Low-Rank Adaptive Filters

Peter Strobach, Senior Member, IEEE

Abstract—We introduce a class of adaptive filters based on sequential adaptive eigendecomposition (subspace tracking) of the data covariance matrix. These new algorithms are completely rank revealing, and hence, they can perfectly handle the following two relevant data cases where conventional recursive least squares (RLS) methods fail to provide satisfactory results: 1) highly oversampled “smooth” data with rank deficient or almost rank deficient covariance matrix and 2) noise-corrupted data where a signal must be separated effectively from superimposed noise. This paper contradicts the widely held belief that rank revealing algorithms must be computationally more demanding than conventional recursive least squares. A spatial RLS adaptive filter has a complexity of $O(N^2)$ operations per time step, where $N$ is the filter order. The corresponding low-rank adaptive filter requires only $O(Nr)$ operations per time step, where $r \leq N$ denotes the rank of the data covariance matrix. Thus, low-rank adaptive filters can be computationally less (or even much less) demanding, depending on the order/rank ratio $N/r$ or the compressibility of the signal. Simulation results substantiate our claims. This paper is the first in a series of papers devoted to the theory and application of fast orthogonal iteration and bi-iteration subspace tracking algorithms.

I. INTRODUCTION

CONVENTIONAL recursive least squares (RLS) techniques are based on two fatal assumptions: 1) They assume that the data covariance matrix has full rank $r = N$, and 2) they assume that the data is all signal and contains no noise. Unfortunately, these ideal conditions are seldom observed in practice. In communications applications and in sensor array processing, the signal is often buried in noise of considerable variance. In control systems applications, the rank $r$ of the signal covariance matrix is often smaller or much smaller than the number of observations or the filter order $N$. In these cases, the relevant information in the data can be mapped or “compressed” without much or no loss of information into a subspace of dimension $r \leq N$ spanned by the “eigensignals” or “natural modes” of the signal. This can be achieved using singular value decomposition (SVD) of the data matrix or eigenvalue decomposition (EVD) of the corresponding data covariance matrix. With this kind of “rank revealing” processing, an effective separation of signal and noise is achieved, and the necessary and sufficient dominant subspace dimension $r$ can be determined. In addition, a considerable complexity reduction can be achieved when the number of sensors or taps $N$ is much larger than the dominant subspace dimension $r$. Thus, the necessary amount of processing that must be expended in a specific application is no longer fixed but is a function of the “compressibility” of the given signal determined by the order/rank ratio $N/r$. These principles of rank revealing processing have been applied extensively in the area of high resolution spectral estimation comprising the Pisarenko [1], MUSIC [2], [3] and ESPRIT [4] algorithms, which all utilize the same principal signal/noise separation capability of a data eigendecomposition.

However, there are various other areas in signal processing where rank revealing processing can be applied successfully to generalize or enhance existing methods. The potential capabilities of rank revealing processing in several fundamental application areas have been studied in a more general context by Scharf and Tufts [5], who discussed the connections between rank reduction and rate distortion theory (e.g., the connection between rank reduction and information loss) and the use of rank reduction principles in the area of signal detection. Tufts and Kumaresan [6] have shown how the signal/noise separation capability of an eigendecomposition can be used effectively to enhance the performance of linear prediction techniques. Van Veen and Scharf have developed low rank detectors for Gaussian random vectors [7] and Strobach [8], [9] has shown how low-rank techniques can be applied for complexity reduction in the area of sensor array detection. A most profound discussion of basic low-rank techniques is available in a book by Scharf [10].

These low-rank techniques are first introduced on block processing examples. Classical EVD and SVD routines [11]–[14] are computationally demanding procedures, and they are block processing techniques. In signal processing, however, fast and computationally efficient sequential algorithms that require at each time step an exact or approximate EVD, SVD, or parts thereof are often required. Thus, a considerable effort has gone into the development of adaptive “subspace trackers.” An excellent survey of existing techniques in this area is given by Comon and Golub in [15]. More recent results are reported and discussed by Dowling et al. [16]. These subspace trackers were mainly developed to facilitate adaptive implementations of spectral estimation or source localization techniques such as “adaptive MUSIC” [17].

In this paper, we develop an adaptive filter theory based on rank revealing processing and subspace tracking. These low-rank adaptive filters project signals onto a dominant signal subspace of reduced rank rather than on the “complete” data subspace, as do conventional RLS adaptive filters. Thus, we completely remedy with problems that stem from oversampled or noise-corrupted data. Low-rank adaptive filters require, at each time step, some information about the dominant linear invariant data subspace. For this purpose, we develop a class
of highly efficient subspace trackers based on sequential orthogonal iteration [18]–[20]. These subspace trackers are quite naturally linked to a concept named the Schur pseudoinverse, which we use to approximate the more well-known Moore–Penrose pseudoinverse in the development of the low-rank adaptive filters. Low-rank adaptive filters with complexity \(O(Nr^2)\) and \(O(Nr)\) are obtained on this basis. These algorithms are completely selfstabilizing. They are started from unit spaces and are suitable for real-time processing because they are well structured, and most of the operations involved are elementary Givens plane rotations.

This paper is organized as follows. In Section II, we summarize the basic principles of adaptive filtering, least squares, EVD, SVD and rank, which are required in the following sections. We further discuss some special aspects of rank reduction in the area of adaptive filtering. In Section III, we develop some highly efficient new subspace tracking concepts based on sequential orthogonal iteration. We use these results to establish three basic variants of a low-rank adaptive filter. The results are compiled into three algorithm summaries. We further introduce a special variant called row subspace adaptive filters for low-rank adaptive filtering of time series. In Section IV, we discuss important practical issues such as adaptive subspace selection and subspace adaptive filtering of signals in correlated (non-white) Gaussian noise. Some computer experiments are provided in Section V to illustrate the behavior of the algorithms. Section VI lists the main conclusions of this paper.

II. FULL-RANK AND LOW-RANK ADAPTIVE FILTERING

In this section, we review the principles of RLS and rank reduction in adaptive filtering.

A. Principles of Recursive Least Squares

Recursive least squares (RLS) adaptive filters [21]–[23] can be named “full-rank” techniques because they project, at each time step, a signal vector \(y(t)\) orthogonally onto a subspace spanned by the \(N\) column vectors \(x_1(t), x_2(t), \cdots, x_N(t)\) of an \(L \times N\) data matrix \(X(t) = [x_1(t), x_2(t), \cdots, x_N(t)]\), where \(N\) is either the number of array sensors or the number of taps of the adaptive filter, and \(L\) is either a finite or infinite number of “snapshots” or a window length. Thus, the data matrix represents either spatial or temporal data. Define the orthogonal projection operator \(P_X(t)\) and its orthogonal complement \(P_X^\perp(t)\) as follows:

\[
P_X(t) = X(t)\Phi^{-1}(t)X^T(t),
\]

\[
P_X^\perp(t) = I - P_X(t)
\]  

where \(I\) is an \(L \times L\) identity matrix, and \(\Phi(t)\) is an \(N \times N\) sample covariance matrix:

\[
\Phi(t) = X^T(t)X(t).
\]

Assume that the inverse of \(\Phi(t)\) exists. Then, \(d(t)\) is the orthogonal projection of \(y(t)\) onto the column space of \(X(t)\), and \(e(t)\) is the orthogonal complement or “residual vector”:

\[
d(t) = P_X(t)y(t)
\]

\[
e(t) = P_X^\perp(t)y(t).
\]

Note from (1a) and (1b) that projection operators are complementary (i.e., they add up to the identity matrix), and therefore, \(y(t) = d(t) + e(t)\), where \(d(t)\) is usually a enhanced estimate of the signal in \(y(t)\). At each time step, an adaptive filter extracts only a single component of either \(d(t)\) or \(e(t)\). Usually, only the top components \(d(t)\) or \(e(t)\) are computed. These top components can be represented as inner products of “in-space” and residual vectors and an \(L \times 1\) top “pinning” vector \(z^T(t) = [1, 0 \cdots 0]\) as follows:

\[
d(t) = \pi^T(t)d(t)
\]

\[
e(t) = \pi^T(t)e(t).
\]

Note further that a pinning of the data matrix \(X(t)\) extracts the actual data snapshot vector \(z(t)\). A pinning of \(y(t)\) extracts the actual reference data sample \(y(t)\):

\[
z^T(t) = [x_1(t), x_2(t), \cdots, x_N(t)] = \pi^T(t)X(t)
\]

\[
y(t) = \pi^T(t)y(t).
\]

Thus, a pinning of (3a) gives

\[
d(t) = \pi^T(t)P_X(t)y(t) = z^T(t)\Phi^{-1}(t)X^T(t)y(t).
\]

Introduce the sample cross correlation vector \(c(t)\) and the adaptive filter weight vector \(a(t)\), both of dimension \(N\):

\[
c(t) = X^T(t)y(t),
\]

\[
a(t) = \Phi^{-1}(t)c(t).
\]

The top components \(d(t)\) and \(e(t)\) can be computed using a transversal filter or adaptive linear combiner of length \(N\) taps with coefficient vector \(a(t)\) and state vector \(z(t)\) as follows:

\[
d(t) = z^T(t)a(t)
\]

\[
e(t) = y(t) - z^T(t)a(t).
\]

In a spatial adaptive filter (array data case), the elements of the snapshot vector \(z(t)\) are sampled outputs of independent sources or sensors. In a temporal adaptive filter (serialized data case), the elements of \(z(t)\) are successive samples of a time series. In any case, an RLS adaptive filter algorithm basically consists of an adjustable filter or linear combiner ((9a) and (9b)) and an associated coefficient adaptation algorithm for updating the elements of the coefficient vector \(a(t)\). In the spatial adaptive filter case, the principal complexity for updating the coefficient vector is \(O(N^2)\) operations per time step. In the serialized data case, the principal complexity of the least squares parameter updating problem reduces to \(O(N)\). This has lead to the development of the class of “fast” \(O(N)\) RLS adaptive filters based on transversal [24], [25], ladder/lattice [26], [27], and QR-based algorithms [28].
B. Rank Reduction in Adaptive Filtering

The parameter or coefficient adaptation for full-rank least squares processing requires, at each time step, the solution of an inverse problem of the kind \( a(t) = \Phi^{-1}(t)e(t) \) as stated in (8). Even if the inverse covariance matrix \( \Phi^{-1}(t) \) is never formed explicitly by the algorithms, its existence at each time instant is a fundamental assumption in any RLS concept and is a necessary prerequisite for the correct operation of any conventional RLS algorithm. Unfortunately, it has been observed that the condition of an invertible \( \Phi(t) \) is frequently violated in practice. The situation is most critical in real-time applications involving large filter orders \( N \) and oversampled data. At time instants where \( \Phi(t) \) becomes defective, conventional RLS algorithms produce invalid results and may react in an unpleasant and unpredictable fashion. This fact constitutes one of the most severe drawbacks in the RLS concept. The only tentative way out of this difficulty is rank revealing processing where a defective \( \Phi(t) \) is assumed as the most general case. We introduce a class of adaptive filters based on an approximate dominant eigendecomposition of the data covariance matrix using a subspace tracking concept. The complexity of these low-rank adaptive filters grows only linearly with the model or filter order \( N \) as expected from fast algorithms. Furthermore, their complexity is a signal-dependent function of the rank \( r \) of \( \Phi(t) \). Begin with the exact EVD of \( \Phi(t) \) as follows:

\[
\Phi(t) = V(t)A(t)V^T(t) ;
\]

\[
V^T(t)V(t) = V(t)V^T(t) = I \tag{10}
\]

where \( V(t) = [v_1(t), v_2(t), \cdots, v_N(t)] \) is the \( N \times N \) orthonormal matrix of eigenvectors, and \( A(t) = \text{diag}(\lambda_1(t), \lambda_2(t), \cdots, \lambda_N(t)) \) is the \( N \times N \) diagonal matrix of eigenvalues. In the sequel, we assume that the eigenvalues in \( A(t) \) appear in descending order of magnitude:

\[
\lambda_1(t) \geq \lambda_2(t) \geq \cdots \geq \lambda_r(t) > \lambda_{r+1}(t) = \cdots = \sigma^2(t) = \lambda_N(t). \tag{11}
\]

The information in the data is represented by the \( r \) dominant eigenvalues that exceed the “noise floor level” \( \sigma^2(t) \). In the outset, we assume that the noise is white. Additional whitening techniques are required in the nonwhite Gaussian noise case. This extension is discussed in Section IV. The signal of interest can be concentrated or compressed in a “dominant signal subspace” of dimension \( r \leq N \). This dominant signal subspace is represented by the first \( r \) eigenvectors \( V_r(t) = [v_1(t), v_2(t), \cdots, v_r(t)] \) and the associated eigenvalues \( A_r(t) = \text{diag}(\lambda_1(t), \lambda_2(t), \cdots, \lambda_r(t)) \) as follows:

\[
V(t) = [V_r(t) \mid V_{N-r}(t)] \tag{12a}
\]

\[
A(t) = \begin{bmatrix} A_r(t) & 0 \\
0 & A_{N-r}(t) \end{bmatrix}. \tag{12b}
\]

It will be shown that adaptive filtering can be posed as an optimization problem in \( V_r(t) \) alone. With this kind of low rank processing, one can simultaneously solve the following three most fundamental and pressing problems in RLS adaptive filtering:

1) The rank deficiency problem of conventional RLS adaptive filtering is solved because we can always determine \( r \) sufficiently small so that the remaining dominant signal subspace is nondefective.

2) Low-rank adaptive filters have an inherent signal/noise separation capability that leads to almost unbiased estimates of the adaptive filter parameters in cases of low SNR data.

3) Low-rank processing leads to a complexity reduction because adaptation is carried out in a subspace of smaller or much smaller dimension. This argument especially holds for spatial adaptive filters.

We next develop the necessary relationships for low-rank adaptive filtering using projections of the data onto a dominant signal subspace. In the outset, we assume exact eigendecompositions. Consider the case where \( \Phi(t) \) can be represented as the sum of a rank \( r \) approximant \( \Phi_r(t) \), which carries the information about the signal of interest, and a complementary approximation error matrix \( \Phi_E(t) \) as follows:

\[
\Phi(t) = \Phi_r(t) + \Phi_E(t) \tag{13}
\]

where \( \Phi_r(t) = V_r(t)A_r(t)V_r^T(t) \). Define the Moore–Penrose pseudo-inverse \( \Phi_r^+(t) \) [12], [14]:

\[
\Phi_r^+(t) = V_r(t)A_r^{-1}(t)V_r^T(t). \tag{14}
\]

The eigenvalues in \( A_r(t) \) appear in descending order for any signal of nonvanishing power, one finds a sufficiently small positive \( r \) such that \( A_r^{-1}(t) \) exists. A low-rank coefficient vector \( a^+(t) \) is next defined by replacing the inverse covariance matrix in (8) by the Moore–Penrose pseudoinverse as follows:

\[
a^+(t) = \Phi_r^+(t)e(t) = V_r(t)A_r^{-1}(t)V_r^T(t)e(t). \tag{15}
\]

We obtain low-rank in-space and residual vectors \( d^+(t) \) and \( e^+(t) \), respectively:

\[
d^+(t) = X(t)a^+(t) = X(t)V_r(t)A_r^{-1}(t)V_r^T(t)e(t) \tag{16a}
\]

\[
e^+(t) = y(t) - d^+(t). \tag{16b}
\]

These in-space and residual vectors are next defined as projections onto a dominant signal subspace. For this purpose, introduce the skinny SVD of \( X(t) \):

\[
X(t) = U(t)S(t)V^T(t) \tag{17}
\]

where \( U(t) \) is the \( L \times N \) orthonormal matrix of left singular vectors, and \( V(t) \) is the \( N \times N \) orthonormal matrix of right singular vectors that are identical to the eigenvectors of \( \Phi(t) \) according to (10). According to (12a) and (12b), this SVD is partitioned as follows:

\[
X(t) = [U_r(t) \mid U_{N-r}(t)] \begin{bmatrix} S_r(t) & 0 \\
0 & S_{N-r}(t) \end{bmatrix}
\begin{bmatrix} V_r^T(t) \\
V_{N-r}^T(t) \end{bmatrix}. \tag{18}
\]
This expression can be used to derive the following relationships:
\[
\begin{align*}
X(t)V_r(t) &= U_r(t)S_r(t) \\
V_r^T(t)c(t) &= V_r^T(t)X^T(t)y(t) = S_r(t)U_r^T(t)y(t).
\end{align*}
\] (19a) (19b)

Substitute (19a) and (19b) into (16a) to find:
\[
\begin{align*}
d^+(t) &= U_r(t)S_r(t)A_r^{-1}(t)S_r(t)U_r^T(t)y(t) \\
&= U_r(t)U_r^T(t)y(t).
\end{align*}
\] (20)

Introduce a projection operator \( P_r(t) = U_r(t)U_r^T(t) \) that projects vectors orthogonally onto the dominant left singular subspace spanned by the columns of \( U_r(t) \) to realize that \( d^+(t) \) is just the orthogonal projection of \( y(t) \) onto the dominant left singular subspace, and \( e^+(t) \) is the associated orthogonal complement:
\[
\begin{align*}
d^+(t) &= P_r(t)y(t) \\
e^+(t) &= P_r^+(t)y(t) = (I - P_r(t))y(t).
\end{align*}
\] (21a) (21b)

### III. LOW-RANK ADAPTIVE FILTERING USING SEQUENTIAL ORTHOGONAL ITERATION

In this section, we develop the necessary relationships for a class of fast subspace trackers based on sequential orthogonal iteration. Moreover, we develop the basic theory for low-rank adaptive filtering using these subspace trackers.

**A. Classical Orthogonal Iteration and the Schur Pseudo-inverse**

Define an \( N \times r \) recursion matrix \( Q(t) \) with orthonormal columns. Further, define
\[
Q(t)R(t) = \Phi(t)Q(t-1)
\] (22)
where \( Q(t) \) and \( R(t) \) are the factors of a skinny QR decomposition of the matrix product \( A(t) = \Phi(t)Q(t-1) \). This concept leads to the following recurrence known as simultaneous orthogonal iteration [14, 18–21]:
\[
\begin{align*}
A(t) &= \Phi(t)Q(t-1) \\
A(t) &= Q(t)R(t) : QR factorization.
\end{align*}
\] (23a) (23b)

Simultaneous orthogonal iteration originates as a special case of Bauer’s classical bi-iteration for calculating linear invariant subspaces and “characteristic numbers” (singular values) of matrices [29]. See also [14] and [20] for details. Provided only that \( \Phi(t) \) does not change with time, one can show (see Stewart [20]) that the sequence of recursion matrices \( Q \) produced by recurrence (23a) and (23b) will converge towards the matrix of dominant eigenvectors, and the sequence of triangular matrices \( R \) will converge towards the diagonal matrix of dominant eigenvalues. Owsley [30] was probably the first who proposed an explicit variant of simultaneous orthogonal iteration on this basis for sequential tracking of dominant subspaces in a signal processing application. In our case, both \( \Phi(t) \) and \( c(t) \) are slowly varying functions of time because they are updated continuously according to
\[
\begin{align*}
\Phi(t) &= \alpha \Phi(t-1) + (1-\alpha)z(t)z^T(t) \\
c(t) &= \alpha c(t-1) + (1-\alpha)z(t)y(t)
\end{align*}
\] (24a) (24b)

where \( \alpha \) is a positive exponential forgetting factor close to 1. In this case, the orthogonal iteration (23a) and (23b) is a device that tracks the dominant eigensubspace \( V_r(t) \) and the associated dominant eigenvalues in \( A_r(t) \). Consequently, a matrix \( \hat{\Phi}_r(t) \) defined as
\[
\hat{\Phi}_r(t) = Q(t)R(t)Q^T(t-1)
\] (25)
will tend to approximate the low-rank covariance matrix \( \Phi_r(t) = V_r(t)A_r(t)V_r^T(t) \). Moreover, the Moore–Penrose pseudoinverse of (14) can be approximated by the Schur pseudoinverse \( \hat{\Phi}_r^+(t) \):
\[
\hat{\Phi}_r^+(t) = Q(t-1)R^{-1}(t)Q^T(t).
\] (26)

Note that \( R^{-1}(t) \) always exists, provided only that the rank parameter \( r \) has been chosen sufficiently small. Introduce a projection operator \( P_Q(t) = Q(t)Q^T(t) \) that projects vectors orthogonally onto the column space of \( Q(t) \) to demonstrate that the Schur pseudoinverse satisfies the following relations:
\[
\begin{align*}
\Phi_r(t)\hat{\Phi}_r^+(t) &= P_Q(t) \\
\hat{\Phi}_r^+(t)\Phi_r(t) &= P_Q(t-1).
\end{align*}
\] (27a) (27b)

Moreover, it is straightforward to prove that the following Moore–Penrose conditions hold:
\[
\begin{align*}
\Phi_r(t)\hat{\Phi}_r^+(t)\Phi_r(t) &= \hat{\Phi}_r(t) \\
\hat{\Phi}_r^+(t)\Phi_r(t)\hat{\Phi}_r^+(t) &= \hat{\Phi}_r^+(t).
\end{align*}
\] (28a) (28b)

\[
\begin{align*}
(\Phi_r(t)\hat{\Phi}_r^+(t))^T &= \Phi_r(t)\hat{\Phi}_r^+(t) \\
(\hat{\Phi}_r^+(t)\Phi_r(t))^T &= \hat{\Phi}_r^+(t)\Phi_r(t).
\end{align*}
\] (28c) (28d)

Replace the Moore–Penrose pseudoinverse in (15) by the Schur pseudoinverse to obtain the following expression for the adaptive filter parameter vector:
\[
\hat{a}^+(t) = \hat{\Phi}_r^+(t)c(t) = Q(t-1)R^{-1}(t)Q^T(t)c(t).
\] (29)

Next, replace \( a^+(t) \) by \( \hat{a}^+(t) \) in (16a). This yields the following expressions for a low-rank adaptive filter based on the Schur pseudoinverse concept:
\[
\begin{align*}
\hat{d}^+(t) &= z^T(t)Q(t-1)R^{-1}(t)Q^T(t)c(t) \\
\hat{e}^+(t) &= y(t) - \hat{d}^+(t).
\end{align*}
\] (30a) (30b)

Define “compressed” \( r \times 1 \) vectors \( h(t) \), \( g(t) \), and \( g^*(t) \) as follows:
\[
\begin{align*}
h(t) &= Q^T(t-1)z(t) \\
g(t) &= Q^T(t-1)c(t) \\
g^*(t) &= Q^T(t)c(t).
\end{align*}
\] (31a) (31b) (31c)

The adaptive filter for the in-space component (30a) can be expressed in terms of \( h(t) \) and \( g^*(t) \):
\[
\hat{d}^+(t) = h^T(t)R^{-1}(t)g^*(t).
\] (32)

Define an auxiliary \( r \times 1 \) vector \( p(t) = R^{-1}(t)g^*(t) \) to see that the adaptive filter (32) can be computed in two steps. First, solve \( R(t)p(t) = g^*(t) \) for \( p(t) \) using simple \( r \times r \) back substitution. Finally compute \( \hat{d}^+(t) = h^T(t)p(t) \). Observe that
in (31a)-(31c), the recursion matrix $Q$ acts as a “data compressor” on both $z(t)$ and $c(t)$. In particular, $g(t)$ can be named an \textit{a priori} compressed correlation vector because $Q(t - 1)$ prior to updating is used for compaction. Consequently, $g^*(t)$ is the corresponding \textit{a posteriori} compressed correlation vector based on the updated basis matrix $Q(t)$.

### B. A Decomposition of Projections and Fast Sequential Orthogonal Iteration

An important issue is the efficient time updating of basis matrices $Q$. We develop a class of fast and ultrafast subspace trackers that accomplish this task with $O(Nr^2)$ and $O(Nr)$ operations per time step, respectively. A key step toward an algorithm for fast subspace tracking using orthogonal iteration is the orthogonal projection of the actual recursion matrix $Q(t)$ onto the previous (one time step delayed) subspace spanned by the columns of $Q(t - 1)$. Hereby, $Q(t)$ is decomposed into an in-space component that represents the “old” information in $Q(t)$ and an orthogonal complement subspace $\Delta(t)$ of dimension $N \times r$, which represents the innovation in $Q(t)$ based on the actual observation $z(t)$:

$$Q(t) = P_{Q(t - 1)}Q(t) + \Delta(t).$$  \hspace{1cm} (33) 

Since $\Delta(t)$ is orthogonal with respect to the column space of $Q(t - 1)$, we must have

$$Q^T(t - 1)\Delta(t) = 0.$$ \hspace{1cm} (34)

Introduce an $r \times r$ matrix $\Theta(t)$ of cosines of angles between subsequent subspaces as follows:

$$\Theta(t) = Q^T(t - 1)Q(t).$$ \hspace{1cm} (35)

Verify that the in-space component of $Q(t)$ can be expressed as a rotated version of $Q(t - 1)$ because $P_{Q(t - 1)}Q(t) = Q(t - 1)Q^T(t - 1)Q(t) = Q(t - 1)\Theta(t)$. The cosine matrix $\Theta(t)$ acts as a rotor on $Q(t - 1)$. Thus, (33) is finally expressed as

$$Q(t) = Q(t - 1)\Theta(t) + \Delta(t).$$ \hspace{1cm} (36)

This expression plays a key role in fast subspace updating. Substitute the time update equation (24a) into the “mapping equation” (23a) of orthogonal iteration to obtain

$$A(t) = [\alpha\Phi(t - 1) + (1 - \alpha)z(t)h^T(t)]Q(t - 1) = \alpha\Phi(t - 1)Q(t - 1) + (1 - \alpha)z(t)h^T(t).$$ \hspace{1cm} (37)

Substitute $Q(t - 1) = Q(t - 2)\Theta(t - 1) + \Delta(t - 1)$ according to (36), and note further that $\Phi(t - 1)Q(t - 2) = A(t - 1)$ according to (23a). This yields

$$A(t) = \alpha A(t - 1)\Theta(t - 1) + \alpha\Phi(t - 1)\Delta(t - 1) + (1 - \alpha)z(t)h^T(t).$$ \hspace{1cm} (38)

Only the term $\Phi(t - 1)\Delta(t - 1)$ in this expression requires $O(N^2)$ operations. In the Appendix, we demonstrate that this term has in fact very little influence on the overall recursion and can be neglected without any performance penalty. This results in the following $O(N^2)$ recursion for a direct time updating of auxiliary matrices $A$:

$$A(t) = \alpha A(t - 1)\Theta(t - 1) + (1 - \alpha)z(t)h^T(t).$$ \hspace{1cm} (39)

This recursion constitutes a basis for a first low-rank adaptive filter named LORAF 1, which is summarized in Table 1. The necessary operations for each formula are counted in terms of multiply-accumulate “mac” operations. A division counts as one “mac.” Finally, we may verify that LORAF 1 is an unconditionally stable algorithm. To see this, we investigate the structure of the time update of the auxiliary matrix $A$ according to (39). This time update has the structure of a matrix-valued feedback system with feedback matrix $\alpha\Theta(t - 1)$ and innovation $(1 - \alpha)z(t)h^T(t)$. Clearly, this feedback system must be unconditionally stable for $0 \leq \alpha \leq 1$ because from (36) and (34), it follows immediately that $\Theta(t - 1)$ is, in general, a norm-reducing (contractive) rotor:

$$\Theta^T(t)\Theta(t) = I - \Delta^T(t)\Delta(t).$$ \hspace{1cm} (40)

### C. Direct QR Factor Updating and the Algorithm LORAF 2

A large number of operations in LORAF 1 is expended in the explicit updating and QR factorization of $A(t)$. Therefore, we next develop a way recursive scheme to truncate $Q$ and $R$ factors of $A(t)$ separately in time. Note that this separate QR factor tracking implies no further simplifications on the algorithm. To aid in the development, replace $A(t)$ and $A(t - 1)$ in (39) by their corresponding QR factors as follows:

$$Q(t)R(t) = \alpha Q(t - 1)R(t - 1)\Theta(t - 1) + (1 - \alpha)z(t)h^T(t).$$ \hspace{1cm} (41)

Introduce the complement of the orthogonal projection of $z(t)$ onto the column space of $Q(t - 1)$:

$$z_{\perp}(t) = P_{Q(t - 1)}z(t) = (I - P_{Q(t - 1)})z(t) = z(t) - Q(t - 1)h(t).$$ \hspace{1cm} (42)

Normalize $z_{\perp}(t)$

$$z_{\perp}(t) = Z^{-1/2}(t)z_{\perp}(t) = Z_{\perp}(t)z(t) - Z_{\perp}(t)Q(t - 1)h(t).$$ \hspace{1cm} (43)

where $Z(t) = z(t)h(t)$. The actual data vector $z(t)$ can be decomposed into a component that can be represented in the “old” subspace spanned by the columns of $Q(t - 1)$ and a normalized orthogonal innovation $z_{\perp}(t)$ as follows:

$$z(t) = Z_{\perp}(t)z_{\perp}(t) + Q(t - 1)h(t).$$ \hspace{1cm} (44)

Substitute this representation of $z(t)$ into the direct QR factor time update (41) to obtain

$$Q(t)R(t) = \alpha Q(t - 1)R(t - 1)\Theta(t - 1) + (1 - \alpha)Z_{\perp}(t)h^T(t) + \alpha Q(t - 1)h(t)h^T(t) + (1 - \alpha)Q(t - 1)\Delta(t - 1) + (1 - \alpha)z(t)h^T(t)$$

$$= Q(t - 1)[\alpha R(t - 1)\Theta(t - 1) + (1 - \alpha)h(t)h^T(t)]$$
TABLE I
LOW-RANK ADAPTIVE FILTER LORAF 1. EQUATIONS NUMBERED AS THEY APPEAR IN THE TEXT. THE RANK

<table>
<thead>
<tr>
<th>Variable</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize:</td>
<td>$Q(t-1) = \begin{bmatrix} 1 \ \emptyset \end{bmatrix}$; $\Theta(t-1) = I$; $c(t-1) = \begin{bmatrix} 0 \ 0 \end{bmatrix}$; $0 \leq \alpha \leq 1$; $r$</td>
</tr>
</tbody>
</table>

FOR EACH TIME STEP DO:

**Subspace Tracking Section:**

- $h(t) = Q^T(t-1) z(t)$
- $A(t) = \alpha A(t-1) \Theta(t-1) + (1-\alpha) z(t) h^T(t)$
- $A(t-1) = Q(t) R(t) :\text{ QR factorization}$
- $\Theta(t) = Q^T(t-1) Q(t)$

**Low Rank Adaptive Filter Section:**

- $c(t) = \alpha c(t-1) + (1-\alpha) z(t) y(t)$
- $G^*(t) = Q^T(t) c(t)$
- $R(t) p(t) = G^*(t)$ \text{ back substitution} \quad p(t)$
- $\hat{z}^*(t) = h^T(t) p(t)$ : In-space component
- $\hat{z}(t) = y(t) - \hat{z}^*(t)$ : Residual

**Expression:**

$$f(t) = Q^T(t) \tilde{z}(t).$$ (48)

The key result is that the QR representation of $A(t)$ can be posed directly in terms of the augmented and updated elements of the QR representation of $A(t-1)$. Thus, we can introduce a sequence of Givens plane rotations $G(t)$ and split (45) into the following two recursions:

$$[R(t)] = G(t) \begin{bmatrix} \alpha R(t-1) \Theta(t-1) + (1-\alpha) h(t) h^T(t) \\ (1-\alpha) Z^{1/2}(t) h^T(t) \end{bmatrix}$$ \quad \text{(46a)}

$$[Q(t) | q(t)] = [Q(t-1) | \tilde{z}(t)] G^T(t).$$ \quad \text{(46b)}

The plane rotations in $G(t)$ must be determined so that the augmented and updated old triangular matrix in (46a) is transformed into a new upper-right triangular matrix. Equation (46b) is a first variant of a direct time updating recursion for the orthonormal basis matrix. We shall develop a second interesting alternative form of this time update. For this purpose, investigate the structure of the multiple Givens plane rotation matrix. From (46b), we obtain

$$G^T(t) = \begin{bmatrix} Q^T(t-1) \\ \tilde{z}^*(t) \end{bmatrix} [Q(t) | q(t)]$$

$$= \begin{bmatrix} \Theta(t) \\ f^T(t) \end{bmatrix} Q^T(t-1) g(t)$$

where $f(t) = Q^T(t) \tilde{z}(t)$. It turns out that $G^T(t)$ contains $\Theta(t)$ as a submatrix. Thus, we can substitute (47) into (46b), which results in the following alternative time updating recursion of the orthonormal matrix:

$$Q(t) = Q(t-1) \Theta(t) + \tilde{z}(t) f^T(t).$$ (49)

Close the circle of considerations by comparing (49) with the basic time updating structure obtained in (36). Clearly, the rank-one innovations matrix $\Delta(t)$ in (36) is constituted by

$$\Delta(t) = \tilde{z}(t) f^T(t).$$ (50)

We next develop a fast updating scheme for $g^*(t)$. Consider a premultiplication of both sides of (24b) by $Q^T(t-1)$, which gives $Q^T(t-1) e(t) = \alpha Q^T(t-1) e(t-1) + (1-\alpha) Q^T(t-1) z(t) y(t)$. A comparison of the terms in this expression with the compressed vectors defined in (31a)–(31c) yields

$$g(t) = \alpha g^*(t-1) + (1-\alpha) h(t) y(t).$$ (51)

Moreover, we can show that $g^*(t)$ can be obtained from $g(t)$ by rotation. To see this, postmultiply a transposed version of (46b) by $e(t)$, which gives

$$Q^T(t) c(t) = G(t) \begin{bmatrix} Q^T(t-1) c(t) \\ \tilde{z}^*(t) c(t) \end{bmatrix}.$$ (52)

Clearly, $Q^T(t) c(t) = g^*(t)$, and $Q^T(t-1) c(t) = g(t)$; therefore

$$\begin{bmatrix} g^*(t) \\ Q^T(t) c(t) \end{bmatrix} = G(t) \begin{bmatrix} g(t) \\ \tilde{z}^*(t) c(t) \end{bmatrix}.\quad \text{(53)}$$
TABLE II
LOW-RANK ADAPTIVE FILTER LORAF 2. EQUATIONS NUMBRED AS THEY APPEAR IN THE
TEXT. THE RANK VARIABLE \( r \) IS A FIXED PARAMETER. "*" DENOTES UNUSED QUANTITIES

\[
\begin{align*}
\text{Initialize:} & \quad \mathbf{Q}(t-1) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} ; \quad \mathbf{c}(t-1) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} ; \quad \mathbf{g}^*(t-1) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} ; \quad \mathbf{R}(t-1) = 0 ; \quad \Theta(t-1) = 1 \\
0 \leq \alpha \leq 1 ; r
\end{align*}
\]

FOR EACH TIME STEP DO:

\[
\begin{align*}
\text{Subspace Tracking Section:} & \\
\mathbf{h}(t) &= \mathbf{Q}(t-1)^T \mathbf{z}(t) \\
\mathbf{z}_z(t) &= \mathbf{z}(t) - \mathbf{Q}(t-1) \mathbf{h}(t) \\
\mathbf{Z}(t) &= \mathbf{Z}_z(t)^T \mathbf{z}_z(t) \\
\mathbf{F}_z(t) &= \mathbf{Z}^{-1/2}(t) \mathbf{z}_z(t) \\
\mathbf{R}(t) &= \mathbf{G}(t) \begin{bmatrix} \alpha \mathbf{R}(t-1) + (1-\alpha) \mathbf{h}(t)^T \mathbf{h}(t) \\ (1-\alpha) \mathbf{Z}^{1/2}(t) \mathbf{h}(t) \\ \mathbf{r}^3 + 3r^2 + r \\ r(r+1)(2r+1) \end{bmatrix} \text{ mac (46a)} \\
\mathbf{G}(t) &= \begin{bmatrix} \Theta(t) \\ \mathbf{f}(t) \end{bmatrix} \text{ rot (47)} \\
\mathbf{Q}(t) &= \mathbf{Q}(t-1) \Theta(t) + \mathbf{F}_z(t) \mathbf{f}(t) \quad \text{Nr}^2 + \text{Nr} \quad \text{mac (49)}
\end{align*}
\]

Low Rank Adaptive Filter Section:
\[
\begin{align*}
\mathbf{c}(t) &= \frac{\alpha \mathbf{c}(t-1) + (1-\alpha) \mathbf{z}(t) \mathbf{y}(t)}{2N + 1} \quad \text{mac (24b)} \\
\mathbf{g}(t) &= \frac{\alpha \mathbf{g}^*(t-1) + (1-\alpha) \mathbf{h}(t) \mathbf{y}(t)}{2r + 1} \quad \text{mac (51)}
\end{align*}
\]

\[
\begin{align*}
\mathbf{G}(t) &= \begin{bmatrix} \mathbf{g}(t) \\ \mathbf{z}_z(t)^T \mathbf{c}(t) \end{bmatrix} \quad \text{mac (53)} \\
\mathbf{p}(t) &= \mathbf{g}^*(t) \quad \text{back substitution rot (53)} \\
\mathbf{e}^*(t) &= \mathbf{y}(t) - \mathbf{h}^*(t) \quad \text{Residual (30b)}
\end{align*}
\]

Hence, only \( \mathbf{h}(t) \) must be computed explicitly from the input data \( \mathbf{z}(t) \) using data compression according to (31a). In the operations count of each recursion, "rot" indicates the number of \( 2 \times 2 \) elementary Givens plane rotations.

D. Ultrafast Subspace Tracking and the Algorithm LORAF 3

The subspace tracker in LORAF 2 is already considerably fast because it requires only a single \( O(Nr^2) \) operation in the computation of the matrix product \( \mathbf{Q}(t-1) \Theta(t) \) in (49). Thus, the key to further computational savings must come from an inspection of \( \Theta(t) \). This matrix represents the distance between consecutive subspaces. When the exponential forgetting factor \( \alpha \) is relatively close to 1, this distance will be small, and \( \Theta(t) \) will tend to an identity matrix. Practical experience has shown that we can replace \( \Theta(t-1) \) in the triangular matrix update (16a) by an \( r \times r \) identity matrix with little or no performance penalty. The constraint \( \Theta(t-1) := I \) simplifies the time updating of subspaces considerably. Realize that now the original time update (46a) reduces to

\[
\begin{align*}
\mathbf{R}(t) &= \mathbf{G}(t) \begin{bmatrix} \alpha \mathbf{R}(t-1) + (1-\alpha) \mathbf{h}(t) \mathbf{h}^T(t) \\ (1-\alpha) \mathbf{Z}^{1/2}(t) \mathbf{h}^T(t) \end{bmatrix} \\
&\quad \text{mac (54)}
\end{align*}
\]

The innovation in this update has only rank 1, and therefore, we can reduce to a new upper-right triangular matrix \( \mathbf{R}(t) \) with only \( (2r-1) \) orthonormal Givens plane rotations in \( \mathbf{G}(t) \). The rotations in this ultrafast reduction are illustrated as follows in an example of \( r = 5 \), where "*" denotes a nonzero matrix element, and "\( \theta \)" denotes a matrix element annihilated by rotation:

**STEP 1:** Reduce to upper triangular plus subdiagonal in \( (r-1) \) row rotations:

\[
\begin{align*}
\mathbf{G}_1(t) &= \begin{bmatrix} \alpha \mathbf{R}(t-1) + (1-\alpha) \mathbf{h}(t) \mathbf{h}^T(t) \\ (1-\alpha) \mathbf{Z}^{1/2}(t) \mathbf{h}^T(t) \end{bmatrix} \quad \text{mac (54)} \\
&\quad \begin{bmatrix} \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} \\ \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} \\ \mathbf{O} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} \\ \mathbf{O} & \mathbf{O} & \mathbf{X} & \mathbf{X} & \mathbf{X} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{X} & \mathbf{X} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{X} \end{bmatrix} \quad \begin{bmatrix} \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} \\ \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} \\ \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} \\ \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} \\ \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} \\ \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} \end{bmatrix}
\end{align*}
\]
TABLE III
ULTRAFAST LOW-RANK ADAPTIVE FILTER LORAF. 3. EQUATIONS NUMBERED AS THEY APPEAR IN THE TEXT. G(t) IS A SEQUENCE OF ONLY (2r - 1) GIVEN PLANE ROTATIONS (SEE THE DETAILED DISCUSSION IN THE TEXT). "*" DENOTES UNUSED QUANTITY

<table>
<thead>
<tr>
<th>Initialize:</th>
<th>$Q(t-1) = \begin{bmatrix} 1 &amp; 0 \ 0 &amp; 0 \end{bmatrix}$; $c(t-1) = \begin{bmatrix} 0 \ 0 \end{bmatrix}$; $g^*(t-1) = \begin{bmatrix} 0 \ 0 \end{bmatrix}$; $R(t-1) = 0$; $0 \leq \alpha \leq 1$; $r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOR EACH TIME STEP DO:</td>
<td></td>
</tr>
<tr>
<td>Input:</td>
<td>$z(t)$; $y(t)$</td>
</tr>
<tr>
<td>Subspace Tracking Section:</td>
<td></td>
</tr>
<tr>
<td>$h(t) = Q^T(t-1) z(t)$</td>
<td>$N_r$ mac (31a)</td>
</tr>
<tr>
<td>$z_1(t) = z(t) - Q(t-1) h(t)$</td>
<td>$N_r$ mac (42)</td>
</tr>
<tr>
<td>$Z(t) = Z^{2r-2}_1(t) z_1(t)$</td>
<td>$N$ mac (43)</td>
</tr>
<tr>
<td>$\bar{z}_1(t) = Z^{-1/2}(t) z_1(t)$</td>
<td>$N$ mac (43)</td>
</tr>
<tr>
<td>$G(t) \begin{bmatrix} \alpha R(t-1) + (t-\alpha) h(t) h^T(t) \ (1-\alpha) Z^{-1/2}(t) h^T(t) \end{bmatrix}$</td>
<td>$N$ mac (43)</td>
</tr>
<tr>
<td>$Q(t) = \begin{bmatrix} Q(t-1) &amp; \bar{z}_1(t) \end{bmatrix} G^T(t)$</td>
<td>$2N r - N$ rot (46b)</td>
</tr>
<tr>
<td>Low Rank Adaptive Filter Section:</td>
<td></td>
</tr>
<tr>
<td>$c(t) = \alpha c(t-1) + (t-\alpha) z(t) y(t)$</td>
<td>$2N + 1$ mac (24b)</td>
</tr>
<tr>
<td>$g(t) = \alpha g^*(t-1) + (t-\alpha) h(t) y(t)$</td>
<td>$2r + 1$ mac (51)</td>
</tr>
<tr>
<td>$\begin{bmatrix} g^*(t) \ \ast \end{bmatrix} = G(t) \begin{bmatrix} g(t) \ -\bar{z}^T(t) c(t) \end{bmatrix}$</td>
<td>$N$ mac (53)</td>
</tr>
<tr>
<td>$R(t) p(t) = g^*(t)$ back substitution $\rightarrow$ $p(t)$</td>
<td>r$^2$ mac</td>
</tr>
<tr>
<td>$\hat{a}^t(t) = h^T(t) p(t)$: In-space component</td>
<td>r mac</td>
</tr>
<tr>
<td>$\hat{a}^t(t) = y(t) - \hat{a}^t(t)$: Residual</td>
<td>1 mac (30b)</td>
</tr>
</tbody>
</table>

**STEP 2:** Reduce to upper triangular in ($r$) row rotations:

$$G_2(t)$$

All elementary rotations used in this reduction are of the type “annihilate bottom component by circular rotation.” Thus, the basic structure of these rotations is

$$\begin{bmatrix} x_1' \\ x_2' = 0 \end{bmatrix} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$  (55)

where $c = x_1/\rho$, $s = x_2/\rho$, and $\rho = (x_1^2 + x_2^2)^{1/2}$. Of course, it would be very unwise if we would accumulate these rotors in a multiple rotation matrix $G(t) = G_2(t) G_1(t)$ here because this would require that we finally update $Q(t)$ in time using the $\Theta(t)$ and $f(t)$ based scheme (49), and the complexity would stay at $O(Nr^2)$. Observe that $\Theta(t)$ does not vanish in (49), although we assumed $\Theta(t-1) = I$ in (46a). Hence, the appropriate time updating scheme here is the rotational $Q$-update (46b), where the rotors are applied directly onto the old augmented orthonormal basis matrix. This yields an ultrafast and well-structured $O(Nr)$ subspace tracker, we use for low-rank adaptive filtering in a third algorithm named LORAF3. Table III is a summary of LORAF3.

**E. Row Subspace Adaptive Filters**

The algorithms discussed in the previous sections can be named “column subspace adaptive filters” because they project vectors $g(t)$ onto the dominant column subspace of a data matrix according to (20). A particularly interesting alternative to this approach is row subspace adaptive filtering, where the snapshot vector $z(t)$ is projected onto the dominant row subspace of $X(t-1)$ spanned by the columns of $Q(t-1)$. The orthogonal projection of $z(t)$ onto the column space of $Q(t-1)$ can be interpreted as an estimate of the dominant modes of the signal in $z(t)$. Note that this projection is easily accomplished in any of the discussed LORAF algorithms as
TABLE IV
LAyERED ROW SUBSPACE ADAPTIVE FILTER

<table>
<thead>
<tr>
<th>Initialize:</th>
<th>(Q(t-1) = \begin{bmatrix} 1 \ 0 \end{bmatrix}; \quad \Theta(t-1) = 1; \quad d_\Sigma(t-1) = \begin{bmatrix} 0 \ 0 \end{bmatrix}; \quad 0 \leq \alpha \leq 1; \quad r )</th>
</tr>
</thead>
</table>

FOR EACH TIME STEP DO:

\[
\begin{bmatrix}
\text{Input: } x(t) = \begin{bmatrix} x(t), x(t-1), \ldots, x(t-N+1) \end{bmatrix}^T; \quad \text{Serialized Data Case}
\end{bmatrix}
\]

Subspace Tracking Section:

Use either LORAF 2 (Table 2) or LORAF 3 (Table 3) subspace tracker

Low Rank Adaptive Filter Section:

\[
\begin{aligned}
z_Q(t) &= z(t) - z_\perp(t) \quad & N \quad \text{mac} \quad (57) \\
d_\Sigma(t) &= D \begin{bmatrix} d_\Sigma(t-1) + N^{-1}z_Q(t) \end{bmatrix} \quad & 2N \quad \text{mac} \quad (59) \\
\hat{d}^+(t-N) &= \begin{bmatrix} 0 \ldots 0, 1 \end{bmatrix} d_\Sigma(t) \quad \text{In-space component} \quad (60)
\end{aligned}
\]

follows:

\[
z_Q(t) = Q(t-1)Q^T(t-1)z(t) = Q(t-1)h(t). \quad (56)
\]

Moreover, verify that \(z(t) = z_Q(t) + z_\perp(t)\). As \(z_\perp(t)\) is directly available in a subspace tracker of the LORAF 2 or LORAF 3 type, we immediately obtain the in-space component \(z_Q(t)\) via

\[
z_Q(t) = z(t) - z_\perp(t). \quad (57)
\]

Row subspace adaptive filters can be used in sensor array processing to estimate dominant time-varying spatial patterns from a series of spatial observations. A second important application area is serial data adaptive filtering where the elements of \(z(t)\) are consecutive samples of a time series. An estimate of the dominant signal in a time series can be obtained using “layered” row subspace adaptive filtering. This method is based on a sequential superposition of shifted vectors \(z_Q(t)\).

Define a downshift matrix \(D\)

\[
D = \begin{bmatrix} 0 & \cdots & 0 \\
0 & \ddots & \vdots \\
I & & 0
\end{bmatrix} \quad (58)
\]

and specify the estimated signal as the sum of shifted consecutive vectors \(z_Q(t)\). A recursive formula for this “layered superposition” of shifted projections is given by

\[
d_\Sigma(t) = Dd_\Sigma(t-1) + N^{-1}z_Q(t). \quad (59)
\]

Finally, extract the sample estimate of each time step as the bottom component of \(d_\Sigma(t)\):

\[
\hat{d}^+(t-N) = \begin{bmatrix} 0 \ldots 0, 1 \end{bmatrix}d_\Sigma(t). \quad (60)
\]

F. Low-Rank Adaptive Filtering in Correlated Gaussian Noise

Until now, we assumed that the noise in the data is white. In the case of nonwhite (correlated) Gaussian noise, the data must be whitened before one of the discussed low-rank adaptive filters is applied. Suppose that the noise is stationary Gaussian with known covariance \(\Phi_n\). Define a whitening matrix \(W = \Phi_n^{-1/2}\) as the inverse symmetric matrix square-root of \(\Phi_n\). Whiten the snapshot vector \(z_w(t) = Wz(t)\). Apply low-rank adaptive filtering to the sequence of vectors \(z_w(t)\). This produces a sequence of dominant whitened subspace basis matrices \(Q(t) = Wz(t)\). Finally, inverse whitening must be applied to generate the desired estimated signal vector. Thus, the overall operation is

\[
z_Q(t) = W^{-1}Q_w(t-1)Q_w^T(t-1)Wz(t). \quad (61)
\]

In the serialized data case, whitening can be performed with a conventional linear prediction whitening filter.

IV. RANK ADAPTIVE PROCESSING

In this section, we introduce the necessary relationships for an adaptive determination of the relevant eigenvectors in a low-rank adaptive filtering problem. We first introduce a method for adaptive discrimination of signal and noise eigenvectors. We further describe a technique that can be used to determine the eigenvectors closest to a reference signal \(y(t)\).

A. Adaptive Discrimination of Signal and Noise Eigenvectors

In our approach, the subspace trackers are operated with a fixed and predetermined rank \(r = r_{\text{max}}\). This fixed rank variable should be chosen sufficiently large so that in any case, all the interesting modes in the data can be accommodated in the reduced rank subspace (see the detailed discussion in the experimental section). We further need a methodology to select a subset of the dominant basis vectors that point in the directions in the reduced rank subspace in which the signal is much stronger than the noise. The number of selected “relevant” basis vectors depends on the number of active signals and can hence be a function of time. Therefore, we need an adaptive estimator that determines the number of active sources or the necessary and sufficient subspace dimension.
\[ r(t) \leq r_{\text{max}}. \]

We compare the estimated eigenvalues with an estimated noise floor level. Those eigenvalues that exceed the noise floor level by a certain factor \( \beta \) are counted, and the corresponding eigenvectors are used for signal reconstruction. This concept requires that we find a scheme to estimate the noise floor level or noise power from the given data power and the available estimated eigenvalues. Such a noise power estimator is described next. Assume that signal and noise are statistically independent \( \Phi = \Phi_s + \Phi_n \), where the time index has been omitted for convenience. Further assume that the noise is white \( \Phi_n = \sigma^2 I \). The goal is the estimation of the noise power \( \sigma^2 \). We exploit the fact that the signal is low-rank with a covariance matrix \( \Phi_s \) with structured EVD:

\[
\Phi_s = \begin{bmatrix} V_s^{(r)} & V_s^{(N-r)} \end{bmatrix} \begin{bmatrix} A_s^{(r)} & 0 \\ 0 & 0 \end{bmatrix} V_s^{(r)^T} = V_s^{(r)} A_s^{(r)} V_s^{(r)^T}. \tag{62}
\]

Recall the structure of the data covariance matrix \( \Phi \) as determined in (10), (11), (12a), and (12b). The elements of the EVD of \( \Phi \) can be expressed in terms of signal and noise components as follows:

\[
\Phi = V_r A_r V_r^T + V_{N-r} A_{N-r} V_{N-r}^T = V_s^{(r)} A_s^{(r)} V_s^{(r)^T} + \sigma^2 I = V_s^{(r)} (A^{(r)} + \sigma^2 I) V_s^{(r)^T} + V_s^{(N-r)} (\sigma^2 I) V_s^{(N-r)^T}. \tag{63}
\]

A comparison of terms in (63) reveals that each of the \( r \) dominant eigenvalues can be posed as the sum of a corresponding signal eigenvalue plus the noise power. Furthermore, the dominant data eigenvectors are equal to the signal eigenvectors:

\[
V_s^{(r)} = V_r \tag{64a}
\]

\[
A_r = A^{(r)} + \sigma^2 I. \tag{64b}
\]

Moreover, we obtain

\[
V_s^{(N-r)} = V_{N-r} \tag{65a}
\]

\[
A_{N-r} = \sigma^2 I. \tag{65b}
\]

Replace \( r \) by \( r_{\text{max}} \), and take the traces of both sides of (64b). This yields

\[
\text{tr} (A_{r_{\text{max}}}) = \text{tr} (A^{(r_{\text{max}})}) + r_{\text{max}} \sigma^2. \tag{66}
\]

Define the signal power \( p_s \) and the data power \( p_x \) as follows:

\[
p_s = \frac{1}{N} \text{tr} (\Phi_s) = \frac{1}{N} \text{tr} (A^{(r_{\text{max}})}) = \frac{1}{N} \text{tr} (A_{r_{\text{max}}}) - \frac{r_{\text{max}}}{N} \sigma^2, \tag{67}
\]

\[
p_x = \frac{1}{N} \text{tr} (\hat{\Phi}_s) = \frac{1}{N} \text{tr} (A_{r_{\text{max}}}) + \frac{r_{\text{max}}}{N} \sigma^2. \tag{68}
\]

Signal and noise are statistically independent. Hence, \( \sigma^2 \) can be posed as the data power minus the signal power. This yields the following relationship:

\[
\sigma^2 = p_x - p_s = p_x - \frac{1}{N} \text{tr} (A_{r_{\text{max}}}) + \frac{r_{\text{max}}}{N} \sigma^2. \tag{69}
\]

Solve (69) for \( \sigma^2 \) to obtain a noise power estimator that depends on the data power and the dominant data eigenvalues as desired:

\[
\sigma^2 = \frac{N}{N-r_{\text{max}}} p_x - \frac{1}{N-r_{\text{max}}} \text{tr} (A_{r_{\text{max}}}). \tag{70}
\]

The adaptive implementation of this estimator requires that we have available, at each time step, estimates of the dominant data eigenvalues in \( A_{r_{\text{max}}} \) and an estimate of the data power \( p_x \). We can use the diagonal elements in \( R(t) \) as estimates of the dominant eigenvalues. More accurate estimates of the dominant eigenvalues may be obtained from the following line of thought: Suppose that the subspace tracker has converged. Then, \( Q(t) \) represents the true dominant eigensubspace basis matrix \( V_{r_{\text{max}}} \). Thus, it follows that

\[
A_{r_{\text{max}}} (t) = Q^T(t) \Phi(t) Q(t). \tag{71}
\]

Now, replace \( \Phi(t) \) by its low-rank approximant (25). This suggests that we define

\[
\hat{A}_{r_{\text{max}}} (t) = Q^T(t) \hat{\Phi}(t) Q(t) = Q^T(t) Q(t) R(t) Q^T(t - 1) Q(t) = R(t) \Theta(t). \tag{72}
\]

The data power estimator must be defined compatible with the forgetting rule used in the subspace trackers. Hence, we must use an exponential forgetting based data power estimator

\[
\hat{p}_x (t) = \alpha \hat{p}_x (t - 1) + \frac{(1 - \alpha)}{N} \text{tr} (z(t) z^T(t)). \tag{73}
\]

### B. Selection of \text{y}-Relevant Eigenvectors

In spatial adaptive filtering, the power of dominant modes in \( z(t) \) may not be the only relevant criterion. It can happen that relatively weak modes are more important for a reconstruction of the information in \( y(t) \). Thus, it can be of some interest to classify the signal eigenvectors according to their relative contribution to the estimated information in \( y(t) \). The squared norm of \( d^s(t) \) is a measure of the overall contribution of all dominant modes to the estimated information in \( y(t) \). According to (21a), we can write

\[
||d^s(t)||^2 = \gamma^T(t) P_r(t) y(t) = e^T(t) V_r(t) A^{-1}_r(t) V^T_r(t) c(t). \tag{74}
\]

In an adaptive implementation, (74) can be computed as follows:

\[
||d^s(t)||^2 = \gamma^T(t) \tilde{A}_r^{-1}(t) y^s(t) = \sum_{k=1}^{\max} \theta_k(t). \tag{75}
\]

where

\[
\theta_k(t) = (\tilde{g}_k^s(t))^2 / \tilde{\lambda}_k(t). \tag{76}
\]

Here, \( \tilde{g}_k^s(t) \) denotes the \( k \)th component of \( g^s(t) \), and \( \tilde{\lambda}_k(t) \) is the \( k \)th estimated eigenvalue. The variable \( \theta_k(t) \) quantifies the strength of the \( k \)th eigenvector with respect to the reference \( y(t) \). Thus, it can be favourable to select eigenvectors corresponding to the largest \( \theta \)'s for reference signal approximation in spatial adaptive filtering.
V. EXPERIMENTAL VERIFICATION

In this section, we discuss the results of computer experiments that illustrate the behavior of the low-rank adaptive filters when applied for reconstruction of two types of nonstationary signals in white and correlated Gaussian noise.

A. Sums of Sinusoids in White and Colored Gaussian Noise

Fig. 1(a) shows the data components used in this experiment. Fig. 1(a) is a first sinusoidal source with normalized frequency $\omega_1 = 10^6$. This source is active in the time interval $300 \leq t \leq 3700$. Fig. 1(b) shows a second source with normalized frequency $\omega_2 = 12^6$. This source is active in the time interval $1300 \leq t \leq 2700$. Fig. 1(c) is the signal defined as the sum of the two sinusoids. Fig. 1(d) is a white Gaussian noise sequence. The raw data sequence Fig. 1(e) used in this experiment is the sum of the signal 1C and the noise 1D. In this raw data, each sinusoid lies $-4.88$ dB below the noise floor level. The layered row subspace adaptive filter of Table IV is now used for reconstruction of the signal given the noisy data without any side information. The order of the adaptive filter is set to a value of $N = 280$, and the subspace trackers are operated with a fixed $r_{\text{max}} = 8$. The forgetting factor is $\alpha = 0.994$, and the eigenvalue threshold is set to $\beta = 8.0$. Fig. 1(f) shows the reconstructed signal when the subspace tracker LORAF 2 was used. Fig. 1(g) is the corresponding reconstruction error. The experiment is repeated with the ultrafast subspace tracker LORAF 3. Fig. 1(h) is the reconstructed signal, and Fig. 1(i) is the reconstruction error. Interestingly, it turns out that the adaptive filter produces almost identical reconstruction results for both subspace trackers. Throughout all experiments, we observed that the ultrafast subspace tracker LORAF 3 reaches almost the same performance as the more sophisticated LORAF 2 tracker. Fig. 2 shows the trajectories of the eight dominant estimated eigenvalues and the corresponding adaptive threshold $\beta e^{2}(t)$ for the LORAF 3 tracker. The power of each of the two sources in this experiment is represented by a pair of eigenvalues. It is seen that these eigenvalue pairs exceed the adaptive threshold level rapidly when the sources are active. Noise eigenvalues stay clearly below the threshold level. Fig. 3 shows the corresponding estimated rank trajectory. We repeat the experiment with colored noise. Fig. 4 shows the data and the results. Fig. 4(a) is again the test signal as described in Fig. 1. Fig. 4(b) is a correlated Gaussian noise sequence generated by an order 6 feedback lattice filter (see [22, ch. 6]) driven by white Gaussian noise. The reflection coefficients of this filter are fixed to the following values: $K_1 = -0.915$, $K_2 = 0.41$, $K_3 = -0.42$, $K_4 = 0.24$, $K_5 = -0.21$, and $K_6 = 0.16$. This model produces highly correlated Gaussian noise that is spectrally similar to the noise encountered in some sonar environments. The input data is whitened using a linear prediction whitening filter. A corresponding inverse whitening filter is applied at the output of the low-rank adaptive filter. Fig. 4(c) shows the raw data sequence (signal 4A plus noise 4B). Fig. 4(d) is the reconstructed signal obtained with the LORAF 3 tracker. Fig. 4(e) is the corresponding reconstruction error.

B. Chirp Signals in White and Colored Gaussian Noise

The layered reconstruction based low-rank adaptive filter with subspace tracker LORAF 3 is applied to a problem of chirp signal enhancement. Fig. 5(a) is a sequence of three chirp transients. Fig. 5(b) is a white Gaussian noise process. Fig. 5(c) is the raw data (chirp transients plus noise). Fig. 5(d) is the reconstructed chirp sequence and Fig. 5(e) is the reconstruction error. The parameter configuration in this experiment was $N = 100$, $r_{\text{max}} = 8$, $\alpha = 0.97$, and $\beta = 8.0$. Fig. 6 shows the eigenvalue, adaptive threshold, and estimated rank trajectories. It is seen that the algorithm can safely detect and reconstruct the chirp transients even in heavy noise. This experiment is also repeated using the correlated noise described in part A of this section. The result is shown in Fig. 7. Fig. 7(d) is again the reconstructed chirp sequence, and Fig. 7(e) is the reconstruction error.

VI. CONCLUSIONS

We have proposed a class of fast rank adaptive filters with a complexity linear in $N$. Unlike classical RLS, where linear complexity algorithms are known only in the serialized data case, our algorithms are fast even in the more general spatial adaptive filter or array data case because complexity reduction is achieved by a clever exploitation of the rank properties of the data covariance matrix in each time step of the recursions. Thus, the proposed “low-rank adaptive filter” (LORAF) algorithms may not be useful only in temporal adaptive filtering (as shown in our simulations) but also in various sensor array and antenna signal processing applications including frequency and direction-of-arrival (DOA) estimation based on subspace tracking. Unlike classical RLS concepts, the new LORAF algorithms cannot be confused by rank-deficient data, and an effective signal/noise separation is naturally guaranteed by the underlying rank adaptive eigendecomposition structure of the algorithms.

A more general purpose of this paper was the introduction and discussion of fast and ultrafast subspace tracking algorithms based on the classical orthogonal iteration concept. The LORAF subspace trackers of this paper are, hence, related to the eigendecomposition of the data covariance matrix. Sometimes, however, one is interested in a direct adaptive factorization of the data matrix. For this purpose, we have developed the class of bi-iteration data adaptive SVD subspace trackers [31]. These algorithms are closely related to the adaptive orthogonal iteration LORAF subspace trackers of this paper; however, they employ Bauer’s bi-iteration [13], [18], [20], [29] as a conceptual basis. Instrumental variable subspace tracking algorithms based on the bi-iteration SVD subspace tracking concept have been developed [32]. These algorithms can be used for low-rank adaptive filtering and frequency estimation of signals in unknown correlated Gaussian noise without prewhitening of the data. A class of sliding window SVD algorithms for serialized data subspace tracking based on an adaptive square Hankel matrix eigendecomposition are developed in [33]. Finally, applications of the LORAF subspace trackers to adaptive frequency estimation are described in [34] and [35].
APPENDIX

This appendix provides an explanation why the term \( \alpha \Phi(t-1)\Delta(t-1) \) in recursion (38) can be neglected without any performance penalty. A comparison of (37) and (38) yields

\[
\Phi(t-1)[Q(t-1) - \Delta(t-1)] = A(t-1)\Theta(t-1). \quad (A1)
\]

Investigate the error that is introduced in this equation by setting \( \Delta(t-1) = 0 \). Recall that \( \Delta(t-1) = \bar{z}_\perp(t-1) f^T(t-1) \) according to (50), and \( f(t-1) = Q^T(t-1)\bar{z}_\perp(t-1) \) according to (48). Further, recall that \( \bar{z}_\perp(t-1) \) is orthogonal with respect to the column space of \( Q(t-2) \). When the exponential forgetting factor is close to 1 (which is usually always the case in practice), the angles between the column spaces of \( Q(t-1) \) and \( Q(t-2) \) will be very small. Thus, \( \bar{z}_\perp(t-1) \) will be almost
orthogonal with respect to the column space of $Q(t-1)$ as well. Therefore, $f(t-1)$ contains the cosines of angles that are almost $90^\circ$. These cosines will hence be much smaller than 1 in magnitude. Since $\mathbf{z}_{\perp}(t-1)$ is a unit-norm vector, it follows immediately that the norm of each column vector of $\Delta(t-1)$ must be much smaller than 1. Hence, the influence of $\Delta(t-1)$ on $Q(t-1)$ will be small. Thus, $A(t-1)\Theta(t-1)$ is indeed a very good approximation to $\Phi(t-1)Q(t-1)$. This explains why we could neglect the term $\alpha\Phi(t-1)\Delta(t-1)$ in (38) without any observable loss in performance.

Fig. 3. Estimated rank trajectory defined as the number of estimated eigenvalues in Fig. 2 that exceed the adaptive eigenvalue threshold.
Fig. 4. Reconstruction of a signal in correlated noise. (a) Signal. (b) Correlated Gaussian noise. (c) Raw data = signal A + noise B. (d) Reconstructed signal. (e) Reconstruction error.

Fig. 5. Reconstruction of chirp transients in white noise. (a) Chirp transients. (b) White Gaussian noise. (c) Raw data = chirp sequence A + noise B. (d) Reconstructed chirp sequence. (e) Reconstruction error.

REFERENCES

Fig. 6. Eigenvalue and rank trajectories for the experiment shown in Fig. 5. (a) Eigenvalue trajectories (dashed line indicates adaptive threshold). (b) Estimated rank trajectory.

Fig. 7. Reconstruction of chirp transients in correlated noise. (a) Chirp transients. (b) Correlated Gaussian noise. (c) Raw data = chirp sequence A + noise B. (d) Reconstructed chirp sequence. (e) Reconstruction error.


Peter Strobach (M'86–SM'91) was born in Passau, Germany, on February 6, 1955. He received the engineer's degree in electrical engineering from Fachhochschule Regensburg in 1978, the Diplom-Ingenieur (M.S.) degree from Technical University Munich, Germany, in 1983, and the Dr.-Ing. degree from Bundeswehr University, Munich, Germany, in 1985. From 1976 to 1977, he was with CERN Nuclear Research, Geneva, Switzerland. From 1978 to 1982, he was a systems engineer at Messerschmitt-Bölkow-Blohm GmbH, Munich, Germany. From May 1986 to December 1992, he was with Siemens AG, Zentralabteilung Forschung und Entwicklung (ZFE) in Munich. His last position at Siemens was head of the Signal and Image Processing Laboratory. In January 1993, he joined the Faculty of Fachhochschule Furtwangen (Black Forest), Germany. He has industrial experience in high-energy physics data processing, high-speed digital hardware, aircraft radar and guided missile systems, image processing, communication and coding systems, and noninvasive biomedical investigation systems. He is the sole author of over 50 reviewed papers in these areas and has authored the book Linear Prediction Theory: A Mathematical Basis for Adaptive Systems (Springer Series in Information Sciences, Heidelberg: Springer-Verlag, 1990). In the summer of 1990, he held the first adaptive filter course ever held in Germany at the University of Erlangen-Nuremberg, Germany. His current research interest is the theory of recursive algorithms. Dr. Strobach is a member of the IEEE Signal Processing Society and an Editorial Board member of Signal Processing. He is a member of the New York Academy of Sciences and is listed in the 1997 Edition of Who's Who in the World.