FAST APPROXIMATE SUBSPACE TRACKING (FAST)

D. W. Tufts
Department Of Electrical Engineering
University of Rhode Island
Kingston, RI 02881
USA

E. C. Real
Sanders, A Lockheed Martin Company
P. O. Box 868
Nashua, NH 03061-0868
USA

J. W. Cooley
Department Of Electrical Engineering
University of Rhode Island
Kingston, RI 02881
USA

ABSTRACT

A new fast and accurate algorithm for tracking singular values, singular vectors and the dimension of the signal subspace through an overlapping sequence of data matrices is presented. The accuracy of the algorithm approaches that of the Prony-Lanczos (PL) method [1] with speed and accuracy superior to both the PAST and PASTd algorithms [2] for moderate to large size problems. The algorithm is described for the special case of changes to two columns of the matrix prior to each update of principal singular vectors and values. Comparisons of speed and accuracy are made with the algorithms named above.

1. INTRODUCTION

In this paper we introduce a new algorithm for efficiently tracking the principal singular values and the associated left (or right) singular vectors of successive data matrices formed from observations of a non-stationary signal in non-stationary noise. The ability to do this accurately and in real time is a critical requirement of many signal processing applications in areas such as radar, sonar, communications, pattern recognition, and speech processing. In some of these applications, the component of the data which we here are calling the "signal" may actually be non-stationary interference; and it's subspace may be tracked for the purpose of suppressing that interference, rather than enhancing a signal or estimating its parameters.

Other algorithms exist which perform this function, and some are discussed by Bin Yang in his paper on the PAST and PASTd algorithms [2]. In this paper we make limited quantitative comparisons between our algorithm, and the PL, PAST and PASTd methods. The PL algorithm is chosen for its accuracy, and the PAST and PASTd algorithms because they have been found to be among the most computationally efficient [3].

2. STATEMENT OF THE PROBLEM

We begin by assuming the existence of a previously analyzed data matrix M_{old} of r rows and c columns, where the columns of the matrix are column vectors of sequential (time or space) samples of either real or complex data. We think of M_{old} as a matrix of observations of a low rank (e.g. rank k) signal matrix S_{old}, made in the presence of a full rank noise matrix N_{old}. That is,.

$$M_{\text{old}} = S_{\text{old}} + N_{\text{old}} \tag{1}$$

The columns of M_{old} are represented here as r by 1 data vectors $\boldsymbol{m_i}$ so that we have a general representation of M_{old} as:

$$M_{\text{old}} = \left[\boldsymbol{m_1}\ \boldsymbol{m_2}\ \cdots\ \boldsymbol{m_c}\right] \tag{2}$$

In representing M_{old} by column vectors we are assuming that the entire vector space and the signal subspace are also to be represented as column vectors. Consequently the singular vectors that are of interest are the left singular vectors. However, the entire argument presented here, and the proposed algorithm, could equally well be formulated in terms of row vectors, row spaces and right singular vectors.

We further assume that the size of the data matrix is fixed, and that as new data vectors $\boldsymbol{m_i}$ become available, the older vectors are removed to keep the row and column size constant. For example, if $\boldsymbol{m_{(c+1)}}$ is a new column vector of observations of a signal in noise that has just become available, then we might form the next data matrix M_{new} as follows:

$$M_{\text{new}} = S_{\text{new}} + N_{\text{new}} \tag{3a}$$

$$M_{\text{new}} = \left[\boldsymbol{m_2}\ \boldsymbol{m_3}\ \cdots\ \boldsymbol{m_{(c+1)}}\right] \tag{3b}$$

Our objective is to track the k singular values and associated left singular vectors of the signal matrix as it makes the transition from S_{old} to S_{new} in the presence of noise. For the sake of clarity, we derive the procedure for performing this tracking for single column updates; but we note here that more general, multi-column updates can be included. We also show how the results of this procedure can be used to update the estimated dimension of the signal subspace.

3. THEORETICAL DEVELOPMENT

We develop the theory supporting the algorithm given below in two steps. The first step is to develop a low rank matrix A that is appropriate for tracking the singular values and left singular vectors of a rank k signal matrix measured in the presence of noise. The matrix A is a reduced rank version of the data matrix M_{new}, and although it achieves the desired fidelity for preserving signal content, it will still have the same number of rows and columns as M_{new}. The major computational advantage is gained in step two by constructing a smaller r by $k+1$ matrix B which is constructed to retain the desirable properties of A.

Step One:

We begin the theoretical development by initially assuming that we have obtained a sufficiently accurate approximation to the set of k principal left singular vectors of M_{old} from a previous step in the algorithm. We organize these k orthonormal left singular vectors into the columns of a matrix U_{old} as

$$U_{\text{old}} = \left[\boldsymbol{u_1}\ \boldsymbol{u_2}\ \cdots\ \boldsymbol{u_k}\right] \tag{4}$$

where the vector $\boldsymbol{u_i}$ is the approximate left singular vector of M_{old} associated with the i-th largest approximate singular value. The columns of U_{old} are basis vectors for the signal subspace of M_{old}. In choosing these basis vectors for the signal subspace, we have also chosen an associated error value, E_{old}, namely the squared Frobenius norm error in the approximation of M_{old} by $U_{\text{old}}U_{\text{old}}^H M_{\text{old}}$ which we represent as:

$$E_{\text{old}} = \left\| M_{\text{old}} - U_{\text{old}}U_{\text{old}}^H M_{\text{old}} \right\|_F^2 \tag{5}$$

Now, for the updated data matrix M_{new} of equation (3b) we form a rank $k+1$ approximation matrix A, with r rows and c columns, in such a way that:

$$\left\| M_{\text{new}} - A \right\|_F^2 \le E_{\text{old}} \tag{6}$$

That is, we require a matrix A that approximates M_{new} at or below the error value, E_{old}, which results from approximating the columns of M_{old} by linear combinations of the columns of U_{old}. One matrix which meets this requirement is

$$A = \left[U_{\text{old}} \; \boldsymbol{q} \right] \begin{bmatrix} \boldsymbol{a_2} & \boldsymbol{a_3} & \cdots & \boldsymbol{a_c} & \boldsymbol{a_{(c+1)}} \\ 0 & 0 & \ldots & 0 & b \end{bmatrix} \tag{7}$$

in which $\boldsymbol{a_j} = U_{\text{old}}^H \boldsymbol{m_j}$ and $\boldsymbol{m_j}$ is the j-th column of M_{old}, and where the column vector $\boldsymbol{q}$ and the scalar b are obtained by decomposition of the new column vector $\boldsymbol{m_{(c+1)}}$ of equation (3b) into two components; one in the column space of U_{old} and one in the space orthogonal to U_{old}. These terms are computed as follows:

$$\boldsymbol{a_{(c+1)}} = U_{\text{old}}^H \boldsymbol{m_{(c+1)}} \tag{8}$$

$$\boldsymbol{z} = \boldsymbol{m_{(c+1)}} - U_{\text{old}} \boldsymbol{a_{(c+1)}} \tag{9}$$

$$b = \|\boldsymbol{z}\| \tag{10}$$

$$\boldsymbol{q} = \boldsymbol{z}/b \tag{11}$$

In short, the rationale for approximating M_{new} by A is this: if we are willing to accept the error resulting from a rank k approximation to the matrix M_{old} in a prior step, then we should be willing to accept the error resulting from approximating M_{new} by A since it will be no greater.

Step Two:

Although we have constructed A to be a good rank $k+1$ approximation to M_{new}, it is nevertheless still a matrix of identical size. In the interest of reducing the amount of computations required for updating U_{old} we wish to work with a matrix of smaller size than M_{new}. Therefore, we now replace the r by c matrix A by an r by $k+1$ matrix B, defined as follows:

$$B = \left[U_{\text{old}} \; \boldsymbol{q} \right] E E^H = \left[U_{\text{old}} \; \boldsymbol{q} \right] F \tag{12}$$

where like terms are the same as in equation (7), and E is defined as:

$$E = \begin{bmatrix} \boldsymbol{a_2} & \boldsymbol{a_3} & \cdots & \boldsymbol{a_c} & \boldsymbol{a_{(c+1)}} \\ 0 & 0 & \ldots & 0 & b \end{bmatrix} \tag{13}$$

With the matrix F defined by equation (12), we compute the singular values and left singular vectors of A as follows. First we form the SVD of the matrix F as:

$$F = U_F \Sigma_F V_F^H \tag{14}$$

The matrix B can then be written as:

$$B = \left[U_{\text{old}} \; \boldsymbol{q} \right] F = \left[U_{\text{old}} \; \boldsymbol{q} \right] U_F \Sigma_F V_F^H \tag{15}$$

The left singular vectors of B are the same as the left singular vectors of A and are computed as:

$$U_B = \left[U_{\text{old}} \; \boldsymbol{q} \right] U_F \tag{16}$$

The $k+1$ singular values of B are the same as the $k+1$ principal singular values of A and are computed as:

$$\Sigma_B = \Sigma_F^{1/2} \tag{17}$$

In words, the $k+1$ singular values of Σ_B and the associated columns of U_B will approximate the $k+1$ principal singular values and vectors of M_{new}. These approximations are computed efficiently now because, given U_{old} and $\boldsymbol{q}$ from a previous iteration, the only SVD required is that of the matrix F, and the matrix F is only of size $k+1$ by $k+1$, where k is typically a small number in comparison to r or c.

4. UPDATING THE PRINCIPAL SINGULAR VALUES AND ASSOCIATED LEFT SINGULAR VECTORS

The basic steps of the updating portion of the FAST algorithm are summarized below. As noted above, the matrix F is a $k+1$ by $k+1$ matrix which is typically much smaller than the original data matrix M_{new}, making an ordinary SVD adequate, from a computational point of view, for many applications. However additional speed improvements may be had by optimizing or replacing the SVD in this step (e.g. replace step (7) with a pruned PL computation). In the results described below the SVD of the small matrix F has not been modified or optimized.

Step (0) Initialize. First obtain an initial estimate of the k principal singular values and associated left singular vectors of the starting data matrix M. One way to do this is to compute the SVD of M, and from

it extract the required k principal singular values and construct the matrix U_{old} as in equation (4).

Step (1) Obtain the next data vector $\boldsymbol{m}_{(c+1)}$, and compute $\boldsymbol{a}_i = U_{\text{old}}^H \boldsymbol{m}_i$ for $i = 2, 3, \ldots, (c+1)$.

Step (2) Compute $\boldsymbol{z} = \boldsymbol{m}_{(c+1)} - U_{\text{old}} \boldsymbol{a}_{(c+1)}$.

Step (3) Compute $b = \|\boldsymbol{z}\|$.

Step (4) Compute $\boldsymbol{q} = \boldsymbol{z} / b$.

Step (5) Form the E matrix

$$E = \begin{bmatrix} \boldsymbol{a}_2 & \boldsymbol{a}_3 & \cdots & \boldsymbol{a}_c & \boldsymbol{a}_{(c+1)} \\ 0 & 0 & \ldots & 0 & b \end{bmatrix}$$

Step (6) Compute the matrix $F = EE^H$.

Step (7) Compute the SVD of $F = U_F \Sigma_F V_F^H$.

Step (8) Using U_{old}, $\boldsymbol{q}$ and U_F, update U_{old} by replacing its columns with the principal singular vectors of $U_B = \left[U_{\text{old}} \; \boldsymbol{q}\right] U_F$.

Step (9) Update the existing singular values by replacing them with the square roots of the principal singular values of Σ_F.

Step (10) Update the data vectors $\boldsymbol{m}_i$ by setting $\boldsymbol{m}_i \leftarrow \boldsymbol{m}_{(i+1)}$, $i = 1, 2, \ldots, c$ and return to Step (1).

5. TRACKING CHANGES IN THE DIMENSION OF THE SIGNAL SUBSPACE

In steps (8) and (9) of the algorithm we have left the number of singular values and associated left singular vectors that are being updated unspecified. This is because in many practical applications (e.g. speech, sonar, and radar) the dimension of the signal subspace is constantly changing. Therefore, it is necessary to both detect any change in signal subspace dimension, and to track the singular values and vectors through that change.

Our technique for determining the dimension of the signal subspace begins by computing the square of the Frobenius norm of the data matrix. This is equal to the sum of all the squared singular values of the data matrix, and thus gives a quick way of determining the total matrix energy. We now assume that we have already determined the dimension of the signal subspace and the corresponding singular values and vectors for the previous matrix M_{old}. From the total matrix energy we subtract successively larger sums of squares of the largest of these estimated singular values, until the difference lies below a chosen threshold value. This event determines the estimated subspace dimension, and is computed as follows. Let the total data matrix energy be given by $E_0 = \|M\|_F^2$, then for $i = 1, 2, \ldots, k$ compute

$$E_i = \|M\|_F^2 - \sum_{j=1}^{i} \sigma_j^2 = \sum_{j=i+1}^{N} \sigma_j^2 \qquad (18)$$

and compare each E_i (including E_0) to the threshold. The number of times E_i exceeds the threshold is the estimated dimension of the signal subspace. This approach only requires a threshold, estimates of the k principal singular values of the data matrix from the tracking algorithm and the total energy in that matrix.

The required threshold may be set either theoretically from knowledge or assumption about the power in the orthogonal complement subspace [4,5,6] or heuristically from estimates of that power. However this is done, the threshold should be chosen large enough so that exceeding it indicates the presence of a signal with high probability.

If the procedure leading to equation (18) above indicates that the signal dimension has increased since the last update of the algorithm, and assuming that in the previous update we were tracking k singular values and left singular vectors, then all $k + 1$ singular values and left singular vectors computed in steps (8) and (9) of the previous section can be used to increase the number of singular values and vectors tracked by one. On the other hand, if the procedure indicates that the signal dimension has decreased, the number of singular values and vectors tracked can be decreased by the indicated amount by simply retaining only the largest singular values and their associated left singular vectors. Thus the FAST algorithm can increase the number of dimensions tracked by one at each update, but can decrease that number by any amount.

6. AN EXAMPLE

An example of the performance of this algorithm is given here, and a comparison is made with other well known algorithms such as the Prony-Lanczos, PAST and PASTd methods. In this example the two largest singular values of two complex sinusoids in complex Gaussian noise are tracked.

Table 1 shows the parameters of this experiment. Figure 1 shows the ability of two of the algorithms considered to track the largest singular value in a sequence of matrices over fifty updates. The PL method is not shown because on this scale its accuracy is virtually identical to the SVD. The PAST algorithm is not shown because in this experiment its performance is comparable to that of the PASTd algorithm. Figure 2 shows similar results for the second largest singular value. Tables 2 and 3 summarize the accuracy results relative to a standard SVD for a run of 1000 updates using the parameters of Table 1. Table 4 summarizes the corresponding speed improvements of each algorithm over a standard SVD computation. That is, the mean speed up refers to the average multiplicative speed up achieved by the indicated algorithm relative to performing a full SVD to estimate the variable.

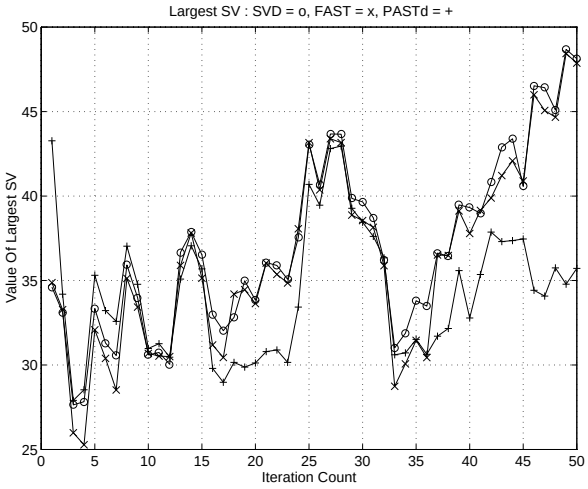

Figure 1: A comparison of the trace of the largest singular value computed by the SVD over 50 iterations vs. the estimate of it from the various algorithms indicated.

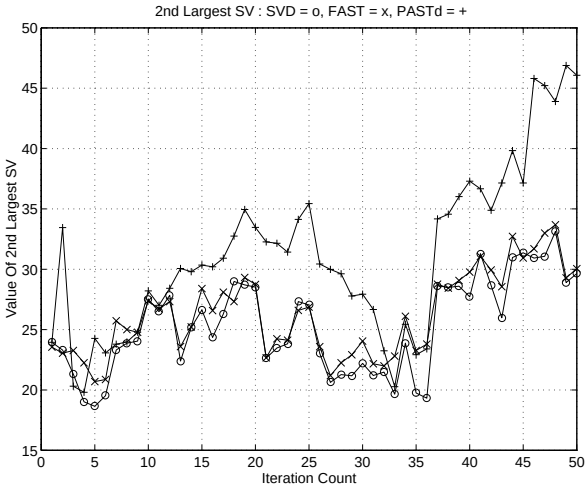

Figure 2: A comparison of the trace of the second largest singular value computed by the SVD over 50 iterations vs. the estimate of it from the various algorithms indicated.

Parameter	Value
Normalized radian frequency 1 (constant)	$2\pi/3$
Normalized radian frequency 2 (constant)	$4\pi/5$
Noise standard deviation (real and imaginary)	0.1
Number of rows of M	64
Number of columns of M	8
Number of singular values/vectors tracked	2
Beta value (for PASTd algorithm)	0.95

Table 1: Parameters Of Experiment

Algorithm	Mean Error	Standard Deviation
PL	-0.07047	0.2386
PAST	-5.581	7.283
PASTd	-3.785	8.932
FAST	-0.5896	0.8188

Table 2: Statistics For Largest Singular Value Estimate

.

Algorithm	Mean Error	Standard Deviation
PL	-0.1848	1.038
PAST	7.124	4.988
PASTd	6.413	4.153
FAST	0.843	1.166

Table 3: Statistics For 2nd Largest Singular Value Estimate

Algorithm	Mean Speed Up
PL	1.7553
PAST	1.7571
PASTd	1.7541
FAST	39.4356

Table 4: Speed Improvements Over A Standard SVD

7. REFERENCES

[1] Donald W. Tufts and Constantinos D. Melissinos, “Simple, Effective Computation of Principal Eigenvectors and Their Eigenvalues and Application to High-Resolution Estimation Of Frequencies”, IEEE Transactions On Acoustics, Speech And Signal Processing, Vol. ASSP-34, No. 5, October 1986, pp. 1046 - 1053.

[2] Bin Yang, “Projection Approximation Subspace Tracking”, IEEE Transactions On Signal Processing, Vol. 43, No. 1, January 1995, pp. 95 - 107.

[3] C. MacInnes, “Analysis of Small and Large Perturbations of Matrix Subspaces”, Ph.D. thesis, University of Rhode Island, Kingston, Rhode Island, USA 02881, 1995.

[4] Abhijit A. Shah and Donald W. Tufts, “Determination of the Dimension of a Signal Subspace from Short Data Records”, IEEE Transactions On Signal Processing, Vol. 42, No. 9, September 1994, pp. 2531 - 2535.

[5] Donald W. Tufts and Abhijit A. Shah, “Rank Determination In Time-Series Analysis”, Proceedings of the 1994 IEEE International Conference on Acoustics, Speech and Signal Processing. Vol. IV, pp. 21 - 24.

[6] D. W. Tufts and C. D. Melissinos, “Threshold Extension Of SVD-Based Algorithms”, Proceedings of the 1988 IEEE International Conference on Acoustics, Speech and Signal Processing, Vol. V, pp. 2825 - 2828.