BI-ITERATIVE LEAST SQUARE VERSUS BI-ITERATIVE SINGULAR VALUE DECOMPOSITION FOR SUBSPACE TRACKING

Shan Ouyang and Yingbo Hua

Department of Electrical Engineering
University of California, Riverside
Riverside, CA, 92521, USA

ABSTRACT

We first revisit the problem of optimal low-rank matrix approximation, from which a bi-iterative least square (Bi-LS) method is formulated. We then show that the Bi-LS method is a natural platform for developing subspace tracking algorithms. Comparing to the well known bi-iterative singular value decomposition (Bi-SVD) method, we demonstrate that the Bi-LS method leads to much simpler (and yet equally accurate) linear complexity algorithms for subspace tracking. This gain of simplicity is a surprising result while the reason behind it is also surprisingly simple as shown in this paper.

1. INTRODUCTION

Subspace has played an important role in many areas of modern signal processing. Although subspace can be obtained in theory via the eigenvalue decomposition (EVD) of a covariance matrix or the singular value decomposition (SVD) of the corresponding data matrix, the cost of directly computing EVD or SVD can be prohibitively high in real-time applications where the data dimension is high. Therefore, there is a practical need for efficient subspace tracking techniques that yield good estimates of the desired subspace.

A number of subspace tracking algorithms have been proposed in the past, see [3], [4], [8], [11], and references therein. Among them, a family of power-based subspace algorithms have the important features: low complexity and fast convergence. The power family includes the Oja algorithm [10], the PAST algorithm [14], the NIC algorithm [9], and the Bi-SVD algorithm [13]. The principle behind the Bi-SVD algorithm has recently received a renewed interest as shown in [2]. Bi-SVD stands for bi-iterative singular value decomposition, which is also known as QR based bi-orthogonal iterations for computation of SVD [12].

In this paper, we show that Bi-SVD is not the best framework for formulating efficient subspace tracking algorithms, and a better framework comes from bi-iterative least square (Bi-LS) method. The Bi-LS method is a simple variation of the alternating power method for computing optimal reduced rank filters [6], [7].

2. THE BI-LS METHOD FOR OPTIMAL LOW-RANK MATRIX APPROXIMATION

It is well known that given a data matrix $\boldsymbol{X} \in \mathbb{C}^{L\times N}$, their $r \leq \min(N, L)$ dominant singular values and vectors can be computed by the Bi-SVD method as listed in Table 1. Under a mild condition, the columns of $\boldsymbol{Q}_A(k) \in \mathbb{C}^{L\times r}$ converge to the r dominant left singular vectors, the columns of $\boldsymbol{Q}_B(k) \in \mathbb{C}^{N\times r}$ converge to the r dominant right singular vectors, and both $\boldsymbol{R}_A(k)$ and $\boldsymbol{R}_B(k)$ converge to the $r \times r$ diagonal matrix of the dominant singular values of $\boldsymbol{X}$.

We now examine the optimal low-rank approximation of $\boldsymbol{X}$. It is known that the optimal low-rank approximation can be obtained by minimizing the following cost function:

$$\boldsymbol{J}(\boldsymbol{A}, \boldsymbol{B}) = \|\boldsymbol{X} - \boldsymbol{A}\boldsymbol{B}^H\|_F^2 \tag{1}$$

By minimizing the above function with respect to $\boldsymbol{A}$ and $\boldsymbol{B}$ alternatingly, we have the following algorithm:

$$\begin{cases} \boldsymbol{A}(k) = \boldsymbol{X}\boldsymbol{B}(k-1)\boldsymbol{G}(k-1) \\ \boldsymbol{B}(k) = \boldsymbol{X}^H\boldsymbol{A}(k)\big(\boldsymbol{A}^H(k)\boldsymbol{A}(k)\big)^{-1} \end{cases} \tag{2}$$

where $\boldsymbol{G}(k-1) = (\boldsymbol{B}^H(k-1)\boldsymbol{B}(k-1))^{-1}$. It can be shown [6], [7] that with a weak condition on the initial matrix $\boldsymbol{B}(0)$ and the data matrix $\boldsymbol{X}$, the product of $\boldsymbol{A}(k)\boldsymbol{B}^H(k)$ from (2) globally and exponentially converges to the optimal rank-r approximation $\boldsymbol{X}_r$ of $\boldsymbol{X}$.

Let us write the QR decomposition of $\boldsymbol{A}(k)$ and $\boldsymbol{B}(k)$ as

$$\begin{cases} \boldsymbol{A}(k) = \boldsymbol{Q}_A(k)\boldsymbol{R}_A(k) \\ \boldsymbol{B}(k) = \boldsymbol{Q}_B(k)\boldsymbol{R}_B(k). \end{cases} \tag{3}$$

Substituting $\boldsymbol{B}(k-1) = \boldsymbol{Q}_B(k-1)\boldsymbol{R}_B(k-1)$ and $\boldsymbol{A}(k) = \boldsymbol{Q}_A(k)\boldsymbol{R}_A(k)$ into the right side of (2) yields

$$\begin{cases} \boldsymbol{A}(k) = \boldsymbol{X}\boldsymbol{Q}_B(k-1) \\ \boldsymbol{B}(k) = \boldsymbol{X}^H\boldsymbol{Q}_A(k)\boldsymbol{R}_A^{-H}(k) \end{cases} \tag{4}$$

where for simplicity, the choice of $\boldsymbol{G}(k-1) = \boldsymbol{R}_B^{-1}(k-1)$ is used without affecting the global convergence property [7]. The above algorithm is named the Bi-LS method and summarized in Table 2.

A unique property of the Bi-LS method is that from Table 2,

$$\begin{aligned} \boldsymbol{Q}_B^H(k-1)\boldsymbol{B}(k) &= \boldsymbol{Q}_B^H(k-1)\boldsymbol{X}^H\boldsymbol{Q}_A(k)\boldsymbol{R}_A^{-H}(k) \\ &= \boldsymbol{A}^H(k)\boldsymbol{Q}_A(k)\boldsymbol{R}_A^{-H}(k) \\ &= \boldsymbol{I}. \end{aligned} \tag{5}$$

This equation together with the second part of (3) implies that

$$\boldsymbol{Q}_B^H(k-1)\boldsymbol{Q}_B(k) = \boldsymbol{R}_B^{-1}(k). \tag{6}$$

This work was supported in part by NSF

Table 1. Bi-SVD Method for SVD Computation

Initialization: $\boldsymbol{Q}_B(0) = \begin{bmatrix} \boldsymbol{I}_r \\ \boldsymbol{O} \end{bmatrix}$

For $k = 1, 2, \cdots$ until convergence Do:

First Step:

$\boldsymbol{A}(k) = \boldsymbol{X}\boldsymbol{Q}_B(k-1)$

$\boldsymbol{A}(k) = \boldsymbol{Q}_A(k)\boldsymbol{R}_A(k)$ skinny QR decomposition

Second Step:

$\boldsymbol{B}(k) = \boldsymbol{X}^H\boldsymbol{Q}_A(k)$

$\boldsymbol{B}(k) = \boldsymbol{Q}_B(k)\boldsymbol{R}_B(k)$ skinny QR decomposition

Hence, for the Bi-LS method, the upper-triangular matrix $\boldsymbol{R}_B(k)$ becomes the identity matrix at convergence.

For the Bi-LS method, we can show that if $\boldsymbol{Q}_B(k-1)$ is right multiplied by a unitary matrix, the corresponding $\boldsymbol{B}(k)$ is also right multiplied by the same unitary matrix. When $\boldsymbol{R}_B(k)$ is identity, the corresponding $\boldsymbol{Q}_B(k)$ remains the same as $\boldsymbol{Q}_B(k-1)$ right multiplied by the same unitary matrix. But for the Bi-SVD method, if $\boldsymbol{Q}_B(k-1)$ is right multiplied by a unitary matrix, the corresponding $\boldsymbol{Q}_B(k)$ is always altered by a different unitary matrix (even at convergence). It is the above property that makes the linear complexity version of the Bi-LS method much simpler than the counter part of the Bi-SVD method. The details of a derivation of the fast Bi-LS algorithm are shown in the next section.

For the Bi-LS method, $\boldsymbol{Q}_A(k)$ converges to a product of the matrix of the left principal singular vectors and an $r \times r$ unitary rotation matrix, and a similar property holds for $\boldsymbol{Q}_B(k)$. Each of the two unknown rotation matrices, when desired, can be computed from the SVD of the $r \times r$ upper-triangular matrix $\boldsymbol{R}_A(k)$. This additional computation does not affect the complexity order of the Bi-LS method provided that r is much smaller than $\min(N, L)$.

3. FAST BI-LS ALGORITHM FOR SUBSPACE TRACKING

We assume that the N-dimensional input vector $\boldsymbol{x}(t)$ is available at time t. To track time-varying subspaces, a data window should be applied at any time. The exponential and sliding windows are two common windows. The sliding window is known to be more suitable for suddenly varying signals than the exponential window. In this paper, we consider a hybrid (sliding exponential) window.

The hybrid window is represented by the following scheme of data matrix update:

$$\begin{bmatrix} \boldsymbol{X}(t) \\ \alpha^{L/2}\beta^{1/2}\boldsymbol{x}^H(t-L) \end{bmatrix} = \begin{bmatrix} \beta^{1/2}\boldsymbol{x}^H(t) \\ \alpha^{1/2}\boldsymbol{X}(t-1) \end{bmatrix} \quad (7)$$

where $0 < \beta \leq 1$ and $0 < \alpha \leq 1$. Now, replace the iteration index k in Table 2 by the discrete time index t to yield the sequential Bi-LS subspace algorithm. Postmultiplying both sides of (7) by $\boldsymbol{Q}_B(t-1)$ and introducing a compressed data vector $\boldsymbol{h}(t) = \boldsymbol{Q}_B^H(t-1)\boldsymbol{x}(t)$, we have

$$\begin{bmatrix} \boldsymbol{A}(t) \\ \boldsymbol{h}_L^H(t) \end{bmatrix} = \begin{bmatrix} \beta^{1/2}\boldsymbol{h}^H(t) \\ \alpha^{1/2}\boldsymbol{X}(t-1)\boldsymbol{Q}_B(t-1) \end{bmatrix} \quad (8)$$

where $\boldsymbol{h}_L(t) \triangleq \alpha^{L/2}\beta^{1/2}\boldsymbol{Q}_B^H(t-1)\boldsymbol{x}(t-L)$.

Table 2. Bi-LS Method for Optimal Low-Rank Matrix Approximation

Initialization: $\boldsymbol{Q}_B(0) = \begin{bmatrix} \boldsymbol{I}_r \\ \boldsymbol{O} \end{bmatrix}$

For $k = 1, 2, \cdots$ until convergence Do:

First Step:

$\boldsymbol{A}(k) = \boldsymbol{X}\boldsymbol{Q}_B(k-1)$

$\boldsymbol{A}(k) = \boldsymbol{Q}_A(k)\boldsymbol{R}_A(k)$ skinny QR decomposition

Second Step:

$\boldsymbol{B}(k) = \boldsymbol{X}^H\boldsymbol{Q}_A(k)\boldsymbol{R}_A^{-H}(k)$

$\boldsymbol{B}(k) = \boldsymbol{Q}_B(k)\boldsymbol{R}_B(k)$ skinny QR decomposition

A key step to developing fast subspace tracking algorithms is to introduce a low-rank approximation to $\boldsymbol{X}(t)$. From Table 2, we have

$$\boldsymbol{X}(t)\boldsymbol{Q}_B(t-1) = \boldsymbol{Q}_A(t)\boldsymbol{R}_A(t). \quad (9)$$

Thus, we obtain

$$\boldsymbol{X}(t-1)\boldsymbol{Q}_B(t-1) \simeq \boldsymbol{Q}_A(t-1)\boldsymbol{R}_A(t-1)\boldsymbol{R}_B^{-1}(t-1). \quad (10)$$

Note that (6) has been employed above. Since both $\boldsymbol{R}_A(t-1)$ and $\boldsymbol{R}_B^{-1}(t-1)$ are upper-triangular matrices, the product of $\boldsymbol{R}_A(t-1)$ and $\boldsymbol{R}_B^{-1}(t-1)$ is still an upper-triangular matrix. This implies that $\boldsymbol{R}_B^{-1}(t-1)$ does not affect the subspace of $\boldsymbol{Q}_A(t-1)$. Furthermore, according to the earlier discussion, $\boldsymbol{R}_B(t)$ can be approximated by an identity matrix. Therefore, (10) can be further simplified as follows:

$$\boldsymbol{X}(t-1)\boldsymbol{Q}_B(t-1) \simeq \boldsymbol{Q}_A(t-1)\boldsymbol{R}_A(t-1). \quad (11)$$

Thus, substituting (11) into (8) yields the following update:

$$\begin{bmatrix} \boldsymbol{A}(t) \\ \boldsymbol{h}_L^H(t) \end{bmatrix} \simeq \begin{bmatrix} \beta^{1/2}\boldsymbol{h}^H(t) \\ \alpha^{1/2}\boldsymbol{Q}_A(t-1)\boldsymbol{R}_A(t-1) \end{bmatrix}. \quad (12)$$

Taking the notation

$$\widetilde{\boldsymbol{Q}}_A(t-1) = \begin{bmatrix} \boldsymbol{0}^T & 1 \\ \boldsymbol{I}_{\boldsymbol{L-1}} & \boldsymbol{0} \end{bmatrix} \boldsymbol{Q}_A(t-1) \quad (13)$$

and letting the L dimensional vector $\boldsymbol{z} = [1, 0 \cdots 0]^T$, $\boldsymbol{A}(t)$ can be computed recursively by

$$\boldsymbol{A}(t) \simeq \alpha^{1/2}\widetilde{\boldsymbol{Q}}_A(t-1)\boldsymbol{R}_A(t-1) + \beta^{1/2}\boldsymbol{z}\widetilde{\boldsymbol{h}}^H(t) \quad (14)$$

where $\widetilde{\boldsymbol{h}}(t) = \boldsymbol{h}(t) - (\alpha/\beta)^{1/2}\boldsymbol{R}_A^H(t-1)\boldsymbol{q}_{A_L}(t-1)$ and $\boldsymbol{q}_{A_L}(t-1) = \widetilde{Q}_A^H(t-1)\boldsymbol{z}$. Decompose the vector $\boldsymbol{z}$ into two vectors orthogonal to each other:

$$\boldsymbol{z} = \boldsymbol{z}_\perp(t) + \widetilde{\boldsymbol{Q}}_A(t-1)\boldsymbol{q}_{A_L}(t-1). \quad (15)$$

Substituting (15) into (14) yields

$$\boldsymbol{A}(t) \simeq \left[\widetilde{\boldsymbol{Q}}_A(t-1) \quad \frac{\boldsymbol{z}_\perp(t)}{\|\boldsymbol{z}_\perp(t)\|}\right] \boldsymbol{G}_A^H(t) \begin{bmatrix} \boldsymbol{R}_A(t) \\ 0 \ldots 0 \end{bmatrix} \quad (16)$$

where $\|\boldsymbol{z}_\perp(t)\|$ denotes the norm of the vector $\boldsymbol{z}_\perp(t)$, $\boldsymbol{G}_A(t)$ is a sequence of orthonormal Givens plane rotations and $\boldsymbol{R}_A(t)$ is an

$r \times r$ upper-triangular matrix that satisfies

$$\begin{bmatrix} \boldsymbol{R}_A(t) \\ 0 \dots 0 \end{bmatrix} = \boldsymbol{G}_A(t) \begin{bmatrix} \alpha^{1/2}\boldsymbol{R}_A(t-1) + \beta^{1/2}\boldsymbol{q}_{A_L}(t-1)\widetilde{\boldsymbol{h}}^H(t) \\ \beta^{1/2}\|\boldsymbol{z}_\perp(t)\|\widetilde{\boldsymbol{h}}^H(t) \end{bmatrix}. \quad (17)$$

Clearly, (16) suggests a QR decomposition of $\boldsymbol{A}(t)$. The desired orthonormal matrix $\boldsymbol{Q_A}(t)$ can be obtained from the following recursion:

$$[\boldsymbol{Q_A}(t) \quad \star] = \left[\widetilde{\boldsymbol{Q}}_A(t-1) \quad \frac{\boldsymbol{z}_\perp(t)}{\|\boldsymbol{z}_\perp(t)\|}\right] \boldsymbol{G_A^H}(t) \quad (18)$$

where the symbol $\star$ denotes a column vector of no interst.

Note that (17) involves a special updating problem of form "upper-triangular plus rank one". This special updating problem can be solved simply by $2r$ Givens plane rotations [5]. Thus, (18) can be recursively computed and requires only $8Lr$ flops (a flop stands for a multiplication and an addition).

To update $\boldsymbol{Q}_B(t)$ recursively, we now consider the following identity:

$$\begin{aligned} &\left[\boldsymbol{X}^H(t) \quad \alpha^{L/2}\beta^{1/2}\boldsymbol{x}(t-L)\right] \begin{bmatrix} \boldsymbol{Q}_A(t)\boldsymbol{R}_A^{-H}(t) \\ 0 \dots 0 \end{bmatrix} \\ &= \left[\beta^{1/2}\boldsymbol{x}(t) \quad \alpha^{1/2}\boldsymbol{X}^H(t-1)\right] \begin{bmatrix} \boldsymbol{Q}_A(t)\boldsymbol{R}_A^{-H}(t) \\ 0 \dots 0 \end{bmatrix}. \end{aligned} \quad (19)$$

According to the second step in Table 2, we get

$$\boldsymbol{B}(t) = \left[\beta^{1/2}\boldsymbol{x}(t) \quad \alpha^{1/2}\boldsymbol{X}^H(t-1)\right] \begin{bmatrix} \boldsymbol{Q}_A(t)\boldsymbol{R}_A^{-H}(t) \\ 0 \dots 0 \end{bmatrix}. \quad (20)$$

Premultiplying both sides of (20) by $\boldsymbol{Q}_B^H(t-1)$ and employing (8) yields

$$\begin{aligned} \boldsymbol{Q}_B^H(t-1)\boldsymbol{B}(t) &= \boldsymbol{A}^H(t)\boldsymbol{Q}_A(t)\boldsymbol{R}_A^{-H}(t) \\ &= \boldsymbol{I}_r. \end{aligned} \quad (21)$$

We decompose the vector $\boldsymbol{x}(t)$ to two vectors orthogonal to each other:

$$\boldsymbol{x}(t) = \boldsymbol{x}_\perp(t) + \boldsymbol{Q}_B(t-1)\boldsymbol{h}(t). \quad (22)$$

Substituting the representation $\boldsymbol{x}(t)$ and the low-rank approximation (11) into (20), after proper manipulations,yields

$$\boldsymbol{B}(t) \simeq \left[\boldsymbol{Q}_B(t-1) \quad \frac{\boldsymbol{x}_\perp(t)}{\|\boldsymbol{x}_\perp(t)\|}\right] \boldsymbol{G}_B^H(t) \begin{bmatrix} \boldsymbol{R}_B(t) \\ 0 \cdots 0 \end{bmatrix} \quad (23)$$

where

$$\begin{bmatrix} \boldsymbol{R}_B(t) \\ 0 \cdots 0 \end{bmatrix} = \boldsymbol{G}_B(t) \begin{bmatrix} \boldsymbol{I}_r \\ \beta^{1/2}\|\boldsymbol{x}_\perp(t)\|\boldsymbol{q}_{A_1}^H(t)\boldsymbol{R}_A^{-H}(t) \end{bmatrix} \quad (24)$$

where $\|\boldsymbol{x}_\perp(t)\|$ denotes the norm of the vector $\boldsymbol{x}_\perp(t)$,and $\boldsymbol{q}_{A_1}(t) \triangleq \boldsymbol{Q}_A^H(t)\boldsymbol{z}$. The desired orthonormal matrix $\boldsymbol{Q}_B(t)$ can be updated recursively from the following:

$$[\boldsymbol{Q}_B(t) \quad \star] = \left[\boldsymbol{Q}_B(t-1) \quad \frac{\boldsymbol{x}_\perp(t)}{\|\boldsymbol{x}_\perp(t)\|}\right] \boldsymbol{G}_B^H(t). \quad (25)$$

Table 3. A Bi-LS Subspace Tracking Algorithm .

Initialization: $\boldsymbol{z} = [1, 0 \dots 0]^T$; $\boldsymbol{Q_B}(0) = \begin{bmatrix} \boldsymbol{I}_r \\ \boldsymbol{O} \end{bmatrix}$;

$\boldsymbol{Q_A}(0) = \begin{bmatrix} \boldsymbol{I}_r \\ \boldsymbol{O} \end{bmatrix}$; $\boldsymbol{R_A}(0) = \boldsymbol{I}_r$

FOR each t, Do:

Input: $\boldsymbol{x}(t)$

First Step:

$\widetilde{\boldsymbol{Q}}_A(t-1) = \begin{bmatrix} \boldsymbol{0}^T & 1 \\ \boldsymbol{I}_{L-1} & \boldsymbol{0} \end{bmatrix} \boldsymbol{Q}_A(t-1)$

$\boldsymbol{q}_{A_L}(t-1) = \widetilde{\boldsymbol{Q}}_A^H(t-1)\boldsymbol{z}$

$\boldsymbol{h}(t) = \boldsymbol{Q}_B^H(t-1)\boldsymbol{x}(t)$

$\widetilde{\boldsymbol{h}}(t) = \boldsymbol{h}(t) - (\alpha/\beta)^{1/2}\boldsymbol{R}_A^H(t-1)\boldsymbol{q}_{A_L}(t-1)$

$\boldsymbol{z}_\perp(t) = \boldsymbol{z} - \widetilde{\boldsymbol{Q}}_A(t-1)\boldsymbol{q}_{A_L}(t-1)$

$\begin{bmatrix} \boldsymbol{R}_A(t) \\ 0 \dots 0 \end{bmatrix} =$

$\boldsymbol{G}_A(t) \begin{bmatrix} \alpha^{1/2}\boldsymbol{R}_A(t-1) + \beta^{1/2}\boldsymbol{q}_{A_L}(t-1)\widetilde{\boldsymbol{h}}^H(t) \\ \beta^{1/2}\|\boldsymbol{z}_\perp(t)\|\widetilde{\boldsymbol{h}}^H(t) \end{bmatrix}$

$[\boldsymbol{Q}_A(t) \quad \star] = \left[\widetilde{\boldsymbol{Q}}_A(t-1) \quad \frac{\boldsymbol{z}_\perp(t)}{\|\boldsymbol{z}_\perp(t)\|}\right] \boldsymbol{G}_A^H(t)$

Second Step:

$\boldsymbol{q}_{A_1}(t) = \boldsymbol{Q}_A^H(t)\boldsymbol{z}$

$\boldsymbol{x}_\perp(t) = \boldsymbol{x}(t) - \boldsymbol{Q}_B(t-1)\boldsymbol{h}(t)$

$\boldsymbol{R}_A(t)\widetilde{\boldsymbol{q}}_{A_1}(t) = \boldsymbol{q}_{A_1}(t) \xrightarrow{\text{back substitution}} \widetilde{\boldsymbol{q}}_{A_1}(t)$

$\begin{bmatrix} \boldsymbol{R}_B(t) \\ 0 \cdots 0 \end{bmatrix} = \boldsymbol{G}_B(t) \begin{bmatrix} \boldsymbol{I}_r \\ \beta^{1/2}\|\boldsymbol{x}_\perp(t)\|\widetilde{\boldsymbol{q}}_{A_1}^H(t) \end{bmatrix}$

$[\boldsymbol{Q}_B(t) \quad \star] = \left[\boldsymbol{Q}_B(t-1) \quad \frac{\boldsymbol{x}_\perp(t)}{\|\boldsymbol{x}_\perp(t)\|}\right] \boldsymbol{G}_B^H(t)$

This updating operation requires $4Nr$ flops. In addition, let

$$\boldsymbol{q}_{A_1}(t) = \boldsymbol{R}_A(t)\widetilde{\boldsymbol{q}}_{A_1}(t). \quad (26)$$

Since $\boldsymbol{R}_A(t)$ is the upper-triangular matrix, the vector $\widetilde{\boldsymbol{q}}_{A_1}(t)$ is easy to be solved only by $O(r^2)$ backsubstitution operations.

Finally, a complete quasicode of the proposed Bi-LS subspace tracking algorithm is summarized in Table 3. The new algorithm has a principal computational complexity of $6Nr + 9Lr + O(r^2)$ flops per each iteration. This Bi-LS algorithm is more efficient in computations than the counter part of the Bi-SVD algorithm, which is the sliding window adaptive SVD (SWASVD3) algorithm [2] that has a principal computational complexity of $27Nr + 13Lr + O(r^2)$ flops. The Bi-SVD-1 algorithm shown in [13] uses an exponential window and yields the right subspace only (i.e., $\boldsymbol{Q}_B(k)$). The corresponding version of the Bi-LS method is also more efficient but of the same accuracy. Due to space limitation, we can not include more details.

4. PERFORMANCE EVALUATION

We now illustrate the performance of the Bi-LS algorithm for subspace tracking by simulations. The data vector consists of two independent complex sinusoidal sources plus a complex white Gaussian noise [2]. The frequencies of two sources change abruptly at different time instants (one at $t = 200$, and another at $t = 400$). The estimated frequencies are obtained by the ESPRIT/MatrixPencil method. The accuracy of the estimated subspace is measured by

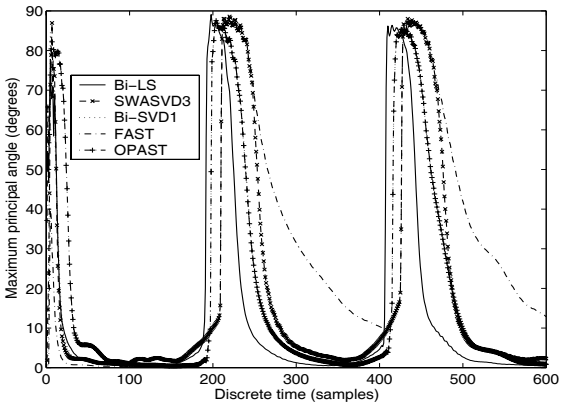

Fig. 1. Subspace tracking performance of several orthonormal subspace algorithms. The curves of Bi-SVD-1 and OPAST overlap each other.

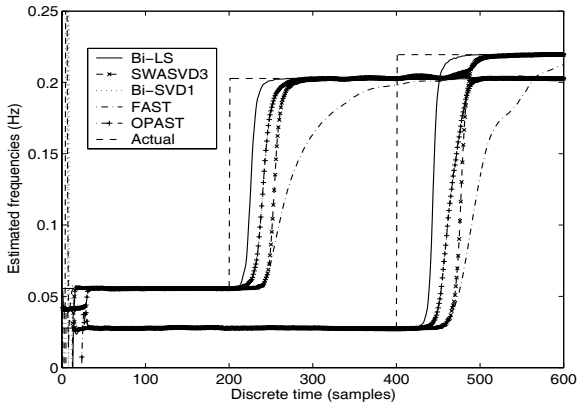

Fig. 2. Frequency tracking performance of several orthonormal subspace algorithms. The curves of Bi-SVD-1 and OPAST overlap each other.

the maximum principal angle [5] between the estimate and the exact. The exact subspace is defined to be the subspace computed by the exact SVD of the same data matrix at each time.

Fig.1 shows the subspace tracking performance of the five algorithms: Bi-LS, SWASVD3 [2], Bi-SVD-1 [13], FAST [11], and OPAST [1]. Among them, the SWASVD3 and the FAST are based on sliding rectangular window, whereas the Bi-SVD-1 and OPAST are based on exponential window. The Bi-SVD-1 and OPAST have the same performance in accuracy. All these algorithms produce orthonormal subspace basis vectors. Except FAST, all algorithms are power-based and have the same tracking performance although the underlying data window affects the tracking pattern. The parameters used in this test are $N = 80, r = 2, L = 100, \alpha = 0.98$, SNR=5.7dB.

5. CONCLUSION

We have presented a new method for developing subspace tracking algorithms. This new method called Bi-LS method contrasts against the Bi-SVD method that has been applied by others. We have shown that for subspace tracking, the Bi-LS method leads to simpler algorithms than the Bi-SVD method while the tracking accuracy is the same (provided that the same data window is considered). Due to lack of space, several simpler versions of the Bi-LS method (with the same accuracy of subspace tracking but relaxed requirement on orthonormality and right versus left subspaces) are not included but will be reported in a upcoming full paper.

6. REFERENCES

[1] K. Abed-Meraim, A. Chkeif, and Y. Hua, "Fast orthonormal PAST algorithm," *IEEE Signal Processing Letters*, vol.7, no.3, pp.60-62, 2000.

[2] R. Badeau, G. Richard, and B. David, "Sliding window adaptive SVD algorithms," *to appear in IEEE Trans. Signal Processing*.

[3] P. Comon and G. H. Golub, "Tracking a few extreme singular values and vectors in signal processing," *Proc. IEEE*, vol.78, no.8, pp.1327-1343, 1990.

[4] R. D. DeGroat, E. M. Dowling, and A. D. Linebarger, "Subspace tracking," *Digital Signal Processing Handbook* Ed. Vijary K. Madisetti and Douglas B. Williams, Boca Ration: CRC Press LLC, 1999.

[5] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore and London, Third Edition, 1996.

[6] Y. Hua and M. Nikpour, "Computing the reduced rank Wiener filter by IQMD," *IEEE Signal Processing Letters*, vol.6, no.9, pp.240-242, 1999.

[7] Y. Hua, M. Nikpour, and P. Stoica, "Optimal reduced-rank estimation and filtering," *IEEE Trans. Signal Processing*, vol.49, no.3, pp.457-469, 2001.

[8] Y. Hua, Y. Xiang, T. Chen, K. Abed-Meraim, and Y. Miao, "A new look at the power method for fast subspace tracking," *Digital Signal Processing*, vol.9, no.4, pp.297-314, Oct. 1999.

[9] Y. Miao and Y. Hua, "Fast subspace tracking and neural network learning by a novel information criterion," *IEEE Trans. Signal Processing*, vol.46, no.7, pp.1967-1979, 1998.

[10] E. Oja, "Neural networks, principal components, and subspaces," *Int. J. Neural Systmes*, vol.1, no.1 pp.61-68, 1989.

[11] E. C. Real, D. W. Tufts, and J. W. Cooley, "Two algorithms for fast approximate subspace tracking ," *IEEE Trans. Signal Processing*, vol.47, no.7, pp.1936-1945, 1999.

[12] G. W. Stewart, *Topics in numerical analysis* , J. J. H. Miller, Ed., New York, Second Edition,1975.

[13] P. Strobach, "Bi-iteration SVD subspace tracking algorithms," *IEEE Trans. Signal Processing*, vol.45, no.5, pp.1222-1240, 1997.

[14] B. Yang, "Projection approximation subspace tracking,"*IEEE Trans. Signal Processing*, vol.43, no.1, pp.95-107, 1995.