RECURSIVE PRINCIPAL COMPONENTS ANALYSIS USING EIGENVECTOR MATRIX PERTURBATION

Hemanth Peddaneni, Deniz Erdogmus, Yadunandana N. Rao,
Anant Hegde, Jose C. Principe

Computational NeuroEngineering Laboratory,
Electrical & Computer Engineering Department
University of Florida, Gainesville, FL 32611.
Email: [hemanth, deniz, yadu, ahegde, principe] @cnel.ufl.edu

Abstract. Principal Components Analysis is an important and well-studied topic in statistics and signal processing. Most algorithms could be grouped into one of the following three approaches: adaptation based on Hebbian updates and deflation, optimization of a second order statistical criterion, and fixed point update rules with deflation. In this paper, we propose a completely different approach that updates the eigenvector and eigenvalue matrices with every new data sample, such that the estimates approximately track their true values. The performance is compared with traditional methods like Sanger and APEX algorithm, as well as with a similar matrix perturbation based method. The results show the efficiency of the algorithm in terms of convergence speed and accuracy.

INTRODUCTION

Principal Components Analysis (PCA) is a well known statistical technique that has been widely applied to solve many signal processing problems like data reduction, feature extraction, signal estimation, detection and speech separation [1-4]. PCA is nothing but the eigendecomposition of the data covariance matrix. Analytical techniques [5] that solve PCA once the entire data is known exist. These methods require extensive matrix operations and hence are not suitable for real-time applications. In such cases, fast, adaptive, online solutions are desirable.

Majority of the existing algorithms for adaptive PCA are based on standard gradient procedures [2,3,6-9]. The problem with these methods is that they are extremely slow converging and their performance heavily depends on step-sizes used. To alleviate this, subspace methods have been explored in [10–12]. However, many of these subspace techniques are computationally expensive. The algorithm described in [13] showed fast convergence with little or no change in complexity compared with the gradient methods. However this method and most of the existing methods in literature rely on deflation technique, which brings in sequential convergence of principal components. This potentially reduces the overall speed of convergence of the algorithm.

The recently developed simultaneous principal component extraction algorithm called SIPEX [14] reduced the gradient search only to the space of orthonormal matrices, resulting in fast and simultaneous convergence. However it has a very high computational complexity due to the involved trigonometric function evaluations. A recently proposed approach suggested the method of calculating the eigenvectors and eigenvalues iteratively using a first order matrix perturbation of the data covariance matrix estimate with every new sample obtained [15]. However the performance (speed and accuracy) of this algorithm is hindered by the general symmetric structure of the perturbed covariance matrix. In this paper, we will present an algorithm that undertakes a similar perturbation approach, but in contrast, the covariance matrix will be decomposed into its eigenvectors and eigenvalues at all times, which will reduce the perturbation step to be employed on the diagonal eigenvalue matrix. This restriction results in a faster converging and more accurate subspace tracking algorithm.

The paper is organized as follows. First, we present a brief overview of the PCA problem. Second, the proposed recursive PCA algorithm (RPCA) is motivated, derived and extended for non-stationary signal environments. Some of the practical aspects of the algorithm are then discussed. Next, a set of computer experiments is presented to demonstrate the convergence speed and accuracy characteristics of RPCA. Finally, we conclude the paper with remarks and observations about the algorithm.

PROBLEM DEFINITION

Without loss of generality, we will consider a real-valued zero-mean, *n*-dimensional random vector $\mathbf{x}$. The projections of $\mathbf{x}$ onto unit-norm vectors $\mathbf{w}_j$'s are given by $y_j = \mathbf{w}_j^T \mathbf{x}$. Principal components are those vectors $\mathbf{w}$ along which the variance of y is extremal.

The first principal component direction is defined as the solution to the following constrained optimization problem, where $\mathbf{R}$ is the input covariance matrix:

$$\mathbf{w}_1 = \arg\max_{\mathbf{w}} \mathbf{w}^T \mathbf{R} \mathbf{w} \quad \text{subject to} \quad \mathbf{w}^T \mathbf{w} = 1 \tag{1}$$

The subsequent principal components are defined by including the additional constraints to the problem that enforce the orthogonality of the sought component to the previously obtained ones:

$$\mathrm{w_j} = \operatorname*{argmax}_{\mathbf{w}} \mathbf{w}^{\mathrm{T}}\mathbf{R}\mathbf{w}, \text{ s.t. } \mathbf{w}^{\mathrm{T}}\mathbf{w} = 1, \mathbf{w}^{\mathrm{T}}\mathbf{w}_{\mathrm{l}} = 0, \mathrm{l} < \mathrm{j} \tag{2}$$

The overall solution to this problem turns out to be the eigenvector matrix of the input covariance matrix $\mathbf{R}$. In particular, the principal component directions are given by the eigenvectors of R arranged according to their corresponding eigenvalues (largest to smallest) [5].

In signal processing applications, the needs are different. The input samples are usually acquired one at a time (i.e., sequentially as opposed to in batches), which necessitates sample-by-sample update rules for the covariance and its eigenvector estimates. In this setting, this analytical solution is of little use, since it is not practical to update the input covariance estimate and solve a full eigendecomposition problem for each sample. However, utilizing the recursive structure of the covariance estimate, it is possible to come up with a recursive formula for the eigenvectors of the covariance as well. This will be described in the next section.

RECURSIVE PCA DESCRIPTION

Let $\mathbf{x}$ be an n dimensional zero mean wide sense stationary input vector. Our aim is to find the principal components of $\mathbf{x}$ at each instant of time k. The correlation matrix of $\mathbf{x}$ at time k is defined as:

$$\mathbf{R}_k = \frac{1}{k}\sum_{i=1}^{k}\mathbf{x}_i\mathbf{x}_i^T = \frac{(k-1)}{k}\mathbf{R}_{k-1} + \frac{1}{k}\mathbf{x}_k\mathbf{x}_k^T \tag{3}$$

If $\mathbf{Q}$ and $\mathbf{\Lambda}$ are the orthonormal eigenvector and diagonal eigenvalue matrices of $\mathbf{R}$, then $\mathbf{R}_k = \mathbf{Q}_k\mathbf{\Lambda}_k\mathbf{Q}_k^T$ and $\mathbf{R}_{k-1} = \mathbf{Q}_{k-1}\mathbf{\Lambda}_{k-1}\mathbf{Q}_{k-1}^T$. So (3) reduces to

$$\mathbf{Q}_k(k\mathbf{\Lambda}_k)\mathbf{Q}_k^T = \mathbf{x}_k\mathbf{x}_k^T + (k-1)\mathbf{Q}_{k-1}\mathbf{\Lambda}_{k-1}\mathbf{Q}_{k-1}^T \tag{4}$$

If we let $\boldsymbol{\alpha}_k = \mathbf{Q}_{k-1}^T\mathbf{x}_k$, (4) can be written as

$$\mathbf{Q}_k(k\mathbf{\Lambda}_k)\mathbf{Q}_k^T = \mathbf{Q}_{k-1}[(k-1)\mathbf{\Lambda}_{k-1} + \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T]\mathbf{Q}_{k-1}^T \tag{5}$$

If $\mathbf{V}_k$ and $\mathbf{D}_k$ are the orthonormal eigenvector and diagonal eigenvalue matrices of $(k-1)\mathbf{\Lambda}_{k-1} + \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T$, then

$$(k-1)\mathbf{\Lambda}_{k-1} + \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T = \mathbf{V}_k\mathbf{D}_k\mathbf{V}_k^T \tag{6}$$

So (5) now reduces to

$$\mathbf{Q}_k(k\mathbf{\Lambda}_k)\mathbf{Q}_k^T = \mathbf{Q}_{k-1}\mathbf{V}_k\mathbf{D}_k\mathbf{V}_k^T\mathbf{Q}_{k-1}^T \tag{7}$$

By comparing both sides of (7), the recursive eigenvector and eigenvalue update rules turn out to be:

$$\begin{aligned}\mathbf{Q}_k &= \mathbf{Q}_{k-1}\mathbf{V}_k \\ \mathbf{\Lambda}_k &= \mathbf{D}_k / k\end{aligned} \tag{8}$$

This is the basic equation for updating the eigenvalues and eigenvectors of the correlation matrix $\mathbf{R}$. The problem now is to determine the eigendecomposition of $(k-1)\mathbf{\Lambda}_{k-1} + \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T$ (i.e. to find $\mathbf{V}_k$ and $\mathbf{D}_k$), which is difficult to solve analytically. So we will make use of first order matrix perturbation analysis.

Perturbation analysis to find $\mathbf{V}_k$ and $\mathbf{D}_k$: Consider the following sample perturbation to the eigenvalue matrix: $(k-1)\mathbf{\Lambda}_{k-1} + \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T$. When k is large, this matrix is basically a diagonal matrix, which means that $\mathbf{D}_k$ will be close to $(k-1)\mathbf{\Lambda}_{k-1}$ and $\mathbf{V}_k$ will be close to the identity matrix $\mathbf{I}$. The matrix $\boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T$ is said to perturb the diagonal matrix $(k-1)\mathbf{\Lambda}_{k-1}$ as a result of which $\mathbf{D}_k = (k-1)\mathbf{\Lambda}_{k-1} + \mathbf{P}_\Lambda$ and $\mathbf{V}_k$=**I**+$\mathbf{P}_\mathrm{V}$, where $\mathbf{P}_\Lambda$ and $\mathbf{P}_\mathrm{V}$ are small perturbation matrices. Now if we can find these perturbation matrices, we would have solved the eigendecomposition problem of $(k-1)\mathbf{\Lambda}_{k-1} + \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T$. Letting $\mathbf{\Lambda} = (k-1)\mathbf{\Lambda}_{k-1}$, $\mathbf{V}_k\mathbf{D}_k\mathbf{V}_k^T$ can be expanded as,

$$\begin{aligned}\mathbf{V}_k\mathbf{D}_k\mathbf{V}_k^T &= (\mathbf{I}+\mathbf{P}_\mathrm{V})(\mathbf{\Lambda}+\mathbf{P}_\Lambda)(\mathbf{I}+\mathbf{P}_\mathrm{V})^T \\ &= \mathbf{\Lambda} + \mathbf{\Lambda}\mathbf{P}_\mathrm{V}^T + \mathbf{P}_\Lambda + \mathbf{P}_\Lambda\mathbf{P}_\mathrm{V}^T + \mathbf{P}_\mathrm{V}\mathbf{\Lambda} \\ &\quad + \mathbf{P}_\mathrm{V}\mathbf{\Lambda}\mathbf{P}_\mathrm{V}^T + \mathbf{P}_\mathrm{V}\mathbf{P}_\Lambda + \mathbf{P}_\mathrm{V}\mathbf{P}_\Lambda\mathbf{P}_\mathrm{V}^T \\ &= \mathbf{\Lambda} + \mathbf{P}_\Lambda + \mathbf{D}\mathbf{P}_\mathrm{V}^T + \mathbf{P}_\mathrm{V}\mathbf{D} \\ &\quad + \mathbf{P}_\mathrm{V}\mathbf{\Lambda}\mathbf{P}_\mathrm{V}^T + \mathbf{P}_\mathrm{V}\mathbf{P}_\Lambda\mathbf{P}_\mathrm{V}^T\end{aligned} \tag{9}$$

Substituting this equation in (6) and assuming $\mathbf{P}_\mathrm{V}\mathbf{\Lambda}\mathbf{P}_\mathrm{V}^T$ and $\mathbf{P}_\mathrm{V}\mathbf{P}_\Lambda\mathbf{P}_\mathrm{V}^T$ are negligible,

$$\boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T = \mathbf{P}_\Lambda + \mathbf{D}_k\mathbf{P}_\mathrm{V}^T + \mathbf{P}_\mathrm{V}\mathbf{D}_k \tag{10}$$

The orthonormality of **V** brings an additional equation that characterizes $\mathbf{P}_\mathrm{V}$. Substituting **V**=**I**+$\mathbf{P}_\mathrm{V}$ in $\mathbf{V}\mathbf{V}^T$=**I**, and assuming that $\mathbf{P}_\mathrm{V}\mathbf{P}_\mathrm{V}^T \approx \mathbf{0}$, we have $\mathbf{P}_\mathrm{V} = -\mathbf{P}_\mathrm{V}^T$. So combining the fact that the $\mathbf{P}_\mathrm{V}$ is anti-symmetric and $\mathbf{P}_\Lambda$, $\mathbf{D}_k$ are diagonal, we can get the solution.

$$\alpha_i^2 = (i,i)^{th} \text{ element of } \mathbf{P}_\Lambda \tag{11}$$

$$\left.\begin{aligned}\frac{\alpha_i\alpha_j}{\lambda_j+\alpha_j^2-\lambda_i^2-\alpha_i^2} &= (i,j)^{th} \text{ element of } \mathbf{P}_\mathrm{V}, i \neq j \\ 0 &= (i,i)^{th} \text{ element of } \mathbf{P}_\mathrm{V}\end{aligned}\right\} \tag{12}$$

where λ_i, λ_j are the elements of the eigenvalue matrix $(k-1)\mathbf{\Lambda}_{k-1}$.

THE RPCA ALGORITHM

The RPCA algorithm is summarized in Table 1. There are a few practical issues regarding the operation of the algorithm, which will be addressed in this section.

Memory Depth Parameter (λ): In a stationary environment, where all samples are weighted equally, the memory depth parameter must be set to λ_k=1/*k*. The recursive update for the covariance matrix is given by (3).

In a non-stationary environment, a first order dynamical forgetting strategy could be employed by selecting a fixed forgetting factor. Setting $\lambda_k = \lambda$,

$$\mathbf{R}_k = (1-\lambda)\mathbf{R}_{k-1} + \lambda\mathbf{x}_k\mathbf{x}_k^T \tag{13}$$

In this forgetting scheme, $\lambda\in(0,1)$ is selected to be very small. Considering that the average memory depth of this recursion is $1/\lambda$ samples, the selection of this

Table 1: The Recursive PCA Algorithm

```
1. Initialize the orthonormal eigenvector matrix $\mathbf{Q}_0 = \mathbf{I}$ and diagonal eigenvalue
   matrix $\Lambda_0$ =diag($\mathbf{R}_{N0}$) where $R_{N0}$ is the estimated covariance matrix using $N_0$
   samples of input

2. At each time instant $k$ do the following:
   a. Get input sample $\mathbf{x}_k$.
   b. Set memory depth parameter $\lambda_k = 1/(k+(\tau-1)N_0)$, where $\tau$ is a positive
      integer.
   c. Calculate $\boldsymbol{\alpha}_k = \mathbf{Q}_{k-1}^T \mathbf{x}_k$.
   d. Use (11) and (12) to find perturbations $\mathbf{P}_V$ and $\mathbf{P}_\Lambda$ corresponding to
      $(1-\lambda_k)\boldsymbol{\Lambda}_{k-1} + \lambda_k \boldsymbol{\alpha}_k \boldsymbol{\alpha}_k^T$.
   e. Update eigenvector and eigenvalue matrices:
      $\tilde{\mathbf{Q}}_k = \mathbf{Q}_{k-1}(\mathbf{I} + \mathbf{P}_V)$
      $\tilde{\boldsymbol{\Lambda}}_k = (1-\lambda_k)\boldsymbol{\Lambda}_{k-1} + \lambda_k \mathbf{P}_\Lambda$
   f. Normalize the norms of eigenvector estimates by $\mathbf{Q}_k = \tilde{\mathbf{Q}}_k \mathbf{T}_k$, where $\mathbf{T}_k$ is
      a diagonal matrix containing the inverses of the norms of each column of
      $\tilde{\mathbf{Q}}_k$.
   g. Correct eigenvalue estimates by $\boldsymbol{\Lambda}_k = \tilde{\boldsymbol{\Lambda}}_k \mathbf{T}_k^{-2}$, where $\mathbf{T}_k^{-2}$ is a diagonal
      matrix containing the squared norms of the columns of $\tilde{\mathbf{Q}}_k$.
```

parameter presents a trade-off between tracking capability and estimation variance.

Initializing the eigenvectors and eigenvalues: The eigenvector matrix $\mathbf{Q}_0$ and eigenvalue matrix $\boldsymbol{\Lambda}_0$ can be initialized by using the first N_0 samples to obtain an unbiased estimate of covariance matrix R. Q_0 can be initialized to I and the eigenvalues can be initialized to the sample variances of the each input entry over N_0 samples (i.e. Λ_0=diag R_{N0}). Since we had already used N_0 samples, the iterations in step 2 of Table 1 can be applied to the subsequent samples i.e. samples from N_0+1 to N.

In the stationary case (where λ_k=1/k), the perturbation approximations in the first few iterations in step 2 of table 1 will not be accurate. This is because for small values of k, $(1-\lambda_k)\boldsymbol{\Lambda}_{k-1} + \lambda_k \boldsymbol{\alpha}_k \boldsymbol{\alpha}_k^T$ is not strongly diagonal, contrary to our assumption. This problem could be avoided if in the step 2, the index k could be started from a large initial value, which effectively means using large number of samples in the initialization (i.e. choosing large N_0). In practice, this is often undesirable. The alternative is to perform the initialization still using a small number of samples, but setting the memory depth parameter to $\lambda_k=1/(k+(\tau-1)N_0)$. By doing this, when the iteration starts at sample $k=N_0+1$, the algorithm thinks that the initialization is actually performed using $\gamma = \tau\, N_0$ samples. Instead a time varying γ ($\gamma=\gamma_0\exp(-k/\tau)$), exponentially decaying profile, is found to produce better estimates of eigenvalues and eigenvectors than a fixed γ. So such γ will be used for all the simulations shown in the next section for a stationary environment.

Therefore, from the point-of-view of the algorithm, the data set looks like

$$\underbrace{\{\mathbf{x}_1,\ldots,\mathbf{x}_{N_0}\},\ldots,\{\mathbf{x}_1,\ldots,\mathbf{x}_{N_0}\}}_{\text{repeated } \tau \text{ times}},\{\mathbf{x}_{N_0+1},\ldots,\mathbf{x}_N\} \tag{14}$$

The resulting covariance estimator will be biased. The estimated covariance matrix at the end of the iterations is :

$$\mathbf{R}_{N,\,biased} = \frac{N}{N+(\tau-1)N_0}\mathbf{R}_N + \frac{(\tau-1)N_0}{N+(\tau-1)N_0}\mathbf{R}_{N_0} \tag{15}$$

So the bias introduced by tricking the algorithm can be asymptotically diminished (i.e. as $N \to \infty$).

In the non-stationary case (i.e., $\lambda_k = \lambda$), the same initialization strategy can be used i.e. $\mathbf{Q}_0 = \mathbf{I}$ and Λ_0 =diagR_{N0}. The initialization bias is not a problem, since its effect will diminish in accordance with the forgetting time constant anyway. Also, in order to guarantee the accuracy of the first order perturbation approximation, we need to choose the forgetting factor λ such that (1- λ)/ λ is large. Typically, a forgetting factor $\lambda < 10^{-2}$ will yield accurate results.

EXPERIMENTAL RESULTS

Convergence Speed Analysis: In this experiment, the goal is to investigate the convergence speed and accuracy of the RPCA algorithm. For this, n-dimensional random vectors are drawn from a normal distribution with an arbitrary covariance matrix. In particular, the theoretical covariance matrix of the data is given by AA^T, where A is an nxn real-valued matrix whose entries are drawn from a zero-mean unit-variance Gaussian distribution. This process results in a wide range of eigenspreads (as shown in Fig. 1), therefore the convergence results shown here encompass such effects.

Specifically, the results of the 3-dimensional case study are presented here, where the data is generated by 3-dimensional normal distributions with randomly selected covariance matrices. A total of 1000 simulations (Monte Carlo runs) are carried out for each of the three target eigenvector estimation accuracies (measured in terms of angle between the estimated and actual eigenvectors): 10°, 5°, and 2°. The convergence time is measured in terms of number of iterations it takes the algorithm to converge to the target eigenvector accuracy in all eigenvectors (not just the principal component). The histograms of convergence times (up to 10000 samples) for these three target accuracies are shown in Fig. 2, where everything above 10000 is also lumped into the last bin. In these Monte Carlo runs, the initial eigenvector estimates were set to the identity matrix and the randomly selected data covariance matrices were forced to have eigenvectors such that all the initial eigenvector estimation errors were at least 25°. The initial γ value was set to 400 and the decay time constant was selected to be τ=50 samples. Values in this range were found to work best in terms of final accuracy and convergence speed in extensive Monte Carlo runs.

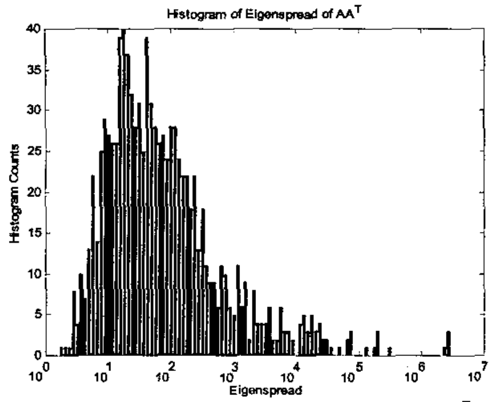

Figure 1. Distribution of eigenspread values for $\mathbf{AA}^T$, where $\mathbf{A}_{3x3}$ is generated to have Gaussian distributed random entries.

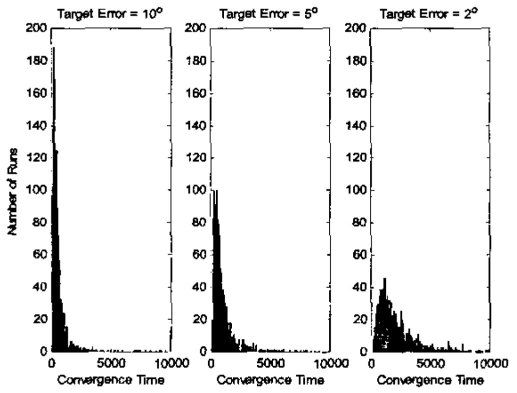

Figure 2. The convergence time histograms for RPCA in the 3-dimensional case for three different target accuracy levels.

It is expected that there are some cases, especially those with high eigenspreads, which require a very large number of samples to achieve very accurate eigenvector estimations, especially for the minor components. The number of iterations required for convergence to a certain accuracy level is also expected to increase with the dimensionality of the problem. For example, in the 3-dimensional case, about 2% of the simulations failed to converge within 10° in 10000 on-line iterations, whereas this ratio is about 17% for 5 dimensions. The failure to converge within the given number of iterations is observed for eigenspreads over $5x10^4$.

In a similar setup, Sanger's rule achieves a mean convergence speed of 8400 iterations with a standard deviation of 2600 iterations. This results in an average

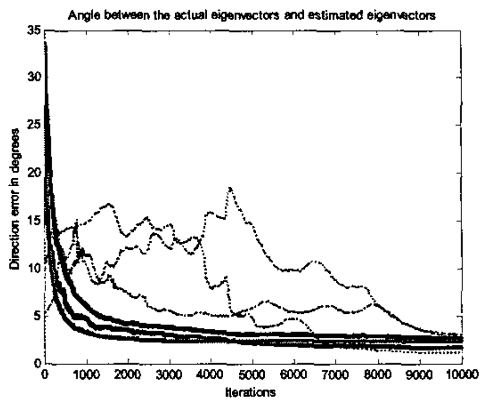

Figure 3. The average eigenvector direction estimation errors versus iterations for the first order perturbation method(thin dotted lines) and for RPCA (thick solid lines)

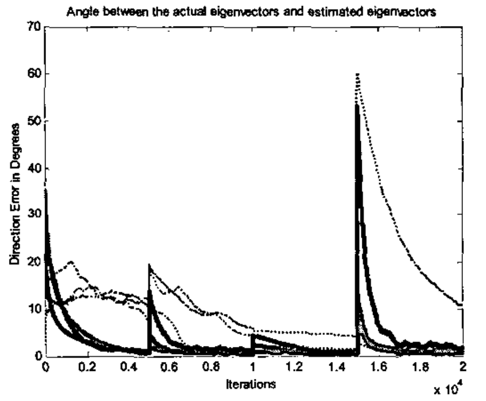

Figure 4. The average eigenvector direction estimation errors versus iterations for the first order perturbation method (thin dotted lines) and for RPCA (thick solid lines) in a piecewise stationary situation. The eigenstructure of the input abruptly changes every 5000 samples.

eigenvector direction error of about 9° with a standard deviation of 8°. APEX on the other hand converges rarely to within 10°. Its average eigenvector direction error is about 30° with a standard deviation of 15°.

Comparison with Alternative First Order Perturbation PCA: The first order perturbation PCA algorithm [15] is structurally similar to the RPCA algorithm presented here. The main difference is the nature of the perturbed matrix: the former works on a perturbation approximation for the complete covariance matrix, whereas the latter considers the perturbation of a diagonal matrix. We expect this structural restriction to improve the performance of the algorithm in terms of overall convergence speed and accuracy.

An experimental setup similar to that in the previous experiment (i.e. a stationary environment) was used. The data is generated by a colored time-series using a time-delay line (making the procedure a temporal PCA case study). Gaussian white noise is colored using a two-pole filter whose poles are selected from a random uniform distribution on the interval (0,1). A set of 15 Monte Carlo simulations was run on 3 dimensional data generated accordingly. The two parameters of the first order perturbation method were optimized to $\varepsilon=10^{-3}/6.5$ and $\delta=10^{-2}$. The parameters of RPCA were set to $\gamma_0=300$ and $\tau=100$. The average eigenvector convergence curves are shown in Fig. 3.

To illustrate the performance of RPCA for non-stationary cases, a piecewise stationary colored noise sequence is generated by filtering white Gaussian noise with single-pole filters with the following poles: 0.5, 0.7, 0.3, 0.9 (in order of appearance). The forgetting factor is set to a constant $\lambda=10^{-3}$. The two parameters of the first order perturbation method were again set to $\varepsilon=10^{-3}/6.5$ and $\delta=10^{-2}$. The results of 30 Monte Carlo runs were averaged to obtain Fig. 4.

CONCLUSIONS

In this paper, a novel perturbation-based fixed-point algorithm for subspace tracking is presented. The fast tracking capability is enabled by the recursive nature of the complete eigenvector matrix updates. The proposed algorithm is feasible for real-time implementation since the recursions are based on well-structured matrix multiplications that are the consequences of the rank-one perturbation updates exploited in the derivation of the algorithm. Performance comparisons with traditional algorithms, as well as a similar perturbation-based approach demonstrated the advantages of the RPCA algorithm.

Acknowledgment: This work is supported by NSF grant ECS-0300340.

REFERENCES

[1] R.O. Duda, P.E. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.

[2] S.Y. Kung, K.I. Diamantaras, J.S. Taur, "Adaptive Principal Component Extraction (APEX) and Applications," IEEE Transactions on Signal Processing, vol. 42, no. 5, pp. 1202-1217, 1994.

[3] J. Mao, A.K. Jain, "Artificial Neural Networks for Feature Extraction and Multivariate Data Projection," IEEE Transactions on Neural Networks, vol. 6., no. 2, pp. 296-317, 1995.

[4] Y. Cao, S. Sridharan, M. Moody, "Multichannel Speech Separation by Eigendecomposition and its Application to Co-Talker Interference Removal,"

IEEE Transactions on Speech and Audio Processing, vol. 5, no. 3, pp. 209-219, 1997.

[5] G. Golub, C.V. Loan, *Matrix Computation,* Johns Hopkins University Press, Baltimore, MD, 1993.

[6] E. Oja, Subspace Methods for Pattern Recognition, Wiley, New York, 1983.

[7] T.D. Sanger, "Optimal Unsupervised Learning in a Single Layer Linear Feedforward Neural Network," Neural Networks, vol. 2, no. 6, pp. 459-473, 1989.

[8] J. Rubner, K. Schulten, "Development of Feature Detectors by Self Organization," Biological Cybernetics, vol. 62, pp. 193-199, 1990.

[9] J. Rubner, P. Tavan, "A Self Organizing Network for Principal Component Analysis," Europhysics Letters, vol. 10, pp. 693-698, 1989.

[10] L. Xu, "Least Mean Square Error Reconstruction Principle for Self-Organizing Neural-Nets", Neural Networks, vol. 6, pp. 627-648, 1993.

[11] B. Yang, "Projection Approximation Subspace Tracking", IEEE Transactions on Signal Processing, vol. 43, no. 1, pp. 95-107, 1995.

[12] Y. Hua, Y. Xiang, T. Chen, K. Abed-Meriam, Y. Miao, "Natural Power Method for Fast Subspace Tracking", Proceedings of NNSP'99, pp. 176-185, 1999.

[13] Y.N. Rao, J.C. Principe, "Robust On-line Principal Component Analysis Based on a Fixed-Point Approach". Proceedings of ICASSP'02, vol. 1, pp. 981-984, 2002.

[14] D. Erdogmus, Y.N. Rao, K.E. Hild II, J.C. Principe, "Simultaneous Principal Component Extraction with Application to Adaptive Blind Multiuser Detection," EURASIP Journal on Applied Signal Processing, vol. 2002, no. 12, pp. 1473-1484, 2002.

[15] B. Champagne, "Adaptive Eigendecomposition of Data Covariance Matrices Based on First-Order Perturbations," IEEE Transactions on Signal Processing, vol. 42, no. 10, 1994.