Stable Subspace Tracking Algorithm Based on a Signed URV Decomposition

Mu Zhou, *Student Member, IEEE*, and Alle-Jan van der Veen, *Fellow, IEEE*

***Abstract*—Subspace estimation and tracking are of fundamental importance in many signal processing algorithms. The class of "Schur subspace estimators" provides a complete parametrization of all "principal subspace estimates," defined as the column spans of corresponding low-rank matrix approximants that lie within a specified 2-norm distance of a given matrix. The parametrization is found in terms of a two-sided hyperbolic decomposition (Hyperbolic URV, or HURV), which can be computed using hyperbolic rotations. Unfortunately, such rotations are commonly associated with numerical instabilities.**

In this paper, we present a numerically stable, non-iterative algorithm to compute the HURV, called the Signed URV (SURV) algorithm. We show that this algorithm implicitly imposes certain constraints on the HURV such that important norm bounds that guarantee stability are satisfied. The constraints also restrict the parametrization of the subspace estimate such that it becomes close to the principal subspace provided by the SVD (which is a special case within this class).

The complexity of the algorithm is of the same order as that of a QR update. Updating and downdating are of the same complexity and are both numerically stable. SURV is proven to provide rank estimates consistent with the SVD with the same rank threshold. It can replace an SVD where only subspace estimation is needed. Typical applications would e.g. be the detection of the number of signals in array signal processing, and subspace estimation for source separation and interference mitigation, such as the first step in MUSIC and ESPRIT-type algorithms. Simulation results demonstrate the numerical stability and confirm that this algorithm provides exact rank estimates and good principal subspace estimates as compared to the SVD.

***Index Terms*—Generalized Schur algorithm, hyperbolic QR, hyperbolic URV, signed Cholesky factorization, subspace tracking.**

I. Introduction

FAST adaptive subspace estimation and tracking plays an important role in modern signal processing. It forms the key ingredient in many algorithms, such as adaptive filtering, system identification, blind channel estimation, and blind signal separation and equalization algorithms [1]–[3].

Typically, in these applications we are interested in a splitting of the space into the principal subspace and the minor subspace. These are spanned by the singular vectors corresponding to the singular values larger respectively smaller than a threshold. Thus, the singular value decomposition (SVD) is commonly used for computing the desired subspaces. Because directly computing and updating the SVD is expensive, computationally cheaper and faster subspace tracking methods have been proposed, such as the Rank Revealing QR [4], [5] and the URV [6], [7]. The literature related to the topic of subspace tracking is extremely rich, a brief overview follows later in this section.

An important observation is that, for splitting the space, we do not need the SVD. Let $\mathbf{X}$ be an $m \times n_2$ data matrix and γ a threshold, and call d the number of singular values of $\mathbf{X}$ larger than γ. Then $\mathbf{X}\mathbf{X}^H - \gamma^2\mathbf{I}$ has d eigenvalues larger than 0, and $m-d$ eigenvalues smaller than 0 ($\mathbf{I}$ denotes an identity matrix; we assume complex matrices $\mathbf{X}$, and the superscript ^{H} denotes the complex conjugate transpose.) Instead of the SVD, we can compute the signed Cholesky factorization

$$\mathbf{X}\mathbf{X}^H - \gamma^2\mathbf{I} = \mathbf{B}\mathbf{B}^H - \mathbf{A}\mathbf{A}^H \tag{1}$$

where $\mathbf{A}, \mathbf{B}$ have minimal dimensions. Although $\mathbf{A}$ and $\mathbf{B}$ are not unique, based on Sylvester's inertia theorem, we know that $\mathbf{A}$ has size $m \times (m-d)$ and $\mathbf{B}$ has size $m \times d$. Moreover, for any matrix $\mathbf{M}$ such that $\|\mathbf{M}\| \leq 1$, we can show [8] that the columns of

$$\mathbf{U} = \mathbf{B} - \mathbf{A}\mathbf{M} \tag{2}$$

span a rank-d "principal subspace", in the sense that there is a corresponding approximant $\hat{\mathbf{X}}$ with column span $\mathrm{ran}(\mathbf{U})$ (obtained e.g. by projecting onto this column span), such that

$$\|\mathbf{X} - \hat{\mathbf{X}}\| \leq \gamma \tag{3}$$

where $\|\cdot\|$ denotes the matrix 2-norm (largest singular value). The matrix $\mathbf{M}$ gives a complete parametrization of all such subspaces [8]; the principal subspace obtained by the SVD is within the class but is not explicitly identified. Note that $\mathbf{B}$ is already a valid principal subspace estimate; we will consider the use of $\mathbf{M} \neq \mathbf{0}$ later in the paper. The important point is that the required signed Cholesky factorization can be computed *implicitly* (by directly acting on $\mathbf{X}$ and not forming $\mathbf{X}\mathbf{X}^H$), and *non-iteratively*, at a complexity comparable to that of a QR factorization. By acting column-by-column, the implicit decomposition can also be *updated* efficiently.

Indeed, the required $\mathbf{A}$ and $\mathbf{B}$ follow from the factorization

$$[\mathbf{N} \quad \mathbf{X}]\mathbf{\Theta} = [\mathbf{A} \quad \mathbf{0} | \mathbf{B} \quad \mathbf{0}] \tag{4}$$

Manuscript received November 18, 2011; revised February 20, 2012; accepted February 29, 2012. Date of publication March 13, 2012; date of current version May 11, 2012. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Alfred Hanssen. This work was supported in part by the project BDREAMS at TU Delft, The Netherlands, and the China Scholarship Council from China. This paper was presented in part at the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Prague, Czech Republic, May 2011.

The authors are with the Circuits and Systems Group, Department of Electrical Engineering, Mathematics and Computer Sciences, Delft University of Technology, Delft, Netherlands (e-mail: m.zhou@tudelft.nl; a.j.vanderveen@tudelft.nl).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. This includes Matlab code for the algorithm presented in this paper. This material is 27.9 KB in size.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2012.2190732

where $\mathbf{N} = \gamma\mathbf{I}$, or more general any matrix such that $\mathbf{N}\mathbf{N}^H = \gamma^2\mathbf{I}$. The sign "|" in the right hand side matrix is a matrix separator which separates columns with a "positive signature" from those with a "negative signature". Also, $\mathbf{\Theta}$ is a $\mathbf{J}$-unitary matrix as defined in Section II: it satisfies $\mathbf{\Theta}\mathbf{J}\mathbf{\Theta}^H = \mathbf{J}$, where $\mathbf{J} = \mathbf{I} \oplus -\mathbf{I}$ is a signature matrix: a diagonal matrix with diagonal entries ± 1. In this paper, we will consider algorithms for computing and updating such a decomposition.

Straightforward generalizations are possible. Let $\mathbf{N}$ be any matrix, and define $\mathbf{R_N} = \mathbf{N}\mathbf{N}^H$, then the factorization (4) provides an implicit signed Cholesky factorization of $\mathbf{X}\mathbf{X}^H - \mathbf{R_N}$, leading to minimal-rank approximants $\hat{\mathbf{X}}$ such that

$$\left\| \mathbf{R_N}^{-1/2}(\mathbf{X} - \hat{\mathbf{X}}) \right\| \leq 1. \tag{5}$$

Such "whitened" approximants and corresponding subspace estimates are often needed, for example, in case of an array with uncalibrated antennas (resulting in different noise powers per antenna) [9], or in case of spatially correlated noise fields. In some cases, a nonwhite $\mathbf{R_N}$ can be due to interference. In signal processing applications, $\mathbf{X}$ is a noisy data matrix, measured column-by-column, and the columns of $\mathbf{N}$ could be generated by taking samples from a noise process similar to the disturbance on $\mathbf{X}$ (e.g., from a reference channel, or taken at a slightly different frequency where the desired signal is absent). Moreover, "downdating" (recomputing the decomposition after removing a column from $\mathbf{X}$) can be implementing by updating the decomposition using that column as an update for $\mathbf{N}$. In this way, sliding window subspace tracking algorithms are easily implemented.

In this paper, we propose a new algorithm for computing and updating the decomposition (4) as more columns of $\mathbf{X}$ and/or $\mathbf{N}$ become available. The algorithm is called the "Signed URV" (SURV). In fact, by introducing a QR factorization of $[\mathbf{A}, \mathbf{B}]$, the algorithm computes the two-sided decomposition

$$\mathbf{Q}^H[\mathbf{N} \quad \mathbf{X}]\mathbf{\Theta} = [\mathbf{R_A} \quad \mathbf{0}|\mathbf{R_B} \quad \mathbf{0}] \tag{6}$$

where $[\mathbf{R_A}, \mathbf{R_B}]$ is lower triangular and $\mathbf{Q}$ is a unitary matrix. This factorization may be called a Hyperbolic URV (HURV).[1] Generally, its computation is done using hyperbolic rotations, which unfortunately can cause numerical problems as the result of the rotation does not need to be bounded. An algorithm for (6) was previously shown in [11], but it used three hyperbolic rotations per update vector, and was not always numerically stable. In the derivation of the new SURV algorithm, care is taken that at most one hyperbolic rotation is used per update, and in such a way that the result is numerically stable.

Since $(\mathbf{A}, \mathbf{B})$ are not unique, we have a choice, and we will be interested in a specific class of "unbiased" decompositions (called SSE-2 in [8]), which are alternatively obtained by adding a certain linear constraint to $\mathbf{\Theta}$ in the HURV decomposition (6). With the constraint, the decomposition satisfies

$$\mathrm{ran}(\mathbf{B}) \subset \mathrm{ran}(\mathbf{X}), \quad \|\mathbf{B}\| \leq \|\mathbf{X}\| \tag{7}$$

$$\mathrm{ran}(\mathbf{A}) \subset \mathrm{ran}(\mathbf{N}), \quad \|\mathbf{A}\| \leq \|\mathbf{N}\|. \tag{8}$$

We will motivate that these norm bounds are the key to numerical stability, and will show that the SURV algorithm implicitly imposes constraints such that an estimate in the SSE-2 class is obtained, hence that the above norm bounds that guarantee stability are satisfied. It is in particular remarkable that, in this way, a stable downdating algorithm for subspace tracking is obtained. The constraints also restrict the parametrization of the subspace estimate such that it becomes close to the principal subspace provided by the SVD (which is a special case within this class).

The proposed algorithm is non-iterative, does not require condition estimations, and has a complexity of the same order as that of a QR factorization of the data matrix. In particular, given a threshold γ on the singular values of the data matrix, SURV tracks the exact rank of the subspace (number of singular values above the threshold), as well as an orthonormal basis of the principal subspace estimate, at a complexity of at most order 21 m^2 multiplications per update or downdate vector of size m.

Context: The literature on subspace tracking is rich and growing, and the right choice of a tracking algorithm is strongly dependent on the application. Here, we provide a brief classification to position the proposed SURV algorithm into context. A recent more elaborate overview can be found in [12], an older overview is in [3].

Algorithms can be classified based on various aspects. An important one is *complexity*. Algorithms of order $O(m^2)$ per update are often two-sided decompositions such as the rank revealing QR [4], [5], the URV [6], [7], or other techniques that track full subspace information in terms of an orthogonal $m \times m$ matrix containing the basis vectors, and an $m \times m$ coefficient matrix (usually triangular). SURV is in this class. With loss of information, it is sometimes possible to track only the $m \times d$ principal subspace estimate, along with a $d \times d$ coefficient matrix; details on the minor subspace are lost. This is interesting if the rank d is very small, and sufficient if the minor subspace is filled by white noise, or if there is a large gap in singular values between both subspaces. A prototype algorithm is by Karasalo [13], which has a complexity of $O(md^2)$ per update, as well as many techniques based on power iterations. Further reductions are possible if the subspace basis is not kept strictly orthogonal. For example, the PAST algorithm [14] is derived from a power iteration but uses an approximately orthogonal basis, which converges to an orthogonal basis under stationarity assumptions, and has complexity $O(md)$ per update. The speed of convergence in these cases usually depends on the gap in singular values between the two subspaces. Other algorithms in this class are PASTd [14], NIC [15], OPAST [16], etc.; see also [17] for an overview.

Another aspect is that of *rank tracking*. Many algorithms simply assume that the rank d of the data matrix is known. This is in particular the case for the $O(md^2)$ and $O(md)$ algorithms, since they only track a basis for the principal subspace. If the norm of a certain residual is too large, it is possible to detect that the rank has to increase; similarly, if the coefficient matrix becomes singular, the rank has to decrease. This requires special operations to resolve (as, e.g., missing information in the coefficient matrix has to be recovered), and iterative algorithms such as rank estimators to detect this. See, e.g., [12], [18], and [19]. Other algorithms are based on a threshold (tolerance) that sits in the gap between the large and small singular values; the rank is the number of singular values above the threshold. URV is an example, also the proposed SURV is in this class. This is more natural for many signal processing applications, where

[1]Not to be confused with the "high-rank" URV, or *hurv*, algorithm presented in [10].

there may be *a priori* information on the noise powers, but not on the number of signals.

Essential to any tracking algorithm is a form of data *windowing*. The majority of adaptive algorithms use an exponential window, where prior to each update, the old data represented in the coefficient matrix is scaled by a factor $\beta < 1$. This is in particular suitable for fixed-rank algorithms. For detecting sudden rank changes in non-stationary applications (e.g., the appearance of data frames in ad hoc communications), sliding window algorithms are more attractive: these perform an update with a new data vector, along with a "downdate" to remove the influence of an old data vector. Sliding window downdating algorithms have been extensively studied for two-sided $O(m^2)$ decompositions such as RRQR and URV, and have led to a series of papers as to how to handle the underlying indefinite Cholesky factorization and the related hyperbolic transformations [6], [20]–[23]. A numerically stable URV has proven to be "very complex" [10], and critically depends on accurate condition estimators; in the end a "relational stability" property can be proven [23]. For the $O(md)$ algorithms, many sliding window algorithms have only more recently been developed. Examples are SW-PAST [24], SWASVD [25], and FAPI [26].

The proposed SURV algorithm is based on the two-sided Hyperbolic URV decomposition proposed in [8] and [11], where an orthogonal basis of both principal and minor subspaces is stored and tracked, has complexity $O(m^2)$, uses a threshold on the singular values to detect rank, and is based on a sliding window. No condition estimation is needed, but nonetheless, the exact rank is obtained with respect to the set tolerance. Using computationally less favorable algorithms, the performance of the resulting subspace estimator for typical subspace tracking applications (DOA estimation) has been assessed and compared to the SVD, RRQR and URV in [27] and [28].

Outline: The paper is organized as follows. Section II introduces the elementary rotations used in this paper. Section III provides the theoretical basis for the proposed Hyperbolic URV decomposition. Section IV proposes the updating algorithm SURV and discusses the computational complexity. Section V proves that the proposed updating algorithm has favorable numerical properties. Section VI presents the simulation results. Section VII concludes this paper.

II. J-Unitary Matrices

In this section, we review some background materials on $\mathbf{J}$-unitary matrices from [8]. A square matrix $\boldsymbol{\Theta}$ is $\mathbf{J}$-unitary if it satisfies

$$\boldsymbol{\Theta}^H \mathbf{J} \boldsymbol{\Theta} = \mathbf{J}, \quad \boldsymbol{\Theta} \mathbf{J} \boldsymbol{\Theta}^H = \mathbf{J}, \tag{9}$$

where $\mathbf{J}$ is a signature matrix which follows some prescribed $(p+q) \times (p+q)$ block-partitioning of $\boldsymbol{\Theta}$:

$$\boldsymbol{\Theta} = \begin{matrix} \\ p \\ q \end{matrix} \begin{matrix} \overset{p}{} & \overset{q}{} \\ \end{matrix} \begin{bmatrix} \boldsymbol{\Theta}_{11} & \boldsymbol{\Theta}_{12} \\ \boldsymbol{\Theta}_{21} & \boldsymbol{\Theta}_{22} \end{bmatrix}, \qquad \mathbf{J} = \begin{bmatrix} +\mathbf{I}_p & \\ & -\mathbf{I}_q \end{bmatrix}. \tag{10}$$

Here, $\mathbf{I}_p$ denotes the $p \times p$ identity matrix, and p or q may be zero. If $\boldsymbol{\Theta}$ is applied to a block-partitioned matrix $[\mathbf{A}\ \mathbf{B}]$, then (9) implies

$$[\mathbf{A} \quad \mathbf{B}]\boldsymbol{\Theta} = [\mathbf{C} \quad \mathbf{D}] \Rightarrow \mathbf{A}\mathbf{A}^H - \mathbf{B}\mathbf{B}^H = \mathbf{C}\mathbf{C}^H - \mathbf{D}\mathbf{D}^H. \tag{11}$$

Hence, $\mathbf{J}$ assigns a positive signature to the columns of $\mathbf{A}, \mathbf{C}$, and a negative signature to those of $\mathbf{B}, \mathbf{D}$. We sometimes write the sign $+$ and $-$ above matrices to denote the positive and negative signatures of the columns in those matrices.

The $\mathbf{J}$-unitarity of $\boldsymbol{\Theta}$ implies $\boldsymbol{\Theta}_{22}^H \boldsymbol{\Theta}_{22} = \mathbf{I} + \boldsymbol{\Theta}_{12}^H \boldsymbol{\Theta}_{12}$ and $\boldsymbol{\Theta}_{22} \boldsymbol{\Theta}_{22}^H = \mathbf{I} + \boldsymbol{\Theta}_{21} \boldsymbol{\Theta}_{21}^H$. From these expressions, we can derive in turn the properties [8]

$$\boldsymbol{\Theta}_{22} \text{ is invertible}, \left\| \boldsymbol{\Theta}_{22}^{-1} \right\| \leq 1, \ \left\| \boldsymbol{\Theta}_{12} \boldsymbol{\Theta}_{22}^{-1} \right\| < 1, \left\| \boldsymbol{\Theta}_{22}^{-1} \boldsymbol{\Theta}_{21} \right\| < 1. \tag{12}$$

Similar properties hold for $\boldsymbol{\Theta}_{11}$:

$$\boldsymbol{\Theta}_{11} \text{ is invertible}, \left\| \boldsymbol{\Theta}_{11}^{-1} \right\| \leq 1, \left\| \boldsymbol{\Theta}_{11}^{-1} \boldsymbol{\Theta}_{12} \right\| < 1, \left\| \boldsymbol{\Theta}_{21} \boldsymbol{\Theta}_{11}^{-1} \right\| < 1. \tag{13}$$

For updating purposes, it will be necessary to work with column permutations of $[\mathbf{A}\ \mathbf{B}]$ and $[\mathbf{C}\ \mathbf{D}]$, which induces row and column permutations of $\boldsymbol{\Theta}$. Thus we introduce more general matrices $\boldsymbol{\Theta}$ that are $(\mathbf{J}_1, \mathbf{J}_2)$-unitary with respect to *unsorted* signature matrices $\mathbf{J}_1, \mathbf{J}_2$ satisfying $\boldsymbol{\Theta}^H \mathbf{J}_1 \boldsymbol{\Theta} = \mathbf{J}_2$, $\boldsymbol{\Theta} \mathbf{J}_2 \boldsymbol{\Theta}^H = \mathbf{J}_1$, where $\mathbf{J}_1$ and $\mathbf{J}_2$ are diagonal matrices with diagonal entries equal to ± 1. If $\mathbf{M}\boldsymbol{\Theta} = \mathbf{N}$, then $\mathbf{M}\mathbf{J}_1\mathbf{M}^H = \mathbf{N}\mathbf{J}_2\mathbf{N}^H$, so that $\mathbf{J}_1$ associates its signature to the columns of $\mathbf{M}$, and $\mathbf{J}_2$ associates its signature to the columns of $\mathbf{N}$. By inertia, the total number of positive entries in $\mathbf{J}_1$ has to be equal to that in $\mathbf{J}_2$, and likewise for the negative entries. It is always possible to return to sorted signature matrices by applying suitable column and row permutations to $\boldsymbol{\Theta}$.

Any $\boldsymbol{\Theta}$ can be constructed from a sequence of elementary 2×2 "rotations". It is similar to constructing an big unitary matrix from 2×2 Givens rotations. Let $\mathbf{j}_1 = \mathrm{diag}[\pm 1, \pm 1]$ be an (unsorted) 2×2 signature matrix, and similar for $\mathbf{j}_2$. A 2×2 matrix $\boldsymbol{\theta}$ is an elementary $(\mathbf{j}_1, \mathbf{j}_2)$-unitary rotation if it satisfies $\boldsymbol{\theta}^H \mathbf{j}_1 \boldsymbol{\theta} = \mathbf{j}_2, \boldsymbol{\theta} \mathbf{j}_2 \boldsymbol{\theta}^H = \mathbf{j}_1$. Similar to Givens rotations, it can be used to zero specific entries of vectors: for a given row vector $[r\ x]$ and signature $\mathbf{j}_1$, we can find $\boldsymbol{\theta}$ and r' such that $[r\ x]\boldsymbol{\theta} = [r'\ 0]$. The "input signature" $\mathbf{j}_1 = \mathrm{diag}[(j_1)_1\ (j_1)_2]$ associates with $[r\ x]$, and the "output signature" $\mathbf{j}_2 = \mathrm{diag}[(j_2)_1\ (j_2)_2]$ is such that $|r|^2 (j_1)_1 + |x|^2 (j_1)_2 = |r'|^2 (j_2)_1$. This determines $(j_2)_1$. The second entry $(j_2)_2$ follows by inertia considerations: the number of $+1$ and -1 entries in $\mathbf{j}_1$ and $\mathbf{j}_2$ are equal.

The precise form that $\boldsymbol{\theta}$ assumes depends on the input signature $\mathbf{j}_1$, and also on whether $|r| > |x|$ or $|r| < |x|$, as listed in Table I. Cases 1 and 2 in the table correspond to an indefinite signature $\mathbf{j}_1$ (i.e. $\mathrm{diag}[+1, -1]$ or $\mathrm{diag}[-1, +1]$), and the required $\boldsymbol{\theta}$ is an hyperbolic rotation, whereas Case 3 occurs when $\mathbf{j}$ is definite which leads to an ordinary circular (unitary or Givens) rotation.

Situations where $|r| = |x|$ with an indefinite signature $\mathbf{j}_1$ are degenerate: the result [0 0] is well defined but $\boldsymbol{\theta}$ is unbounded, and the output signature $\mathbf{j}_2$ is arbitrary.

III. Hyperbolic URV Decomposition

Let $\mathbf{N} : m \times n_1$ and $\mathbf{X} : m \times n_2$ be given matrices. We consider implicit factorizations of $\mathbf{X}\mathbf{X}^H - \mathbf{N}\mathbf{N}^H$ as

$$\mathbf{X}\mathbf{X}^H - \mathbf{N}\mathbf{N}^H = \mathbf{B}\mathbf{B}^H - \mathbf{A}\mathbf{A}^H \tag{14}$$

where $\mathbf{A}$ and $\mathbf{B}$ are nonsingular and together have m columns. $\mathbf{A}$ and $\mathbf{B}$ follow from the factorization

$$\begin{matrix} \\ m \end{matrix} \begin{matrix} \overset{n_1}{+} & \overset{n_2}{-} \\ \end{matrix} \begin{bmatrix} \mathbf{N} & \mathbf{X} \end{bmatrix} \boldsymbol{\Theta} = \begin{matrix} \\ m \end{matrix} \begin{matrix} \overset{n_1}{+} & \overset{n_2}{-} \\ \end{matrix} \begin{bmatrix} \mathbf{A}_e & \mathbf{B}_e \end{bmatrix} \tag{15}$$

where the sign $+$ and $-$ above the matrices denote the positive and negative signatures of the corresponding columns,

$$\mathbf{A}_e = m\ \overset{m-d}{\underset{}{[}}\mathbf{A}\ \overset{n_1-}{\underset{m+d}{\mathbf{0}]}}\ , \quad \mathbf{B}_e = m\ \overset{d}{\underset{}{[}}\mathbf{B}\ \overset{n_2-d}{\underset{}{\mathbf{0}]}}, \tag{16}$$

and $\mathbf{\Theta}$ is a $\mathbf{J}$-unitary matrix partitioned conformably as

$$\mathbf{\Theta} = \begin{matrix} \\ n_1 \\ n_2 \end{matrix} \overset{\begin{matrix} n_1 & n_2 \end{matrix}}{\begin{bmatrix} \mathbf{\Theta}_{11} & \mathbf{\Theta}_{12} \\ \mathbf{\Theta}_{21} & \mathbf{\Theta}_{22} \end{bmatrix}}, \quad \mathbf{J} = \begin{bmatrix} +\mathbf{I}_{n_1} & \\ & -\mathbf{I}_{n_2} \end{bmatrix}. \tag{17}$$

The factorization (15) always exists although $\mathbf{\Theta}$ is unbounded when $\mathbf{X}\mathbf{X}^H - \mathbf{N}\mathbf{N}^H$ is singular [8]; this corresponds to the case where a singular value of $\mathbf{R}_{\mathbf{N}}^{-1/2}\mathbf{X}$ is exactly equal to 1. However, the factorization is not unique, and even the subspaces $\mathrm{ran}(\mathbf{A})$ and $\mathrm{ran}(\mathbf{B})$ are not unique.

A straightforward way to find a factorization (15) is by the hyperbolic QR factorization (HQR) [8], [29], [30]

$$\overset{n_1}{\underset{+}{[}}\mathbf{N}\ \ \overset{n_2}{\underset{-}{\mathbf{X}]}}\,\mathbf{\Theta} = \overset{m}{\underset{\pm}{[}}\mathbf{R}\ \ \overset{n_1+n_2-m}{\underset{\pm}{\mathbf{0}]}} \tag{18}$$

where the sign "$\pm$" indicates an unsorted signature, $\mathbf{R}$ is a lower or upper triangular $m \times m$ matrix, and $\mathbf{\Theta}$ is $(\mathbf{J}_1, \mathbf{J}_2)$-unitary. Here, $\mathbf{J}_1 = \mathbf{J} = \mathbf{I}_{n_1} \oplus -\mathbf{I}_{n_2}$ is the sorted signature given from the outset, and $\mathbf{J}_2$ is an unsorted signature matrix which associates with the right hand side of (18). Although this factorization is straightforward to compute and update, it has the drawback that it does not always exist: the triangular form of $\mathbf{R}$ is too restrictive [8]. Although the set of exceptions is finite, in the neighborhood of an exception the resulting $\mathbf{R}$ may become numerically unbounded and nearly rank deficient, equivalently $\mathbf{A}$ and $\mathbf{B}$ may become unbounded with parts of their column spans nearly collinear; these cancel each other when forming $\mathbf{B}\mathbf{B}^H - \mathbf{A}\mathbf{A}^H$.

To get around this, introduce a QR factorization of $[\mathbf{A}\ \mathbf{B}]$:

$$\mathbf{R} = [\mathbf{R}_\mathbf{A} \quad \mathbf{R}_\mathbf{B}] = \mathbf{Q}^H[\mathbf{A} \quad \mathbf{B}] = \overset{m-d}{\underset{}{[}}\mathbf{Q}_\mathbf{A} \quad \overset{d}{\underset{}{\mathbf{Q}_\mathbf{B}]}}^H[\mathbf{A} \quad \mathbf{B}] \tag{19}$$

where $\mathbf{R}$ is triangular and $\mathbf{Q}$ is unitary. This leads to the more general two-sided decomposition [11]

$$\mathbf{Q}^H\,\overset{+}{[}\mathbf{N}\ \ \overset{-}{\mathbf{X}]}\,\mathbf{\Theta} = \overset{+}{[}\mathbf{R}_\mathbf{A}\ \ \overset{+}{\mathbf{0}}\ \ |\ \ \overset{-}{\mathbf{R}_\mathbf{B}}\ \ \overset{-}{\mathbf{0}]}\,. \tag{20}$$

Note that still $[\mathbf{A}\ \mathbf{0}|\mathbf{B}\ \mathbf{0}] = [\mathbf{N}\ \mathbf{X}]\mathbf{\Theta}$. This two-sided decomposition always exists [8], although some columns of $\mathbf{R}$ may be zero if $\mathbf{A}$ and $\mathbf{B}$ together are rank-deficient. The decomposition will be called a Hyperbolic URV (HURV).

In the next section, we will propose a stable algorithm to compute and update the HURV, called the Signed URV (SURV) algorithm. The decomposition is not unique, and we will subsequently place an additional constraint that leads to favorable properties (Section V). The derived SURV algorithm will then be shown to satisfy this constraint.

IV. SURV: An Updating Algorithm for the Hyperbolic URV

A. Overview

In this section, we propose an algorithm to update the HURV (20) as new columns for $\mathbf{X}$ and $\mathbf{N}$ become available. With proper initialization, this also serves as an algorithm to compute the HURV. We will use the 2×2 $\mathbf{j}$-unitary rotations specified in Table I as building blocks (cf. Givens rotations). Requirements on the updating algorithm are i) it should avoid the storage of $\mathbf{\Theta}$, as this matrix grows in dimensions and can potentially be unbounded in magnitude (even if the resulting $[\mathbf{A}, \mathbf{B}]$ are well defined), and ii) the update should be done in a numerically stable way.

TABLE I
Elementary $\mathbf{j}$-Unitary Zeroing Rotations[1]

Input: r, x, $\mathbf{j}_1 = \mathrm{diag}[\pm 1, \pm 1]$	
Output: $\boldsymbol{\theta}$ such that $[r\ \ x]\boldsymbol{\theta} = [r'\ \ 0]$, $\mathbf{j}_2 = \mathrm{diag}[\pm 1, \pm 1]$	
1.	If $\mathbf{j}_1 = \mathrm{diag}[+1, -1]$ or $\mathbf{j}_1 = \mathrm{diag}[-1, +1]$, and $\lvert r\rvert > \lvert x\rvert$: (Hyperbolic rotation) $\boldsymbol{\theta} = \begin{bmatrix} 1 & -s \\ -s^* & 1 \end{bmatrix} \frac{1}{c}$ where $s = \frac{x}{r}$, $c = \sqrt{1 - \lvert s\rvert^2}$; $\mathbf{j}_2 = \mathbf{j}_1$; $r' = cr$.
2.	If $\mathbf{j}_1 = \mathrm{diag}[+1, -1]$ or $\mathbf{j}_1 = \mathrm{diag}[-1, +1]$, and $\lvert r\rvert < \lvert x\rvert$: (Hyperbolic rotation) $\boldsymbol{\theta} = \begin{bmatrix} -s^* & 1 \\ 1 & -s \end{bmatrix} \frac{1}{c}$ where $s = \frac{r}{x}$, $c = \sqrt{1 - \lvert s\rvert^2}$; $\mathbf{j}_2 = -\mathbf{j}_1$ (sign reversal); $r' = cx$.
3.	If $\mathbf{j}_1 = \mathrm{diag}[+1, +1]$ or $\mathbf{j}_1 = \mathrm{diag}[-1, -1]$: (Givens rotation) $\boldsymbol{\theta} = \begin{bmatrix} c^* & -s \\ s^* & c \end{bmatrix}$ where $s = \frac{x}{\sqrt{\lvert r\rvert^2 + \lvert x\rvert^2}}$, $c = \sqrt{1 - \lvert s\rvert^2}$; $\mathbf{j}_2 = \mathbf{j}_1$; $r' = \sqrt{\lvert r\rvert^2 + \lvert x\rvert^2}$.

[1] * denotes the complex conjugate operator.

Several updating algorithms are possible, depending on one's objectives. Because of potential numerical instability associated with hyperbolic rotations, we design the updating steps to use at most only one hyperbolic rotation per vector update, which corresponds to a single decision whether the dimension of the principal subspace grows, remains constant, or shrinks. Moreover, that hyperbolic rotation should occur only when all the other entries in the two vectors on which the rotation is acting are already zero, as discussed later, so that an unbounded rotation has no detrimental numerical effects. (In [11], an updating algorithm was proposed that had at most three hyperbolic rotations per update, which is not minimal, and that algorithm was not always numerically stable.)

Because we would like to track the principal subspace $\mathrm{ran}(\mathbf{B})$, we keep $\mathbf{R} = [\mathbf{R}_\mathbf{A}\ \mathbf{R}_\mathbf{B}]$ in (20) lower triangular, so that the columns of $\mathbf{Q}_\mathbf{B}$ are an orthonormal basis of the principal subspace, and likewise the columns of $\mathbf{Q}_\mathbf{A}$ are an orthonormal basis of the minor subspace.

To start, suppose we have already computed the decomposition $\mathbf{Q}^H[\mathbf{N}\ \mathbf{X}]\mathbf{\Theta} = [\mathbf{R}_\mathbf{A}\ \mathbf{0}|\mathbf{R}_\mathbf{B}\ \mathbf{0}]$, where $\mathbf{R} = [\mathbf{R}_\mathbf{A}\ \mathbf{R}_\mathbf{B}]$ is square, lower triangular and sorted according to signature. In principle, updating the factorization with new columns of $\mathbf{X}$ or $\mathbf{N}$ is straightforward. Indeed, let us say that we want to find a new factorization $\mathbf{Q}'^H[\mathbf{N}'\ \mathbf{X}']\mathbf{\Theta}' = [\mathbf{R}'_\mathbf{A}\ \mathbf{0}|\mathbf{R}'_\mathbf{B}\ \mathbf{0}]$, where either $\mathbf{N}' = [\mathbf{N}\ \mathbf{n}]$, $\mathbf{X}' = \mathbf{X}$ if we want to add a new column $\mathbf{n}$ to $\mathbf{N}$ (also known as a "downdate" where we want to remove the effect of a previous $\mathbf{x}$), or $\mathbf{N}' = \mathbf{N}$, $\mathbf{X}' = [\mathbf{X}\ \mathbf{x}]$ if we augment $\mathbf{X}$ by $\mathbf{x}$ (the usual "update"). We will refer to all these cases as "updating". The only difference is in the signature of the update vector. Making use of the previously computed decomposition, it suffices to find $\mathbf{Q}_c$ and $\mathbf{\Theta}_c$ such that

$$\mathbf{Q}_c^H\,\overset{m-d}{\underset{+}{[}}\mathbf{R}_\mathbf{A}\ \ \overset{d}{\underset{-}{\mathbf{R}_\mathbf{B}}}\ \ \overset{1}{\underset{j_\mathbf{c}}{\mathbf{c}]}}\,\mathbf{\Theta}_c = \overset{m-d'}{\underset{+}{[}}\mathbf{R}'_\mathbf{A}\ \ \overset{d'}{\underset{-}{\mathbf{R}'_\mathbf{B}}}\ \ \overset{1}{\underset{j'_\mathbf{c}}{\mathbf{0}]}}$$
$$\mathbf{Q}' := \mathbf{Q}\mathbf{Q}_c \tag{21}$$

TABLE II
ZEROING SCHEMES FOR c_k

GCR:	**Givens Column Rotations** Apply only if $j_k = j_{\mathbf{c}}$: 1. Compute Givens rotation $\boldsymbol{\theta}$ such that $[r_{k,k}\ \ c_k]\boldsymbol{\theta} = [*\ 0]$; 2. Apply $\boldsymbol{\theta}$ to the k-th column of $\mathbf{R}$ and $\mathbf{c}$ (no sign change).
HCR:	**Hyperbolic Column Rotations** Apply if $j_k = -j_{\mathbf{c}}$: 1. Set $\mathbf{j}_1 = \mathrm{diag}[j_k, j_{\mathbf{c}}]$, and compute $\boldsymbol{\theta}$ and $\mathbf{j}_2$ such that $[r_{k,k}\ \ c_k]\boldsymbol{\theta} = [*\ 0]$; 2. Apply $\boldsymbol{\theta}$ to the k-th column of $\mathbf{R}$ and $\mathbf{c}$; update signatures $j_k, j_{\mathbf{c}}$ following $\mathbf{j}_2$ (possible sign change).
GRCR:	**Givens Row and Column Rotations** Apply only if $j_k = j_{k+1}$: 1. Compute Givens row rotation $\mathbf{q}$ such that $\mathbf{q}^H \left[\begin{smallmatrix} c_k \\ c_{k+1} \end{smallmatrix}\right] = \left[\begin{smallmatrix} 0 \\ * \end{smallmatrix}\right]$; 2. Apply $\mathbf{q}^H$ to rows $(k, k+1)$ of $[\mathbf{R},\ \mathbf{c}]$; apply $\mathbf{q}$ to columns $(k, k+1)$ of $\mathbf{Q}$; 3. Compute Givens column rotation $\boldsymbol{\theta}$ such that $[r_{k,k}\ \ r_{k,k+1}]\boldsymbol{\theta} = [*\ 0]$; 4. Apply $\boldsymbol{\theta}$ to columns $(k, k+1)$ of $\mathbf{R}$ (no sign change).
GRR:	**Givens Row Rotations to zero** $r_{k,k+1}$ Apply only if $c_k = c_{k+1} = 0$: 1. Compute Givens row rotation $\mathbf{q}$ such that $\mathbf{q}^H \left[\begin{smallmatrix} r_{k,k+1} \\ r_{k+1,k+1} \end{smallmatrix}\right] = \left[\begin{smallmatrix} 0 \\ * \end{smallmatrix}\right]$; 2. Apply $\mathbf{q}^H$ to rows $(k, k+1)$ of $[\mathbf{R},\ \mathbf{c}]$; apply $\mathbf{q}$ to columns $(k, k+1)$ of $\mathbf{Q}$.

where $\mathbf{c} = \mathbf{Q}^H\mathbf{n}$ if we add a column $\mathbf{n}$ to $\mathbf{N}$ or $\mathbf{c} = \mathbf{Q}^H\mathbf{x}$ if we add a column $\mathbf{x}$ to $\mathbf{X}$. (Note that we need to store and update $\mathbf{Q}$ to apply this transformation.) In the first case, $\mathbf{c}$ has a positive signature $j_{\mathbf{c}} = +1$; in the second case, $j_{\mathbf{c}} = -1$. Denote the signature of $\mathbf{R}$ by $\mathbf{J} = \mathbf{I}_{m-d} \oplus -\mathbf{I}_d = \mathrm{diag}[j_1, j_2, \ldots, j_m]$. Let c_k denote the kth entry of $\mathbf{c}$ and $r_{k,l}$ denote the (k,l)th entry of $\mathbf{R}$. The rank of the principal subspace before the update is d, and after the update it is d'.

The updating algorithm consists of a sequence of elementary rotations for reducing $[\mathbf{R}\ \mathbf{c}]$ to $[\mathbf{R}'\ \mathbf{0}]$, i.e., to zero $\mathbf{c}$. In the algorithm, the $m \times m$ matrices $\mathbf{Q}$ and $\mathbf{R}$ are stored and tracked, but $\boldsymbol{\Theta}$ is not stored.

B. Algorithm

To compute the factorization (21), the entries $c_1, c_2, \cdots, c_m$ of $\mathbf{c}$ are zeroed in turn. As listed in Table II, there are three possibilities to do this: by elementary column rotations $\boldsymbol{\theta}$ [Givens column rotations (GCR) and hyperbolic column rotations (HCR)], or by elementary row rotations $\mathbf{q}$ in combination with Givens rotations $\boldsymbol{\theta}$ (GRCR). The "GCR" and "HCR" schemes to zero entry c_k are the most natural and efficient, as they directly zero c_k against $r_{k,k}$, using a rotation $\boldsymbol{\theta}$ such that $[r_{k,k}\ c_k]\boldsymbol{\theta} = [r'_{k,k}\ 0]$. However, in the case that the signatures of $r_{k,k}$ and c_k differ, the required rotation is hyperbolic, leading to potential numerical instability if $|c_k| \approx |r_{k,k}|$. To avoid this, the "GRCR" scheme first computes an elementary circular (unitary) rotation $\mathbf{q}$ to zero c_k against c_{k+1}, and then a $\boldsymbol{\theta}$-rotation to zero the resulting fill-in in $r_{k,k+1}$ against $r_{k,k}$. If the signatures of $r_{k,k+1}$ and $r_{k,k}$ are the same, then this rotation is a Givens rotation.

For reasons of numerical stability, it is desirable to minimize the number of *hyperbolic* rotations, i.e. rotations $\boldsymbol{\theta}$ that act on columns with unequal signatures. The "Signed URV" algorithm proposed next is designed such that at most one hyperbolic rotation (HCR) is needed, which corresponds to a single decision whether the rank of the subspace estimate should be increased or decreased. As we will show later, this makes the algorithm numerically stable.

The algorithm is described as follows. At the start of the algorithm, the data is represented as in Fig. 1(a) for the case $j_{\mathbf{c}} = +1$, and in Fig. 3(a) for the case $j_{\mathbf{c}} = -1$. These cases are treated separately.

- If $j_{\mathbf{c}} = +1$, then

  (*a*) For $k = 1, \cdots, m-d$, we have $j_{\mathbf{c}} = j_k = +1$. Thus, we cancel c_k against $r_{k,k}$ using GCR elementary rotations. The result is shown in Fig. 1(b). (Note that for $k > m-d$, the signature are unequal and GCR rotations are not applicable.)

  If $d = 0$, then we are done, else we continue:

  (*b*) For $k = m-d+1, \cdots, m-1$, we have $j_{\mathbf{c}} = +1$, $j_k = -1$. Thus, we cancel c_k using GRCR. The result is in Fig. 1(c).

  (*c*) At this point, only c_m is nonzero, and has to be cancelled against $r_{m,m}$. Since $j_{\mathbf{c}} = +1$ and $j_m = -1$, the signatures differ and a HCR is applied to cancel c_m.
  If during the HCR the signatures stay constant ($|r_{m,m}| > |c_m|$, see Fig. 1(d1), then we are done without rank changes ($d' = d$). See later for the exceptional case where $|r_{m,m}| = |c_m|$. On the other hand, if $|r_{m,m}| < |c_m|$, see Fig. 1(d2), the signatures are reversed and we continue: (see Fig. 2(d2)).

  (*d2*) Now $j_m = +1$ and $d' = d-1$: a rank decrease. As seen in Fig. 2(d2), the signatures are not sorted, as the last column of $\mathbf{R}$ has a $+1$ signature and belongs to $\mathbf{R}_{\mathbf{A}}$. Thus move the last column $\mathbf{r}_m$ of $\mathbf{R}$ to fit at the end of $\mathbf{R}_{\mathbf{A}}$ and update the signatures in a corresponding way: $\mathbf{J}' = \mathbf{I}_{m-d'} \oplus -\mathbf{I}_{d'}$. See Fig. 2(e).

  (*e*) Restore the lower triangular structure of $\mathbf{R}$ using GRR (see Table II), for $k = m-1$ down to $m-d+1$ (or $m-d'$). See Fig. 2(f). This concludes the update.

- If $j_{\mathbf{c}} = -1$ and $d < m$ (see Fig.3(a)), then

  (*a*) For $k = 1, \cdots, m-d-1$, we have $j_k = +1$, whereas $j_{\mathbf{c}} = -1$. Thus, to avoid hyperbolic rotations, we cancel c_k against c_{k+1} using GRCR. (Entry c_{m-d} cannot yet be cancelled as it would lead to a hyperbolic rotation.) See Fig. 3(b).

  (*b*) Swap column $\mathbf{r}_{m-d}$ of $\mathbf{R}$ with $\mathbf{c}$, also swap the corresponding signatures, and set $d := d+1$. This potential rank increase can later become undone. See Fig. 1(c).
  Now continue as in the case $j_{\mathbf{c}} = +1$, step (*b*). In step (*c*) we apply an HCR, the rank may decrease again in which case $d' = d$ (and we continue with step (*d2*)), else we remain with $d' = d+1$ and we are done.

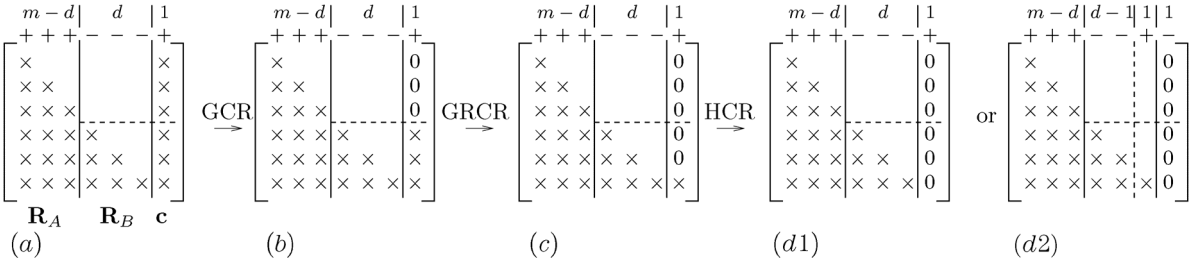

Fig. 1. Updating sequence for $j_{\mathbf{c}} = +1$.

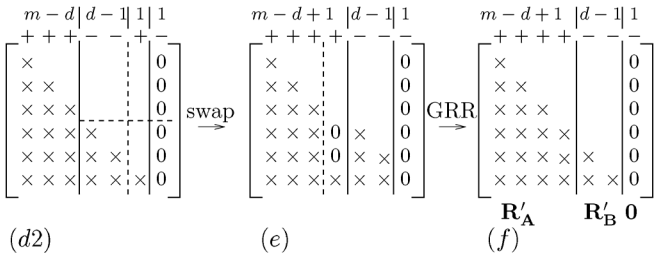

Fig. 2. Updating sequence for $j_{\mathbf{c}} = +1$ (continued): signature sorting steps.

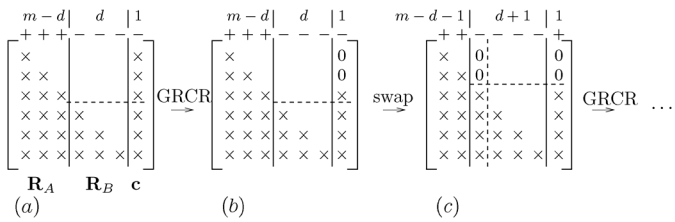

Fig. 3. Updating sequence for $j_{\mathbf{c}} = -1$. Except for dimensions, (c) corresponds with Fig. 1(b), and the algorithm continues at that place.

- If $j_{\mathbf{c}} = -1$ and $d = m$, then
  - (*a*) For $k = 1, \cdots, m$, we have $j_k = -1$ and also $j_{\mathbf{c}} = -1$. Cancel c_k against $r_{k,k}$ using GCR. There are no rank changes.

The only hyperbolic rotation in the algorithm is in step (*c*). Depending on the sign change in this rotation (which occurs if $|r_{m,m}| < |c_m|$) and on the value of $j_{\mathbf{c}}$, the rank d of the subspace increases, decreases or remains constant. This directly corresponds to the number of singular values that are larger or smaller than the selected threshold. If at this stage $|r_{m,m}| = |c_m|$, i.e. a singular value is equal to the threshold, then the rank decision is arbitrary. In the algorithm, $\boldsymbol{\theta}$ will be unbounded but the result $[r_{m,m}\ c_m]\boldsymbol{\theta} = [0\ 0]$ is well defined, and as $\boldsymbol{\theta}$ is not stored or used anyway, the result is numerically stable. This works because we took care that the hyperbolic rotation $\boldsymbol{\theta}$ acts on the last column of $\mathbf{R}$ and $\mathbf{c}$, which are zero except for their last entries $r_{m,m}$ and c_m; after the rotation both columns are $\mathbf{0}$. This is unlike the previously proposed related algorithm [11], which used up to three hyperbolic rotations per update, acting on nonzero columns, where each of these rotations could lead to unbounded results. (The numerical stability is further studied in Section V.)

C. Initialization

Prior to any updates, the algorithm is initialized as follows:

$$d = 0; \quad \mathbf{R} = \mathbf{0}_{m\times m}; \quad \mathbf{Q} = \mathbf{I}_m.$$

TABLE III
COMPUTATIONAL COMPLEXITY (COMPLEX MULTIPLICATIONS; ONLY DOMINANT TERMS SHOWN)

	$j_{\mathbf{c}} = +1$ case		$j_{\mathbf{c}} = -1$ case
$\mathbf{Q}^H\mathbf{n}$:	m^2	$\mathbf{Q}^H\mathbf{x}$:	m^2
(a):	$4m(m-d)$	(a):	$12m(m-d)$
(b):	$8md + 4d^2$	(b):	$8md + 4d^2$
(e):	$8md$	(e):	$8md$
Total:	$5m^2 + 12md + 4d^2$ multiplications	Total:	$13m^2 + 4md + 4d^2$ multiplications

If we choose $\mathbf{N} = \gamma\mathbf{I}$, i.e., compute the principal subspace of $\mathbf{X}$ in the sense of (3), then the first m updates after this initialization are using the columns of $\mathbf{N}$, and the result is

$$d = 0; \quad \mathbf{R} = \gamma\mathbf{I}_m; \quad \mathbf{Q} = \mathbf{I}_m.$$

Hence, this is also a valid initialization for the case $\mathbf{N} = \gamma\mathbf{I}$. More in general, we can also initialize using $\mathbf{R} = \mathbf{R}_{\mathbf{N}}^{\frac{1}{2}}$, where $\mathbf{R}_{\mathbf{N}}^{\frac{1}{2}}$ is a lower triangular Cholesky factor of $\mathbf{R}_{\mathbf{N}} = \mathbf{N}\mathbf{N}^H$, which is in most applications a suitably scaled noise covariance matrix.

D. Computational Complexity

The computational complexity per update vector is assessed as a fixed number of m^2 multiplications (for the initial transformation of $\mathbf{n}$ or $\mathbf{x}$ to $\mathbf{c}$ by $\mathbf{Q}$), followed by $O(m^2)$ elementary rotations. More precise numbers are given in Table III, where applying a 2×2 rotation to a 2×1 vector has been counted as four complex multiplications. The complexity depends on the sign $j_{\mathbf{c}}$ (update or downdate), the rank d, and potential rank changes. The worst case is attained for rank $d = m$, at a complexity of $O(21m^2)$ multiplications per update vector.

In the proposed algorithm, only the square matrix $\mathbf{Q} : m \times m$ and the lower triangular matrix $\mathbf{R} : m \times m$ need to be stored.

V. UPDATING THE SSE-2

A. SSE-2 Definition and Properties

As mentioned at the end of Section III, the proposed HURV decomposition (20) is not unique, and we can place additional constraints to reach desired properties. This is already the case for the original one-sided decomposition (15). At the same time, all valid subspace estimates have the form

$$\mathbf{U} = \mathbf{B} - \mathbf{A}\mathbf{M}, \qquad \|\mathbf{M}\| \leq 1$$

where $\mathbf{M} : (m-d) \times d$ is a contractive matrix that parametrizes all solutions. These freedoms are connected: Given a specific $\mathbf{M}$, it is always possible to transform $(\mathbf{A}, \mathbf{B})$ using additional

rotations $\mathbf{\Theta_M}$ to a new $(\mathbf{A}', \mathbf{B}')$ such that $\mathrm{ran}(\mathbf{B}') = \mathrm{ran}(\mathbf{B} - \mathbf{AM})$, i.e., the same subspace is obtained using $\mathbf{B}'$ and a new parameter $\mathbf{M}' = \mathbf{0}$. The existence of this transformation is shown in the proof of Theorem 1, below.

For a given decomposition, define a block partitioning of the corresponding $\mathbf{\Theta}$ as

$$\mathbf{\Theta} = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \overset{\begin{array}{cc} n_1 & n_2 \end{array}}{\begin{bmatrix} \mathbf{\Theta}_{11} & \mathbf{\Theta}_{12} \\ \mathbf{\Theta}_{21} & \mathbf{\Theta}_{22} \end{bmatrix}}. \tag{22}$$

In [8], a specific choice for $\mathbf{M}$ was taken, namely a function of $\mathbf{\Theta}$:

$$\mathbf{M_\Theta} := [\mathbf{I}_{m-d} \quad \mathbf{0}]\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\begin{bmatrix}\mathbf{I}_d \\ \mathbf{0}\end{bmatrix}. \tag{23}$$

The corresponding subspace estimate $\mathbf{U}_{\mathrm{SSE}-2} = \mathbf{B} - \mathbf{AM_\Theta}$ was called the "SSE-2" subspace estimate, and was shown in [8] to enjoy important properties that make it numerically stable and "unbiased" in the sense that $\mathrm{ran}(\mathbf{U}_{\mathrm{SSE}-2}) \subset \mathrm{ran}(\mathbf{X})$. Specifically:

Lemma 1: Given an HURV decomposition (20), consider $\mathbf{B}' = \mathbf{B} - \mathbf{AM_\Theta}$, with $\mathbf{M_\Theta}$ given by (23). Then, $\mathrm{ran}(\mathbf{B}') \subset \mathrm{ran}(\mathbf{X})$, $\|\mathbf{B}'\| \leq \|\mathbf{X}\|$.

Proof: The proof is a straightforward generalization of [8], Lemma 3.4, which derived this for $\mathbf{N} = \epsilon\mathbf{I}$. ∎

It was also shown in simulations [8] that the resulting subspace estimates $\mathrm{ran}(\mathbf{B}')$ are quite close to the principal subspace estimate that follows from the SVD, which is in fact a special case of an SSE-2 subspace (see Appendix A). Unfortunately, the direct computation of $\mathbf{B}' = \mathbf{B} - \mathbf{AM_\Theta}$ with $\mathbf{\Theta}$ given by (23) is quite cumbersome, as it requires the inversion of a submatrix of $\mathbf{\Theta}$, which itself grows as we add more updates. Also, it is possible that $\mathbf{\Theta}$ is unbounded (or very large) even if the related $(\mathbf{A}, \mathbf{B})$ are well defined.

B. The SURV Algorithm Provides an SSE-2 Decomposition

In this section, we show that the proposed SURV updating algorithm of Section IV automatically leads to a $\mathbf{B}$ that is already the intended SSE-2 subspace estimate. This is achieved by restricting the available freedom in $\mathbf{\Theta}$ such that $\mathbf{M_\Theta}$ in (23) satisfies $\mathbf{M_\Theta} = \mathbf{0}$, i.e., the inversion present in (23) is avoided, storage of $\mathbf{\Theta}$ is not needed, and $\mathbf{U}_{\mathrm{SSE}-2} = \mathbf{B}$. The results are summarized as follows[2]:

Theorem 1 [11]: For given matrices $\mathbf{N} : m \times n_1$ and $\mathbf{X} : m \times n_2$, there exist matrices $\mathbf{Q} : m \times m$, $\mathbf{\Theta} : (n_1 + n_2) \times (n_1 + n_2)$, $\mathbf{T} : n_1 \times n_1$ such that

$$\mathbf{Q}^H \overset{\begin{array}{cc} n_1 & n_2 \\ + & - \end{array}}{[\mathbf{N} \quad \mathbf{X}]}\mathbf{\Theta} = \overset{\begin{array}{cc|cc} m-d & n_1- & d & n_2-d \\ + & + & - & - \end{array}}{[\mathbf{R_A} \quad \mathbf{0} \quad | \quad \mathbf{R_B} \quad \mathbf{0}]}, \tag{24}$$

$$\mathbf{T}\overset{\begin{array}{cc} n_1 & n_2 \\ + & - \end{array}}{[\mathbf{I} \quad \mathbf{0}]}\mathbf{\Theta} = \begin{array}{c} \\ m-d \\ n_1-m+d \end{array} \overset{\begin{array}{cc|cc} m-d & n_1- & d & n_2-d \\ + & + & - & - \end{array}}{\left[\begin{array}{cc|cc} \mathbf{I} & \mathbf{0} & \mathbf{0} & * \\ \mathbf{0} & \mathbf{I} & * & * \end{array}\right]} \tag{25}$$

where $\mathbf{Q} = [\mathbf{Q_A}\ \mathbf{Q_B}]$ is unitary, $\mathbf{\Theta}$ is $\mathbf{J}$-unitary, $[\mathbf{R_A}\ \mathbf{R_B}]$ is lower triangular, and $\mathbf{T}$ is an invertible matrix.

Let $[\mathbf{A}\ \mathbf{B}] = \mathbf{Q}[\mathbf{R_A}\ \mathbf{R_B}]$. Then, $\mathrm{ran}(\mathbf{Q_B}) = \mathrm{ran}(\mathbf{B})$ is an SSE-2 subspace estimate.

Proof: See Appendix B. ∎

[2]The theorem was proposed in [11] without a complete proof.

Corrolary 1: For the decomposition in Theorem 1, $\mathbf{\Theta}$ is bounded if $\mathbf{NN}^H - \mathbf{XX}^H$ is nonsingular. In any case we have

1) $\mathrm{ran}(\mathbf{B}) \subset \mathrm{ran}(\mathbf{X})$, $\|\mathbf{R_B}\| = \|\mathbf{B}\| \leq \|\mathbf{X}\|$;
2) $\mathrm{ran}(\mathbf{A}) \subset \mathrm{ran}(\mathbf{N})$, $\|\mathbf{R_A}\| = \|\mathbf{A}\| \leq \|\mathbf{N}\|$.

Proof: The result for $\mathbf{\Theta}$ follows from [8], Theorem 3.1, which proved it for the more general HURV. Theorem 1 shows that for the present decomposition (constrained by (25)), $\mathbf{M_\Theta} = \mathbf{0}$. Thus, by Lemma 1, the range and norm properties for $\mathbf{B}$ follow. Since $\mathbf{NN}^H - \mathbf{AA}^H = \mathbf{XX}^H - \mathbf{BB}^H \geq 0$, it follows $\|\mathbf{A}\| \leq \|\mathbf{N}\|$, hence also $\mathrm{ran}(\mathbf{A}) \subset \mathrm{ran}(\mathbf{N})$ (see [31, p. 471]). Finally, $\|\mathbf{R_A}\| = \|\mathbf{Q}^H\mathbf{A}\| = \|\mathbf{A}\|$ and $\|\mathbf{R_B}\| = \|\mathbf{Q}^H\mathbf{B}\| = \|\mathbf{B}\|$, so the same norm inequalties hold for $\mathbf{R_A}$ and $\mathbf{R_B}$. ∎

Thus, we see that the results $(\mathbf{A}, \mathbf{B})$ of the decomposition are bounded by the inputs, even if $\mathbf{\Theta}$ may be unbounded. Also the corresponding subspaces $\mathbf{Q_A}, \mathbf{Q_B}$ are well-defined.

It remains to show:

Theorem 2: The SURV algorithm presented in Section IV provides the decomposition in Theorem 1 (without explicitly computing or storing $\mathbf{\Theta}, \mathbf{T}$ and the right-hand side of (25)).

Proof: See Appendix C. ∎

C. Numerical Stability

Generally, algorithms using hyperbolic rotations have an issue of numerical stability: such rotations may have a large norm. Some related literature is found in [20]–[23] and [32].

Regarding the numerical stability of the proposed SURV algorithm, we note first of all that Corrolary 1 states that the corresponding "SSE-2" decomposition in Theorem 1 is well behaved: the resulting outputs $\mathbf{A}, \mathbf{B}$ are bounded by the inputs, and in the algorithm the potentially unbounded $\mathbf{\Theta}$ is not explicitly computed or used. Furthermore, in the implementation, the algorithm has only one hyperbolic rotation per update vector $\mathbf{c}$. This hyperbolic rotation $\boldsymbol{\theta}$ is computed from two entries $r := r_{m,m}$ and $x := c_m$, and is applied only to these two entries as the corresponding columns of $\mathbf{R}$ and $\mathbf{c}$ to which $\boldsymbol{\theta}$ should be applied are already zero except for these two entries. Given the numerically stable forms for hyperbolic rotations in Case 1 and Case 2 of Table I:

Case 1) $(|r| > |x|)$: $s = \frac{x}{r}$; $[r\ x]\boldsymbol{\theta} = [r\sqrt{1-|s|^2}\ 0]$,
Case 2) $(|r| < |x|)$: $s = \frac{r}{x}$; $[r\ x]\boldsymbol{\theta} = [0\ x\sqrt{1-|s|^2}]$,

it is possible to compute the resulting output vector (or entry r' in $[r\ x]\boldsymbol{\theta} = [r'\ 0]$) directly, avoiding the potential singularity problem arising from the intermediate computation of $\boldsymbol{\theta}$ in case $|r| \approx |x|$. Indeed, $|r'| < |r|$, resp. $|r'| < |x|$, which is also seen as a particular case of Corrolary 1. All other operations in the SURV algorithm are ordinary Givens rotations (Case 3 in Table I), which are known to be numerically stable [33]. Thus, we claim that the proposed SURV algorithm is also stable.

If it happens that $|r| = |x|$ (this can occur only if $\mathbf{NN}^H - \mathbf{XX}^H$ is singular, e.g. for $\mathbf{N} = \gamma\mathbf{I}$ if a singular value of $\mathbf{X}$ is precisely equal to the threshold γ), then the numerically stable forms given above result in $r' = 0$. A decision will have to be made on the corresponding output signature ($\mathbf{j}_2 = \mathbf{j}_1$ or $\mathbf{j}_2 = -\mathbf{j}_1$); this decision is arbitrary and corresponds to a rank decision on $\mathbf{X}$ when a singular value is precisely at the threshold. The numerical properties of the algorithm do not depend on this decision.

VI. Simulation Results

In this section, we demonstrate the numerical stability of the proposed SURV algorithm, and compare its performance to the original HURV algorithm [11] and to the URV [10]. For the simulations, we use a generic subspace tracking data model as

$$\mathbf{x}(t) = \mathbf{H}(t)\mathbf{s}(t) + \mathbf{n}(t), \quad t = 1, 2, \cdots, N \tag{26}$$

where $\mathbf{x}(t) : m \times 1$, $\mathbf{s}(t) : d \times 1$ $(d \leq m)$ is formed by i.i.d. Gaussian random variables with zero mean and standard deviation $\sigma_s = 1$, and $\mathbf{n}(t) : m \times 1$ is formed by i.i.d. Gaussian noise with zero mean and standard deviation σ_n. The mixing matrix $\mathbf{H}(t) : m \times d$ has d singular values all set to 1. The complete data matrix is formed by $\mathbf{X}_N = [\mathbf{x}(1), \cdots, \mathbf{x}(N)]$, with size $m \times N$.

In the subspace tracking, we use a sliding window of size n on $\mathbf{X}_N$ and compute the subspace of the corresponding data matrix $\mathbf{X}^{(i)} = [\mathbf{x}(i), \cdots, \mathbf{x}(i+n-1)]$, for $i = 1, 2, \cdots$. Starting from the decomposition for $\mathbf{X}^{(i-1)}$, this is implemented by applying an update by $\mathbf{x}(i+n-1)$ followed by a downdate by $\mathbf{x}(i-1)$ (i.e., an update by $\mathbf{x}(i-1)$ with an opposite signature). The SURV and HURV are initialized by setting $\mathbf{R} = \gamma \mathbf{I}_m$, where γ is the selected threshold (see below), followed by updates by $\mathbf{x}(1), \cdots, \mathbf{x}(n)$.

For reference, each $\mathbf{X}^{(i)}$ is also processed by an SVD to generate the "optimal" rank estimate $\hat{d}_{\text{svd}}$ corresponding to the threshold γ, and the principal subspace $\hat{\mathbf{U}}_{\text{svd}}$ formed by the singular vectors corresponding to the $\hat{d}_{\text{svd}}$ largest singular values. These are called the SVD estimates. Rank errors in the SURV occur when estimates $\hat{d} \neq \hat{d}_{\text{svd}}$ (compared to the SVD) or $\hat{d} \neq d$ (the true rank), depending on the criterion.

The "rank error rate" (RER) is defined by

$$\text{RER} = \frac{\text{Number of rank errors}}{\text{Total number of tests}}. \tag{27}$$

The error $e_{\hat{\mathbf{U}}}$ of a principal subspace estimate $\hat{\mathbf{U}}$ (where $\hat{\mathbf{U}}$ is a matrix whose columns form an orthonormal basis of the estimated subspace) is defined by the subspace angle,

$$\text{If} \quad \hat{d} = d, \quad \text{then} \quad e_{\hat{\mathbf{U}}} = \left\| \mathbf{P}_{\hat{\mathbf{U}}}^{\perp} \mathbf{U}_{\mathbf{H}} \right\|; \quad \text{else} \quad e_{\hat{\mathbf{U}}} = 1 \tag{28}$$

where $\mathbf{P}_{\hat{\mathbf{U}}}^{\perp} = \mathbf{I} - \hat{\mathbf{U}}\hat{\mathbf{U}}^H$, and columns of $\mathbf{U}_{\mathbf{H}}$ are the orthonormal basis of $\text{ran}(\mathbf{H})$.

For SURV and HURV, the "factorization error" is defined for each i as

$$e_{\text{f}}^{(i)} = \left\| \left(\gamma^2 \mathbf{I} - \mathbf{X}^{(i)} \mathbf{X}^{(i)^H} \right) - \mathbf{Q}\mathbf{R}\mathbf{J}\mathbf{R}^H\mathbf{Q}^H \right\| \tag{29}$$

where $\mathbf{Q}, \mathbf{R}$ are as defined in (20). For URV, which computes a decomposition $\mathbf{X}^{(i)} = \mathbf{U}\mathbf{R}\mathbf{V}^H$, it is defined as

$$e_{\text{f}}^{(i)} = \left\| \mathbf{X}^{(i)} \mathbf{X}^{(i)^H} - \mathbf{U}\mathbf{R}\mathbf{R}^H\mathbf{U}^H \right\|. \tag{30}$$

For the SVD, when computes a decomposition $\mathbf{X}^{(i)} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H$, it is defined as

$$e_{\text{f}}^{(i)} = \left\| \mathbf{X}^{(i)} \mathbf{X}^{(i)^H} - \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^H \right\|. \tag{31}$$

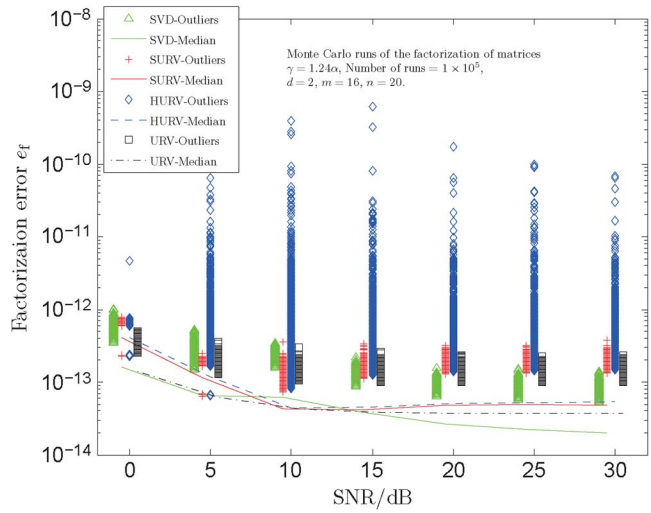

Fig. 4. Factorization error of SURV, HURV, and URV on random matrices.

The SNR is defined as $\text{SNR} = 10 \log \left(\frac{\sigma_s^2}{\sigma_n^2} \right)$. The same rank threshold

$$\gamma = 1.24\alpha \quad \text{where} \quad \alpha = \sigma_n \left(1 + \sqrt{\frac{m}{n}} \right) \sqrt{n}$$

is given to all algorithms, where α is an estimate of the largest singular value of a "noise only" data matrix $\mathbf{N} : m \times n$ [34]–[36], and γ is a scaled version that usually results in correct rank estimates (no "false alarm").

In the simulations, we used $m = 16$ and $n = 20$ unless specified otherwise. "HURV" refers to the previously proposed algorithm in [11] for computing the same decomposition (Theorem 1) as the SURV. The "URV" algorithm is based on the Matlab code taken from UTV-tools [10], which we slightly modified to make it run in nonstationary cases.

Fig. 4 shows the boxplot of the factorization error of SURV, HURV and URV on random matrices of size 16×20 over 10^5 Monte Carlo runs at given SNRs. The factorization error of the SVD is used as the reference. It is seen from Fig. 4 that HURV sometimes is not as stable as SURV and URV. The reason is that HURV uses at most three hyperbolic rotations per vector update (and also tracks two entries of $\mathbf{\Theta}$), and a situation that two large hyperbolic rotations cancel each other might happen. However, SURV uses at most one hyperbolic rotation per vector update and then gives a stable factorization.

Fig. 5 shows the factorization error of SURV, HURV and URV for tracking in stationary and nonstationary cases (single run). $N \approx 10^5$. In Fig. 5(a), $d = 2$. In Fig. 5(b), d switches between 2 and 4 every 150 samples. In Fig. 5(c), $m = n = 16$ and d switches between 8 and 16 every 150 samples at very high SNR (SNR $= 250$ dB). It is seen that URV encounters breakdowns frequently and HURV encounters breakdowns at high SNR while such breakdowns never occur for SURV.

Fig. 6 shows the subspace tracking performance of SURV, HURV and URV in a stationary case as a function of SNR. Fig. 6(a), (b), (c) show the rank error rate compared with the real number of signals d, the rank error rate compared with $\hat{d}_{\text{svd}}$, and the averaged subspace error of the estimated principal subspace over $2 \cdot 10^4$ updates per run $(N \approx 10^4)$ and 10 runs, respectively. It is seen that the rank estimate of URV is unreliable at

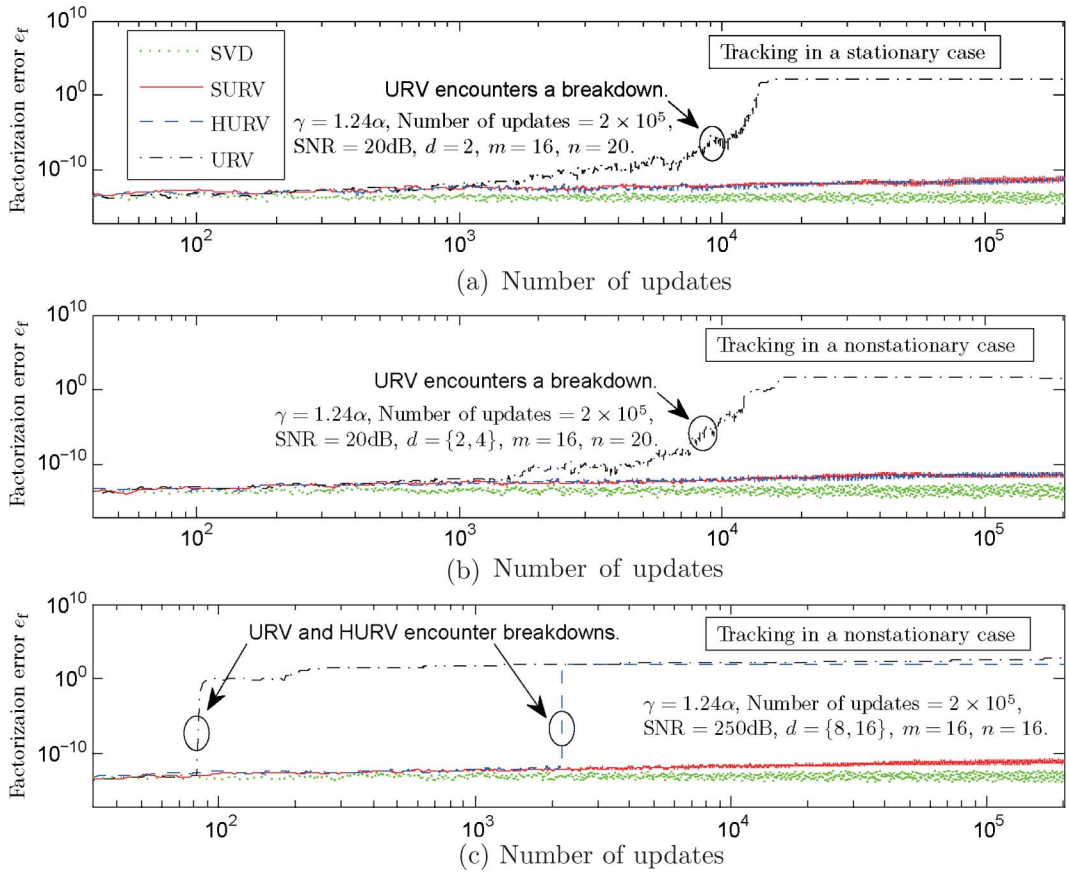

Fig. 5. Factorization error of SURV, HURV and URV for tracking in stationary and nonstationary cases (single run).

low SNR, while SURV and HURV always give rank estimates consistent with the SVD. This implies that URV cannot track the rank of the subspace well in nonstationary cases. The subspace estimates of SURV and HURV stay closer to $\mathrm{ran}(\mathbf{H})$ than the others.

Fig. 7 shows the subspace tracking performance of SURV, HURV, and URV in a nonstationary case as a function of SNR, with the true rank d switching between 2 and 4 sources every 150 samples. Fig. 7 is similar to Fig. 6 except that, in Fig. 7(a) and (c), we collect the performance statistics only for the stationary parts, omitting the size-n transient parts during rank changes. It is seen from Fig. 7 that SURV and HURV always give rank estimates consistent with the SVD, as well as good estimates of the true principal (signal) subspace $\mathrm{ran}(\mathbf{H})$. However, URV gives many rank errors and encounters breakdowns even at high SNR.

In addition, we show a case where the subspace estimates of the compared algorithms are applied for direction-of-arrival (DOA) estimation where a classic algorithm, ESPRIT, is used (relevant results can be found in [27]). A linear antenna array with $m = 8$ elements spacing at half wavelengths is used. The channel matrix $\mathbf{H}(t, \theta)$ now consists of steering vectors, which are complex vectors nonorthogonal to each other. A nonstationary scenario is chosen, where d switches between 2 (two sources with DOA $\boldsymbol{\theta} = [-20^\circ, 20^\circ]^T$) and 4 (four sources with DOA $\boldsymbol{\theta} = [40^\circ, -50^\circ, 0^\circ, 70^\circ]^T$) every 150 samples. The source signal is BPSK modulated (real-valued symbols from the alphabet $\{+1, -1\}$). All sources have equal signal power. Due to the limitation that URV codes from [10] run only on real-valued data matrices, we split the real and imaginary parts of the complex matrix $\mathbf{X}^{(i)}$ and stack them up to form a new real-valued data matrix

$$\mathbf{X}^{(i)}_{\mathrm{stack}} = \begin{matrix} m \\ m \end{matrix} \begin{bmatrix} \mathrm{Re}\left(\mathbf{X}^{(i)}\right) \\ \mathrm{Im}\left(\mathbf{X}^{(i)}\right) \end{bmatrix}. \tag{32}$$

The subspace estimate $\mathbf{U} = \begin{matrix} m \\ m \end{matrix} \begin{bmatrix} \hat{\mathbf{U}}_1 \\ \hat{\mathbf{U}}_2 \end{bmatrix}$ from each of the tracking algorithms is split into two submatrices to form the complex matrix

$$\hat{\mathbf{U}}_{\mathrm{unstack}} = \hat{\mathbf{U}}_1 + j\hat{\mathbf{U}}_2 \tag{33}$$

which is passed to ESPRIT. We set $2m \leq n$ due to the problem that URV [10] refuses to run when $\mathbf{X}^{(i)}_{\mathrm{stack}}$ is tall. In contrast, SURV and HURV do not have this limitation and still work well on any tall matrices. For fairness of comparison, we applied SURV, HURV, and the SVD to the same $\mathbf{X}^{(i)}_{\mathrm{stack}}$. The performance measure of DOA estimation is the root-mean-square error (RMSE)

$$\mathrm{RMSE} = \sqrt{\mathrm{avg}\left(\sum_{k=1}^{10}\sum_{i} \min_{\boldsymbol{\Pi}_1 \text{ or } \boldsymbol{\Pi}_2} \mathrm{avg}\|\boldsymbol{\Pi}_1\hat{\boldsymbol{\theta}}_{ik} - \boldsymbol{\Pi}_2\boldsymbol{\theta}_{ik}\|_E^2\right)} \tag{34}$$

where $\|\cdot\|_E$ denotes the Euclidean norm, $\hat{\boldsymbol{\theta}}_{ik}$ and $\boldsymbol{\theta}_{ik}$ are the DOA estimates and the true DOA for $\mathbf{X}^{(i)}$ in the kth run respectively, the length of $\hat{\boldsymbol{\theta}}_{ik}$ is limited to at most 4 (as the length of $\boldsymbol{\theta}_{ik}$ is at most 4), and $\boldsymbol{\Pi}_1$ and $\boldsymbol{\Pi}_2$ are row selection matrices with compatible size to $\hat{\boldsymbol{\theta}}_{ik}$ and $\boldsymbol{\theta}_{ik}$ to generate all possible row

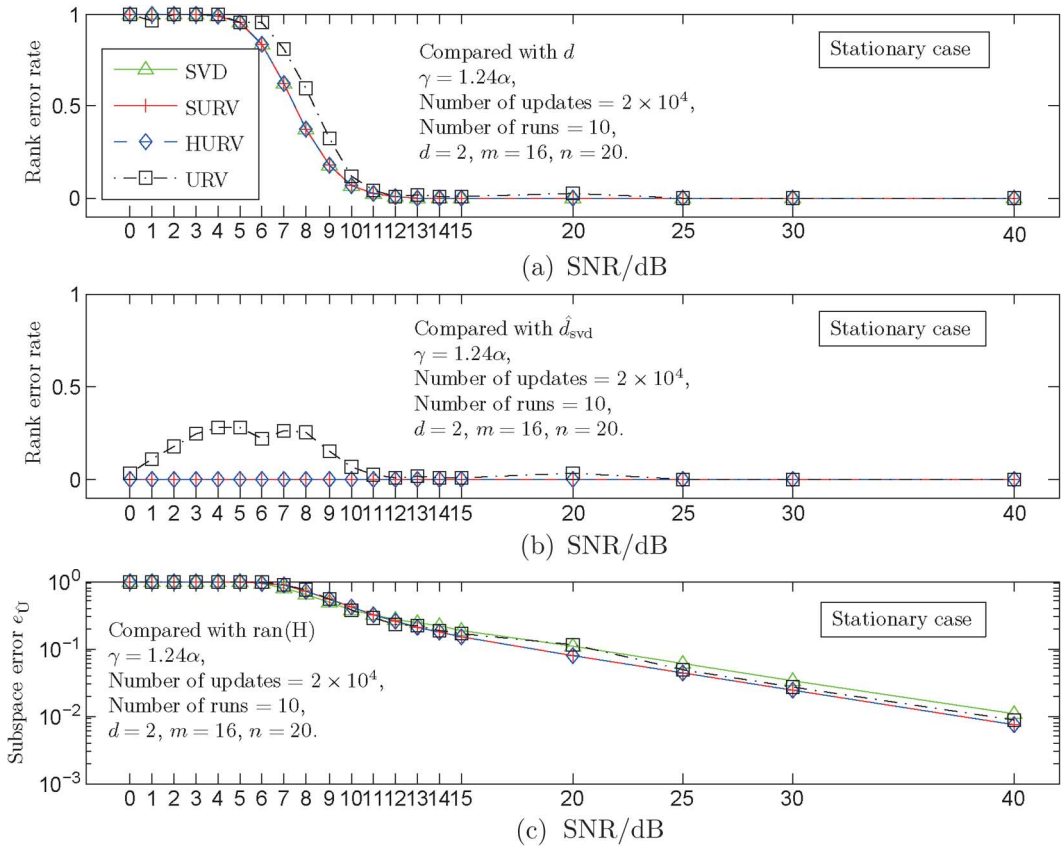

Fig. 6. Subspace tracking performance of SURV, HURV, and URV in a stationary case as a function of SNR.

selections. $\mathbf{\Pi}_1 = \mathbf{I}$, when the length of $\hat{\boldsymbol{\theta}}_{ik}$ is shorter than $\boldsymbol{\theta}_{ik}$; $\mathbf{\Pi}_2 = \mathbf{I}$, otherwise.

Fig. 8 shows the performance of DOA estimation using subspace estimates from SURV, HURV, and URV tracking in the above nonstationary case. The performance of the SVD is used as the reference. The rank threshold is set to $\gamma = 1.28\alpha$ due to the change of m. In Fig. 8(a) and (c), we collect the performance statistics only for the stationary parts like that in Fig. 7. SURV and HURV again show very good tracking performance that the rank estimates are exact and the subspace estimates are good and useful. The averaged DOA performance of SURV and HURV stays quite close to and even better than the SVD. URV shows unstable performance as it frequently encounters breakdowns during tracking, where the rank can be overestimated resulting in bad DOA estimates.

VII. Conclusion

This paper proposed a Hyperbolic URV decomposition and its updating algorithm (SURV) for subspace tracking. It is a noniterative algorithm that provides exact rank estimates and very good principal subspace estimates compared with the SVD with the same rank threshold. The subspace is provided in terms of an orthogonal basis $\mathbf{Q_B}$. The proposed updating algorithm uses at most only a single hyperbolic rotation per vector update and provides a numerically stable computation of the "SSE-2" Schur subspace estimator (unlike the previously proposed algorithm in [11]).

The proposed updating algorithm has a computational complexity of $O(m^2)$ per size-m update vector (similar to a QR update), and has constant $O(m^2)$ memory requirements. The algorithm is based on a sliding-window update scheme, but is easily tailored to exponential windowing by scaling matrix $\mathbf{R}$ after every update. All operations in the proposed updating algorithm are local, consisting of elementary rotations and column permutations, and the computational flow is not iterative, thus facilitating parallel implementations.

The proposed algorithm can replace an SVD in cases where we are only interested in the subspace, and have a threshold on the noise power for the splitting of the subspace. The computational complexity is higher than some of the tracking algorithms with knowledge of the rank, such as PAST, but the advantage is that the estimate is "exact" with known properties and does not rely on convergence.

Future work can include the derivation of a spherical update, where the noise subspace is not tracked but replaced by an average property.

Appendix A
The Principal Subspace is an SSE-2

We show that for a given matrix $\mathbf{X}$, there is an SSE-2 decomposition of the required form (24), (25) such that the "SSE-2 subspace" provided by $\mathbf{Q_B}$ is equal to the left principal subspace of $\mathbf{X}$ as provided by the SVD.

Suppose for simplicity of notation that $\mathbf{X}$ is square, and has d singular values larger than d. Let the SVD of $\mathbf{X}$ be defined

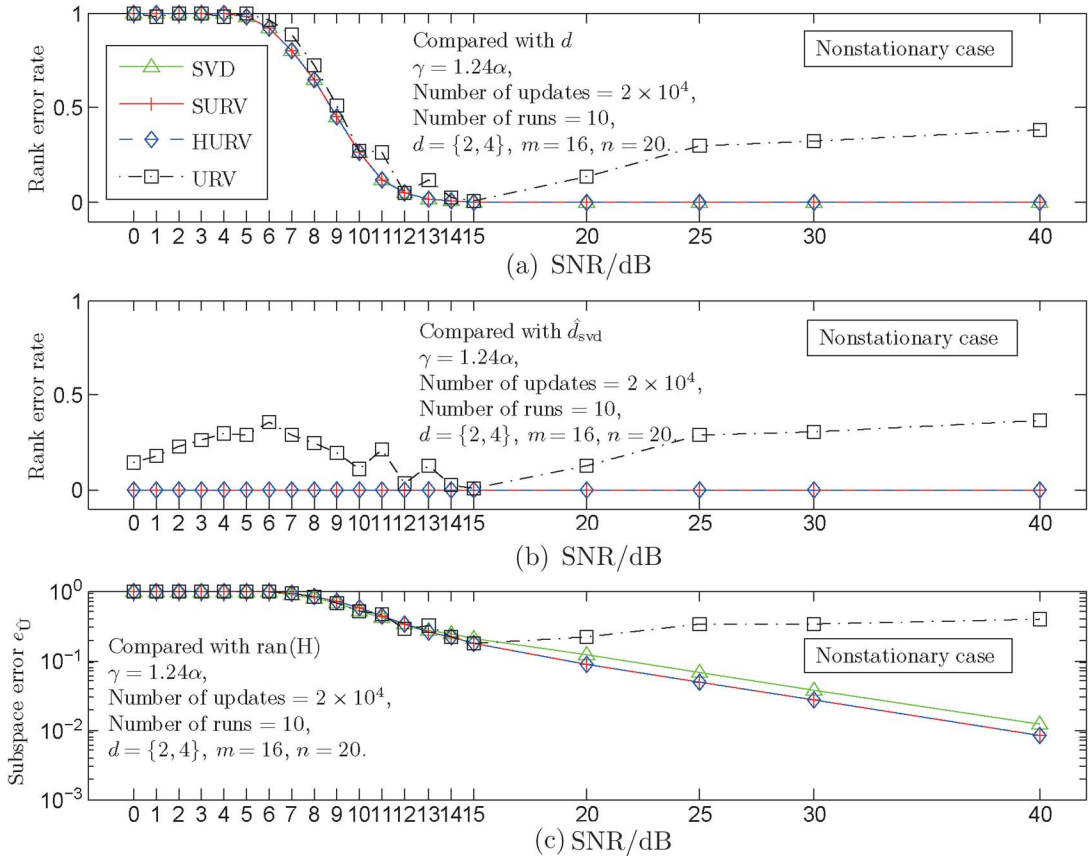

Fig. 7. Subspace tracking performance of SURV, HURV, and URV in a nonstationary case as a function of SNR.

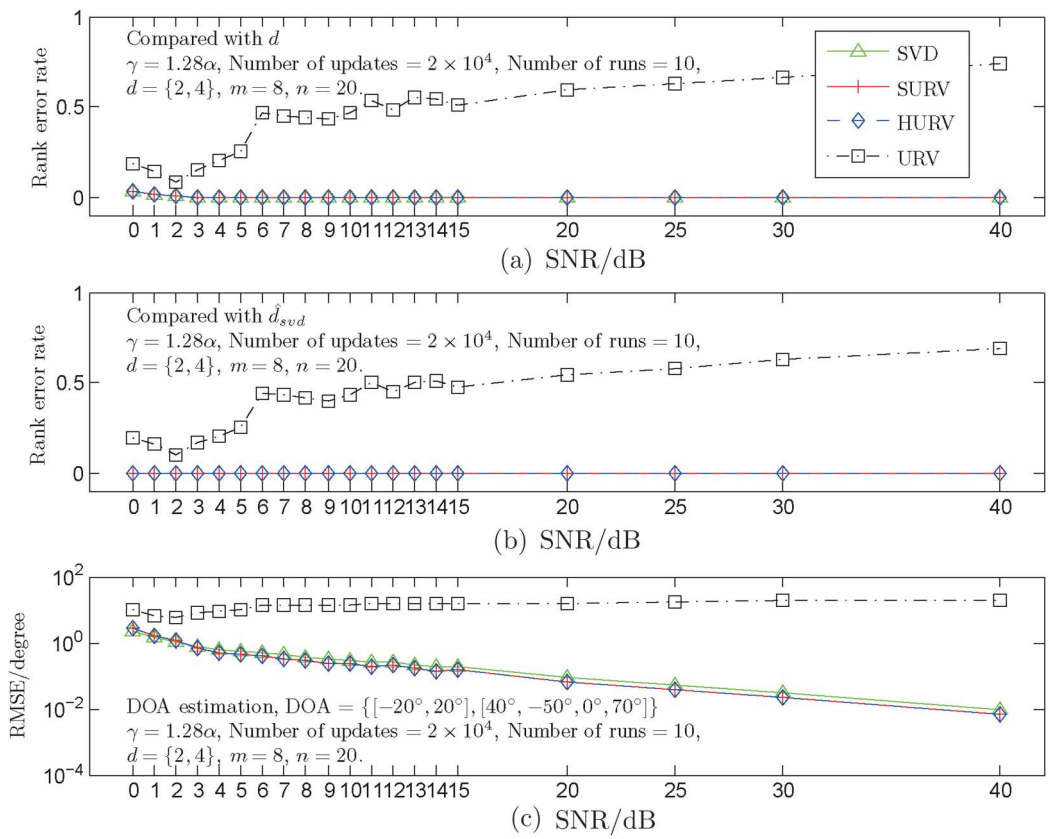

Fig. 8. Performance of DOA estimation using subspace estimates from SURV, HURV, and URV tracking in a nonstationary case as a function of SNR.

by $\mathbf{X} = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^H + \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^H$, where $\mathbf{U}_1$ and $\mathbf{V}_1$ have d columns, $\mathbf{U}_2$ and $\mathbf{V}_2$ have $m-d$ columns, $\mathbf{\Sigma}_1 : d \times d$ and $\mathbf{\Sigma}_2 : (m-d)\times(m-d)$ are diagonal matrices, and $\mathbf{\Sigma}_1 > \gamma$ and $\mathbf{\Sigma}_2 < \gamma$. Define $\mathbf{Q}, \mathbf{\Theta}, \mathbf{T}$ as

$$\mathbf{Q} = [\,\mathbf{U}_2 \quad \mathbf{U}_1\,]$$

$$\mathbf{\Theta} = \left[\begin{array}{cc|cc} \mathbf{U}_2\gamma & \mathbf{U}_1\mathbf{\Sigma}_1 & -\mathbf{U}_1\gamma & -\mathbf{U}_2\mathbf{\Sigma}_2 \\ -\mathbf{V}_2\mathbf{\Sigma}_2 & -\mathbf{V}_1\gamma & \mathbf{V}_1\mathbf{\Sigma}_1 & \mathbf{V}_2\gamma \end{array}\right] \cdot \begin{bmatrix} \gamma^2-\mathbf{\Sigma}_2^2 & & & \\ & \mathbf{\Sigma}_1^2-\gamma^2 & & \\ & & \mathbf{\Sigma}_1^2-\gamma^2 & \\ & & & \gamma^2-\mathbf{\Sigma}_2^2 \end{bmatrix}^{-1/2}$$

$$\mathbf{T} = \mathbf{\Theta}_{11}^{-1} = \begin{bmatrix} \left(\gamma^2-\mathbf{\Sigma}_2^2\right)^{\frac{1}{2}}\gamma^{-1}\mathbf{U}_2^H \\ \left(\mathbf{\Sigma}_1^2-\gamma^2\right)^{\frac{1}{2}}\mathbf{\Sigma}_1^{-1}\mathbf{U}_1^H \end{bmatrix}.$$

It is readily verified that $\mathbf{Q}$ is unitary, $\mathbf{\Theta}$ is $\mathbf{J}$-unitary, and that

$$\mathbf{Q}^H[\,\gamma\mathbf{I} \quad \mathbf{X}\,]\mathbf{\Theta} = \left[\begin{array}{cc|cc} \left(\gamma^2-\mathbf{\Sigma}_2^2\right)^{\frac{1}{2}} & 0 & 0 & 0 \\ 0 & 0 & \left(\mathbf{\Sigma}_1^2-\gamma^2\right)^{\frac{1}{2}} & 0 \end{array}\right]$$

$$\mathbf{T}[\,\mathbf{I} \quad 0\,]\mathbf{\Theta} = \left[\begin{array}{cc|cc} \mathbf{I} & \mathbf{0} & \mathbf{0} & -\gamma^{-1}\mathbf{\Sigma}_2 \\ \mathbf{0} & \mathbf{I} & -\gamma\mathbf{\Sigma}_1^{-1} & \mathbf{0} \end{array}\right]$$

so that (24), (25) hold. Since $\mathbf{Q}_\mathbf{B} = \mathbf{U}_1$, the SSE-2 subspace is equal to the principal subspace obtained by an SVD.

Appendix B
Proof for Theorem 1 in Section V

The proof is technical and consists of several steps (lemmas).

Lemma 2: For given matrices $\mathbf{A} : m \times (m-d)$, $\mathbf{B} : m \times d$, and a $\mathbf{J}$-unitary matrix $\mathbf{\Theta} : (n_1+n_2)\times(n_1+n_2)$, consider a transformation by any $\mathbf{J}$-unitary matrix $\mathbf{\Theta}_\mathbf{M}$:

$$[\mathbf{A} \quad 0|\mathbf{B} \quad 0]\mathbf{\Theta}_\mathbf{M} = [\mathbf{A}' \quad 0|\mathbf{B}' \quad 0] \tag{35}$$

$$\mathbf{\Theta}\mathbf{\Theta}_\mathbf{M} = \mathbf{\Theta}' \tag{36}$$

where $\mathbf{\Theta}_\mathbf{M}$ only acts on the columns of $\mathbf{A}, \mathbf{B}$ (and corresponding columns of $\mathbf{\Theta}$). Define $\mathbf{M}_\mathbf{\Theta}$ as in (23), and likewise for $\mathbf{M}_{\mathbf{\Theta}'}$.

Then $\mathrm{ran}(\mathbf{B} - \mathbf{A}\mathbf{M}_\mathbf{\Theta}) = \mathrm{ran}(\mathbf{B}' - \mathbf{A}'\mathbf{M}_{\mathbf{\Theta}'})$, i.e., the SSE-2 subspace is invariant under $\mathbf{\Theta}_\mathbf{M}$.

Proof: Use the block partitioning (22) of $\mathbf{\Theta}$, and note that

$$[\mathbf{\Theta}_{11} \quad \mathbf{\Theta}_{12}]\begin{bmatrix} -\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12} \\ \mathbf{I}_{n_2} \end{bmatrix} = \mathbf{0} \tag{37}$$

$$\begin{bmatrix} \mathbf{\Theta}'_{11} & \mathbf{\Theta}'_{12} \end{bmatrix}\begin{bmatrix} -\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12} \\ \mathbf{I}_{n_2} \end{bmatrix} = \mathbf{0}. \tag{38}$$

Also note that the right hand side matrix factors are full rank complements with equal column dimensions n_2. From $\mathbf{\Theta}\mathbf{\Theta}_\mathbf{M} = \mathbf{\Theta}'$, we obtain

$$[\mathbf{\Theta}_{11} \quad \mathbf{\Theta}_{12}]\mathbf{\Theta}_\mathbf{M} = \begin{bmatrix} \mathbf{\Theta}'_{11} & \mathbf{\Theta}'_{12} \end{bmatrix}. \tag{39}$$

Inserting in (38) gives

$$[\mathbf{\Theta}_{11} \quad \mathbf{\Theta}_{12}]\mathbf{\Theta}_\mathbf{M}\begin{bmatrix} -\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12} \\ \mathbf{I}_{n_2} \end{bmatrix} = \mathbf{0}. \tag{40}$$

Comparing with (37) and using the full rank complement property reveals that there exists a $n_2 \times n_2$ invertible matrix $\mathbf{V}$ such that

$$\mathbf{\Theta}_\mathbf{M}\begin{bmatrix} -\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12} \\ \mathbf{I}_{n_2} \end{bmatrix} = \begin{bmatrix} -\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12} \\ \mathbf{I}_{n_2} \end{bmatrix}\mathbf{V} = \begin{bmatrix} -\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\mathbf{V} \\ \mathbf{V} \end{bmatrix}. \tag{41}$$

We next show that $\mathbf{V}$ is a block-upper triangular matrix. Because $\mathbf{\Theta}_\mathbf{M}$ only acts on the columns of $\mathbf{A}, \mathbf{B}$, we can write the block structure of $\mathbf{\Theta}_\mathbf{M}$ as

$$\mathbf{\Theta}_\mathbf{M} = \begin{array}{c} \\ n_1 \\ n_2 \end{array}\!\!\begin{array}{c} \overset{n_1}{} \\ \left[\begin{array}{cc} (\mathbf{\Theta}_\mathbf{M})_{11} & (\mathbf{\Theta}_\mathbf{M})_{12} \\ (\mathbf{\Theta}_\mathbf{M})_{21} & (\mathbf{\Theta}_\mathbf{M})_{22} \end{array}\right] \end{array}\!\!\overset{n_2}{}$$

$$= \begin{array}{c} \\ m-d \\ n_1-m+d \\ d \\ n_2-d \end{array}\!\!\begin{array}{c} \overset{m-d \quad n_1-_{m+d}}{} \\ \left[\begin{array}{cc|cc} \begin{bmatrix}\mathbf{P}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}\end{bmatrix} & & \begin{bmatrix}\mathbf{P}_{12} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{bmatrix} & \\ \hline \begin{bmatrix}\mathbf{P}_{21} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{bmatrix} & & \begin{bmatrix}\mathbf{P}_{22} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}\end{bmatrix} & \end{array}\right] \end{array}\!\!\overset{d \quad n_2-d}{} \tag{42}$$

where the identity submatrices in $\mathbf{\Theta}_\mathbf{M}$ ensure that the columns outside $\mathbf{A}, \mathbf{B}$ remain unchanged, and the submatrices $\mathbf{P}_{ij}$, $i, j = 1, 2$ transform the columns of $\mathbf{A}, \mathbf{B}$. Now, using the expression for $\mathbf{V}$ in the lower part of (41), and the partitioning of $\mathbf{\Theta}_\mathbf{M}$, we find

$$\mathbf{V} = (\mathbf{\Theta}_\mathbf{M})_{22} - (\mathbf{\Theta}_\mathbf{M})_{21}\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12} \tag{43}$$

$$= \begin{bmatrix}\mathbf{P}_{22} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{n_2-d}\end{bmatrix} - \begin{bmatrix}\mathbf{P}_{21} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{bmatrix} \cdot \begin{bmatrix} \left(\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12}\right)_{11} & \left(\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12}\right)_{12} \\ \left(\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12}\right)_{21} & \left(\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12}\right)_{22} \end{bmatrix}$$

$$= \begin{bmatrix}\mathbf{V}_{11} & \mathbf{V}_{12} \\ \mathbf{0} & \mathbf{V}_{22}\end{bmatrix}. \tag{44}$$

As $\mathbf{V}$ is invertible, this block structure also implies that $\mathbf{V}_{11}$ is invertible. Furthermore, we have

$$\mathbf{B}' - \mathbf{A}'\mathbf{M}_{\mathbf{\Theta}'} = [\mathbf{B}' \quad 0]\begin{bmatrix}\mathbf{I} \\ \mathbf{0}\end{bmatrix} - [\mathbf{A}' \quad 0]\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12}\begin{bmatrix}\mathbf{I} \\ \mathbf{0}\end{bmatrix}$$

$$= [\mathbf{A}' \quad \mathbf{0} \quad | \quad \mathbf{B}' \quad \mathbf{0}]\begin{bmatrix} -\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12} \\ \mathbf{I}_{n_2} \end{bmatrix} \tag{45}$$

$$\overset{(35)}{=} [\mathbf{A} \quad \mathbf{0} \quad | \quad \mathbf{B} \quad \mathbf{0}]\mathbf{\Theta}_\mathbf{M}\begin{bmatrix} -\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12} \\ \mathbf{I}_{n_2} \end{bmatrix}\cdot\begin{bmatrix}\mathbf{I}_d \\ \mathbf{0}\end{bmatrix} \tag{46}$$

$$\overset{(41)}{=} [\mathbf{A} \quad \mathbf{0} \quad | \quad \mathbf{B} \quad \mathbf{0}]\begin{bmatrix} -\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12} \\ \mathbf{I}_{n_2} \end{bmatrix}\mathbf{V}\begin{bmatrix}\mathbf{I}_d \\ \mathbf{0}\end{bmatrix}$$

$$\overset{(44)}{=} \left([\mathbf{B} \quad 0] - [\mathbf{A} \quad 0]\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)\cdot\begin{bmatrix}\mathbf{V}_{11} \\ \mathbf{0}\end{bmatrix}. \tag{47}$$

Now, we partition $\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}$ into blocks as

$$\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12} = \begin{array}{c} \\ m-d \\ n_1-m+d \end{array}\!\!\begin{array}{c} \overset{d \quad n_2-d}{} \\ \left[\begin{array}{cc} \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{11} & \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{12} \\ \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{21} & \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{22} \end{array}\right] \end{array}. \tag{48}$$

Using this partitioning, we obtain

$$\begin{aligned}\mathbf{B}' - \mathbf{A}'\mathbf{M}_{\mathbf{\Theta}'} &= \left([\mathbf{B} \quad \mathbf{0}] - [\mathbf{A} \quad \mathbf{0}]\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right) \cdot \begin{bmatrix}\mathbf{V}_{11}\\ \mathbf{0}\end{bmatrix} \\ &= \left[\mathbf{B} - \mathbf{A}\left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{11} \quad - \mathbf{A}\left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{12}\right]\begin{bmatrix}\mathbf{V}_{11}\\ \mathbf{0}\end{bmatrix} \\ &= (\mathbf{B} - \mathbf{A}\mathbf{M}_{\mathbf{\Theta}})\mathbf{V}_{11}. \end{aligned} \tag{49}$$

Because $\mathbf{V}_{11}$ is invertible, we reach the desired conclusion:

$$\operatorname{ran}(\mathbf{B}' - \mathbf{A}'\mathbf{M}_{\mathbf{\Theta}'}) = \operatorname{ran}(\mathbf{B} - \mathbf{A}\mathbf{M}_{\mathbf{\Theta}}). \tag{50}$$

■

Lemma 3: In the context of Lemma 2, there exists a transformation $\mathbf{\Theta}_{\mathbf{M}}$ such that $\mathbf{M}_{\mathbf{\Theta}'} = \mathbf{0}$, i.e., such that $\operatorname{ran}(\mathbf{B}')$ is the SSE-2 subspace.

Proof: We will first explicitly construct a suitable $\mathbf{\Theta}_{\mathbf{M}}$ from $\mathbf{M}_{\mathbf{\Theta}}$. Since $\|\mathbf{M}_{\mathbf{\Theta}}\| = \|(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12})_{11}\| \leq \|\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\| < 1$ [8], $\mathbf{M}_{\mathbf{\Theta}}$ is strictly contractive. Thus, we know from Theorem 2.1 in [8] that there exists a $\mathbf{J}$-unitary matrix $\mathbf{P}$, with

$$\mathbf{J} = \begin{bmatrix}\mathbf{I}_{m-d} & \mathbf{0}\\ \mathbf{0} & -\mathbf{I}_d\end{bmatrix}, \qquad \mathbf{P} = \begin{bmatrix}\mathbf{P}_{11} & \mathbf{P}_{12}\\ \mathbf{P}_{21} & \mathbf{P}_{22}\end{bmatrix},$$

and an invertible matrix $\mathbf{Y} : (m-d) \times (m-d)$ such that

$$[\mathbf{I}_{m-d} \quad \mathbf{M}_{\mathbf{\Theta}}]\begin{bmatrix}\mathbf{P}_{11} & \mathbf{P}_{12}\\ \mathbf{P}_{21} & \mathbf{P}_{22}\end{bmatrix} = [\mathbf{Y} \quad \mathbf{0}_{(m-d)\times d}]. \tag{51}$$

Equation (51) implies

$$\mathbf{P}_{12} + \mathbf{M}_{\mathbf{\Theta}}\mathbf{P}_{22} = \mathbf{P}_{12} + \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{11}\mathbf{P}_{22} = \mathbf{0}. \tag{52}$$

A suitable $\mathbf{\Theta}_{\mathbf{M}}$ related to $\mathbf{M}_{\mathbf{\Theta}}$ is now obtained by augmenting $\mathbf{P}$ to a full-size $\mathbf{J}$-unitary matrix $\mathbf{\Theta}_{\mathbf{M}}$ as in (42). After defining $\mathbf{\Theta}' = \mathbf{\Theta}\mathbf{\Theta}_{\mathbf{M}}$, we now need to show the claim that $\mathbf{M}_{\mathbf{\Theta}'} = \mathbf{0}$. Using the block partitioning of $\mathbf{\Theta}$ as in (22), we obtain

$$[\mathbf{\Theta}'_{11} \quad \mathbf{\Theta}'_{12}] = [\mathbf{\Theta}_{11} \quad \mathbf{\Theta}_{12}]\mathbf{\Theta}_{\mathbf{M}} \tag{53}$$

and hence

$$\mathbf{\Theta}_{11}^{-1}[\mathbf{\Theta}'_{11} \quad \mathbf{\Theta}'_{12}] = \left[\mathbf{I} \quad \mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right]\mathbf{\Theta}_{\mathbf{M}} \tag{54}$$

After completing the matrix multiplication and inserting (52), we obtain

$$\begin{aligned}\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}'_{11} &= \begin{bmatrix}\mathbf{P}_{11} & \\ & \mathbf{I}\end{bmatrix} + \mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\begin{bmatrix}\mathbf{P}_{21} & \\ & \mathbf{I}\end{bmatrix} \\ &= \begin{bmatrix}\mathbf{P}_{11} + \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{11}\mathbf{P}_{21} & \mathbf{0}\\ \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{21}\mathbf{P}_{21} & \mathbf{I}\end{bmatrix} \end{aligned} \tag{55}$$

$$\begin{aligned}\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}'_{12} &= \begin{bmatrix}\mathbf{P}_{12} & \\ & \mathbf{I}\end{bmatrix} + \mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\begin{bmatrix}\mathbf{P}_{22} & \\ & \mathbf{I}\end{bmatrix} \\ &\overset{(52)}{=} \begin{bmatrix}\mathbf{0} & \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{12}\\ \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{21}\mathbf{P}_{22} & \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{22}\end{bmatrix}. \end{aligned} \tag{56}$$

Due to the block triangular structure of these matrices, we obtain that

$$\begin{aligned}\mathbf{M}_{\mathbf{\Theta}'} &= [\mathbf{I} \quad \mathbf{0}]\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12}\begin{bmatrix}\mathbf{I}\\ \mathbf{0}\end{bmatrix} \\ &= [\mathbf{I} \quad \mathbf{0}]\left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}'_{11}\right)^{-1}\left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}'_{12}\right)\begin{bmatrix}\mathbf{I}\\ \mathbf{0}\end{bmatrix} \\ &\overset{(55)(56)}{=} [\mathbf{I} \quad \mathbf{0}]\begin{bmatrix}* & \mathbf{0}\\ * & *\end{bmatrix}\begin{bmatrix}\mathbf{0} & *\\ * & *\end{bmatrix}\begin{bmatrix}\mathbf{I}\\ \mathbf{0}\end{bmatrix} = \mathbf{0}, \end{aligned} \tag{57}$$

and hence

$$\operatorname{ran}(\mathbf{B}' - \mathbf{A}'\mathbf{M}_{\mathbf{\Theta}'}) = \operatorname{ran}(\mathbf{B}').$$

■

Hence, we showed that there exists a matrix $\mathbf{\Theta}_{\mathbf{M}}$ (defined explicitly from $\mathbf{M}_{\mathbf{\Theta}}$ via (51)), which transforms any $\mathbf{\Theta}$ to $\mathbf{\Theta}' = \mathbf{\Theta}\mathbf{\Theta}_{\mathbf{M}}$, such that after the transformation we have $\mathbf{M}_{\mathbf{\Theta}'} = \mathbf{0}$, implying that we simply can take the corresponding $\mathbf{B}'$ to get the desired SSE-2 subspace. Knowing this, there are easier ways to find this transformation. Suppose $\mathbf{\Theta}$ is partitioned as

$$\begin{aligned}\mathbf{\Theta} &= \begin{bmatrix}\mathbf{\Theta}_{11} & \mathbf{\Theta}_{12}\\ \mathbf{\Theta}_{21} & \mathbf{\Theta}_{22}\end{bmatrix} \\ &= \begin{array}{c} \\ m-d \\ n_1-m+d \\ d \\ n_2-d \end{array}\begin{array}{c}\begin{array}{cc}\overset{m-d}{} & \overset{n_1-m+d}{}\end{array}\\ \left[\begin{array}{cc|cc} (\mathbf{\Theta}_{11})_{11} & (\mathbf{\Theta}_{11})_{12} & (\mathbf{\Theta}_{12})_{11} & (\mathbf{\Theta}_{12})_{12}\\ * & * & * & *\\ \hline * & * & * & *\\ * & * & * & * \end{array}\right.\end{array}\begin{array}{c}\begin{array}{cc}\overset{d}{} & \overset{n_2-d}{}\end{array}\\ \left.\vphantom{\begin{array}{cc|cc} a & a & a & a\\ a & a & a & a\\ \hline a & a & a & a\\ a & a & a & a \end{array}}\right]\end{array}. \end{aligned} \tag{58}$$

To have $\mathbf{M}_{\mathbf{\Theta}'} = \mathbf{0}$, we just have to find a transformation on $\mathbf{\Theta}$ such that the resulting $\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12}$ has a zero (11)-block. This will be the case, for example, if both $(\mathbf{\Theta}'_{11})_{12} = \mathbf{0}$ and $(\mathbf{\Theta}'_{12})_{11} = \mathbf{0}$. One complication is that, by definition, $\mathbf{\Theta}_{\mathbf{M}}$ is not allowed to change the columns of $(\mathbf{\Theta}_{11})_{12}$, for else the range of $\mathbf{B} - \mathbf{A}\mathbf{M}_{\mathbf{\Theta}}$ will not stay invariant. Thus, we cannot use $\mathbf{\Theta}$ to zero this block. However, we may apply any invertible transformation $\mathbf{T} : n_1 \times n_1$ to the *rows* of $[\mathbf{\Theta}_{11} \ \mathbf{\Theta}_{12}]$:

$$[\mathbf{\Theta}'_{11} \quad \mathbf{\Theta}'_{12}] = \mathbf{T}[\mathbf{\Theta}_{11} \quad \mathbf{\Theta}_{12}],$$

because $\mathbf{\Theta}_{11}'^{-1}\mathbf{\Theta}'_{12} = \mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}$ is invariant under $\mathbf{T}$. This motivates the specified decomposition (24) and (25) in Theorem 1. We continue to show its existence:

Lemma 4: Let be given matrices $\mathbf{A} : m \times (m-d)$, $\mathbf{B} : m \times d$, and a $\mathbf{J}$-unitary matrix $\mathbf{\Theta} : (n_1+n_2) \times (n_1+n_2)$. Then there exists a $\mathbf{J}$-unitary matrix $\mathbf{\Theta}_{\mathbf{M}}$ (only acting on the columns of $\mathbf{A}, \mathbf{B}$) and an invertible matrix $\mathbf{T}$ such that

$$[\mathbf{A} \quad \mathbf{0}|\mathbf{B} \quad \mathbf{0}]\mathbf{\Theta}_{\mathbf{M}} = [\mathbf{A}' \quad \mathbf{0}|\mathbf{B}' \quad \mathbf{0}], \tag{59}$$

$$\mathbf{\Theta}\mathbf{\Theta}_{\mathbf{M}} = \mathbf{\Theta}', \tag{60}$$

$$\mathbf{T}[\mathbf{\Theta}'_{11} \quad \mathbf{\Theta}'_{12}] = \begin{array}{c} \\ m-d \\ n_1-m+d \end{array}\begin{array}{c}\begin{array}{cc}\overset{m-d}{+} & \overset{n_1-m+d}{+}\end{array}\\ \left[\begin{array}{cc|cc} \mathbf{I} & \mathbf{0} & \mathbf{0} & *\\ \mathbf{0} & \mathbf{I} & * & * \end{array}\right.\end{array}\begin{array}{c}\begin{array}{cc}\overset{d}{-} & \overset{n_2-d}{-}\end{array}\\ \left.\vphantom{\begin{array}{cc|cc} a & a & a & a\\ a & a & a & a \end{array}}\right]\end{array}, \tag{61}$$

hence such that $\mathbf{M}_{\mathbf{\Theta}'} = \mathbf{0}$.

Proof: The proof is by construction. Given $\mathbf{\Theta}$, we form $\mathbf{M}_{\mathbf{\Theta}} = [\mathbf{I} \ \mathbf{0}]\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\left[\begin{smallmatrix}\mathbf{I}\\ \mathbf{0}\end{smallmatrix}\right]$, and we take $\mathbf{\Theta}_{\mathbf{M}}$ as in the previous

lemma, equation (51). After forming $\boldsymbol{\Theta}'$, we take $\mathbf{T} = \boldsymbol{\Theta}'^{-1}_{11}$. We can now use (57) to obtain

$$\mathbf{T}\,[\,\boldsymbol{\Theta}'_{11} \quad \boldsymbol{\Theta}'_{12}\,] = [\,\mathbf{I} \quad \boldsymbol{\Theta}'^{-1}_{11}\boldsymbol{\Theta}'_{12}\,] = \begin{array}{c} \\ m-d \\ n_1-m+d \end{array} \overset{\begin{array}{cc} m-d & n_1- \\ + & m+d \\ + & \end{array} \quad \begin{array}{cc} d & n_2-d \\ - & - \end{array}}{\left[\begin{array}{cc|cc} \mathbf{I} & \mathbf{0} & \mathbf{0} & * \\ \mathbf{0} & \mathbf{I} & * & * \end{array}\right]} \tag{62}$$

which has the desired structure and guarantees $\mathbf{M}'_{\boldsymbol{\Theta}} = \mathbf{0}$. ■

For the proof of Theorem 1, it remains to insert the QR decomposition $\mathbf{Q}[\mathbf{R_A}\ \mathbf{R_B}] = [\mathbf{A}\ \mathbf{B}]$.

Appendix C
Proof of the Updating Algorithm for SURV

We prove that the proposed SURV algorithm for updating (24) automatically satisfies the additional structural requirements represented by (25), viz.

$$\mathbf{Q}^H \overset{n_1\ n_2}{m\,[\mathbf{N}|\mathbf{X}]}\,\boldsymbol{\Theta} = \overset{m-d \quad \overset{n_1-}{m+d} \quad d \quad n_2-d}{m\,[\,\mathbf{R_A} \quad \boxed{\mathbf{0}}\,|\mathbf{R_B} \quad \boxed{\mathbf{0}}\,]}, \tag{63}$$

$$\mathbf{T}_{n_1}\,\overset{n_1 n_2}{[\mathbf{I}|\mathbf{0}]}\,\boldsymbol{\Theta} = \begin{array}{c} \\ m-d \\ n_1-m+d \end{array} \overset{\begin{array}{cc} m-d & \overset{n_1-}{m+d} \end{array} \quad \begin{array}{cc} d & n_2-d \end{array}}{\left[\begin{array}{cc|cc} \mathbf{I} & \boxed{\mathbf{0}} & \boxed{\mathbf{0}} & * \\ \mathbf{0} & \mathbf{I} & * & * \end{array}\right]}. \tag{64}$$

In these equations, it is important to show that the required "zero" structure (represented by $\boxed{\mathbf{0}}$) is satisfied, in particular for (64), as SURV was derived to satisfy (63). We will first write down the equations valid for updating an existing decomposition by a single new vector. We will then show that only one rotation per update, the HCR hyperbolic rotation, can possibly destroy the zero structure in (64), but that the SURV algorithm maintains the structure without additional operations (this is not valid for general algorithms that compute (63), e.g., in [11] an additional hyperbolic rotation was needed). Finally, we will also show that additional storage of matrices in (25) is not needed.

The proof is technical; unfortunately we have to study four cases separately, depending on whether $j_{\mathbf{c}} = \pm 1$ and $j'_{\mathbf{c}} = \pm 1$.

A. $j_{\mathbf{c}} = +1$ and $j'_{\mathbf{c}} = +1$

Assume that we update an existing decomposition by a new vector $\mathbf{n}$ with signature $+1$. For the present case where the output signature is also $+1$, the rank is constant ($d' = d$), and the required decomposition after the update is

$$\mathbf{Q}'^H \overset{n_1 \quad 1 \quad n_2}{m\,[\mathbf{N} \quad \mathbf{n}|\mathbf{X}]}\,\boldsymbol{\Theta}' = \overset{m-d \quad 1 \quad \overset{n_1-}{m+d} \quad d \quad n_2-d}{m\,[\,\mathbf{R}'_{\mathbf{A}} \quad \boxed{\mathbf{0}} \quad \boxed{\mathbf{0}}\,|\mathbf{R}'_{\mathbf{B}} \quad \boxed{\mathbf{0}}\,]}, \tag{65}$$

$$\mathbf{T}'\,\overset{n_1}{1}\left[\begin{array}{c|c} \overset{n_1 \quad 1}{\mathbf{I}} & \overset{n_2}{\mathbf{0}} \\ \hline 1 & \mathbf{0} \end{array}\right]\boldsymbol{\Theta}' = \begin{array}{c} \\ m-d \\ 1 \\ n_1-m+d \end{array} \overset{\begin{array}{ccc} m-d & 1 & \overset{n_1-}{m+d} \end{array} \quad \begin{array}{cc} d & n_2-d \end{array}}{\left[\begin{array}{ccc|cc} \mathbf{I} & \boxed{\mathbf{0}} & \boxed{\mathbf{0}} & \boxed{\mathbf{0}} & * \\ & 1 & & * & * \\ & & \mathbf{I} & * & * \end{array}\right]}. \tag{66}$$

Assuming the initial decomposition (63), (64), and dropping the rows and columns that do not play a role because they will not change, we obtain the following more compact *update* equations:

$$\mathbf{Q}_c^H \overset{m-d \quad 1 \quad d}{m\,[\,\mathbf{R_A} \quad \mathbf{c} \quad \mathbf{R_B}\,]}\,\boldsymbol{\Theta}_c = \overset{m-d \quad \overset{1}{\boxed{\mathbf{0}}} \quad \overset{d}{}}{m\,[\,\mathbf{R}'_{\mathbf{A}} \quad \boxed{\mathbf{0}}\,|\mathbf{R}'_{\mathbf{B}}\,]}, \tag{67}$$

$$\mathbf{T}_c\,\overset{m-d}{1}\left[\begin{array}{c|c} \overset{m-d \quad 1}{\mathbf{I}} & \overset{d}{\mathbf{0}} \\ \hline 1 & \mathbf{0} \end{array}\right]\boldsymbol{\Theta}_c = \overset{m-d}{1}\left[\begin{array}{cc|c} \mathbf{I} & \boxed{\mathbf{0}} & \boxed{\mathbf{0}} \\ \hline & 1 & * \end{array}\right]. \tag{68}$$

Here, $\mathbf{c} = \mathbf{Q}^H\mathbf{n}$, and $\mathbf{Q}_c$, $\mathbf{T}_c$, and $\boldsymbol{\Theta}_c$ are the updates of $\mathbf{Q}$, $\mathbf{T}$ and $\boldsymbol{\Theta}$, respectively (using proper extensions to make dimensions fit).

The SURV algorithm acts on (67) and nulls $\mathbf{c}$ using the rotations described in Section IV. A difference with the algorithm description is that, now, the columns of the matrices $[\mathbf{R_A}\ \mathbf{c}|\mathbf{R_B}]$ have been written sorted according to signature, whereas in Section IV this was written $[\mathbf{R_A}\ \mathbf{R_B}\ \mathbf{c}]$ for convenience in the algorithm description there. Also the ordering of the resulting columns is a bit different. Nevertheless, there is an obvious one-to-one match of the rotation operations in both cases—the algorithm operations are the same.

We follow the algorithm operations step by step and investigate the additional updates required to satisfy (68) as well. To facilitate the discussion, (68) is written as $\mathbf{T}_c\mathbf{E}\boldsymbol{\Theta}_c = \mathbf{E}'$, with $\mathbf{E}$ defined as

$$\mathbf{E} := [\,\mathbf{E}_+|\mathbf{E}_-\,] = \left[\begin{array}{cc|c} \mathbf{E}_{11} & \mathbf{E}_{12} & \mathbf{E}_{13} \\ \mathbf{E}_{21} & \mathbf{E}_{22} & \mathbf{E}_{23} \end{array}\right] = \begin{array}{c} \\ m-d \\ 1 \end{array} \overset{\begin{array}{cc} m-d & 1 \end{array} \quad d}{\left[\begin{array}{cc|c} \mathbf{I} & & \mathbf{0} \\ & 1 & \mathbf{0} \end{array}\right]}.$$

(We will use this subpartitioning of $\mathbf{E}$ to identify its various subblocks throughout the algorithm.)

(*a*) GCR: The SURV operations belonging to $\boldsymbol{\Theta}_c$ for this step consist of column rotations acting on $[\mathbf{R_A}\ \mathbf{c}]$, and on $\mathbf{E}_+$. They destroy the identity matrices in $\mathbf{E}_+$, but these are easily recovered by matching rotations in $\mathbf{T}_c$. Since the entries in the $\mathbf{E}_-$ blocks are zero, these do not change. Thus, after this update, we still have $\mathbf{E} = [\mathbf{I}\ \mathbf{0}]$.

(*b*) GRCR: The column rotations in the GRCR operations only act on $\mathbf{E}_-$. Since $\mathbf{E}_- = \mathbf{0}$, it is not changed.

(*c*) HCR: The HCR operation is a hyperbolic rotation $\boldsymbol{\theta}$ acting on $\mathbf{c}$ and a column of $\mathbf{R_B}$, and on corresponding columns of $\mathbf{E}_+$ and $\mathbf{E}_-$. This creates a fill-in in subblock $\mathbf{E}_{23}$, but that matches the required structure as seen from (68). Entry $\mathbf{E}_{22}$ is now unequal to 1 but it is easily scaled back to 1 by a row scaling operation in $\mathbf{T}_c$. At this point, we have achieved the required decomposition (67), (68).

Thus, we have seen that the required structure (68) is obtained by certain operations $\mathbf{T}_c$; no additional operations $\boldsymbol{\Theta}_c$ are

needed beyond those of the SURV algorithm. Since the resulting decomposition $\mathbf{E}'$ is known (the last row with arbitrary entries $*$ is not needed in future and can be dropped), we do not have to store $\mathbf{E}$ nor $\mathbf{T}_c$, and the operations do not actually have to be performed as they do not influence (67). As mentioned, this is a special feature of SURV and not valid for other more general updating algorithms to achieve (67).

B. $j_{\mathbf{c}} = +1$ and $j'_{\mathbf{c}} = -1$

In this case, the update of $\mathbf{N}$ decreases the rank $(d' = d-1)$. Consequently, the size of $\mathbf{R_A}$ will grow with one column, and $\mathbf{R_B}$ will drop by one column. For brevity, we skip some details and directly write down the update equations:

$$\mathbf{Q}_c^H \, {}_m[\, \overset{m-d}{\mathbf{R_A}} \quad \overset{1}{\mathbf{c}} | \overset{d}{\mathbf{R_B}} \,]\, \mathbf{\Theta}_c = {}_m[\, \overset{m-d}{\mathbf{R''_A}} \quad \overset{1}{\mathbf{a}} | \overset{d-1}{\mathbf{R'_B}} \quad \overset{1}{\boxed{\mathbf{0}}} \,]; \tag{69}$$

$$\mathbf{R}'_{\mathbf{A}} = [\mathbf{R}''_{\mathbf{A}} \mathbf{a}],$$

$$\mathbf{T}_c \, {}_{1}^{m-d}\left[\begin{array}{cc|cc} \overset{m-d}{\mathbf{I}} & & \overset{1}{} & \overset{d}{} \\ & 1 & & \\ \hline & & \mathbf{0} & \mathbf{0} \end{array}\right] \mathbf{\Theta}_c = {}_{1}^{m-d}\left[\begin{array}{cc|cc} \mathbf{I} & & \boxed{\mathbf{0}} & \overset{d-1}{} \\ & 1 & & \\ \hline & & \boxed{\mathbf{0}} & * \end{array}\right]. \tag{70}$$

- $(a), (b)$ As before, these SURV operations consist of rotations only acting on $\mathbf{E}_+$ and $\mathbf{E}_-$, respectively, and are easily matched with corresponding operations in $\mathbf{T}$ to restore the structure of $\mathbf{E}$.
- (c) The HCR operation acting on $\mathbf{c}$ and the last column of $\mathbf{R_B}$ creates a fill-in in the last column of $\mathbf{E}$, but that matches the required structure as seen from (70). Entry $\mathbf{E}_{22}$ is now unequal to 1 but it is easily scaled back to 1 without affecting the structure. Thus, we have achieved the required decomposition (69), (70) without additional operations in $\mathbf{\Theta}_c$, no additional operations are needed beyond those of the SURV algorithm.

C. $j_{\mathbf{c}} = -1$ and $j'_{\mathbf{c}} = -1$

In this case, we update $\mathbf{X}$ by $\mathbf{x}$, but the rank remains constant $(d' = d)$. The update equations are (with $\mathbf{c} = \mathbf{Q}^H\mathbf{x}$):

$$\mathbf{Q}_c^H \, {}_m[\, \overset{m-d}{\mathbf{R_A}} | \overset{d}{\mathbf{R_B}} \quad \overset{1}{\mathbf{c}} \,]\, \mathbf{\Theta}_c = {}_m[\, \overset{m-d}{\mathbf{R'_A}} | \overset{d}{\mathbf{R'_B}} \quad \overset{1}{\boxed{\mathbf{0}}} \,], \tag{71}$$

$$\mathbf{T}_c \, {}_{m-d}^{m-d\;d} [\, \mathbf{I} | \mathbf{0} \quad \overset{1}{\mathbf{0}} \,] \quad \mathbf{\Theta}_c = {}_{m-d}^{m-d}[\, \mathbf{I} | \overset{d}{\boxed{\mathbf{0}}} \quad \overset{1}{*} \,]. \tag{72}$$

As before, the SURV operations that act only on the "+" part or on the "−" part of $\mathbf{E}$ can be easily inverted by corresponding rotations in $\mathbf{T}_c$ to maintain the same structure of $\mathbf{E}$. The HCR operation acting on $\mathbf{c}$ and the last column of $\mathbf{R_A}$ creates a fill-in in the last column of $\mathbf{E}$, but that matches the required structure. Subsequently, the last row of $\mathbf{E}$ is scaled by $\mathbf{T}_c$ and the required $\mathbf{E}'$ is obtained. No additional operations in $\mathbf{\Theta}_c$ are needed beyond those of the SURV algorithm.

D. $j_{\mathbf{c}} = -1$ and $j'_{\mathbf{c}} = +1$

In this case, we update $\mathbf{X}$ and the rank increases $(d' = d+1)$. This implies that $\mathbf{R_A}$ shrinks and $\mathbf{R_B}$ grows by one column. The update equations are

$$\mathbf{Q}_c^H \, {}_m[\, \overset{m-d}{\mathbf{R_A}} | \mathbf{R_B} \quad \overset{d}{} \quad \overset{1}{\mathbf{c}} \,]\, \mathbf{\Theta}_c = {}_m[\, \overset{m-d'}{\mathbf{R'_A}} \quad \overset{1}{\boxed{\mathbf{0}}} \quad \overset{d}{\mathbf{R''_B}} \quad \overset{1}{\mathbf{b}} \,];$$
$$\mathbf{R}'_{\mathbf{B}} = [\, \mathbf{R}''_{\mathbf{B}} \quad \mathbf{b} \,], \tag{73}$$

$$\mathbf{T}_c \, {}_{m-d}^{m-d\;d} [\, \mathbf{I} | \mathbf{0} \quad \overset{1}{\mathbf{0}} \,]\, \mathbf{\Theta}_c = {}_{1}^{m-d'}\left[\begin{array}{cc|cc} \overset{m-d'}{\mathbf{I}} & \overset{1}{\boxed{\mathbf{0}}} & \overset{d}{} & \overset{1}{} \\ & & \boxed{\mathbf{0}} & \boxed{\mathbf{0}} \\ \hline 1 & & * & * \end{array}\right]. \tag{74}$$

The first steps are as before and the structure of $\mathbf{E}$ is maintained. The HCR operation acting on $\mathbf{c}$ and the last column of $\mathbf{R_A}$ zeroes that last column, and creates a fill-in in the last column of $\mathbf{E}$ (corresponding with $\mathbf{c}$). However, that fill-in occurs in the last row, which will be dropped as $m - d'$ is one less than $m - d$: it matches the structure in (74). No additional operations in $\mathbf{\Theta}_c$ are needed beyond those of the SURV algorithm.

REFERENCES

[1] A.-J. van der Veen, E. F. Deprettere, and A. L. Swindlehurst, "Subspace based signal analysis using singular value decomposition," *Proc. IEEE*, vol. 81, no. 9, pp. 1277–1308, Sep. 1993.

[2] A.-J. van der Veen, "Algebraic methods for deterministic blind beamforming," *Proc. IEEE*, vol. 86, no. 10, pp. 1987–2008, Oct. 1998.

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

[4] T. Chan, "Rank revealing QR factorizations," *Linear Algebra Appl.*, vol. 88–89, pp. 67–82, 1987.

[5] C. Bischof and G. Schroff, "On updating signal subspaces," *IEEE Trans. Signal Process.*, vol. 40, no. 1, pp. 96–105, Jan. 1992.

[6] G. W. Stewart, "An updating algorithm for subspace tracking," *IEEE Trans. Signal Process.*, vol. 40, no. 6, pp. 1535–1541, 1992.

[7] G. W. Stewart, "Updating a rank-revealing ULV decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 14, no. 2, pp. 494–499, 1993.

[8] A.-J. van der Veen, "A Schur method for low-rank matrix approximation," *SIAM J. Matrix Anal. Appl.*, vol. 17, no. 1, pp. 139–160, 1996.

[9] A.-J. Boonstra and A.-J. van der Veen, "Gain calibration methods for radio telescope arrays," *IEEE Trans. Signal Process.*, vol. 51, no. 1, pp. 25–38, Jan. 2003.

[10] R. D. Fierro, P. C. Hansen, and P. S. K. Hansen, "UTV Tools: Matlab templates for rank-revealing UTV decompositions," *Numer. Algorithms*, vol. 20, no. 2–3, pp. 165–194, Aug. 1999.

[11] A.-J. van der Veen, "Subspace tracking using a constrained hyperbolic URV decomposition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, May 1998, vol. 4, pp. 1953–1956.

[12] X. Doukopoulos and G. Moustakides, "Fast and stable subspace tracking," *IEEE Trans. Signal Process.*, vol. 56, no. 4, pp. 1452–1465, Apr. 2008.

[13] I. Karasalo, "Estimating the covariance matrix by signal subspace averaging," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 34, no. 1, pp. 8–12, Feb. 1986.

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

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

[16] K. Abed-Meraim, A. Chkeif, and Y. Hua, "Fast orthonormal PAST algorithm," *IEEE Signal Process. Lett.*, vol. 7, no. 3, pp. 60–62, Mar. 2000.

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

[18] B. Yang, "An extension of the PASTd algorithm to both rank and subspace tracking," *IEEE Signal Process. Lett.*, vol. 2, no. 9, pp. 179–182, Sep. 1995.

[19] R. Badeau, G. Richard, and B. David, "Fast and stable YAST algorithm for principal and minor subspace tracking," *IEEE Trans. Signal Process.*, vol. 56, no. 8, pp. 3437–3446, Aug. 2008.
[20] G. W. Stewart, "On the stability of sequential updates and downdates," *IEEE Trans. Signal Process.*, vol. 43, no. 11, pp. 2642–2648, Nov. 1995.
[21] Y.-J. J. Wu, "Downdating a rank-revealing URV decomposition," Univ. of Maryland at College Park, College Park, MD, Tech. Rep., 1995.
[22] H. Park and L. Eldén, "Downdating the rank-revealing URV decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 16, no. 1, pp. 138–155, 1995.
[23] M. Stewart and G. W. Stewart, "On hyperbolic triangularization: Stability and pivoting," *SIAM J. Matrix Anal. Appl.*, vol. 19, no. 4, pp. 847–860, 1998.
[24] R. Badeau, K. Abed-Meraim, G. Richard, and B. David, "Sliding window orthonormal PAST algorithm," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Apr. 2003, vol. 5, pp. 261–264.
[25] R. Badeau, G. Richard, and B. David, "Sliding window adaptive SVD algorithms," *IEEE Trans. Signal Process.*, vol. 52, no. 1, pp. 1–10, Jan. 2004.
[26] R. Badeau, B. David, and G. Richard, "Fast approximated power iteration subspace tracking," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 2931–2941, Aug. 2005.
[27] J. Götze and A.-J. van der Veen, "On-line subspace estimation using a Schur-type method," *IEEE Trans. Signal Process.*, vol. 44, no. 6, pp. 1585–1589, Jun. 1996.
[28] J. Götze, M. Haardt, and J. A. Nossek, "Subspace estimation using unitary Schur-type methods," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 1995, pp. 1153–1156.
[29] A. Bunse-Gerstner, "An analysis of the HR algorithm for computing the eigenvalues of a matrix," *Linear Algebra Appl.*, vol. 35, pp. 155–173, 1981.
[30] R. Onn, A. Steinhardt, and A. Bojanczyk, "The hyperbolic singular value decomposition and applications," *IEEE Trans. Signal Process.*, vol. 39, no. 7, pp. 1575–1588, Jul. 1991.
[31] R. Horn and C. Johnson, *Matrix Analysis*. Cambridge, MA: Cambridge Univ. Press, 1990.
[32] A. Bojanczyk, S. Qiao, and A. O. Steinhardt, "Unifying unitary and hyperbolic transformations," *Linear Algebra Appl.*, vol. 316, pp. 183–197, 2000.
[33] *The Algebraic Eigenvalue Problem*, J. H. Wilkinson, Ed. New York: Oxford Univ. Press, 1988.
[34] A. Edelman, "The distribution and moments of the smallest eigenvalue of a random matrix of Wishart type," *Linear Algebra Appl.*, vol. 159, pp. 55–80, Dec. 1991.
[35] C. A. Tracy and H. Widom, "On orthogonal and symplectic matrix ensembles," *Commun. Math. Phys.*, vol. 177, pp. 727–754, 1996.
[36] M. Johnstone, "On the distribution of the largest eigenvalue in principal components analysis," *Ann. Stat.*, vol. 29, pp. 295–327, 2001.

Mu Zhou (S'11) was born in China in 1982. He received the B.Sc. and M.Sc. degrees from the School of Information Science and Engineering, Southeast University, Nanjing, China, in 2005 and 2008, respectively, all in electrical engineering.

Since 2008, he has been working towards the Ph.D. degree in the Circuits and Systems Group in the Department of Electrical Engineering, Mathematics and Computer Science, TU Delft, The Netherlands. His current research interests include matrix decomposition for array signal processing, receiver design for nano-satellite communication and mixed signal system modeling.

Alle-Jan van der Veen (F'05) was born in The Netherlands in 1966. He received the Ph.D. degree (*cum laude*) from TU Delft, The Netherlands, in 1993.

Throughout 1994, he was a postdoctoral scholar at Stanford University, Stanford, CA. At present, he is a Full Professor in signal processing at TU Delft. His research interests are in the general area of system theory applied to signal processing, and in particular algebraic methods for array signal processing, with applications to wireless communications and radio astronomy.

Dr. van der Veen is the recipient of the 1994 and the 1997 IEEE Signal Processing Society (SPS) Young Author Paper Award. He was Chairman of the IEEE SPS Signal Processing for Communications Technical Committee from 2002 to 2004, the Editor-in-Chief of the IEEE SIGNAL PROCESSING LETTERS from 2002 to 2005, and the Editor-in-Chief of the IEEE TRANSACTIONS ON SIGNAL PROCESSING from 2006 to 2008. He was Technical Cochair of the International Conference on Acoustics, Speech and Signal Processing (ICASSP), Prague, 2011, and currently chairs the IEEE SPS Fellow Reference Committee.