section II.

A. Network Implementation of Algorithm (10)

We present an implementation of algorithm (10) where the weight matrix $W(k)$ is updated by sequential update rules. Let $\mathbf{x}(k)$ be a training input for (10), and let $\mathbf{o}(k) = W(k)\mathbf{x}(k)$ be its output. Let $w^{ij}(k)$ denote the (i,j)th element of $W(k)$, and

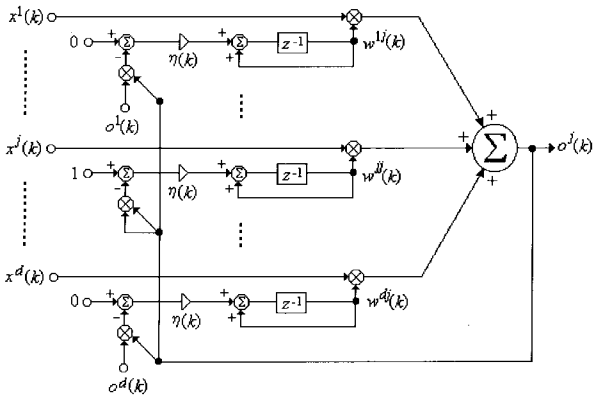

Fig. 2. Functional block diagram illustrating the $Q^{-1/2}$ training algorithm (10).

$o^i(k)$ denote the ith component of $\mathbf{o}(k)$. The weight update for (10) can be written as

$$\Delta w^{ij}(k) = p\eta(k)(\delta_{ij} - o^i(k)o^j(k)) \text{ for } i,j = 1,\cdots,d \quad (21)$$

where δ_{ij} is the Kronecker's delta. Since ΔW is symmetric, we need to update only the lower or upper triangular block of ΔW. Fig. 2 shows a network implementation for updating w^{ij} for $i,j = 1,\cdots,d$. The network above for algorithm (10) will be used as a block in the feature extraction networks described below. We shall refer to this network as the $Q^{-1/2}$ *network.*

B. Networks for Optimal Features from Gaussian Data

Unimodal Case: Recall from (2), that when the data is unimodal Gaussian, the feature $f_i(\mathbf{x})$, for class ω_i is $f_i(\mathbf{x}) = \|Q_i^{-1/2}(\mathbf{x} - \mathbf{m}_i)\|^2$ for $i = 1,\cdots,m$, where $\mathbf{m}_i$ is the class mean and Q_i is the class covariance. Hence, it involves the computation of $Q_i^{-1/2}$, and the network described in Fig. 2. The feature extraction network consists of two layers (see Fig. 3): 1) The first layer is trained by the $Q^{-1/2}$ algorithm (10). It has weight matrix $W_i(k)$ that tends to $Q_i^{-1/2}$ w.p.1 as $k \to \infty$. 2) The second layer computes the square of the norm of the first layer outputs. The weights for the second layer are fixed at one. The training of the network is discussed below.

Let $\{\mathbf{x}_i(k)\}$ denote a sequence of training patterns from $\omega_i,\ i = 1,\cdots,m$. Note that we have assumed that the input patterns are a *sequence of data*, and we can not assume that the class mean $\mathbf{m}_i$ is known. We, therefore, obtain a running estimate $\mathbf{m}_i(k)$ of the class mean by (19) from $\{\mathbf{x}_i(k)\}$.

Algorithm (10) is used *in parallel* for each class with weight matrix $W_i(k)$. A training sequence $\{\mathbf{y}_i(k)\}$ is generated as follows: $\{\mathbf{y}_i(k) = \mathbf{x}_i(k) - \mathbf{m}_i(k)\}$ for each ω_i. Note that if the network is trained with $\{\tilde{\mathbf{y}}_i(k) = \mathbf{x}_i(k) - \mathbf{m}_i\}$, then by Theorem 1, $W(k)$ converges w.p.1 to $Q_i^{-1/2}$. However, we are using the sequence $\{\mathbf{y}_i(k) = \mathbf{x}_i(k) - \mathbf{m}_i(k)\}$ instead. Theorem 2 below proves that also for $\{\mathbf{y}_i(k)\}, W_i(k)$ converges w.p.1 to $Q_i^{-1/2}$. Fig. 3 shows the network for the unimodal case. Given an input pattern $\mathbf{x} \in \omega_i$, the feature $f_i(\mathbf{x})$ is obtained at the output of the network. In Fig. 3, $y_i^s(k), s = 1,\cdots,d$, is the sth component of $\mathbf{y}_i(k)$. The activation function is $g(x) = x^2$ at the hidden nodes, and linear at the output node.

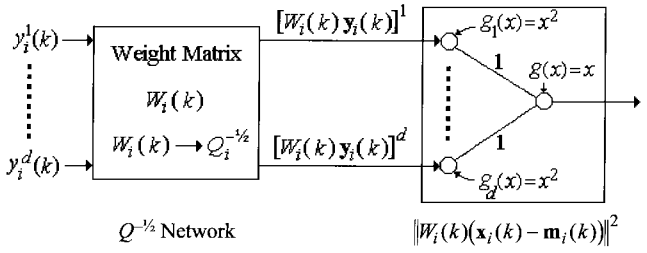

Fig. 3. Network for unimodal Gaussian data for class $\omega_i,\ i = 1, \cdots, m$.

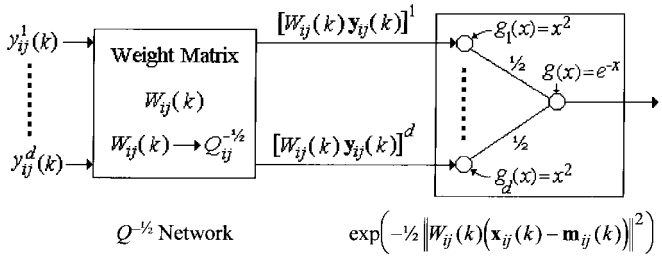

Fig. 4. Network for multicluster Gaussian data for cluster $j,\ j = 1, \cdots, r_i$ in Class $\omega_i,\ i = 1, \cdots, m$.

Convergence with Estimated Class Means: We need to prove the convergence of the network weights $W_i(k)$ with estimated class means $\mathbf{m}_i(k)$ instead of the true means $\mathbf{m}_i$. For this proof, we need the following assumptions for $\{\mathbf{x}_i(k)\}$.

Assumption (B1): The training sequence $\{\mathbf{x}_i(k) \in R^d\}$ for class ω_i consists of random vectors, where each $\mathbf{x}_i(k)$ is uniformly bounded. Moreover, $\lim_{k\to\infty} E[\mathbf{x}_i(k)] = \mathbf{m}_i$ and $\lim_{k\to\infty} E[(\mathbf{x}_i(k)-\mathbf{m}_i)(\mathbf{x}_i(k)-\mathbf{m}_i)^T] = Q_i$. Further assume that $\{\mathbf{x}_i(k)\}$ satisfies A3, and $\{\mathbf{x}_i(k) - \mathbf{m}_i\}$ satisfies A4.

Assumption (B2): Sequence $\{\eta(k)\}$ satisfies A2, and the uniform upper bound in Lemma 5.

Theorem 2: Let B1 and B2 hold. If $W_i(0)$ is assigned random weights such that it is symmetric and nonnegative definite, and within the uniform upper bound stated in Lemma 4, then for the training sequence $\{\mathbf{y}_i(k)\}$, algorithm (10) will converge with probability one, and $W_i(k)$ will approach the unique symmetric positive definite solution for $Q_i^{-1/2}$ as $k \to \infty$.

Proof: Proof given in the Appendix. □

Multimodal Case: If the data for each class is not unimodal and consists of r_i clusters in ω_i, then we shall modify the above network to extract a feature for each cluster. Recall from (4) that the feature $f_{ij}(\mathbf{x})$ for cluster j in ω_i is $f_{ij}(\mathbf{x}) = \exp(-h_{ij}(\mathbf{x}))$ where $h_{ij}(\mathbf{x}) = 1/2\|Q_{ij}^{-1/2}(\mathbf{x} - \mathbf{m}_{ij})\|^2$ and $\mathbf{m}_{ij}$ and Q_{ij} are the cluster mean and covariance, respectively. We have assumed that the r_i clusters can be determined from the training patterns for each class. The training procedure for each cluster uses the $Q^{-1/2}$ algorithm (10) for the first layer, with the current estimate $\mathbf{m}_{ij}(k)$ of the cluster mean $\mathbf{m}_{ij}$. Given an input sequence $\{\mathbf{x}_{ij}(k)\}$, the training sequence for the first layer is $\{\mathbf{y}_{ij}(k) = \mathbf{x}_{ij}(k) - \mathbf{m}_{ij}(k)\}$, and weight matrix $W_{ij}(k)$ for $j = 1, \cdots, r_i,\ i = 1, \cdots, m$. Weights for the second layer are fixed at $1/2$. The modified network is shown in Fig. 4. In Fig. 4, $y_{ij}^s(k),\ s = 1, \cdots, d$, is the sth component of $\mathbf{y}_{ij}(k)$ for cluster j in ω_i. The activation function is $g(x) = x^2$ at the hidden nodes, and $g(x) = e^{-x}$ at the output node.

C. Network for Linear Discriminant Analysis

Recall that for linear discriminant analysis (LDA), we need to compute the eigenvector matrix Φ of $S_w^{-1}S_m$. In particular, we need to extract the eigenvectors corresponding to the $p(\leq d)$ largest eigenvalues of $S_w^{-1}S_m$.

As discussed in Section II, we can solve this problem in two-steps. The first step is the $Q^{-1/2}$ algorithm that estimates $S_w^{-1/2}$. The second step computes the eigenvectors of the symmetric matrix $S_w^{-1/2}S_mS_w^{-1/2}$ by a PCA algorithm such as the Sanger's rule, which yields the eigenvector matrix $\Psi = S_w^{1/2}\Phi$. Then the combined algorithm yields Φ. This methodology is the basis for the following architecture. However, since the class and mixture means of the data are unknown, running estimates of these means are required. The convergence of the algorithm with the running estimates of the means is proven. We give a two-layer solution for the LDA problem, where (1) the first layer is trained with the $Q^{-1/2}$ algorithm (10) whose architecture is shown in Fig. 2; and (2) the second layer is trained with the modified Sanger's algorithm (17) whose architecture is well known [13], [30]. The convergence of the two layers trained *simultaneously* is also given.

Training of the First Layer: Let $\{\mathbf{x}(k)\}$ denote a training sequence with samples from classes $\{\omega_1, \cdots, \omega_m\}$. For each $\mathbf{x}(k)$, let $c(\mathbf{x}(k)) \in \{1, \cdots, m\}$ denote its class, which is known in advance. The training sequence $\{\mathbf{y}(k)\}$ for the first layer is obtained as follows: $\{\mathbf{y}(k) = \mathbf{x}(k) - \mathbf{m}_{c(\mathbf{x}(k))}(k)\}$. Here $\mathbf{m}_i(k)$ for $i = 1, \cdots, m$, is the current estimate of the class mean for ω_i, obtained by algorithm (19) from $\{\mathbf{x}(k): \mathbf{x}(k) \in \omega_i\}$. The correlation of $\{\mathbf{y}(k)\}$ is the within-class scatter matrix S_w. This can be seen as follows:

$$\begin{aligned} E[\mathbf{y}(k)\mathbf{y}(k)^T] &= \sum_{i=1}^{m} P(\omega_i) E\left[\mathbf{y}(k)\mathbf{y}(k)^T \mid \omega_i\right] \\ &= \sum_{i=1}^{m} P(\omega_i) E\left[(\mathbf{x}(k) - \mathbf{m}_i(k))(\mathbf{x}(k) \right. \\ &\qquad \left. - \mathbf{m}_i(k))^T \mid \omega_i\right]. \end{aligned}$$

Here, $\lim_{k\to\infty} E[(\mathbf{x}(k)-\mathbf{m}_i(k))(\mathbf{x}(k)-\mathbf{m}_i(k))^T \mid \omega_i] = Q_i$. Thus, $\lim_{k\to\infty} E[\mathbf{y}(k)\mathbf{y}(k)^T] = \sum_{i=1}^{m} P(\omega_i)Q_i = S_w$. The training algorithm for the first layer is (10) with weight matrix $W(k)$. Then, by Theorem 2, as $k \to \infty$, $W(k) \to S_w^{-1/2}$ w.p.1. Note that this layer has a single network (irrespective of class), for which the training sequence $\{\mathbf{y}(k)\}$ is obtained by modifying $\{\mathbf{x}(k)\}$ by the class means.

Training of the Second Layer: In order to train the second layer, we generate a sequence $\{\mathbf{z}(k) = \mathbf{x}(k) - \mathbf{m}(k)\}$. Note that $\mathbf{x}(k)$ is the training input irrespective of class assignments. Here $\mathbf{m}(k)$ is the current estimate of the mixture mean $\mathbf{m}_0$ obtained by (19) from $\{\mathbf{x}(k)\}$. Sequence $\{\mathbf{z}(k)\}$ is passed through the first layer to obtain $\{\mathbf{u}(k) = W(k)\mathbf{z}(k)\}$. The second layer is trained with $\{\mathbf{u}(k)\}$. The training algorithm is the modified Sanger's rule (17) with weight matrix $V(k)$.

As $k \to \infty, V(k)$ tends to the eigenvectors of the matrix $\lim_{k\to\infty} E[\mathbf{u}(k)\mathbf{u}(k)^T]$, ordered by decreasing eigenvalue.

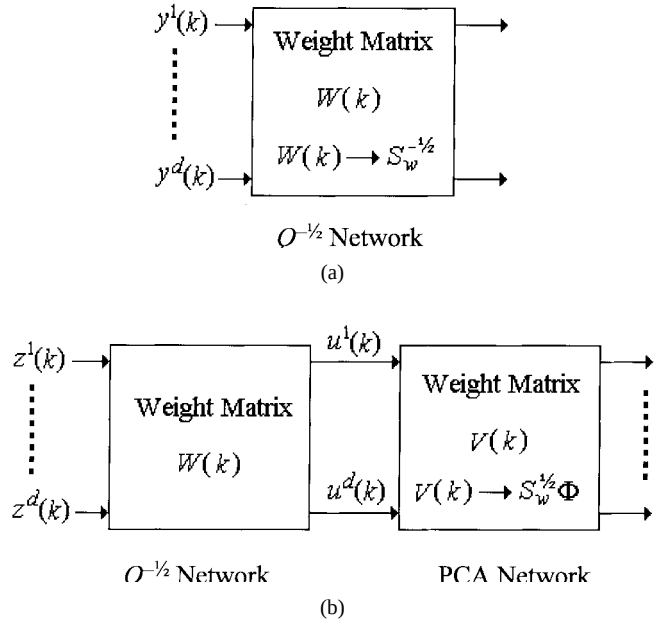

Fig. 5. Network for linear discriminant analysis. All nodes are linear: (a) training of the first layer; (b) training of the second layer.

Intuitively, the operation of the two-layer network can be described as follows: as $k \rightarrow \infty, W(k) \rightarrow S_w^{-1/2}$ w.p.1 by Theorem 2, and $E[\mathbf{z}(k)\mathbf{z}(k)^T] \rightarrow S_m$. Thus, $E[\mathbf{u}(k)\mathbf{u}(k)^T] \rightarrow S_w^{-1/2} S_m S_w^{-1/2}$ w.p.1, and $V(k)$ tends to the eigenvector matrix $\Psi = S_w^{1/2}\Phi$ of the symmetric real matrix $S_w^{-1/2} S_m S_w^{-1/2}$ w.p.1, ordered by decreasing eigenvalue. A rigorous proof of this intuitive argument is given in Theorem 3.

Simultaneous Training of Two Layers: From the convergence of $W(k)$ and $V(k)$, we obtain $\lim_{k\rightarrow\infty}\{W(k)V(k)\} = S_w^{-1/2} S_w^{1/2}\Phi = \Phi$ w.p.1. Thus, the combined two-layer network weight matrix $W(k)V(k)$ converges w.p.1 to the generalized eigenvector matrix Φ. Note that $V(k)$ is obtained from the modified Sanger's algorithm, and its columns are arranged in the order of decreasing eigenvalues. Therefore, the first $m-1$ columns of $W(k)V(k)$ form the relevant feature extraction transform.

Given an input $\mathbf{x}$, the output of this cascaded network tends to $\Phi^T\mathbf{x}$ w.p.1 as $k \rightarrow \infty$. Since this is already arranged in decreasing eigenvalue, the first $m-1$ values of the output are the relevant projections of $\mathbf{x}$ onto the $m-1$ dimensional feature subspace. The two-layer network is shown in Fig. 5. In Fig. 5, $y^s(k)$ and $z^s(k), s = 1, \cdots, d$, are the sth component of $\mathbf{y}(k)$ and $\mathbf{z}(k)$, respectively.

Theorem 3: Let B1 hold for $\{\mathbf{x}(k)\}$, and let B2 hold. Let $\{\gamma(k)\}$ satisfy A2. If $V(0)$ is assigned random weights, then for the training sequence $\{\mathbf{u}(k) = W(k)\mathbf{z}(k)\}$, algorithm (17) will converge with probability one, and $V(k)$ will approach the matrix whose columns are the eigenvectors $\Psi = S_w^{1/2}\Phi$ of $S_w^{-1/2} S_m S_w^{-1/2}$ ordered by decreasing eigenvalue.

Proof: Proof given in the Appendix. □

D. Network for the Bhattacharyya Distance Features

From the above networks, we can extract at most $m-1$ features for an m-class problem. In order to complement this result, the covariance part of the Bhattacharyya distance measure (9) is used to extract features along which the covariances of two classes are different. Let the two classes be ω_1 and ω_2. Like the LDA problem, the optimum linear transform for the Bhattacharyya distance features consists of the eigenvectors of $Q_1^{-1}Q_2$, where Q_i is the class covariance matrix for ω_i for $i = 1, 2$. The architecture for the two-layer network is similar to the one given in Fig. 5.

Network Training: Let $\{\mathbf{x}_i(k)\}$ for $i = 1, 2$, denote the training sequences from ω_1 and ω_2, respectively. For this network, the first layer is trained with data from ω_1, and the second layer with data from ω_2. We, therefore, generate two sequences $\{\mathbf{y}_i(k) = \mathbf{x}_i(k) - \mathbf{m}_i(k)\}$ for $i = 1, 2$. Here $\mathbf{m}_i(k)$ is the current estimate of the class mean for ω_i. The first layer is trained with $\{\mathbf{y}_1(k)\}$ by the $Q^{-1/2}$ algorithm (10) with weight matrix $W(k)$. By Theorem 2, $W(k) \rightarrow Q_1^{-1/2}$ w.p.1 as $k \rightarrow \infty$.

Sequence $\{\mathbf{y}_2(k)\}$ is passed through the first layer to obtain $\{\mathbf{z}(k) = W(k)\mathbf{y}_2(k)\}$, which is used to train the second layer by (17). By Theorem 3, the weight matrix $V(k)$ of the second layer tends to the eigenvectors $\Psi = Q_1^{1/2}\Phi$ of $Q_1^{-1/2} Q_2 Q_1^{-1/2}$ w.p.1 as $k \rightarrow \infty$, ordered by decreasing eigenvalue. Here Φ is the eigenvector matrix of $Q_1^{-1}Q_2$. From the convergence of $W(k)$ and $V(k)$, the combined weight matrix $W(k)V(k) \rightarrow \Phi$ w.p.1 as $k \rightarrow \infty$.

Order of Significance of the Features: Although the features are arranged in the order of decreasing eigenvalues $\lambda_i, i = 1, \cdots, d$, of $Q_1^{-1}Q_2$, unlike the LDA features, the significance of the Bhattacharyya distance features depends on the values of $\theta_i = \lambda_i + (1/\lambda_i), i = 1, \cdots, d$. However, for some ranges of λ_i, the significance of the features can be readily determined from the order of λ_i. Note that $\theta_i = \lambda_i + (1/\lambda_i)$ is a monotonically decreasing function of λ_i for $0 < \lambda_i \leq 1$, and increasing for $\lambda_i \geq 1$. Thus, if all $\lambda_i \geq 1$, then the significance of the features have the same order as λ_i for $i = 1, \cdots, d$. On the other hand, if all $\lambda_i \in (0, 1]$, then the significance of the features have the reverse order as λ_i for $i = 1, \cdots, d$.

If, however, an estimate of θ_i is required, we compute the correlation matrix $C(k)$ of the network output for input $\{\mathbf{y}_2(k)\}$. Note that for input $\{\mathbf{y}_2(k)\}$ the network output is $\{\mathbf{v}(k) = V(k)^T W(k)\mathbf{y}_2(k)\}$. By applying Theorem 3 to $\{\mathbf{v}(k)\}$, we can show that $C(k) = E[\mathbf{v}(k)\mathbf{v}(k)^T] \rightarrow \Phi^T Q_2 \Phi = \Lambda$ w.p.1, where Λ is the eigenvalue matrix of $Q_1^{-1}Q_2$. Correlation matrix $C(k)$ is obtained by algorithm (20) from $\{\mathbf{v}(k)\}$, and $C(k)^{-1}$ can be found by using the so called Sherman–Morrison formula [12]

$$C(k)^{-1} = (1 - \delta(k))^{-1}\left(C(k-1)^{-1} - \frac{\delta(k)C(k-1)^{-1}\mathbf{v}(k)\mathbf{v}(k)^T C(k-1)^{-1}}{1 - \delta(k) + \delta(k)\mathbf{v}(k)^T C(k-1)^{-1}\mathbf{v}(k)}\right). \tag{22}$$

Estimates of θ_i are obtained from the diagonal elements of $\Theta(k) = C(k) + C(k)^{-1}$. Note, however, that (22) may be unnecessary, and the diagonal elements of $C(k)$ may be sufficient to estimate λ_i, and therefore θ_i. The components

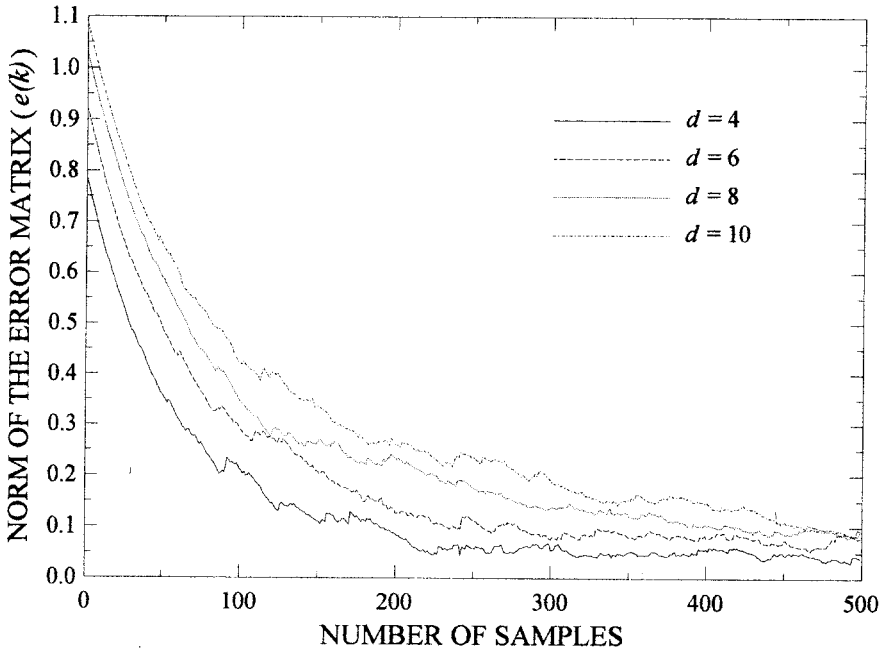

Fig. 6. Convergence of the $Q^{-1/2}$ Algorithm (10).

of the network output corresponding to the p largest θ_i are the relevant projections of the input pattern $\mathbf{x}$ in the feature subspace.

V. Simulation Results

In this section we use the networks described in Section IV to extract features, and compare them with results from nonadaptive methods in pattern recognition. Since the features for unimodal and multicluster Gaussian data and the Bhattacharyya distance features have an underlying Gaussian assumption, and since Gaussian distributions are common in many pattern recognition problems [6], [9], [10], [24], [35], it is appropriate to generate multiclass multivariate Gaussian data for our experiments. The mean vectors and the covariance matrices for the Gaussian distributions are chosen from the experiments of Okada and Tomita [24] with the covariance matrices multiplied by 20.

A. Experiments on $Q^{-1/2}$ Algorithm (10)

Here we do four experiments on the $Q^{-1/2}$ algorithm (10). We used the first covariance matrix in [24], and multiplied it by 20 (as given in the equation shown at the bottom of the page) for data in R^{10} The ten eigenvalues of Q are 117.996, 55.644, 34.175, 20.589, 7.873, 5.878, 1.743, 1.423, 1.213, and 1.007.

We did four experiments, respectively, with the 4×4, 6×6, and 8×8 principal minors, and with Q. For each covariance matrix, we generated 500 samples of zero-mean multivariate Gaussian data, and estimated the $Q^{-1/2}$ matrix with algorithm (10). We did the experiments for only *one epoch* of the data, where *one epoch* means presenting all training data once in random order. The actual value of $Q^{-1/2}$ (denoted by $Q^{-1/2}_{\text{actual}}$) is obtained from the sample correlation matrix using a standard eigenvector computation method [12]. For each experiment, at kth update, we computed the L_2-norm of the error between the estimated and actual $Q^{-1/2}$ matrices, and we denote it by $e(k)$; i.e.,

$$e(k) = \left\| W(k) - Q^{-1/2}_{\text{actual}} \right\|_2. \tag{23}$$

For each covariance matrix, we reported the convergence of the $Q^{-1/2}$ algorithm in Fig. 6. The final values of e after 500

$$Q = 20 \begin{bmatrix} 0.091 & & & & & & & & & \\ 0.038 & 0.373 & & & & & & & & \\ -0.053 & 0.018 & 1.430 & & & & & & & \\ -0.005 & -0.028 & 0.017 & 0.084 & & & & & & \\ 0.010 & -0.011 & 0.055 & -0.005 & 0.071 & & & & & \\ -0.136 & -0.367 & -0.450 & 0.016 & 0.088 & 5.720 & & & & \\ 0.155 & 0.154 & -0.038 & 0.042 & 0.058 & -0.544 & 2.750 & & & \\ 0.030 & -0.057 & -0.298 & -0.022 & -0.069 & -0.248 & -0.343 & 1.450 & & \\ 0.002 & -0.031 & -0.041 & 0.001 & -0.008 & 0.005 & -0.011 & 0.078 & 0.067 & \\ 0.032 & -0.065 & -0.030 & 0.005 & 0.003 & 0.095 & -0.120 & 0.028 & 0.015 & 0.341 \end{bmatrix}.$$

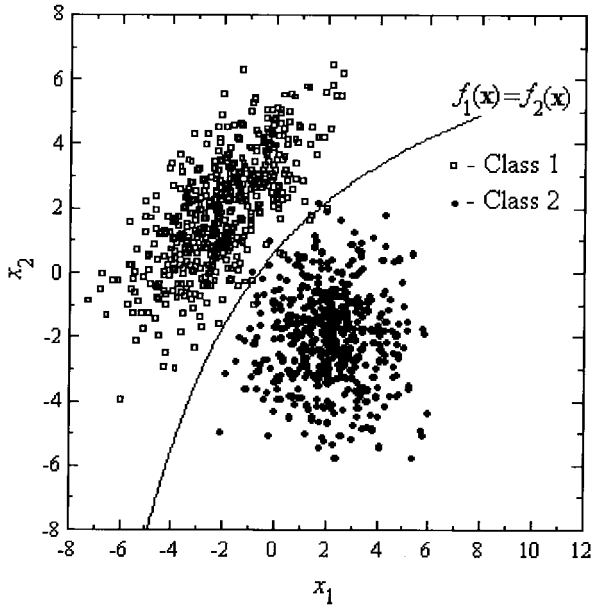

Fig. 7. Distributions of the two-class 2–D Gaussian data.

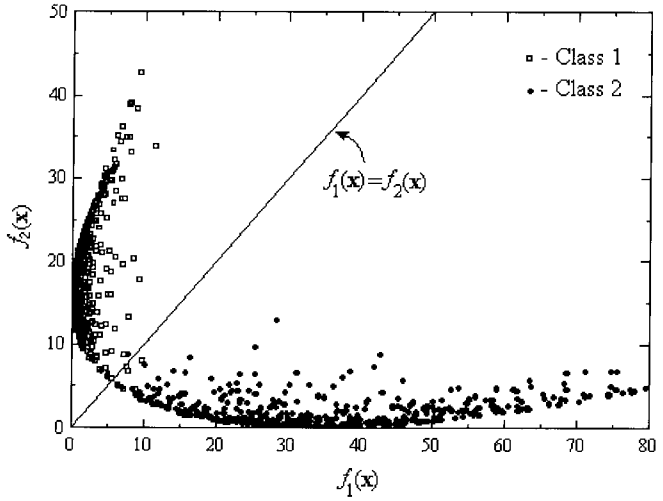

Fig. 8. Display of two-class 2-D Gaussian data in the feature space.

samples are $e = 0.0398$ for $d = 4$, $e = 0.0784$ for $d = 6$, $e = 0.0935$ for $d = 8$, and $e = 0.0959$ for $d = 10$.

It is clear from Fig. 6 that the error matrices are all close to the zero matrix. The small differences compared to the actual values are due to random fluctuations in the elements of $W(k)$ caused by the varying input data. Experiments with higher epochs show an improvement in the estimation accuracy. For example, for the ten-dimensional (10-D) data, with five epochs we obtained $e = 0.0226$, with ten epochs $e = 0.0116$, and with 20 epochs $e = 0.0057$. Similar results as above are obtained for uniform distributions.

B. Experiments on Optimal Features from Gaussian Data

Here we use the unimodal network described in IV-B1 to extract the optimal features for the Gaussian data. We do the following: 1) generate 500 samples of 2-D Gaussian data, each from two classes with different covariance matrices and mean vectors; 2) for each pattern $\mathbf{x}$, use the unimodal network to extract the quadratic features $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ [see (1)] for classes ω_1 and ω_2, respectively, and show the classification results; 3) compare these features with the actual features obtained from the sample means and covariances; 4) repeat these steps for a 10-D five-class Gaussian data; and 5) repeat these steps for 10-D five-class uniform data to show the effectiveness of these features on non-Gaussian distributions.

Note that in the following experiments, the classification result is not of central importance. We need to demonstrate that the networks estimate the features accurately, and preserve the classifying ability of the true features. The main goal is to create clusters of data, that are linearly separable. We, therefore, used simple classifiers such as the minimum or maximum selectors or simple thresholds, and used the "resubstitution method" [10] for testing the classification performance.

Fig. 7 shows a two-class 2-D Gaussian data with the following means and covariances for ω_1 and ω_2, respectively,

$$\mathbf{m}_1 = \begin{bmatrix} -2 \\ 2 \end{bmatrix},\ Q_1 = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix},\ \mathbf{m}_2 = \begin{bmatrix} 2 \\ -2 \end{bmatrix},\ Q_2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$
$$\text{and} \quad n_1 = n_2 = 500.$$

Clearly, the distributions overlap, and are not linearly separable. The optimum feature is a quadratic function as shown in Fig. 7. After training the unimodal network for *one epoch*, we extracted the features f_1 and f_2 from the training data. For each training pattern $\mathbf{x}$, the extracted features $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ are classified by simply assigning $\mathbf{x}$ to class ω_i if $f_i(\mathbf{x})$ is the minimum of the two. We obtained a classification error of six out of 1000 total samples. Fig. 8 shows the data in the feature space. Clearly, the data from the two classes are two different clusters separable by a straight line.

The above result is compared to the actual values of f_1 and f_2. For each pattern $\mathbf{x}$, let $f_i(\mathbf{x})$ for $i = 1, 2$, be the actual feature computed from the sample means and covariances, and $\hat{f}_i(\mathbf{x})$ be the estimated feature obtained from the unimodal network after one epoch. We define *normalized error* of the features as follows:

$$E_{f_i} = \frac{\sum_{j=1}^{n} \left| f_i(\mathbf{x}_j) - \hat{f}_i(\mathbf{x}_j) \right|}{\sum_{j=1}^{n} f_i(\mathbf{x}_j)} \quad \text{for } i = 1, \cdots, m \qquad (24)$$

where n is the total number of samples. We obtained $E_{f_1} = 0.0044$ and $E_{f_2} = 0.0019$, which are both less than 1%, showing that the estimated features are close to their true values. Classification with the actual features gave us the same error of six samples out of 1000.

Next, we repeated the experiment with a 10-D Gaussian data from five classes with 500 samples from each class. The means and covariances are obtained from [24] with the covariance matrices multiplied by 20. We trained the unimodal network with this data, and utilized it to extract features from the training data. We computed the normalized error given in (24), and obtained $E_{f_1} = 0.0218$, $E_{f_2} = 0.0396$, $E_{f_3} = 0.0290$, $E_{f_4} = 0.0431$, and $E_{f_5} = 0.0479$, which show that the estimates are close to their true values. Using a simple classification rule as above, we obtained one misclassification out of 2500 samples.

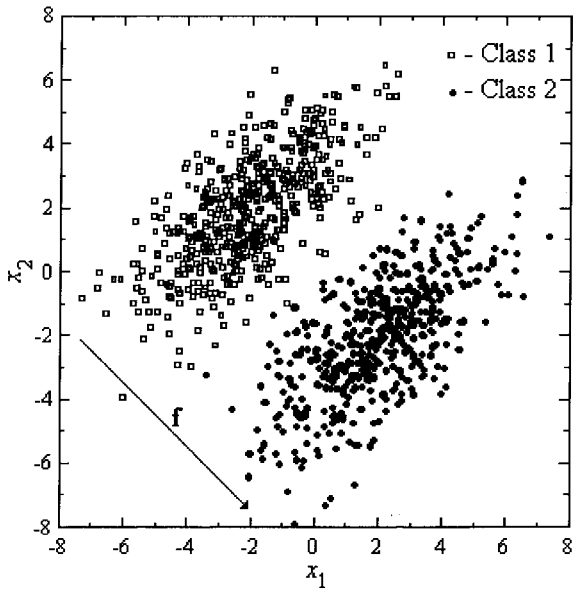

Fig. 9. Distributions of the two-class 2–D Gaussian data for LDA features.

Finally, we repeated the above simulation for uniform distributions. We generated uniform data with the above mean vectors, trained the unimodal network with this data, and extracted the features for the training data. We obtained zero misclassification, and $E_{f_1} = 0.0316$, $E_{f_2} = 0.0327$, $E_{f_3} = 0.0315$, $E_{f_4} = 0.0443$, and $E_{f_5} = 0.0419$. Clearly, this feature can also be effective for non-Gaussian data.

C. Experiments on Linear Discriminant Analysis Features

Here we test the LDA network described in Section IV-C. We do the following: 1) generate 500 samples of 2-D Gaussian data, each from two classes with different mean vectors and same covariance matrix; 2) use the LDA network to extract the relevant features for classification, and show the classification results; 3) compare these features with their actual values computed from the sample scatter matrices; and 4) repeat this experiment for a 10-D five-class Gaussian data.

Fig. 9 shows a 2-D two-class Gaussian data with the following means and covariances for ω_1 and ω_2, respectively,

$$\mathbf{m}_1 = \begin{bmatrix} -2 \\ 2 \end{bmatrix},\ Q_1 = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix},\ \mathbf{m}_2 = \begin{bmatrix} 2 \\ -2 \end{bmatrix},\ Q_2 = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix}$$
$$\text{and} \quad n_1 = n_2 = 500.$$

Here, $S_w = \left[\begin{smallmatrix} 3 & 2 \\ 2 & 3 \end{smallmatrix}\right]$ $S_b = \left[\begin{smallmatrix} 4 & -4 \\ -4 & 4 \end{smallmatrix}\right]$, and the eigenvector matrix Φ of $S_w^{-1}S_b$ is $\left[\begin{smallmatrix} 0.7071 & 0.3162 \\ -0.7071 & 0.3162 \end{smallmatrix}\right]$, corresponding to eigenvalues eight and zero. Thus, the first column $\mathbf{f}$ of Φ is the relevant feature vector. Projections on $\mathbf{f}$ contain all information for classification. Note that the above numbers are exact values and not the sample estimates.

After training the LDA network for just *one epoch*, we estimated $\hat{\mathbf{f}} = [0.7018 - 0.6936]^T$. Define normalized error $E_{\mathbf{f}}$ as $E_{\mathbf{f}} = \frac{\|\mathbf{f} - \hat{\mathbf{f}}\|}{\|\mathbf{f}\|}$, where $\mathbf{f}$ is computed from the sample scatter matrices, and $\hat{\mathbf{f}}$ is estimated from the LDA network. We obtained $E_{\mathbf{f}} = 0.0027$. Clearly, the network estimated the feature vector accurately in just one epoch. For classification of the training data, we used a simple threshold $t = 0.009$ in the feature space. Classification performance with the true and estimated features are same at two misclassifications out of 1000 samples.

Next, we repeated this experiment with the 10-D five-class Gaussian data, with 500 samples from each class. The means and covariances are obtained from [24] with the covariance matrices multiplied by 20. The eigenvalues of $S_w^{-1}S_b$ are 10.84, 7.01, 0.98, 0.34, 0, 0, 0, 0, 0, 0. Thus, the data has intrinsic dimensionality of four for classification, of which only two features are significant. The LDA network is trained with this data for one epoch. Let the two significant feature vectors be denoted by $\mathbf{f}_1$ and $\mathbf{f}_2$, and their estimates from the LDA network be $\hat{\mathbf{f}}_1$ and $\hat{\mathbf{f}}_2$, respectively. The normalized errors at the end of 2500 samples (i.e., *one epoch*) are $E_{\mathbf{f}_1} = 0.0806$ and $E_{\mathbf{f}_2} = 0.0292$, suggesting an accurate estimation by the LDA network.

In Fig. 10, we show the convergence of feature vectors $\hat{\mathbf{f}}_1$ and $\hat{\mathbf{f}}_2$ for the LDA network by computing the normalized error at each update. We observe that both feature vectors converge close to their true values in a finite number of samples.

Fig. 11 shows the projection of the training data on $\hat{\mathbf{f}}_1$ and $\hat{\mathbf{f}}_2$. Although four features are necessary for complete class separation (as shown by the eigenvalues of $S_w^{-1}S_b$), we see that the data is separated into five clusters (although overlapping) with just the two significant feature vectors $\hat{\mathbf{f}}_1$ and $\hat{\mathbf{f}}_2$. Note that these clusters are not apparent in the original 10-D data.

D. Experiments on Features from Bhattacharyya Distance Measure

Here we test the Bhattacharyya distance network described in Section IV-D to extract features along which the variances of two classes are different. We do the following: 1) generate 500 samples each of 2-D two-class Gaussian data with same mean and different covariances; 2) use the Bhattacharyya network to extract feature vectors $\mathbf{f}_1$ and $\mathbf{f}_2$ from ω_1 and ω_2, respectively; 3) compare these feature vectors with their actual values computed from the sample covariance matrices; and 4) repeat this experiment for a 10-D two-class Gaussian data. Fig. 12 shows a two-class 2-D Gaussian data with the following means and covariances:

$$\mathbf{m}_1 = \mathbf{m}_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix},\ Q_1 = \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix},\ Q_2 = \begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix}$$
$$\text{and} \quad n_1 = n_2 = 500.$$

Note that $S_w = 3I$, and $S_b = 0$. Thus, both eigenvalues of $S_w^{-1}S_b$ are zero, leading to no *LDA features*. The eigenvector matrix of $Q_1^{-1}Q_2$ is $\Phi = \left[\begin{smallmatrix} 0.5 & 0.3536 \\ -0.5 & 0.3536 \end{smallmatrix}\right]$ corresponding to $\lambda_1 = 2$ and $\lambda_2 = 0.5$ with $\theta_1 = \theta_2 = 2.5$. Thus, both features are important, as is also clear from Fig. 12. We trained the network for one epoch, and obtained $\hat{\Phi} = \left[\begin{smallmatrix} 0.5042 & 0.1984 \\ -0.5055 & 0.3891 \end{smallmatrix}\right]$ with normalized errors $E_{\mathbf{f}_1} = 0.0199$, and $E_{\mathbf{f}_2} = 0.3296$. Estimation of $\mathbf{f}_1$ is accurate, whereas the estimation of $\mathbf{f}_2$ is relatively poor. Note that the corresponding eigenvalue $\lambda_2 = 0.5$ is smaller compared to $\lambda_1 = 2$, giving us a slower convergence, and resulting in a poorer estimation. Hence, we repeated the experiment for higher epochs. With five epochs,

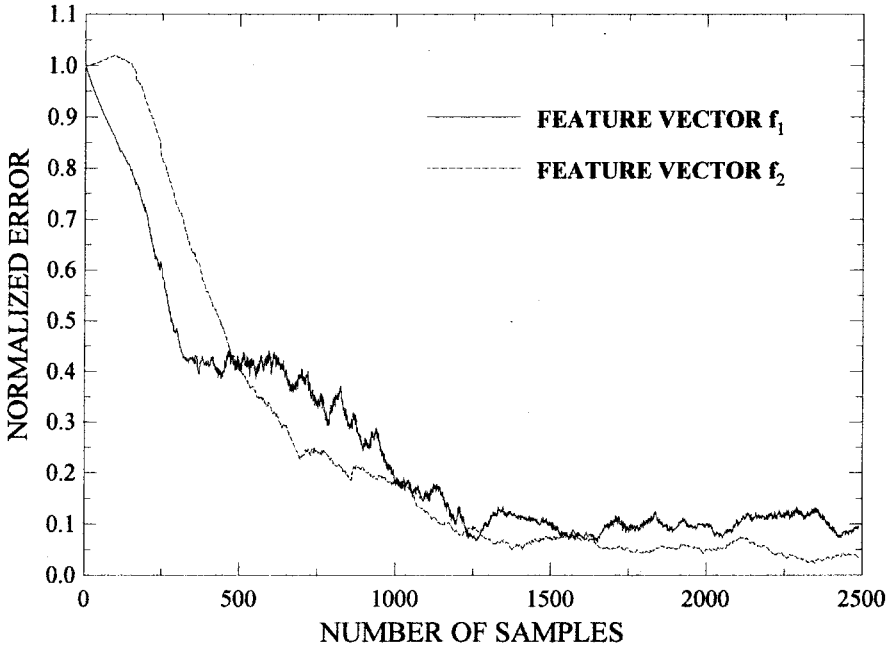

Fig. 10. Convergence of the LDA network.

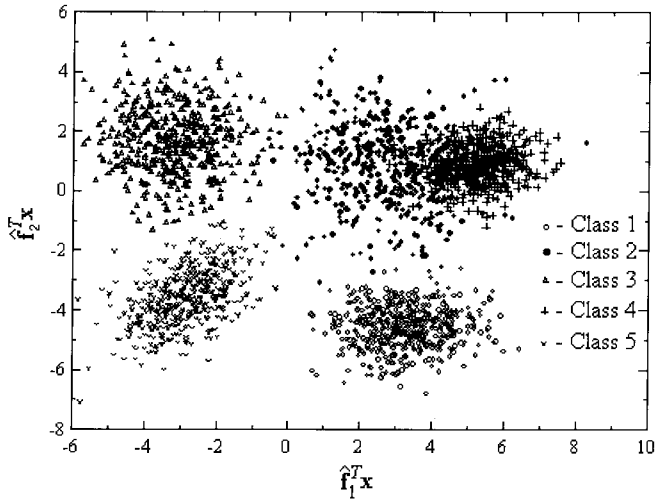

Fig. 11. Display of five-class 10–D Gaussian data projected on feature vectors $\hat{\mathbf{f}}_1$ and $\hat{\mathbf{f}}_2$.

we obtained $E_{\mathbf{f}_1} = 0.0076$, and $E_{\mathbf{f}_2} = 0.2030$, with 10 epochs $E_{\mathbf{f}_1} = 0.0046$, and $E_{\mathbf{f}_2} = 0.1513$, and with 20 epochs $E_{\mathbf{f}_1} = 0.0027$, and $E_{\mathbf{f}_2} = 0.1041$. Clearly, the estimation of $\mathbf{f}_1$ reaches high accuracy much faster than $\mathbf{f}_2$, whereas the estimation of $\mathbf{f}_2$ keeps on evolving for higher epochs.

Next, we used the 10-D five-class Gaussian data with 500 samples from each class. We used the first two classes in [24] with the covariance matrices multiplied by 20. The eigenvalues of $Q_1^{-1}Q_2$ are $\lambda_1 = 66.03$, $\lambda_2 = 34.28$, $\lambda_3 = 22.53$, $\lambda_4 = 11.52$, $\lambda_5 = 6.07$, $\lambda_6 = 0.72$, $\lambda_7 = 0.31$, $\lambda_8 = 0.14$, $\lambda_9 = 0.09$, $\lambda_{10} = 0.05$, with $\theta_1 = 66.04$, $\theta_2 = 34.31$, $\theta_3 = 22.58$, $\theta_4 = 11.60$, $\theta_5 = 6.24$, $\theta_6 = 2.11$, $\theta_7 = 3.54$, $\theta_8 = 7.56$, $\theta_9 = 11.80$, and $\theta_{10} = 21.84$. We trained the network for three epochs. The estimation errors for the first five feature vectors are $E_{\mathbf{f}_1} = 0.0632$, $E_{\mathbf{f}_2} = 0.1113$, $E_{\mathbf{f}_3} = 0.1202$, $E_{\mathbf{f}_4} = 0.0237$, and $E_{\mathbf{f}_5} = 0.0168$. Once, again, the estimation accuracy keeps on evolving for higher epochs.

VI. Concluding Remarks

In this study, we discussed a class of feature extraction problems that can be implemented with artificial neural networks by two sets of stochastic approximation algorithms: 1) the $Q^{-1/2}$ algorithm in (10) and 2) the PCA algorithm in (17). Note that in this study, we emphasized the methods of feature extraction, and not the techniques of classification with these features. Since neural networks are useful in classification, a separate network may be used for this task. We, therefore, used simple classifiers that are commonly used in pattern recognition [10], to test the feature extraction methods.

Our experiments (see Section V) indicate that the estimates converge "close" to their actual values in a finite number of samples, and the convergence plot (see Fig. 6) resembles an exponential rate. Although stochastic approximation theory gives us the asymptotic convergence results (as described in Section III), we ask a more practical question, on how fast does algorithm (10) converge for finite samples? The performance of stochastic approximation algorithms, such as (10), for finite samples is a subject of ongoing research [3], [19], and is beyond the scope of this study. However, stochastic approximation literature [3], [19] states that the asymptotic errors of the estimates have a zero-mean Gaussian distribution. Our ongoing research into this area shows that the variance of the Gaussian distribution is small for most practical pattern recognition applications. The related results will appear in a later publication.

The algorithms discussed here can be easily applied to a number of other criteria for feature extraction. Examples are the Chernoff and divergence criteria [10] for class separability. Most properties of these criteria can be discussed in terms

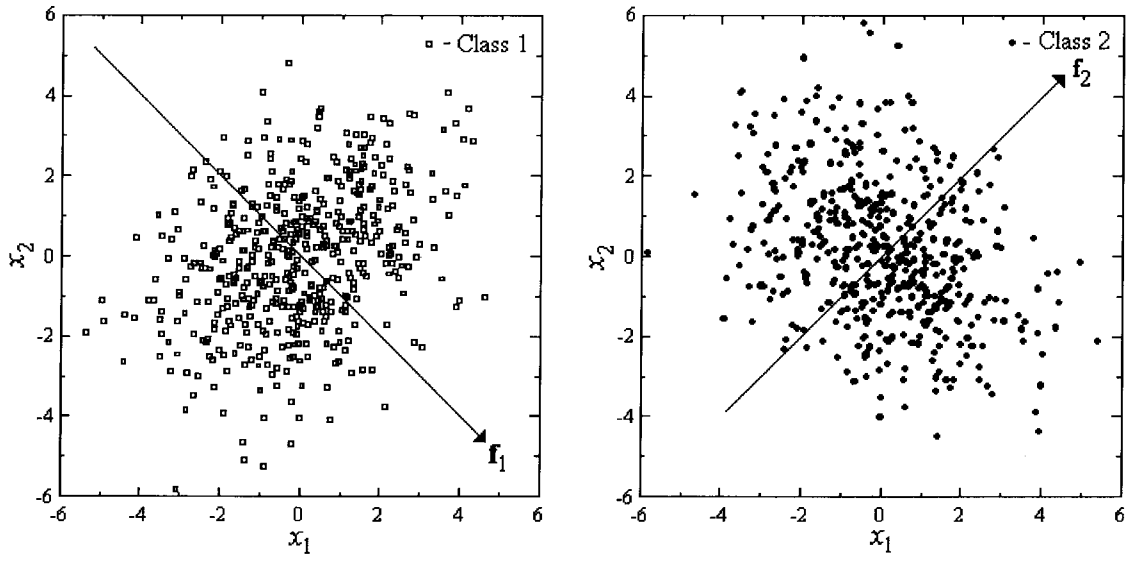

Fig. 12. Distributions of the two-class 2–D Gaussian data for Bhattacharyya distance features.

similar to those used for the Bhattacharyya distance criterion. For a two-class Gaussian distribution, the covariance term of the divergence criterion is $\mathrm{tr}[Q_1^{-1}Q_2+Q_2^{-1}Q_1-2I]$ similar to (9). Feature extraction with this criterion can be implemented by the methods discussed in Section IV-C.

Another application of the $Q^{-1/2}$ algorithm is feature extraction by a combination of the Karhunen-Loeve transform (KLT) and LDA [33]. In some applications, due to the high dimension of the input patterns, S_w may become singular and the LDA method fails. In such cases, a KLT of the inputs is first performed. Components of the transformed inputs corresponding to nonzero eigenvalues of the input correlation matrix are selected. This transformed data is used as inputs to the LDA network. This new transform, known as the discriminant Karhunen–Loeve (DKL) transform [33], can be realized by cascading an LDA network at the end of a PCA network. For S_w to be nonsingular in the projected space, at most $n-m$ components of the KLT are considered. Here n is the number of training samples from m classes.

APPENDIX

Proof of Lemma 4: We write (10) as

$$W(k+1)=W(k)+\eta(k)I-\eta(k)\mathbf{y}(k)\mathbf{y}(k)^T$$

where

$$\mathbf{y}(k)=W(k)\mathbf{x}(k).$$

We use the following identity [12]: Suppose $B=A+\tau\mathbf{c}\mathbf{c}^T$, where $A\in R^{d\times d}$ is symmetric, $c\in R^d$ and $\tau\in R$, then $\lambda_i(B)=\lambda_i(A)+\tau m_i\|\mathbf{c}\|^2$, where $\sum_{i=1}^d m_i=1$ and each $m_i\geq 0$. Moreover, if $\mathbf{c}$ has a nonzero projection on the eigenvector corresponding to $\lambda_i(A)$ then $m_i>0$.

For simplicity of notation, we shall use $\lambda_1(k)$ instead of $\lambda_1(W(k))$ for the rest of this analysis. Let $\mathbf{x}(k)$ have the projections $(c_1(k),\cdots,c_d(k))$ on the d eigenvectors of $W(k)$ in the order of decreasing eigenvalues. By the above identity, for each k, there exists $m(k)\in[0,1]$ such that

$$\lambda_1(k+1)=\lambda_1(k)+\eta(k)-\eta(k)m(k)\sum_{i=1}^{d}c_i(k)^2\lambda_i(k)^2. \quad (25)$$

By A4, for any set of d training vectors $\{\mathbf{x}(k),\cdots,\mathbf{x}(k+d-1)\}$, at least one $\mathbf{x}(p)$, $p\in\{k,\cdots,k+d-1\}$, has a nonzero projection on the eigenvector corresponding to $\lambda_1(k)$. Furthermore, the projection is bounded away from zero. Therefore, from (25), for training vectors $\{\mathbf{x}(k),\cdots,\mathbf{x}(k+d-1)\}$, for at least one vector $\mathbf{x}(p)$, we have

$$\lambda_1(k+d)=\lambda_1(k)+\sum_{i=k}^{k+d-1}\eta(i)-\eta(p)m(p)\sum_{i=1}^{d}c_i(p)^2\lambda_i(p)^2. \quad (26)$$

Define an index set $I=\{i: m(i)c_1(i)^2>0\}$. Then by A4, there exists an $\alpha^2>0$, such that $\alpha^2=\inf_{i\in I}\{m(i)c_1(i)^2\}$. From (26), $\lambda_1(k+d)<\lambda_1(k)$ if $\lambda_1(p)^2>\frac{\sum_{i=k}^{k+d-1}\eta(i)}{\eta(p)\alpha^2}$. Using the common choice of $\eta(k)\propto k^{-\delta}$ for $0<\delta\leq 1$, we obtain $\frac{\sum_{i=k}^{k+d-1}\eta(i)}{\eta(p)}<d^2$. Similar bounds can be obtained for other choices of $\eta(k)$. Therefore, $\lambda_1(k+d)<\lambda_1(k)$ if $\lambda_1(p)>\frac{d}{\alpha}$. Note that if $\lambda_1(k)<\frac{d}{\alpha}$, then the maximum value for $\lambda_1(k+d)$, from (26), is $\lambda_1(k)+\sum_{i=k}^{k+d-1}\eta(i)$. Therefore, if $\lambda_1(0)<\frac{d}{\alpha}$, then we have

$$\lambda_1(k)<d\eta(0)+\frac{d}{\alpha}\quad\text{for all } k>0. \quad (27)$$

□

Proof of Lemma 5: We shall prove by induction. We start (10) with $W(0)$ symmetric and nonnegative definite. Clearly, $W(k)$ remains symmetric for all k. Define $G(k)=W(k)\mathbf{x}(k)\mathbf{x}(k)^TW(k)$. If for some $k, W(k)\geq 0$ and $I-G(k)\geq 0$ then $W(k+1)\in D(W_+^*)$. However, if for some $k, W(k)\geq 0$ and $I-G(k)$ is indefinite, then we need to make sure that $W(k+1)=W(k)+\eta(k)(I-G(k))\geq 0$.

Let us consider the case where $G(k)$ has the maximum effect on $\lambda_d(k)$. This happens when $\mathbf{x}(k)$ is oriented along the eigenvector corresponding to $\lambda_d(k)$. Then, $||W(k)\mathbf{x}(k)||^2 = ||\mathbf{x}(k)||^2\lambda_d(k)^2$. By the identity in Lemma 4

$$\lambda_d(k+1) = \lambda_d(k) + \eta(k)\big(1 - ||\mathbf{x}(k)||^2\lambda_d(k)^2\big)$$

where $||\mathbf{x}(k)||^2\lambda_d(k)^2 > 1$ by hypothesis. From the above equation, $\lambda_d(k+1) > 0$ if

$$\eta(k) < f(\lambda_d(k)), \quad \text{where } f(\lambda_d(k)) = \frac{\lambda_d(k)}{||\mathbf{x}(k)||^2\lambda_d(k)^2 - 1}. \tag{28}$$

Let ρ be the uniform upper bound of $\lambda_1(k)$ in Lemma 4. In (28), a tighter upper bound for $\eta(k)$ is obtained by minimizing $f(\lambda_d)$ under constraint C: $||\mathbf{x}(k)||^2\lambda_d(k)^2 > 1$. Within $C, f(\lambda_d)$ is a monotonically decreasing function of λ_d, whose minimum is reached for $\lambda_d = \rho$. Further note that $||\mathbf{x}(k)||^2 < \beta^2$ by Assumption A1. Therefore, a uniform upper bound for $\eta(k)$ is

$$\eta(k) < \frac{\rho}{\rho^2\beta^2 - 1} \approx \frac{\alpha}{d\beta^2} \text{ for all } k. \tag{29}$$

□

Proof of Theorem 2: For simplicity, we shall drop the subscript i in the proof. We see that $||\mathbf{y}(k)|| \le ||\mathbf{x}(k)|| + ||\mathbf{m}(k)|| < 2\beta$, where $||\mathbf{x}(k)|| < \beta$. Therefore, $\{\mathbf{y}(k)\}$ is uniformly bounded. Notice that

$$\begin{aligned} &E[(\mathbf{x}(k) - \mathbf{m}(k))(\mathbf{x}(k) - \mathbf{m}(k))^T] \\ &\quad = E[(\mathbf{x}(k) - \mathbf{m})(\mathbf{x}(k) - \mathbf{m})^T] \\ &\qquad + E[(\mathbf{x}(k) - \mathbf{m})(\mathbf{m} - \mathbf{m}(k))^T] \\ &\qquad + E[(\mathbf{m} - \mathbf{m}(k))(\mathbf{x}(k) - \mathbf{m})^T] \\ &\qquad + [(\mathbf{m} - \mathbf{m}(k))(\mathbf{m} - \mathbf{m}(k))^T]. \end{aligned}$$

Since by assumption B1 $\mathbf{x}(k)$ is bounded, we have $\mathbf{m}(k)$ and $\mathbf{m}$ bounded. Due to w.p.1 convergence of $\mathbf{m}(k)$ to $\mathbf{m}$, we have

$$\begin{aligned} &\lim_{k\to\infty} E[(\mathbf{x}(k) - \mathbf{m})(\mathbf{m} - \mathbf{m}(k))^T] = 0 \\ &\lim_{k\to\infty} E[(\mathbf{m} - \mathbf{m}(k))(\mathbf{x}(k) - \mathbf{m})^T] = 0 \\ &\lim_{k\to\infty} E[(\mathbf{m} - \mathbf{m}(k))(\mathbf{m} - \mathbf{m}(k))^T] = 0. \end{aligned}$$

By B1, $\lim_{k\to\infty} E[(\mathbf{x}(k) - \mathbf{m})(\mathbf{x}(k) - \mathbf{m})^T] = Q$. Thus, $\{\mathbf{y}(k)\}$ satisfies assumption A1.

Next, we need to show that $\{\mathbf{y}(k)\}$ satisfies A3. Note that $\{\mathbf{y}(k)\}$ is generated by the following linear process:

$$\begin{aligned} \tilde{\mathbf{y}}(k) &= (1 - \delta(k))\tilde{\mathbf{y}}(k-1) + \delta(k)\mathbf{x}(k) \\ \mathbf{y}(k) &= -\tilde{\mathbf{y}}(k) + \mathbf{x}(k). \end{aligned} \tag{30}$$

If $\{\mathbf{x}(k)\}$ is generated by (11), then $\{\mathbf{y}(k)\}$ is generated by the following linear processes:

$$\begin{aligned} \begin{bmatrix} \tilde{\mathbf{x}}(k) \\ \tilde{\mathbf{y}}(k) \end{bmatrix} &= \begin{bmatrix} A(k) & 0 \\ \delta(k)C(k)A(k) & (1 - \delta(k))I \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{x}}(k-1) \\ \tilde{\mathbf{y}}(k-1) \end{bmatrix} \\ &\quad + \begin{bmatrix} B(k) \\ \delta(k)C(k)B(k) \end{bmatrix} \mathbf{e}(k) \\ \mathbf{y}(k) &= [C(k) \quad -I] \begin{bmatrix} \tilde{\mathbf{x}}(k) \\ \tilde{\mathbf{y}}(k) \end{bmatrix}. \end{aligned} \tag{31}$$

Since $0 < \delta(k) < 1$, we see that $\{\mathbf{y}(k)\}$ is stable and regular if $\{\mathbf{x}(k)\}$ is stable and regular (which is assumed in A3). Thus, $\{\mathbf{y}(k)\}$ satisfies (11) and assumption A3. A similar result can be shown for $\{\mathbf{x}(k)\}$ generated by (12). The result follows from Theorem 1. □

Proof of Theorem 3: The proof of this theorem can be considered as an application of the bounded convergence theorem [34]. By Theorem 2, under assumptions B1 and B2, $W(k) \to S_w^{-1/2}$ w.p.1 as $k \to \infty$. Therefore, $W(k) = S_w^{-1/2} + F(k)$, where the random matrix $F(k) \to 0$ w.p.1 as $k \to \infty$. Thus,

$$\begin{aligned} &\mathbf{u}(k)\mathbf{u}(k)^T \\ &\quad = \big(S_w^{-1/2} + F(k)\big)\mathbf{z}(k)\mathbf{z}(k)^T\big(S_w^{-1/2} + F(k)\big) \\ &\quad = S_w^{-1/2}\mathbf{z}(k)\mathbf{z}(k)^T S_w^{-1/2} + F(k)\mathbf{z}(k)\mathbf{z}(k)^T F(k) \\ &\qquad + S_w^{-1/2}\mathbf{z}(k)\mathbf{z}(k)^T F(k) + F(k)\mathbf{z}(k)\mathbf{z}(k)^T S_w^{-1/2}. \end{aligned}$$

We shall prove that $E[F(k)\mathbf{z}(k)\mathbf{z}(k)^T F(k)] \to 0$ as $k \to \infty$. Since $W(k)$ is uniformly bounded by Lemma 4, let $||F(k)|| \le \rho$. Since $\mathbf{x}(k)$ is bounded, let $||\mathbf{z}(k)|| \le \kappa$. For any $\varepsilon > 0$

$$E[||F(k)\mathbf{z}(k)||^2] \le \kappa\rho(\kappa\varepsilon P(||F(k)|| \le \varepsilon) + \kappa\rho P(||F(k)|| > \varepsilon)).$$

Due to the w.p.1 convergence of $W(k)$, there exists a positive integer $K(\varepsilon)$, such that whenever $k > K(\varepsilon), P(||F(k)|| > \varepsilon) \le \varepsilon$, and

$$E[||F(k)\mathbf{z}(k)||^2] \le \kappa\rho(\kappa\varepsilon + \kappa\rho\varepsilon) \to 0 \text{ since } \varepsilon \text{ is arbitrary.}$$

This implies that $E[F(k)\mathbf{z}(k)\mathbf{z}(k)^T F(k)] \to 0$ w.p.1 as $k \to \infty$. Similar analysis for the remaining terms gives us

$$\begin{aligned} \lim_{k\to\infty} E[\mathbf{u}(k)\mathbf{u}(k)^T] &= S_w^{-1/2} \lim_{k\to\infty} E[\mathbf{z}(k)\mathbf{z}(k)^T] S_w^{-1/2} \\ &= S_w^{-1/2} S_m S_w^{-1/2} \end{aligned}$$

where $\lim_{k\to\infty} E[\mathbf{z}(k)\mathbf{z}(k)^T] = S_m$. Further note that $||\mathbf{u}(k)|| = ||W(k)\mathbf{z}(k)|| \le ||W(k)||||\mathbf{z}(k)|| \le \rho\kappa$. Thus, $\{\mathbf{u}(k)\}$ satisfies A1. By the convergence proof of Sanger's algorithm [30], $V(k)$ tends to the matrix whose columns are the eigenvectors of $S_w^{-1/2} S_m S_w^{-1/2}$ ordered by decreasing eigenvalue. □

REFERENCES

[1] B. D. O. Anderson and J. B. Moore, *Optimal Control—Linear Quadratic Methods*. Englewood Cliffs, NJ: Prentice-Hall, 1990.

[2] 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.

[3] A. Benveniste, A. Metivier, and P. Priouret, *Adaptive Algorithms and Stochastic Approximations*. New York: Springer-Verlag, 1990.

[4] G. Birkhoff and G.-C. Rota, *Ordinary Differential Equations*, 2nd ed. London, U.K.: Blaisdell, 1969.

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

[6] K. A. Brakke, J. M. Mantock, and K. Fukunaga, "Systematic feature extraction," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-4, pp. 291–297, 1982.

[7] P. A. Devijver, "Relationship between statistical risks and the least-mean-square error design criterion in pattern recognition," in *Proc. 1st Int. Joint Conf. Pattern Recognition*, Washington, D.C., 1973, pp. 139–148.

[8] P. Foldiak, "Adaptive network for optimal linear feature extraction," in *Proc. IJCNN*, Washington, 1989, pp. 401–405.

[9] D. H. Foley and J. W. Sammon, "An optimal set of discriminant vectors," *IEEE Trans. Computers*, vol. C-24, pp. 281–289, Mar. 1975.
[10] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd ed. New York: Academic, 1990.
[11] P. Gallinari, S. Thiria, F. Badran, and F. Fogelman-Soulie, "On the relations between discriminant analysis and multilayer perceptrons," *Neural Networks*, vol. 4, pp. 349–360, 1991.
[12] G. H. Golub and C. F. VanLoan, *Matrix Computations*. Baltimore, MD: Johns Hopkins Univ. Press, 1983.
[13] S. Haykin, *Neural Networks—A Comprehensive Foundation*. New York: Macmillan, 1994.
[14] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*. Reading, MA: Addison-Wesley, 1991.
[15] A. K. Jain and J. Mao, "Artificial neural network for nonlinear projection of multivariate data," in *Proc. Int. Joint Conf. Neural Networks*, Baltimore, MD, vol. 3, June 1992.
[16] S. Y. Kung and K. I. Diamantaras, "A neural network learning algorithm for adaptive principal component extraction (APEX)," *Int. Conf. Acoust., Speech. Signal Procesing*, Albuquerque, NM, 1990, pp. 861–864.
[17] H. J. Kushner and D. S. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. New York: Springer-Verlag, 1978.
[18] L. Ljung, "Analysis of recursive stochastic algorithms," *IEEE Trans. Automat. Contr.*, vol. AC-22, pp. 551–575, Aug. 1977.
[19] L. Ljung, G. Pflug, and H. Walk, *Stochastic Approximation and Optimization of Random Systems*. Boston, MA: Birkhauser, 1992.
[20] J. Mao and A. K. Jain, "Discriminant analysis neural networks," *IEEE Int. Conf. Neural Networks*, San Francisco, CA, vol. 1, Mar. 1993, pp. 300–305.
[21] B. K. Moor, "ART 1 and pattern clustering," in *Proc. 1988 Connectionist Summer School*. San Mateo, CA: Morgan Kaufmann, 1988, pp. 174–185.
[22] E. Oja and J. Karhunen, "On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix," *J. Math. Anal. Applicat.*, vol. 106, pp. 69–84, 1985.
[23] E. Oja, "Principal components, minor components, and linear neural networks," *Neural Networks*, vol. 5, pp. 927–935, 1992.
[24] T. Okada and S. Tomita, "An optimal orthonormal system for discriminant analysis," *Pattern Recognition*, vol. 18, no. 2, pp. 139–144, 1985.
[25] N. R. Pal, J. C. Bezdek, and E. C.-K. Tsao, "Generalized clustering networks and Kohonen's self-organizing scheme," *IEEE Trans. Neural Networks*, vol. 4, pp. 549–557, 1993.
[26] M. D. Richard and R. P. Lippmann, "Neural network classifiers estimate Bayesian *a posteriori* probabilities," *Neural Computa.*, vol. 3, pp. 461–483, 1991.
[27] J. Rubner and K. Schulten, "Development of feature detectors by self-organization—A network model," *Biol. Cybern.*, vol. 62, pp. 193–199, 1990.
[28] J. Rubner and P. Tavan, "A self-organizing network for principal component analysis," *Europhys. Lett.*, vol. 10, no. 7, pp. 693–698, 1989.
[29] D. W. Ruck, S. K. Rogers, M. Kabrisky, M. E. Oxley, and B. W. Suter, "The multilayer perceptron as an approximation to a Bayes optimal discriminant function," *IEEE Trans. Neural Networks*, vol. 1, pp. 296–298, 1990.
[30] T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network," *Neural Networks*, vol. 2, pp. 459–473, 1989.
[31] E. A. Wan, "Neural network classification: A Bayesian interpretation," *IEEE Trans. Neural Networks*, vol. 1, pp. 303–305, Dec. 1990.
[32] A. R. Webb and D. Lowe, "The optimised internal representation of multilayer classifier networks performs nonlinear discriminant analysis," *Neural Networks*, vol. 3, pp. 367–375, 1990.
[33] J. Weng, "On comprehensive visual learning," in *Proc. NSF/ARPA Wkshp. Performance Versus Methodology in Computer Vision*, Seattle, WA, June 1994, pp. 152–166.
[34] R. L. Wheeden and A. Zygmund, *Measure and Integral—An Introduction to Real Analysis*. New York: Marcel Dekker, 1977.
[35] H.-C. Yau and M. T. Manry, "Iterative improvement of a Gaussian classifier," *Neural Networks*, vol. 3, pp. 437–443, 1990.

Chanchal Chatterjee received the B.Tech degree in electrical engineering from the Indian Institute of Technology, Kanpur, India, and the M.S.E.E. degree and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, in 1983 and 1984, and 1996, respectively.

Between 1985 and 1995 he worked at Machine Vision International and Medar Inc., both in Detroit, MI. He is currently a Senior Scientist with Newport Corporation, Irvine, CA. He is also affiliated with the Department of Electrical Engineering at the University of California at Los Angeles. His interests include image processing, computer vision, neural networks, and adaptive algorithms and systems for pattern recognition and signal processing.

Vwani P. Roychowdhury received the B.Tech. degree from the Indian Institute of Technology, Kanpur, India, and the Ph.D. degree from Stanford University, Stanford, CA, in 1982 and 1989, respectively, both in electrical engineering.

From September 1989 to August 1991 he was a Research Associate in the Department of Electrical Engineering at Stanford University. From August 1991 to June 1996, he was a Faculty Member at the School of Electrical and Computer Engienering at Purdue University, West Lafayette, IN. He is currently a Professor in the Department of Electrical Engineering at the University of California at Los Angeles. His research interests include parallel algorithms and architectures, design and analysis of neural networks, computational principals to nanoelectronics, special-purpose computing arrays, VLSI design, and fault-tolerant computation. He has coauthored several books, including *Discrete Neural computation: A Theoretical Foundation* (Englewood Cliffs, NJ: Prentice-Hall, 1995) and *Theoretical Advances in Neural Computation and Learning* (Boston, MA: Kluwer, 1994).