A Recursive SVD Algorithm for Array Signal Processing

by

Manuel Duarte Ortigueira(*) and Miguel-Angel Lagunas(**)

* Dep. TSC, ETSIB - UPC, Apdo. 30002, 08080 Barcelona, Spain and Instituto Superior Técnico, Av Rovisco Pais, 1000 Lisbon Portugal

** Dep. TSC, ETSIB - UPC, Apdo. 3002, 08080 Barcelona, Spain

Abstract
The usual algorithms for the implementation of algorithms like MUSIC or Min-Norm are based on the eigen decomposition of covariance matrix since SVD has been considered hard from a numerical point of view. Here, a SVD based algorithm is proposed which is equivalent to that one. The main difficult task of the algorithm resides in the computation of the singular values, which, however, can be computed in a parallel way. The squares of the singular values are the zeros of a rational function. Once the singular values are known, the corresponding singular vectors are easily computed, with 3 complex operations by component. The behaviour of the algorithm is illustrated.

1 - Introduction
In recent years there has been a great interest in adaptive array algorithms. This follows from the increasing application fields and success in several areas of adaptive arrays. The usual algorithms for the implementation of algorithms like MUSIC or Min-Norm are based on the eigen decomposition (ED) of the covariance matrix. The singular value decomposition (SVD) of the data matrix does not have being used, since the available algorithms are too expensive to be useful, mainly in a recursive implementation. This problem explains why most attention and contributions to the construction of recursive algorithms were done in covariance matrix based algorithms. The main contributions were done by Sharman [9], Yang and Kaveh [12], and Schreiber [11]. In [6,7] the authors proposed an alternative algorithm based on the spectral decomposition of the sum of a diagonal matrix and a dyad (rank one matrix). In that algorithm, we use the fact that the correlation matrix is updated as:

\[ R_n = \alpha_1 R_{n-1} + \left(1 - \alpha_1 \right) x_n x_n^H \]

where \( x_n \) is a vector representing the array measurements at a given instant \( n \) and \( H \) means conjugate transpose, to obtain a recursive algorithm to compute the spectral decomposition of \( R_n \) from the corresponding decomposition of \( R_{n-1} \). The problem is reduced to the computation of the spectral decomposition of the sum of a diagonal matrix with a hermitian dyad. The eigenvectors of this matrix are easily computed with only 1 real and 1 complex operations by component and the eigenvalues are the zeros of a rational function.

Here, we will consider a similar algorithm to obtain the singular value decomposition (SVD). This algorithm works directly with the data matrix \( X \). We assume it to be updated accordingly to the three steps:

a) The first column is deleted.
b) The other columns are shifted towards the left
c) The new snapshot is the last column.

It is equivalent to consider \( X \) as a sliding time window. In practice, we found better to modify slightly this procedure to actually obtain a recursive algorithm. Let \( x_1 \) be the 1st column of \( X_{n-1} \). And \( P \) a LxL permutation matrix, obtained from the identity matrix by a unity circular permutation of its columns,

\[ P_{ij} = \delta_{i,j-1} \quad i=1,...,L, \quad j=1,...,L-1 \]

\[ = \delta_{i,j-1} \quad i=1,...,L, \quad j=L \]

and \( x_n \) the new snapshot. The new matrix is formed...
accordingly to the recursion:

\[ X_n = X_{n-1} \cdot P + v_n \cdot e_n^T \]  \hspace{1cm} (3)

where

\[ v_n = X_n - x_1 \quad \text{and} \quad n>L \]  \hspace{1cm} (4)

and \( e_n^T \) is the transposed Lth column of the identity matrix.

Initially we do not slide the window, updating accordingly to:

\[ X_n = X_{n-1} + x_n \cdot e_n^T \quad n=1, \ldots, L \]  \hspace{1cm} (5)

The singular value decomposition has two steps. In the first, we reduce the main problem to the decomposition of the sum of a diagonal matrix with a dyad. The singular values are found by solving a bilinear equation, which roots are inside the disjoint intervals defined by the old singular values. This means that we can compute the new singular values in a parallel way.

Once the singular values are known, the corresponding singular vector is easily computed, with \( 3 \) complex operations by component.

The algorithm is used to implement Min-Norm method. Some examples will be presented to illustrate the performance of the algorithm in tracking a moving source.

2- The data model

Some usual assumptions are made to lead to analytically tractable algorithms. The medium is assumed to be isotropic and non dispersive in order to assure a straight-line propagation. Sources are assumed to be in the far-field region of the array, so that the radiation impinging on the array is a sum of plane waves. We are going to consider the situation in which we use an L-element linear array (the generalization to other arrays is easy) where plane waves from M narrowband sources impinge on. We assume, also, that the signals have the same center frequency \( \omega_0 \) and with complex envelopes that are slowly varying time functions. With these assumptions, the complex signal output at the sensor j is:

\[ x_j(t) = \sum_{i=1}^{M-1} a_j(\theta_i) \cdot s_i(t - \tau_j(\theta_i)) \]  \hspace{1cm} (6)

or

\[ x_j(t) = \sum_{i=1}^{M-1} a_j(\theta_i) \cdot s_i(t) \cdot e^{j\omega_0 \tau_j(\theta_i)} \]  \hspace{1cm} (7)

onde \( \tau_j(\theta_i) \) is the propagation delay between the 1st and jth sensors for the ith wavefront impinging on the array and coming from direction \( \theta_i \); \( a_j(\theta_i) \) is the jth sensor complex response at frequency \( \omega_0 \). Introducing the array response or array steering vector for direction \( \theta_i \):

\[ a(\theta_i) = [ a_j(\theta_i) \cdot e^{j\omega_0 \tau_j(\theta_i)} ] , \quad j=1, \ldots, L \]  \hspace{1cm} (8)

the complex envelopes vector

\[ s(t) = [ s_j(t) , \quad j=1, \ldots, M ] \]  \hspace{1cm} (9)

and the noise vector, \( n(t) \), assumed to be independent of the source signals, the measurement model becomes:

\[ x(t) = A(\theta) \cdot s(t) + n(t) \]  \hspace{1cm} (10)

where \( A(\theta) \) is a LxM matrix, with L>M. Its columns are elements of the called array manifold. This measurement model leads to the following autocorrelation:

\[ R = A(\theta) R_s A^H(\theta) + R_n \]  \hspace{1cm} (11)

where \( R_s \) and \( R_n \) are the autocorrelation of \( s(t) \) and \( n(t) \), assumed to be positive definite.

The ultimate objective is to find \( A(\theta) \). An intermediate objective is to find M linearly independent vectors contained in the subspace spanned by the steering vectors: the signal subspace. In the no noise case, those vectors are the M eigenvectors corresponding to the non zero eigenvalues of the autocorrelation matrix; in the noisy case, these vectors are the M generalized eigenvectors of \( R \) corresponding to the M largest eigenvalues. The other eigenvectors span the noise subspace. Once each subspace is identified, MUSIC or MIN-NORM methods can be used to identify the steering vectors. Alternatively to the eigendecomposition of the autocorrelation matrix, we can perform the SVD of the data measurement matrix, \( X \); the M singular vectors corresponding to the M largest singular values form a M rank matrix, which best approximates \( X \) in a least-squares sense. It is not difficult to see that ED and SVD give the same subspace estimate.

3- Recursive singular value decomposition

Let \( A_n \) a (NxM) matrix and \( v_n \) and \( w_n \) two vectors of dimensions N and M (N>M) with complex components. We are going to assume as having the SVD of \( R_{n-1} \):

\[ A_{n-1} = V_{n-1} \Sigma_{n-1} W_{n-1}^H \]  \hspace{1cm} (11)
The matrices $V_{n-1}$ and $W_{n-1}$ are normalized. If we add a dyad to $A_{n-1}$ we obtain a matrix $A_n$ which SVD we want:

$$A_n = V_n \Sigma_n W_n^H$$  \hspace{1cm} (12)

If we let

$$v = V_{n-1}^H v_n$$  \hspace{1cm} (13)

$$w = W_{n-1}^H w_n$$  \hspace{1cm} (14)

then, the problem is reduced to the SVD of:

$$D + v.w^H = S \cdot \Sigma \cdot U^H$$  \hspace{1cm} (15)

with $D = \Sigma_{n-1} = \text{Diag}(d_i \ i = 1, \ldots K-1)$ \hspace{0.5cm} $n \leq M$, con $d_K = 0$.

The solution of the problem is easily obtained (see appendix A) and it is given by:

$$u = [\sigma^2 - \Delta_M]^{-1} [D^T v + \rho \sigma w]$$  \hspace{1cm} (16)

$$s = [\sigma^2 - \Delta_N]^{-1} [\sigma v + \rho D w]$$  \hspace{1cm} (17)

where $u$ and $s$ are, respectively, the right and left singular vectors(1) and

$$\Delta_M = D^T.D \quad \text{and} \quad \Delta_N = D.D^T$$  \hspace{1cm} (18)

The singular values are the zeros of the following equation:

$$[w^H F_M D^T v - 1].[v^H F_N D w - 1] - \sigma^2 (w^H F_M w)(v^H F_N v) = 0$$  \hspace{1cm} (19)

where

$$F_\sigma = [\sigma^2 I - \Delta_\sigma]^{-1} \sigma = M \cdot N$$  \hspace{1cm} (20)

The resolution of (19) gives us $M$ singular values. The corresponding singular vectors are obtained from (16) and (17). The solution of (19) is the main task of the algorithm. To solve it we perform a partial fraction decomposition:

$$\sum_{i=1}^M \frac{\alpha_i}{\lambda - \delta_i} = 1 \quad \text{with} \quad \lambda = \sigma^2 \text{and} \quad \delta_i = \sigma_i^2$$  \hspace{1cm} (21)

If the residues $\alpha_i$ are of the same sign, there is one root between any two consecutive poles and one is either greater than the maximum pole or less than the minimum pole according to the sign of $\alpha_i$ (+/-).

This is the situation we find, respectively, during the construction of the bases, corresponding to eq. (5), or when we subtract a snapshot, reducing the rank (in this case, (19) has zero as one solution). This works well and the results are good. If the residues do not have the same sign, as it happens when we are using recursion (3), the number of roots inside each interval may be 0, 1, or 2. In this case, a solution can be found in two steps: identifying the number of roots in each interval, which is easy by looking at the signs of the residues and placing a pole between the two roots belonging to the same interval. This is conceptually very simple and it works. The major difficulty appears when a singular value does not change. In this case, a root is near a pole, but not on the pole. We could not find an efficient way of solving this problem, which forced us to process the two vectors in recursion (3) separately. This has the drawback of increasing the computational cost and, as a consequence, a decrease in the precision. To avoid, at least partially this problem, in our simulations we reset the matrices periodically (at every multiple of four times the snapshot length).

4- The FB recursive SVD

The implementation of the FB method in with the RSVD is very easy. We substitute each snapshot $v_n$ by $w_n = [v_n : J.v_n]^T$ (22)

with this modification the columns of matrix $V$ will verify the centrohermitian property and those of $W$ will be real. On the other hand, the rest of the algorithm will deal with real quantities. This is equivalent to enlarge artificially the dimension of the snapshot, as if we duplicate the number of sensors.

5. Experimental verification

Experimental framework - In order to perform an experimental evaluation of the algorithm, we simulated a system with the following characteristics :

Array length : $L = 8$ \hspace{0.5cm} active sources : 2 \hspace{0.5cm} Center frequency : 1MHz; magnitudes : 1; minimum signal to noise ratio per source : 10dB; method used for DOA calculation: min-norm method.

Simulation results - In fig.1 we present the results obtained with 1000 snapshots. The DOA were computed every 8 snapshots. The results are not very good, at least when compared with the results obtained
with the recursive eigendecomposition [7,8]. The possibility of better results remains depending on a better algorithm to solve eq. (19). This is under research.

6 - Conclusions

In this paper we proposed a new recursive algorithms for array processing. The proposed algorithm implements the SVD of the data matrix and it is equivalent to use an autocorrelation constructed with L snapshots, obtained with a moving window. The forward-backward version was also considered. This algorithm has a high degree of parallelism, which is suitable for on-line implementations. The discussion of the ED/SVD trade-off remains by now an open subject which will be a subject of another publication.

References


Appendix A

Let s and u be the left and right singular; we have

\[ [D + v.w^H]u = \sigma s \]

and

\[ s^H[D + v.w^H] = \sigma u^H \]

The solution of this system gives

\[ u = [\sigma^2 - \Delta_M]^{-1} [\alpha D^H v + \beta \sigma w] \]

\[ s = [\sigma^2 - \Delta_N]^{-1} [\alpha \sigma v + \beta D w] \]

where \( \alpha = w^H u \) and \( \beta = v^H s \) and \( \Delta_M \) and \( \Delta_N \) are given by (18) and \( F_\sigma \) by (20). We conclude easily that

\[ \alpha = \alpha(w^H F_M D^H v) + \beta \sigma(w^H F_M w) \]

\[ \beta = \sigma \sigma (v^H F_N v) + \beta (v^H F_M Dw) \]

This sistem has a non null solution if the determinant is zero. This gives equation (19) which allows us to obtain \( \lambda = \sigma^2 \). On the other hand, \( \rho = \beta/\alpha \) is obtained from (a5) or (a6):

\[ \rho = -[(w^H F_M D^H v) - 1]/(\sigma w^H F_M w) \]

\[ = -[(\sigma v^H F_N v)/(v^H F_N D) - 1]/[\sigma^2 (v^H F_N D) - 1] \]

the singular vectors being given, aside normalization factors by Eqs. (16) and (17) obtained from (a3 and a4).