IMPROVED FAST ADAPTIVE SUBSPACE TRACKING

Timothy M. Toolan, Donald W. Tufts

Deptartment of Electrical Engineering
University of Rhode Island
4 East Alumni Avenue
Kingston, RI 02881
toolan@ele.uri.edu, tufts@ele.uri.edu

ABSTRACT

A new subspace tracking algorithm which gives accurate estimates of the r largest singular values and corresponding left singular vectors of overlapping rectangular matrices is presented. This algorithm has evolved from the Fast Approximate Subspace Tracking (FAST) algorithm by Real, Tufts, and Cooley, but has significantly better accuracy and computational efficiency.

When there are abrupt changes in data, or the data is changing rapidly, a rectangular window can often give better performance than an exponential window because it can limit exactly how much older data is included. Some methods for estimating the signal subspace dimension require the singular values of the strong subspace. This algorithm can update the r largest singular values and corresponding left singular vectors in $O(nr^2)$, where n is the number of channels and r is the dimension of the strong subspace that we are tracking. There are no assumptions made about the strong subspace, but the accuracy of our singular vector and singular value estimates is related to the separation between the strong "signal" subspace and the weak "noise" subspace, which is just the signal to noise ratio.

In this paper we present, (a) why this algorithm works, (b) why it is fast, and (c) how to use it to track a strong subspace.

1. INTRODUCTION

A new subspace tracking algorithm which gives accurate estimates of the r largest singular values and corresponding left singular vectors of overlapping rectangular matrices is presented. This algorithm, which we will call *Improved Fast Approximate Subspace Tracking* (IFAST), has evolved from the Fast Approximate Subspace Tracking (FAST) algorithm by Real, Tufts, and Cooley [1], but has significantly better accuracy and computational efficiency. A detailed analytical analysis of the effect that advancing a rectangular window by one column has on the singular value decomposition (SVD) is presented, along with its applicability to the IFAST algorithm. Additionally, we present techniques for starting without an initial SVD, and moving the overlapping matrices more than one column at a time.

We have been motivated by problems of detection and estimation in a non-stationary environment. Often the "signal" subspace is really a rapidly varying subspace of interference or clutter and we wish to track the subspace in order to facilitate removal of the interference. Two examples are (1) the rapidly time-and-space varying clutter in multispectral images [2] and (2) Terrain Scattered Interference in airborne radar [3].

We start with an $n \times c$ matrix M, whose columns consist of sequential complex-valued samples in time and/or space. For example, each column may be the output of an array of sensors at a given time, or a new column in a Hankel matrix created from a single sensor. M is of the form

$$M = S + N \tag{1}$$

where S is a rank r signal matrix and N is a full rank noise matrix. Our goal is to efficiently and accurately determine the signal subspace dimension, r, along with the r largest singular values and corresponding left singular vectors of M. The problem of determining r when we have the singular values of M has been addressed in [4] and [5], and generalized in [6].

2. THE IFAST ALGORITHM

In this section we describe the improved fast adaptive subspace tracking (IFAST) algorithm. Given a sequence of length n column vectors, we can define the two $n \times c$ matrices for time $t-1$ and t as

$$M = \begin{bmatrix} \boldsymbol{x}_{t-c} & \boldsymbol{x}_{t-c+1} & \cdots & \boldsymbol{x}_{t-1} \end{bmatrix}, \tag{2}$$

$$\tilde{M} = \begin{bmatrix} \boldsymbol{x}_{t-c+1} & \boldsymbol{x}_{t-c+2} & \cdots & \boldsymbol{x}_{t} \end{bmatrix}. \tag{3}$$

Assuming we have the r largest singular values and corresponding left singular vectors of M, which we will call Σ' and U', we would like to determine the r largest

TABLE I

IFAST ALGORITHM

Step	Calculation
1)	$\boldsymbol{q}_1 = \dfrac{\left(I - U'U'^H\right)\boldsymbol{x}_t}{\lVert\left(I - U'U'^H\right)\boldsymbol{x}_t\rVert}$
2)	$\boldsymbol{q}_2 = \dfrac{\left(I - [\,U' \mid \boldsymbol{q}_1\,]\,[\,U' \mid \boldsymbol{q}_1\,]^H\right)\boldsymbol{x}_{t-c}}{\left\lVert\left(I - [\,U' \mid \boldsymbol{q}_1\,]\,[\,U' \mid \boldsymbol{q}_1\,]^H\right)\boldsymbol{x}_{t-c}\right\rVert}$
3)	$\tilde{F}_c = \Sigma'^2 - U'^H\boldsymbol{x}_{t-c}\boldsymbol{x}_{t-c}^H U' + U'^H\boldsymbol{x}_t\boldsymbol{x}_t^H U'$
	$\tilde{F} = \left[\begin{array}{c\|c} \tilde{F}_c & U'^H\tilde{M}\tilde{M}^H Q \\ \hline Q^H\tilde{M}\tilde{M}^H U' & Q^H\tilde{M}\tilde{M}^H Q \end{array}\right]$
4)	$U_F\Sigma_F U_F^H = \tilde{F}$
5)	$\tilde{U}' = [\,U' \mid Q\,]U_F$
	$\tilde{\Sigma}'^2 = \Sigma_F$

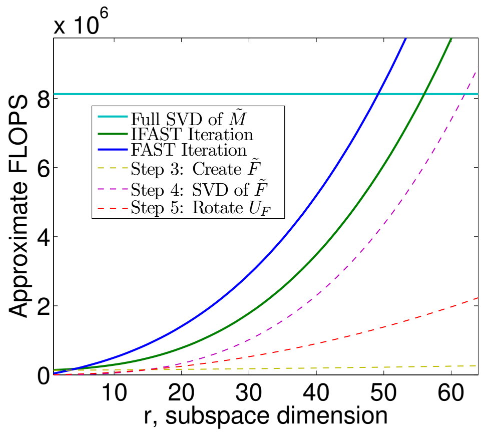

Fig. 1. Computation of the IFAST algorithm for $n = c = 64$ for different subspace dimensions vs. a full SVD of $\tilde{M}$ and the original FAST algorithm.

singular values and corresponding left singular vectors of $\tilde{M}$ without performing a full SVD on $\tilde{M}$.

2.1. The Steps of the Algorithm

The steps of the algorithm are given in table I. The order of evaluation is important in efficiently performing the calculations (for example step one should be evaluated as $\boldsymbol{x}_t - U'(U'\boldsymbol{x}_t)$).

The first two steps use a Gram-Schmidt method to create the matrix $Q \in \mathbb{C}^{n\times 2}$, which is an orthogonal and normal basis for the subspace defined by the vectors that we are adding and discarding, $\boldsymbol{x}_t$ and $\boldsymbol{x}_{t-c}$, that is orthogonal to U'.

The third step creates the matrix $\tilde{F} \in \mathbb{C}^{r+2\times r+2}$, which is equivalent to

$$\tilde{F} = [\,U' \;\; Q\,]^H\tilde{M}\tilde{M}^H[\,U' \;\; Q\,]. \tag{4}$$

The reason $\tilde{F}$ is calculated as in step three of table I, is because the matrix product in (4) is an $O(rnc + rc^2)$ computation, and would dominate the algorithm for small r, while $\tilde{F}_c$ from step 3 is a diagonal matrix plus two rank one matrices.

Taking the SVD of $\tilde{F}$ is equivalent to taking the SVD of the rank $r+2$ matrix

$$\tilde{M}' = [\,U' \;\; Q\,][\,U' \;\; Q\,]^H\tilde{M}, \tag{5}$$

thus steps four and five give us the true singular values and singular vectors of $\tilde{M}'$.

2.2. Why IFAST Works

The details of why IFAST is accurate are presented later, but a general explanation is given here. Since the columns of U' are the r largest singular vectors for M, and M has all but one column in common with $\tilde{M}$, then U' must be a pretty good approximation to $\tilde{U}'$ to begin with. The construction of Q ensures that the part of the two vectors that differ between M and $\tilde{M}$, and is not included in U', is included in $\tilde{F}$.

2.3. Computation and Accuracy

Figure 1 shows the approximate FLOPS required for a single iteration, when M is complex, and $n = c = 64$ for both the full SVD, an IFAST update, and an original FAST update. The dominant steps, steps three, four and five, are also shown.

Figure 2 illustrates the accuracy of the IFAST algorithm. The signal subspace consists of two complex chirps, and one complex sinusoid which starts at time 100, and ends at time 350. The data is generated from a single sensor, then made into 32×32 hankel matrices. The noise is uncorrelated complex Gaussian, with a variance of 1.5. The signal is the same as that used in the example from [1], with a much lower SNR. The top three plots show $10\log_{10}((\tilde{\sigma}_i^2 - \tilde{\sigma}_i'^2)/\tilde{\sigma}_i^2)$, for $i = 1, 2, 3$, where $\tilde{\sigma}_i$ is the true ith singular value of $\tilde{M}$, and $\tilde{\sigma}_i'$ is the estimate of the ith singular value using the specified algorithm. The rank is tracked using the method from [6]. The blue and green line use the estimate for U' from the previous iteration, while the red line uses the true U' generated by taking a full SVD of M.

3. SLIDING A RECTANGULAR WINDOW

In this section we will take a look at exactly what happens to the singular values and left singular vectors when we advance a rectangular window by one snapshot.

3.1. The Full Subspace

When we have a length c rectangular window of our data matrix $X \in \mathbb{C}^{n\times L}$, and advance the window by

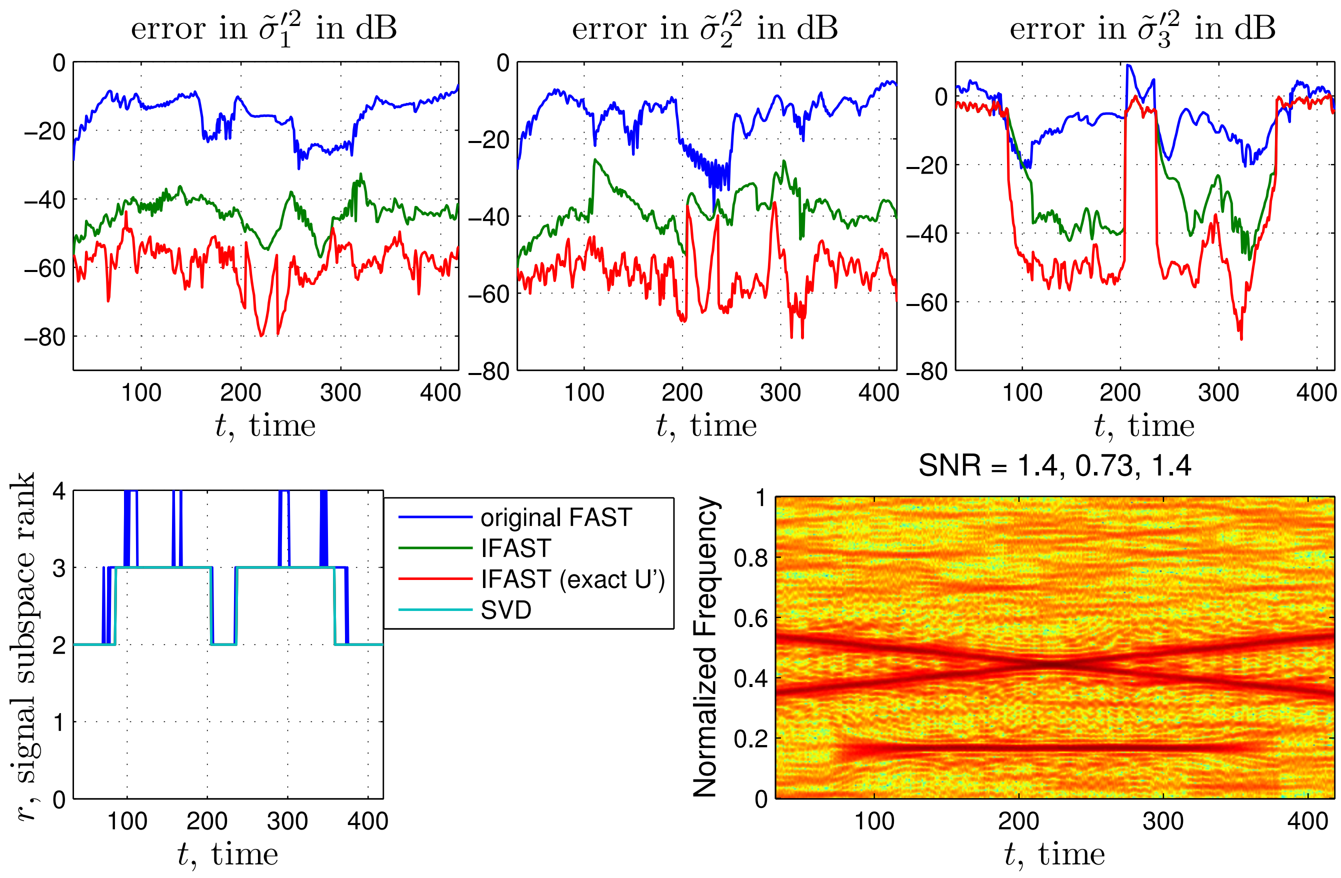

Fig. 2. An example to to illustrate the accuracy of IFAST.

one snapshot, the new matrix will have $c-1$ columns in common with the previous matrix. We can write $\tilde{M}$ as

$$\tilde{M} = \left(M - \boldsymbol{x}_{t-c}\boldsymbol{e}_1^T\right)P + \boldsymbol{x}_t\boldsymbol{e}_c^T, \tag{6}$$

where $\boldsymbol{e}_1$ and $\boldsymbol{e}_c$ are the 1st and cth length c canonical vectors, and P is a $c \times c$ identity matrix with the first column moved to the far right, and all other columns shifted one to the left. Note that the ith length c canonical vector, $\boldsymbol{e}_i$, is the ith column of a $c \times c$ identity matrix. Multiplying $\tilde{M}$ as defined in (6) by its conjugate transpose, we get

$$\tilde{M}\tilde{M}^H = MM^H - \boldsymbol{x}_{t-c}\boldsymbol{x}_{t-c}^H + \boldsymbol{x}_t\boldsymbol{x}_t^H, \tag{7}$$

which makes it clear that advancing the rectangular window by one snapshot, is a rank-two modification to the underlying symmetric eigenproblem.

If we write the singular value decomposition of M as $M = U\Sigma V^H$, and define

$$\boldsymbol{a} = U^H\boldsymbol{x}_{t-c}, \qquad \boldsymbol{b} = U^H\boldsymbol{x}_t, \tag{8}$$

then multiply (7) by the left singular vectors of M from both the left and right, we get

$$\tilde{G} = U^H\tilde{M}\tilde{M}^H U = \Sigma^2 - \boldsymbol{a}\boldsymbol{a}^H + \boldsymbol{b}\boldsymbol{b}^H, \tag{9}$$

where the Σ^2 term comes from $U^H MM^H U$. Note that (9) is a diagonal matrix plus two rank one matrices. If we write the SVD of $\tilde{G}$ as $U_G\Lambda_G U_G^H$, then we can write the singular values and left singular vectors of $\tilde{M}$ as

$$\tilde{\Sigma} = \sqrt{\Lambda_G}, \qquad \tilde{U} = UU_G. \tag{10}$$

What this means, is that we can analyze what happens to the SVD of M when we create $\tilde{M}$ by analyzing the the eigendecomposition of $\tilde{G}$.

The eigenvalues of $\tilde{G}$ are the roots of the rank-two secular equation

$$w(\lambda) = \left(1 - \sum_{j=1}^{n}\frac{|a_j|^2}{\sigma_j^2 - \lambda}\right)\left(1 + \sum_{j=1}^{n}\frac{|b_j|^2}{\sigma_j^2 - \lambda}\right) + \left|\sum_{j=1}^{n}\frac{a_j^* b_j}{\sigma_j^2 - \lambda}\right|^2. \tag{11}$$

The blue line in figure (3) shows an example of $w(\lambda)$. Note that it is concealed by the red line for much of the plot. The singular vectors can be determined by a simple matrix product, which is shown in [7].

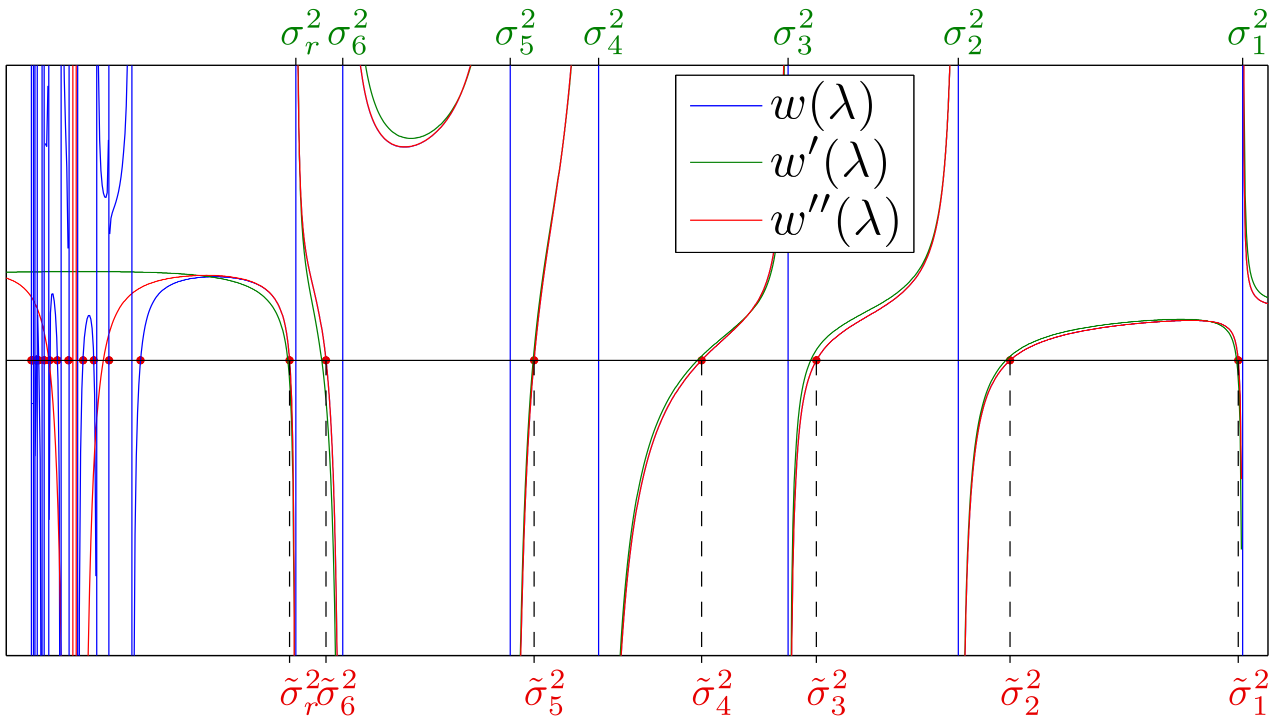

Fig. 3. An example of the rank-two secular equation vs. λ for $r = c = 20$, the principal subspace $r = 7$, and an SNR of 4. The poles are the old singular values and the roots are the new singular values.

3.2. The Principal Subspace

If we separate M into a strong r dimensional principal subspace, and an orthogonal $(c - r)$ dimensional weak subspace, we can write the SVD of M as

$$M = \left[\begin{array}{cc} U' & U^\perp \end{array}\right] \left[\begin{array}{cc} \Sigma' & 0 \\ 0 & \Sigma^\perp \end{array}\right] \left[\begin{array}{cc} V' & V^\perp \end{array}\right]^H . \quad (12)$$

where $\Sigma' \in \mathbb{R}^{r\times r}$ contains the r largest singular values of M, and $U' \in \mathbb{C}^{n\times r}$ contains the first r columns of U.

If we define the reduced rank version of $\tilde{G}$ to be

$$\tilde{G}' = U'^H \tilde{M}\tilde{M}^H U' = \Sigma'^2 - \boldsymbol{a}'\boldsymbol{a}'^H + \boldsymbol{b}'\boldsymbol{b}'^H , \quad (13)$$

where $\boldsymbol{a}'$ and $\boldsymbol{b}'$ are just the first r elements of $\boldsymbol{a}$ and $\boldsymbol{b}$ respectively, then the eigenvalues of $\tilde{G}'$ are the roots of

$$w'(\lambda) = \left(1 - \sum_{j=1}^{r} \frac{|a_j|^2}{\sigma_j^2 - \lambda}\right)\left(1 + \sum_{j=1}^{r} \frac{|b_j|^2}{\sigma_j^2 - \lambda}\right) + \left|\sum_{j=1}^{r} \frac{a_j^* b_j}{\sigma_j^2 - \lambda}\right|^2 . \quad (14)$$

Equation (14) differs from (11) only by the upper limit of the summation. The green line in figure (3) shows an example of $w'(\lambda)$.

3.3. The IFAST Approximation

Since the vectors $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ from steps one and two of table I are only intended to determine an orthogonal and normal basis for the subspace defined by the vectors that we are adding and discarding, $\boldsymbol{x}_t$ and $\boldsymbol{x}_{t-c}$, that is orthogonal to U', we can rotate Q by any unitary matrix, and the algorithm will be unchanged. If we rotate Q, such that the 2×2 matrix

$$\hat{\Sigma} = Q^H M M^H Q \quad (15)$$

is diagonal, then the lower portion of $\tilde{F}$ from step three, $Q^H \tilde{M}\tilde{M}^H Q$, will be a diagonal matrix plus two rank one matrices, which although trivial for a 2×2 matrix, will be necessary later.

If we assume that U' consists of the r true left singular vectors of M (as opposed to approximations), we can write $U'^H \tilde{M}\tilde{M}^H Q = \Sigma'^2 U'^H Q$, where where U' and Σ' are from (12). The construction of Q ensures that $U'^H Q$ equals zero, therefore we can write the whole of $\tilde{F}$, and not just $\tilde{F}_c$, as a diagonal matrix plus two rank one matrices, whose secular equation is

$$\begin{aligned} w''(\lambda) = & \left(1 - \sum_{j=1}^{r} \frac{|a_j|^2}{\sigma_j^2 - \lambda} - \sum_{j=1}^{2} \frac{\left|\boldsymbol{q}_j^H \boldsymbol{x}_{t-c}\right|^2}{\hat{\sigma}_j - \lambda}\right) \\ & \cdot \left(1 + \sum_{j=1}^{r} \frac{|b_j|^2}{\sigma_j^2 - \lambda} + \sum_{j=1}^{2} \frac{\left|\boldsymbol{q}_j^H \boldsymbol{x}_t\right|^2}{\hat{\sigma}_j - \lambda}\right) \\ & + \left|\sum_{j=1}^{r} \frac{a_j^* b_j}{\sigma_j^2 - \lambda} + \sum_{j=1}^{2} \frac{\boldsymbol{q}_j^H \boldsymbol{x}_t \boldsymbol{x}_{t-c}^H \boldsymbol{q}_j}{\hat{\sigma}_j - \lambda}\right|^2 , \end{aligned} \quad (16)$$

where $\hat{\sigma}_j$ comes from $\hat{\Sigma}$ in (15). The red line in figure (3) shows an example of $w''(\lambda)$.

When we perform binomial expansions of the two terms added to each summation by IFAST, we find that the first two terms of this binomial expansion are the same as the first two terms of the binomial expansion of the $n-r$ terms from $w(\lambda)$ that are missing in $w'(\lambda)$. We will show this for just one part of $w''(\lambda)$, but the procedure to show it for the other two parts is similar.

3.4. Binomial Expansions

For λ greater than σ_{r+1} and $\hat{\sigma}_1$, we can use a binomial expansion to write

$$\sum_{j=1}^{2} \frac{\left|\boldsymbol{q}_j^H \boldsymbol{x}_t\right|^2}{\hat{\sigma}_j - \lambda} = -\sum_{j=1}^{2} \frac{\left|\boldsymbol{q}_j^H \boldsymbol{x}_t\right|^2}{\lambda} \sum_{i=0}^{\infty} \left(\frac{\hat{\sigma}_j}{\lambda}\right)^i \tag{17}$$

and

$$\sum_{j=r+1}^{n} \frac{|b_j|^2}{\sigma_j^2 - \lambda} = -\sum_{j=r+1}^{n} \frac{|b_j|^2}{\lambda} \sum_{i=0}^{\infty} \left(\frac{\sigma_j^2}{\lambda}\right)^i . \tag{18}$$

Removing the first two terms from each of the binomial expansions, we get

$$-\sum_{j=1}^{2} \frac{\left|\boldsymbol{q}_j^H \boldsymbol{x}_t\right|^2}{\lambda} \left(1 + \frac{\hat{\sigma}_j}{\lambda} + \sum_{i=0}^{\infty} \left(\frac{\hat{\sigma}_j}{\lambda}\right)^i\right) \tag{19}$$

and

$$-\sum_{j=r+1}^{n} \frac{|b_j|^2}{\lambda} \left(1 + \frac{\sigma_j^2}{\lambda} + \sum_{i=2}^{\infty} \left(\frac{\sigma_j^2}{\lambda}\right)^i\right) . \tag{20}$$

Addressing the first order term from the binomial expansion in (19), we get

$$\sum_{j=1}^{2} \left|\boldsymbol{q}_j^H \boldsymbol{x}_t\right|^2 = \sum_{j=1}^{2} \boldsymbol{q}_j^H \boldsymbol{x}_t \boldsymbol{x}_t^H \boldsymbol{q}_j \tag{21}$$

which we can write in matrix form as $\boldsymbol{x}_t^H QQ^H \boldsymbol{x}_t$. Because of the way Q was constructed, $U^\perp U^{\perp H} \boldsymbol{x}_t$ equals $QQ^H \boldsymbol{x}_t$, therefore (21) equals $\boldsymbol{x}_t^H U^\perp U^{\perp H} \boldsymbol{x}_t$, which is the first order term from (20).

The second order term from the binomial expansion in (20), can be written as

$$\sum_{j=r+1}^{n} |b_j|^2 \sigma_j^2 = \sum_{j=r+1}^{n} \boldsymbol{u}_j^H \boldsymbol{x}_t \boldsymbol{x}_t^H \boldsymbol{u}_j \sigma_j^2, \tag{22}$$

which we can write in matrix form as $\boldsymbol{x}_t^H U^\perp \Sigma^{\perp 2} U^{\perp H} \boldsymbol{x}_t$. We can insert the identity matrix $U^{\perp H} U^\perp$ to get

$$\boldsymbol{x}_t^H U^\perp U^{\perp H} U^\perp \Sigma^{\perp 2} U^{\perp H} U^\perp U^{\perp H} \boldsymbol{x}_t. \tag{23}$$

Substituting $QQ^H \boldsymbol{x}_t = U^\perp U^{\perp H} \boldsymbol{x}_t$, we get

$$\boldsymbol{x}_t^H QQ^H U^\perp \Sigma^{\perp 2} U^{\perp H} QQ^H \boldsymbol{x}_t, \tag{24}$$

which is equal to

$$\boldsymbol{x}_t^H QQ^H U \Sigma^2 U^H QQ^H \boldsymbol{x}_t \tag{25}$$

because Q is orthogonal to the columns of U'. This can be written as

$$\boldsymbol{x}_t^H QQ^H MM^H QQ^H \boldsymbol{x}_t. \tag{26}$$

Finally, since $Q^H MM^H Q = \hat{\Sigma}$, we get

$$\boldsymbol{x}_t^H Q \hat{\Sigma} Q^H \boldsymbol{x}_t, \tag{27}$$

which is the same as the second order term from (19).

Combining all of this, the difference of (18) and (17) can be written as

$$e_b'' = \sum_{j=1}^{2} \frac{\left|\boldsymbol{q}_j^H \boldsymbol{x}_t\right|^2}{\lambda} \sum_{i=2}^{\infty} \left(\frac{\hat{\sigma}_j}{\lambda}\right)^i - \sum_{j=r+1}^{n} \frac{|b_j|^2}{\lambda} \sum_{i=2}^{\infty} \left(\frac{\sigma_j^2}{\lambda}\right)^i , \tag{28}$$

which can be written as

$$e_b'' = \sum_{i=2}^{\infty} \boldsymbol{x}_t^H Q \left(\frac{\hat{\Sigma}^i - Q^H U^\perp \Sigma^{\perp 2i} U^{\perp H} Q}{\lambda^{i+1}}\right) Q^H \boldsymbol{x}_t, \tag{29}$$

or in terms of M,

$$e_b'' = \sum_{i=2}^{\infty} \boldsymbol{x}_t^H Q \left(\frac{(Q^H MM^H Q)^i - Q^H (MM^H)^i Q}{\lambda^{i+1}}\right) Q^H \boldsymbol{x}_t. \tag{30}$$

The other difference terms can be written similarly as

$$e_a'' = \sum_{i=2}^{\infty} \boldsymbol{x}_{t-c}^H Q \left(\frac{(Q^H MM^H Q)^i - Q^H (MM^H)^i Q}{\lambda^{i+1}}\right) Q^H \boldsymbol{x}_{t-c}. \tag{31}$$

and

$$e_{ab}'' = \sum_{i=2}^{\infty} \boldsymbol{x}_{t-c}^H Q \left(\frac{(Q^H MM^H Q)^i - Q^H (MM^H)^i Q}{\lambda^{i+1}}\right) Q^H \boldsymbol{x}_t. \tag{32}$$

Figure 4 shows the difference term $w_a(\lambda) - w_a''(\lambda)$, which is e_a'', along with $w_a''(\lambda) - w_{a,com}(\lambda)$, where $w_{a,com}(\lambda)$ is the part of $w_a''(\lambda)$ and $w_a(\lambda)$ that is common to both of them.

4. MODIFICATIONS TO ALGORITHM

It is possible to make a simple modification to the algorithm so that an initial SVD is not required to get our first $\tilde{U}'$ and $\tilde{\Sigma}'$. To do this we start with a single column, $\boldsymbol{x}_0$, thus $c = 1$. Its singular value is just its two norm, $\tilde{\Sigma}' = \|\boldsymbol{x}_0\|$, and its singular vector is just itself divided by its singular value, $\tilde{U}' = \boldsymbol{x}_0 / \tilde{\Sigma}'$. Each time we get a new column, we increment c until M reaches the dimensions we want. While M is growing, we do not need to calculate step two from the IFAST algorithm, therefore Q will only be an $n \times 1$ matrix, and $\tilde{F}$ will be a $r+1 \times r+1$ matrix.

When more than one new column is added and removed each iteration, it is possible to make a simple modification

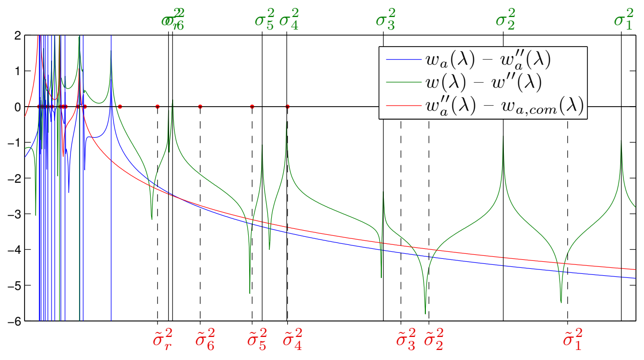

Fig. 4. $\log_{10}$ of the some difference terms using the secular equations.

to the algorithm to account for this. Basically, the matrix Q will have one column for each column added to $\tilde{M}$ and one column for each column that is removed, thus if k new columns are available, and the number of columns in M is kept the same, the matrix Q will have dimensions $n \times 2k$ and the matrix F will have dimensions $r + 2k \times r + 2k$. Each new column in Q must be normal, and will contain the portion of the corresponding $\boldsymbol{x}_i$ that is orthogonal to U' as well as the previously calculated columns of Q.

When the matrix M has Hankel structure, the calculations required to construct F can be further reduced using the fast Fourier transform. This has been present for the original version of FAST in [8], and can easily be applied to the new version of this paper.

5. ADDITIONAL COMMENTS

It should be noted that the IFAST algorithm works for both complex and real data, while the rank tracking method from [6] is designed for complex data only. The IFAST algorithm is designed to update the singular values and singular vectors of M, thus it is independent of the rank tracking algorithm, therefore any rank tracking method can be used.

When there is a computational limit on the dimensions of $\tilde{F}$ for the small SVD in step 4 from table I, we can still calculate the number of singular values and singular vectors that we are capable of, and they will represent the r largest singular values of M.

6. REFERENCES

[1] E. C. Real, D. W. Tufts, and J. W. Cooley, "Two algorithms for fast approximate subspace tracking," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 47, no. 7, pp. 1036–1045, July 1999.

[2] E. C. Real, R. M. Yannone, and D. W. Tufts, "Comparison of two methods for multispectral 3-d detection of single pixel features in strong textured clutter," in *IMDSP Conference, Alpbach, Austria*, July 1998.

[3] D. W. Tufts, "Keynote address," in *MIT Lincoln Laboratory ASAP Conference*, Mar. 2001.

[4] A. A. Shah and D. W. Tufts, "Determination of the dimension of a signal subspace from short data records," *IEEE Trans. Signal Processing*, vol. 42, no. 9, pp. 2531–2535, Sept. 1994.

[5] D. W. Tufts and A. A. Shah, "Rank determination in time-series analysis," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Proc.*, Apr. 1994, pp. IV–21–IV–24.

[6] T. M. Toolan and D. W. Tufts, "Detection and estimation in non-stationary environments," in *Proc. IEEE Asilomar Conference on Signals, Systems & Computers*, Nov. 2003, pp. 797–801.

[7] ——, "Rank-two modification of the symmetric eigenproblem," June 2005, submitted to SIAM J. Matrix Anal. and Appl.

[8] J. W. Cooley, T. M. Toolan, and D. W. Tufts, "A subspace tracking algorithm using the fast fourier transform," *IEEE Signal Processing Lett.*, vol. 11, no. 1, Jan. 2004.