Journal of VLSI Signal Processing 45, 85–95, 2006

DOI: 10.1007/s11265-006-9773-6

Perturbation-Based Eigenvector Updates for On-Line Principal Components Analysis and Canonical Correlation Analysis

ANANT HEGDE AND JOSE C. PRINCIPE
CNEL, ECE Department, University of Florida, Gainesville, FL, USA

DENIZ ERDOGMUS AND UMUT OZERTEM
CSEE Department, Oregon Health & Science University, Portland, OR, USA

YADUNANDANA N. RAO
Motorola, Plantation, FL, USA

HEMANTH PEDDANENI
Neurotronics, Inc, Gainesville, FL, USA

Abstract. Principal components analysis is an important and well-studied subject in statistics and signal processing. Several algorithms for solving this problem exist, and could be mostly grouped into one of the following three approaches: adaptation based on Hebbian updates and deflation, optimization of a second order statistical criterion (like reconstruction error or output variance), and fixed point update rules with deflation. In this study, we propose an alternate approach that avoids deflation and gradient-search techniques. The proposed method is an on-line procedure based on recursively updating the eigenvector and eigenvalue matrices with every new sample such that the estimates approximately track their true values as would be calculated analytically from the current sample estimate of the data covariance matrix. The perturbation technique is theoretically shown to be applicable for recursive canonical correlation analysis, as well. The performance of this algorithm is compared with that of a structurally similar matrix perturbation-based method and also with a few other traditional methods like Sanger's rule and APEX.

Keywords: principal component analysis (PCA), eigenvector, eigenvalue

1. Introduction

Principal component analysis (PCA) is a well-known statistical technique that has been widely applied to solve important signal-processing problems like feature extraction, signal estimation, detection and speech separation [1–4]. Many analytical techniques exist, which can solve PCA once the entire input data is known [5]. However, most of the analytical methods require extensive matrix operations and hence they are unsuited for real-time applications. Further, in many applications such as direction of arrival (DOA) tracking, adaptive subspace estimation, etc., signal statistics change over time rendering

the block methods virtually unacceptable. In such cases, fast, adaptive, on-line solutions are desirable. Majority of the existing algorithms for PCA are based on standard gradient procedures [2, 3, 6–9], which are extremely slow converging, and their performance heavily depends on step-sizes used. To alleviate this, subspace methods have been explored [10–12]. However, many of these subspace techniques are computationally intensive. The recently proposed fixed-point PCA algorithm [13] showed fast convergence with little or no change in complexity compared with gradient methods. However, this method and most of the existing methods in literature rely on using the standard deflation technique, which brings in sequential convergence of principal components that potentially reduces the overall speed of convergence.

We recently explored a simultaneous principal component extraction algorithm called SIPEX [14] which reduced the gradient search only to the space of orthonormal matrices by using Givens rotations. Although SIPEX resulted in fast and simultaneous convergence of all principal components, the algorithm suffered from high computational complexity due to the involved trigonometric function evaluations. A recently proposed alternative approach suggested iterating the eigenvector estimates using a first order matrix perturbation formalism for the sample covariance estimate with every new sample obtained in real time [15]. However, the performance (speed and accuracy) of this algorithm is hindered by the general Toeplitz 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 further restriction of structure, as expected, alleviates the difficulties encountered in the operation of the previous first order perturbation algorithm, resulting in a fast converging and accurate subspace tracking algorithm.

This paper is organized as follows. First, we present a brief definition of the PCA problem to have a self-contained paper. Second, the proposed recursive PCA algorithm (RPCA) is motivated, derived, and extended to non-stationary and complex-valued signal situations. The technique is also illustrated on canonical correlation analysis, which is a generalized eigenvector problem. 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.

2. Problem Definition

As pointed out earlier in the introduction, PCA is a well-defined problem and has been extensively studied in the literature. However, for the sake of completeness, we will provide a brief definition of the problem in this section. Without loss of generality, let us consider that $\mathbf{x}$ is a real-valued zero-mean, n-dimensional random vector. We write its n projections $y_1,...,y_n$ as $y_j = \mathbf{w}_j^T\mathbf{x}$, where $\mathbf{w}_j$'s are n-dimensional unit–norm vectors, corresponding to the projection of $\mathbf{x}$. The first principal component direction is defined as the solution to the following constrained optimization problem, where $\mathbf{R}$ is the input covariance matrix:

$$\mathbf{w}_1 = \arg\max_{\mathbf{w}} \mathbf{w}^T\mathbf{R}\mathbf{w} \text{ subject to } \mathbf{w}^T\mathbf{w} = 1 \qquad (1)$$

By introducing additional constraints wherein the subsequent components are enforced to be orthogonal with the previously discovered ones, we define the subsequent principal components as:

$$\mathbf{w}_j = \arg\max_{\mathbf{w}} \mathbf{w}^T\mathbf{R}\mathbf{w}, \text{s.t. } \mathbf{w}^T\mathbf{w} = 1, \mathbf{w}^T\mathbf{w}_l = 0, l < j \qquad (2)$$

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

However, in many signal processing applications, there are many instances when samples are acquired one at a time and therefore the computations need to be done, real time with just the available samples. Specifically, the solution demands sample-by-sample update rules for the covariance and its eigenvector estimates. In such situation, it is not practical to

update the input covariance estimate and solve a full eigen decomposition problem per sample for an analytical solution. A better alternative lies in utilizing the recursive structure of the covariance estimate to come up with a recursive formula for estimating the eigenvectors. This will be described in the next section.

3. Recursive PCA Description

Suppose, at time k, we have a sequence of n-dimensional zero-mean wide-sense stationary input vectors $\mathbf{x}_k$. The sample covariance estimate at time k for the input vector is[1]

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

Let $\mathbf{R}_k = \mathbf{Q}_k\Lambda_k\mathbf{Q}_k^T$ and $\mathbf{R}_{k-1} = \mathbf{Q}_{k-1}\Lambda_{k-1}\mathbf{Q}_{k-1}^T$, where $\mathbf{Q}$ and $\mathbf{\Lambda}$ denote the orthonormal eigenvector and diagonal eigenvalue matrices, respectively. Also define $\boldsymbol{\alpha}_k = \mathbf{Q}_{k-1}^T\mathbf{x}_k$. Substituting these definitions in Eq. (3), the recursive formula for the eigenvectors and eigenvalues can be written as:

$$\mathbf{Q}_k(k\Lambda_k)\mathbf{Q}_k^T = \mathbf{Q}_{k-1}\left[(k-1)\Lambda_{k-1} + \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T\right]\mathbf{Q}_{k-1}^T \qquad (4)$$

If we denote the eigen decomposition of the matrix $\left[(k-1)\Lambda_{k-1} + \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T\right]$, as $\mathbf{V}_k\mathbf{D}_k\mathbf{V}_k^T$, where $\mathbf{V}$ is orthonormal and $\mathbf{D}$ is diagonal, Eq. (4) becomes

$$\mathbf{Q}_k(k\Lambda_k)\mathbf{Q}_k^T = \mathbf{Q}_{k-1}\mathbf{V}_k\mathbf{D}_k\mathbf{V}_k^T\mathbf{Q}_{k-1}^T \qquad (5)$$

It is easy to see from Eq. (5) that the recursive update rules for the eigenvectors and the eigenvalues turn out to be:

$$\begin{aligned}\mathbf{Q}_k &= \mathbf{Q}_{k-1}\mathbf{V}_k \\ \Lambda_k &= \mathbf{D}_k/k\end{aligned} \qquad (6)$$

In spite of the fact that the matrix $[(k-1)\Lambda_{k-1} + \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T]$ has a special structure much simpler than that of a general covariance matrix, determining the eigendecomposition $\mathbf{V}_k\mathbf{D}_k\mathbf{V}_k^T$ analytically is difficult. As we show in the next section, if k is large, the problem can be solved in a much simpler way using a matrix perturbation analysis approach.

3.1. Perturbation Analysis for Rank-One Update

Consider the matrix $\left[(k-1)\Lambda_{k-1} + \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T\right]$; for large k, this matrix becomes diagonally dominant and therefore its eigenvalues will be close to $(k-1)\mathbf{\Lambda}_{k-1}$ and also, its eigenvectors will also be close to identity (i.e., the eigenvectors of the diagonal portion of the sum).

In summary, the problem reduces to finding the eigendecomposition of a matrix in the form $(\mathbf{\Lambda}+\alpha\alpha^T)$, i.e., a rank-one update on a diagonal matrix $\mathbf{\Lambda}$, using the following approximations: $\mathbf{D}=\Lambda+\mathbf{P}_\Lambda$ and $\mathbf{V}=\mathbf{I}+\mathbf{P}_\mathbf{v}$, where $\mathbf{P}_\mathbf{\Lambda}$ and $\mathbf{P}_\mathbf{V}$ are small perturbation matrices. The eigenvalue perturbation matrix $\mathbf{P}_\mathbf{\Lambda}$ is naturally diagonal. With these definitions, when $\mathbf{VDV}^T$ is expanded, we get

$$\begin{aligned}\mathbf{VDV}^T &= (\mathbf{I}+\mathbf{P}_\mathbf{V})(\Lambda+\mathbf{P}_\Lambda)(\mathbf{I}+\mathbf{P}_\mathbf{V})^T \\ &= \Lambda + \Lambda\mathbf{P}_\mathbf{V}^T + \mathbf{P}_\Lambda + \mathbf{P}_\Lambda\mathbf{P}_\mathbf{V}^T + \mathbf{P}_\mathbf{V}\Lambda \\ &\quad + \mathbf{P}_\mathbf{V}\Lambda\mathbf{P}_\mathbf{V}^T + \mathbf{P}_\mathbf{V}\mathbf{P}_\Lambda + \mathbf{P}_\mathbf{V}\mathbf{P}_\Lambda\mathbf{P}_\mathbf{V}^T \\ &= \Lambda + \mathbf{P}_\Lambda + \mathbf{D}\mathbf{P}_\mathbf{V}^T + \mathbf{P}_\mathbf{V}\mathbf{D} \\ &\quad + \mathbf{P}_\mathbf{V}\Lambda\mathbf{P}_\mathbf{V}^T + \mathbf{P}_\mathbf{V}\mathbf{P}_\Lambda\mathbf{P}_\mathbf{V}^T \qquad (7)\end{aligned}$$

Assuming that the terms $\mathbf{P}_\mathbf{V}\Lambda\mathbf{P}_\mathbf{V}^T$ and $\mathbf{P}_\mathbf{V}\mathbf{P}_\Lambda\mathbf{P}_\mathbf{V}^T$ are negligible, equating Eq. (7) to $(\mathbf{\Lambda}+\alpha\alpha^T)$ yields,

$$\boldsymbol{\alpha}\boldsymbol{\alpha}^T = \mathbf{P}_\Lambda + \mathbf{D}\mathbf{P}_\mathbf{V}^T + \mathbf{P}_\mathbf{V}\mathbf{D} \qquad (8)$$

Knowing that $\mathbf{V}$ is orthonormal and therefore substituting $\mathbf{V}=\mathbf{I}+\mathbf{P}_\mathbf{v}$ in $\mathbf{VV}^T=\mathbf{I}$ and assuming that $\mathbf{P}_\mathbf{V}\mathbf{P}_\mathbf{V}^T \approx \mathbf{0}$, we have $\mathbf{P}_\mathbf{V} = -\mathbf{P}_\mathbf{V}^T$.

Utilizing the fact that $\mathbf{P}_\mathbf{\Lambda}$ and $\mathbf{D}$ are diagonal, the solution for the perturbation matrices are found from Eq. (8) as follows:

$$\left.\begin{aligned}\alpha_i^2 &= (i,i)^{th}\text{element of } \mathbf{P}_\Lambda \\ \frac{\alpha_i\alpha_j}{\lambda_j + \alpha_j^2 - \lambda_i^2 - \alpha_i^2} &= (i,j)^{th}\text{element of } \mathbf{P}_\mathbf{V}, i \neq j \\ 0 &= (i,i)^{th}\text{element of } \mathbf{P}_\mathbf{V}\end{aligned}\right\} \qquad (9)$$

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

3.2. The Recursive PCA 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 subsection.

Selecting the memory depth parameter For stationary cases, we weight all the samples equally and therefore set $\lambda_k = 1/k$ in Eq. (3). In a non-stationary environment, a first-order dynamical forgetting strategy could be employed by selecting a fixed decay rate. Setting $\lambda_k = \lambda$, in Eq. (3) would lead to:

$$\mathbf{R}_k = (1-\lambda)\mathbf{R}_{k-1} + \lambda \mathbf{x}_k \mathbf{x}_k^T \tag{10}$$

where, $\lambda \in (0,1)$ is selected to be very small. Considering that the average memory depth of this recursion is $1/\lambda$ samples, the selection of this parameter presents a trade-off between tracking capability and estimation variance.

Initializing the eigenvectors and the eigenvalues One can simply initialize the estimated eigenvector to identity ($\mathbf{Q}_0=\mathbf{I}$) and the eigenvalues to the sample variances of each input entry over N_0 samples ($\Lambda_0 = \text{diag}\,\mathbf{R}_{N_0}$). We then start the iterations over the samples $k=1,\ldots,N$ and set the memory depth parameter to $\lambda_k = 1/(k-1+\gamma)$. Effectively this corresponds to the following biased (but asymptotically unbiased as $N\to\infty$) covariance estimate:

$$\mathbf{R}_{N,\,\text{biased}} = \frac{N}{N+\gamma}\mathbf{R}_N + \frac{\gamma}{N+\gamma}\Lambda_0 \tag{11}$$

This initialization strategy is utilized in the computer experiments that are presented in the following sections.[2] 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 $\Lambda_0=\text{diag}\mathbf{R}_{N0}$. 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 $(1-\lambda)/\lambda$ is large. Typically, a forgetting factor $\lambda<10^{-2}$ will yield accurate results.

The RPCA algorithm extends to complex-valued signals easily: we employ $\mathbf{P}_\mathbf{V} = -\mathbf{P}_\mathbf{V}^H$ leading to $\alpha_i\alpha_j^*/(\lambda_j + |\alpha_j|^2 - \lambda_i - |\alpha_i|^2)$for off-diagonal entries, and $\mathbf{P}_\mathbf{\Lambda}$ is $|\alpha_i|^2$ on the diagonal.

Table 1. The recursive PCA algorithm outline.

RPCA algorithm summary
1. Initialize $\mathbf{Q}_0$ and $\mathbf{\Lambda}_0$
2. At each time instant k do the following:
a. Get input sample $\mathbf{x}_k$
b. Set memory depth parameter λ_k
c. Calculate $\boldsymbol{\alpha}_k = \mathbf{Q}_{k-1}^T\mathbf{x}_k$
d. Find perturbations $\mathbf{P}_\mathbf{V}$ and $\mathbf{P}_\mathbf{\Lambda}$ corresponding to $(1-\lambda_k)\Lambda_{k-1} + \lambda_k\boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T$
e. Update eigenvector and eigenvalue matrices: $\widetilde{\mathbf{Q}}_k = \mathbf{Q}_{k-1}(\mathbf{I}+\mathbf{P}_\mathbf{V})$ $\widetilde{\Lambda}_k = (1-\lambda_k)\Lambda_{k-1} + \mathbf{P}_\Lambda$
f. Normalize the norms of eigenvector estimates by $\mathbf{Q}_k = \widetilde{\mathbf{Q}}_k\mathbf{T}_k$, where $\mathbf{T}_k$ is a diagonal matrix containing the inverses of the norms of each column of $\widetilde{\mathbf{Q}}_k$
g. Correct eigenvalue estimates by $\Lambda_k = \widetilde{\mathbf{\Lambda}}_k\mathbf{T}_k^{-2}$, where $\mathbf{T}_k^{-2}$ is a diagonal matrix containing the squared norms of the columns of $\widetilde{\mathbf{Q}}_k$

3.3. Recursive Canonical Correlation Analysis

The matrix perturbation analysis can be extended to canonical correlation analysis (CCA) in order to obtain a recursive on-line CCA algorithm similar to the Recursive PCA algorithm. We start by briefly discussing the CCA formulation. Given two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$, the goal of CCA is to find two projections $\mathbf{w}_1$ and $\mathbf{w}_2$ such that the correlation coefficient of $y_1 = \mathbf{w}_1^T\mathbf{x}_1$ and $y_2 = \mathbf{w}_2^T\mathbf{x}_2$ is maximized. This is equivalent to solving the following generalized eigenvector problem $\mathbf{AQ} = \mathbf{BQ}\widetilde{\mathbf{\Lambda}}$, where $\mathbf{Q}$ is the generalized eigenvector matrix and the largest eigenvector $\mathbf{q}_1 = \left[\mathbf{w}_1^T, \mathbf{w}_2^T\right]^T$ and the matrices of interest are defined in terms of the auto- and cross-covariance matrices of the data $\mathbf{x}_1$ and $\mathbf{x}_2$, denoted by $\mathbf{R}_{11}$, $\mathbf{R}_{22}$, $\mathbf{R}_{12}$, and $\mathbf{R}_{21}$ [16]:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{R}_{12} \\ \mathbf{R}_{21} & \mathbf{0} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} \mathbf{R}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{22} \end{bmatrix} \tag{12}$$

Adding $\mathbf{BQ}$ to both sides, and defining $\mathbf{C} = \mathbf{A} + \mathbf{B}$, $\Lambda = \widetilde{\Lambda} + \mathbf{I}$ an equivalent problem is obtained: $\mathbf{CQ}=\mathbf{BQ}\mathbf{\Lambda}$. Notice that $\mathbf{C}$ is the covariance of the concatenated data vector $\mathbf{x} = \left[\mathbf{x}_1^T, \mathbf{x}_2^T\right]^T$, whereas $\mathbf{B}$ is the concatenation of the individual covariance matrices.

In on-line operation, as the samples arrive, we assume that the covariance matrices are updated using the following update rule (with a time-varying

forgetting factor as discussed in RPCA if the data is stationary):

$$\begin{aligned}\mathbf{C}_k &= (1-\lambda)\mathbf{C}_{k-1} + \lambda \mathbf{x}_k \mathbf{x}_k^T \\ \mathbf{B}_k &= (1-\lambda)\mathbf{B}_{k-1} + \lambda \mathbf{U}_k \mathbf{U}_k^{\mathrm{T}} \\ \text{where } \mathbf{U}_k &= \begin{bmatrix} \mathbf{x}_{1k} & \mathbf{0} \\ \mathbf{0} & \mathbf{x}_{2k} \end{bmatrix}\end{aligned} \tag{13}$$

For small λ (i.e., after a large number of samples have been processed in the stationary case), these rank-one and rank-two updates of $\mathbf{C}$ and $\mathbf{B}$ will not change the eigenvectors and the eigenvalues drastically. Therefore, similar to RPCA, we assume that the following update rules for both matrices are valid with $\mathbf{P_V}$ and $\mathbf{P_\Lambda}$ denoting the corresponding perturbations:

$$\mathbf{Q}_k = \mathbf{Q}_{k-1}(\mathbf{I} + \mathbf{P}_{\mathrm{V}})\Lambda_k = \Lambda_{k-1} + \mathbf{P}_\Lambda \tag{14}$$

At all time instants including time k, the eigenvector equation must be satisfied by the eigenvector and eigenvalue estimates: $\mathbf{C}_k\mathbf{Q}_k$=$\mathbf{B}_k\mathbf{Q}_k\mathbf{\Lambda}_k$. Substituting Eqs. (13) and (14) into this equation and exploiting the fact that the equation was satisfied at time instant $k-1$, we obtain the following relationship after some rearranging and cancellation of appropriate terms (specifically higher order terms due to the small perturbation assumption):

$$\begin{aligned}&\mathbf{P_V}\Lambda_{k-1} + \mathbf{P}_\Lambda \\ &\quad = \lambda \mathbf{Q}_{k-1}^T \mathbf{B}_k^{-1} \begin{bmatrix} \mathbf{x}_k \mathbf{x}_k^T \mathbf{Q}_{k-1} \\ -\mathbf{U}_k \mathbf{U}_k^T \mathbf{Q}_{k-1} \Lambda_{k-1} \end{bmatrix}\end{aligned} \tag{15}$$

The inverse of $\mathbf{B}_k$ can be iteratively updated similar to RLS, using the matrix inversion lemma [17]. Since the update at each time instant is of rank-two, the update rule involves inverting a 2×2 matrix, which is still computationally simple. After the application of the inversion lemma on the update of $\mathbf{B}_k$ given in Eq. (13), the update rule for its inverse is obtained as:

$$\mathbf{B}_k^{-1} = \begin{bmatrix} \dfrac{1}{(1-\lambda)}\mathbf{B}_{k-1}^{-1} - \\ \dfrac{1}{(1-\lambda)^2}\mathbf{B}_{k-1}^{-1}\mathbf{U}_k\left(\dfrac{1}{\lambda}\mathbf{I}_{2\times2} + \mathbf{U}_k^T\mathbf{B}_{k-1}^{-1}\mathbf{U}_k\right)^{-1}\mathbf{U}_k^T\mathbf{B}_{k-1}^{-1} \end{bmatrix} \tag{16}$$

Substituting Eq. (16) into Eq. (15), the expression for $\mathbf{P} = \mathbf{P_V}\Lambda_{k-1} + \mathbf{P}_\Lambda$ is obtained in terms of past estimates of the eigenvectors, eigenvalues, and the inverse data covariances (as in RLS). Knowing $\mathbf{P}$ is not sufficient to determine both $\mathbf{P_V}$ and $\mathbf{P_\Lambda}$., we need to utilize the following constraint on the eigenvectors as well: $\mathbf{Q}^T\mathbf{BQ}$=$\mathbf{I}$. Defining $\Gamma_k = \mathbf{Q}_{k-1}^T\mathbf{U}_k$ and substituting Eq. (14) in this constraint as well as the update rule for $\mathbf{B}$ from Eq. (13), after some manipulations the following additional equation restricting the construction of $\mathbf{P_V}$ is obtained:

$$\begin{bmatrix} (1-\lambda)\left(\mathbf{P_V} + \mathbf{P_V}^T\right) + \\ \lambda\left(\Gamma_k\Gamma_k^T\mathbf{P_V} + \mathbf{P_V}^T\Gamma_k\Gamma_k^T\right) = \lambda\left(\Gamma_k\Gamma_k^T - \mathbf{I}\right) \end{bmatrix} \tag{17}$$

Substituting the fact $\mathbf{P_V} = \Lambda_{k-1}^{-1}(\mathbf{P} - \mathbf{P}_\Lambda)$ in Eq. (17), we obtain the following expression from which the diagonal matrix $\mathbf{P_\Lambda}$ can be determined:

$$\begin{aligned}&2(1-\lambda)\mathbf{D} + \mathbf{T}_k\mathbf{D} + \mathbf{D}\mathbf{T}_k = \Sigma \\ &\mathbf{D} = \Lambda_{k-1}^{-1}\mathbf{P}_\Lambda \\ &\mathbf{T}_k = \Gamma_k\Gamma_k^T \\ &\begin{bmatrix} \sum = (1-\lambda)\left(\Lambda_{k-1}^{-1}\mathbf{P} + \mathbf{P}^T\Lambda_{k-1}^{-1}\right) + \\ \lambda\left(\mathbf{T}_k\Lambda_{k-1}^{-1}\mathbf{P} + \mathbf{P}^T\Lambda_{k-1}^{-1}\mathbf{T}_k\right) - \lambda(\mathbf{T}_k - \mathbf{I}) \end{bmatrix}\end{aligned} \tag{18}$$

The iterative steps followed by the perturbation-based recursive CCA algorithm can be summarized as follow: Obtain new data, update $\mathbf{B}_k$ according to Eq. (16), calculate $\mathbf{P}$ using Eq. (15), solve for $\mathbf{D}$ and from that $\mathbf{P_\Lambda}$ using Eq. (18). Finally evaluate $\mathbf{P_V} = \Lambda_{k-1}^{-1}(\mathbf{P} - \mathbf{P}_\Lambda)$. Once $\mathbf{P_V}$ and $\mathbf{P_\Lambda}$ are calculated, the eigenvectors and eigenvalues are updated according to Eq. (14). Both matrices can be initialized appropriately in many ways (similar to those discussed in the RPCA section). The forgetting factor can be made time-varying to utilize all data samples for stationary signals, and the perturbation approximation errors can be made arbitrarily small by selecting smaller values for λ at the cost of longer convergence time.

4. Numerical Experiments

In this section, we perform simulations to demonstrate the capability of our proposed technique to find an on-line PCA solution. As we noted earlier, there already exists a rich literature on estimating PCA

solution. Therefore, we avoid an exhaustive comparison of the proposed method with all the existing algorithms. Instead, we present a comparison with a structurally similar algorithm (which is also based on first order matrix perturbations) [15]. Without going into explicit numerical results, we also comment on the performances of traditional benchmark algorithms like Sanger's rule and APEX in similar setups.

4.1. Convergence Speed Analysis

In the first experimental setup, the goal is to investigate the convergence speed and accuracy of the proposed RPCA algorithm. Note that theoretically the covariance estimator in Eq. (3) converges at a rate of $1/k$ in terms of estimation variance for each entry, while it is always unbiased. In practice, the initial covariance estimate is arbitrarily selected and this leads to a biased estimate as shown in Eq. (11). The bias introduced by this initialization also decays at the rate of $1/k$. Thus, the RPCA algorithm is expected to converge to the true eigendecomposition at a rate of $1/k$ in the case of stationary signals. In the case of nonstationary signals, a constant forgetting factor is introduced in Eq. (10). The expected value of this estimator converges to the true covariance exponentially as $(1-\lambda)^k$. The variances of the estimated $\mathbf{R}$ matrix entries never converge to zero, since this recursion has an exponential memory. However, it decays exponentially at the same rate as the bias to its final value, the fourth order joint moment matrix: $\mathrm{E}[(\mathbf{R}-\mathrm{E}[\mathbf{R}])(\mathbf{R}-\mathrm{E}[\mathbf{R}])^T] \rightarrow \mathrm{E}[\mathbf{x}\mathbf{x}^T\mathbf{x}\mathbf{x}^T]$. The actual convergence rate depends on the actual data distribution, but as an example, for Gaussian $\mathbf{x}$, this final value would be on the order of the square of the covariance of $\mathbf{x}$.

Consider that 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 $\mathbf{A}\mathbf{A}^T$, where $\mathbf{A}$ is an $n \times n$ 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 1,000 simulations (Monte Carlo runs) are carried out for each of the three target eigenvector estimation accuracies (measured in terms of degrees 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 10,000 samples) for these three target accuracies are shown in Fig. 2, where everything above 10,000 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 50 samples. Values in this range were found to work best in terms of final accuracy and convergence speed in extensive Monte Carlo runs.

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,

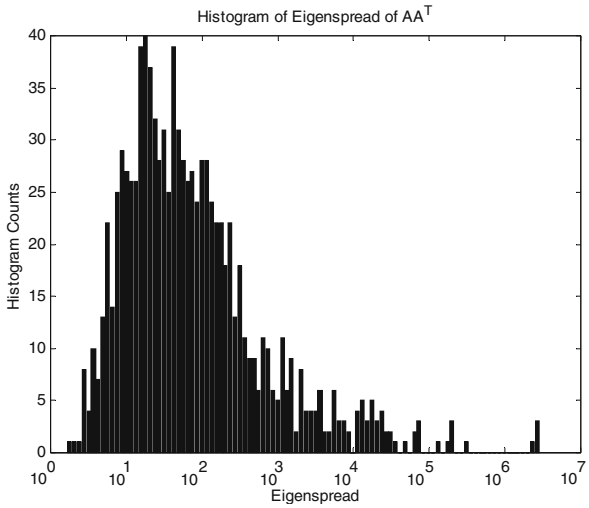


Figure 1. Distribution of eigenspread values for $\mathbf{A}\mathbf{A}^T$, where $\mathbf{A}_{3\times3}$ is generated to have Gaussian distributed random entries.

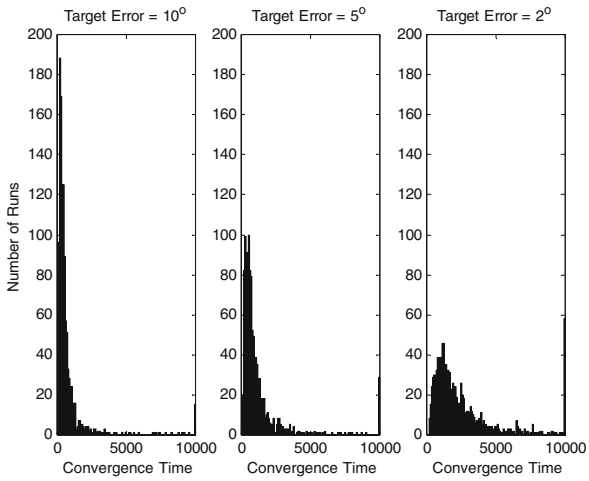


Figure 2. The convergence time histograms for RPCA in the 3-dimensional case for three different target accuracy levels.

about 2% of the simulations failed to converge within 10° in 10,000 on-line iterations, whereas this ratio is about 17% for five dimensions. The failure to converge within the given number of iterations is observed for eigenspreads over 5×10^4.

In a similar setup, Sanger's rule achieves a mean convergence speed of 8,400 iterations with a standard deviation of 2,600 iterations. This results in an average 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°.

4.2. Comparison with 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 performance in terms of overall algorithm performance. To test this hypothesis, an experimental setup similar to the one in Section 4.1 is utilized. This time, however, 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 according to this procedure. The two parameters of the first order perturbation method were set 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 direction estimation convergence curves are shown in Fig. 3.

Often, signal subspace tracking is necessary in signal processing applications dealing with nonstationary signals. To illustrate the performance of RPCA for such 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.

4.3. Direction of Arrival Estimation

Applicability of RPCA, as a subspace method to estimate the directions of arrival in sensor arrays is presented here. In Fig. 5, a sample run from a computer simulation of DOA according to the experimental setup described in [14] is presented to

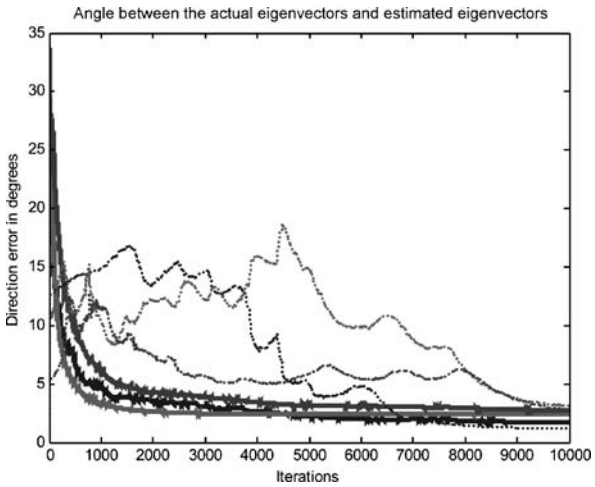


Figure 3. The average eigenvector direction estimation errors versus iterations are shown for the first order perturbation method (*thin dotted lines*) and for RPCA (*thick solid lines*).

illustrate the performance of the complex-valued RPCA algorithm. To provide a benchmark (and an upper limit in convergence speed, we also performed this simulation using Matlab's *eig* function several times on the sample covariance estimate. The latter typically converged to the final accuracy demonstrated here within 10–20 samples. The RPCA estimates on the other hand take a few hundred samples due to the transient in the γ value. The difference is that, while the typical DOA algorithm converts the complex PCA problem into a structured PCA problem with double the number of dimensions, the RPCA algorithm works directly with the complex-valued input vectors to solve the original complex PCA problem.

4.4. An Example with 20 Dimensions

To demonstrate the applicability to higher dimensional situations, an example with 20 dimensions is presented here. In high dimensional situations, the PCA algorithms generally struggle to converge because the interplay between two competing structural properties of the eigenspace. In particular, these two characteristics are the eigenspread ($\max\lambda_i/\min\lambda_i$) and the distribution of ratios of consecutive eigenvalues ($\lambda_n/\lambda_{n-1},...,\lambda_2/\lambda_1$) when they are ordered from largest to smallest (where $\lambda_n>...>\lambda_1$ are the ordered eigenvalues). It is always desirable to have reasonably low or moderate eigen spreads in order to avoid the problem slow convergence, which mainly occurs due to the scarcity of samples representing the minor components. In small dimensional problems, this is typically the dominant issue that controls the convergence speeds of PCA algorithms.

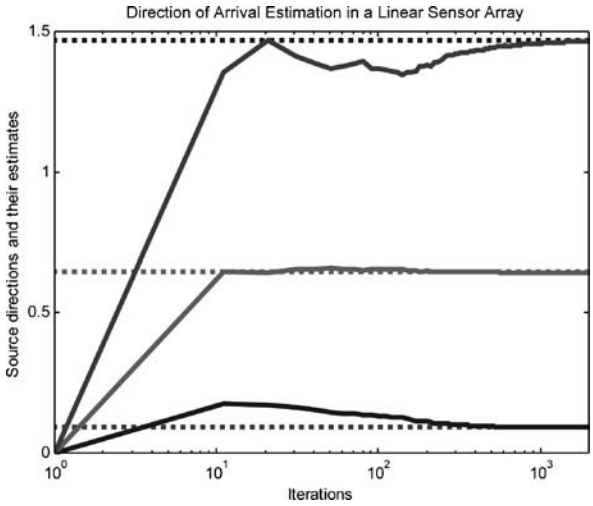


Figure 5. Direction of arrival estimation using complex-valued RPCA in a 3-source 6-sensor case.

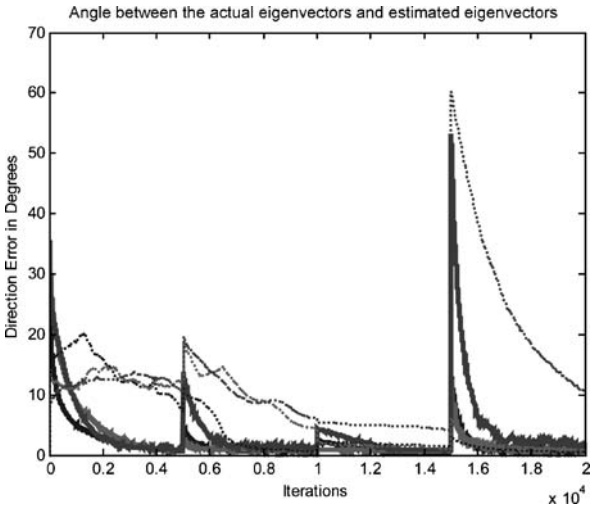


Figure 4. The average eigenvector *director* estimation errors versus iterations for the first order perturbation method (*thin dotted lines*) and for the RPCA (*thick solid lines*) in a piecewise stationary situation are shown. The eigenstructure of the input abruptly changes every 5,000 samples.

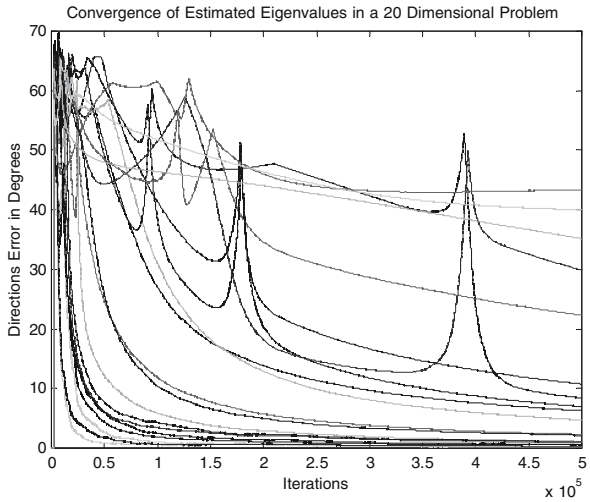


Figure 6. The angle error between the estimated eigenvectors (using RPCA) and their corresponding true eigenvectors in a 20-dimensional PCA problem is shown versus on-line iterations.

On the other hand, as the dimensionality increases, while very large eigenspreads are still undesirable due to the same reason, smaller and previously acceptable eigenspread values too become undesirable because consecutive eigenvalues approach each other. This causes the discriminability of the eigenvectors corresponding to these eigenvalues diminish as their ratio approaches unity. Therefore, the trade-off between small and large eigenspreads becomes significantly difficult. Ideally, the ratios between consecutive eigenvalues must be identical for equal discriminability of all subspace components. Variations from this uniformity will result in faster convergence in some eigenvectors, while others will suffer from almost spherical subspaces indiscriminability.

In Fig. 6, the convergence of the 20 estimated eigenvectors to their corresponding true values is illustrated in terms of the angle between them (in degrees) versus the number of on-line iterations. The data is generated by a 20-dimensional jointly Gaussian distribution with zero-mean, and a covariance matrix with eigenvalues equal to the powers (from 0 to 19) of 1.5 and eigenvectors selected randomly (corresponding to an eigenspread of $1.5^{19} \approx 2{,}217$). This result is typical of higher dimensional cases where major components converge relatively fast and minor components take much longer (in terms of samples and iterations) to reach the same level of accuracy.

5. 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 facilitates 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. Extension of this technique to CCA analysis resulted in obtaining an on-line recursive CCA algorithm. Performance comparisons with traditional algorithms, as well as a structurally similar perturbation-based approach demonstrated the advantages of the recursive PCA algorithm in terms of convergence speed and accuracy.

Acknowledgment

This work is supported by NSF grants ECS-0300340 and ECS-0524835.

Notes

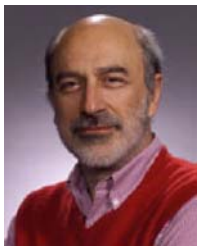
1. In practice, if the samples are not generated by a zero-mean process, a running sample mean estimator could be employed to compensate for this fact. Then this biased estimator can be replaced by the unbiased version and the following derivations can be modified accordingly.
2. A further modification that might be installed is to use a time-varying γ value. In the experiments, we used an exponentially decaying profile for γ,$\gamma = \gamma_0 \exp(-k/\tau)$. This forces the covariance estimation bias to diminish even faster.

References

1. R.O. Duda and P.E. Hart, "Pattern Classification and Scene Analysis," Wiley, New York, 1973.
2. S.Y. Kung, K.I. Diamantaras, and J.S. Taur, "Adaptive Principal Component Extraction (APEX) and Applications," *IEEE Trans. Signal Process.*, vol. 42, no. 5, 1994, pp. 1202–1217.
3. J. Mao and A.K. Jain, "Artificial Neural Networks for Feature Extraction and Multivariate Data Projection," *IEEE Trans. Neural Netw.*, vol. 6, no. 2, 1995, pp. 296–317.
4. Y. Cao, S. Sridharan, and M. Moody, "Multichannel Speech Separation by Eigendecomposition and its Application to Co-Talker Interference Removal," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 3, 1997, pp. 209–219.
5. G. Golub and C.V. Loan, "*Matrix Computation*," Johns Hopkins University Press, Baltimore, Maryland, 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 Netw.*, vol. 2, no. 6, 1989, pp. 459–473.
8. J. Rubner and K. Schulten, "Development of Feature Detectors by Self Organization," *Biol. Cybern.*, vol. 62, 1990, pp. 193–199.
9. J. Rubner and P. Tavan, "A Self Organizing Network for Principal Component Analysis," *Europhys. Lett.*, vol. 10, 1989, pp. 693–698.
10. L. Xu, "Least Mean Square Error Reconstruction Principle for Self-Organizing Neural-Nets," *Neural Netw.*, vol. 6, 1993, pp. 627–648.
11. B. Yang, "Projection Approximation Subspace Tracking," *IEEE Trans. Signal Process.*, vol. 43, no. 1, 1995, pp. 95–107.
12. Y. Hua, Y. Xiang, T. Chen, K. Abed-Meriam, and Y. Miao, "Natural Power Method for Fast Subspace Tracking," in *Proceedings of NNSP'99*, 1999, pp. 176–185.
13. Y.N. Rao and J.C. Principe, "Robust On-line Principal Component Analysis Based on a Fixed-Point Approach," in *Proceedings of ICASSP'02*, vol. 1, 2002, pp. 981–984.

14. D. Erdogmus, Y.N. Rao, K.E. Hild II, and J.C. Principe, "Simultaneous Principal Component Extraction with Application to Adaptive Blind Multiuser Detection," *EURASTP J. Appl. Signal Process.*, vol. 2002, no. 12, 2002, pp. 1473–1484.
15. B. Champagne, "Adaptive Eigendecomposition of Data Covariance Matrices Based on First-Order Perturbations," *IEEE Trans. Signal Process.*, vol. 42, no. 10, 1994, pp. 2758–2770.
16. J. Via, I. Santamaria, and J. Perez, "A Robust RLS Algorithm for Adaptive Canonical Correlation Analysis," in *Proceedings of ICASSP'05*, Philadelphia, Pennsylvania, 2005.
17. S. Haykin, "Adaptive Filter Theory," 4th ed., Prentice Hall, Upper Saddle River, New Jersey, 2001, pp. 231–319.

Anant Hegde received his BE in Electronics & Communication Engineering from Mysore University in 1998. For a brief period, he worked in the personal communications sector at the Bangalore design center in Motorola India Ltd. Later he worked on evoked potentials at the Bio-Signal Analysis Laboratory under the supervision of Dr. Ben Jansen (2000–2002), where he received his MS degree in Electrical Engineering. Currently, he is completing his PhD studies in Electrical Engineering at the University of Florida. His research at the Computational Neuroengineering Laboratory involves investigating the functional connectivity between brain areas at various states of epileptic seizures. His research interests are in biomedical signal processing, broad band communication technologies and machine learning networks adapted from information theoretic learning principles.



Jose C. Principe is Distinguished Professor of Electrical and Biomedical Engineering at the University of Florida since 2002. He joined the University of Florida in 1987, after an eight-year appointment as Professor at the University of Aveiro, in Portugal. Dr. Principe holds degrees in electrical engineering from the University of Porto (Bachelors), Portugal, University of Florida (Master and Ph.D.), USA and a Laurea Honoris Causa degree from the Universita Mediterranea in Reggio Calabria, Italy. Dr. Principe interests lie in nonlinear non-Gaussian optimal signal processing and modeling and in biomedical engineering. He created in 1991 the Computational NeuroEngineering Laboratory to synergistically focus the research in biological information processing models.

Dr. Principe is a Fellow of the IEEE, past President of the International Neural Network Society, and Editor in Chief of the Transactions of Biomedical Engineering since 2001, as well as a former member of the Advisory Science Board of the FDA. He holds 5 patents and has submitted seven more. Dr. Principe was supervisory committee chair of 47 Ph.D. and 61 Master students, and he is author of more than 400 refereed publications (3 books, 4 edited books, 14 book chapters, 116 journal papers and 276 conference proceedings).

Deniz Erdogmus received the B.S. in Electrical & Electronics Engineering (EEE), and the B.S. in Mathematics both in 1997, and the M.S. in EEE in 1999 from the Middle East Technical University, Turkey. He received his PhD in Electrical & Computer Engineering from the University of Florida (UF) in 2002. He worked as a research engineer at TUBITAK-SAGE, Turkey from 1997 to 1999, focusing on the design of navigation, guidance, and flight control systems. He was also a research assistant and a postdoctoral research associate at UF from 1999 to 2004, concentrating on signal processing, adaptive systems, machine learning, and information theory, specifically with applications in biomedical engineering including brain machine interfaces. Currently, he is holding an Assistant Professor position jointly at the Computer Science and Electrical Engineering Department and the Biomedical Engineering Department of the Oregon Health and Science University. His research focuses on information theoretic adaptive signal processing and its applications to biomedical signal processing problems. Dr. Erdogmus has over 35 articles in international scientific journals and numerous conference papers and book chapters. He has also served as associate editor and guest editor for various journals, participated in various conference organization and scientific committees, and he is a member of Tau Beta Pi, Eta Kappa Nu, IEEE, and IEE. He was the recipient of the IEEE-SPS 2003 Best Young Author Paper Award and 2004 INNS Young Investigator Award.

Umut Ozertem received his BS in Electrical & Electronics Engineering from the Middle East Technical University in 2003. After working at Tubitak-BILTEN between August 2003 and July 2004 under the supervision of A. Aydin Alatan, he joined the Adaptive Systems Lab in the CSEE Department of the Oregon Health & Science University as a PhD student. He is currently working with Deniz Erdogmus, and his research interests include adaptive and statistical signal processing, information theory and its applications to signal processing and machine learning.

Yadunandana N. Rao was born in Mysore, India. He received his BE in Electronics and Communication Engineering from the University of Mysore, India in 1997 and MS and PhD in Electrical and Computer Engineering from the University of Florida, Gainesville, FL in 2000 and 2004, respectively. Between August 1997 and July 1998, he worked a Software Engineer in Bangalore, India. From May 2000 to January 2001, he was a Design Engineer at GE Medical Systems, WI. He is currently with Motorola, Inc. His research interests include adaptive signal processing theory, algorithms and analysis, neural networks for signal processing, communications, and biomedical applications.

Hemanth Peddaneni obtained his M.S. degree in Electrical and Computer Engineering from the University of Florida in 2004. Since then, he has been working as a Research and Development Engineer with Neurotronics Inc.

His research interests include biological signal processing with particular emphasis on sleep analysis, adaptive signal processing, wavelet methods for time series analysis and digital filter design/implementation.