An Adaptive Quasi-Newton Algorithm for Eigensubspace Estimation

Zhengjiu Kang, *Member, IEEE*, Chanchal Chatterjee, *Member, IEEE*, and Vwani P. Roychowdhury

***Abstract*—In this paper, we derive and discuss a new adaptive quasi-Newton eigen-estimation algorithm and compare it with the RLS-type adaptive algorithms and the quasi-Newton algorithm proposed by Mathew *et al.* through experiments with stationary and nonstationary data.**

***Index Terms*—Adaptive estimation, deflation, eigen-subspace, principal component.**

I. Introduction

It IS well known that the gradient-descent-based adaptive principal component analysis (PCA) algorithms are slow to converge. Bannour *et al.* [3] and Yang [4] derive RLS-type adaptive eigendecomposition algorithms from neural network and optimization frameworks, respectively. Strictly speaking, they are gradient descent algorithms with time-varying, self-tuning step sizes and are still slow to converge. In [1], Mathew *et al.* present a rapidly convergent quasi-Newton adaptive algorithm derived from a cost function based on the penalty function method of optimization. This algorithm sequentially estimates the eigenvector corresponding to the smallest eigenvalue up to the eigenvector corresponding to the largest eigenvalue. In many applications, however, we are only interested in estimating the principal eigenvectors. Reversing the estimation order of this algorithm (i.e., from the "largest" eigenvector to the "smallest" eigenvector sequentially) is not straightforward if we employ the Hessian approximation method proposed in [1] directly in the quasi-Newton algorithm.

In this paper, using a different method to approximate Hessian, we derive a novel quasi-Newton-based adaptive eigensubspace estimation algorithm that first estimates the principal eigenvector and then estimates the minor eigenvectors sequentially. Moreover, in our algorithm, the choice of the penalty coefficient does not need any *a priori* information of the data covariance matrix while in the Mathew *et al.* algorithm, this choice needs *a priori* information of data. In next section, we study the landscape of the cost function. Section III proposes the new Quasi-Newton algorithm and gives the convergence analysis. The extensive experiments with stationary and nonstationary data are given in Section IV. Section V contains concluding remarks.

Manuscript received August 30, 1999; revised July 5, 2000. The associate editor coordinating the review of this paper and approving it for publication was Dr. Kristine L. Bell.

Z. Kang and V. P. Roychowdhury are with the Department of Electrical Engineering, University of California, Los Angeles, CA 90095-1594 USA (e-mail: zjkang@ee.ucla.edu).

C. Chatterjee is with Marconi Integrated Systems, Inc., San Diego, CA 92127-1806 USA.

Publisher Item Identifier S 1053-587X(00)10152-7.

II. Landscape of Cost Function

Let $\mathbf{x}_k \in \Re^N$ be a snapshot of wide sense stationary data sequence $\{\mathbf{x}_k\}$ at time instant k. We want to adaptively compute the first $p\ (\leq N)$ principal eigenvectors and corresponding eigenvalues of a positive definite symmetric matrix R, which is the covariance matrix of $\{\mathbf{x}_k\}$. Without loss generality, we make the following assumption.

Assumption A1): Matrix R is positive definite. The $p\ (p \leq N)$ principal eigenvalues of R are each of unit multiplicity.

Let $\lambda_1 > \lambda_2 > \cdots > \lambda_p > \lambda_{p+1} \geq \cdots \geq \lambda_N > 0$ be the eigenvalues of R in decreasing order, and let $\phi_i\ (i = 1, 2, \ldots, N)$ denote the corresponding orthonormal eigenvectors. Let $\Phi = [\phi_1\ \phi_2\ \cdots\ \phi_N]$ and $\Lambda = \mathrm{diag}(\lambda_1\ \lambda_2\ \cdots\ \lambda_N)$ be the eigenvector and eigenvalue matrices, respectively, of R. In this paper, we only consider zero mean data.

We want to minimize following cost function with respect to $\mathbf{w}^i$

$$J(\mathbf{w}^i; R) = -\mathbf{w}^{i^T} R^i \mathbf{w}^i + \mu \left(\mathbf{w}^{i^T}\mathbf{w}^i - 1\right)^2 \tag{1}$$

where

$$R^i = R - \sum_{j<i} \mathbf{w}^j \mathbf{w}^{j^T} R \quad (1 \leq i \leq p) \tag{2}$$

defines a deflation on R, and $\mathbf{w}^i \in \Re^N, \mu > 0$ is a penalty coefficient.

Chauvin [5] shows that when $i = 1$, the cost function consists of one local maximum, a set of saddle points, and one global minimum that is oriented along the first principal eigenvector of R. Our extension analysis shows that using such a deflation (2), the strict global minimum points of (1) are the principal eigenvectors of R. Similar results have been studied in [2] from the noise subspace.

Theorem 1: Let A1) hold. Then, all equilibrium points of the joint objective functions $J(\mathbf{w}^i; R)$ for $i = 1, 2, \ldots, p$ are up to an arbitrary permutation of the eigenvectors of R, i.e., any point $W = [\rho_{(1)}\phi_{(1)}\ \rho_{(2)}\phi_{(2)}\ \cdots\ \rho_{(p)}\phi_{(p)}]$, where $\rho_{(i)} = 0$ or $\pm\sqrt{1 + (\lambda_{(i)}/2\mu)}$, is an equilibrium point of the objective functions $J(\mathbf{w}^i; R)$ for $i = 1, 2, \ldots, p$.

Proof: We need to find a $W^* = [\mathbf{w}^{1*}\ \mathbf{w}^{2*}\ \cdots\ \mathbf{w}^{p*}]$ such that $\mathbf{g}_{ij} = \nabla_{\mathbf{w}^j} J(\mathbf{w}^{i*}; R) = 0$ for $i, j = 1, \ldots, p$. We expand $\mathbf{w}^i$ in terms of the entire orthonormal set of eigenvectors of R as $\mathbf{w}^i = \sum_{r_i=1}^N a_{r_i}\phi_{r_i}$. By induction, we are able to get above result. ∎

Theorem 2: Let A1) hold. Then, the points $W = [\rho_1\phi_1\ \rho_2\phi_2\ \cdots\ \rho_p\phi_p]$, where $\rho_i = \pm\sqrt{1 + (\lambda_i/2\mu)}$, are the strict global minimum points of the joint objective functions $J(\mathbf{w}^i; R)$ for $i = 1, 2, \ldots, p$. In addition, the points

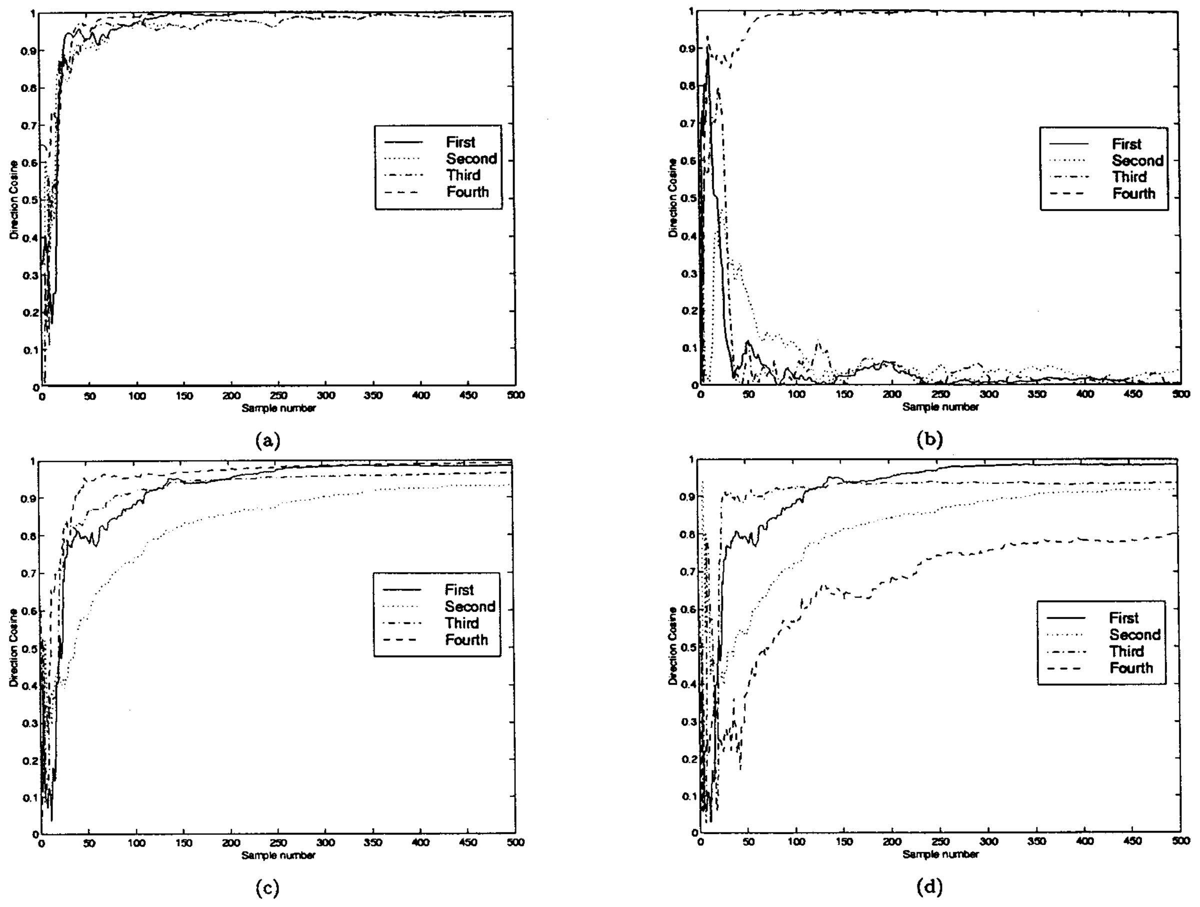


Fig. 1. Convergence of the first four principal eigenvectors with the (a) QN1, (b) QN2, (c) PASTd, and (d) RLS algorithms.

$W = [\rho_{(1)}\phi_{(1)}\ \rho_{(2)}\phi_{(2)}\ \cdots\ \rho_{(p)}\phi_{(p)}]$, where $\rho_{(i)} = 0$ for $i \le p$ or $\phi_{(i)} \ne \phi_i$ are unstable equilibrium points of the objective functions $J(\mathbf{w}^i; R)$ for $i = 1, 2, \ldots, p$.

Proof: The proof is given in the Appendix. ■

III. Algorithm Derivation and Analysis

The gradient of $J(\mathbf{w}^i; R^i)$ with respect to $\mathbf{w}^i$ at time instant k is given by

$$\mathbf{g}_k^i = -2R_k^i\mathbf{w}_k^i + 4\mu\left(\mathbf{w}_k^{i^T}\mathbf{w}_k^i - 1\right)\mathbf{w}_k^i \tag{3}$$

and the Hessian matrix is

$$H_k^i = -2R_k^i + 8\mu\mathbf{w}_k^i\mathbf{w}_k^{i^T} + 4\mu\left(\mathbf{w}_k^{i^T}\mathbf{w}_k^i - 1\right)I_N \tag{4}$$

where I_N is an $N \times N$ identity matrix. R_k^i is an estimate of R^i at k. $R_k^1 = R_k$ is updated by the following rank-one equation:

$$R_k = \frac{1}{k}\sum_{j=1}^{k}\theta^{k-j}\mathbf{x}_j\mathbf{x}_j^T = \theta\left(\frac{k-1}{k}\right)R_{k-1} + \frac{1}{k}\mathbf{x}_k\mathbf{x}_k^T \tag{5}$$

where $0 < \theta \le 1$ is a forgetting factor. For a stationary system, we choose $\theta = 1$, and for a nonstationary system, we choose $0 < \theta < 1$. The exact value of θ depends on the specific application.

In order to compute the inverse of Hessian efficiently, we define

$$Q_k^i = -R_k^i + 2\mu\left(\mathbf{w}_k^{i^T}\mathbf{w}_k^i - 1\right)I_N. \tag{6}$$

Therefore, the inverse of the Hessian becomes

$$\begin{aligned} H_k^{i^{-1}} &= \left(2Q_k^i + 8\mu\mathbf{w}_k^i\mathbf{w}_k^{i^T}\right)^{-1} \\ &= \frac{1}{2}\left[Q_k^{i^{-1}} - \frac{Q_k^{i^{-1}}\mathbf{w}_k^i\mathbf{w}_k^{i^T}Q_k^{i^{-1}}}{\frac{1}{4\mu} + \mathbf{w}_k^{i^T}Q_k^{i^{-1}}\mathbf{w}_k^i}\right]. \end{aligned} \tag{7}$$

Assuming $\mathbf{w}_k^i$ is close to the solution $\mathbf{w}^{i^*}$ at time instant k, the inverse of Q_k^i can be approximated as

$$Q_k^{i^{-1}} \approx \frac{1}{2\mu\left(\mathbf{w}_k^{i^T}\mathbf{w}_k^i - 1\right)}\left[I_N + \frac{R_k^i}{2\mu\left(\mathbf{w}_k^{i^T}\mathbf{w}_k^i - 1\right)}\right]. \tag{8}$$

After simplification, we get the following adaptive quasi-Newton algorithm:

$$\mathbf{w}_{k+1}^i = L_k^i\mathbf{w}_k^i \tag{9}$$

$$L_k^i = \frac{\mathbf{w}_k^{i^T}\mathbf{w}_k^i}{\tau_k^i}\left[2\mu\left(\mathbf{w}_k^{i^T}\mathbf{w}_k^i - 1\right)I_N + R_k^i\right] \tag{10}$$

$$\begin{aligned}\tau_k^i = {} & \mu\left(\mathbf{w}_k^{i^T}\mathbf{w}_k^i - 1\right)^2 + 2\mu\mathbf{w}_k^{i^T}\mathbf{w}_k^i\left(\mathbf{w}_k^{i^T}\mathbf{w}_k^i - 1\right) \\ & + \mathbf{w}_k^{i^T}R_k^i\mathbf{w}_k^i \end{aligned} \tag{11}$$

$$R_k^i = \begin{cases} R_k & i = 1 \\ R_k^{i-1} - \mathbf{w}_k^{i-1}\mathbf{w}_k^{i-1^T}R_k & 2 \le i \le p \end{cases} \tag{12}$$

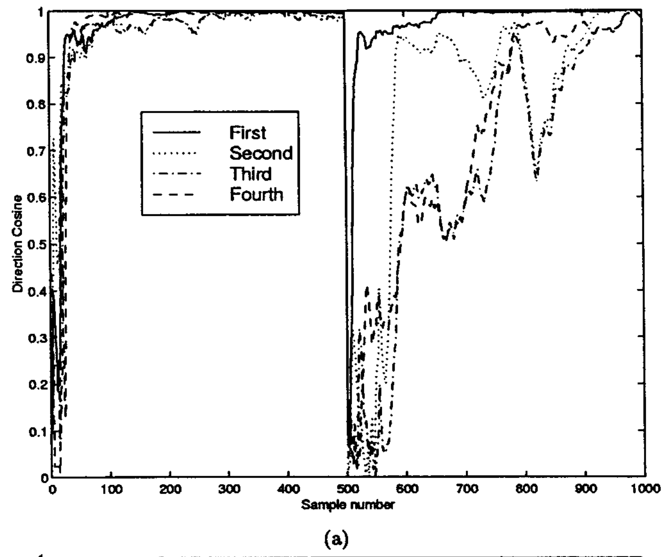


Fig. 2. Convergence performance of QN1 and QN2 algorithm to compute the first four eigenvectors under nonstationary data sequences with $\mu = 15$ and $\theta = 0.9955$. (a) QN1. (b) QN2.

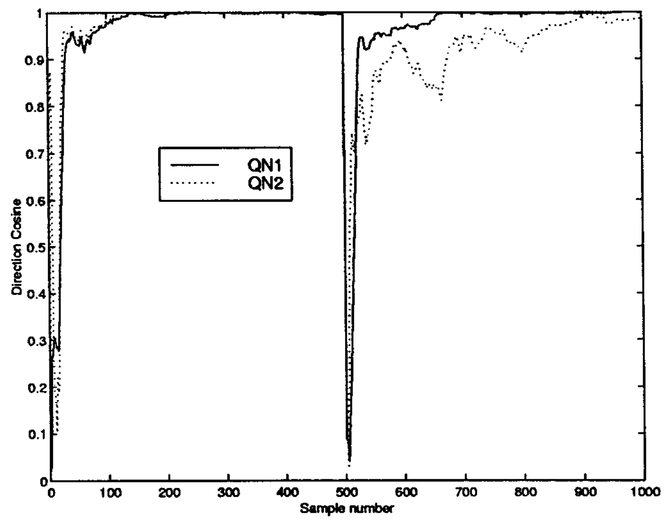


Fig. 3. Convergence performance of the QN1 and QN2 algorithms to track the first eigenvector under nonstationary data sequences with $\mu = 1000$ and $\theta = 0.9955$.

$$R_k = \frac{k-1}{k} R_{k-1} + \frac{1}{k} \mathbf{x}_k \mathbf{x}_k^T. \tag{13}$$

The approximate Hessian is obtained by dropping one term of the Hessian in [1]. We reached an approximate Hessian through geometric progression approximation. In such a geometric progression approximation, more terms can be used to keep the approximate Hessian closer to the exact Hessian.

We have the following result for the convergence of the above algorithm.

Assumption A2): Each R_k is bounded with probability one (w.p. 1), symmetric, real, non-negative definite, and $\lim_{k\to\infty} E[R_k] = R$ w.p. 1.

Theorem 3: Let A1) and A2) hold. For any $\mu > 0$ and initial values $\mathbf{w}_0^i$ sufficiently close to the desired eigenvectors, if process $\{\mathbf{w}_k^i\}$ is generated by the quasi-Newton algorithm (9)–(13), then a) $\lim_{k\to\infty} \mathbf{w}_k^i = \phi_i$, and b) $\lim_{k\to\infty} \tau_k^i = \lambda_i$ for $i = 1, 2, \ldots, p$, where λ_i is the ith eigenvalue of R, and ϕ_i is the corresponding eigenvector of R.

Proof: a) can be completed by first proving $\lim_{k\to\infty} \mathbf{w}_k^i = \rho_i \phi_i$ and then proving $\rho_i = \pm 1$; b) immediately follows a).

IV. Simulations

In this section, we compare our adaptive PCA algorithm (QN1) with the following:

1) the Mathew *et al.* quasi-Newton algorithm (QN2);
2) the Yang RLS algorithm (PASTd);
3) the Bannour *et al.* RLS algorithm (RLS).

We demonstrate these comparisons with two different simulation methods. In both methods, high-dimensional data sequences are generated.

A. Method 1

In this simulation method, we use the first covariance matrix from [6] multiplied by 2. We generate samples of 10-dimensional Gaussian data $(N = 10)$ with mean zero and covariance R. The eigenvalues of the this covariance matrix R in decreasing order are

$$\begin{array}{ccccc} 11.7996 & 5.5644 & 3.4175 & 2.0589 & 0.7873 \\ 0.5878 & 0.1743 & 0.1423 & 0.1213 & 0.1007. \end{array}$$

Clearly, the first four eigenvalues are significant, and corresponding eigenvectors are important. We use *direction cosine* (DC) to measure the convergence and accuracy of the adaptive algorithms. It is defined as

$$\mathrm{DC}_k^i = \mathbf{w}_k^{i^T} \phi_i \Big/ \left\|\mathbf{w}_k^i\right\| \|\phi_i\| \quad i = 1, 2, \ldots, p \quad (p \le N)$$

where $\mathbf{w}_k^i$ is the estimated ith eigenvector of R_k corresponding to the ith eigenvalue at the kth update, and ϕ_i is the actual eigenvector of R computed by a conventional numerical analysis method. Clearly, $\lim_{k\to\infty} \mathrm{DC}_k^i = 1$ if $\lim_{k\to\infty} \mathbf{w}_k^i = \phi_i$. We start our algorithm with $\mathbf{w}_0^i = 0.8 \cdot \mathbf{i}_i$ for $i = 1, 2, \ldots, p$ $(p \le N)$, where $\mathbf{i}_i$ denotes the ith column of $N \times N$ identity matrix I_N. For other adaptive algorithms, the initial values are QN2: $\mathbf{w}_0^i = 0.8 \cdot \mathbf{i}_i, R_0^{i^{-1}} = 100 \cdot I_N$; PASTd: $\mathbf{w}_0^i = 0.8 \cdot \mathbf{i}_i$, $d_0^i = 1$; RLS:$\mathbf{w}_0^i = 0.8 \cdot \mathbf{i}_i$, $P_0^i = 100 \cdot \mathbf{i}_i$.

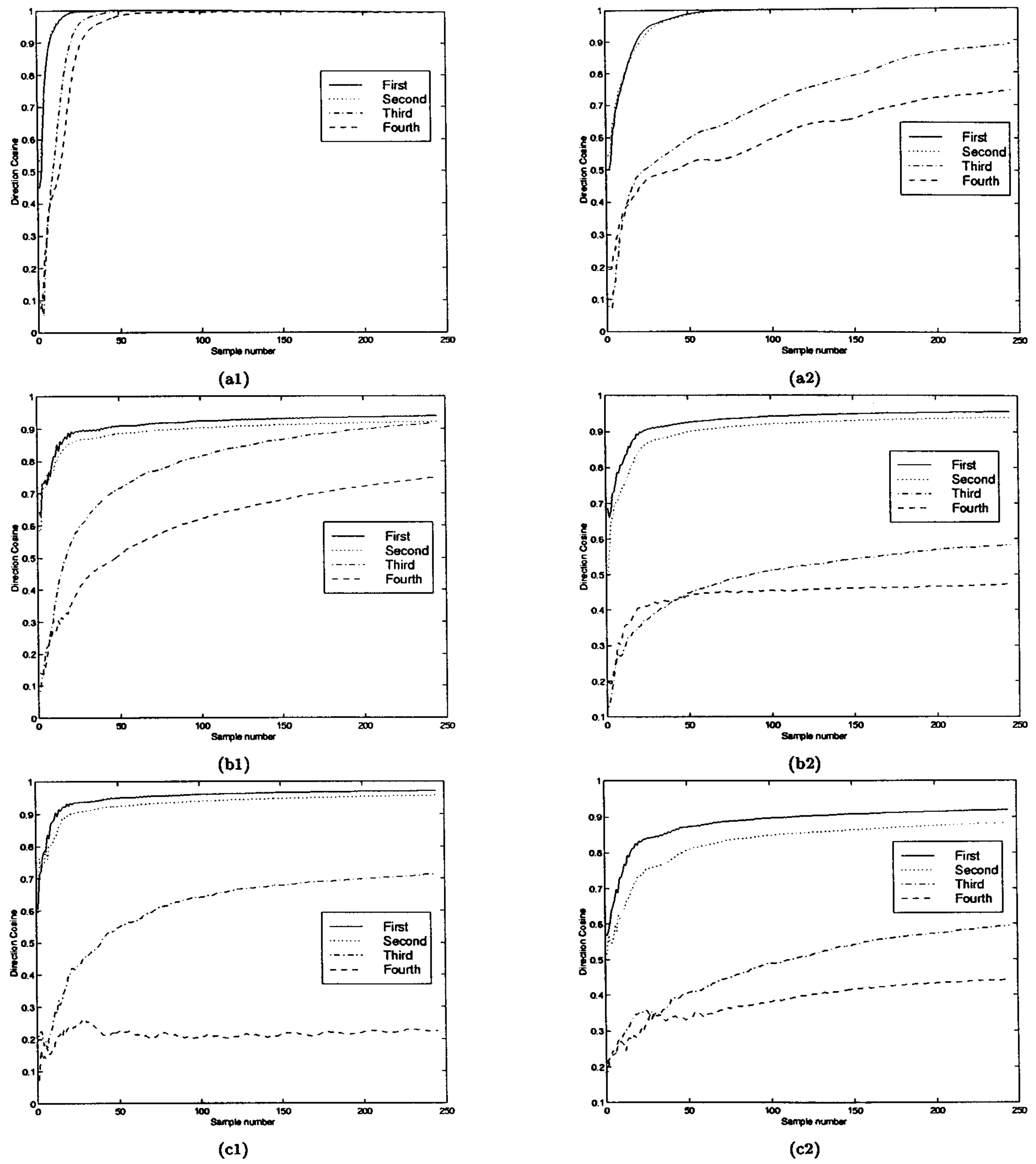


Fig. 4. Convergence of the first four principal eigenvectors with the (a) QN1, (b) PASTd, and (c) RLS algorithm for (1) SNR $= 10$ dB and (2) SNR $= 0$ dB.

Suppose we do not know any *a priori* information on data sequence, we simply choose $\mu = 1$ to compute the first four principal eigenvectors for all of the adaptive algorithms. Fig. 1 shows the convergence of these adaptive algorithms.

The simulation results demonstrate that QN1 and QN2 algorithms offer faster convergence rates than the PASTd and RLS algorithms. We also see that the QN2 algorithm does not estimate the first three principal eigenvectors when $\mu = 1$. In the QN2 algorithm, μ has to be larger than $\lambda_{\max}/2$ or $\mu > \text{Trace}(R)/2$. It turns out that the QN2 algorithm requires a $\mu > \lambda_{\max}/2 \approx 5.8998$ to estimate all eigenvectors. Clearly, in the QN2 algorithm, an ad hoc selection of μ is risky for applications where we do not have *a priori* information on the data covariance matrix R or where there are jumps in eigenvalues of R in the time-varying data sequences. However, when a large suitable penalty coefficient μ is chosen, the QN2 algorithm exhibits very similar convergence performance as the QN1 algorithm for stationary signals.

From experimental results, we also find that the QN1 and QN2 algorithms have good tracking properties, even for minor eigenvectors, whereas PASTd and RLS do not have this performance. The results are omitted here.

In order to demonstrate the tracking abilities of the two quasi-Newton algorithms (QN1 and QN2) under a nonstationary environment, we provide a time-varying scenario in which the covariance matrix of data sequences change suddenly. The first

500 data samples of the data sequence have the covariance R, as described above. After 500 samples, we abruptly change to covariance R', which is obtained from the second matrix in [6] multiplied by 30. Accordingly, the eigenvalues change from

$$\begin{array}{lllll} 11.7996 & 5.5644 & 3.4175 & 2.0589 & 0.7873 \\ 0.5878 & 0.1743 & 0.1423 & 0.1213 & 0.1007 \end{array}$$

to

$$\begin{array}{lllll} 178.8639 & 101.2184 & 86.0674 & 65.3080 & 22.4001 \\ 18.3543 & 11.3537 & 7.6971 & 1.4791 & 2.2881. \end{array}$$

As discussed in Section III, we use forgetting factor θ to reduce the effect of old data and enhance the tracking ability in the nonstationary case. Therefore, we use iterative equation (5) to update R_k in the QN1 algorithm. Accordingly, we use

$$R_k^{-1} = \frac{k}{(k-1)\theta}\left[R_{k-1}^{-1} - \frac{R_{k-1}^{-1}\mathbf{x}_k\mathbf{x}_k^T R_{k-1}^{-1}}{(k-1)\theta + \mathbf{x}_k^T R_{k-1}^{-1}\mathbf{x}_k}\right] \quad (14)$$

to update R_k^{-1} in the QN2 algorithm. Fig. 2 shows the simulation results for both quasi-Newton adaptive algorithms when $\mu = 15$ and $\theta = 0.9955$. We observe that the QN2 algorithm does not track the first principal eigenvector, whereas the QN1 algorithm tracks all of eigenvectors quite well. Fig. 3 shows the results of QN1 and QN2 algorithms to track the first principal eigenvector when $\mu = 1000$ and $\theta = 0.9955$. It demonstrates that the QN1 algorithm converges faster and tracks the changes in data much better than the QN2 algorithm. It is clear that QN1 is superior to QN2 under a nonstationary environment.

B. Method 2

In this simulation, we follow the experimental setup in [1], where the data x_k is generated as

$$x_k = A\sin(0.4\pi k) + A\sin(0.48\pi k + \vartheta) + v_k \qquad k = 1, 2, \ldots$$

The initial phase ϑ is uniformly distributed in $[-\pi\ \pi]$, and v_k is white noise with zero mean and unit variance. Scalar A determines the desired *signal-to-noise ratio* (SNR) as $10\log(A^2/2)$. Here, we choose a window of length N and the dimension of the covariance as 6. We estimate the first $p = 4$ principal eigenvectors.

The initial values for each algorithm are the same as those used in Method 1. We run 100 Monte Carlo simulations for each algorithm. We only compare the QN1 algorithm with the RLS-type algorithms since the QN2 algorithm has very similar tracking performance as QN1 under stationary environment for a large value of μ. We choose $\mu = 1000$ for the QN1 algorithm. Fig. 4 presents the simulation results for SNR $=$ 10 dB and 0 dB.

We observe that the QN1 adaptive algorithm gives the best tracking performance, irrespective of a high or low SNR.

V. Conclusion

We study the landscape of a cost function and derive a new quasi-Newton adaptive algorithm for PCA. Comparison with RLS-type algorithms shows that the QN1 algorithm has faster and better tracking ability. We especially compare two QN algorithms (QN1 and QN2) since they offer the fastest convergence rates under a stationary environment for a large value of penalty coefficient. Nonstationary experiments show that the QN1 algorithm has superior tracking performance than the QN2 algorithm. Further consideration should be given to the computational complexity of the algorithms since the QN1 and QN2 algorithms have the computational complexity of $O(pN^2)$, whereas the PASTd and RLS algorithms offer $O(pN)$ complexity.

Appendix

This Appendix proves Theorem 2. From the objective function, we define the following energy functions:

$$E_i(\mathbf{w}^1, \ldots, \mathbf{w}^i) = -\mathbf{w}^{i^T} R\mathbf{w}^i + \sum_{j=1}^{i-1} \mathbf{w}^{i^T}\mathbf{w}^j\mathbf{w}^{j^T}R\mathbf{w}^i + \mu\left(\mathbf{w}^{i^T}\mathbf{w}^i - 1\right)^2 \quad (15)$$

for $i = 1, \ldots, p$ $(p \le N)$. Let us assume that $\rho_{(i)} = 0$ for some $i \le p$, i.e.,

$$W = \left[\rho_{(1)}\phi_{(1)}, \ldots, \rho_{(i-1)}\phi_{(i-1)}, 0 \quad \rho_{(i+1)}\phi_{(i+1)}, \ldots, \rho_{(p)}\phi_{(p)}\right].$$

Next, we perturb the ith column of W by $\delta\phi_{(i)}$. We observe that

$$E_i\left(\rho_{(1)}\phi_{(1)}, \ldots, \rho_{(i-1)}\phi_{(i-1)}, \delta\phi_{(i)}\right) - E_i\left(\rho_{(1)}\phi_{(1)} \ldots, \rho_{(i-1)}\phi_{(i-1)}, 0\right) = -\delta^2\lambda_{(i)} + \mu(\delta^2 - 1)^2 - \mu.$$

Clearly, the energy function decrease for some $\delta < \sqrt{2 + (\lambda_{(i)}/\mu)}$.

We next prove that if $\phi_{(i)} \ne \phi_i$ for some $j = 1, \ldots, p$, then the critical points W are unstable equilibrium points of (15). We obtain the Hessian of E_i with respect to $\mathbf{w}^i$ as

$$\nabla^2_{\mathbf{w}^i} E_i(\mathbf{w}^1, \ldots, \mathbf{w}^i) = -2R + \sum_{j=1}^{i-1}\mathbf{w}^j\mathbf{w}^{j^T}R + \sum_{j=1}^{i-1} R\mathbf{w}^j\mathbf{w}^{j^T} + 4\mu\left(\mathbf{w}^{i^T}\mathbf{w}^i - 1\right)I + 8\mu\mathbf{w}^i\mathbf{w}^{i^T}. \quad (16)$$

We prove the result by induction. We note that $\nabla^2_{\mathbf{w}^1}E_1(\rho_1\phi_1) = -2R + 4\mu(\rho_1^2 - 1)I + 8\mu\rho_1^2\phi_1\phi_1^T$. The eigenvectors of $\nabla^2_{\mathbf{w}^1}E_1(\rho_1\phi_1)$ are $\phi_1, \ldots, \phi_N$. The eigenvalues are $8\mu\rho_1^2$ for ϕ_1 and $2(\lambda_1 - \lambda_r)$ for ϕ_r, $(r > 1)$. Clearly, $\nabla^2_{\mathbf{w}^1}E_1(\rho_1\phi_1)$ is positive definite. On the other hand, we observe that

$$\nabla^2_{\mathbf{w}^1}E_1(\rho_r\phi_r) = -2R + 4\mu(\rho_r^2 - 1)I + 8\mu\rho_r^2\phi_1\phi_1^T \quad (r > 1).$$

We see that the eigenvectors of $\nabla^2_{\mathbf{w}^1}E_1(\rho_r\phi_r)$ are $\phi_1, \ldots, \phi_N$. However, the eigenvalues are $-2(\lambda_1 - \lambda_r)$ for ϕ_1, and $8\mu\rho_r^2$ for $\phi_r (r > 1)$. Clearly, $\nabla^2_{\mathbf{w}^1}E_1(\rho_r\phi_r)$ for $r > 1$ is an indefinite matrix. Thus, $\rho_1\phi_1$ is a stable equilibrium point for the energy function E_1, whereas $\rho_1\phi_1$ for $r > 1$ are unstable equilibrium points of E_1.

We next assume that $(\rho_1\phi_1, \ldots, \rho_{i-1}\phi_{i-1})$ are the stable equilibrium points for the energy function E_{i-1}. Then, for $\phi_{(i)} = \phi_i$, we have from (15)

$$\nabla^2_{\mathbf{w}^i} E_i(\rho_1\phi_1, \ldots, \rho_{i-1}\phi_{i-1}, \rho_i\phi_i) = -2R + 4\mu\left(\rho_i^2 - 1\right) I + 8\mu\rho_i^2\phi_i\phi_i^T + 2\sum_{j=1}^{i-1}\rho_j^2\phi_j\phi_j^T.$$

The eigenvectors of $\nabla^2_{\mathbf{w}^i} E_i(\rho_1\phi_1, \ldots, \rho_{i-1}\phi_{i-1}, \rho_i\phi_i)$ are $\phi_1, \ldots, \phi_N$. The eigenvalues are $2\lambda_i + \lambda_r^2/\mu$ for $\phi_r(r < i), 8\mu\rho_i^2$ for ϕ_i, and $2(\lambda_i - \lambda_r)$ for $\phi_r(r > i)$. Thus, $\nabla^2_{\mathbf{w}^i} E_i(\rho_1\phi_1, \ldots, \rho_{i-1}\phi_{i-1}, \rho_i\phi_i)$ is positive definite. On the other hand, if $\phi_{(i)} = \phi_i(r > i)$ then

$$\nabla^2_{\mathbf{w}^i} E_i(\rho_1\phi_1, \ldots, \rho_{i-1}\phi_{i-1}, \rho_r\phi_r) = -2R + 4\mu(\rho_r^2 - 1)I + 8\mu\rho_r^2\phi_r\phi_r^T + 2\sum_{j=1}^{i-1}\rho_j^2\phi_j\phi_j^T.$$

The eigenvalues of $\nabla^2_{\mathbf{w}^i} E_i(\rho_1\phi_1, \ldots, \rho_{i-1}\phi_{i-1}, \rho_r\phi_r)$ are $-2(\lambda_i - \lambda_r)$ for ϕ_i, and $8\mu\rho_r^2$ for ϕ_r $(r > i)$. Thus, $\nabla^2_{\mathbf{w}^i} E_i(\rho_1\phi_1, \ldots, \rho_{i-1}\phi_{i-1}, \rho_r\phi_r)$ is an indefinite matrix. We therefore see that $(\rho_1\phi_1, \ldots, \rho_i\phi_i)$ are the stable equilibrium points of E_i.

We conclude that $W = [\rho_1\phi_1 \;\; \rho_2\phi_2 \;\; \cdots \;\; \rho_p\phi_p]$, where $\rho_i = \pm\sqrt{1 + (\lambda_i/2\mu)}$ are stable equilibrium points of the energy functions (15), whereas $W = [\rho_{(1)}\phi_{(1)} \;\; \rho_{(2)}\phi_{(2)} \;\; \cdots \;\; \rho_{(p)}\phi_{(p)}]$, where $\rho_{(i)} = 0$ for $i \leq p$ or $\phi_{(i)} \neq \phi_i$ are unstable equilibrium points.

References

[1] G. Mathew, V. U. Reddy, and S. Dasgupta, "Adaptive estimation of eigensubspace," *IEEE Trans. Signal Processing*, vol. 43, pp. 401–411, Feb. 1995.

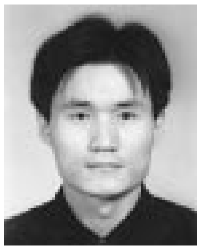
[2] G. Mathew and V. U. Reddy, "Orthogonal eigensubspace estimation using neural networks," *IEEE Trans. Signal Processing*, vol. 42, pp. 1803–1811, July 1994.

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

[4] B. Yang, "Projection approximation subspace tracking," *IEEE Trans. Signal Processing*, vol. 43, pp. 95–107, Jan. 1995.

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

[6] T. Okada and S. Tomita, "An optimal orthonormal system for discriminant analysis," *Pattern Recognit.*, vol. 18, no. 2, pp. 139–144, 1985.



Zhengjiu Kang (M'98) received the B.S. degree in control engineering from Harbin Shipbuilding Engineering Institute, Harbin, China, in 1991 and the Ph.D. degree in system engineering from Xi'an Jiaotong University, Xi'an, China, in 1996. He is now pursuing the Ph.D. degree in wireless communications at the Electrical Engineering Department at University of California, Los Angeles.

From 1996 to 1997, he was a Researcher at Control Information Systems Laboratory, Seoul National University, Seoul, Korea. His current research interests include adaptive algorithms, wireless communications, neural networks, and parallel processing.

Chanchal Chatterjee (M'88) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, and the M.S.E.E. degree from Purdue University, West Lafayette, IN, in 1983 and 1984, respectively. In 1996, he received the Ph.D. degree in electrical and computer engineering from Purdue University.

Between 1985 and 1995, he worked at Machine Vision International and Medar, Inc., both in Detroit, MI. He is currently a Senior Algorithms Specialist at BAE Systems, San Diego, CA. He is also affiliated with the Electrical Engineering Department, University of California, Los Angeles. His areas of interest include image processing, computer vision, neural networks, and adaptive algorithms and systems for pattern recognition and signal processing.

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

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