STABLE SUBSPACE TRACKING ALGORITHM BASED ON SIGNED URV DECOMPOSITION

Mu Zhou and Alle-Jan van der Veen

Delft University of Technology, Dept. of Elec. Eng., Mekelweg 4, 2628 CD Delft, The Netherlands.

The class of Schur subspace estimators provides a parametrization of all minimal-rank matrix approximants that lie within a specified distance of a given matrix, and in particular gives expressions for the column spans of these approximants. Unlike previous numerically unstable algorithms, this paper presents a signed URV decomposition (SURV) that efficiently and stably computes the Schur subspace estimator. Given a threshold on the singular values of the data matrix, SURV tracks the orthonormal basis of the principal/minor subspace and the rank of the subspace at the same time exactly with respect to the threshold at a computational complexity of $O(m^2)$ per vector update or downdate. SURV is not an iterative method.

***Index Terms*—** Subspace tracking, generalized Schur algorithm, signed URV, signed Cholesky factorization.

1 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.

The literature related to the topic of subspace tracking is extremely rich, such as [1]. A URV decomposition [2] supports efficient tracking of the principal/minor subspace and the rank of the subspace. URV provides good estimates within few data samples at a computational complexity of $O(m^2)$ per vector update or downdate. The drawbacks with URV are that its updating steps are either imprecise or complex since they are based on an iteration to find the smallest singular value, and its downdating steps are not stable [3] without the help of exponential windows.

An algorithm comparable with URV is the Schur subspace estimation (SSE) technique [4], [5], [6], [7], [8] based on the knowledge of an upper bound γ on the noise power. Given a $m \times n_2$ data matrix $\mathbf{X}$, measured column-by-column, that satisfies the model $\mathbf{X} = \tilde{\mathbf{X}} + \tilde{\mathbf{N}}$, where $\tilde{\mathbf{X}}$ is a low rank matrix and $\tilde{\mathbf{N}}$ is a disturbance, this technique gives a parametrization of the class of all $\hat{\mathbf{X}}$ that satisfy

$$\|\mathbf{X} - \hat{\mathbf{X}}\| \le \gamma. \tag{1}$$

where $\|\cdot\|$ denotes the matrix 2-norm and $\hat{\mathbf{X}}$ has a minimal rank. In fact, this minimal rank is d, where d is the number of singular values of $\mathbf{X}$ that are larger than γ. The truncated SVD (TSVD) is within the class, but it is not explicitly identified. The computation is based on an implicit signed Cholesky factorization

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

where $\mathbf{A}, \mathbf{B}$ have minimal dimensions $m \times (m-d)$ and $m \times d$, respectively, and are not unique (H denotes the Hermitian transpose). If we are interested only in the column span of $\hat{\mathbf{X}}$, i.e., the rank d principal subspace of $\mathbf{X}$ in the sense of (1), then the subspace is given by $\mathrm{ran}\,\{\mathbf{B} - \mathbf{A}\mathbf{M}\}$ for any pair $(\mathbf{A}, \mathbf{B})$ and any matrix $\mathbf{M}$ such that $\|\mathbf{M}\| \le 1$. $\mathbf{A}$ and $\mathbf{B}$ follow from the factorization

$$\begin{bmatrix}\mathbf{N} & \mathbf{X}\end{bmatrix} \mathbf{\Theta} = \begin{bmatrix}\mathbf{A} & \mathbf{0} & | & \mathbf{B} & \mathbf{0}\end{bmatrix}, \tag{3}$$

where $\mathbf{N}$ is any matrix such that $\mathbf{N}\mathbf{N}^H = \gamma^2 \mathbf{I}$, and $\mathbf{\Theta}$ is a $\mathbf{J}$-unitary matrix [4]. Straightforward generalizations of above subspace estimates are possible. Suppose that we know $\mathbf{R}_{\tilde{\mathbf{N}}} \le \gamma^2 \mathbf{R}_{\mathbf{N}}$ instead of $\|\tilde{\mathbf{N}}\| \le \gamma$, where $\mathbf{R}_{\mathbf{N}}$ could be an estimate of the noise covariance matrix $\mathbf{R}_{\tilde{\mathbf{N}}}$, an implicit signed Cholesky factorization of $\mathbf{X}\mathbf{X}^H - \gamma^2 \mathbf{R}_{\mathbf{N}}$ can provide a minimal-rank approximant $\hat{\mathbf{X}}$ such that

$$\|\mathbf{R}_{\mathbf{N}}^{-1/2} (\mathbf{X} - \hat{\mathbf{X}})\| \le \gamma. \tag{4}$$

In [4], a specific "unbiased" subspace estimator within the class of (1) is given as

$$\begin{aligned} &\mathrm{SSE}-2: \mathbf{U}_{\mathrm{SSE}-2} = \mathbf{B} - \mathbf{A}\mathbf{M}_{\mathbf{\Theta}}, \\ &\mathbf{M}_{\mathbf{\Theta}} = \left(\mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12}\right)_{11} = \begin{bmatrix}\mathbf{I}_{m-d} & \mathbf{0}\end{bmatrix} \mathbf{\Theta}_{11}^{-1}\mathbf{\Theta}_{12} \begin{bmatrix}\mathbf{I}_d \\ \mathbf{0}\end{bmatrix}. \end{aligned} \tag{5}$$

This estimator shows good performance in tracking principal/minor subspace [4]. Previously, the hyperbolic URV (HURV) algorithm [5] was derived to compute and update the decomposition (3), (5), and the resulting $\mathbf{A}$, $\mathbf{B}$ have good properties as

$$\mathrm{ran}\{\mathbf{B}\} \subset \mathrm{ran}\{\mathbf{X}\},\ \|\mathbf{B}\| \le \|\mathbf{X}\|, \tag{6}$$

$$\mathrm{ran}\{\mathbf{A}\} \subset \mathrm{ran}\{\mathbf{N}\},\ \|\mathbf{A}\| \le \|\mathbf{N}\|. \tag{7}$$

In this paper, we propose a new SSE-2 updating algorithm, called the signed URV decomposition (SURV). SURV keeps the good properties as (6), (7). In contrast to HURV, SURV is numerically stable as it uses at most one hyperbolic rotation per vector update or downdate.

2 J-UNITARY MATRICES

At this section, we review some materials on $\mathbf{J}$-unitary matrices from [4]. A square matrix $\mathbf{\Theta}$ is $\mathbf{J}$-unitary if it satisfies

$$\mathbf{\Theta}^H \mathbf{J} \mathbf{\Theta} = \mathbf{J},\ \mathbf{\Theta} \mathbf{J} \mathbf{\Theta}^H = \mathbf{J}, \tag{8}$$

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

$$\mathbf{\Theta} = \begin{array}{c c} & \begin{array}{cc} p & q \end{array} \\ \begin{array}{c} p \\ q \end{array} & \begin{bmatrix}\mathbf{\Theta}_{11} & \mathbf{\Theta}_{12} \\ \mathbf{\Theta}_{21} & \mathbf{\Theta}_{22}\end{bmatrix} \end{array},\ \mathbf{J} = \begin{bmatrix} +\mathbf{I}_p & \\ & -\mathbf{I}_q \end{bmatrix}. \tag{9}$$

If $\mathbf{\Theta}$ is applied to a block-partitioned matrix $\begin{bmatrix}\mathbf{A} & \mathbf{B}\end{bmatrix}$, then

$$\begin{bmatrix}\mathbf{A} & \mathbf{B}\end{bmatrix} \mathbf{\Theta} = \begin{bmatrix}\mathbf{C} & \mathbf{D}\end{bmatrix} \Rightarrow \mathbf{A}\mathbf{A}^H - \mathbf{B}\mathbf{B}^H = \mathbf{C}\mathbf{C}^H - \mathbf{D}\mathbf{D}^H. \tag{10}$$

[1]This work was supported in part by the project BDREAMS at Delft University of Technology in Netherlands, and China Scholarship Council from P.R.China.

Table 1. Elementary $\mathbf{j}$-unitary zeroing rotations.[1]

1. Input: $\begin{bmatrix} r & x \end{bmatrix}$, $|r|>|x|$, $\mathbf{j}$=diag$\{+1,-1\}$ or $\mathbf{j}$=diag$\{-1,+1\}$;
   Output: $\boldsymbol{\theta}$ such that $\begin{bmatrix} r & x \end{bmatrix} \boldsymbol{\theta} = \begin{bmatrix} r' & 0 \end{bmatrix}$;
   Solution: $\boldsymbol{\theta} = \begin{bmatrix} 1 & -s \\ -s^* & 1 \end{bmatrix} \frac{1}{c}$, $s = \frac{x}{r}$, $c = \sqrt{1-|s|^2}$.
2. Input: $\begin{bmatrix} r & x \end{bmatrix}$, $|r|<|x|$, $\mathbf{j}$=diag$\{+1,-1\}$ or $\mathbf{j}$=diag$\{-1,+1\}$;
   Output: $\boldsymbol{\theta}$ such that $\begin{bmatrix} r & x \end{bmatrix} \boldsymbol{\theta} = \begin{bmatrix} 0 & r' \end{bmatrix}$;
   Solution: $\boldsymbol{\theta} = \begin{bmatrix} 1 & -s^* \\ -s & 1 \end{bmatrix} \frac{1}{c}$, $s = \frac{r}{x}$, $c = \sqrt{1-|s|^2}$.
3. Input: $\begin{bmatrix} r & x \end{bmatrix}$, $|r|+|x|\neq 0$, $\mathbf{j}$=diag$\{+1,+1\}$ or $\mathbf{j}$=diag$\{-1,-1\}$;
   Output: $\boldsymbol{\theta}$ such that $\begin{bmatrix} r & x \end{bmatrix} \boldsymbol{\theta} = \begin{bmatrix} r' & 0 \end{bmatrix}$;
   Solution: $\boldsymbol{\theta} = \begin{bmatrix} c^* & -s \\ s^* & c \end{bmatrix}$, $s = x/\sqrt{r^*r+x^*x}$, $c = \sqrt{1-|s|^2}$.
   This is essentially a Givens rotation.

[1] $*$ denotes the conjugate operator.

Hence, $\mathbf{J}$ assigns a positive signature to the columns of $\mathbf{A}$, $\mathbf{C}$, and a negative signature to those of $\mathbf{B}$, $\mathbf{D}$.

A 2×2 matrix $\boldsymbol{\theta}$ is an elementary $\mathbf{j}$-unitary rotation if it satisfies $\boldsymbol{\theta}^H \mathbf{j} \boldsymbol{\theta} = \mathbf{j}$, $\boldsymbol{\theta} \mathbf{j} \boldsymbol{\theta}^H = \mathbf{j}$. In Table 1 we list the elementary $\mathbf{j}$-unitary zeroing rotations used in this paper .

3 SIGNED URV DECOMPOSITION

Let $\mathbf{N}$: $m \times n_1$ and $\mathbf{X}$: $m \times n_2$ be given matrices. $\boldsymbol{\Theta}$ is a $\mathbf{J}$-unitary matrix partitioned conformably as

$$\boldsymbol{\Theta} = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} n_1 & n_2 \end{array} \\ \begin{bmatrix} \boldsymbol{\Theta}_{11} & \boldsymbol{\Theta}_{12} \\ \boldsymbol{\Theta}_{21} & \boldsymbol{\Theta}_{22} \end{bmatrix} \end{array}, \mathbf{J} = \begin{bmatrix} +\mathbf{I}_{n_1} & \\ & -\mathbf{I}_{n_2} \end{bmatrix}. \tag{11}$$

Introduce a QR factorization of $\begin{bmatrix} \mathbf{A} & \mathbf{B} \end{bmatrix}$:

$$\mathbf{R} = \begin{bmatrix} \mathbf{R_A} & \mathbf{R_B} \end{bmatrix} = \mathbf{Q}^H \begin{bmatrix} \mathbf{A} & \mathbf{B} \end{bmatrix} = \begin{bmatrix} \overset{m-d}{\mathbf{Q_A}} & \overset{d}{\mathbf{Q_B}} \end{bmatrix}^H \begin{bmatrix} \mathbf{A} & \mathbf{B} \end{bmatrix}, \tag{12}$$

where $\mathbf{R}$ is triangular and $\mathbf{Q}$ is unitary. Hyperbolic URV decomposition computes and updates the factorization satisfying following two equations,

$$\mathbf{Q}^H \begin{bmatrix} \overset{\substack{n_1 \\ +}}{\mathbf{N}} & \overset{\substack{n_2 \\ -}}{\mathbf{X}} \end{bmatrix} \boldsymbol{\Theta} = \begin{bmatrix} \overset{\substack{m-d \\ +}}{\mathbf{R_A}} & \overset{\substack{n_1- \\ m+d \\ +}}{\mathbf{0}} & | & \overset{\substack{d \\ -}}{\mathbf{R_B}} & \overset{\substack{n_2-d \\ -}}{\mathbf{0}} \end{bmatrix}, \tag{13}$$

where in addition $\boldsymbol{\Theta}$ satisfies (because (3) is not unique)

$$\mathbf{T} \begin{bmatrix} \overset{n_1}{\boldsymbol{\Theta}_{11}} & \overset{n_2}{\boldsymbol{\Theta}_{12}} \end{bmatrix} = \begin{array}{c} \\ m-d \\ n_1-m+d \end{array} \begin{array}{c} \begin{array}{cccc} \substack{m-d \\ +} & \substack{n_1- \\ m+d \\ +} & \substack{d \\ -} & \substack{n_2-d \\ -} \end{array} \\ \left[\begin{array}{c|c|c|c} \mathbf{I} & \mathbf{0} & \mathbf{0} & * \\ \hline * & \mathbf{I} & * & * \end{array} \right] \end{array}, \tag{14}$$

where $\mathbf{T}$ is an invertible matrix, and the sign $+$ and $-$ above matrices denote the positive and negative signatures of the columns in those matrices, respectively. This condition guarantees $\mathbf{M}_{\boldsymbol{\Theta}} = \mathbf{0}$ in [5] so that $\mathbf{U}_{\mathrm{SSE}-2} = \mathbf{B}$. The proposed algorithm, SURV, only tracks (13) and is such that the structure of (14) is automatically satisfied. SURV uses at most one hyperbolic rotation per vector update or downdate which occurs between only two entries (not vectors) and hence can be numerically stable (unlike HURV which uses up to three hyperbolic rotations).

If we choose $\mathbf{R}$ to be lower triangular, then

$$\operatorname{ran}\{\mathbf{Q_B}\} = \operatorname{ran}\{\mathbf{B}\}. \tag{15}$$

Table 2. Four types of zeroing schemes.[1]

1. Givens row-column-combined rotations(GRCR) to zero c_k.
   (a) Determine $\mathbf{q}$ such that $\mathbf{q}^H \begin{bmatrix} c_k \\ c_{k+1} \end{bmatrix} = \begin{bmatrix} 0 \\ c'_{k+1} \end{bmatrix}$;
   (b) Apply $\mathbf{q}^H$ to row pair $(k, k+1)$ of $\begin{bmatrix} \mathbf{R} & \mathbf{c} \end{bmatrix}$. Apply $\mathbf{q}$ to column pair $(k, k+1)$ of $\mathbf{Q}$;
   (c) Determine $\boldsymbol{\theta}$ such that $\begin{bmatrix} r_{kk} & r_{k(k+1)} \end{bmatrix} \boldsymbol{\theta} = \begin{bmatrix} r'_{kk} & 0 \end{bmatrix}$;
   (d) Apply $\boldsymbol{\theta}$ to column pair $(k, k+1)$ of $\mathbf{R}$.
2. Givens column rotations (GCR) to zero c_k ($j_k = j_{\mathbf{c}}$).
   (a) Determine $\boldsymbol{\theta}$ such that $\begin{bmatrix} r_{kk} & c_k \end{bmatrix} \boldsymbol{\theta} = \begin{bmatrix} r'_{kk} & 0 \end{bmatrix}$;
   (b) Apply $\boldsymbol{\theta}$ to the k-th column of $\mathbf{R}$ and $\mathbf{c}$.
3. Hyperbolic column rotations (HCR) to zero c_k (similar to Type 2 but $j_k = -j_{\mathbf{c}}$).
4. Givens row rotations (GRR) to zero $r_{k(k+1)}$.
   (a) Determine $\mathbf{q}$ such that $\mathbf{q}^H \begin{bmatrix} r_{k(k+1)} \\ r_{(k+1)(k+1)} \end{bmatrix} = \begin{bmatrix} 0 \\ r'_{(k+1)(k+1)} \end{bmatrix}$;
   (b) Apply $\mathbf{q}^H$ to row pair $(k, k+1)$ of $\begin{bmatrix} \mathbf{R} & \mathbf{c} \end{bmatrix}$. Apply $\mathbf{q}$ to column pair $(k, k+1)$ of $\mathbf{Q}$.

[1] Type 1, 2 and 4 are using Case 3 in Table 1. Type 3 is using Case 1 and 2 in Table 1.

Hence, the columns of $\mathbf{Q_B}$ form an orthonormal basis of SSE-2. If our objective was to estimate a subspace basis of ran $\{\mathbf{A}\}$, then we would swap $\mathbf{A}$ and $\mathbf{B}$ or take $\mathbf{R}$ upper triangular so that ran $\{\mathbf{Q_A}\}$ = ran $\{\mathbf{A}\}$. Complete theorems to motivate SURV were proven but here is no place to show them. The theorems proposed in [5] could be a reference.

In view of (10), it is seen that a downdating scheme for $\mathbf{X}$ or $\mathbf{N}$, i.e., recomputing the decomposition after removing some columns of $\mathbf{X}$ or $\mathbf{N}$ can easily be implemented by updating $\mathbf{N}$ or $\mathbf{X}$ with these columns of opposite signatures, respectively. In fact, updating and downdating problems have no essential difference in SURV, and hence we will refer to "updating" in general.

4 UPDATING SURV

In this section, we investigate how this factorization can be updated when new columns for $\mathbf{X}$ and $\mathbf{N}$ become available. Here, $\mathbf{R}$ is chosen to be lower triangular. For a new coming vector $\mathbf{c}_{\mathrm{new}}$, we define $\mathbf{c} = \mathbf{Q}^H \mathbf{c}_{\mathrm{new}}$. Denote the signature of $\mathbf{c}_{\mathrm{new}}$ by $j_{\mathbf{c}}$, where $j_{\mathbf{c}} = +1$ if we are extending $\mathbf{N}$ by $\mathbf{c}_{\mathrm{new}}$ else $j_{\mathbf{c}} = -1$ if we are extending $\mathbf{X}$ by $\mathbf{c}_{\mathrm{new}}$. Denote the signature of $\mathbf{R}$ by diag$\{j_1, j_2, \ldots, j_m\}$. Let c_k denote the k-th entry of $\mathbf{c}$ and r_{kl} denote the (k,l) entry of $\mathbf{R}$.

Because of potential numerical instability associated with hyperbolic rotations, the updating steps are designed to use at most only one hyperbolic rotation per vector update. Due to the limitation of pages, the detailed analysis of the updating steps is not shown here. The conclusion of this analysis is that the updating algorithm only consists of the steps for reducing $\begin{bmatrix} \mathbf{R} & \mathbf{c} \end{bmatrix}$ to $\begin{bmatrix} \mathbf{R}' & \mathbf{0} \end{bmatrix}$ (although $\mathbf{R}'$ is also used for an intermediate quantity when $\mathbf{c}'$ is not yet $\mathbf{0}$). $\mathbf{Q}$ and $\mathbf{R}$ are stored and tracked, but $\boldsymbol{\Theta}$ and $\mathbf{T}$ do not need to be stored and tracked because (14) is automatically satisfied. To list the updating algorithm, we define four types of zeroing schemes as listed in Table 2.

The actual steps of the updating algorithm are determined by d. The summarized updating algorithm is listed in Table 3, which contains two phases, one to zero $\mathbf{c}$ and the other one to recover the lower triangular structure of $\mathbf{R}$.

After the zeroing steps described in the upper part of Table 3, the signature of the resulting matrix $\begin{bmatrix} \mathbf{R}' & \mathbf{c}' \end{bmatrix}$ may be unsorted. If we store $\mathbf{c}'$ into the last column of $\mathbf{R}'$ and let $j'_m = j'_{\mathbf{c}}$ in the case

Table 3. The updating algorithm.[1]

$j_{\mathbf{c}} = +1$ case	$j_{\mathbf{c}} = -1$ case
The zeroing steps:	
for $k = 1$:1:$(m{-}d)$	for $k = 1$:1:$(m{-}d{-}1)$
zero c_k using GCR;	zero c_k using GRCR;
end	end
	if $d < m$
for $k{=}(m{-}d{+}1)$:1:$(m{-}1)$	for $k{=}(m{-}d)$:1:$(m{-}1)$
zero c_k using GRCR;	swap col$(k, k{+}1)$ of $\mathbf{R}$;
end	zero $r_{k(k+1)}$ using GRR;
	end
if $d > 0$	for $k = (m{-}d)$:1:$(m{-}1)$
zero c_m using HCR.	zero c_k using GCR;
end	end
	zero c_m using HCR.
	else
	for $k = 1$:1:m
	zero c_k using GCR;
	end
	end
$\star$ Store $\mathbf{c}'$ into the last col of $\mathbf{R}'$ and let $j'_m{=}j'_{\mathbf{c}}$ in the case $\lvert r_{mm}\rvert{<}\lvert c_m\rvert, j_m = -j_{\mathbf{c}}$. Then calculate $d' = d{-}(j'_m{+}j_{\mathbf{c}})/2$.	
The sorting steps (if $j'_m = +1$):	
for $k = m$:−1:$(m{-}d{+}2)$	for $k = m$:−1:$(m{-}d{+}1)$
swap col$(k-1, k)$ of $\mathbf{R}'$;	swap col$(k-1, k)$ of $\mathbf{R}'$;
zero $r_{(k-1)k}$ using GRR;	zero $r_{(k-1)k}$ using GRR;
end	end

[1] "col" denotes "column". The matlab syntax is used for expressing the loop index k. Steps violating this syntax do not exist.

$|r_{mm}| < |c_m|, j_m = -j_{\mathbf{c}}$ (at this time $\mathbf{c}'$ is nonzero and the last column of $\mathbf{R}'$ is zero), then only the columns of $\mathbf{R}'$ are involved in the sorting steps. The number of columns of $\mathbf{R}'_{\mathbf{B}}$, d', i.e., the rank of this principal subspace after updating, which is related to d, $j_{\mathbf{c}}$ and j'_m at this moment, is defined as $d' = d - (j'_m + j_{\mathbf{c}})/2$. Now if the last column of $\mathbf{R}'$ has a positive signature, i.e., $j'_m = +1$, sorting is possibly needed. The lower part of Table 3 shows the sorting steps.

The computational complexity of the proposed updating algorithm is of $O(m^2)$ per vector update. SURV runs at least three times faster than URV. For the proposed updating algorithm, the highlight is that only one hyperbolic rotation is needed by entries at the bottom right corner of $\begin{bmatrix}\mathbf{R} & \mathbf{c}\end{bmatrix}$. This hyperbolic rotation only acts on the two nonzero entries, r_{mm} and c_m. Given the numerically stable forms for hyperbolic rotations in Case 1 and Case 2:

$$\text{Case 1}: \begin{bmatrix} r & x \end{bmatrix} \boldsymbol{\theta} = \begin{bmatrix} r\sqrt{1-|s|^2} & 0 \end{bmatrix}, \tag{16}$$

$$\text{Case 2}: \begin{bmatrix} r & x \end{bmatrix} \boldsymbol{\theta} = \begin{bmatrix} 0 & x\sqrt{1-|s|^2} \end{bmatrix}, \tag{17}$$

it is possible to compute $\begin{bmatrix} r_{mm} & c_m \end{bmatrix} \boldsymbol{\theta}$ directly avoiding the potential singularity problem arising from the intermediate computation of $\boldsymbol{\theta}$. Since Case 3 is actually a Givens rotation, which is known to be stable, our proposed algorithm is clearly stable. This algorithm efficiently computes SSE-2 without the inversion computation in the definition of SSE-2, which may introduce another issue of numerical stability.

We should regard the extreme case $|r_{mm}| = |c_m|, j_m = -j_{\mathbf{c}}$ as a type of critical state, which corresponds to the case that a singular value of the data matrix equals to γ. Our strategy is to slightly perturb c_m and then this case is dealt as a $|r_{mm}| > |c_m|$ case if $j_{\mathbf{c}} = +1$ or a $|r_{mm}| < |c_m|$ case if $j_{\mathbf{c}} = -1$. After that processing, the actual result is $\begin{bmatrix} 0 & 0 \end{bmatrix}$. The updating of SURV is still continuous, while HURV encounters a break down in this case.

The initialization is set as

$$\begin{gathered} n_1 = m,\ n_2 = 0,\ d = 0, \\ \mathbf{R} = \mathbf{R_A} = \mathbf{N} = \gamma \mathbf{I}_m, \mathbf{R_B} = \emptyset,\ \mathbf{X} = \emptyset,\ \mathbf{Q} = \mathbf{I}_m, \end{gathered} \tag{18}$$

where $\mathbf{R}$ can also be assigned a suitable other lower triangular matrix.

5 SIMULATION RESULTS

In this section, we demonstrate the numerical stability of SURV, and compare its performance with HURV and URV. The data model is defined as

$$\mathbf{x}(t) = \mathbf{A}(t)\mathbf{s}(t) + \mathbf{n}(t), \tag{19}$$

where $m \geq d$, $\mathbf{A}(t)$: $m \times d$ has d singular values all 1, $\mathbf{s}(t) : d \times 1$ is formed by i.i.d. Gaussian random variables with zero mean and standard deviation $\sigma_s = 1$, $\mathbf{n}(t)$: $m \times 1$ is formed by i.i.d. Gaussian noise with zero mean and standard deviation σ_n, and $\mathbf{x}(t)$: $m \times 1$. The received data matrix is formed as $\mathbf{X} = [\mathbf{x}(1), \cdots, \mathbf{x}(k)]_{m \times k}$.

A n-vector ($n{<}k$) wide sliding window is used to slide on $\mathbf{X}$ to compute the subspace of the received data matrix. The $\mathbf{x}(t)$ inside the window form the data matrix $\mathbf{W}$. Every $\mathbf{W}$ is processed by SVD to generate the rank estimate $\hat{d}_{\text{svd}}$ corresponding to the threshold, and the principal subspace $\mathbf{U}_{\text{svd}}$ corresponding to the known number of signals d. Rank errors occur when estimates $\hat{d} \neq \hat{d}_{\text{svd}}$ or $\hat{d} \neq d$ depending on the criterion. The rank error rate (RER) is defined by

$$\text{RER} = \text{Number of rank errors/Total number of tests}. \tag{20}$$

The error of the principal subspace estimate $\hat{\mathbf{U}}$ is defined by

$$\text{If } \hat{d} = d,\ e_{\hat{\mathbf{U}}} = \left\| \hat{\mathbf{U}}\mathbf{Z} - \mathbf{U}_{\text{svd}} \right\|;\ \text{else } e_{\hat{\mathbf{U}}} = 1, \tag{21}$$

where $\mathbf{Z} = \hat{\mathbf{U}}^H \mathbf{U}_{\text{svd}}$. The factorization error is defined as

$$e_{\text{f}} = \| \mathbf{TJT}^H - \mathbf{QRJR}^H \mathbf{Q}^H \|, \tag{22}$$

where $\mathbf{T} = \begin{bmatrix} \gamma \mathbf{I} & \mathbf{W} \end{bmatrix}$ for SURV and HURV, and $\mathbf{T} = \mathbf{W}, \mathbf{J} = \mathbf{I}$ for URV. The signal to noise ratio (SNR) is defined as $\text{SNR} = 10\log(\sigma_s^2/\sigma_n^2)$. The same threshold $\gamma = 1.24\alpha$ is given to all algorithms, where $\alpha = \sigma_n(1 + \sqrt{m/n})\sqrt{n}$ [9] and γ is a reasonable threshold on the largest singular value due to noise (resulting in no false alarm rate for $m = 16$ and $n = 20$). The matlab code of URV is taken from UTV-tools [10] but we slightly modified it to make it run in nonstationary cases.

Fig. 1 shows the factorization error of SURV, HURV and URV in different cases. Fig. 1(a) shows the boxplot of the factorization error over 10^5 Monte Carlo runs at given SNRs. Fig. 1(b) and Fig. 1(c) show the factorization error for tracking in a stationary case and in a nonstationary case with d switching between 1 and 3 every 150 samples, respectively. It is seen from Fig. 1(a) 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 (also tracks two entries of $\boldsymbol{\Theta}$), 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. It is seen from Fig. 1(b) and Fig. 1(c) that URV encounters break downs frequently while never a break down occurs in SURV.

Fig. 2 shows the subspace tracking performance of SURV, HURV and URV in a stationary case at given SNRs. Fig. 2(a), Fig. 2(b) and Fig. 2(c) show the rank error rate compared with the real number of signals d, the rank error rate compared with $\hat{d}_{\text{svd}}$

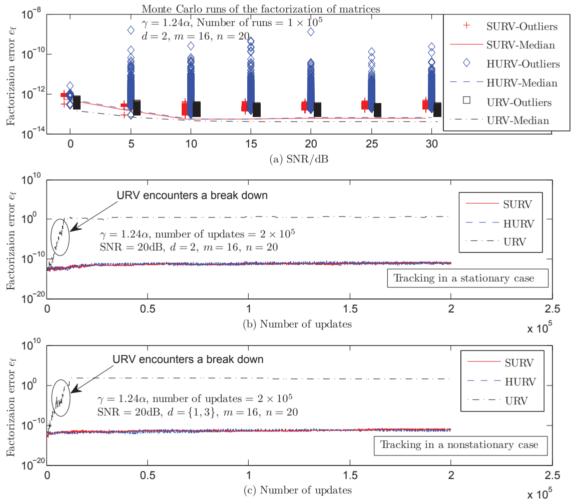

Fig. 1. Factorization error of SURV, HURV and URV.

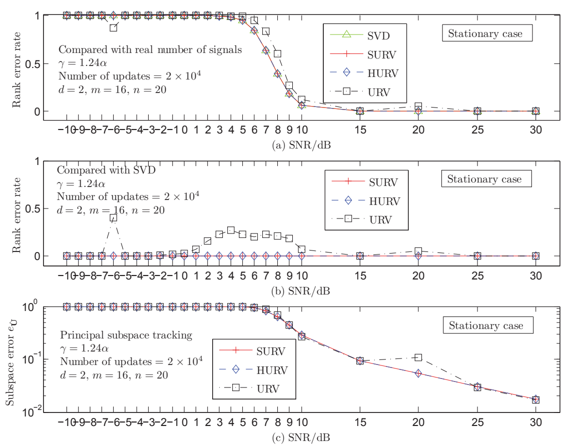

Fig. 2. Subspace tracking performance of SURV, HURV and URV in a stationary case at given SNRs.

and the averaged subspace error of the estimated principal subspace over 2×10^4 updates, respectively. The rank estimate of URV is unreliable at low SNRs, while SURV and HURV always give rank estimates consistent with SVD. This implies that URV cannot track the rank of the subspace well in nonstationary cases.

Fig. 3 shows the subspace tracking performance of SURV, HURV and URV in a nonstationary case at given SNRs with d switching between 2 and 4 every 150 samples. Fig. 3 is similar to Fig. 2 except that in Fig. 3(a) and Fig. 3(c) we collect data only in stationary parts without those in the transient parts at rank changes. It is seen from Fig. 3 that SURV and HURV always give rank estimates consistent with SVD and good estimates of the principal subspace. However, URV gives a lot of rank errors and encounters break downs even at high SNRs.

6 Conclusion

This paper proposed a signed URV decomposition and its updating algorithm for subspace tracking. The proposed updating algorithm

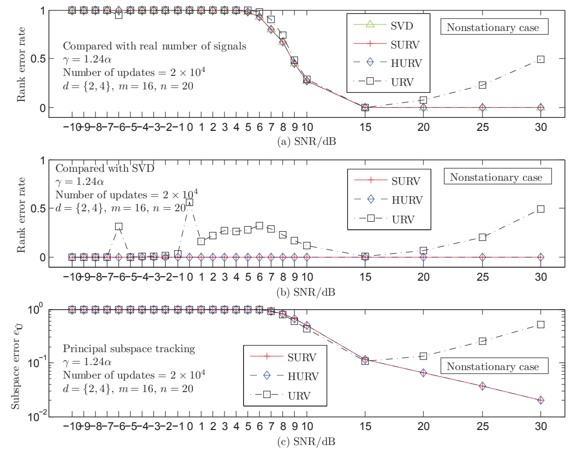

Fig. 3. Subspace tracking performance of SURV, HURV and URV in a nonstationary case at given SNRs.

uses at most only a single hyperbolic rotation per vector update and provides a numerically stable computation of the Schur subspace estimators (unlike the previously proposed algorithm in [5]).

7 References

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

[2] G. W. Stewart, "An updating algorithm for subspace tracking," *IEEE Trans. Signal Process.*, vol. 40, pp. 1535–1541, 1992.

[3] H. Park and L. Eldén, "Downdating the rank-revealing URV decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 16, no. 1, pp. 138–155, 1995.

[4] 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.

[5] ——, "Subspace tracking using a constrained hyperbolic URV decomposition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 4, May 1998, pp. 1953–1956.

[6] 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.

[7] J. Götze, M. Haardt, and J. A. Nossek, "Subspace estimation using unitary Schur-type methods," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1995, pp. 1153–1156.

[8] Y. Li, H. Huang, C. Zhang, and S. Li, "New Schur-type-based PCI algorithms for reverberation suppression in active sonar," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 4, Mar. 2005, pp. 641–644.

[9] A. Edelman, "The distribution and moments of the smallest eigenvalue of a random matrix of Wishart type," *Lin. Alg. Applicat.*, vol. 159, pp. 55–80, Dec. 1991.

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