Pergamon

*Pattern Recognition*, Vol. 31, No. 7, pp. 881–888, 1998

0031-3203/98 $19.00 + 0.00

PII: S0031-3203(97)00108-8

# RECURSIVE DIMENSIONALITY REDUCTION USING FISHER'S LINEAR DISCRIMINANT

WENDY L. POSTON[†,*] and DAVID J. MARCHETTE[‡]

[†]NSWCDD, G33, Dahlgren, VA 22448, U.S.A.
[‡]NSWCDD, B10, Dahlgren, VA 22448, U.S.A.

(*Received* 9 *January* 1997; *in revised form* 8 *September* 1997)

**Abstract**—Dimensionality reduction is an important part of the pattern recognition process. It would be very useful to have a recursive form for dimensionality reduction that is suitable for implementation on massive data sets and real-time automatic pattern recognition systems. It would also be beneficial to have a version where the dimensionality reduction can be updated based on new partially identified data that are obtained in real systems. Versions of Fisher's Linear Discriminant for dimensionality reduction that address these problems are derived in this article. © 1998 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved

Dimensionality reduction  Fisher's Linear Discriminant  Expectation-maximization algorithm

## 1. INTRODUCTION

In many applications, numerous features are obtained in an attempt to ensure accurate classification of the unknown classes. Sometimes, these features must be reduced to a smaller number before classification schemes can be applied, because classifiers can become computationally and analytically unmanageable in high dimensions. For instance, some classification techniques use Bayes' decision theory where *a posteriori* probabilities are estimated based on data labeled with the true class membership. Because of the curse of dimensionality,[1] probability density estimation is best done with three or fewer variables. Hence, the dimensionality of the feature space must often be reduced before classification is undertaken, and some pattern recognition systems include dimensionality reduction as well.

It would be useful to have a recursive form for dimensionality reduction that can be applied to massive data sets, where data cannot be stored prior to processing. An example of this is the Earth Observing Satellite, where gigabytes of data are obtained every minute and must be processed as they are received. It would also be beneficial to have a version where the dimensionality reduction can be updated based on new partially identified data. For example, if new data are obtained that can be partially classified (e.g. in an image, a pixel is classified with a certain probability as land or water), then the dimensionality reduction method could be updated based on this partial knowledge.

Several techniques are available for dimensionality reduction. Some of these are Fisher's Linear Discriminant (FLD),[2] principal components,[3] and projection pursuit.[4] In this paper, two recursive versions of the FLD are derived. The first is based on the assumption that it is known which class each data point belongs to. This could be used with massive data sets where each observation is labeled with the true class and must be processed as it is obtained to build the classifiers. The other version recursively updates the FLD based on partially classified data. This new method uses the recursive update equations based on the Expectation–Maximization (EM)[5] algorithm to obtain the FLD projection.

A discussion of the FLD and the EM algorithm is provided to facilitate the derivation of the recursive FLD. The recursive equations for the two-class FLD when we know which group a new observation belongs to and when we have partial class knowledge are derived. The next section derives similar equations for the multi-class FLD. The appendix provides recursive update equations for the inverse and the determinant of the covariance matrix based on the EM algorithm. These are useful in evaluating the class-conditional probabilities, which are needed for the partial knowledge FLD. Monte Carlo simulations that examine the behavior of the partial knowledge FLD are presented in the last section.

## 2. BACKGROUND

### 2.1. *Fisher's Linear Discriminant* (*FLD*)

The description of the FLD given in this section follows that in Duda and Hart[2]. There are two types of FLD reduction techniques, which are based on the number of classes under consideration. The simplest is the two-class case, where the dimensionality is

*Author to whom correspondence should be addressed.

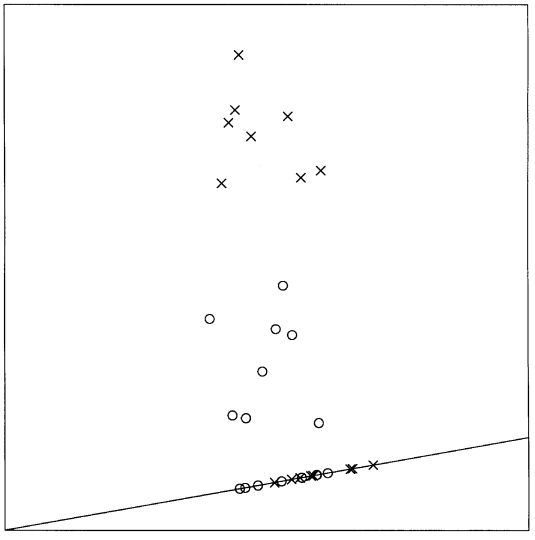

Fig. 1. The data are projected onto an arbitary line and cannot be separated.[2]

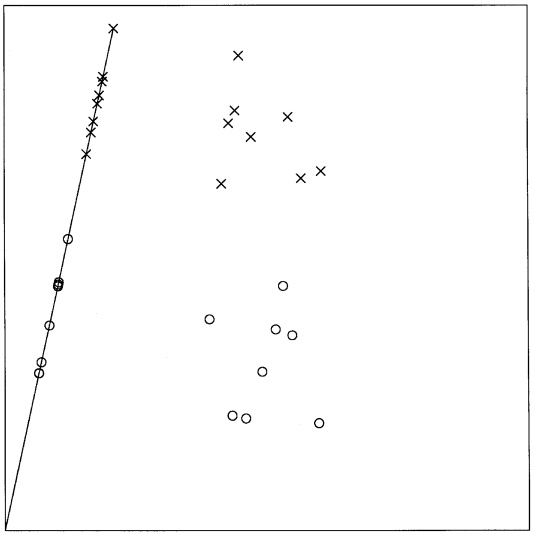

Fig. 2. The data are projected onto the FLD line and are still easily separated.[2]

reduced to one. This is accomplished by projecting the $p$-dimensional data onto a line such that the projected data are well separated. The orientation of the projection line is important because the data might be well-separated in $p$-dimensional space, but become overlapped in the one-dimensional (1-D) space. An example of this is shown in Fig. 1, where a data set is projected onto a line, and it is now impossible to separate the different classes. However, by choosing the projection in a certain orientation, it is possible to project the data onto a line and still have a sample that is well separated, as illustrated in Fig. 2.

Assume we have a set of $n$ $P$-dimensional data points, each one labeled with the true class membership and that the covariances of each group are the same. We are interested in finding the projection w, such that the classes are separated as much as possible. This projection is given by

$$y = \mathbf{w}^{\mathrm{T}}\mathbf{x}, \tag{1}$$

where $y$ is a scalar, $\mathbf{w}$ is a $p \times 1$ transformation vector, and $\mathbf{x}$ is a $p \times 1$ data vector. A sensible choice for a measure of separation between classes is the difference of the sample means for each projected class. Using this criterion, the FLD is defined as the linear function $\mathrm{w}^{\mathrm{T}}\mathrm{x}$ for which the following is maximized:

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_B w}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_W \mathbf{w}}. \tag{2}$$

The between-class scatter matrix is defined as

$$\mathbf{S}_B = (\mathbf{m}^{(1)} - \mathbf{m}^{(2)})\,(\mathbf{m}^{(1)} - \mathbf{m}^{(2)})^{\mathrm{T}}, \tag{3}$$

where $\mathbf{m}^{(i)}$ denotes the sample mean for the $i$th class. The within-class scatter matrix is given by

$$\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2, \tag{4}$$

where an individual class scatter matrix is

$$\mathbf{S}_i = \sum_{x \in X_i} (\mathbf{x} - \mathbf{m}^{(i)})\,(\mathbf{x} - \mathbf{m}^{(i)})^{\mathrm{T}}. \tag{5}$$

The solution that maximizes equation (2) is given by[2]

$$\mathbf{w} = \mathrm{S}_W^{-1}\,(\mathbf{m}^{(1)} - \mathbf{m}^{(2)}). \tag{6}$$

The $c$-class problem is more complicated. In this case, the FLD projects data from a $P$-dimensional space to a $(c-1)$-dimensional space, where it is assumed that $c \leq p$. The class sample means and the within-class scatter matrix are easily generalized to $c$ classes. These are

$$\mathbf{m}^{(i)} = \frac{1}{n^i} \sum_{x \in X_i} \mathbf{x} \quad i = 1, \ldots, c \tag{7}$$

and

$$\mathbf{S}_W = \sum_{i=1}^{c} \mathbf{S}_i, \tag{8}$$

where $\mathbf{S}_i$ is given by equation (5) and $n_i$ denotes the number of data points in the $i$-th class. The between-class scatter matrix is defined somewhat differently from the two-class case as

$$\mathbf{S}_B = \sum_{i=1}^{c} n_i\,(\mathbf{m}^{(i)} - \mathbf{m})\,(\mathbf{m}^{(i)} - \mathbf{m})^{\mathrm{T}}, \tag{9}$$

where $\mathbf{m}$ is the global sample mean.

In the $c$-class situation, the FLD projection is calculated from the solution to the following eigenvalue problem

$$\mathbf{S}_B \mathbf{w}_i = \lambda_i \mathbf{S}_W \mathbf{w}_i. \tag{10}$$

If the within-class scatter matrix is non-singular, then this generalized eigenvalue problem can be converted to a conventional one. That is not necessarily desirable, since it involves the computation of an inverse.

Therefore, it is better to find the eigenvalues of the following characteristic polynomial

$$|\mathbf{S}_B - \lambda_i \mathbf{S}_W| = 0; \qquad i = 1, \ldots, p \tag{11}$$

and solve

$$(\mathbf{S}_B - \lambda_i \mathbf{S}_W)\, \mathbf{w}_i = 0 \tag{12}$$

for the eigenvectors $\mathbf{w}_i$. These eigenvectors are the columns of the transformation matrix $\mathbf{W}$, and the dimensionality of the data points are reduced to $c$-1 using the following transformation

$$\mathbf{y} = \mathbf{W}^{\mathrm{T}}\mathbf{x}, \tag{13}$$

where $\mathbf{y}$ is a $(c-1)\times 1$ vector and $\mathbf{W}$ is a $p\times\ (c-1)$ matrix. It should be noted that the rank of the between-class scatter matrix is at most $c-1$ because it is the summation of $c$ rank one matrices and only $c-1$ of them are independent, and hence no more than $c-1$ of the eigenvalues will be non-zero.

### 2.2. *Expectation–Maximization (EM) algorithm*

The EM algorithm is a way of finding maximum likelihood estimates of parameters in finite mixture distributions; [6–8] via an iterative or a recursive technique. Given the objective of this paper, only the recursive EM algorithm will be discussed here.

With a finite mixture, a probability density function is modeled as the summation of weighted component densities,

$$\hat{f}(x;\, \pi,\, \Phi) = \sum_{i=1}^{c} \pi^{(i)}\, g\,(\mathrm{x};\, \Phi^{(i)}), \tag{14}$$

where $\Phi^{(i)}$ denotes the density parameters and $\pi = (\pi^{(1)}, \ldots, \pi^{(c)})$ are the weights such that $0 < \pi^{(i)} < 1$ and $\sum \pi^{(i)} = 1$. Usually, the component densities are assumed to be multivariate normal, so

$$g\ (\mathbf{x};\, \Phi^{(i)}) = \varphi\,(\mathbf{x}; \Sigma^{(i)}, \boldsymbol{\mu}^{(i)}), \tag{15}$$

where $\Sigma^{(i)}$ denotes a $p\times\ p$ covariance matrix and $\boldsymbol{\mu}^{(i)}$ denotes a $p$-dimensional mean.

The EM algorithm is basically a two-step procedure. First, we determine the probability that a data point belongs to the $i$th component by estimating the *a posteriori* probability given by

$$\hat{\tau}^{(i)}_{n+1} = \frac{\hat{\boldsymbol{\pi}}^{(i)}_n \hat{\varphi}\ (\mathbf{x}_{n+1};\, \hat{\Sigma}^{(i)}_n,\, \hat{\boldsymbol{\mu}}^{(i)}_n}{\sum_{k=1}^{c} \hat{\boldsymbol{\pi}}^{(k)}_n\ \hat{\varphi}\ \ (\mathbf{x}_{n+1};\ \hat{\Sigma}^{(k)}_n, \hat{\boldsymbol{\mu}}^{(k)}_n)}, \tag{16}$$

where $\hat{\Sigma}^{(j)}_n$ and $\hat{\boldsymbol{\mu}}^{(i)}_n$ are estimated from the previous $n$ points. This is the expectation step and is used to update the parameter estimates. The covariance matrices are updated using

$$\hat{\Sigma}^{(i)}_{n+1}, = \hat{\Sigma}^{(i)}_n + \ + \frac{\hat{\boldsymbol{\tau}}^{(i)}_{n+1}}{n\hat{\boldsymbol{\pi}}^{(i)}_n}$$

$$[(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}^{(i)}_n)\, (\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}^{(i)}_n)^{\mathrm{T}} - \hat{\Sigma}^{(i)}_n]. \tag{17}$$

The new means for each component are given by

$$\hat{\boldsymbol{\mu}}^{(i)}_{n+1} = \hat{\boldsymbol{\mu}}^{(i)}_n + \frac{\hat{\boldsymbol{\tau}}^{(i)}_{n+1}}{n\hat{\boldsymbol{\pi}}^{(i)}_n}\,(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}^{(i)}_n), \tag{18}$$

and the *a priori* probabilities or weights are then updated using

$$\hat{\boldsymbol{\pi}}^{(i)}_{n+1} = \hat{\boldsymbol{\pi}}^{(i)}_n + \frac{1}{n}\,(\hat{\boldsymbol{\tau}}^{(i)}_{n+1} - \hat{\boldsymbol{\pi}}^{(i)}_n). \tag{19}$$

These recursive update formulae comprise the maximization step, since these maximize the likelihood equations.

## 3. TWO-CLASS RECURSIVE FLD

To recursively update the FLD in the two-class case, we need update equations for the mean vectors for each class, $\mathbf{m}^{(1)}$, $\mathbf{m}^{(2)}$and the within-class scatter, matrix $\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2$. In this section, recursions for these will be derived. We consider two situations: (1) we know which class the new data point belongs to and (2) we know or can estimate the probability that the new data point belongs to each class.

### 3.1. *Perfect class knowledge*

If we know which class the new point belongs to, then we can use that information to update the FLD. This situation might occur when we have data that is labeled with the true class membership or when the new data points will be classified with the group that has the highest *a posteriori* % probability. In the following discussion, we will assume that the new data point belongs to the $j$th class.

Using the well-known[2] recursive update equations for the mean, the sample means for each class are updated using

$$\mathbf{m}^{(i)}_{n+1} = \begin{cases} \mathbf{m}^{(i)}_n, & i \neq j, \\ \mathbf{m}^{(1)}_n + \dfrac{1}{n_i + 1}\,(\mathbf{x}_{n+1} - \mathbf{m}^{(i)}_n), & i = j. \end{cases} \tag{20}$$

The within-class scatter matrix, after the addition of the next data point, is given by

$$\mathbf{S}_{W_{n+1}} = \mathbf{S}_{i_{n+1}} + \mathbf{S}_{j_{n+1}}. \tag{21}$$

However, the recursion for the within-class scatter matrix for the $j$th class can be written as

$$\mathbf{S}_{j_{n+1}} = \mathbf{S}_{j_n} + \frac{n_j}{n_j + 1}\,(\mathbf{x}_{n+1} - \mathbf{m}^{(j)}_n)\,(\mathbf{x}_{n+1} - \mathbf{m}^{(j)}_n)^{\mathrm{T}}. \tag{22}$$

Substituting this in equation (21) and using the fact that $\mathbf{S}_{i_{n+1}} = \mathbf{S}_{i_n}$ we get

$$\mathbf{S}_{W_{n+1}} = \mathbf{S}_{i_n} + \frac{n_j}{n_j + 1}\,(\mathbf{x}_{n+1} - \mathbf{m}^{(j)}_n)\,(\mathbf{x}_{n+1} - \mathbf{m}^{(j)}_n)^{\mathrm{T}} + \mathrm{S}_{j_n}. \tag{23}$$

Then the within-class scatter matrix is given by

$$\mathbf{S}_{W_{n+1}} = \mathbf{S}_{W_n} + \frac{n_j}{n_j+1}(\mathbf{x}_{n+1} - \mathbf{m}_n^{(j)})(\mathbf{x}_{n+1} - \mathbf{m}_n^{(j)})^{\mathrm{T}}. \tag{24}$$

It would be much more useful to have the inverse of equation (24), since that is used to find the projection vector $\mathbf{w}$. Using the matrix inversion lemma[2] we have

$$\mathbf{S}_{W_{n+1}}^{-1} = (\mathbf{S}_{W_n} + \frac{n_j}{n_j+1}(\mathbf{x}_{n+1} - \mathbf{m}_n^{(j)})(\mathbf{x}_{n+1} - \mathbf{m}_n^{(j)})^{\mathrm{T}})^{-1}$$
$$= \mathbf{S}_{W_n}^{-1} - \frac{\mathbf{S}_{W_n}^{-1}(\mathbf{x}_{n+1} - \mathbf{m}_n^{(j)})(\mathbf{x}_{n+1} - \mathbf{m}_n^{(j)})^{\mathrm{T}})\mathbf{S}_{W_n}^{-1}}{n_j + 1/n_j + (\mathbf{x}_{n+1} - \mathbf{m}_n^{(j)})^{\mathrm{T}}\mathbf{S}_{W_n}^{-1}(\mathbf{x}_{n+1} - \mathbf{m}_n^{(j)})}. \tag{25}$$

The following summarizes the procedure for recursively updating the FLD when there are two-classes and perfect class-membership knowledge: update the inverse of the within-class scatter matrix using equation (25), update the means using equation (20), increment the number of data points, and calculate the new FLD from equation (6).

### 3.2. Partial class knowledge

Suppose we do not know which class the new observation belongs to, but we do know or can estimate the *a posteriori* probabilities. Then we can use equations corresponding to those from the EM algorithm to update the FLD based on our partial knowledge. In essence, we are assuming that the new data point is a partial member of each class based on its *a posteriori* % probability.

Before the new class means can be updated, we need to determine the *a posteriori* probabilities using equation (16). The update equation for the within-class scatter matrix is somewhat more complicated, because it is not the same as the covariance matrix used in the EM algorithm. When we have partial class knowledge, points are given partial membership in each class according to their *a posteriori* probability. We can estimate the number of points grouped with each class as $\tilde{n}^{(i)} = n\hat{\pi}_n^{(i)}$ and define the scatter matrix for the $i$th class as

$$\hat{\mathbf{S}}_{i_{n+1}} = \tilde{n}^{(i)}\,\hat{\boldsymbol{\Sigma}}_{n+1}^{(i)}, \tag{26}$$

where the "hat" notation is used to indicate that this is an estimate based on our partial knowledge. Using equations (17) and (26), we can write the update equation for the scatter matrix for the $i$th class as

$$\frac{\hat{\mathbf{S}}_{i_{n+1}}}{\tilde{n}^{(i)}} = \frac{\hat{\mathbf{S}}_{i_n}}{\tilde{n}^{(i)} - 1} + \frac{\hat{\tau}_{n+1}^{(i)}}{n\hat{\pi}_n^{(i)}} \times \left[(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})^{\mathrm{T}}) - \frac{\hat{\mathbf{S}}_{i_n}}{\tilde{n}^{(i)} - 1}\right], \tag{27}$$

where the estimate of the mean is now denoted by $\hat{\mathbf{m}}$ for consistency with the FLD notation. Multiplying both sides by $\tilde{n}^{(i)}$ and simplifying yields

$$\hat{\mathbf{S}}_{i_{n+1}} = \frac{\tilde{n}^{(i)} - \hat{\tau}_{n+1}^{(i)}}{\tilde{n}^{(i)} - 1}\hat{\mathbf{S}}_{i_n} + \hat{\tau}_{n+1}^{(i)}(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(j)})(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(j)})^{\mathrm{T}}. \tag{28}$$

The within-class scatter matrix is given by the sum of these updated matrices yielding

$$\hat{\mathbf{S}}_{W_{n+1}} = \sum_{i=1}^{2}\left\{\frac{\tilde{n}^{(i)} - \hat{\tau}_{n+1}^{(i)}}{\tilde{n}^{(i)} - 1}\hat{\mathbf{S}}_{i_n} + \hat{\tau}_{n+1}^{(i)}(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})^{\mathrm{T}}\right\}. \tag{29}$$

Due to the form of equation (29), it is not possible to use the recursive update formula for the inverse. One solution to this is to update the within-class scatter matrix using equation (29) and then to invert the matrix. There is still a savings in storage space and memory with this method, because we do not have to retain every data point to get current estimates of the scatter matrix. The procedure for updating the FLD in the two-class, partial knowledge case is to first find the probability that it belongs to each group, then update the within-class scatter matrix, the means, the mixing coefficients, and the number of data points.

If we are working with classes that we have modeled using a multivariate normal distribution, then we also need to update the inverse covariance matrix and the determinant of the covariance matrix for each class. These are used in the multivariate normal densities, which is the assumed functional form for the class-conditional probability densities $g(\mathbf{x}; \boldsymbol{\phi})$. Having recursive equations for these will increase computational efficiency and are given in the appendix. The procedure for updating the two-class FLD when we have partial knowledge is to: determine the class-conditional probability, find the estimate of the within-class scatter matrix, update the *a priori* probabilities and the mean, invert the within-class scatter matrix, and calculate the projection.

## 4. MULTI-CLASS RECURSIVE FLD

The two-class recursive FLD can be extended to the $c$-class case. Recall that the projection is from a $p$-dimensional space to a $(c-1)$-dimensional space, with $c \leqslant p$. In this case, we need recursive update equations for the following values:

1. The mean vectors for each class:

$$\mathbf{m}^{(i)} = \frac{1}{n_i}\sum_{\mathrm{x}\in X_i}\mathrm{x}; \qquad i = 1, \ldots, c.$$

2. The total mean vector:

$$\mathbf{m} = \frac{1}{n}\sum_{x}\mathbf{x} = \frac{1}{n}\sum_{i=1}^{c} n_i\mathbf{m}^{(i)}.$$

3. The within-class scatter matrix:

$$\mathbf{S}_W = \sum_{i=1}^{c}\mathbf{S}_i.$$

4. The between-class scatter matrix:

$$\mathbf{S}_B = \sum_{i=1}^{c} n_i\,(\mathbf{m}^{(i)} - \mathbf{m})\,(\mathbf{m}^{(i)} - \mathbf{m})^{\mathrm{T}}.$$

As before, we consider the two situations where we have perfect or partial class knowledge.

### 4.1. Perfect class knowledge

Recall that in the perfect class knowledge case, we know for certain where to classify the new data point. As before, we will assume that the new data point belongs to the $j$th class. The class means and the within-class scatter matrix are direct extensions of the two-class FLD. Formulae for the total mean and the between-class scatter matrix have to be derived.

The class means after the addition of a new data point are given by equation (20). The update formula for the within-class scatter matrix generalizes easily from the two-class case and is given by equation (24). The total mean can be updated using the familiar recursion[2]

$$\mathbf{m}_{n+1} = \mathbf{m}_n + \frac{1}{n+1}(\mathbf{x}_{n+1} - \mathbf{m}_n). \tag{30}$$

The update for the between-class scatter matrix is somewhat more complicated and is easier to obtain from the definition for the total scatter matrix given by

$$\mathbf{S}_T = \sum_{x}(\mathbf{x} - \mathbf{m}\,)\,(\mathbf{x} - \mathbf{m})^{\mathrm{T}} = \mathbf{S}_W + \mathbf{S}_B. \tag{31}$$

We can write the update equation for the total scatter matrix as

$$\mathbf{S}_{T_{n+1}} = \mathbf{S}_{T_n} + \frac{n}{n+1}(\mathbf{x}_{n+1} - \mathbf{m}_n)\,(\mathbf{x}_{n+1} - \mathbf{m}_n)^{\mathrm{T}}. \tag{32}$$

Since the total scatter matrix is the sum of the within-class and the between-class scatter matrices, the above equation can also be written as

$$\mathbf{S}_{T_{n+1}} = \mathbf{S}_{W_{n+1}} + \mathbf{S}_{B_{n+1}}$$

$$= \mathbf{S}_{W_n} + \frac{n_j}{n_j+1}(x_{n+1} - \mathbf{m}_n^{(j)})(\mathbf{x}_{n+1} - m_n^{(j)}) \; + \mathrm{S}_{B_{n+1}}. \tag{33}$$

Equating the right-hand sides of equations (32) and (33) and re-arranging terms yields the update equation for the between-class scatter matrix

$$\mathbf{S}_{B_{n+1}} = \mathbf{S}_{B_n} + \frac{n}{n+1}(\mathbf{x}_{n+1} - \mathbf{m}_n)\,(\mathbf{x}_{n+1} - m_n)^{\mathrm{T}}$$

$$- \frac{n_j}{n_j+1}(\mathbf{x}_{n+1} - \mathbf{m}_n^{(1)})\,(\mathbf{x}_{n+1} - \mathbf{m}_n^{(j)})^{\mathrm{T}}. \tag{34}$$

To summarize, first update the scatter matrices, then update the means, and solve for the transformation matrix. Recall that the FLD transformation matrix is given by the eigenvectors corresponding to the non-zero eigenvalues of the generalized eigenvalue problem involving the between-class and within-class scatter matrices. Alternatively, one could use the eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$, in which case, the recursion for the inverse given in equation (25) is appropriate.

### 4.2. Partial class knowledge

As in the two-class case, we can estimate the *a posteriori* probabilities and use the equations corresponding to those from the EM algorithm. The recursions for the *a posteriori* probabilities, the weights and the class means are exactly the same as the two-class partial knowledge situation.

The update equation for the within-class scatter matrix for the two-class, partial knowledge case can easily be extended to $c$ classes as

$$\hat{\mathbf{S}}_{W_{n+1}} = \sum_{i=1}^{c}\left\{\frac{\tilde{n}^{(i)} - \hat{\tau}_{n+1}^{(i)}}{\tilde{n}^{(i)} - 1}\,\hat{\mathbf{S}}_{i_n} + \hat{\tau}_{n+1}^{(i)}\right.$$

$$\left.\times(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})\,(\mathrm{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})^{\mathrm{T}}\right\}. \tag{35}$$

The between-class scatter matrix can be calculated using the definition, since we now have most of the necessary elements. The only thing we do not know for certain is the number of points belonging to each class, because we are partially grouping new data points with each class. As before, we use the weights to indicate the proportion of points that are classified with each group. Thus, we can write the new between-class scatter matrix as

$$\hat{\mathbf{S}}_{B_{n+1}} = \sum_{i=1}^{c}\left\{\hat{\pi}_{n+1}^{(i)}\,(n+1)\,(\hat{\mathbf{m}}_{n+1}^{(i)} - \hat{\mathbf{m}}_{n+1})\right.$$

$$\left.\times(\hat{\mathbf{m}}_{n+1}^{(i)} - \hat{\mathbf{m}}_{n+1})^{\mathrm{T}}\right\}. \tag{36}$$

To find the transformation matrix W, one could solve the generalized eigenvalue problem [equation 10)] or use the eigenvectors of $\hat{\mathbf{S}}_W^{-1}\hat{\mathbf{S}}_B$.

## 5. SIMULATIONS

The recursions for the FLD when we know which class the new data point belongs to are exact, but that is not the case when we have partial knowledge of the class membership. Hence, Monte Carlo simulations

are needed to verify the usefulness of the recursive FLD in this case. The following simulations indicate that the recursive updates described in this report provide excellent estimates of the FLD when new data are added based on partial knowledge.

In these simulations, we were interested in the behavior of the recursive FLD as a function of the number of data points recursively added to the data set and the dimensionality of the data. We also looked at how good the estimates were as the class weights and class separation were varied. As expected, our procedure performs very well with data that has unequal weights and is well separated. We present only those cases where the procedure exhibited some poor estimates of the FLD projections. This will provide information about when this recursive method is useful and when it breaks down.

To test the behavior of the algorithm, we need some metric that indicates the fitness of the FLD projections after recursion. Since the orientation of the transformation is important in providing well-separated projected data, we use the cosine of the angle between the true FLD vector and the estimated FLD (calculated from the recursions) as our measure of closeness. A value of one for this metric means that the vectors are parallel and have the same orientation, and a value of zero indicates the vectors are perpendicular. Hence, the closer this value is to one, the better the estimate of the FLD. We have a similar metric in the multi-class case; we calculate the cosine of the angle between corresponding columns of the projection matrix.

In the simulations, 15 Monte Carlo trials were performed for each situation. The data generated for this study were multivariate normal with equal covariances, since that is required by the assumptions for the FLD. We also used equal class weights, because that would be the more difficult case. The procedure followed in the simulations is described here.

1. Generate multivariate data for each class.
2. Starting with a given percentage of the data in each class, determine the FLD projection.
3. Using the remaining data points, determine the estimate of the FLD recursively.
4. Use the entire data set to calculate the true FLD.
5. Determine the cosine of the angle between the vectors found in steps 3 and 4 to evaluate the fitness of the estimate.

The results for the two-class simulations are shown in Table 1. These results represent the behavior of the recursive FLD as a function of the number of points that were used in the recursion. Note that one-half of the specified points came from each class and were used to update the FLD. These simulations indicate that the recursive FLD works very well in the two-class case.

In the multi-class case, we determined the cosine of the angle between columns of the transformation

Table 1. Results from two-class recursive FLD

| Num of Points | Avg GOF($\sigma$), 2-D | Avg GOF%($\sigma$) 5-D, | Avg GOF($\sigma$), 7-D |
|---|---|---|---|
| 9980 | 0.8121 (0.2387) | 0.6786 (0.1287) | 0.6152 (0.1721) |
| 9480 | 0.9971 (0.0034) | 0.9926 (0.0036) | 0.9875 (0.0046) |
| 8980 | 0.9985 (0.0022) | 0.9954 (0.0031) | 0.9945 (0.0032) |
| 8480 | 0.9992 (0.0009) | 0.9978 (0.0015) | 0.9972 (0.0016) |
| 7980 | 0.9997 (0.0005) | 0.9989 (0.0009) | 0.9983 (0.0012) |
| 7480 | 0.9998 (0.0002) | 0.9994 (0.0003) | 0.9992 (0.0004) |
| 6980 | 0.9999 (0.0002) | 0.9997 (0.0002) | 0.9997 (0.0002) |
| 5480 | 0.9999 (0.0002) | 0.9998 (0.0002) | 0.9998 (0.0001) |
| 5980 | 0.9999 (0.0002) | 0.9999 (0.0001) | 0.9999 (0.0001) |
| 5480 | 0.9999 (0.0002) | 0.9999 (0.0001) | 0.9999 (0.0001) |

Table 2. Results from multi-class recursive FLD—smallest eigenvalue

| Num of Points | Avg GOF($\sigma$), 3-D | Avg GOF($\sigma$), 5-D | AvgGOF($\sigma$), 7-D |
|---|---|---|---|
| 9498 | 0.6701 (0.29) | 0.6002 (0.24) | 0.5204 (0.24) |
| 8499 | 0.5844 (0.37) | 0.5590 (0.26) | 0.5422 (0.21) |
| 7500 | 0.7577 (0.30) | 0.6550 (0.26) | 0.6399 (0.24) |
| 6498 | 0.7104 (0.28) | 0.7961 (0.20) | 0.7997 (0.15) |
| 5499 | 0.7409 (0.26) | 0.8032 (0.21) | 0.8489 (0.15) |
| 4500 | 0.8388 (0.26) | 0.7871 (0.22) | 0.8696 (0.08) |
| 3501 | 0.9266 (0.08) | 0.9025 (0.12) | 0.9345 (0.06) |
| 2499 | 0.7314 (0.29) | 0.8040 (0.26) | 0.9312 (0.08) |

Table 3. Results from multi-class recursive FLD—largest eigenvalue

| Num of Points | Avg GOF($\sigma$), 3-D | Avg GOF%($\sigma$), 5-D | Avg GOF($\sigma$), 7-D |
|---|---|---|---|
| 9498 | 0.9543 (0.045) | 0.9911 (0.0045) | 0.9847 (0.009) |
| 8499 | 0.9956 (0.0035) | 0.9945 (0.0029) | 0.9950 (0.0035) |
| 7500 | 0.9987 (0.0007) | 0.9981 (0.0009) | 0.9982 (0.0008) |
| 6498 | 0.9995 (0.0005) | 0.9995 (0.0004) | 0.9991 (0.0005) |
| 5499 | 0.9998 (0.0001) | 0.9997 (0.0002) | 0.9997 (0.0005) |
| 4500 | 0.9998 (0.0001) | 0.9998 (0.0002) | 0.9997 (0.0005) |
| 3501 | 0.9999 (0.0001) | 0.9999 (0.0001) | 0.9999 (0.0001) |
| 2499 | 0.9999 (0.0001) | 0.9999 (0.0001) | 0.9999 (0.0001) |

matrix **W**, where the first column is the eigenvector corresponding to the largest eigenvalue and the second column corresponds to the smallest eigenvalue. There were three classes with equal weights, and the number of points used in the recursion came equally from the three groups. Table 2 shows the results for the smallest eigenvalue, and Table 3 contains the results for the largest eigenvalue. These results show that the recursive FLD was successful in the multi-class case also.

## 6. SUMMARY

In this report, we develop recursive formulae for the FLD dimensionality reduction technique. Recursions are provided for the two-class and the multi-class versions of the FLD. The formulae are exact when we are basing our updates on perfect class knowledge. In the situation where we have partial knowledge of class membership for new data, we use variations of the recursive update equations based on the EM algorithm for obtaining maximum-likelihood estimates of finite mixture parameters. Simulations that examine the accuracy of the recursive FLD based on partial class knowledge indicate that the procedure works very well for two-class and multi-class applications.

*Acknowledgments*—The authors would like to thank the reviewers for their helpful comments and to acknowledge the support of the Naval Surface Warfare Center, Dahlgren Division, In-house Laboratory Independent Research Program.

## APPENDIX A

If we are working with classes that have been modeled using a multivariate normal distribution, then we also need to update the inverse covariance matrix and the determinant of the covariance matrix. These are used in evaluating the *a posteriori* probabilities. Having a recursive equation will save computations as well as storage space.

Recall the following equation for the recursive update of the covariance matrix:

$$\hat{\Sigma}_{i_{n+1}} = \hat{\Sigma}_{i_n} + \frac{\hat{\tau}_{n+1}^{(i)}}{n\hat{\pi}_n^{(i)}} [(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})^{\mathrm{T}} - \hat{\Sigma}_{i_n}]. \quad \text{(A.1)}$$

Rearranging the above equation and taking the inverse of both sides, we have

$$(\hat{\Sigma}_{i_{n+1}})^{-1} = \left( \frac{n\tilde{\pi}_n^{(i)} - \hat{\tau}_{n+1}^{(i)}}{n\hat{\pi}_n^{(i)}} \hat{\Sigma}_{i_n} + \frac{\hat{\tau}_{n+1}^{(i)}}{n\hat{\pi}_n^{(i)}} (\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})^{\mathrm{T}} \right)^{-1}. \quad \text{(A.2)}$$

To make things simpler, we define the following notation:

$$a = \frac{n\hat{\pi}_n^{(i)} - \hat{\tau}_{n+1}^{(i)}}{n\hat{\pi}_n^{(i)}}, \quad b = -\frac{\hat{\tau}_{n+1}^{(i)}}{n\hat{\pi}_n^{(i)}}, \quad = \tilde{\mathbf{x}}_{n+1}(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)}). \quad \text{(A.3)}$$

We can now write the update equation for the inverse as

$$(\hat{\Sigma}_{i_{n+1}})^{-1} = (a\hat{\Sigma}_{i_n} - b\tilde{\mathbf{x}}_{n+1}\tilde{\mathbf{x}}_{n+1}^{\mathrm{T}})^{-1} = \tilde{\Sigma}_{i_n}^{-1} + \frac{\tilde{\Sigma}_{i_n}^{-1}\tilde{\mathbf{x}}_{n+1}\tilde{\mathbf{x}}_{n+1}^{\mathrm{T}}\tilde{\Sigma}_{i_n}^{-1}}{\frac{1}{b} - \tilde{\mathbf{x}}_{n+1}^{\mathrm{T}}\tilde{\Sigma}_{i_n}^{-1}\tilde{\mathbf{x}}_{n+1}}, \quad \text{(A.4)}$$

where

$$\tilde{\Sigma}_{i_n}^{-1} = \frac{1}{a}\hat{\Sigma}_{i_n}^{-1}. \quad \text{(A.5)}$$

The update equation for the determinant of the covariance matrix is derived similarly.[2] We start with the desired update

$$|\hat{\Sigma}_{i_{n+1}}| = \left| \frac{n\hat{\pi}_n^{(i)} - \hat{\tau}_{n+1}^{(i)}}{n\hat{\pi}_n^{(i)}} \hat{\Sigma}_{i_n} + \frac{\hat{\tau}_{n+1}^{(i)}}{n\hat{\pi}_n^{(i)}} (\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})(\mathbf{x}_{n+1} - \hat{\mathbf{m}}_n^{(i)})^{\mathrm{T}} \right|. \quad \text{(A.6)}$$

Using the same notation as before, this can be written as

$$|\hat{\Sigma}_{i_{n+1}}| = |a\hat{\Sigma}_{i_n} - b\tilde{\mathbf{x}}_{n+1}\tilde{\mathbf{x}}_{n+1}^{\mathrm{T}}| = |a\hat{\Sigma}_{i_n}| \, |(1/\mathrm{b}) - \tilde{\mathbf{x}}_{n+1}^{\mathrm{T}}(a\hat{\Sigma}_{i_n})^{-1}\tilde{\mathbf{x}}_{n+1}| / |1/\mathrm{b}|. \quad \text{(A.7)}$$

We can simplify this further using the fact that the determinant of a scalar is the scalar itself and the determinant depends linearly on each row or column

of the matrix. Thus, we have

$$|\hat{\Sigma}_{i_{n+1}}| = a^p \, (1 - b\tilde{\mathrm{x}}_{n+1}^{\mathrm{T}} \, \hat{\Sigma}_{i_n}^{-1} \, \tilde{\mathrm{x}}_{n+1}) \, |\hat{\Sigma}_{i_n}|, \quad \text{(A.8)}$$

where $p$ is the dimensionality of the data.

## REFERENCES


1. D. W. Scott, *Multivariate Density Estimation*, Wiley, New York (1992).
2. R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, Wiley, New York (1973).
3. J. E. Jackson, *A User's Guide to Principal Components*, Wiley, New York (1991).
4. J. K. Friedman, and J. W. Tukey, A projection pursuit algorithm for exploratory data analysis, *IEEE Trans. Comput.* **C-23,** 881–890 (1974).
5. R. A. Redner and H. F. Walker, Mixture Densities, Maximum Likelihood and the EM Algorithm, *SIAM Rev.* **26**, 195–239 (1984).
6. B. S. Everitt and D. J. Hand, *Finite Mixture Distributions*, Chapman & Hall, New York (1981).
7. G. J. McLachlan and K. E. Basford, *Mixture Models: Inference and Applications to Clustering*, Marcel Dekker, New York (1988).
8. D. M. Titterington, A. F. Smith and V. E. Makov, *Statistical Analysis of Finite Mixture Distributions*, Wiley, New York (1985).



**About the Author**—WENDY L. POSTON received a Bachelor of Science degree with high honors from Cameron University in 1989 with majors in physics and mathematics. She received a Master's degree in aerospace engineering in 1991 from George Washington University and a doctorate in Computational Statistics at George Mason University in 1995. Dr Poston is currently employed at the Naval Surface Warfare Center in Dahlgren, Virginia. Since 1991, she has been working in statistical pattern recognition, density estimation, and signal processing. Dr Poston has written over 50 papers and has been granted four patents. She also holds a position as an adjunct faculty member at George Mason University.

**About the Author**—DAVID J. MARCHETTE received a B.A. in 1980, an M.A. in mathematics in 1982, from the University of California at San Diego, and a Ph.D. in Computational Sciences and Informatics in 1996 from George Mason University. From 1985 to 1994, he worked at the Naval Ocean Systems Center doing research on pattern recognition and computational statistics. In 1994, he moved to the Naval Surface Warfare Center, where he conducts research in computational statistics and pattern recognition, primarily applied to image processing and automatic target recognition.