PAPER *Special Section on Papers Selected from ITC-CSCC 2000*

Fast Algorithm for Online Linear Discriminant Analysis

Kazuyuki HIRAOKA[†a)], **Masashi HAMAHIRA**[†], **Ken-ichi HIDAI**[†], **Hiroshi MIZOGUCHI**[†], ***Nonmembers***, **Taketoshi MISHIMA**[†], ***and*** **Shuji YOSHIZAWA**[†], ***Regular Members***

SUMMARY Linear discriminant analysis (LDA) is a basic tool of pattern recognition, and it is used in extensive fields, e.g. face identification. However, LDA is poor at adaptability since it is a batch type algorithm. To overcome this, new algorithms of online LDA are proposed in the present paper. In face identification task, it is experimentally shown that the new algorithms are about two times faster than the previously proposed algorithm in terms of the number of required examples, while the previous algorithm attains better final performance than the new algorithms after sufficient steps of learning. The meaning of new algorithms are also discussed theoretically, and they are suggested to be corresponding to combination of PCA and Mahalanobis distance.

key words: *linear discriminant analysis, online learning, face identification, matrix dynamics*

1. Introduction

Linear discriminant analysis (LDA) is a basic tool of pattern recognition, and it is used in extensive fields, e.g. face identification [8], [9]. It is also pointed out that dimension reduction via LDA is useful as a preprocessing for other methods such as support vector machines [10]. However, LDA is poor at adaptability since it is a batch type algorithm. Namely, LDA is designed in the following manner: (1) all sample patterns are given at once, (2) the discriminant matrix A is calculated for the sample patterns, and then (3) identification is performed by use of A. Owing to this design, we have to re-calculate A every time when we add new data to update the identification system. This calculation is heavy for high dimensional data such as face images.

When situation changes gradually or suddenly, one time learning is not sufficient and additional learning is indispensable for adaptability. Thus identification systems must have the ability to learn new data and update itself with small calculations. Such algorithms which have this ability is called *online* learning algorithms. As we have mentioned above, conventional LDA is not online learning.

To overcome this disadvantage of LDA, the authors have been proposed an online LDA algorithm [15]–[17]. In contrast to the conventional LDA, updating the identification system according to new additional data can be executed with low computational cost by online LDA. Hence online LDA has the ability of adaptation to changes of environment. Our online LDA also has an advantage that huge matrices never appear in its calculation.

Though iterative algorithms have been proposed[*] for neural network based LDA [5], [6], they are not sufficiently "online." Those algorithms keep $n^2 \times n^2$ matrices when they are applied to image recognition tasks with image size $n \times n$. Then they require $O(n^4)$ time for one step updating and $O(n^4)$ memory. These computational costs are still too large to update the system on the fly.

In the present paper, two types of new algorithms for online LDA are proposed. The following points are shown experimentally:

- The new algorithms are about two times faster than the previously proposed algorithm. Namely, the new algorithms attain the same level of performance by half steps of learning compared with the original one. This advantage is mainly owed to the fact that new algorithms are robust for choice of the learning coefficient η.
- The previously proposed algorithm attains better final performance after sufficient steps of learning.

The meaning of new algorithms are also discussed. By theory and numerical experiments, it is suggested that the new algorithms correspond to combination of principal component analysis (PCA) and Mahalanobis distance.

2. Linear Discriminant Analysis

In pattern recognition task, sample pattern vectors $\boldsymbol{x}(1), \cdots, \boldsymbol{x}(t) \in R^N$ and their classes $c(1), \cdots, c(t) \in \{1, 2, \cdots, M\}$ are given first. Then, a new pattern $\boldsymbol{x}$ with unknown class is presented and we want to estimate the class of this $\boldsymbol{x}$.

Manuscript received October 2, 2000.
Manuscript revised December 23, 2000.

[†]The authors are with the Department of Information and Computer Sciences, Saitama University, Urawa-shi, 338-8570 Japan.
a) E-mail: hira@me.ics.saitama-u.ac.jp

[*]The original algorithm in [5] is a batch learning. However, we can easily modify it to an iterative learning by replacing the calculation of the standard deviation in [5] with an iterative one.

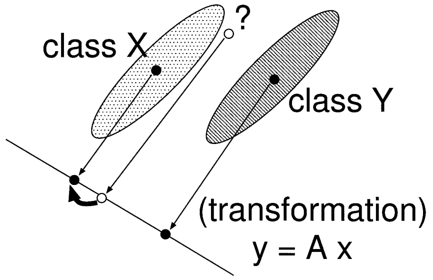


Fig. 1 Pattern recognition via LDA. The presented pattern vector $\boldsymbol{x}$ is converted to a feature vector $\boldsymbol{y}$ by use of a certain "discriminant matrix" A. Then it is compared with class mean vectors $\bar{\boldsymbol{y}}^c$.

The procedures of pattern recognition via LDA is as follows (Fig. 1):

1. Generate a certain "discriminant matrix" A.
2. Transform the presented pattern vector $\boldsymbol{x}$ to a feature vector: $\boldsymbol{y} = A\boldsymbol{x}$.
3. Compare it with mean vector $\bar{\boldsymbol{y}}^c = A\bar{\boldsymbol{x}}^c$ for each class c. Here $\bar{\boldsymbol{x}}^c$ is the mean pattern vector of class c.
4. Answer the "nearest" class from the point of view of $\|\boldsymbol{y}-\bar{\boldsymbol{y}}^c\|^2$. Here $\|\boldsymbol{z}\| \equiv \sqrt{\boldsymbol{z}^T\boldsymbol{z}}$ denotes Euclidean norm.

Performance of recognition via LDA depends crucially on the discriminant matrix A. The most popular criterion to determine A is Fisher criterion [11]: select such A which maximizes $F = \mathrm{Tr}\left[(A^TWA)^{-1}(A^TBA)\right]$, where B and W are the variance matrices "between classes" and "within classes." The precise definitions of B and W will be given soon later.

Concrete procedures to calculate A based on Fisher criterion is as follows:

1. Calculate the mean pattern $\bar{\boldsymbol{x}}$ of the whole data and the mean pattern $\bar{\boldsymbol{x}}^c$ of each class c:

$$\bar{\boldsymbol{x}} = \frac{1}{t}\sum_{\tau=1}^{t}\boldsymbol{x}(\tau), \tag{1}$$

$$\bar{\boldsymbol{x}}^c = \frac{1}{\sum_{\tau=1}^{t}\delta(c,c(\tau))}\sum_{\tau=1}^{t}\delta(c,c(\tau))\boldsymbol{x}(\tau) \qquad (c = 1,\cdots,M), \tag{2}$$

where $\delta(c,c')$ is 1 if $c = c'$ and 0 if $c \neq c'$.

2. Calculate the variance matrices B of the mean patterns and W of the errors of patterns from corresponding mean patterns:

$$B = \frac{1}{M}\sum_{c=1}^{M}\left(\bar{\boldsymbol{x}}^c - \bar{\boldsymbol{x}}\right)\left(\bar{\boldsymbol{x}}^c - \bar{\boldsymbol{x}}\right)^T, \tag{3}$$

$$W = \frac{1}{t}\sum_{\tau=1}^{t}\left(\boldsymbol{x}(\tau) - \bar{\boldsymbol{x}}^{c(\tau)}\right)\left(\boldsymbol{x}(\tau) - \bar{\boldsymbol{x}}^{c(\tau)}\right)^T. \tag{4}$$

3. Solve the generalized eigenvalue problem

$$B\boldsymbol{a} = \lambda W\boldsymbol{a}, \tag{5}$$

where λ is a real number and $\boldsymbol{a}$ is an N-dimensional vector. Normalize the solutions $\boldsymbol{a}_1,\cdots,\boldsymbol{a}_N$ so that $\boldsymbol{a}_i^TW\boldsymbol{a}_j = \delta(i,j)$ holds for all i and j. Sort the solutions so that $\lambda_1 \geq \cdots \geq \lambda_N$ holds for the corresponding λs.

4. Set the first L solutions $\boldsymbol{a}_1,\cdots,\boldsymbol{a}_L$ of (5) as the column vectors of A: $A = (\boldsymbol{a}_1,\cdots,\boldsymbol{a}_L)$.

This procedure is a batch learning type algorithm. Because of its design, we have to re-calculate A every time when we add new data to update the identification system. This calculation is heavy for high dimensional data such as face images. Online LDA is thus desired.

3. Matrix Dynamics for Online LDA—Theoretical Foundation of New Algorithms

3.1 Summary of This Section

By the procedure in the previous section, we obtain a matrix $A = (\boldsymbol{a}_1,\cdots,\boldsymbol{a}_L)$ which satisfies

$$B\boldsymbol{a}_i = \lambda_i W\boldsymbol{a}_i \qquad (i = 1,\cdots,L), \tag{6}$$

$$\boldsymbol{a}_i^TW\boldsymbol{a}_j = \delta_{ij} \qquad (i,j = 1,\cdots,L). \tag{7}$$

Putting $\Gamma = \mathrm{diag}(\lambda_1,\cdots,\lambda_L)$, we can write these equations in matrix form as

$$\begin{aligned} &BA = WA\Gamma \qquad (\Gamma\text{: square matrix}),\\ &A^TWA = I \qquad (I\text{: identity matrix}). \end{aligned} \tag{8}$$

As we have described in the previous section, this A maximizes Fisher criterion F. See textbooks on LDA (e.g. [11]) for details. Online learning algorithm to find such A is proposed in [15]–[17].

In the present paper, two types of novel algorithms for online LDA are proposed. Though property of the new algorithms is not completely investigated, it is suggested that we obtain a matrix A which satisfy

$$\begin{aligned} &BA = A\Gamma \qquad (\Gamma\text{: square matrix}),\\ &A^TWA = I \qquad (I\text{: identity matrix}). \end{aligned} \tag{9}$$

Exact descriptions of this assertion will be shown through propositions 1 to 4 later. When A satisfies (9), this A corresponds not to Fisher criterion but to combination of principal component analysis (PCA) and Mahalanobis distance. Namely,

1. reduce the dimension of input pattern by PCA on B (PCA stage), and then,

2. measure the distance between the present pattern and the class means by Mahalanobis distance on W (Mahalanobis distance stage).

The reason is as follows: the first equation in (9) means that column vectors $\boldsymbol{a}_1, \cdots, \boldsymbol{a}_L$ are on a hyperplane which is spanned by L eigenvectors (i.e. principal components) of B, while the second equation in (9) means that $\mathrm{Var}[\boldsymbol{y} - \bar{\boldsymbol{y}}^c] = A^T \mathrm{Var}[\boldsymbol{x} - \bar{\boldsymbol{x}}^c] A = A^T W A = I$ and therefore Mahalanobis distance between $\boldsymbol{y}$ and $\bar{\boldsymbol{y}}^c$ is equal to Euclidean norm $\|\boldsymbol{y} - \bar{\boldsymbol{y}}^c\|$.

3.2 Strategy to Construct Online LDA

Our online LDA algorithms are constructed based on matrix dynamics and stochastic approximation method [7]. Namely, we follow the below strategy so as to obtain online LDA algorithms:

1. Consider a dynamics whose fixedpoints are desired solutions of the current problem.
2. Transform the continuous-time dynamics to its corresponding discrete-time dynamics.
3. Replace the sample mean in the dynamics with its instantaneous value.

First of all, novel matrix dynamics are proposed and their convergence properties are discussed in this section.

3.3 Novel Matrix Dynamics

Let B be a positive semi-definite symmetric $N \times N$ matrix, and W be a positive definite symmetric $N \times N$ matrix. This B corresponds to the variance matrix "between classes," while W corresponds to the variance matrix "within classes."

Let $A(t)$ be an $N \times L$ matrix and consider matrix dynamics

$$\text{[type I]} \quad \frac{dA(t)}{dt} = BA(t) - \frac{1}{2}BA(t)A(t)^T WA(t) - \frac{1}{2}A(t)A(t)^T WBA(t), \quad (10)$$

$$\text{[type II]} \quad \frac{dA(t)}{dt} = BA(t) - \frac{1}{2}A(t)A(t)^T BWA(t) - \frac{1}{2}A(t)A(t)^T WBA(t). \quad (11)$$

Sample flows of these dynamics are shown in Fig. 2.

These dynamics are slightly different from the dynamics for previous online LDA [15]–[17] (Fisher criterion)

$$\frac{dA(t)}{dt} = BA(t) - \frac{1}{2}BA(t)A(t)^T WA(t) - \frac{1}{2}WA(t)A(t)^T BA(t) \quad (12)$$

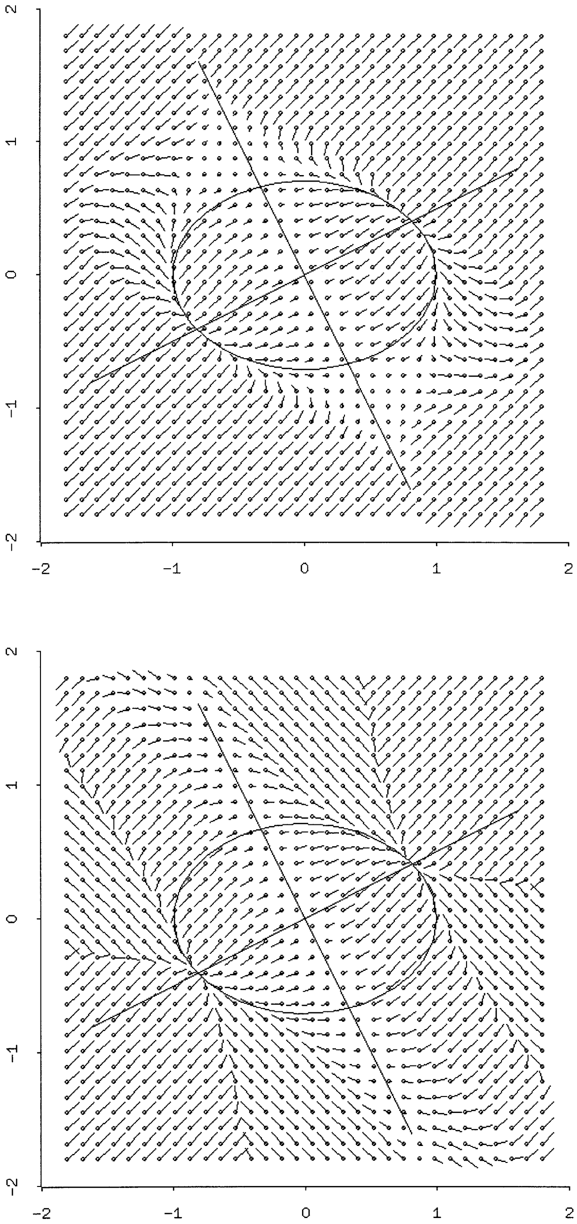


Fig. 2 Sample flows of the proposed dynamics (upper: type I, lower: type II). In these samples, A is 2×1 matrix, and it is plotted as 2-dimensional vector in the figures. $B = \begin{pmatrix} 4 & 2 \\ 2 & 1 \end{pmatrix}$, $W = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$. The ellipse in each figure shows the contour $A^T WA = 1$. The cross lines show the eigenspaces of B: directions "/" and "\" correspond to eigenvalues 5 and 0, respectively. It is observed that (i) the eigenvector A_* of B with a non-zero eigenvalue and $A_*^T WA_* = 1$ is a stable fixedpoint [desired solution], (ii) other stable fixed points [spurious solutions] or "divergence solution" exist, (iii) if an initial point $A(0)$ is not "too far" from the origin O, $A(t)$ converges to one of the desired solutions.

and Oja's dynamics for PCA

$$\frac{dA(t)}{dt} = BA(t) - A(t)A(t)^T BA(t). \quad (13)$$

3.4 Fixedpoints of Dynamics

We are interested in "fixedpoints" of the proposed dynamics since they decide asymptotic behavior of corresponding online LDA algorithms. Though A is not a *point* but a *matrix*, the term "fixedpoint" is used for a matrix A which satisfy $dA/dt = O$ through the present paper. The following propositions suggest characteristics of the fixedpoints. The proofs of the propositions are given in Appendix.

First propositions are sufficient conditions of fixedpoints.

Proposition 1: If an $N \times L$ matrix A satisfies (9) for an $L \times L$ matrix Γ, A is a fixedpoint of the type I dynamics. □

Proposition 2: If an $N \times L$ matrix A satisfies (9) for an $L \times L$ *symmetric* matrix Γ, A is a fixedpoint of the type II dynamics. □

On the other hand, necessary conditions of fixedpoints are as follows.

Proposition 3:

1. If an $N \times L$ matrix A is a fixedpoint of the type I dynamics and A^TWA does not have an eigenvalue 2, there exists an $L \times L$ matrix Γ such that $BA = A\Gamma$.
2. In addition, if rank$A = L$,

$$\Gamma(A^TWA - I) + (A^TWA - I)\Gamma = O. \qquad (14)$$

□

Proposition 4:

1. If an $N \times L$ matrix A is a fixedpoint of the type II dynamics, there exists an $L \times L$ *symmetric* matrix Γ such that $BA = A\Gamma$.
2. In addition, if rank$A = L$,

$$\Gamma(A^TWA - I) = O. \qquad (15)$$

3. In addition, if rank$(BA) = L$, $A^TWA = I$.

□

There exist gaps between necessary conditions and sufficient conditions above. Indeed, there are fixedpoints which do not satisfy (9). This problem is discussed next.

3.5 Desired Solutions and Nuisance Solutions

Let $\boldsymbol{p}_1, \cdots, \boldsymbol{p}_N$ be the unit eigenvectors of B and their corresponding eigenvalues be $\lambda_1 \geq \cdots \geq \lambda_N$. Let $P_L = (\boldsymbol{p}_1, \cdots, \boldsymbol{p}_L)$. If A is represented as $A = P_L\Theta$ with an arbitrary matrix Θ which satisfy $\Theta^T P_L^T WP\Theta = I$, equation (9) holds for this A. We call such $A = P_L\Theta$ as "desired" solutions. By propositions 1 and 2, desired solutions are fixedpoints of the proposed dynamics. Moreover, Fig. 2 suggests that they are stable fixedpoints. At the same time, Fig. 2 also shows that there can be other stable fixedpoints ("spurious" solutions) or "divergence" solutions.

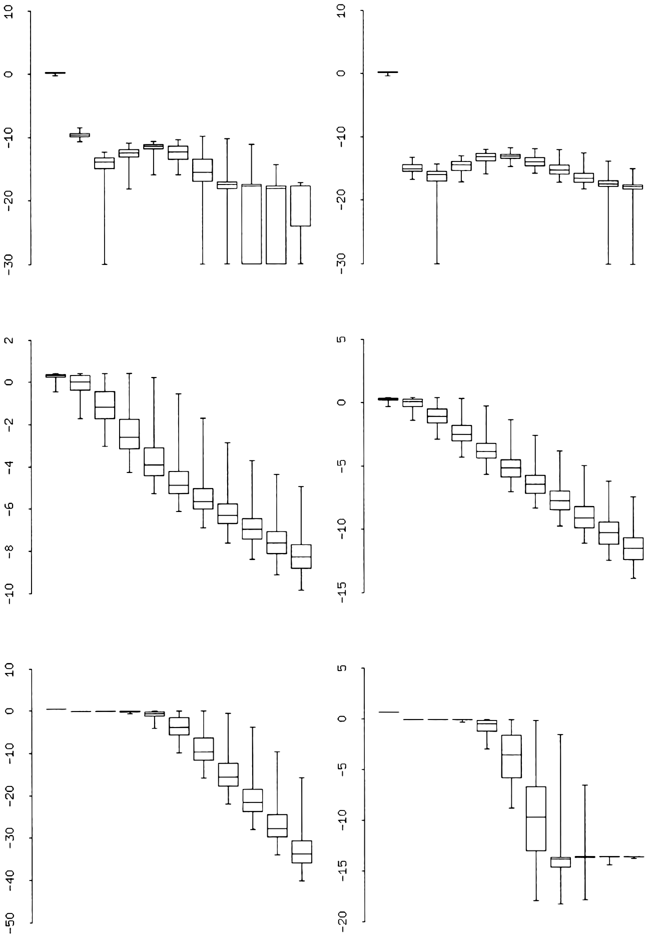


Fig. 3 Convergence of $A(t)$ to "desired" solutions by the proposed dynamics. In spite of existence of "spurious" solutions or "divergence" solutions, it is observed that $A(t)$ is not trapped by these nuisance solutions if the initial value $A(0)$ is near the origin O. Left: type I, right: type II. Upper: log("angle between $\boldsymbol{p}_1$ and $\Pi(t)$ [rad]"), middle: log("angle between $\boldsymbol{p}_2$ and $\Pi(t)$ [rad]"), lower: $\log(\mathrm{Tr}(A(t)^T WA(t)-I)^T(A(t)^TWA(t)-I))$, where $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ are the two major eigenvectors of B, and $\Pi(t) = \mathrm{span}\{$"column vectors of $A(t)$"$\}$. The horizontal axis of each plot is time $t = 0, 0.1, 0.2, \cdots, 1.0$. On the vertical axis, the five-number summaries (minimum, first quartile, median, third quartile, and maximum) for 100 trials are displayed.

In spite of these nuisance solutions, it is observed in numerical experiments that $A(t)$ is not trapped by nuisance solutions and $A(t)$ converges successfully to desired solutions if the initial value $A(0)$ is near the origin O.

Examples are shown in Fig. 3, where $B = \tilde{B}^T\tilde{B}$,

$$\tilde{B} = \begin{pmatrix} 1 & 2 & 1 & 2 & 1 & 2 & 1 & 2 & 1 & 2 \\ 3 & 2 & 1 & 3 & 2 & 1 & 3 & 2 & 1 & 3 \\ 3 & 1 & 4 & 1 & 5 & 3 & 1 & 4 & 1 & 5 \end{pmatrix},$$

$W = \mathrm{diag}(1, 2, \cdots, 10)$. The size of A is 10×2. In each trial, initial values of elements of A are generated randomly according to normal distribution of mean 0 and standard deviation 0.001. From this initial value $A(0)$, evolution of $A(t)$ is numerically calculated by simple Euler method with time step $\Delta t = 0.001$. The upper and middle plots show that there exists a matrix Γ which satisfy $BA \approx A\Gamma$, after sufficient time. Moreover, this A corresponds to the *major* eigenvectors of B. The lower plots show that $A(t)^T W A(t) \to I$.

From the above observation, we select $A(0) \approx O$ in Sect. 5.

4. Fast Online LDA Algorithm

In order to construct online LDA algorithm, we transform the dynamics type I and II, which are continuous-time dynamics, to their discrete-time versions:

[type I]

$$\begin{aligned} A(t+1) = & A(t) + \eta \Big[BA(t) - \frac{1}{2} BA(t)A(t)^T W A(t) \\ & - \frac{1}{2} A(t)A(t)^T W B A(t) \Big], \end{aligned} \tag{16}$$

[type II]

$$\begin{aligned} A(t+1) = & A(t) + \eta \Big[BA(t) - \frac{1}{2} A(t)A(t)^T B W A(t) \\ & - \frac{1}{2} A(t)A(t)^T W B A(t) \Big], \end{aligned} \tag{17}$$

where the learning coefficient η is a small positive number. In these discrete-time dynamics, B and W are $N \times N$ matrices and we do not like to keep their values explicitly because N can be very large. As for B, see the note on efficient calculation at the bottom of this section. As for W, we replace the "mean value" $W = \frac{1}{T} \sum_{t=1}^{T} (\boldsymbol{x}(t) - \bar{\boldsymbol{x}}^c(t))(\boldsymbol{x}(t) - \bar{\boldsymbol{x}}^c(t))^T$ with its "instantaneous value" $(\boldsymbol{x}(t) - \bar{\boldsymbol{x}}^c(t))(\boldsymbol{x}(t) - \bar{\boldsymbol{x}}^c(t))^T$. This replacement is justified by the theory of stochastic approximation [7]: $A(t)$ converges to the same point as for the original continuous-time dynamics when $\eta \to 0$. Then the following online LDA algorithms are obtained.

At every time step $t = 1, 2, 3, \cdots$, a new pair $(\boldsymbol{x}(t), c(t))$ is presented, where $\boldsymbol{x}(t)$ is an N-dimensional data vector, $c(t) \in \{1, \cdots, M\}$ is the class of $\boldsymbol{x}(t)$, and M is the number of classes. Based on this pair, auxiliary variables are updated as follows:

$$t^c(t) = t^c(t-1) + \delta(c, c(t)), \tag{18}$$

$$\bar{\boldsymbol{x}}(t) = \left(1 - \frac{1}{t}\right) \bar{\boldsymbol{x}}(t-1) + \frac{1}{t} \boldsymbol{x}(t), \tag{19}$$

$$\begin{aligned} & \bar{\boldsymbol{x}}^c(t) \\ & = \begin{cases} \left(1 - \frac{1}{t^c(t)}\right) \bar{\boldsymbol{x}}^c(t-1) + \frac{1}{t^c(t)} \boldsymbol{x}(t) & (c = c(t)), \\ \bar{\boldsymbol{x}}^c(t-1) & (c \neq c(t)), \end{cases} \end{aligned} \tag{20}$$

$$\boldsymbol{v}^c(t) = \bar{\boldsymbol{x}}^c(t) - \bar{\boldsymbol{x}}(t), \tag{21}$$

$$\boldsymbol{w}(t) = \boldsymbol{x}(t) - \bar{\boldsymbol{x}}^{c(t)}(t), \tag{22}$$

$$B(t) = \frac{1}{M} \sum_{c=1}^{M} \boldsymbol{v}^c(t) \boldsymbol{v}^c(t)^T, \tag{23}$$

where $c = 1, \cdots, M$, $\delta(c, c(t)) = 1\,(c = c(t))$, $0\,(c \neq c(t))$, and $\bar{\boldsymbol{x}}^{c(t)}(t)$ means $\bar{\boldsymbol{x}}^c(t)$ for $c = c(t)$. Then $N \times L$ discriminant matrix A is updated:

[type I]

$$\begin{aligned} A(t) = & A(t-1) + \eta \Big[B(t)A(t-1) \\ & - \frac{1}{2} B(t)A(t-1)\, A(t-1)^T \boldsymbol{w}(t)\boldsymbol{w}(t)^T A(t-1) \\ & - \frac{1}{2} A(t-1)\, A(t-1)^T \boldsymbol{w}(t)\boldsymbol{w}(t)^T B(t) A(t-1) \Big], \end{aligned} \tag{24}$$

[type II]

$$\begin{aligned} A(t) = & A(t-1) + \eta \Big[B(t)A(t-1) \\ & - \frac{1}{2} A(t-1)\, A(t-1)^T B(t) \boldsymbol{w}(t)\boldsymbol{w}(t)^T A(t-1) \\ & - \frac{1}{2} A(t-1)\, A(t-1)^T \boldsymbol{w}(t)\boldsymbol{w}(t)^T B(t) A(t-1) \Big], \end{aligned} \tag{25}$$

where the learning coefficient η is a small positive number.

These updating rules are slightly different from the previous algorithm

$$\begin{aligned} A(t) = & A(t-1) + \eta \Big[B(t)A(t-1) \\ & - \frac{1}{2} B(t)A(t-1)\, A(t-1)^T \boldsymbol{w}(t)\boldsymbol{w}(t)^T A(t-1) \\ & - \frac{1}{2} \boldsymbol{w}(t)\boldsymbol{w}(t)^T A(t-1)\, A(t-1)^T B(t) A(t-1) \Big] \end{aligned} \tag{26}$$

in [15]–[17].

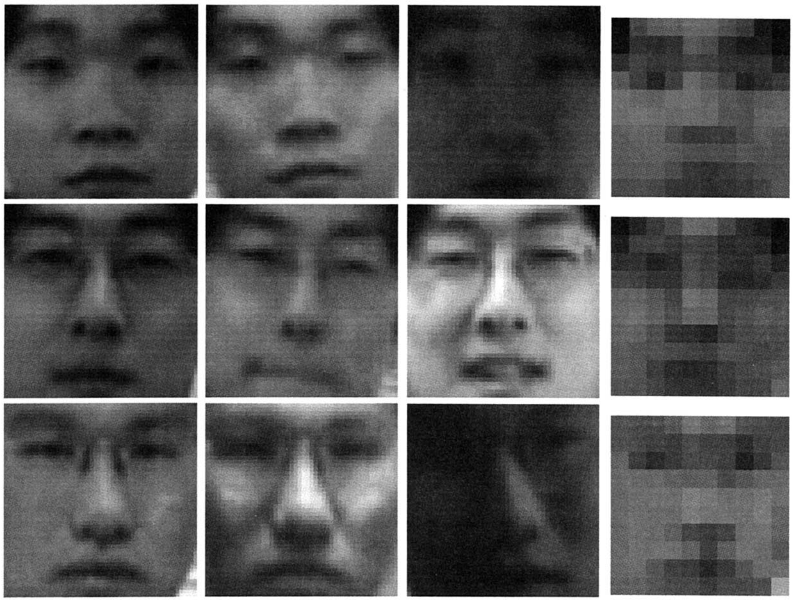


Fig. 4 Face images for simulation: 9 samples of original images and 3 samples of 10×10 reduced images. See Table 1 for details.

Table 1 Setting of the simulation.

task	identification of face images
data vector $\boldsymbol{x}(t)$	face images under various illumination conditions (front view, 256 level gray scale, normalized to $[-1, +1]$)
size of $\boldsymbol{x}(t)$ number of classes to be identified number of features	$N = 10 \times 10 = 100$ (pixels) $M = 3$ (persons) $L = 2$ (= number of columns in A)
initial values of elements in A regularization coefficient	random values from the uniform distribution on $[-0.001, +0.001]$ $\epsilon = 0.01$ (applied only to the original algorithm (28))
procedure of learning procedure of evaluation number of face images for learning number of face images for evaluation	Face images for learning is presented in a random order. The ratio of the correct identification is evaluated for face images which are different from the face images for learning. up to 500(images per person) × 3(persons) = 1500 100(images per person) × 3(persons) = 300

The number L of features is less than or equal to $\min(N, M-1)$. As for the initial values, $t^c(0) = 0$, $\bar{\boldsymbol{x}}(0)$ and $\bar{\boldsymbol{x}}^c(0)$ are arbitrary vectors, and $A(0)$ is an arbitrary matrix which satisfies $\mathrm{rank}A(0) = L$.

In (24), (25) and (26), the learning coefficient η affects the performance of algorithms. In order to obtain fast convergence of the discriminant matrix $A(t)$, we want to set η as larger as possible. However, if η is too large, $A(t)$ can diverge. In Sect. 5, it will be shown that the boundary of "acceptable" η in (24),(25) is larger than that in (26). Thus, we can obtain fast convergence by the algorithms (24),(25).

Note that the right hand sides of (24),(25) can be calculated efficiently in the following manner for the case $M \ll N$.

1. Instead of calculating B explicitly, calculate

$$BA = \frac{1}{M} \sum_{c=1}^{M} \boldsymbol{v}^c \left(\boldsymbol{v}^{cT} A \right). \tag{27}$$

2. Calculate $A^T \boldsymbol{w}$ and $A^T B \boldsymbol{w} = (BA)^T \boldsymbol{w}$.
3. Calculate $A(A^T \boldsymbol{w})$ and $A(A^T B \boldsymbol{w})$.
4. Calculate $BAA^T \boldsymbol{w}\boldsymbol{w}^T A = ((BA)(A^T \boldsymbol{w}))(A^T \boldsymbol{w})^T$ [type I] or $AA^T B \boldsymbol{w}\boldsymbol{w}^T A = (AA^T B \boldsymbol{w})(A^T \boldsymbol{w})^T$ [type II], and $AA^T \boldsymbol{w}\boldsymbol{w}^T BA = (AA^T \boldsymbol{w})(A^T B \boldsymbol{w})^T$.

5. Simulation

5.1 Comparison with Original Online LDA

In this section, performance of online LDA algorithms is tested for face identification task (Fig. 4). It is experimentally shown that the presented algorithms are about two times faster than the original one. Namely, the presented algorithms attain the same level of performance by half steps of learning compared with the original one. Type I and II show similar result.

The setting of the simulation is written in Table 1. In the simulation of the original algorithm, the updating rule

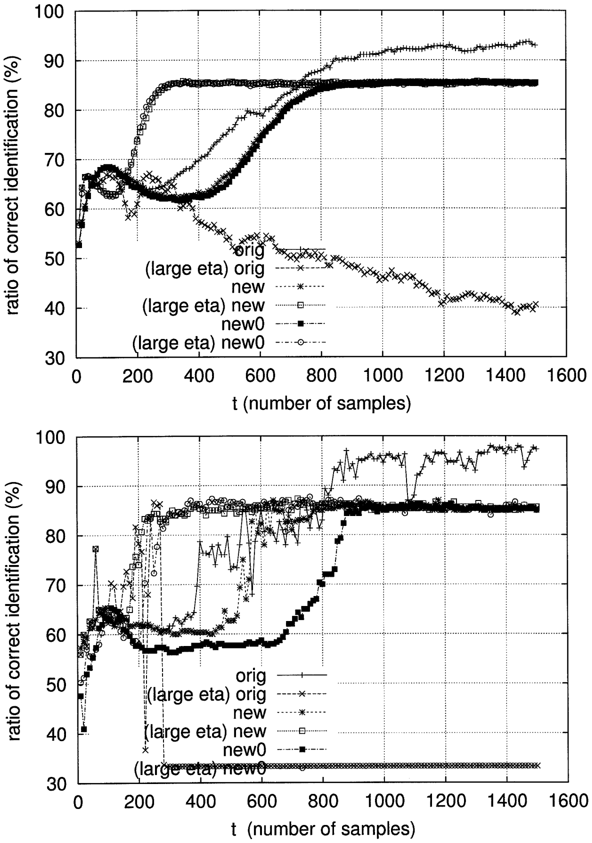


Fig. 5 Learning curves of the original algorithm ("orig") and the presented algorithms ("new0": type I, "new": type II). Upper: mean for 100 independent trials. Lower: an example of one trial. Horizontal axis: number of presented samples. Vertical axis: percentage of correct identification. Two cases $\eta = 0.01$ and $\eta = 0.03$ ("large eta") are shown for each algorithm. The presented algorithms have an advantage that we can take a large learning coefficient η so as to obtain fast convergence. Type I and II show similar result. On the other hand, the original algorithm is superior with regard to the final ratio of correct identification after sufficient learning.

$$A(t)=A(t-1)+\eta\Big[B(t)A(t-1) -\frac{1}{2}B(t)A(t-1)\,A(t-1)^T(\boldsymbol{w}(t)\boldsymbol{w}(t)^T+\epsilon I)A(t-1) -\frac{1}{2}(\boldsymbol{w}(t)\boldsymbol{w}(t)^T+\epsilon I)A(t-1)\,A(t-1)^T B(t)A(t-1)\Big] \tag{28}$$

is used instead of (26), where I is the identity matrix and a regularization coefficient ϵ is a small positive number. The term $+\epsilon I$ is useful for stabilization of the algorithm [17].

The result of the simulation is shown in Figs. 5, 6. The presented algorithms have an advantage that we can take a larger learning coefficient η so as to obtain faster convergence. On the other hand, the original algorithm is superior with regard to the final ratio of correct identification after sufficient learning. This result suggests that a part of useful information is lost during "PCA stage" in Sect. 3.1, because the variance W within classes is out of consideration there.

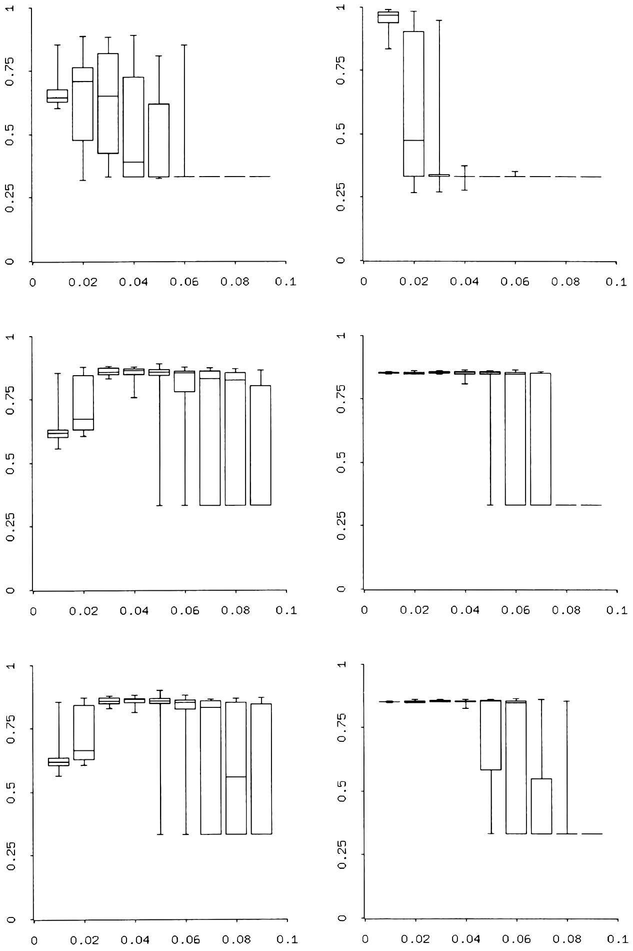


Fig. 6 The ratio of correct identification after $t = 300$ (left) and $t = 1500$ (right) steps of learning. Horizontal axis is the learning coefficient η. Upper: original algorithm, middle: type I algorithm, lower: type II algorithm. The five-number summaries (minimum, first quartile, median, third quartile, and maximum) for 20 trials are displayed.

Combination of the above advantages will be discussed in Sect. 6.

5.2 Comparison with Other Methods

Next, online LDA methods are compared with other "online learning" methods: (1) linear filter[†] (LF) and (2) three-layered feedforward neural network (NN) with sigmoid neurons $y = (1 + \exp(-s))^{-1}$ [13]. The same task as table 1 is used for comparison.

Since we are interested in the case that N is large, keeping and updating $O(N^2)$ variables are unwelcome. For this reason, we adopt gradient learning method

[†] In other words, two-layered feedforward neural networks with linear neurons.

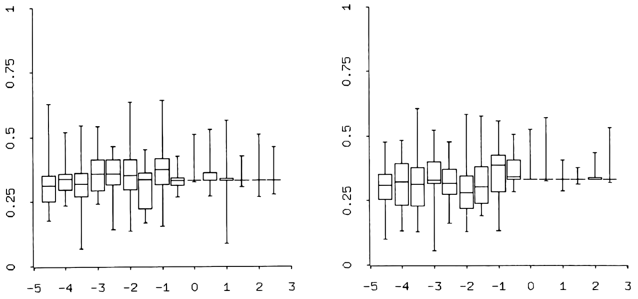


Fig. 7 The ratio of correct identification by linear filter, after $t = 1500$ steps of learning. Horizontal axis is $\log_{10}\eta$. Left: momentum coefficient $\mu = 0$, right: $\mu = 0.9$. The five-number summaries (minimum, first quartile, median, third quartile, and maximum) for 20 trials are displayed.

(backpropagation):

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) + \eta\Delta\boldsymbol{\theta}(t), \tag{29}$$

$$\Delta\boldsymbol{\theta}(t) = (1-\mu)\frac{\partial\|\boldsymbol{z}(t) - \boldsymbol{f}(\boldsymbol{x}(t);\boldsymbol{\theta})\|^2}{\partial\boldsymbol{\theta}} + \mu\Delta\boldsymbol{\theta}(t-1), \tag{30}$$

where $\boldsymbol{\theta}(t)$ is the parameters of LF or NN at step t, $\boldsymbol{f} = (f_1, \cdots, f_M)$ is the output of LF or NN, η is learning coefficient and μ is momentum coefficient. The target output $\boldsymbol{z}(t) = (z_1(t), \cdots, z_M(t))$ is defined as $z_c(t) = \delta_{c,c(t)}$. Initial values of $\boldsymbol{\theta}$ are generated by Nguyen-Widrow method [14]. After learning, the guessed class for $\boldsymbol{x}$ is obtained as $\hat{c} = \arg\max_c f_c(\boldsymbol{x};\boldsymbol{\theta})$.

Results are shown in Figs. 7, 8, 9. Neither LF (Fig. 7) nor NN (Fig. 8) work efficiently. Though NN has potential ability to attain higher performance than OLDAs, convergence of NN is unendurably slow (Fig. 9). This slow convergence is likely to be caused by high dimensionality of the input space R^{100}.

6. Concluding Remarks

Online LDA algorithms are desired in order to increase adaptability of LDA. In the present paper, new algorithms of online LDA are proposed. It is experimentally shown that the new algorithms are about two times faster than the previously proposed algorithm in terms of the number of required examples, while the latter attains better final performance than the former after sufficient steps of learning. The meaning of new algorithms are also discussed theoretically, and they are suggested to be corresponding to combination of PCA and Mahalanobis distance.

A key improvement of the presented algorithm is the fact that it is robust for choice of the learning coefficient η. By choosing a large learning coefficient, we can accelerate the learning process. A method for automatic tuning of η is proposed in [20]. However, as for the result after presentation of many samples, the original online LDA algorithm is superior. Thus, we may be able to obtain a better performance by combining them. Namely, the presented algorithm is used only

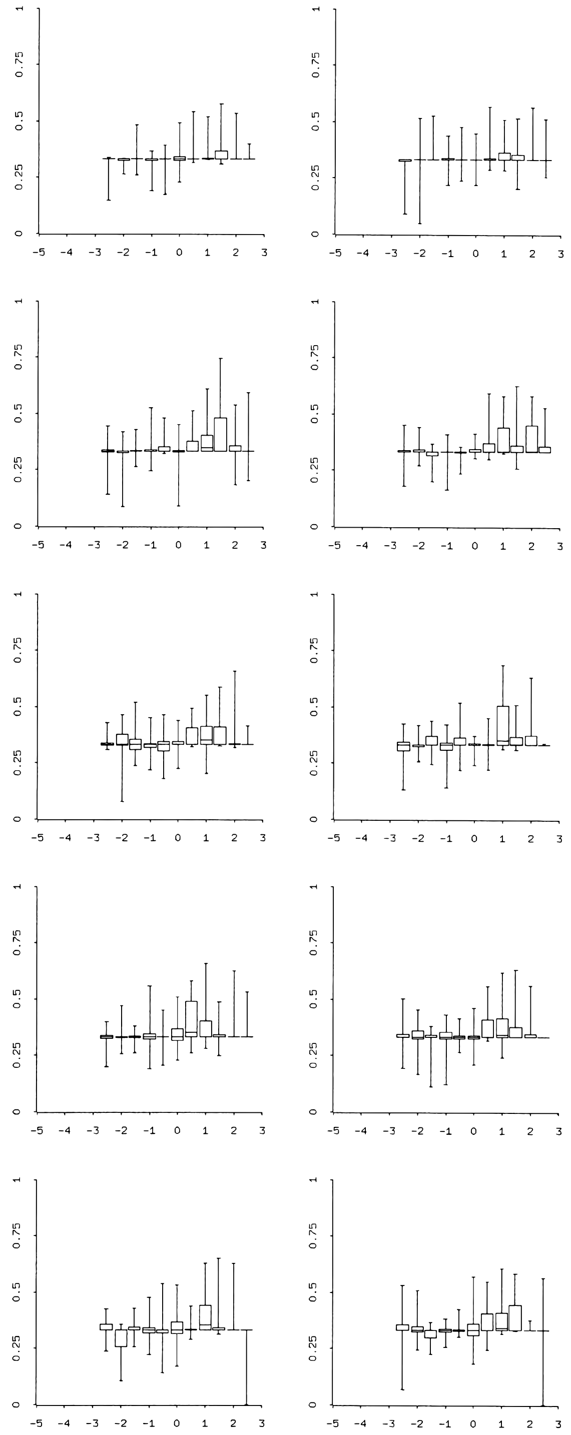


Fig. 8 The ratio of correct identification by three-layered feed-forward neural networks, after $t = 1500$ steps of learning. Horizontal axis is $\log_{10}\eta$. Left: momentum coefficient $\mu = 0$, right: $\mu = 0.9$. From upper to lower: (number of neurons in the hidden layer) $= 2, 5, 10, 20, 50$. The five-number summaries (minimum, first quartile, median, third quartile, and maximum) for 20 trials are displayed. In any cases, median of identification ratio is less than 0.4.

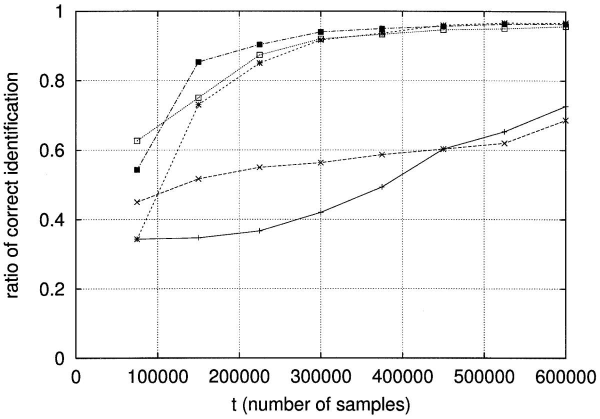
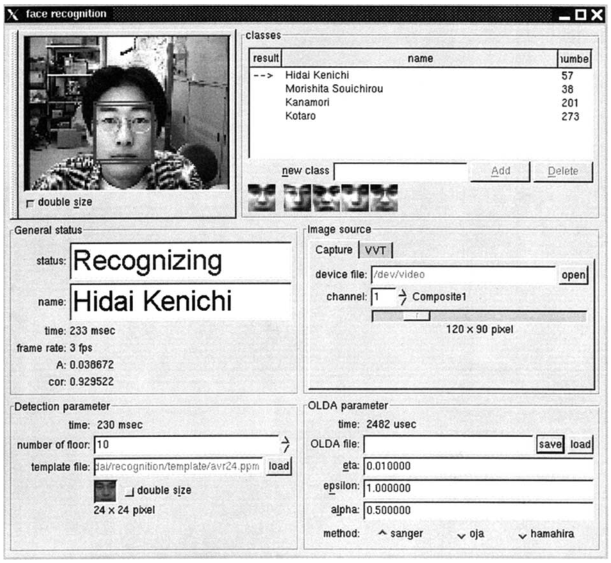


Fig. 9 Examples of learning curves for three-layered feedforward neural networks with 5 neurons in the hidden layer, $\eta = 0.3$, $\mu = 0$. Results of 5 trials are displayed. Since only 1500 samples are available for learning, they are used repeatedly after $t = 1500$. Convergence is much slower compared with online LDA.



result	name	umbe
-->	Hidai Kenichi	57
	Morishita Souichirou	38
	Kanamori	201
	Kotaro	273

status: Recognizing
name: Hidai Kenichi
time: 233 msec
frame rate: 3 fps
A: 0.038672
cor: 0.929522

device file: /dev/video
channel: 1 Composite1
120 × 90 pixel

time: 230 msec
number of floor: 10
template file: dai/recognition/template/avr24.ppm
☐ double size
24 × 24 pixel

time: 2482 usec
OLDA file:
eta: 0.010000
epsilon: 1.000000
alpha: 0.500000
method: ● sanger ○ oja ○ hamahira

Fig. 10 Real-time face recognition system via online LDA.

in early stage. When saturation of the performance is observed, the algorithm is switched to the original one.

Presently, the authors are constructing a real-time face recognition system for robot vision (Fig. 10). This system finds a face in a image and answer the name of the person in about 240 msec: most of the processing time is consumed not in discrimination but in finding phase (CPU: PentiumII 450 MHz × 2, Memory: 256 Mbytes). The previous and new online LDA algorithms are used in this system.

This work has been partly supported by CREST of JST (Japan Science and Technology) 279102 and JSPS (Japan Society for the Promotion of Science) 12750203.

References

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

[2] T.D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network," Neural Networks, vol.2, no.6, pp.459–473, 1989.

[3] K. Hornik and C.-M. Kuan, "Convergence analysis of local feature extraction algorithms," Neural Networks, vol.5, no.2, pp.229–240, 1992.

[4] W.-Y. Yan, U. Helmke, and J.B. Moore, "Global analysis of Oja's flow for neural networks," IEEE Trans. Neural Networks, vol.5, no.5, pp.674–683, 1994.

[5] J. Mao and A.K. Jain, "Artificial neural networks for feature extraction and multivariate data projection," IEEE Trans. Neural Networks, vol.6, no.2, pp.296–317, 1995.

[6] C. Chatterjee and V.P. Roychowdhury, "On self-organizing algorithms and networks for class-separability features," IEEE Trans. Neural Networks, vol.8, no.3, pp.663–678, 1997.

[7] M.B. Nevel'son and R.Z. Has'minskiĭ, Stochastic Approximation and Sequential Estimation, Nauka, 1968. (in Russian; translated into Japanese by T. Kitagawa and K. Tajima, 1983).

[8] T. Kurita and S. Hayamizu, "Gesture recognition using HLAC features of PARCOR images and HMM based recognizer," Proc. 3rd Int. Conference on Automatic Face and Gesture Recognition, pp.422–427, 1998.

[9] W. Zhao, R. Chellappa, and A. Krishnaswamy, "Discriminant analysis of principal components for face recognition," Proc. 3rd Int. Conference on Automatic Face and Gesture Recognition, pp.336–341, 1998.

[10] K. Jonsson, J. Matas, J. Kittler, and Y.P. Li, "Learning support vectors for face verification and recognition," Proc. 4th Int. Conference on Automatic Face and Gesture Recognition, pp.208–213, 2000.

[11] K. Fukunaga, Statistical Pattern Recognition, Academic Press, New York, 1989.

[12] E. Oja, H. Ogawa, and Wangviwattana, "Principal component analysis by homogeneous neural networks, part I and part II," IEICE Trans. Inf. & Syst., vol.E75-D, no.3, pp.366–381, 1992.

[13] D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, eds., Parallel Distributed Processing, vols.1 and 2, Cambridge, The M.I.T. Press, 1986.

[14] D. Nguyen and B. Widrow, "Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights," Proc. International Joint Conference on Neural Networks, vol.3, pp.21–26, 1990.

[15] K. Hiraoka and M. Hamahira, "On successive learning type algorithm for linear discriminant analysis," IEICE Technical Report, NC99-73, 1999.

[16] K. Hiraoka, K. Hidai, M. Hamahira, H. Mizoguchi, T. Shigehara, and T. Mishima, "Derivation of online LDA algorithm and its application for face identification," Fifth Robotics Symposia, pp.226–231, 2000.

[17] K. Hiraoka, S. Yoshizawa, K. Hidai, M. Hamahira, H. Mizoguchi, and T. Mishima, "Convergence analysis of online linear discriminant analysis," International Joint Conference on Neural Networks (IJCNN), vol.III, pp.387–391, 2000.

[18] K. Hiraoka, K. Hidai, M. Hamahira, H. Mizoguchi, T. Mishima, and S. Yoshizawa, "Successive learning of linear discriminant analysis: Sanger-type algorithm," 15th International Conference on Pattern Recognition (ICPR), vol.2, pp.664–667, 2000.

[19] K. Hiraoka, K. Hidai, H. Mizoguchi, T. Mishima, and S. Yoshizawa, “Fast algorithm for online linear discriminant analysis,” Proc. ITC-CSCC’2000, pp.274–277, 2000.

[20] S. Morishita, K. Hiraoka, H. Mizoguchi, and T. Mishima, “Study on automatic setting method of learning coefficient in online LDA towards robust convergence,” Proc. 2000 Information and Systems Society Conferences of IEICE, p.217, 2000.

Appendix: Proofs of Propositions

Proof of proposition 1: If the conditions of the proposition are satisfied, we obtain $dA/dt = A\Gamma - \frac{1}{2}A\Gamma - \frac{1}{2}A\Gamma = O$ □

Proof of proposition 2: If the conditions of the proposition are satisfied, we obtain $dA/dt = A\Gamma - \frac{1}{2}A\Gamma^T - \frac{1}{2}A\Gamma = O$. □

Proof of proposition 3: From the condition

$$\frac{dA}{dt} = BA - \frac{1}{2}BAA^TWA - \frac{1}{2}AA^TWBA = O, \tag{A$\cdot$1}$$

we obtain $BA(2I - A^TWA) = A(A^TWBA)$. Here, the inverse of the $L \times L$ matrix $(2I - A^TWA)$ exists because of the condition that A^TWA does not have an eigenvalue 2. Thus $BA = A\Gamma$ holds when $\Gamma = (A^TWBA)(2I - A^TWA)^{-1}$. In addition, putting $BA = A\Gamma$ into (A$\cdot$1) and multiplying A^T from left, we obtain $(A^TA)\Gamma(A^TWA-I)+(A^TA)(A^TWA-I)\Gamma = O$. This means (14) when rank$A = L$, because $(A^TA)^{-1}$ then exists. □

Proof of proposition 4: From the condition

$$\frac{dA}{dt} = BA - \frac{1}{2}AA^TBWA - \frac{1}{2}AA^TWBA = O, \tag{A$\cdot$2}$$

we obtain $BA = A\left(\frac{1}{2}A^T(BW + WB)A\right)$. Thus $BA = A\Gamma$ holds and Γ is symmetric when $\Gamma = \frac{1}{2}A^T(BW + WB)A$. In addition, (14) is proved in a similar way as the proof of Proposition 3 when rank$A = L$. Now we introduce a lemma:

Lemma 1: Let $Z = (z_{ij})$ and $H = (h_{ij})$ be square matrices of same dimension. If $ZH + HZ = O$ and H is symmetric positive semi-definite, $ZH = HZ = O$.

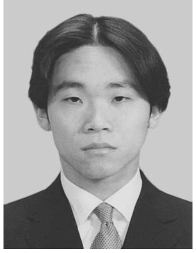
□

(Proof of Lemma: We can assume that H is diagonal without loss of generality, by putting Q^TZQ and Q^THQ as Z' and H' respectively for a certain orthogonal matrix Q: $H = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$ and $\lambda_1, \cdots, \lambda_n \geq 0$. Then $(ZH + HZ)_{ij} = (\lambda_i + \lambda_j)z_{ij} = 0$ means $\lambda_i = \lambda_j = 0$ or $z_{ij} = 0$. Therefore, $(ZH)_{ij} = \lambda_j z_{ij} = 0$ for all i, j.)

Applying this lemma to (14), we obtain (15) because Γ is guaranteed to be symmetric positive semi-definite. This guarantee is derived as follows: Since Γ is symmetric, there exists an orthogonal matrix Q and a diagonal matrix $D = \mathrm{diag}(d_1, \cdots, d_l)$ such that $\Gamma = Q^TDQ$. Then, $BA' = A'D$, where $A' = AQ^T$. This implies that $d_1, \cdots, d_l$ are the eigenvalues of B. Since B is symmetric positive semi-definite, $d_1, \cdots, d_l \geq 0$. Therefore, Γ is positive semi-definite.

Finally, if rank$(BA) = L$, rankΓ must be L since $BA = A\Gamma$. Then Γ^{-1} exists and $A^TWA = I$ is derived from (15). □

Kazuyuki Hiraoka received B.E. degree in 1992 and M.E. degree in 1994 in Mathematical Engineering and Information Physics, and D.E. degree in 1998 in Information Engineering, from the University of Tokyo. He is currently a Research Associate at the Department of Information and Computer Sciences, Saitama University. He is interested in learning systems.



Masashi Hamahira received B.E. degree in 2000 in Information and Computer Sciences, from Saitama University. He is currently in the Fujitsu Limited. He is interested in learning systems.

Ken-ichi Hidai received B.E. degree in 1999 in Information and Computer Sciences from Saitama University. He is currently a Graduate Student at the Master Course in Information and Computer Sciences, Saitama University. He is interested in real-time human machine interaction.

Hiroshi Mizoguchi received B.E. degree in 1980 in Mathematical Engineering and Information Physics, M.E. degree in 1982 and D.E. degree in 1985 in Information Engineering, from the University of Tokyo. He is currently an Associate Professor at the Department of Information and Computer Sciences, Saitama University. His research interests are intelligent machines, real-time system, human machine interaction, real world computing, and sensor-motor interaction.

Taketoshi Mishima received the B.E., M.E. and Ph.D. degrees in Electrical Engineering from Meiji University, in 1968, 1970 and 1973, respectively. He is currently a Professor at the Department of Information and Computer Sciences, Saitama University. His research interests are foundation of symbolic and algebraic computation, axiomatic logic system, mathematical pattern recognition, quantum chaos, and parallel computation.

Shuji Yoshizawa received B.E. degree in 1962 and M.E. degree in 1964 in Applied Physics, and D.E. degree in 1971 in Mathematical Engineering and Information Physics, from the University of Tokyo. He is currently a Professor at the Department of Information and Computer Sciences, Saitama University. His academic interest covers nonlinear dynamics, biocybernetics, and modeling of brain functions.