ALDA Algorithms for Online Feature Extraction

Youness Aliyari Ghassabeh, Hamid Abrishami Moghaddam

Abstract

In this paper, we present new adaptive algorithms for the computation of the square root of the inverse covariance matrix. In contrast to the current similar methods, these new algorithms are obtained from an explicit cost function that is introduced for the first time. The new adaptive algorithms are used in a cascade form with a well known adaptive principal component analysis (APCA) to construct linear discriminant features. The adaptive nature and fast convergence rate of the new adaptive linear discriminant analysis (ALDA) algorithms make them appropriate for on-line pattern recognition applications. All adaptive algorithms discussed in this paper are trained simultaneously using a sequence of random data. Experimental results using the synthetic and real multi-class, multi-dimensional input data demonstrate the effectiveness of the new adaptive algorithms to extract the optimal features for the classification purpose.

Index Terms

Adaptive Linear Discriminant Analysis, Adaptive Principal Component Analysis, Optimal Feature Extraction, Fast Convergence Rate.

I. Introduction

Feature extraction consists of choosing those features which are most effective for preserving class separability in addition to the dimensionality reduction [1]. Linear discriminant analysis (LDA) has been widely used as a feature extraction method in pattern recognition applications such as face and gesture recognition [2] [3], mobile robotics [4] [5], and hyper-spectral image analysis [6] as well as some data mining and knowledge discovery applications [7] [8]. Principal component analysis (PCA), that seeks efficient directions for the representation of data, has also been used for dimensionality reduction. Unlike the criterion for the data representation (as used by PCA), the class-separability criterion are independent of the coordinate systems and depend on the class distribution and the classifier. It is generally believed that when it comes to solve problems of the pattern classification, LDA-based algorithms outperform PCA-based ones [9]; since the former optimize the low dimensional representation of the objects and focus on the most discriminant features, while the latter achieve simply object reconstruction [10] [11]. For typical implementation of these two techniques, in general it is assumed that a complete data set for the training phase is available and learning is carried out in one batch. However, when PCA/LDA algorithms are used over data sets in real world applications, we confront difficult situations where a complete set of the training samples is not given in advance. For example, in applications such as on-line face recognition and mobile robotics, data are presented as a stream and the entire data is not available beforehand. Therefore, the need for the dimensionality reduction in real time applications motivated researchers to introduce adaptive versions of PCA and LDA.

Hall et al. [12] proposed incremental PCA (IPCA) by updating the covariance matrix through the training procedure. In a later work, Hall et al. [13] improved their algorithm by proposing a method of merging and splitting eigen-space models that allows a chunk of new samples to be learned in a single step. Miao and Hua [14] used an objective function and presented gradient descent and recursive least squares algorithms [15] for adaptive principal subspace analysis (APSA). Xu [16] used a different objective function to derive an algorithm for PCA by applying the gradient descent optimization method. Later, Weng et al. [17] proposed a fast IPCA approximation algorithm for the incremental computation of the principal components of a sequence of samples without estimating the covariance matrix. Oja and Karhunen [18] [19] and Sanger [20] derived an algorithm for the adaptive computation of the PCA features, using the generalized Hebbian learning. In another work, Chatterjee et al. [21] derived accelerated versions of PCA that converge faster than the previous PCA algorithms given by Oja [18] [19], Sanger [20], and Xu [16].

Pang et al. [22] presented a constructive method for updating the LDA eigen-space. They derived incremental algorithms for updating the between-class and within-class scatter matrices using the sequential or chunk input data. Authors in [23] applied the concept of the sufficient spanning set approximation for updating the mixed and between-class scatter matrices. A recently introduced incremental dimension reduction (IDR) algorithm uses QR decomposition method to obtain adaptive algorithms for the computation of the reduced forms of within-class and between-class scatter matrices [24]. Although, the methods introduced in [22] [23] [24] update the scatter matrices in a sequential manner, they do not give any solution for the adaptive computation of the LDA features. In other words, the new incoming input samples only update the scatter matrices and the desired LDA features are not updated directly using the incoming input samples. For adaptive LDA (ALDA), Mao and Jain [25] proposed a two layer network, each of which was a PCA network. Chatterjee and Roychowdhury [26] presented adaptive algorithms and a self-organized LDA network for the incremental feature extraction. They described algorithms and networks for the linear discriminant analysis in multi-class case. Method presented in [26] has been obtained without using an explicit cost function and suffers from the low convergence rate. In another work, Demir and Ozmehmet [27] proposed on-line local learning algorithms for ALDA using the error correcting and the Hebbian learning rules. Similar to the algorithm given in [26], the main drawback of the method in [27] is dependency to the step size that is difficult to set a priori. Abrishami Moghaddam et al. [28] [29] accelerate the convergence rate of ALDA based on the steepest descent, conjugate direction, Newton-Raphson, and quasi-Newton methods. However, the authors in [28] [29] used an implicit cost function to obtain their adaptive algorithms and no convergence analysis was given and convergence of the proposed algorithms have not been guaranteed. Y. Rao et al. [30] introduced a fast RLS like algorithm for the generalized eigen-decomposition problem. Although they solved the problem of the optimization of the learning rate, their method suffers from the errors caused by the adaptive estimation of the inverse within-class covariance matrix. In the algorithm given in [30], it is not possible to update the desired number of LDA features in one step. Instead, estimation of the LDA features is done sequentially; the most significant LDA feature is estimated first, then using that feature and the incoming input sample the second LDA feature is estimated and this procedure is repeated for the rest of the LDA features.

In this study, we present new adaptive learning algorithms for the computation of the squared root of the inverse covariance matrix $\mathbf{\Sigma}^{-1/2}$. The square root of the inverse of the covariance matrix can be used directly to extract the optimal features from the Gaussian data [31]. As a more general application of the square root of the inverse covariance matrix $\mathbf{\Sigma}^{-1/2}$, we show that it can be used for the LDA feature extraction. We combine the proposed $\mathbf{\Sigma}^{-1/2}$ algorithms with an adaptive PCA algorithm for ALDA. In contrast to the methods in [26] [27] [28] [29], the new adaptive algorithms are derived by optimization of a new explicit cost function. Moreover, a new accelerated adaptive algorithm for ALDA is obtained by optimizing the learning rate in each iteration. Introduced methodology is novel in the following ways:

- New adaptive algorithms for the computation of the square root of the inverse covariance matrix $\mathbf{\Sigma}^{-1/2}$ are derived by minimization of a cost function that is introduced for the first time. In contrast to the previous methods, the convergence analysis of the proposed algorithms using the cost function is straightforward.
- Five new algorithms are derived for the adaptive computation of the square root of the inverse covariance matrix $\mathbf{\Sigma}^{-1/2}$ with the similar convergence rate of the algorithm given in [26].
- New fast convergent version of ALDA is derived by optimizing the learning rate in each iteration. In contrast to [28] [29], the convergence of the new accelerated algorithms is proved and guaranteed using the differentiability of the cost function with respect to the learning rate.
- The accuracy of the LDA features obtained using the new adaptive algorithms can be evaluated in each iteration using a criterion based on the cost function. Availability of such criterion is a major advantage of the proposed method comparing to the techniques proposed in [26] [27] [28] [29] [30], that have not provided any criteria for evaluation of the final estimates.
- In contrast to the algorithm presented in [30], which requires an adaptive estimation of the inverse of the within-class scatter matrix, for the new proposed algorithms no matrix inversion is involved that results smaller error in the extracted features compared to the results in [30].
- In contrast to the algorithm presented in [30], the desired number of the LDA features can be estimated simultaneously in one step.

The adaptive nature of the algorithms discussed in this paper prevents keeping a large volume of data in memory and instead a sequence of the input samples has been considered for the training phase. The memory size reduction and the fast convergence rate provided by the new ALDA algorithms make them appropriate for on-line pattern recognition applications [32] [33]. The effectiveness of these new adaptive algorithms for extracting the LDA features has been tested during different on-line experiments with stationary, non-stationary, synthetic, and real world input samples.

This paper is organized as follows, section II gives a brief review of the fundamentals of the PCA and the LDA algorithms. Section III presents the new adaptive algorithms for estimation of the square root of the inverse covariance matrix $\mathbf{\Sigma}^{-1/2}$ and analyzes their convergence by introducing the related cost function. The new ALDA algorithms are implemented by combining these algorithms with an adaptive PCA (APCA) in a cascade form. The convergence proof is given for each algorithm. Section IV is devoted to simulations and experimental results. Finally, concluding remarks the future works are discussed in section V.

II. Feature Extraction Algorithms

There are many techniques for feature extraction in the literature, among them the principal component analysis (PCA) and the linear discriminant analysis (LDA) are two widely used techniques for data classification and dimensionality reduction. In this section, we give a brief review of the fundamentals and the current adaptive versions of these two algorithms.

A. Principal Component Analysis

The PCA, also known as the Karhunen-Loeve transform, is a popular linear dimensionality reduction technique. The PCA algorithm is a procedure that linearly transforms correlated variables into uncorrelated variables called principal components such that the first principal component has the maximum variance, the second has the maximum variance under the constraint that it is uncorrelated with the first one, and so on. The PCA is used to represent multidimensional data sets in a linear lower dimensional subspace. It has been shown that using the PCA algorithm minimizes the average reconstruction error.

Let $\mathbf{\Phi}_{PCA}^{p}$ denote a linear $n \times p$ transformation matrix that maps the original n dimensional space into a p dimensional feature subspace, where $p < n$. The new feature vectors $\mathbf{y}_k \in \mathbb{R}^p, k = 1, \ldots, N$ are obtained by [34]

$$\mathbf{y}_k = (\mathbf{\Phi}_{PCA}^{p})^t \mathbf{x}_k, k = 1, \ldots, N, \tag{1}$$

where $\mathbf{x}_k \in \mathbb{R}^n, k = 1, \ldots, N$ represent the input samples. It has been shown that if the columns of $\mathbf{\Phi}_{PCA}^{p}$ are the eigenvectors of the input samples' covariance matrix $\mathbf{\Sigma}$, corresponding to its p largest eigenvalues in decreasing order, then the optimum feature space for the representation of data is achieved. Different adaptive methods have been proposed for on-line estimation of the transformation matrix. The following method was proposed by Oja and Karhunen [18] [19] and Sanger [20]

$$\mathbf{T}_{k+1} = \mathbf{T}_k + \gamma_k(\mathbf{y}_k \mathbf{x}_k - LT(\mathbf{y}_k \mathbf{y}_k^t)\mathbf{T}_k), \tag{2}$$

where $\mathbf{y}_k = \mathbf{T}_k \mathbf{x}_k$ and $\mathbf{T}_k$ is a $p \times n$ matrix that converges to a matrix $\mathbf{T}$ whose rows are the first p eigenvectors of the covariance matrix $\mathbf{\Sigma}$. The operator $LT[.]$ sets the element of the above of the main diagonal of its input argument to zero, γ_k is the learning rate which meets Ljung's conditions [36], and $\mathbf{x}_k$ represent the data samples, respectively.

B. Linear Discriminant Analysis

The LDA searches the directions for the maximum discrimination of classes in addition to the dimensionality reduction. To achieve this goal, within-class and between-class scatter matrices are defined. If the total number of classes are K, then a within-class scatter matrix $\mathbf{\Sigma}_W$ is the scatter of the samples around their respective class mean $\mathbf{m}_i, i = 2, \ldots, K$ and is given by

$$\mathbf{\Sigma}_W = \sum_{i=1}^{K} P(\omega_i) E[(\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^t | \mathbf{x} \in \omega_i] = \sum_{i=1}^{K} P(\omega_i) \mathbf{\Sigma}_i, \tag{3}$$

where $\mathbf{\Sigma}_i$ and $P(\omega_i), i = 1, \ldots, K$ denote the covariance matrix and the probability of the i-th class ω_i, respectively. The between-class scatter matrix $\mathbf{\Sigma}_B$ is the scatter of class means $\mathbf{m}_i$ around the mixture mean $\mathbf{m}$, and is given by

$$\mathbf{\Sigma}_B = \sum_{i=1}^{K} P(\omega_i)(\mathbf{m} - \mathbf{m}_i)(\mathbf{m} - \mathbf{m}_i)^t. \tag{4}$$

The mixture scatter matrix is the covariance of all samples regardless of class assignments and we have

$$\mathbf{\Sigma} = \mathbf{\Sigma}_W + \mathbf{\Sigma}_B. \tag{5}$$

To achieve a high class separability, all features belonging to the same class have to be close together and well separated from the features of other classes. This implies that the transformed data in the LDA feature space must have a relatively small within-class scatter matrix and a large between-class scatter matrix. Different objective functions have been used as LDA criteria mainly based on a family of functions of scatter matrices. For example, the widely used class separability measure functions are $tr(\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B)$, $det(\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B)$, and $\mathbf{\Sigma}_B/\mathbf{\Sigma}_W$ [35]. It has been shown that for the LDA, the optimal linear transformation matrix is composed of $p(< n)$ eigenvectors of $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B$, corresponding to its p largest eigenvalues. In general, the between-class scatter matrix $\mathbf{\Sigma}_B$ is not a full rank matrix and therefore is not a covariance matrix. Since both $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B$ and $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}$ have the same eigenvectors [1], the between-class scatter matrix $\mathbf{\Sigma}_B$ can be replaced by the covariance matrix $\mathbf{\Sigma}$. Let $\mathbf{\Phi}_{LDA}$ denote the matrix whose columns are eigenvectors of $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B$. The computation of the LDA eigenvector matrix $\mathbf{\Phi}_{LDA}$ is equivalent to the solution of the generalized eigenvalue problem $\mathbf{\Sigma}\mathbf{\Phi}_{LDA} = \mathbf{\Sigma}_W\mathbf{\Phi}_{LDA}\mathbf{\Lambda}$, where $\mathbf{\Lambda}$ is the generalized eigenvalue matrix. Under assumption of the positive definiteness of $\mathbf{\Sigma}_W$, there exists a symmetric matrix $\mathbf{\Sigma}_W^{-1/2}$ such that the above problem can be reduced to the following symmetric eigenvalue problem

$$\mathbf{\Sigma}_W^{-1/2}\mathbf{\Sigma}\mathbf{\Sigma}_W^{-1/2}\mathbf{\Psi} = \mathbf{\Psi}\mathbf{\Lambda}, \tag{6}$$

where $\mathbf{\Psi} = \mathbf{\Sigma}_W^{1/2}\mathbf{\Phi}_{LDA}$. The adaptive versions of the LDA algorithm have been proposed for on-line feature extraction [26] [27] [28] [29] [30]. Chatterjee and Roychowdhurry [26] have proposed a two layers network in which the first layer is used for the estimation of the square root of the inverse covariance matrix and is trained using the following adaptive equation

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k(\mathbf{I} - \mathbf{W}_k\mathbf{x}_{k+1}\mathbf{x}_{k+1}^t\mathbf{W}_k^t), \tag{7}$$

where $\mathbf{W}_{k+1}$ represents estimation of the square root of the inverse covariance matrix at $k+1$-th iteration, $\mathbf{x}_{k+1}$ is the $k+1$-th input training vector and η_k is the learning rate that meets Ljung's conditions [36]. The second layer is an APCA network and trained using (2). The authors in [26] have not introduced any cost function for the adaptive equation given in (7); hence they used the stochastic approximation theory in order to prove the convergence of the proposed algorithm. The main drawback of the method introduced in [26] is its dependency to the step size. In other words, there is no way to find the optimal step sizes in each iteration and the algorithm suffers from the low convergence rate. In addition, the non-existence of the cost function prevents to evaluate the accuracy of the final estimates. In a later work Rao et al. [30] suggested the following iterative algorithm for extracting the most significant eigenvector

$$\mathbf{w}_{k+1} = \frac{\mathbf{w}_k^t\mathbf{\Sigma}_{W,k+1}\mathbf{w}_k}{\mathbf{w}_k^t\mathbf{\Sigma}_{B,k+1}\mathbf{w}_k}\mathbf{\Sigma}_{W,k+1}^{-1}\mathbf{\Sigma}_{B,k+1}\mathbf{w}_k, \tag{8}$$

where $\mathbf{\Sigma}_{W,k+1}$ and $\mathbf{\Sigma}_{B,k+1}$ are estimates of the within-class and between-class scatter matrices at $k+1$-th iteration and $\mathbf{w}_{k+1}$ represent the estimate of the most significant LDA feature at $k+1$-th iteration. For the minor components, authors in [30] resort to the deflation technique [37] and using the deflation procedure, second, third and other eigenvectors are estimated. Although (8) is not dependent to the step size but it uses estimates of $\mathbf{\Sigma}_W$, $\mathbf{\Sigma}_B$, and $\mathbf{\Sigma}_W^{-1}$ matrices that will increase the estimation error and complexity. The first eigenvector is estimated from (8) and the rest of the eigenvectors are estimated using equations derived by the deflation technique. This procedure prevents us to compute the desired number of the eigenvectors simultaneously and instead the eigenvectors have to be estimated sequentially. Furthermore, none of authors in [26] [27] [28] [29] [30] proposed any criterion to evaluate the accuracy of the final estimates resulting from different initial values or learning rates; therefore it is not possible to determine which estimation is more accurate compared to others.

Introducing a cost function for the proposed adaptive algorithms in this paper has the following advantages: i) it simplifies the convergence analysis; ii) it helps to evaluate the accuracy of the current estimates. For example, in the cases of different initial conditions and various learning rates, one can evaluate estimate resulting from which initial condition and learning rate outperform the others; iii) the adaptive method given in (7) uses a fixed or monotonically decreasing learning rate which results in low convergence rate. Existence of the cost function makes it possible to find the optimal learning rates in each iteration in order to accelerate the convergence rate.

III. New Training Algorithms for ALDA

We use two adaptive algorithms in a cascade form to extract the optimal LDA features adaptively. The first one, called $\mathbf{\Sigma}^{-1/2}$ algorithm, computes the square root of the inverse covariance matrix and is derived by minimization of a relevant cost function using the gradient descent method. The second algorithm is an APCA and computes the eigenvectors of the covariance matrix using (2). We show the convergence of the cascade architecture as an ALDA feature selection tool. Furthermore, we introduce a fast convergence version of the proposed ALDA algorithms.

A. Adaptive Estimation of $\mathbf{\Sigma}^{-1/2}$ and Convergence Proof

Let the cost function $J(\mathbf{W}) : \mathbb{R}^{n\times n} \to \mathbb{R}$ be

$$J(\mathbf{W}) = \frac{1}{3} tr\left[(\mathbf{W}\mathbf{\Sigma}^{1/2} - \mathbf{I})^2(\mathbf{W} + 2\mathbf{\Sigma}^{-1/2})\right], \tag{9}$$

where $tr[.]$ computes trace of its input matrix, $\mathbf{W} \in \mathbb{R}^{n\times n}$ is a symmetric and positive definite matrix satisfying $\mathbf{W}\mathbf{\Sigma}^{1/2} = \mathbf{\Sigma}^{1/2}\mathbf{W}$, and $\mathbf{I}$ is the identity matrix. In (9), both terms $(\mathbf{W}\mathbf{\Sigma}^{1/2} - \mathbf{I})^2$ and $(\mathbf{W} + 2\mathbf{\Sigma}^{-1/2})$ are positive semi-definite matrices; hence, the trace of their product is nonnegative [38]. The minimum value of the cost function in (9) is zero and occurs at $\mathbf{W} = \mathbf{\Sigma}^{-1/2}$ and $\mathbf{W} = -2\mathbf{\Sigma}^{-1/2}$. Only the first solution $\mathbf{W} = \mathbf{\Sigma}^{-1/2}$ can be considered as a symmetric and positive definite solution (see Appendix A, for a proof on convexity of the cost function). Doing some mathematical operations, $J(\mathbf{W})$

in (9) may be rewritten as (Appendix A)

$$J(\mathbf{W}) = \frac{1}{3}tr(\mathbf{W}^3\mathbf{\Sigma}) - tr(\mathbf{W}) + \frac{2}{3}tr(\mathbf{\Sigma}^{-1/2}). \tag{10}$$

Taking the first derivative of the cost function in (10) with respect to $\mathbf{W}$ gives

$$\frac{\partial J(\mathbf{W})}{\partial \mathbf{W}} = \frac{\mathbf{W}^2\mathbf{\Sigma} + \mathbf{W}\mathbf{\Sigma}\mathbf{W} + \mathbf{\Sigma}\mathbf{W}^2}{3} - \mathbf{I}. \tag{11}$$

Using the commutative property $\mathbf{W}\mathbf{\Sigma}^{1/2} = \mathbf{\Sigma}^{1/2}\mathbf{W}$, equation (11) can be reduced to one of the following three compact forms

$$\frac{\partial J(\mathbf{W})}{\partial \mathbf{W}} = \mathbf{W}^2\mathbf{\Sigma} - \mathbf{I} = \mathbf{W}\mathbf{\Sigma}\mathbf{W} - \mathbf{I} = \mathbf{\Sigma}\mathbf{W}^2 - \mathbf{I}. \tag{12}$$

By equating (12) to zero, we observe that $\mathbf{W} = \mathbf{\Sigma}^{-1/2}$ is the only solution. Since $J(\mathbf{W})$ is a convex function and its first derivative has only one stationary point ($\mathbf{W} = \mathbf{\Sigma}^{-1/2}$), then the cost function in (9) has only one strict absolute minimum that occurs at $\mathbf{\Sigma}^{-1/2}$ and there is no other local minimums. Using the gradient descent optimization method , the following equations are obtained for the adaptive estimation of the $\mathbf{\Sigma}^{-1/2}$,

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k(\mathbf{I} - \mathbf{W}_k^2\mathbf{\Sigma}), \tag{13}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k(\mathbf{I} - \mathbf{W}_k\mathbf{\Sigma}\mathbf{W}_k), \tag{14}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k(\mathbf{I} - \mathbf{\Sigma}\mathbf{W}_k^2) \tag{15}$$

where $\mathbf{W}_{k+1}$ is a $n \times n$ matrix that represent $\mathbf{\Sigma}^{-1/2}$ estimate at $k+1$-th iteration and η_k is the step size. As will be illustrated in the next Section, equations (13), (14), and (15) have similar convergence rate. It is quite straightforward to show that the commutative property ($\mathbf{W}_{k+1}\mathbf{\Sigma}^{1/2} = \mathbf{\Sigma}^{1/2}\mathbf{W}_{k+1}, k = 1, 2, \ldots$) is held in each iteration if and only if $\mathbf{W}_0\mathbf{\Sigma}^{1/2} = \mathbf{\Sigma}^{1/2}\mathbf{W}_0$. Therefore, the only constraint for the above equations is the initial condition where $\mathbf{W}_0$ must be a symmetric and positive definite matrix satisfying $\mathbf{W}_0\mathbf{\Sigma}^{1/2} = \mathbf{\Sigma}^{1/2}\mathbf{W}_0$. In all experiments in this paper, the initial condition matrix $\mathbf{W}_0$ is selected to be the identity matrix multiplied by a positive constant number α, i.e. $\mathbf{W}_0 = \alpha\mathbf{I}$. Since in most of on-line applications, the covariance matrix $\mathbf{\Sigma}$ is not known in advance, it can be replaced in (13-15) by an estimate of the covariance matrix as follows

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k(\mathbf{I} - \mathbf{W}_k^2\mathbf{\Sigma}_k), \tag{16}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k(\mathbf{I} - \mathbf{W}_k\mathbf{\Sigma}_k\mathbf{W}_k), \tag{17}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k(\mathbf{I} - \mathbf{\Sigma}_k\mathbf{W}_k^2). \tag{18}$$

The covariance matrix can be estimated by

$$\mathbf{\Sigma}_{k+1} = \mathbf{\Sigma}_k + \gamma_k(\mathbf{x}_{k+1}\mathbf{x}_{k+1}^t - \mathbf{\Sigma}_k), \tag{19}$$

where $\mathbf{\Sigma}_{k+1}$ is the covariance estimate at $k+1$-th iteration, $\mathbf{x}_{k+1}$ is $k+1$-th zero-mean input sample, and γ_k is the learning rate. The covariance matrix estimate at k-th iteration can be replaced by $\mathbf{x}_k\mathbf{x}_k^t$ [21], then the above equations simplify to the following forms

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k(\mathbf{I} - \mathbf{W}_k^2\mathbf{x}_k\mathbf{x}_k^t), \tag{20}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k(\mathbf{I} - \mathbf{W}_k\mathbf{x}_k\mathbf{x}_k^t\mathbf{W}_k), \tag{21}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k(\mathbf{I} - \mathbf{x}_k\mathbf{x}_k^t\mathbf{W}_k^2). \tag{22}$$

Equation (21) is similar to (7) and formerly was given in [26]. In order to prove the convergence of (16-18) and (20-22) to the minimum of the cost function $J(\mathbf{W})$, note that $J(\mathbf{W}) = E[\hat{J}(\mathbf{W}, \mathbf{x})] + 2/3tr[\mathbf{\Sigma}^{-1/2}]$, where $\hat{J}(\mathbf{W}, \mathbf{x}) = 1/3tr[\mathbf{W}^3\mathbf{x}\mathbf{x}^t] - tr[\mathbf{W}]$. Let $\mathbf{F}(\mathbf{W}, \mathbf{x})$ denote the gradient of $\hat{J}(\mathbf{W}, \mathbf{x})$ with respect to $\mathbf{W}$, i.e. $\mathbf{F}(\mathbf{W}, \mathbf{x}) = \nabla_W \hat{J}(\mathbf{W}, \mathbf{x})$. Then, the stochastic gradient algorithm defined by $\mathbf{W}_{k+1} = \mathbf{W}_k - \eta_k\mathbf{F}(\mathbf{W}_k, \mathbf{x}_k)$ converges almost surely to $\mathbf{W}^*$, such that $\mathbf{W}^*$ is a solution of the following minimization problem

$$\mathbf{W}^* = \min_{\mathbf{W}\in\mathbb{R}^{n\times n}} E[\hat{J}(\mathbf{W}, \mathbf{x})], \tag{23}$$

where $E[.]$ denote the expected value, $\mathbf{W} \in \mathbb{R}^{n\times n}$ is a positive definite matrix, and $\mathbf{x} \in \mathbb{R}^n$ is a random variable [39] [40]. The expected value of $\hat{J}(\mathbf{W}, \mathbf{x})$ is $1/3tr(\mathbf{W}^3\mathbf{\Sigma}) - tr(\mathbf{W})$, thus the minimum of

$E[\hat{J}(\mathbf{W},\mathbf{x})]$ is as same as (10) and occurs at $\mathbf{\Sigma}^{-1/2}$. The above argument shows that (16-18) and (20-22) converge almost surely to $\mathbf{\Sigma}^{-1/2}$. (In contrast to the ordinary gradient methods in (13-15) that use the expectation value $\mathbf{\Sigma}$, the stochastic gradient equation uses the random variable $\mathbf{F}(\mathbf{W},\mathbf{x})$ in order to minimize $E[\hat{J}(\mathbf{W},\mathbf{x})]$).

B. Estimate evaluation

A number of criteria may be used for terminating the iterations in (16-18) and (20-22). For example, the iterations may be terminated when the cost function $J(\mathbf{W})$ becomes less than a predefined threshold or when the norm of its gradient with respect to $\mathbf{W}$ becomes small enough. The availability of the cost function is an advantage for the proposed algorithms; since it makes possible evaluate the accuracy of estimates resulting from different initial conditions and learning rates. In other words, by substituting the final estimations into the cost function, the lower values of the cost function mean more accurate estimates. This criterion makes it possible compare quality of several solutions that have been obtained by running the algorithm using the different initial conditions and learning rates. The proposed algorithms in [26] [27] [28] [29] suffer from the non-existence of an explicit cost function. Hence, the authors have used other termination criteria such as the norm of the difference between two consecutive estimates. However, such criterion can not be used to evaluate the accuracy of the final estimate as well as the proposed criterion in this paper. Since, it is possible that the current estimate be significantly far from the actual value, but we obtain a small value for the norm of the difference between two consecutive estimates and terminate the iterations.

C. Fast Convergent $\mathbf{\Sigma}^{-1/2}$ Algorithms

Instead of using a fixed or decreasing learning rate, we use the cost function in (10) to obtain the optimal learning rate in each iteration in order to accelerate the convergence rate of $\mathbf{\Sigma}^{-1/2}$ algorithm. By taking the derivative of the cost function $J(\mathbf{W})$ with respect to $\mathbf{W}$ and equating it to zero, we can find the optimal learning rate in each iteration. For $k+1$-th iteration and using (13-15), we have

$$\frac{\partial J(\mathbf{W}_{k+1})}{\partial \eta_k} = \frac{\partial tr(\mathbf{W}_k + \eta_k \mathbf{G}_k)^3}{3\partial \eta_k} - \frac{\partial tr(\mathbf{W}_k + \eta_k \mathbf{G}_k)}{\partial \eta_k}, \tag{24}$$

where $\mathbf{G}_k$ has one of the following form, $\mathbf{G}_k = (\mathbf{I} - \mathbf{W}_k^2 \boldsymbol{\Sigma})$, $\mathbf{G}_k = (\mathbf{I} - \mathbf{W}_k \boldsymbol{\Sigma} \mathbf{W}_k)$, or $\mathbf{G}_k = (\mathbf{I} - \boldsymbol{\Sigma} \mathbf{W}_k^2)$. Equating (24) to zero and doing some mathematical operations (for details see Appendix B), we get the following quadratic form

$$a_k \eta_k^2 + b_k \eta_k + c_k = 0, \tag{25}$$

where $a_k = tr(\mathbf{G}_k^3 \boldsymbol{\Sigma})$, $b_k = 2tr(\mathbf{W}_k \mathbf{G}_k^2 \boldsymbol{\Sigma})$, and $c_k = tr(\mathbf{W}_k^2 \mathbf{G}_k \boldsymbol{\Sigma}) - tr(\mathbf{G}_k)$ and $\mathbf{G}_k$ is defined as before. Using the formula for roots of a quadratic equation, the optimal learning rate $\eta_{k,optimal}$ is given by

$$\eta_{k,optimal} = \frac{-b_k \pm \sqrt{b_k^2 - 4a_k c_k}}{2a_k}. \tag{26}$$

For situations where the covariance matrix $\boldsymbol{\Sigma}$ is not given in advance, we can replace it with $\boldsymbol{\Sigma}_k$ or $\mathbf{x}_k \mathbf{x}_k^t$. Thus, the accelerated adaptive $\boldsymbol{\Sigma}^{-1/2}$ algorithms have one of the following forms

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_{k,optimal}(\mathbf{I} - \mathbf{W}_k^2 \boldsymbol{\Sigma}_k), \tag{27}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_{k,optimal}(\mathbf{I} - \mathbf{W}_k \boldsymbol{\Sigma}_k \mathbf{W}_k), \tag{28}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_{k,optimal}(\mathbf{I} - \boldsymbol{\Sigma}_k \mathbf{W}_k^2), \tag{29}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_{k,optimal}(\mathbf{I} - \mathbf{W}_k^2 \mathbf{x}_k \mathbf{x}_k^t), \tag{30}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_{k,optimal}(\mathbf{I} - \mathbf{W}_k \mathbf{x}_k \mathbf{x}_k^t \mathbf{W}_k), \tag{31}$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_{k,optimal}(\mathbf{I} - \mathbf{x}_k \mathbf{x}_k^t \mathbf{W}_k^2), \tag{32}$$

where $\eta_{k,optimal}$ and $\boldsymbol{\Sigma}_k$ are updated using (26) and (19), respectively.

D. Adaptive LDA Algorithm and Convergence Proof

As discussed in section II, the LDA features are the significant eigenvectors of $\boldsymbol{\Sigma}_W^{-1} \boldsymbol{\Sigma}_B$. Since both $\boldsymbol{\Sigma}_W^{-1} \boldsymbol{\Sigma}_B$ and $\boldsymbol{\Sigma}_W^{-1} \boldsymbol{\Sigma}$ have the same eigenvectors, we present an adaptive algorithm for the computation of the eigenvectors of $\boldsymbol{\Sigma}_W^{-1} \boldsymbol{\Sigma}$. For this purpose, two adaptive algorithms introduced in the previous subsections are combined in a cascade form and we show that the resulting architecture computes the desired LDA features.

Let $\mathbf{x}_k \in \mathbb{R}^n, k = 1, 2, \dots$ denote the sequence of input samples such that each sample belongs to exactly one of K available classes $\omega_i, i = 1, 2, \dots, K$. Let $\mathbf{m}_k^{\omega_i}, i = 1, 2, \dots, K$ denote the estimated mean at k-th iteration for class ω_i. We define the training sequence $\{\mathbf{y}_k\}_{k=1,2,\dots}$ for the $\mathbf{\Sigma}^{-1/2}$ to be $\mathbf{y}_k = \mathbf{x}_k - \mathbf{m}_k^{\omega_{\mathbf{x}_k}}$, where $\mathbf{x}_k \in \mathbb{R}^n, k = 1, 2, \dots$ represent the input samples and $\mathbf{m}_k^{\omega_{\mathbf{x}_k}}$ denote the sample mean of the class that $\mathbf{x}_k$ belongs to it at k-th iteration. If the new incoming training sample $\mathbf{x}_{k+1}$ belongs to the class $\omega_{\mathbf{x}_{k+1}}$, all the class means except the mean for the class $\omega_{\mathbf{x}_k}$ remain unchanged and $\mathbf{m}_k^{\omega_{\mathbf{x}_k}}$ will be updated by [25]

$$\mathbf{m}_{k+1}^{\omega_{\mathbf{x}_{k+1}}} = \mathbf{m}_k^{\omega_{\mathbf{x}_{k+1}}} + \mu_k(\mathbf{x}_{k+1} - \mathbf{m}_k^{\omega_{\mathbf{x}_{k+1}}}), \tag{33}$$

where $\mu_k, k = 1, 2, \dots$ is the learning rate at k-th iteration that meets the Ljung's conditions [37]. It has been shown that under certain conditions $\mathbf{x}_k^{\omega_{\mathbf{x}_k}}$ converges to the class mean $\mathbf{m}^{\omega_{\mathbf{x}_k}}$ [25]. Alternatively, one may use the following equation for updating the class mean $\mathbf{m}_k^{\omega_{\mathbf{x}_{k+1}}}$,

$$\mathbf{m}_{k+1}^{\omega_{\mathbf{x}_{k+1}}} = \frac{k\mathbf{m}_k^{\omega_{\mathbf{x}_{k+1}}} + \mathbf{x}_{k+1}}{k+1}. \tag{34}$$

It is straightforward to show that the limit of the correlation of the training sequence $\{\mathbf{y}_k\}$ converges to the within-class scatter matrix $\mathbf{\Sigma}_W$ (See appendix C for details) and we have

$$\lim_{k\to\infty} E[(\mathbf{x}_k - \mathbf{m}_k^{\omega_{\mathbf{x}_k}})(\mathbf{x}_k - \mathbf{m}_k^{\omega_{\mathbf{x}_k}})^t] = \lim_{k\to\infty} E[\mathbf{y}_k\mathbf{y}_k^t] = \mathbf{\Sigma}_W. \tag{35}$$

Therefore, if we train the proposed algorithms in (13-15), (16-18), and (20-22) using the sequence $\{\mathbf{y}_k\}$, then $\mathbf{W}_k$ will converge to the square root of the inverse of the within-class scatter matrix $\mathbf{\Sigma}_W$. in other words, we have

$$\lim_{k\to\infty} \mathbf{W}_k = \mathbf{\Sigma}_W^{-1/2}. \tag{36}$$

We define the new sequence $\{\mathbf{z}_k\}_{k=1,2,\dots}$ to be $\mathbf{z}_k = \mathbf{x}_k - \mathbf{m}_k, k = 1, 2, \dots$, where $\mathbf{m}_k$ is the mixture mean vector at k-th iteration. We train the proposed $\mathbf{\Sigma}^{-1/2}$ algorithms using the sequence $\{\mathbf{y}_k\}$, then resulting

$\mathbf{W}_k$ and the sequence $\{\mathbf{z}_k\}$ are multiplied to generate a new sequence $\{\mathbf{u}_k\}_{k=1,2,\ldots}$, i.e. $\mathbf{u}_k = \mathbf{W}_k\mathbf{z}_k, k = 1, 2, \ldots$.

It has been shown that if we start from a $p \times n$ random matrix, the algorithm in (2) will converge to a matrix whose rows are the first p eigenvectors of the covariance matrix corresponding to the largest eigenvalues [20]. The sequence $\{\mathbf{u}_k\}$ is used to train the APCA algorithm in (2) and as a result the APCA algorithm converges to the eigenvectors of the covariance matrix of its input sequence. Using some mathematical operations (See appendix C for details), it is straightforward to show $\lim_{k\to\infty} E(\mathbf{u}_k\mathbf{u}_k^t) = \mathbf{\Sigma}_W^{-1/2}\mathbf{\Sigma}\mathbf{\Sigma}_W^{-1/2}$. Let $\mathbf{\Psi}$ and $\mathbf{\Phi}_{LDA}$ denote the eigenvector matrix for $\mathbf{\Sigma}_W^{-1/2}\mathbf{\Sigma}\mathbf{\Sigma}_W^{-1/2}$ and the eigenvector matrix for $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}$, respectively. Our goal is to estimate the columns of $\mathbf{\Phi}_{LDA}$ as the LDA features in an adaptive manner. Note that the eigenvector matrix $\mathbf{\Psi}$ is given by $\mathbf{\Psi} = \mathbf{\Sigma}_W^{1/2}\mathbf{\Phi}_{LDA}$. Therefore, if we initialize (2) with a random $p\times n$ matrix and train the algorithm using the sequence $\{\mathbf{u}_k\}$, then it converges to a $p \times n$ matrix $(\mathbf{\Psi}^p)^t$, whose rows are the first p columns of $\mathbf{\Psi}$. In other words, we have

$$\lim_{k\to\infty} \mathbf{T}_k = (\mathbf{\Psi}^p)^t = (\mathbf{\Phi}_{LDA}^p)^t\mathbf{\Sigma}_W^{1/2}, \tag{37}$$

where the superscript t denote the transpose operator, and $\mathbf{\Phi}_{LDA}^p$ is a $n \times p$ matrix whose columns are the first p columns of $\mathbf{\Phi}_{LDA}$. The matrix containing the significant LDA features is obtained by multiplying the equations given in (36) and (37) as follows,

$$\lim_{k\to\infty} \mathbf{W}_k\mathbf{T}_k^t = \mathbf{\Sigma}^{-1/2}\mathbf{\Sigma}^{1/2}\mathbf{\Phi}_{LDA}^p = \mathbf{\Phi}_{LDA}^p. \tag{38}$$

Therefore, the combination of a $\mathbf{\Sigma}^{-1/2}$ algorithm and an APCA algorithms will converge to the desired matrix $\mathbf{\Phi}_{LDA}^p$, whose columns are the first p eigenvectors of $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}$ corresponding to the largest eigenvalues. The proposed ALDA algorithms can be summarized as follows

1) Construct the sequences $\{\mathbf{y}_k\}$ and $\{\mathbf{u}_k\}$ using the input sequence $\{\mathbf{x}_k\}$ as mentioned before.
2) Train the proposed $\mathbf{\Sigma}^{-1/2}$ algorithms, equations (16-18) or (20-22), using the sequence $\{\mathbf{y}_k\}$.
   - If you use equations (16-18), estimate the covariance matrix from equation (19).

- For the fast convergent $\mathbf{\Sigma}^{-1/2}$ algorithms, compute the optimal learning rate $\eta_{k,optimal}$ using (26) in each iteration.

3) Define the new sequence $\{\mathbf{u}_k\}$ sequentially by $\mathbf{u}_k = \mathbf{W}_k\mathbf{z}_k$.
4) Train the APCA algorithm in (2) using the sequence $\{\mathbf{u}_k\}$.
5) Multiply the output of the APCA algorithm in (2) and the output of the $\mathbf{\Sigma}^{-1/2}$ algorithm. This product gives the an estimate of the desired number of the LDA features.

The proposed ALDA algorithm is illustrated in Fig. 1. The input sequence $\{\mathbf{x}_k\}$ is given to the algorithm sequentially. Each new incoming input sample improves the LDA feature estimates and our estimates asymptotically converge to the true LDA features. Adding a block to compute the optimal learning $\eta_{k,optimal}$ at k-th iteration fasten the convergence rate but it also increase the computational cost.

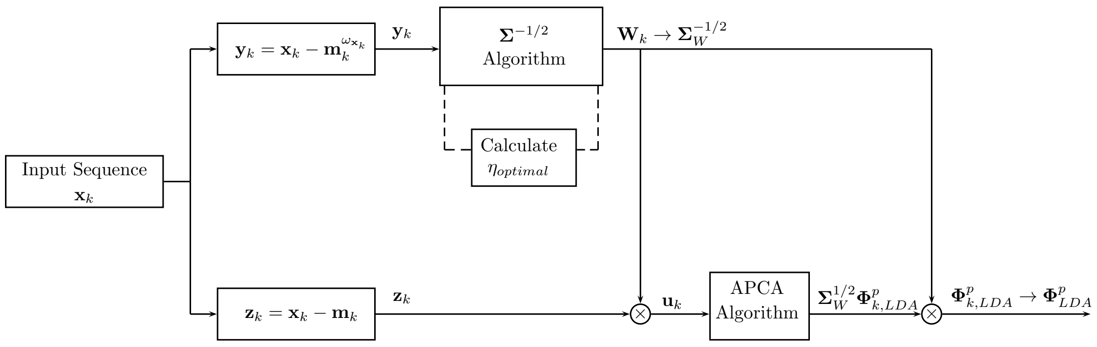
Input Sequence $\mathbf{x}_k$
$\mathbf{y}_k = \mathbf{x}_k - \mathbf{m}_k^{\omega_{\mathbf{x}_k}}$
$\mathbf{y}_k$
$\mathbf{\Sigma}^{-1/2}$ Algorithm
Calculate $\eta_{optimal}$
$\mathbf{W}_k \rightarrow \mathbf{\Sigma}_W^{-1/2}$
$\mathbf{z}_k = \mathbf{x}_k - \mathbf{m}_k$
$\mathbf{z}_k$
$\mathbf{u}_k$
APCA Algorithm
$\mathbf{\Sigma}_W^{1/2}\mathbf{\Phi}_{k,LDA}^p$
$\mathbf{\Phi}_{k,LDA}^p \rightarrow \mathbf{\Phi}_{LDA}^p$

Fig. 1: Proposed ALDA algorithm to estimate the LDA features in an adaptive manner. The input sequence $\{\mathbf{x}_k\}$ is observed sequentially and used to generate the sequences $\{\mathbf{y}_k\}$ and $\{\mathbf{z}_k\}$. The proposed $\mathbf{\Sigma}^{-1/2}$ algorithm is trained using the sequence $\{\mathbf{y}_k\}$ and its output is multiplied by the sequence $\{\mathbf{z}_k\}$ to generate the new sequence $\{\mathbf{u}_k\}$. The sequence $\{\mathbf{u}_k\}$ is used to train the APCA algorithm. As the number of iterations increase, we get a better estimation of the LDA features and our estimate asymptotically converge to the true LDA features.

IV. Simulation Results

In all experiments in this section, a sequence of input data is used to train each algorithm in an adaptive manner. It is assumed that the whole data set is not available in advance and the input data is fed to the proposed algorithms sequentially. For these simulations, MATLAB codes are run on a PC with Intel Pentium 4, 2.6-GHZ CPU and 2048-Mb RAM.

A. Experiments on $\mathbf{\Sigma}^{-1/2} algorithms$

In this experiment, we first compare the convergence rate of equations (16-18) and (20-22) to estimate the square root of the inverse covariance matrix $\mathbf{\Sigma}^{-1/2}$ in a ten dimensional space. The incoming input sequence $\{\mathbf{x}_k\}$ is in $\mathbb{R}^{10}$ and the covariance matrix $\mathbf{\Sigma}$ for generating the data samples is the first covariance matrix introduced in [41] multiplied by 20. We generated 500 samples of ten dimensional, zero mean Gaussian data with the covariance matrix $\mathbf{\Sigma}$. The actual value of $\mathbf{\Sigma}^{-1/2}$ was obtained from the sample correlation matrix using a standard eigenvector computation method. The input samples are fed to the proposed $\mathbf{\Sigma}^{-1/2}$ algorithms and the algorithms are initialized to be the identity matrix, i.e. $\mathbf{W}_0 = \mathbf{I}$. The $L2$ norm of the error at k-th iteration between the estimated and the actual $\mathbf{\Sigma}^{-1/2}$ matrices is computed by

$$e_k = \sqrt{\sum_{i=1}^{10}\sum_{j=1}^{10}(\mathbf{W}_k(i,j) - \mathbf{\Sigma}^{-1/2}(i,j))^2}, \tag{39}$$

where $\mathbf{W}_k(i,j)$ and $\mathbf{\Sigma}^{-1/2}(i,j)$ represent the ij-th element of the estimated square root of the inverse covariance matrix at k-iteration and the ij-th element of the actual square root of the inverse covariance matrix, respectively. Fig. 2 and Fig. 3 illustrate the convergence of the proposed algorithms in (16-18) and (20-22). From Fig. 2 it can be observed that the proposed $\mathbf{\Sigma}^{-1/2}$ algorithms in (16-18) have very similar convergence rate and the convergence graphs overlap each other. The proposed algorithms in (16-18) use an estimate of the covariance matrix and in each iteration they update this estimate using (19). As a result, graphs in Fig. 2 show a smooth convergence to $\mathbf{\Sigma}^{-1/2}$. In contrast, the graphs in Fig. 3 show an erratic behavior, since the proposed algorithms in (20-22) instead of using an estimate of the covariance matrix use the k-th input sample to estimate the covariance matrix. Fig. 4 compares convergence rate of algorithms in (17) and (21). Although, algorithm in (17) converges smoothly to $\mathbf{\Sigma}^{-1/2}$, but the convergence rate for both algorithm is nearly the same and after 500 iterations the estimation error is negligible. It can be observed from Fig. 2 and Fig. 3 that the proposed $\Sigma^{-1/2}$ algorithms can successfully be applied for estimation of the square root of the inverse of the covariance matrix. Since the proposed $\Sigma^{-1/2}$ algorithms have similar convergence rates, they can interchangeably be used to estimate $\Sigma^{-1/2}$ in an adaptive manner. Table I compares the normalized error resulting from the proposed $\Sigma^{-1/2}$ algorithms in (16-18) and (20-22) as the number of iterations changes from 100 to 500. The measurements in Table I indicate that the estimates achieved using the proposed $\Sigma^{-1/2}$ algorithms are close to the actual value and the estimation errors after 500 iterations are negligible.

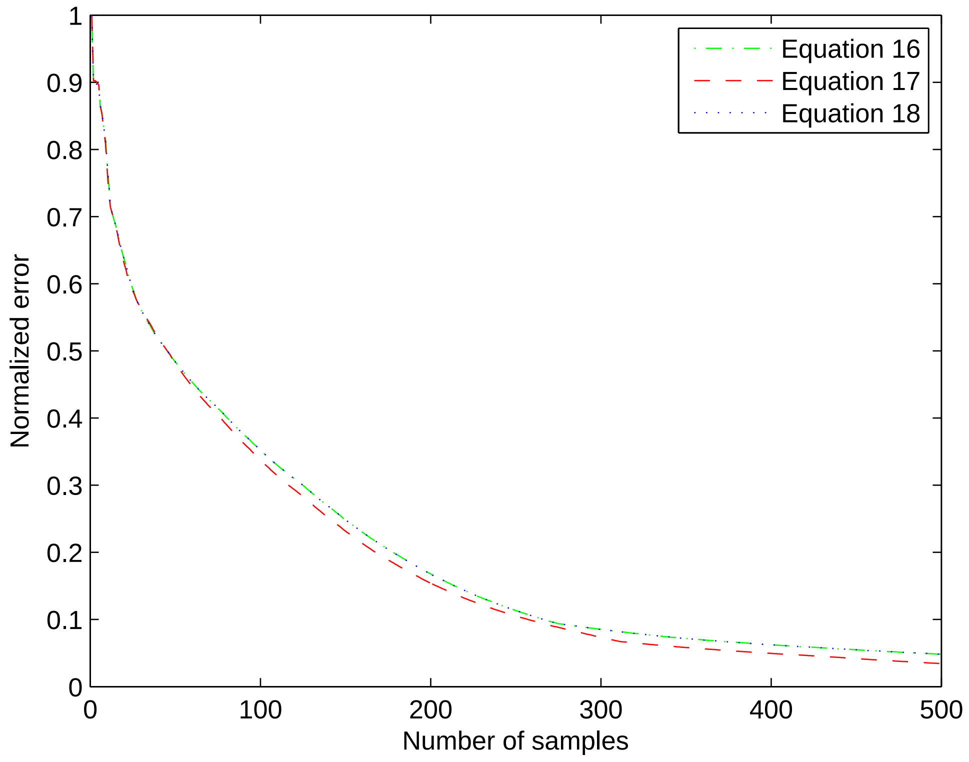

Fig. 2: *Convergence rate of the proposed $\Sigma^{-1/2}$ algorithms in (16-18).* They use the incoming input sample to update the estimate of the covariance matrix using (19). The updated covariance matrix is used to improve the estimate of the square root of the inverse covariance matrix. These algorithms have very similar convergence rate and overlap each other.

We also compared convergence rate of the proposed $\Sigma^{-1/2}$ algorithm in 4, 6, 8, and 10 dimensional spaces. For the ten dimensional space we used the same covariance matrix Σ as before (that is the first covariance matrix in [41] multiplied by 20). The ten eigenvalues of the covariance matrix Σ in descending

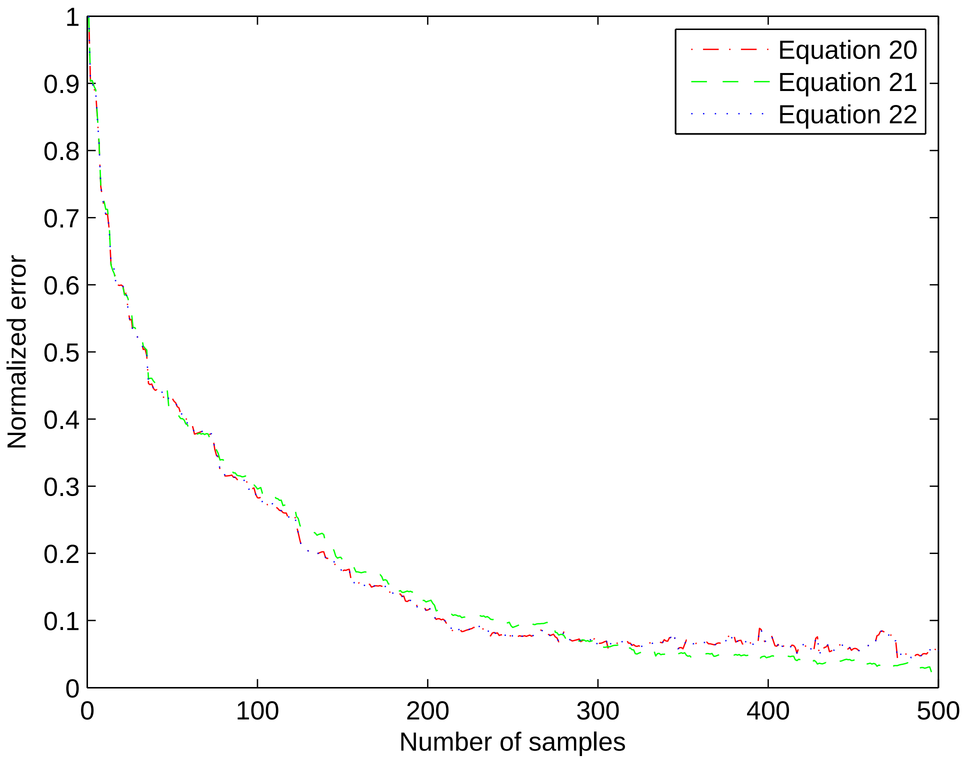

Fig. 3: *Convergence rate of the proposed $\mathbf{\Sigma}^{-1/2}$ algorithms in (20-22).* For these algorithms, only the incoming input sample is used to find an estimate of the covariance matrix in each iteration.

order are 117.996, 55.644, 34.175, 7.873, 5.878, 1.743, 1.423, 1.213, and 1.007. The covariance matrices for 4, 6, and 8 dimensional spaces are selected as the principal minors of $\mathbf{\Sigma}$.The initial estimate $\mathbf{W}_0$ is chosen to be the identity matrix and the proposed algorithm are trained using a sequence of Gaussian data in each space separately. For each experiment, at k-th update, we compute the normalized error between the estimated and the actual matrices. The convergence rate of the proposed $\mathbf{\Sigma}^{-1/2}$ algorithm in (22) for each covariance matrix is illustrated in Fig. 5. It can be observed from Fig. 5 that after 500 iterations, the normalized errors are less than 0.1 for all covariance matrices. The same experiment is repeated using the other proposed $\mathbf{\Sigma}^{-1/2}$ algorithms and we got similar graphs.

In order to show the tracking ability of the introduced $\mathbf{\Sigma}^{-1/2}$ algorithms for non-stationary data, we generated 500 zero mean Gaussian samples in $\mathbb{R}^{10}$with the covariance matrix as stated before. Then, We

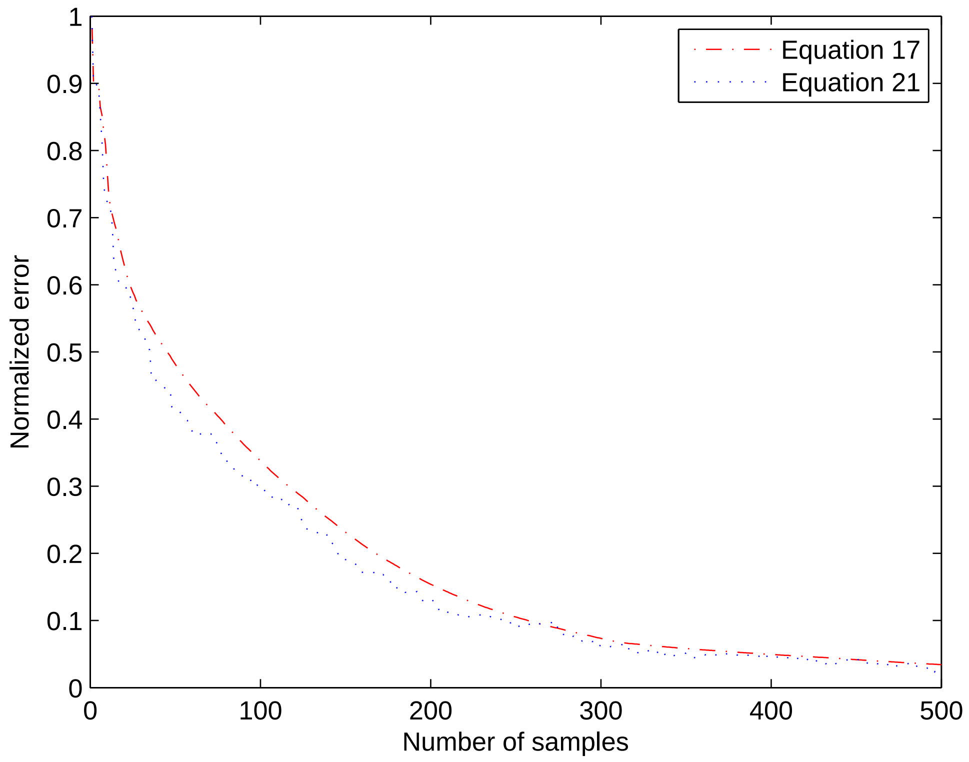

Fig. 4: *Convergence rate of algorithms (17) and (21).* Although algorithm (17) converges smoothly to $\mathbf{\Sigma}^{-1/2}$, but both algorithms have similar convergence rate.

drastically changed the nature of the data sequence by generating new 500 zero mean 10-dimensional Gaussian data with the second covariance matrix introduced in [41]. The tracking ability of the proposed $\mathbf{\Sigma}^{-1/2}$ algorithm in (20) is illustrated in Fig. 6. We initiated the algorithm with $\mathbf{W}_0 = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. It is clear from Fig. 6 that the proposed $\mathbf{\Sigma}^{-1/2}$ algorithm tracks the changes in the data. The normalized error increases when the covariance matrix alters after 500 iterations but the algorithm gradually adapts itself to the new covariance matrix and the normalized error starts to decrease by observing new samples from the new covariance matrix.

We show the effectiveness of the proposed accelerated $\mathbf{\Sigma}^{-1/2}$ algorithm to estimate $\mathbf{\Sigma}^{-1/2}$ by comparing its convergence rate with the algorithm introduced in [26]. We generated 500 zero mean 10-dimensional Gaussian data with the first covariance matrix given in [41]. The algorithm in [26] and the proposed

TABLE I: The normalized error for estimation of the square root of the inverse covariance matrix $\mathbf{\Sigma}^{-1/2}$ resulting from the proposed equations (16-18) and (20-22), as the number of iterations increases from 100 to 500.

Number of Iteration	100	200	300	400	500
Equation (16)	0.2915	0.1545	0.1056	0.0823	0.0658
Equation (17)	0.2923	0.1558	0.1023	0.0789	0.0619
Equation (18)	0.2915	0.1545	0.1055	0.0823	0.0658
Equation (20)	0.2933	0.1461	0.0951	0.0716	0.0545
Equation (21)	0.2889	0.1467	0.0892	0.0667	0.0447
Equation (22)	0.2933	0.1462	0.0950	0.0716	0.0545

accelerated $\mathbf{\Sigma}^{-1/2}$ algorithm are trained using the input sequence. The optimal learning rate at k-th iteration $\eta_{k,Optimal}$ for the proposed algorithm is computed using (26). For the algorithm introduced in [26] a decreasing learning rate is used. The normalized error in each iteration is computed and Fig. 7 illustrates the convergence rates for these two algorithms. It can be observed from Fig. 7, the new accelerated $\mathbf{\Sigma}^{-1/2}$ algorithm reaches to an estimation error of 0.1 after observing about 100 samples. In contrast, the proposed algorithm in [26] gives an estimation error of 0.1 after training by 500 samples. This means that by applying the new accelerated $\mathbf{\Sigma}^{-1/2}$ algorithm, we can achieve to the desired estimation error much faster than using the algorithm proposed in [26].

B. Experiments on Adaptive LDA Algorithm

The performance of the proposed ALDA algorithms are tested using i) ten dimensional, five class Gaussian data, ii) four dimensional, three class Iris data set, and iii) five class, 40×40 gray scale face images.

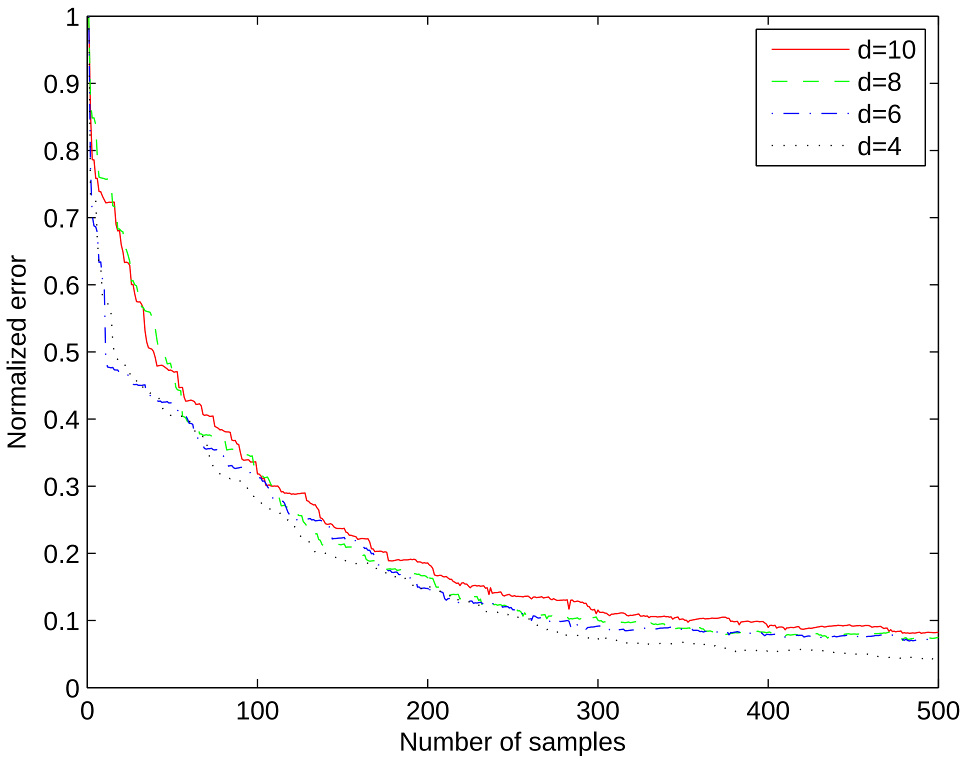

Fig. 5: *Convergence rate of the proposed algorithms.* We used algorithm in (22) to estimate $\mathbf{\Sigma}^{-1/2}$ in 4, 6, 8, and 10 dimensional spaces. For simulation in 10 dimensional space, we used the first covariance matrix in [41] and multiplied it by 20. The covariance matrices for the lower dimensions are chosen as the principal minors of the covariance matrix in 10 dimensional space.

1) Experiment on ten dimensional data: We test the performance of the proposed ALDA algorithm to extract the significant LDA features [1] adaptively. We generated 500 samples of 10-dimensional Gaussian data for each of five classes with different mean value and covariance matrices, i.e. 2500 samples in total. The means and the covariance matrices are obtained from [41] with the covariance matrices multiplied by 20. The eigenvalues of $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B$ in descending order are 10.84, 7.01, 0.98, 0.34, 0, 0, 0, 0, 0, and 0. Thus, the data has intrinsic dimensionality of four and only two features corresponding to the eigenvalues 10.84 and 7.01 are significant. We use the proposed ALDA algorithms to estimate the two significant LDA features

[1]By significant LDA features, we mean directions that are the most important for the class separability purpose. These directions are the eigenvectors of $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B$ corresponding to the largest eigenvalues.

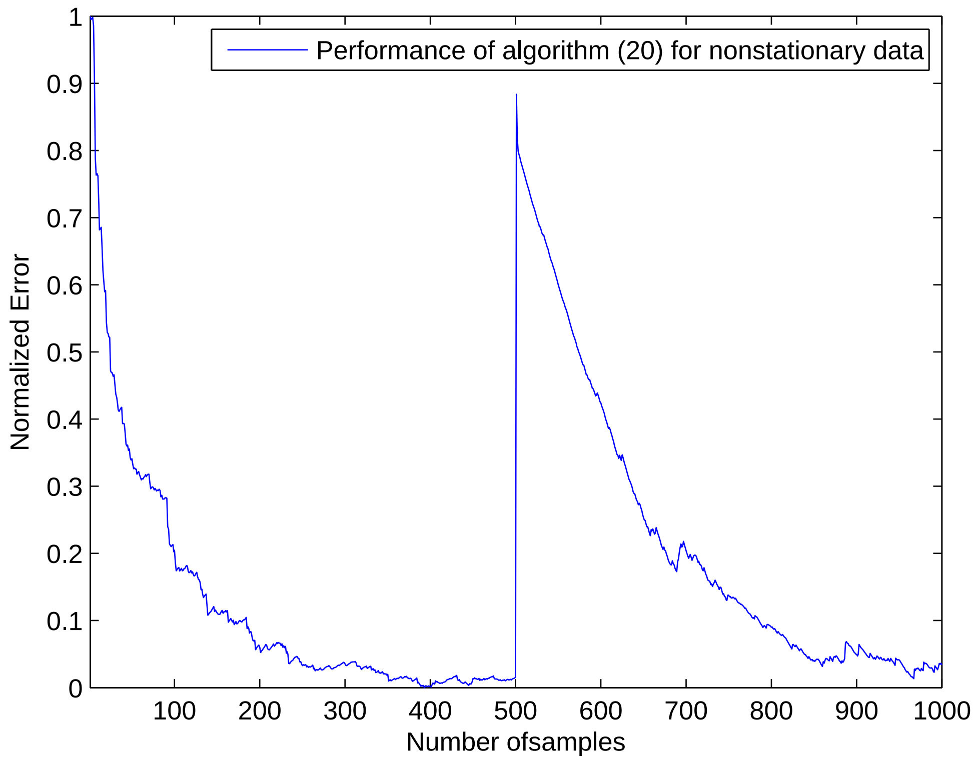

Fig. 6: *Tracking ability of the proposed* $\mathbf{\Sigma}^{-1/2}$ *algorithm.* The first 500 samples are zero mean 10 dimensional Gaussian data with the first covariance matrix in [41] multiplied by 20. The second 500 samples are zero mean Gaussian data in $\mathbb{R}^{10}$ with the second covariance matrix in [41] multiplied by 20. The training data is fed to the proposed $\mathbf{\Sigma}^{-1/2}$ algorithm in (20) and the normalized errors between the estimated and real $\mathbf{\Sigma}^{-1/2}$ matrices are computed at each iteration.

adaptively and use them for the classification purpose. We reduce the dimensionality of the sample space to two by projecting the data samples into the estimated LDA features. The normalized error between the estimated LDA features and their actual values, found using the scatter matrices, is computed in each iteration. Let ϕ_i and $\hat{\phi}_i, i = 1, 2$ denote the actual significant LDA features and the estimated versions, respectively. We define the normalized error in each iteration by

$$E_{\phi_i} = \frac{\|\phi_i - \hat{\phi}_i\|}{\|\phi_i\|}, i = 1, 2. \tag{40}$$

As described in section II, the proposed ALDA techniques consist of two algorithms; a $\mathbf{\Sigma}^{-1/2}$ algorithms and an APCA algorithm (2). For $\mathbf{\Sigma}^{-1/2}$ algorithm, we can use any of algorithms given in (16-18), (20-

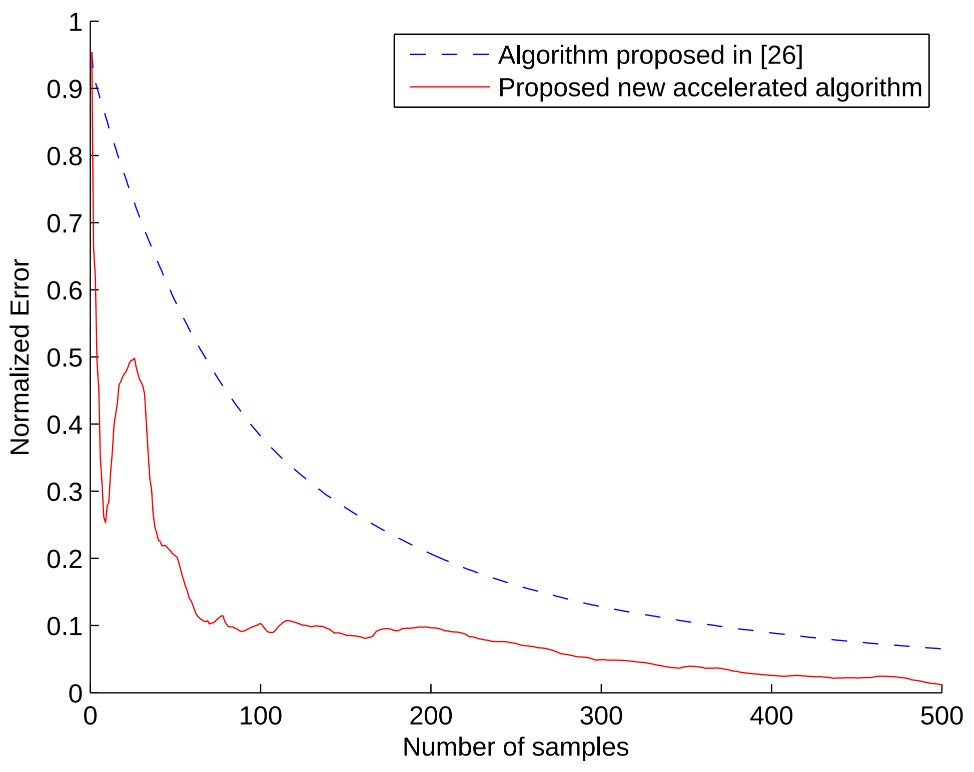

Fig. 7: *Convergence rate of the proposed accelerated $\mathbf{\Sigma}$ algorithm and the algorithm given in [26].* The proposed accelerated algorithm gives a good estimate of $\mathbf{\Sigma}^{-1/2}$ in fewer iterations compared with the algorithm proposed in [26].

22), or the accelerated versions (27-32). Fig. 8 illustrates the convergence of the estimated LDA features, $\hat{\phi}_1$ and $\hat{\phi}_2$ for the proposed ALDA algorithm in (16)[1]. It is observed from Fig. 8 that the LDA feature vectors estimated by the proposed ALDA algorithm converge to their actual values through the training phase. Optimization of the learning rate in each iteration using (26) increases the convergence rate. Fig. 9 illustrates convergence of the two significant LDA features when the proposed ALDA algorithm finds the optimal learning rate in each iteration. Although finding the optimal learning rate in each iteration increases the computational cost, it is clear from Fig. 9 that we achieve a good estimate of the LDA features in fewer iterations compared to the previous case (illustrated in Fig. 8). To show the effectiveness

[1]The proposed $\mathbf{\Sigma}^{-1/2}$ algorithms in (16-18) and (20-22) have the similar convergence rate and they can be used interchangeably in order to estimate the square root of the inverse covariance matrix $\mathbf{\Sigma}^{-1/2}$.

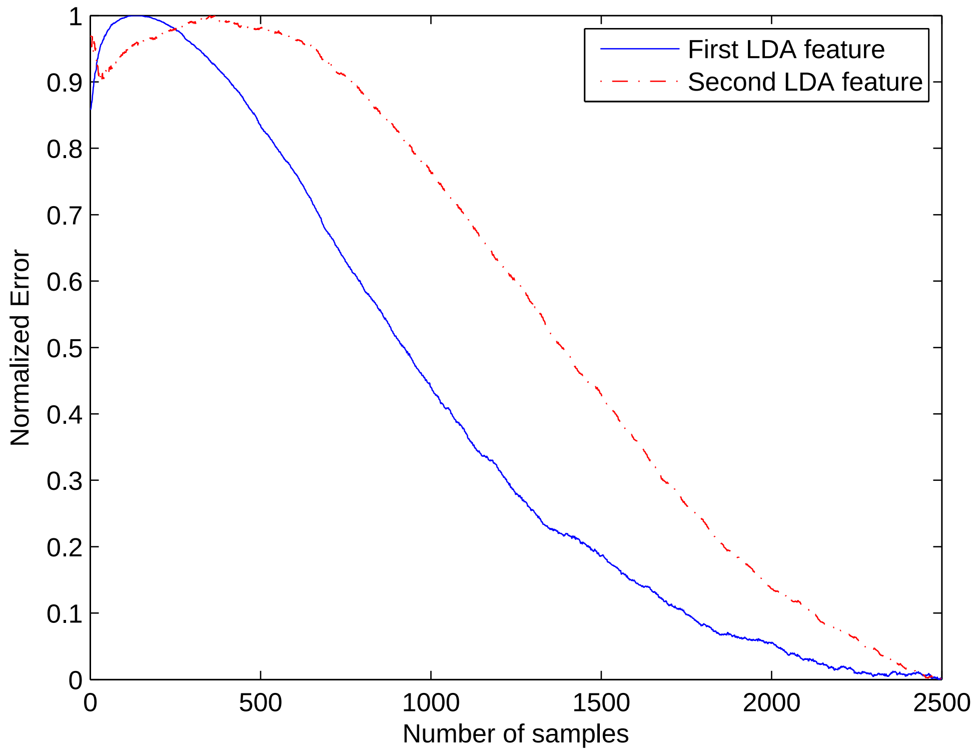

Fig. 8: *Convergence of estimated significant LDA features.* In this simulation, the proposed ALDA algorithm uses $\mathbf{\Sigma}^{-1/2}$ algorithm in (16) in order to estimate $\mathbf{\Sigma}_W^{-1/2}$.

of the proposed fast ALDA algorithm [1], we compare its performance to estimate the first LDA feature with the techniques introduced in [26] and [30]. The normalized error as a function of the number of samples is shown in Fig. 10 for these three algorithms. It is clear from Fig. 10, the proposed fast ALDA algorithm gives a better estimate of the LDA feature in fewer iteration compared to the proposed algorithms in [26] and [30]. Table II compares the normalized error for the estimation of the first LDA feature resulting from applying the proposed algorithm and the algorithms introduced in [26] and [30], respectively, as a function of the number of the observed training samples. It is clear from Table II, the proposed fast ALDA algorithm outperforms the algorithms in [26] and [30] in term of the convergence rate.

In Fig. 11, We show the convergence of the second LDA feature using the proposed fast ALDA algorithm

[1]By fast ALDA algorithm we mean the ALDA algorithm that compute the optimal learning rate using (26) in each iteration in order to accelerate the convergence rate.

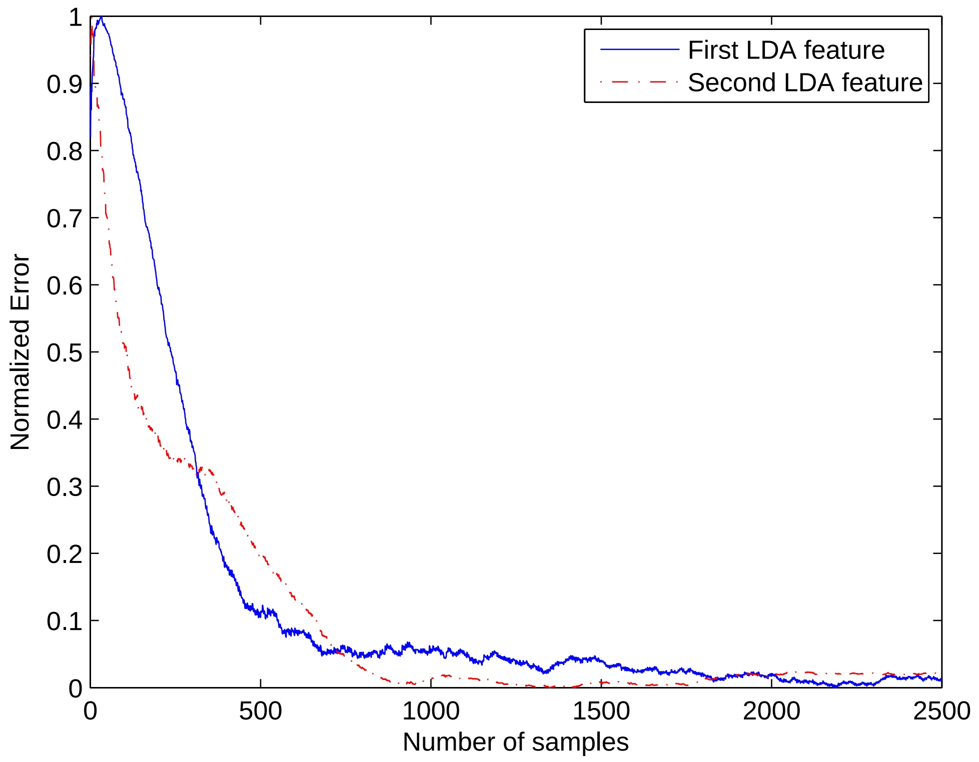

Fig. 9: *Convergence of estimated significant LDA features.* The proposed fast ALDA algorithm is used to estimate the two significant LDA features. The learning rate in each iteration is optimized in order to accelerate the convergence rate.

TABLE II: The normalized error for estimation of the first LDA feature resulting from applying the proposed ALDA algorithm and the algorithms in [26] and [30] as a function of the number of the observed training samples.

Number of Samples	100	300	500	1000	1500	1800	2000	2200	2500
Proposed fast ALDA	0.7693	0.3427	0.1799	0.0504	0.0114	0.0257	0.0173	0.0149	0.0089
proposed algorithm in [26]	0.9994	0.9488	0.8158	0.4324	0.2275	0.1428	0.0844	0.0530	0.0087
proposed algorithm in [36]	0.9637	0.8276	0.4254	0.1004	0.0161	0.0292	0.0183	0.0697	0.0637

and the algorithm proposed in [26] by computing the normalized error at each update. It can be observed from Fig. 11 that the second LDA feature obtained from the proposed fast ALDA algorithm converges to

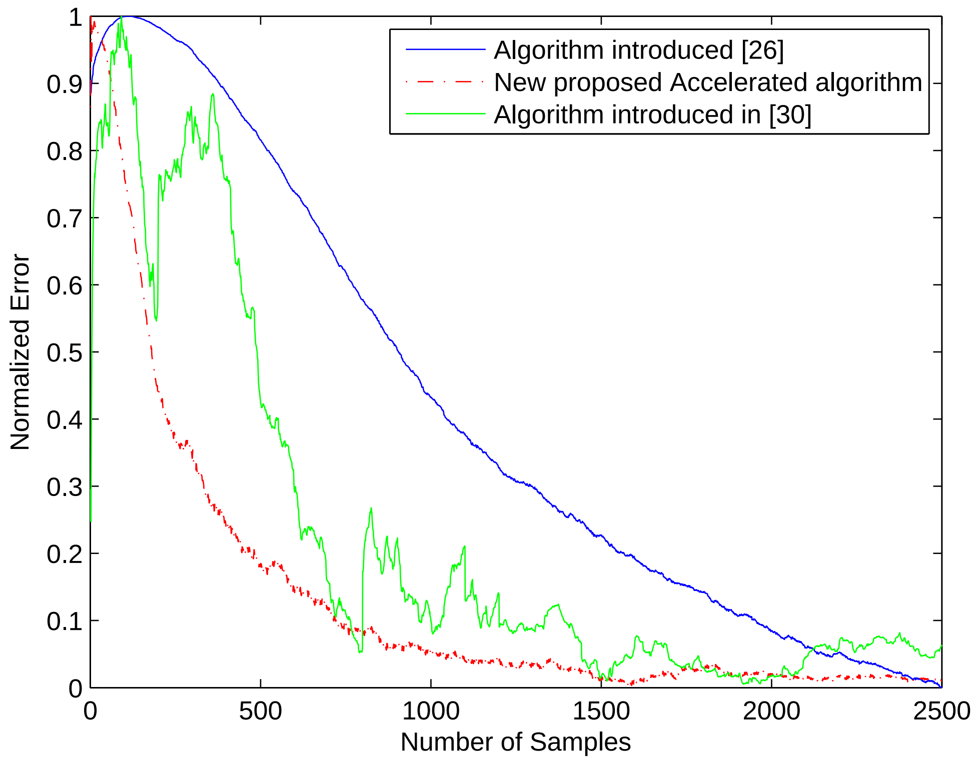

Fig. 10: *Convergence of the estimated first LDA feature.* Convergence rate of the proposed fast ALDA algorithm to estimate the first significant LDA feature is compared with that of the algorithms introduced in [26] and [30].

its true value faster than the algorithm introduced in [26][1]. We project the training data on the subspace spanned by the estimated LDA features $\hat{\phi}_1$ and $\hat{\phi}_2$ in order to show the effectiveness of the proposed ALDA algorithm to find directions for the maximum class separability. Fig. 12 demonstrates the distribution of the input data on the subspace spanned by the estimated two significant LDA feature vectors resulting after 500, 1000, 1500, and 2500 iterations. From Fig. 12, it can be observed that when the number of iterations increases, the distributions of the five classes on the estimated LDA feature subspace become more separable. As the number of the iterations increases, we get a more precise estimate of the significant LDA features and the projected data points start to be classified into five separable clusters. We can observe

[1]Since the proposed algorithm in [30] does not have the capability of estimating the desired number of LDA features simultaneously, we compared the convergence rate of the proposed algorithm to estimate the second significant LDA feature with the algorithm introduced in [26]. The algorithm in [30] estimates the LDA features sequentially, it first estimates the most significant LDA feature and then uses it to estimate the second significant LDA feature and this procedure continues to estimate the rest of LDA features.

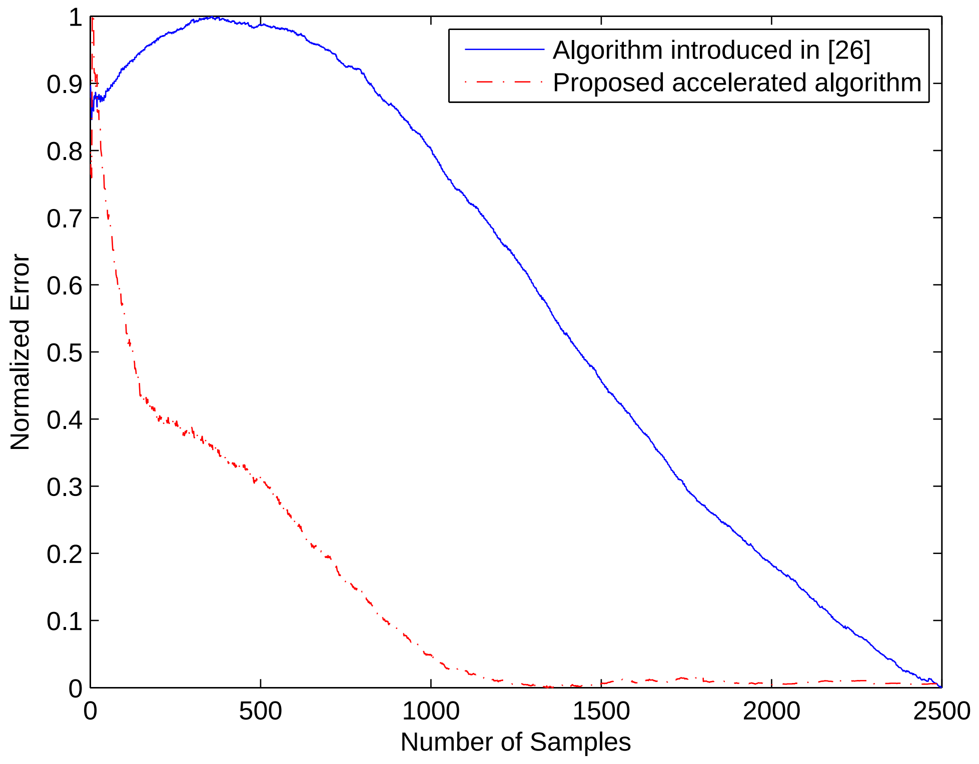

Fig. 11: *Convergence of the estimated second LDA feature.* Convergence rate of the proposed fast ALDA algorithm to estimate the second LDA feature is compared with that of the algorithm introduced in [26].

from Fig. 12, after 500 iterations we have a poor estimate of the significant LDA features and fives classes in the estimated subspace mix together. However, when gradually the number of iterations increase, the classes start to separate from each other and finally after 2500 iterations they are linearly separable in the estimated LDA feature subspace[1]. Although four LDA features are necessary for the complete class separation[2], Fig. 12 shows that the projection onto the subspace spanned by two significant LDA features separate the data into five classes.

2) Experiment with Iris data set: The Iris flower data set [3] is a well-known, multivariate data set that has been widely used as a non-synthetic data to evaluate classification techniques [42] [43]. This data set consists of 50 samples from each of three species of Iris flowers including Iris setosa, Iris virginica, and Iris versicolor.

[1]There are some overlappings and they are not perfectly linearly separable.

[2]As mentioned earlier, the ten dimensional matrix $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B$ used in this experiment has only four nonzero eigenvalues.

[3]The data set is available at http://archive.ics.uci.edu/ml/machine-learning-databases/iris/

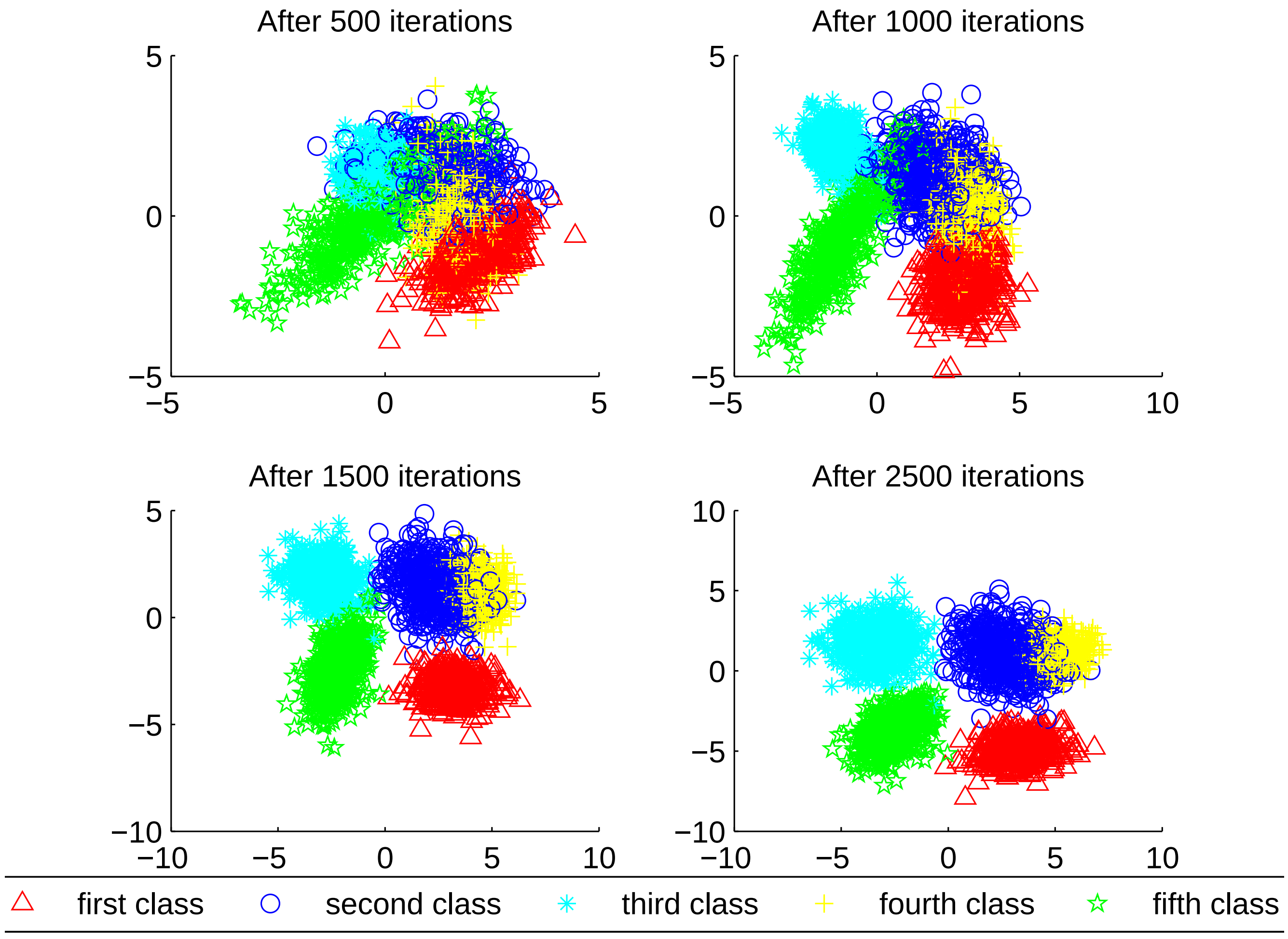

Fig. 12: *Distribution of the input data in the estimated LDA subspace.* The input data is projected into the subspace spanned by the two significant LDA features estimated using the proposed ALDA algorithm. Increasing the number of iterations improve the estimate of the LDA features and five classes become more separable.

Four features were measured in centimeters from each sample. The measured features are the length and the width of sepal and petal [44]. One class is linearly separable from the other two classes, the latter are not linearly separable from each other. The eigenvalues of $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B$ in descending order are 32.7467, 1.2682, 0.9866, and 0.9866. Although the data set has intrinsic dimensionality of four for the classification, only one feature is significant. The first eigenvalue of $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B$, that is 32.7467, greatly dominates others. Therefore, is clear that almost all the necessary information for the class separability lies on the direction of the eigenvector corresponding to the largest eigenvalue and the rest of the eigenvectors do not play

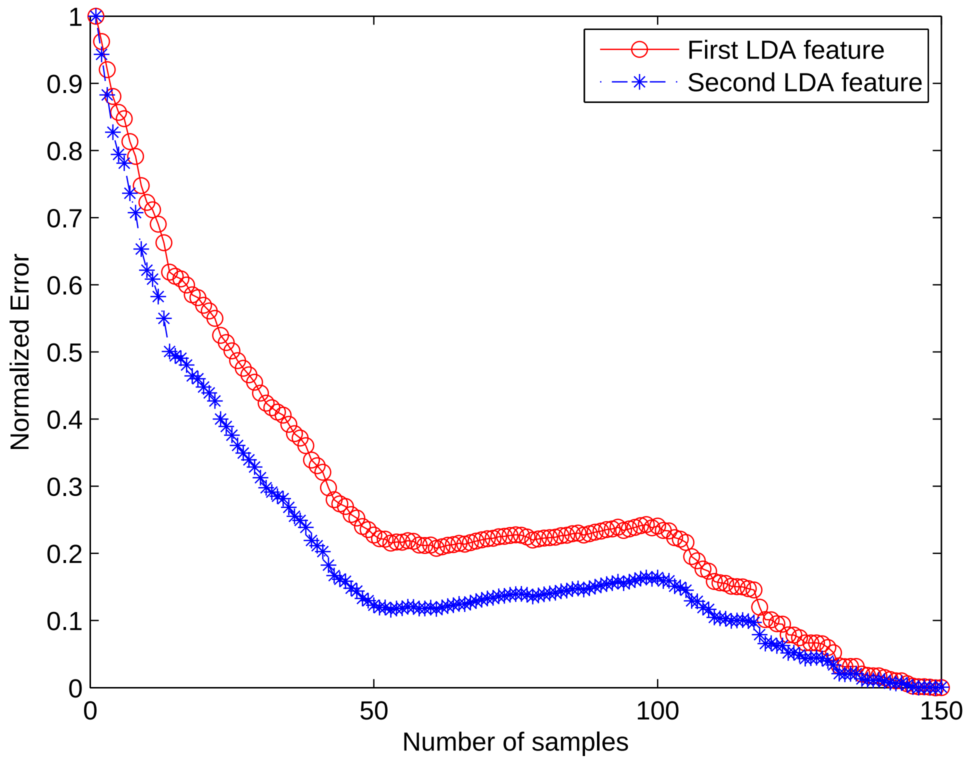

Fig. 13: *Convergence of two significant LDA features for the Iris data set.* The proposed ALDA algorithm is applied to the Iris data set in order to estimate two significant LDA features. It can be observed that the normalized error after 150 iterations is negligible.

an important role for the classification purpose. To get a better visual representation of the data samples in the feature space, we estimate two significant LDA features (LDA feature vectors corresponding to 32.7467 and 1.2682) and project the four dimensional data onto the subspace spanned by the estimated features. In order to achieve a fast convergence, we used the proposed accelerated $\mathbf{\Sigma}^{-1/2}$ algorithm. The data samples are fed to the proposed ALDA algorithm sequentially and the LDA features are estimated in an adaptive manner. Fig. 13 illustrates convergence of two significant LDA features as a function of the number of the training samples. We project the four dimensional samples onto the estimated significant LDA features. Fig. 14 shows distribution of the Iris data set in the subspace spanned by the estimated two significant LDA features. It can be observed from Fig. 14 that the data samples distributed linearly

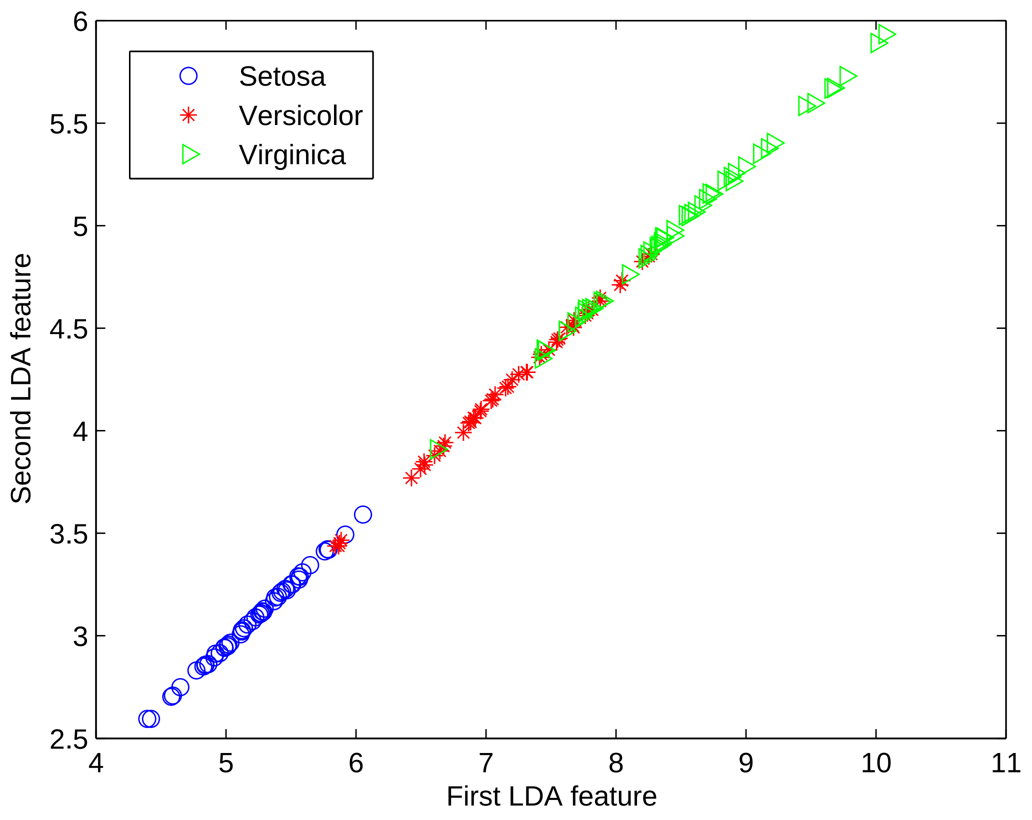

Fig. 14: *Distribution of the Iris data set in the subspace spanned by the estimated LDA features.* The four dimensional samples of the Iris data set are projected onto the subspace spanned by the LDA features estimated using the proposed ALDA algorithm. To get a better understanding of the distribution of data samples in the estimated subspace, we did projection onto the two significant LDA features. The first class, Iris Setosa, is linearly separable from other classes, the two remaining classes overlap each other.

in the estimated two dimensional LDA subspace, that confirms using just the first significant LDA feature would be enough for the classification.

3) Experiment on the Yale face data base B: To test the performance of the proposed ALDA algorithm to extract the significant LDA features, we applied it to the Yale face data base B. The Yale face data base B contains 5760 single light source images of 10 subjects each seen under 576 viewing conditions (9 poses $\times$ 64 illumination conditions) [45][1]. We selected 5 individual subjects and considered 64 images

[1]The Yale face data base B can be accessed online at urlhttp://cvc.yale.edu/projects/yalefacesB/yalefacesB.html

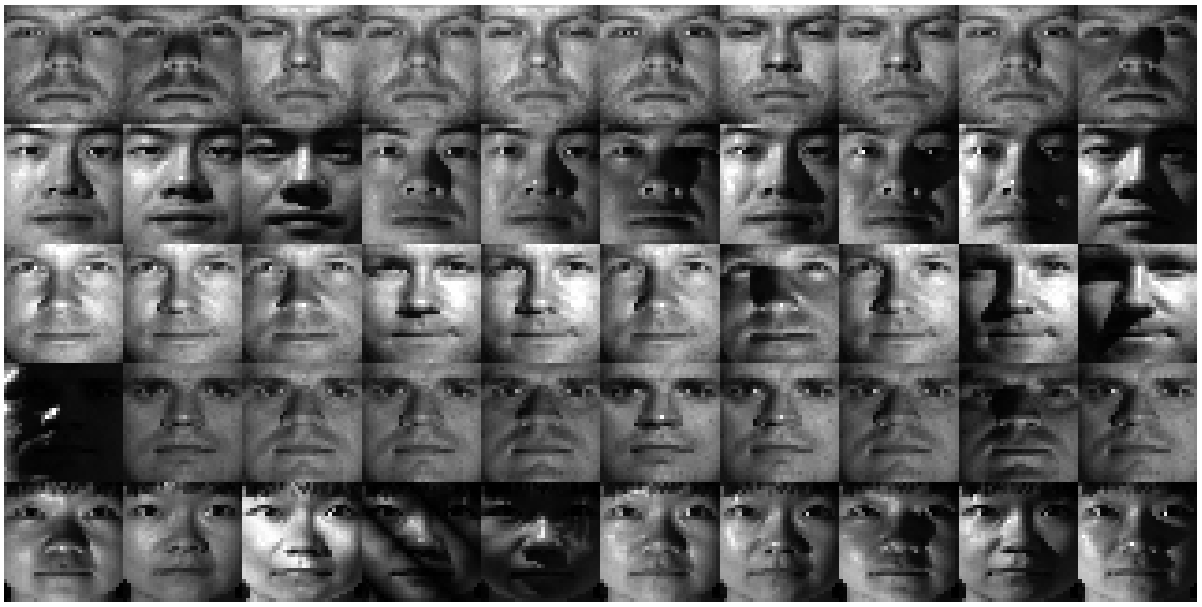

Fig. 15: Sample face images of the selected 5 subjects from the Yale face data base B under different poses and illumination conc

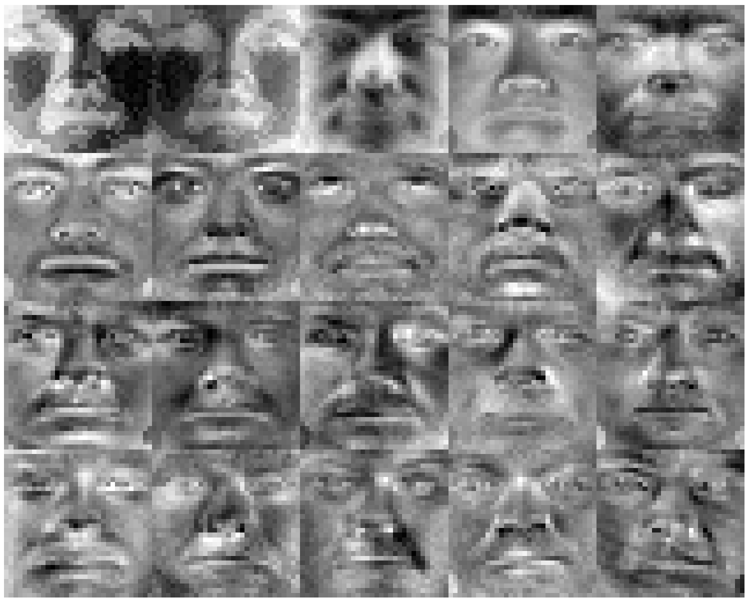

Fig. 16: The first 20 significant Eigenfaces of the selected face images, computed using the PCA algorithm.

for each subject under different illuminations and poses. We cropped every face images to remove the background and resized them to 32×32 pixels. Fig. 15 shows sample images of the selected subjects under different poses and illumination conditions. The histogram for all face images is equalized. The

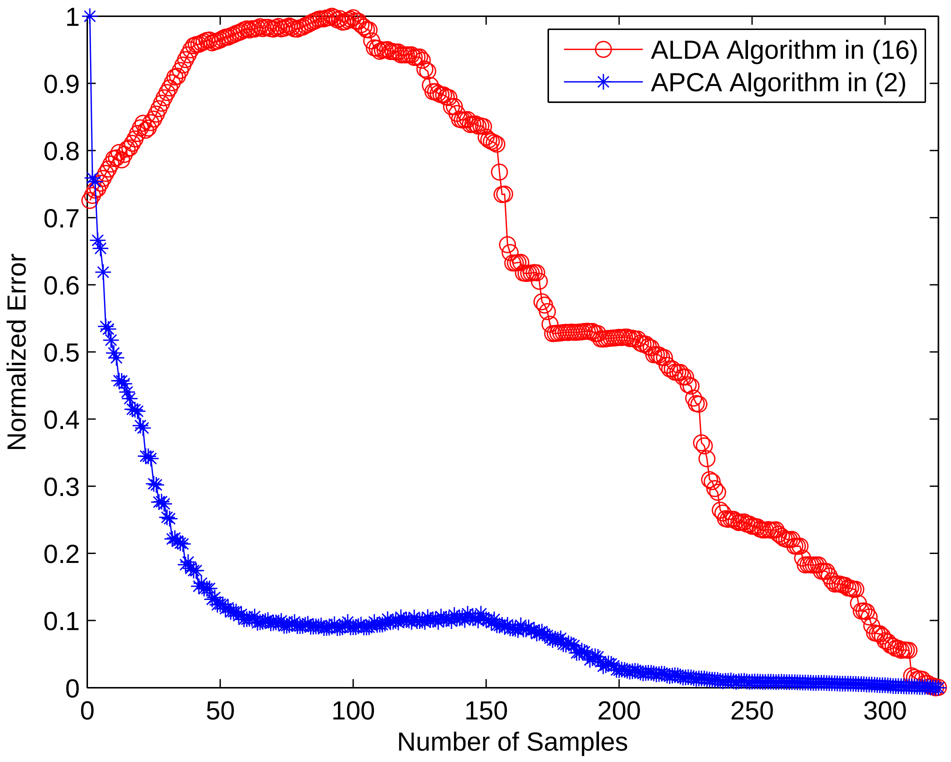

Fig. 17: The red curve shows convergence of the proposed $\mathbf{\Sigma}^{-1/2}$ algorithm in (16) and the blue curve represents the convergence of the APCA algorithm in (2). We obtain an estimate of the LDA features by multiplying the outputs of these two algorithms.

histogram equalization process spreads out the intensity in an image and provides the resultant image to be as flat as possible [46]. Each face image can be considered as a vector in a high dimensional space, i.e. a point in 1024 (32×32) dimensional space in our case. We mean centered face images by computing the mean vector and subtracting it from each image vector. Before applying the proposed ALDA algorithm to the face data base, we need to reduce the dimensionality of the data base. By applying the PCA algorithm to the data base, we found 40 significant eigenvectors of the covariance matrix. Fig. 16 shows significant Eigenfaces [47] resulting from the PCA algorithm. Projection of the face images onto the subspace spanned by 40 significant eigenvectors of the covariance matrix reduces the image vector size into 40. The forty-dimensional vectors are used to train the proposed ALDA algorithm. The reduced sized vectors are fed to the proposed $\mathbf{\Sigma}^{-1/2}$ and APCA algorithms sequentially and train them in an adaptive manner. In this particular experiment, we estimated three significant LDA features and represent the face

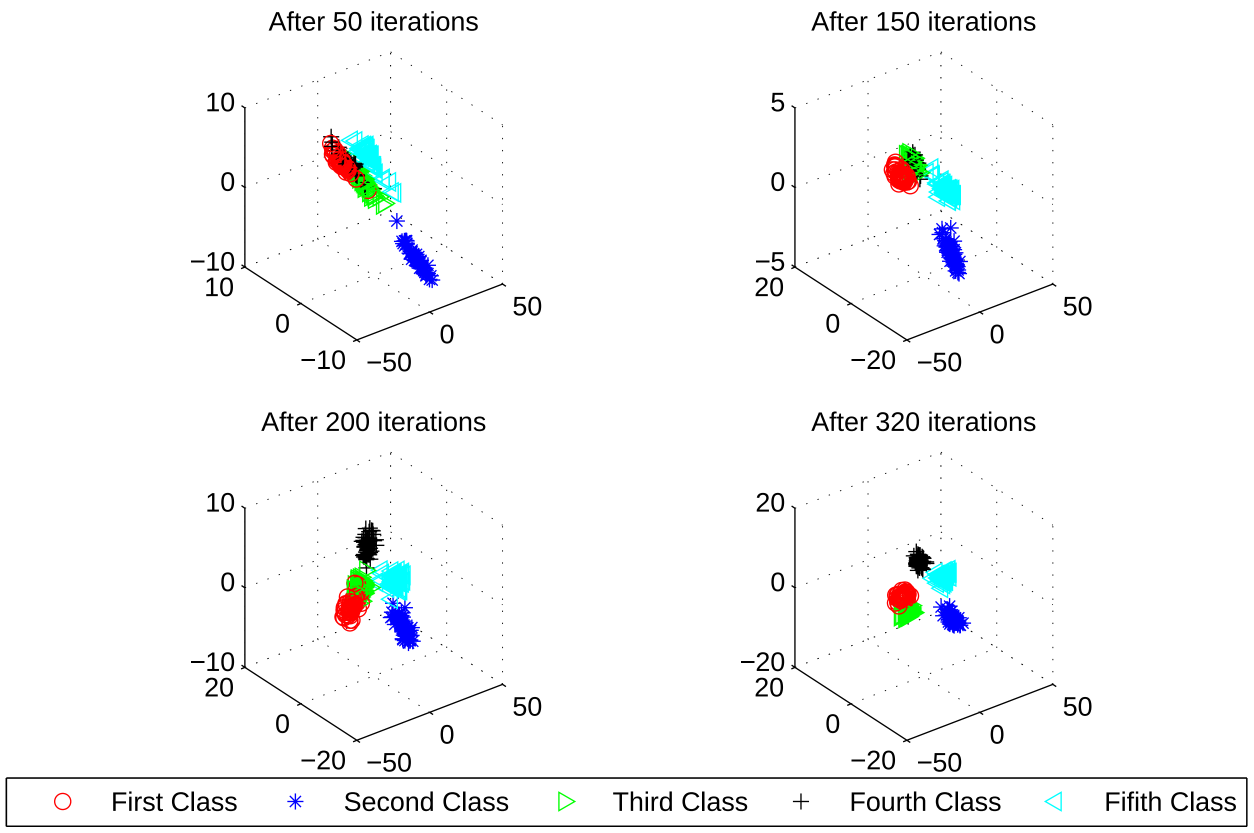

Fig. 18: Distribution of the input samples in the three dimensional estimated LDA feature subspace. It can be observed that the five classes are mixed together after 50 iterations. As the number of iterations increases we get a better estimate of the significant LDA features and classes starts to separate from each other. After 320 iterations five classes are linearly separable in the estimated LDA feature subspace.

images on the subspace spanned by three significant LDA features. Fig. 17 shows the convergence of the proposed $\mathbf{\Sigma}^{-1/2}$ algorithm in (16) and the APCA algorithm in (2), respectively. The algorithm in (16) converges to the square root of the inverse of the within covariance matrix $\mathbf{\Sigma}_W^{-1/2}$ and the APCA algorithm in (2) converges to $\mathbf{\Sigma}^{1/2}\mathbf{\Phi}_{LDA}^{p}$, where p is the number of the desired LDA features (that is three in this experiment). In order to show how the accuracy of the significant LDA estimates affect the distribution of the data samples in the estimated LDA subspace, we project the face vectors onto the subspace spanned by the estimated LDA features during the training phase. Fig. 18 illustrates the distribution of the face images in the subspace spanned by the estimated three significant LDA features after 50, 150, 200, and 320 iterations. It can be observed from Fig. 18 after 50 iterations five classes are mixed and are not linearly separable. By observing new samples and improving the estimate of the LDA features, classes start to separate from each other. As the number of iterations increase, the amount of overlapping between classes reduces and after 320 iterations five classes are almost linearly separable.

V. Conclusion

In this paper, we presented new adaptive algorithms to estimate the square root of the inverse covariance matrix $\mathbf{\Sigma}^{-1/2}$. The proposed algorithms are derived by optimization of an appropriate cost function that is introduced for the first time. The optimal learning rate in each iteration is computed in order to accelerate the convergence rate of the proposed $\mathbf{\Sigma}^{-1/2}$ algorithms. We used the output of $\mathbf{\Sigma}^{-1/2}$ algorithm to construct a new training sequence for an APCA algorithm. It has been shown that the product of the outputs of the APCA algorithm and $\mathbf{\Sigma}^{-1/2}$ algorithm converges to the desired LDA features. The convergence rate of the proposed accelerated ALDA algorithms to estimate the LDA features is compared with that of the current algorithms. The simulation results indicate that the proposed accelerated ALDA algorithms give an accurate estimate of the LDA features in fewer iterations compared with that of the current algorithms. Furthermore, existence of the cost function makes it possible evaluate the accuracy of the estimates obtained by using different initial conditions and learning rates. A sequence of the input data is used to train the ALDA algorithms in an adaptive manner. The adaptive nature of the proposed ALDA algorithms makes them appropriate for online applications such as mobile robotic or online face recognition. The proposed $\mathbf{\Sigma}^{-1/2}$ algorithms require to start from a symmetric and positive definite matrix that commutes with the covariance matrix. This is the only constraint that needs to be satisfied in order to guaranty the convergence of the proposed algorithms. The constraint on the initial condition can be satisfied by choosing the initial estimate $\mathbf{W}_0$ to be the identity matrix. Note that, the proposed ALDA algorithms have limitations of the LDA method. They give us at most $K-1$ features, where K is the number of classes. Therefore for applications that more features are needed, some other methods must be employed to provide additional features. Similar to the LDA method, the proposed ALDA algorithms implicitly assume Gaussian distribution for the available classes. If the distributions are significantly non-Gaussian, the LDA features may not be useful for the classification purpose. The proposed ALDA algorithms are tested on the real world data set with the small number of classes (four classes for the Iris data set and five classes for the Yale face data base B). It is obvious that when the number of classes increases they may not be linearly separable. In this case we have to use nonlinear techniques such as neural networks for the classification of the data samples. Design and training a neural network with many input can be complicated and time consuming. The proposed ALDA algorithm can be used as a preprocessing step to reduce the dimensionality of the input data. Then the outputs of the ALDA algorithm, that lie on an lower dimensional subspace spanned by the LDA features, are fed into a neural network for the nonlinear classification task.

Appendix A

We prove the following lemma for the cost function $J(\mathbf{W})$ defined in (10),

Lemma 1. *Let $\mathbf{\Sigma}$ be the covariance matrix of the input samples. Let $\mathbf{S}$ denote the set of all symmetric and positive definite matrices $\mathbf{W}$ that commute with the square root of the covariance matrix, i.e., $\mathbf{S} = \{\mathbf{W} : |\mathbf{W}^t = \mathbf{W}, \mathbf{x}^t\mathbf{W}\mathbf{x} > 0,$ for all non-zero $\mathbf{x} \in \mathbb{R}^n,$ and $\mathbf{W}\mathbf{\Sigma}^{1/2} = \mathbf{\Sigma}^{1/2}\mathbf{W}\}$. Then the cost function $J(\mathbf{W}) : \mathbf{S} \to \mathbb{R}$ is a convex function and its global minimum occurs at $\mathbf{W} = \mathbf{\Sigma}^{-1/2}$.*

Proof: If the positive definite and symmetric matrix $\mathbf{W}$ commutes with the square root of the covariance matrix, i.e. $\mathbf{W}\mathbf{\Sigma}^{1/2} = \mathbf{\Sigma}^{1/2}\mathbf{W}$, then we have

$$\mathbf{W}\mathbf{\Sigma} = \mathbf{W}\mathbf{\Sigma}^{1/2}\mathbf{\Sigma}^{1/2} = \mathbf{\Sigma}^{1/2}\mathbf{W}\mathbf{\Sigma}^{1/2} = \mathbf{\Sigma}^{1/2}\mathbf{\Sigma}^{1/2}\mathbf{W} = \mathbf{\Sigma}\mathbf{W}. \tag{A.1}$$

That means the matrix $\mathbf{W}$ commutes also with the covariance matrix $\mathbf{\Sigma}$, i.e. $\mathbf{W}\mathbf{\Sigma} = \mathbf{\Sigma}\mathbf{W}$. From equation (9), we have

$$\begin{aligned} J(\mathbf{W}) &= \frac{1}{3}tr[(\mathbf{W}\mathbf{\Sigma}^{1/2} - \mathbf{I})^2(\mathbf{W} + 2\mathbf{\Sigma}^{-1/2})] \\ &= \frac{1}{3}tr[(\mathbf{W}\mathbf{\Sigma}^{1/2}\mathbf{W}\mathbf{\Sigma}^{1/2} - 2\mathbf{W}\mathbf{\Sigma}^{1/2} + \mathbf{I})(\mathbf{W} + 2\mathbf{\Sigma}^{-1/2})]. \end{aligned} \tag{A.2}$$

Using the commutative property between $\mathbf{W}$ and $\mathbf{\Sigma}$, we can simplify (A.2) as follows

$$\begin{aligned}
J(\mathbf{W}) &= \frac{1}{3} tr[(\mathbf{W}^2\mathbf{\Sigma} - 2\mathbf{W}\mathbf{\Sigma}^{1/2} + \mathbf{I})(\mathbf{W} + 2\mathbf{\Sigma}^{-1/2})] \\
&= \frac{1}{3} tr[\mathbf{W}^3\mathbf{\Sigma} + 2\mathbf{W}^2\mathbf{\Sigma}^{1/2} - 2\mathbf{W}^2\mathbf{\Sigma}^{1/2} - 4\mathbf{W} + \mathbf{W} + 2\mathbf{\Sigma}^{-1/2}] \\
&= \frac{1}{3} tr[\mathbf{W}^3\mathbf{\Sigma} + 2\mathbf{\Sigma}^{-1/2} - 3\mathbf{W}] \\
&= \frac{1}{3} tr(\mathbf{W}^3\mathbf{\Sigma}) + \frac{2}{3} tr(\mathbf{\Sigma}^{-1/2}) - tr(\mathbf{W}). \qquad \text{(A.3)}
\end{aligned}$$

The first derivative of the cost function $J(\mathbf{W})$ with respect to $\mathbf{W}$ is given by

$$\begin{aligned}
\frac{\partial J(\mathbf{W})}{\partial \mathbf{W}} &= \frac{\mathbf{\Sigma}\mathbf{W}^2 + \mathbf{W}\mathbf{\Sigma}\mathbf{W} + \mathbf{\Sigma}\mathbf{W}^2}{3} - \mathbf{I} \\
&= \mathbf{\Sigma}\mathbf{W}^2 - \mathbf{I}, \qquad \text{(A.4)}
\end{aligned}$$

For the last equality, we used the commutation property given in (A.1). Equating the first derivative of the cost function $J(\mathbf{W})$ to zero, we get $\mathbf{W} = \mathbf{\Sigma}^{-1/2}$. Hence, $\mathbf{W} = \mathbf{\Sigma}^{-1/2}$ is the only stationary point of the cost function $J(\mathbf{W})$. The product of two positive definite matrices is also a positive definite matrix and we have the same property for the trace of a positive definite matrix. Therefore, the cost function in (9), that is trace of the product of two positive definite matrices, is a positive definite matrix and we have $J(\mathbf{W}) \geq 0$. The cost function $J(\mathbf{W})$ at the stationary point $\mathbf{W} = \mathbf{\Sigma}^{-1/2}$ is zero, therefore $\mathbf{\Sigma}^{-1/2}$ is the only global minimum of the cost function. An alternative way to show that the stationary point $\mathbf{\Sigma}^{-1/2}$ is a global minimum of the cost function $J(\mathbf{W})$ is using the following theorem [38]

Theorem 1. *A real valued, twice differentiable function $J : \mathbf{S} \to \mathbb{R}$ defined on an open convex set $\mathbf{S}$ is convex if and only if its Hessian matrix is positive semi-definite.*

The Hessian matrix of the cost function $J(\mathbf{W})$ is given by

$$\mathcal{H}(J(\mathbf{W})) = \mathbf{\Sigma} \otimes \mathbf{W} + \mathbf{\Sigma}\mathbf{W} \otimes \mathbf{I}, \qquad \text{(A.5)}$$

where $\mathbf{I}$ is the identity matrix and $\otimes$ denote the Kronecker product. Since $\mathbf{W}$ and $\mathbf{\Sigma}$ are positive definite matrices and commute, then their product also is a positive definite matrix [38]. The Kronecker product of two positive definite matrices is also a positive definite matrix, therefore the Hessian matrix $\mathcal{H}$ given in (A.5) is also a positive definite matrix. Using theorem 1 the cost function $J(\mathbf{W})$ is a convex function and $\mathbf{W} = \mathbf{\Sigma}^{-1/2}$ is a absolute minimum of the cost function. ■

Appendix B

The cost function $J(\mathbf{W})$ at $k+1$-th iteration is given be

$$J(\mathbf{W}_{k+1}) = \frac{tr(\mathbf{W}_{k+1}^3 \mathbf{\Sigma})}{3} - tr(\mathbf{W}_{k+1}) + \frac{2}{3} tr(\mathbf{\Sigma}^{-1/2}). \tag{B.1}$$

The estimate at $k+1$-th iteration is computed using the following adaptive algorithm

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta_k (\mathbf{I} - \mathbf{W}_k \mathbf{\Sigma} \mathbf{W}_k). \tag{B.2}$$

Let $\mathbf{G}_k = \mathbf{I} - \mathbf{W}_k \mathbf{\Sigma} \mathbf{W}_k$. Introducing $\mathbf{G}_k$ into (B.1) and using A.1 yields,

$$\begin{aligned} J(\mathbf{W}_{k+1}) &= \frac{tr((\mathbf{W}_k + \eta_k \mathbf{G}_k)^3 \mathbf{\Sigma})}{3} - tr(\mathbf{W}_k + \eta_k \mathbf{G}_k) + \frac{2}{3} tr(\mathbf{\Sigma}^{-1/2}) \\ &= \frac{tr(\mathbf{W}_k^3 \mathbf{\Sigma} + 3\eta_k \mathbf{W}_k^2 \mathbf{G}_k \mathbf{\Sigma} + 3\eta_k^2 \mathbf{W}_k \mathbf{G}_k^2 \mathbf{\Sigma} + \eta_k^3 \mathbf{G}_k^3 \mathbf{\Sigma})}{3} - tr(\mathbf{W}_k + \eta_k \mathbf{G}_k) + \frac{2}{3} tr(\mathbf{\Sigma}^{-1/2}). \end{aligned} \tag{B.3}$$

We take derivative of the cost function in B.3 with respect to the learning rate and equate it to zero in order to find the optimal learning rate. We obtain

$$\begin{aligned} \frac{\partial J(\mathbf{W}_{k+1})}{\partial \mathbf{W}_{k+1}} &= tr(\mathbf{W}_k^2 \mathbf{G}_k \mathbf{\Sigma}) + 2\eta_k tr(\mathbf{W}_k \mathbf{G}_k^2 \mathbf{\Sigma}) - tr(\mathbf{G}_k) + \eta_k^2 tr(\mathbf{G}_k^3 \mathbf{\Sigma}) \\ &= c_k + b_k \eta_k + a_k \eta_k^2, \end{aligned} \tag{B.4}$$

where $a_k = tr(\mathbf{G}_k^3 \mathbf{\Sigma})$, $b_k = 2tr(\mathbf{W}_k \mathbf{G}_k^2 \mathbf{\Sigma})$, and $c_k = tr(\mathbf{W}_k^2 \mathbf{G}_k \mathbf{\Sigma}) - tr(\mathbf{G}_k)$.

Appendix C

Lemma 2. *Let $\mathbf{x}_k \in \mathbb{R}^n, k = 1, 2, \ldots$ represent the input samples. Let $\mathbf{m}_k$ be the total mean estimate at k-th iteration. Let $\mathbf{m}_k^{\omega_{\mathbf{x}_k}}$ denote the estimate of the sample mean of the class that $\mathbf{x}_k$ belongs to it at k-th iteration. Let the sequence $\{\mathbf{y}\}_{k=1,2,\ldots}$ defined by $\mathbf{y}_k = \mathbf{x}_k - \mathbf{m}_k^{\omega_{\mathbf{x}_k}}$ Define the sequence $\{\mathbf{z}_k\}_{k=1,2,\ldots}$ to be $\mathbf{z}_k = \mathbf{x}_k - \mathbf{m}_k$. Let the sequence $\{\mathbf{u}_k\}_{k=1,2,\ldots}$ defined by $\mathbf{u}_k = \mathbf{W}_k \mathbf{z}_k$, where $\mathbf{W}_k$ is the output of a*

$\mathbf{\Sigma}^{-1/2}$ *algorithm trained by the sequence* $\{\mathbf{y}_k\}$*. Then the correlation of the sequence* $\{\mathbf{y}_k\}$ *as* k *goes to infinity converges to the within-class scatter matrix* $\mathbf{\Sigma}_W$*. Furthermore, limit of the correlation matrix for the sequence* $\{\mathbf{u}_k\}$ *converges to* $\mathbf{\Sigma}_W^{-1/2}\mathbf{\Sigma}\mathbf{\Sigma}_W^{-1/2}$.

Proof: We first prove that the correlation of the sequence $\{\mathbf{y}_k\}$ converges to the within-class scatter matrix as k goes to infinite. The limit of the correlation matrix of the sequence $\{\mathbf{y}_k\}$ is computed as follows

$$\begin{aligned}\lim_{k\to\infty}\mathbf{y}_k\mathbf{y}_k^t &= \lim_{k\to\infty} E[(\mathbf{x}_k - \mathbf{m}_k^{\omega_{\mathbf{x}_k}})(\mathbf{x}_k - \mathbf{m}_k^{\omega_{\mathbf{x}_k}})^t] \\ &= E[(\mathbf{x}_k - \mathbf{m}^{\omega_{\mathbf{x}_k}})(\mathbf{x}_k - \mathbf{m}^{\omega_{\mathbf{x}_k}})^t] \\ &= \sum_{i=1}^{K} P(\omega_i)E[(\mathbf{x}_k - \mathbf{m}^{\omega_{\mathbf{x}_k}})(\mathbf{x}_k - \mathbf{m}^{\omega_{\mathbf{x}_k}})^t|\omega_{\mathbf{x}_k} = \omega_i] \\ &= \sum_{i=1}^{K} P(\omega_i)\mathbf{\Sigma}^{\omega_i} \\ &= \mathbf{\Sigma}_W, \end{aligned} \tag{C.1}$$

where K is the number of available classes, $\omega_i i = 1, \ldots, K$ denote the i-th class, and $\mathbf{\Sigma}^{\omega_i}$ denote the covariance matrix for the class ω_i.

The correlation matrix for the sequence $\{\mathbf{u}_k\}$ is computed as follows

$$\begin{aligned}\lim_{k\to\infty}\mathbf{u}_k\mathbf{u}_k^t &= \lim_{k\to\infty} E[\mathbf{W}_k(\mathbf{x}_k - \mathbf{m}_k)(\mathbf{x}_k - \mathbf{m}_k^t\mathbf{W}_k^t] \\ &= E[\mathbf{\Sigma}_W^{-1/2}(\mathbf{x} - \mathbf{m})(\mathbf{x} - \mathbf{m})^t(\mathbf{\Sigma}_W^{-1/2})^t] \\ &= \mathbf{\Sigma}_W^{-1/2}E[(\mathbf{x} - \mathbf{m})(\mathbf{x} - \mathbf{m})^t](\mathbf{\Sigma}_W^{-1/2})^t \\ &= \mathbf{\Sigma}_W^{-1/2}\mathbf{\Sigma}\mathbf{\Sigma}_W^{-1/2}. \end{aligned} \tag{C.2}$$

■

References

[1] K. Fukunaga, *Introduction to Statistical Pattern Recognition,* 2nd Edition, Academic Press, New York, 1990.

[2] L. Chen, H. M. Liao, M. Ko, J. Lin,G. Yu, "New LDA based face recognition system which can solve the small sample size problem," *Pattern Recognition,* vol. 33, no. 10, pp. 1713-1726, 2000.

[3] J. Lu, K. N. Platantinos, A. N. Venetsanopoulos, "Face recognition using LDA-based algorithms," *IEEE Trans. Neural Networks,* vol. 14, no. 1, pp. 195-200, 2003.

[4] H. Yu,J. Yang, "A direct LDA algorithm for high-dimensional data with application to face recognition," *Pattern Recognition,* vol. 34, no. 10, pp. 2067-2070, 2001.

[5] Y. Koren, L. Carmel, "Visualization of labeled data using linear transformation," *in Proc. the Ninth IEEE conf. on Information visualization,* pp. 121-128, Oct. 2003.

[6] C. Chang, H. Ren, "An Experimented-based quantitative and comparative analysis of target detection and image classification algorithms for hyper-spectral imagery," *IEEE Trans. Geoscience and Remote Sensing,* vol. 38, no. 2, pp. 1044-1063, 2000.

[7] C. C. Aggarwal, J. Han, J. Wang, P. S. Yu, "On demand classification of data streams," *in Proc. ACM SIGKDD Int. Conf. Knowledge discovery data mining,* pp. 503-508, Aug. 2004.

[8] D. L. Swets, J. J. Weng, "Using Discriminant eigen-feature for image retrieval," *IEEE Trans. Pattern Analysis and Machine Intelliigence,* vol. 18, no. 8, pp. 831-836, 1996.

[9] W. Zhao, R. Chellappa, A. Krishnaswamy, "Discriminant analysis of principal components for face recognition," *in Proc. IEEE Int. Conf. on Automatic face and gesture recognition,* Nara, Japan, pp. 336-341, 1998.

[10] P. N. Belhumeur, J. P. Hespanha, D. J. Kriegman, "Eigen-faces vs. Fisher-faces: Recognition using class specific linear projection," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, pp. 711-720, 1997.

[11] A. M. Martinez, A. C. Kak, "PCA versus LDA," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 2, pp.228-233, 2001.

[12] P. Hall, A. D. Marshall, R. Martin, "Incremental eigen analysis for classification," *in Proc. Brit. Machine Vision Conf.,* vol. 1, pp. 286-295, 1998.

[13] P. Hall, A. D. Marshal, R. Martin, "Merging and splitting eigen space models," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 9, pp. 1042-1049, 2000.

[14] Y. Miao, Y. Hua, "Fast subspace tracking and neural network learning by a novel information criterion," *IEEE Trans. Signal Processing,* vol. 46, pp. 1964-1979, 1998.

[15] S. Bannour, M. R. Azimi-Sadjadi, "Principal component extraction using recursive least squares learning," *IEEE Trans. Neural Network,* vol. 6, pp. 457-469, 1995.

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

[17] J. Weng, Y. Zhang, W. S. Hwang, "Candid covariance free incremental principal component analysis" *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 25, no. 8, pp 1034-1040, Aug. 2003.

[18] E. Oja, j. Karhunen, "On stochastic approximation of eigenvectors and eigen-values of the expectation of a random matrix," *Journal of Mathematical Analysis and Applications,* vol. 106, pp. 69-84, 1990.

[19] E. Oja, "Principal components, minor components, and linear neural networks," *Neural Networks,* vol.5, pp. 927-935, 1992.

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

[21] C. Chatetrjee, A. Kang, V. Roychodhury, "Algorithms for Accelerated Convergence of Adaptive PCA," *IEEE Trans. on Neural Networks,* vol. 11, no. 2, pp. 338-355, 2000.

[22] S. Pang, S. Ozawa, N. Kasabov, "Incremental linear discriminant analysis for classification of data streams," *IEEE Trans. on System, Man and Cybernetics-Part B,* vol. 35, no. 5, pp. 905-914, 2005.

[23] T. Kim, S. Wong, B. Stenger, J. Kittler, R. Cipolla, "Incremental Linear Discriminant Analysis Using Sufficient Spanning Set Approximation," *in Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition (CVPR),* pp. 1-8, June 2007.

[24] J. Ye, Q. Li, H. Xiong, H. Park, R. Janardan, V. Kumar, "IDR/QR: An incremental dimension reduction algorithm via QR decomposition," *IEEE Trans. on Knowledge and Data Engineering,* vol. 17, no. 9, 2005.

[25] J. Mao, A.K. Jain, "Discriminant analysis neural networks," *in Proc. IEEE Int. Conf. on Neural Networks,* CA, pp. 300-305, 1993.

[26] C. Chatterjee, V. P. Roychowdhurry, "On self-organizing algorithm and networks for class separability features," *IEEE Trans. Neural Network,* vol. 8, no. 3, pp. 663-678, 1997.

[27] G.K. Demir, K. Ozmehmet, "On-line local learning algorithm for linear discriminant analysis," *Pattern Recognition Letters,* vol. 26, pp. 421-431, 2005.

[28] H. Abrishami Moghaddam, M. Matinfar, S. M. Sajad Sadough, K. Amiri Zadeh, "Algorithms and networks for accelerated convergence of adaptive LDA," *Pattern Recognition,* vol. 38, no. 4, pp. 473-483, 2005.

[29] H. Abrishami Moghaddam, M. Matinfar, "Fast adaptive LDA using quasi-Newton algorithm," *Pattern Recognition Letters,* vol. 28, no. 4, pp. 613-621, 2007.

[30] Y. Rao, N. Principe, J. C. Wong, "Fast RLS like algorithm for generalized eigen decomposition and its applications," *Journal of VLSI Signal processing systems,* vol. 37, no. 3, pp. 333-344, 2004.

[31] Y. Aliyari Ghassabeh, H. Abrishami Moghaddam, "Adaptive algorithms and networks for optimal feature extraction from Gaussian data," *Pattern Recognition Letters,* vol. 31, pp. 1331-1341, Aug. 2010.

[32] H. Hongo, N. Yasumoto, Y. Niva, K. Yamamoto, "Hierarchical face recognition using an adaptive discriminant space," *in Proc. IEEE Int. Conf. Computers Communications, Control and Power Engineering (TENCON),* vol. 1, pp. 523-528, 2002.

[33] S. Ozawa, S. L. Toh, S. Abe, S. Pang, N. Kasabov, "Incremental learning of feature space and classifier for face recognition," *Neural Networks,* vol. 18, pp. 575-584, 2005.

[34] S. Theodoridis, *Pattern Recognition.* 2nd Edition, Academic Press, New York, 2003.

[35] M. Woelfel, J. McDonough, *Distant Speech Recognition.* Wiley, 2009.

[36] L. Ljung, "Analysis of recursive stochastic algorithms," *IEEE Trans. Automatic Control,* vol. 22, pp. 551-575, 1977.

[37] K. Diamantaras, S. Kung, *Principal Component Neural Networks, Theory and Applications.* Wiley, New York, 1996.

[38] J. R. Magnus, H. Neudecker, *Matrix Differential Calculus.* John Wiley, 1999.

[39] H. J. Kushner, D. S. Clarck, *Stochastic approximation methods for constrained and unconstrained systems* Springer-Verlag, New York, 1978.

[40] A. Benveniste, M. Metivier, P. Priouret, *Adaptive algorithms and stochastic approximations.* 2nd Edition, Academic Press, New York, 1990.

[41] T. Okada, S. Tomita, "An optimal orthonormal system for discriminant analysis," *Pattern Recognition,* vol. 18, no.2, pp. 139-144, 1985.

[42] E. R. Hruschka, N. F. Ebecken,"Extracting rules from multilayer perceptrons in classification problems: A clustering-based approach," *Neurocomputing,* vol. 70, pp. 384397, Dec. 2006.

[43] C. H. Wang, Y. Y, Chi, "Dynamic Optimal Training of A Three Layer Neural Network with Sigmoid Function," *in Proc. IEEE int. conf. on Networking, Sensing and Control,* Fort Lauderdale, USA, pp. 392-397, Aug. 2006.

[44] http://en.wikipedia.org/wiki/Iris_flower_data_set

[45] A. S. Georghiades, P. N. Belhumeur, D. J. Kriegman, "From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 6, pp/ 643-660, Jun. 2001.

[46] N. Rajkumar, S. Vijayakumar, C. Murukesh, "Intellectually combined face recognition using curvelet based principle component analysis for feature

extraction and Bayesian Classifier," *in Proc. IEEE int. conf. on Signal Processing, Communication, Computing and Networking Technologies (ICSCCN),* Chennai, India, pp. 374-378, Sep. 2011.

[47] M. Turk, A. Pentland, "Eigenfaces for face recognition," *Journal Cognitive Neuroscience,* vol. 3, no.1, pp 71-86, 1991.