A SCHUR METHOD FOR LOW-RANK MATRIX APPROXIMATION*

ALLE-JAN VAN DER VEEN

Abstract. The usual way to compute a low-rank approximant of a matrix H is to take its singular value decomposition (SVD) and truncate it by setting the small singular values equal to 0. However, the SVD is computationally expensive. This paper describes a much simpler generalized Schur-type algorithm to compute similar low-rank approximants. For a given matrix H which has d singular values larger than ε, we find all rank d approximants $\hat{H}$ such that $H-\hat{H}$ has 2-norm less than ε. The set of approximants includes the truncated SVD approximation. The advantages of the Schur algorithm are that it has a much lower computational complexity (similar to a QR factorization), and directly produces a description of the column space of the approximants. This column space can be updated and downdated in an on-line scheme, amenable to implementation on a parallel array of processors.

Key words. matrix approximation, rank revealing factorizations, subspace estimation.

AMS subject classifications. primary 47A58, secondary 15A60, 65F35.

1. Introduction. We consider the following problem: for a given matrix $H \in \mathbf{C}^{m\times n}$ and tolerance level $\varepsilon \geq 0$, describe all matrices $\hat{H}$ such that

$$\begin{array}{ll} (a) & \|H-\hat{H}\| \leq \varepsilon\,, \\ (b) & \mathrm{rank}(\hat{H}) = d\,, \end{array} \tag{1}$$

where d is equal to the number of singular values of H that are larger than ε. ($\|\cdot\|$ denotes the matrix 2-norm.) Such a matrix $\hat{H}$ is a low-rank approximation of H in 2-norm. Note that there are no approximants of lower rank than d, and that we do not try to compute an approximant $\hat{H}$ of rank d that *minimizes* $\|H-\hat{H}\|$, but rather one in which the approximation error is limited. These approximants can be computed with significantly less effort.

One way to obtain an approximant which satisfies (1) is by computing a singular value decomposition (SVD) of H:

$$H = U\Sigma V^* = [U_1 \quad U_2] \begin{bmatrix} \Sigma_1 & \\ & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^* \\ V_2^* \end{bmatrix} \tag{2}$$
$$(\Sigma_1)_{ii} > \varepsilon\,, \quad (\Sigma_2)_{ii} \leq \varepsilon\,.$$

Here, U and V are unitary matrices, and Σ is a diagonal matrix which contains the singular values σ_k of H. The matrices are partitioned such that Σ_1 contains the singular values that are larger than ε, and Σ_2 contains those that are smaller than ε. With this decomposition, a rank d approximant $\hat{H}$ is

$$\hat{H} = U_1\Sigma_1 V_1^*\,.$$

This 'truncated SVD' approximant is widely used and effectively obtained by setting the singular values that are smaller than ε equal to zero. It actually minimizes the approximation error: $\|H-\hat{H}\| = \sigma_{d+1} < \varepsilon$, and is optimal in Frobenius norm as well. However, the SVD is expensive to compute, and much of the information that it reveals is not even used. Often, we are not so much interested in the individual singular vectors, but in the principal subspaces, spanned by the columns of U_1 and V_1. As we show in this paper, it is indeed possible to obtain

* Delft University of Technology, Dept. Electrical Eng., 2628 CD Delft, The Netherlands. Research performed in part while on leave at Stanford University, Department of Computer Science/SCCM, Stanford, CA 94305-2140. This research was supported by the commission of the EC under the ESPRIT BRA program 6632 (NANA2), by ARPA contract no. F49620-91-C-0086, monitored by the AFOSR, and by the NSF, grant no. DMS-9205192.

[0] To appear in SIAM J. Matrix Anal. Appl., Jan. 1996

a parametrization of these subspaces and of all rank-d 2-norm approximants. All necessary information is gleaned from an implicit and non-unique factorization of $HH^* - \varepsilon^2 I$ as

$$HH^* - \varepsilon^2 I \;=\; BB^* - AA^*\,, \qquad [A, B] \text{ invertible}\,,$$

which is provided by a 'hyperbolic' QR factorization of $[\varepsilon I \;\; H]$. Such a factorization is similar to an ordinary QR factorization, except that it uses a matrix which is unitary with respect to an indefinite inner product. Under additional regularity assumptions on H, this factorization may be computed using a generalized Schur-type method, which requires only about $1/2m^2n$ operations (elementary rotations) for a matrix of size $m \times n$.[1] The column span of the approximants is directly obtained from B and A: it is proven that all suitable column spans are given by the range of

$$B - AM\,, \qquad \|M\| \leq 1\,.$$

The computation of an approximant itself requires an additional $n \times n$ matrix inversion, or a projection of H onto this subspace.

Continuing efforts on SVD algorithms have reduced its computational complexity to be mainly that of reducing a matrix to a bidiagonal form: not much more than the complexity of a QR factorization. However, a remaining disadvantage of the SVD in demanding applications is that it is difficult to update the decomposition for a growing number of columns of H. Indeed, there are important applications in signal processing (e.g. adaptive beamforming, model identification, adaptive least squares filters) that require on-line estimation of the principal subspace, for growing values of n. A number of other methods have been developed that alleviate the computational requirements, yet retain important information such as numerical rank and principal subspaces. Some of these techniques are the URV decomposition [1], which is a rank revealing form of a complete orthogonal decomposition [2], and the rank revealing QR decomposition (RRQR), [3–8], see [8] for a review. The URV algorithm is iterative and requires estimates of the conditioning of certain submatrices at every step of the iteration. This is a global and data-dependent operation: not a very attractive feature. The SVD and URV decomposition can be updated [9, 1], which is still an iterative process, although it has been shown recently that a simpler scheme is feasible if the updating vectors satisfy certain stationarity assumptions [10, 11]. An initial computation of the RRQR consists of an ordinary QR, followed by an iteration that makes the decomposition rank revealing. As a one-sided decomposition, the RRQR is easier to update than an SVD, but also requires (incremental) condition estimations at each updating step.

An important aspect of the hyperbolic QR factorization is that, similar to a QR factorization, it can be updated very straightforwardly for growing n. The rank of the approximants (the dimension of the principal subspace) is updated as part of the process without any condition estimation. Nonetheless, the method provides an exact error bound on the subspace estimates, in terms of the associated matrix approximation error ε. Similar to the URV and the RRQR, the value of ε has to be fixed beforehand. Another aspect is that the Schur method for computing the hyperbolic QR factorization is a parallel algorithm with only local and regular data dependencies, and is very straightforward to implement on a systolic array of processors. The structure of the array is the same as that of the well-known Gentleman-Kung triangular array for the computation of a QR factorization using Givens rotations [12]. One negative aspect of the Schur algorithm is that it uses hyperbolic rotations, which are potentially unbounded and could make the approximation scheme less robust than the SVD or the URV. This is more

[1] To set our mind, we usually assume that $m \leq n$, but all results remain true when $m > n$.

a property of the implementation than of the overall technique: it occurs only if certain submatrices have a singular value close to ε, and in these cases, a simple local pivoting scheme suffices to virtually eliminate any risk of breakdown. Alternatively, we may derive factorization schemes that minimize the number of hyperbolic rotations.

Connections. Schur methods *an sich* are well known. Originally, Schur devised this algorithm to test whether a polynomial is bounded within the complex unit disc [13]. Schur algorithms occur in certain constrained interpolation problems (viz. [14,15]), rational approximation by analytic functions [16], factorization and inversion of positive definite and later also indefinite Toeplitz matrices (viz. the review in [17]), and have been generalized in a number of senses to non-Toeplitz matrices. A generalization that comes close to the description here is by Dewilde and Deprettere [18], for Schur-parametrizations and band approximations of positive definite matrices, and by Diepold and Pauli [19, 20], for indefinite matrix cases. In the linear algebra community, similar generalized Schur methods, but known under other names, have been used for the solution of positive definite systems [21, 22] and for the downdating of QR and Cholesky factorizations [23], although the main emphasis has been on hyperbolic Householder transformations for the same purposes [24–28]. The HR-decomposition in [24], later known as the hyperbolic QR factorization (*e.g.*, [29]), is in fact precisely the tool we need.

The present application to low rank matrix approximation has been unknown so far. It is derived as a special case of a new theory for model reduction of time-varying systems [30,31]. The time-invariant counterpart (approximation of a Hankel matrix by one of lower rank) has been known for more than a decade and is widely used in systems theory and control for model reduction and for solving H_∞-control problems (viz. [32, 33]). This theory goes back to the work of Adamjan, Arov, and Krein, in 1971, on the solution of the Schur-Takagi interpolation problem [16].

Structure of the paper. The remainder of the paper is organized as follows. Section 2 is a review of those properties of J-unitary matrices that we need in this paper, such as the existence of a hyperbolic QR factorization. In section 3, this factorization is used to prove a basic version of the approximation theorem, and we introduce a parametrization of all 2-norm approximants of rank d. Some values of the parameters that lead to interesting approximants are discussed. The computation of the factorization is the topic of section 4. It is shown that the factorization can be computed using a Schur-type algorithm, although certain extra conditions have to be imposed on the matrix H. We derive necessary and sufficient conditions so that the algorithm does not break down, and discuss some simple pivoting schemes to alleviate these conditions. Finally, in section 5, the algorithm is applied to a test case, to show the behavior of some of the approximants and the effectiveness of the pivoting scheme.

Notation. The superscript $(\cdot)^*$ denotes complex conjugate transposition, $\mathcal{R}(A)$ is the column span (range) of the matrix A, I_m is the $m\times m$ identity matrix, and $0_{m\times n}$ is an $m\times n$ matrix with zero entries. $A^\dagger$ denotes the pseudo-inverse of A. At some point, we will use the notation $A_{[i]}$ to be the submatrix of A, consisting of its first till the i-th row, and $A_{[i,k]}$ to denote the first k columns of $A_{[i]}$.

2. J-Unitary matrices. We review the definition and some properties of J-unitary matrices, most of which are well-known. A matrix Θ is *J-unitary* if it satisfies

$$\Theta^* J \Theta = J\,, \qquad \Theta J \Theta^* = J\,, \qquad \text{where } J = \begin{bmatrix} I & \\ & -I \end{bmatrix}. \tag{3}$$

J is a *signature matrix;* the identity matrices need not have equal sizes. Θ is necessarily a square matrix, and invertible as well: $\Theta^{-1} = J\Theta^* J$. We partition Θ according to J as

$$\Theta = \begin{bmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{bmatrix}. \tag{4}$$

The J-unitarity of Θ implies a.o. $\Theta_{22}^*\Theta_{22} = I + \Theta_{12}^*\Theta_{12}$. From this expression, we derive in turn the properties

$$\begin{array}{llll} (a) & \Theta_{22} \text{ is invertible,} & (c) & \|\Theta_{12}\Theta_{22}^{-1}\| < 1, \\ (b) & \|\Theta_{22}^{-1}\| \le 1, & (d) & \|\Theta_{11} - \Theta_{12}\Theta_{22}^{-1}\Theta_{21}\| \le 1. \end{array} \tag{5}$$

Indeed, (a) follows because Θ_{22} is also square, (b) is obtained from

$$\Theta_{22}^{-*}\Theta_{22}^{-1} + (\Theta_{22}^{-*}\Theta_{12}^*)(\Theta_{12}\Theta_{22}^{-1}) = I, \tag{6}$$

so that $\Theta_{22}^{-*}\Theta_{22}^{-1} \le I$, and (c) results from the same expression because $\Theta_{22}^{-*}\Theta_{22}^{-1} > 0$. By elementary algebra, one verifies that the matrix

$$\begin{bmatrix} \Theta_{11} - \Theta_{12}\Theta_{22}^{-1}\Theta_{21} & -\Theta_{12}\Theta_{22}^{-1} \\ \Theta_{22}^{-1}\Theta_{21} & \Theta_{22}^{-1} \end{bmatrix}$$

is, in fact, unitary, which implies (d).

Similarly, we have

$$\begin{array}{llll} (a) & \Theta_{11} \text{ is invertible,} & (c) & \|\Theta_{11}^{-1}\Theta_{12}\| < 1, \\ (b) & \|\Theta_{11}^{-1}\| \le 1, & (d) & \|\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12}\| \le 1. \end{array} \tag{7}$$

Another important property of J-unitary matrices is the preservation of the J-inner product. Suppose that A, B, C, D are matrices, related as $[C \;\; D] = [A \;\; B]\Theta$. The J-unitarity of Θ implies

$$\begin{aligned} AA^* - BB^* &= [A \;\; B]J[A \;\; B]^* \\ &= [A \;\; B]\Theta J\Theta^*[A \;\; B]^* \\ &= CC^* - DD^*. \end{aligned} \tag{8}$$

Motivated by this equation, we say that J associates a positive signature to the columns of A, a negative signature to the columns of B, and likewise for C and D. We will sometimes denote this in equations by writing $+$ and $-$ on top of A and B.

So far, the signature matrix J in (3) was sorted: the diagonal has first all the positive entries, then the negative ones. We will at some point also need *unsorted* signature matrices $\tilde{J}$, which is any diagonal matrix with diagonal entries equal to $+1$ or -1. As a generalization of the definition in (3), we will say that a matrix $\tilde{\Theta}$ is $(\tilde{J}_1, \tilde{J}_2)$-unitary[2] with respect to signature matrices $\tilde{J}_1, \tilde{J}_2$ if it satisfies

$$\tilde{\Theta}^*\tilde{J}_1\tilde{\Theta} = \tilde{J}_2, \qquad \tilde{\Theta}\tilde{J}_2\tilde{\Theta}^* = \tilde{J}_1. \tag{9}$$

Again, $\tilde{\Theta}$ is square and invertible: $\tilde{\Theta}^{-1} = \tilde{J}_2\tilde{\Theta}^*\tilde{J}_1$. Sylvester's law of inertia claims that the number of positive entries in $\tilde{J}_1$ must be equal to the number of positive entries in $\tilde{J}_2$, and similar for the negative entries. An unsorted signature matrix $\tilde{J}_1$ can always be sorted by a permutation matrix Π_1 such that $J_1 = \Pi_1\tilde{J}_1\Pi_1^*$ is sorted. If also $J_2 = \Pi_2\tilde{J}_2\Pi_2^*$ is sorted, and $\tilde{\Theta}$

[2] We will sometimes generically write J-unitary, to avoid being overly detailed.

satisfies (9), then actually $J_1 = J_2 =: J$, and $\Theta := \Pi_1 \tilde{\Theta} \Pi_2^*$ is J-unitary in the sorted sense of (3). The permutation, and hence Θ, is not unique, but this is usually irrelevant. We work with Θ rather than $\tilde{\Theta}$ in cases where its partitioning into submatrices, as in (4), is important, but the properties (5) are independent of precisely which Θ is chosen.

A matrix A is said to be *$\tilde{J}$-nonsingular*, with respect to a certain signature matrix $\tilde{J}$, if $A\tilde{J}A^*$ is nonsingular. It is immediate that if A is $\tilde{J}_1$-nonsingular and $\tilde{\Theta}$ is a $(\tilde{J}_1, \tilde{J}_2)$-unitary matrix, then $A\tilde{\Theta}$ is $\tilde{J}_2$-nonsingular. The following basic result claims that J-nonsingular matrices can be factored:

THEOREM 2.1. *A matrix $A : m \times (m+n)$ is $\tilde{J}_1$-nonsingular if and only if there exists a signature matrix $\tilde{J}_2$ and a $(\tilde{J}_1, \tilde{J}_2)$-unitary matrix $\tilde{\Theta}$ such that*

$$A\tilde{\Theta} = [X \quad 0_{m\times n}], \qquad X : m \times m, \text{ invertible.} \tag{10}$$

Proof. Assume that A is $\tilde{J}$-nonsingular. Then we can factor $A\tilde{J}_1A^*$ as

$$A\tilde{J}_1A^* = X\tilde{J}'X^*, \qquad X : m \times m, \text{ invertible,}$$

for some $m \times m$ signature matrix $\tilde{J}'$. This factorization exists and can in principle be computed from an LDU factorization with pivoting, or from an eigenvalue decomposition of $A\tilde{J}_1A^*$. Since A is $\tilde{J}_1$-nonsingular, it is also nonsingular in the ordinary sense, so that there exists a matrix $T : (m+n) \times m$, such that $AT = X$. T is not unique. Because X is invertible, we can take

$$T = \tilde{J}_1A^*(A\tilde{J}_1A^*)^{-1}X\,.$$

Using $(A\tilde{J}_1A^*)^{-1} = X^{-*}\tilde{J}'X^{-1}$, it is directly verified that this T satisfies $T^*\tilde{J}_1T = \tilde{J}'$. The remainder of the proof is technical and shows that T can be extended to a square, J-unitary matrix. From $T^*\tilde{J}_1T = \tilde{J}'$ it follows that the m columns of $\tilde{J}_1T$ are linearly independent. Taking any basis of the orthogonal complement of the range of $\tilde{J}_1T$ gives a matrix K, with n independent columns, such that $T^*\tilde{J}_1K = 0$. The matrix $[T \;\; K]$ is invertible, because it is square and its kernel is empty:

$$\begin{aligned} [T \quad K]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0 \qquad &\Rightarrow \qquad T^*\tilde{J}_1[T \quad K]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0 \\ &\Rightarrow \qquad x_1 = 0 \end{aligned}$$

and subsequently, it also follows that $x_2 = 0$. It remains to normalize the columns of K. Put

$$[T \quad K]^*\tilde{J}_1[T \quad K] = \begin{bmatrix} \tilde{J}' & \\ & N \end{bmatrix}.$$

N is nonsingular because $[T \;\; K]$ is invertible, and we can factor it as $N = R^*\tilde{J}''R$. Put

$$\tilde{\Theta} = [T \quad KR^{-1}], \qquad \tilde{J}_2 = \operatorname{diag}[\tilde{J}', \tilde{J}'']\,.$$

Then $\tilde{\Theta}$ is $(\tilde{J}_1, \tilde{J}_2)$-unitary, and satisfies (10). □

A recursive application of this theorem proves that, under additional conditions, A has a triangular factorization:

COROLLARY 2.2. *Let $A : m \times (m+n)$ be $\tilde{J}_1$-nonsingular. Denote by $A_{[i]}$ the submatrix of A, consisting of its first i rows. Then there exists a signature matrix $\tilde{J}_2$, and a $(\tilde{J}_1, \tilde{J}_2)$-unitary matrix $\tilde{\Theta}$ such that*

$$A\tilde{\Theta} = [X \quad 0_{m\times n}], \qquad X : m \times m, \text{ lower triangular, invertible}$$

if and only if $A_{[i]}$ is $\tilde{J}_1$-nonsingular, for $i = 1, \ldots, m$. If the diagonal entries of X are chosen to be positive, then X is unique. Such a factorization was proven in [24] for square matrices A and upper triangular X, but this result extends directly to the rectangular case. In [24], it was called the HR-decomposition, and it is also known as the hyperbolic QR factorization [29].

3. Approximation theory.

3.1. Central approximant. For a given $m \times n$ data matrix H and threshold ε, denote the SVD of H as in (2). Suppose that d singular values of H are larger than ε, and that none of them are equal to ε. Our approximation theory is based on an implicit factorization of

$$HH^* - \varepsilon^2 I = BB^* - AA^*. \tag{11}$$

This is a Cholesky factorization of an indefinite Hermitian matrix. A and B are chosen to have full column rank. They are not unique, but by Sylvester's inertia law, their dimensions are well-defined. Using the SVD of H, we obtain one possible decomposition as

$$HH^* - \varepsilon^2 I = U_1(\Sigma_1^2 - \varepsilon^2 I)U_1^* + U_2(\Sigma_2^2 - \varepsilon^2 I)U_2^*,$$

where the first term is positive semidefinite and has rank d, and the second term is negative semidefinite and has rank $m-d$. Hence, B has d columns, and A has $m-d$ columns.

To obtain an implicit factorization which avoids computing HH^*, we make use of theorem 2.1.

THEOREM 3.1. *Let $H : m \times n$ have d singular values larger than ε, and none equal to ε. Then there exists a J-unitary matrix Θ such that*

$$[\varepsilon I_m \quad H]\,\Theta = [A' \quad B'] \tag{12}$$

where $A' = [A \;\; 0_{m\times d}]$, $B' = [B \;\; 0_{m\times n-d}]$, $A: m \times (m-d)$, $B: m \times d$, and $[A \;\; B]$ is of full rank.

Proof. The matrix $[\varepsilon I_m \;\; H]$ is J-nonsingular: by assumption, $\varepsilon^2 I - HH^*$ has d negative, $m-d$ positive, and no zero eigenvalues. Hence theorem 2.1 implies that there exists $\tilde{\Theta} : [\varepsilon I_m \;\; H]\tilde{\Theta} = [X \;\; 0_{m\times n}]$. The columns of X are the columns of $[A, B]$, in some permuted order, where A, B correspond to columns of X that have a positive or negative signature, respectively. After sorting the columns of $[X \;\; 0]$ according to their signature, equation (12) results. □

Note that, by the preservation of J-inner products (equation (8)), equation (12) implies (11). From the factorization (12), we can immediately derive a 2-norm approximant satisfying the conditions in (1). To this end, partition Θ according to its signature J into 2×2 blocks, like in (4).

THEOREM 3.2. *Let $H : m \times n$ have d singular values larger than ε, and none equal to ε. Define the factorization $[\varepsilon I_m \;\; H]\Theta = [A' \;\; B']$ as in theorem 3.1. Then*

$$\hat{H} = B'\Theta_{22}^{-1} \tag{13}$$

is a rank d approximant such that $\|H - \hat{H}\| < \varepsilon$.

Proof. $\hat{H}$ is well-defined because Θ_{22} is invertible (equation (5*a*)). It has rank d because $B' = [B \quad 0]$ has rank d. By equation (12), $B' = \varepsilon I\Theta_{12} + H\Theta_{22}$, hence $H - \hat{H} = -\varepsilon\Theta_{12}\Theta_{22}^{-1}$. It remains to use the fact that $\Theta_{12}\Theta_{22}^{-1}$ is contractive (equation (5*c*)). □

We mentioned in the introduction that the column span (range) of the approximant is important in signal processing applications. From theorem 3.2, it is seen that this column span is equal to that of B: it is directly produced by the factorization. However, remark that $[A \;\; B]$ in (12) is not unique: for any J-unitary matrix Θ_1, $[A_1 \;\; B_1] = [A \;\; B]\Theta_1$ also satisfies $\varepsilon^2 I - HH^* = A_1A_1^* - B_1B_1^*$, and could also have been produced by the factorization. E.g., for some choices of Θ_1, we will have $\mathcal{R}(B) = \mathcal{R}(U_1)$, and $\mathcal{R}(A) = \mathcal{R}(U_2)$. Using Θ_1, we can find more approximants. A parametrization of all 2-norm approximants is the topic of the following section.

3.2. Parametrization of all approximants. We will now give a formula of all possible 2-norm approximants $\hat{H}$ of H of rank equal to d; there are no approximants of rank less than d. The formula is in terms of a chain fraction description. Similar formulas frequently occur in constrained interpolation theory (see *e.g.*, [14, 15] and references therein).

The set of all minimal-rank 2-norm approximants will be parametrized by matrices $S_L : m \times n$, with 2×2 block partitioning as

$$S_L = \begin{matrix} \\ m-d \\ d \end{matrix} \begin{matrix} \begin{matrix} d & n-d \end{matrix} \\ \begin{bmatrix} (S_L)_{11} & (S_L)_{12} \\ (S_L)_{21} & (S_L)_{22} \end{bmatrix} \end{matrix}, \tag{14}$$

and satisfying the requirements

$$\begin{array}{ll} (i) & \text{contractive: } \|S_L\| \le 1\,, \\ (ii) & \text{block lower: } (S_L)_{12} = 0\,. \end{array} \tag{15}$$

The first condition on S_L will ensure that $\|H - \hat{H}\| \le \varepsilon$, whereas the second condition is required to have $\hat{H}$ of rank d.

THEOREM 3.3. *With the notation and conditions of theorem 3.2, all rank d 2-norm approximants $\hat{H}$ of H are given by*

$$\hat{H} \;=\; (B' - A'S_L)(\Theta_{22} - \Theta_{21}S_L)^{-1}\,,$$

where S_L satisfies (i): $\|S_L\| \le 1$, and (ii): $(S_L)_{12} = 0$. The approximation error is

$$S := H - \hat{H} \;=\; \varepsilon(\Theta_{11}S_L - \Theta_{12})(\Theta_{22} - \Theta_{21}S_L)^{-1}\,. \tag{16}$$

Proof. The proof is given in the appendix. □

By this theorem, an estimate of the principal subspace of H is given by $\mathcal{R}(\hat{H}) = \mathcal{R}(B' - A'S_L) = \mathcal{R}(B - A(S_L)_{11})$, for any valid choice of S_L. Note that $(S_L)_{11}$ ranges over the set of all contractive $(m-d) \times d$ matrices, so that all suitable principal subspace estimates are given by

$$\mathcal{R}(B - AM)\,, \qquad \|M\| \le 1\,.$$

The distance of a subspace estimate with the actual principal subspace, $\mathcal{R}(U_1)$, is measured only implicitly, in the sense that there exists an approximant $\hat{H}$ with this column span that is ε-close to H. Actually, for each subspace estimate there are many such approximants, since the subspace estimate only depends on $(S_L)_{11}$, whereas the approximant also depends on $(S_L)_{21}$ and $(S_L)_{22}$.

The choice of a particular approximant $\hat{H}$, or subspace estimate $\mathcal{R}(\hat{H})$, boils down to a suitable choice of the parameter S_L. Various choices are interesting:

1. The approximant $\hat{H}$ in theorem 3.2 is obtained by taking $S_L = 0$. This approximant is the most simple to compute; the principal subspace estimate is equal to the range of B. The approximation error is given by $\varepsilon\|\Theta_{12}\Theta_{22}^{-1}\|$. Note that, even if all nonzero singular values of H are larger than ε so that it is possible to have $\hat{H} = H$, the choice $S_L = 0$ typically does not give zero error. Hence, this simple choice of S_L could lead to 'biased' estimates. This is confirmed in the simulation example in section 5, and occurs in cases where σ_d is close to ε.

2. As the truncated SVD solution satisfies the requirements, there is an S_L which yields this particular solution and minimizes the approximation error. However, computing this S_L requires an SVD, or a hyperbolic SVD [29].

3. It is sometimes possible to obtain a uniform approximation error. First write equation (16) in a more implicit form,

$$\begin{bmatrix} \varepsilon^{-1}SG \\ -G \end{bmatrix} = \begin{bmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{bmatrix} \begin{bmatrix} S_L \\ -I_n \end{bmatrix},$$

where G is an invertible $n\times n$ matrix. This equation implies

$$G^*(\varepsilon^{-2}S^*S - I_n)G \;=\; S_L^*S_L - I_n\,.$$

Suppose $m \le n$. If we can take S_L to be an isometry, $S_LS_L^* = I_m$, then $\mathrm{rank}(S_L^*S_L - I_n) = n-m$. It follows that $\varepsilon^{-1}S$ must also be an isometry, so that all singular values of $S = H-\hat{H}$ are equal to ε: the approximation error is uniform. S_L can be an isometry and have $(S_L)_{12} = 0$ only if $d \ge m-d$, *i.e.*, $d \ge m/2$. In that case, we can take for example $S_L = [I_m\ 0]$. This approximant might have relevance in signal processing applications where a singular data matrix is distorted by additive uncorrelated noise with a covariance matrix $\sigma^2 I_m$.

4. If we take $S_L = \Theta_{11}^{-1}\Theta_{12}$, then we obtain $\hat{H} = H$ and the approximation error is zero. Although this S_L is contractive (viz. equation (7)), it does not satisfy the condition $(S_L)_{12} = 0$, unless $d = m$ or $d = n$. Simply putting $(S_L)_{12} = 0$ might make the resulting S_L non-contractive. To satisfy both conditions on S_L, a straightforward modification is by setting

$$S_L = \Theta_{11}^{-1}\Theta_{12}\begin{bmatrix} I_d & \\ & 0_{n-d} \end{bmatrix} = \begin{bmatrix} (\Theta_{11}^{-1}\Theta_{12})_{11} & 0 \\ (\Theta_{11}^{-1}\Theta_{12})_{21} & 0 \end{bmatrix} \tag{17}$$

The corresponding approximant is

$$\hat{H}^{(1)} := (B' - A'\Theta_{11}^{-1}\Theta_{12}\left[\begin{smallmatrix} I & 0 \\ 0 & 0 \end{smallmatrix}\right])(\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12}\left[\begin{smallmatrix} I & 0 \\ 0 & 0 \end{smallmatrix}\right])^{-1}, \tag{18}$$

and the corresponding principal subspace estimate is given by the range of

$$B^{(1)} := B - A(\Theta_{11}^{-1}\Theta_{12})_{11}\,. \tag{19}$$

Both the subspace estimate and the approximant itself can be computed by a Schur complement formula. The subspace estimate is 'unbiased' in a sense discussed below, and is usually quite accurate when σ_d is not very close to ε, as shown in simulation examples (section 5). The approximation error is determined by

$$S = H - \hat{H}^{(1)} = \varepsilon\Theta_{12}\begin{bmatrix} 0_d & \\ & -I_{n-d} \end{bmatrix}(\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12}\left[\begin{smallmatrix} I & 0 \\ 0 & 0 \end{smallmatrix}\right])^{-1}. \tag{20}$$

This shows that the rank of S is at most equal to $\min(m, n-d)$. If $m = n$, then the rank of S is $m-d$, *i.e.*, the error has the same rank as a truncated SVD solution would give.

5. To improve on the approximation error, we propose to take $(S_L)_{11} = (\Theta_{11}^{-1}\Theta_{12})_{11}$, as in the previous item, and use the freedom provided by $(S_L)_{21}$ and $(S_L)_{22}$ to minimize the norm of the error. The subspace estimate is only determined by $(S_L)_{11}$ and is the same as before. Instead of minimizing in terms of S_L, which involves a non-linear function and a contractivity constraint, we make use of the fact that we know already the column span of the approximant: we are looking for $\hat{H} = B^{(1)}N$, with $B^{(1)}$ given by (19) and $N : d\times n$ a minimizer of

$$\min_N \|H - B^{(1)}N\|\,.$$

A solution is given by $N = B^{(1)\dagger}H$, and the resulting approximant is

$$\begin{aligned} \hat{H} &= B^{(1)}B^{(1)\dagger}H \\ &=: \hat{H}^{(2)}, \end{aligned} \tag{21}$$

the projection of H onto $\mathcal{R}(B^{(1)})$. Although we do not compute the S_L to which this approximant corresponds, the residual error is guaranteed to be less than or equal to ε, because it is at most equal to the norm of S in (20). Hence, there will be some S_L that satisfies the constraints, although we never compute it explicitly. For this S_L, the rank of the residual error is always at most equal to $m-d$, the rank of $I_m - B^{(1)}B^{(1)\dagger}$.

One other important feature of the subspace estimate $B^{(1)}$ in (19) is that it is *unbiased*, in the following sense.

LEMMA 3.4. $\mathcal{R}(B^{(1)}) \subset \mathcal{R}(H)$.

Proof. From $[(A\ 0)\ (B\ 0)] = [A'\ B'] = [\varepsilon I\ H]\Theta$, we have

$$\begin{cases} [A \quad 0] &= \varepsilon\Theta_{11} + H\Theta_{21} \\ [B \quad 0] &= \varepsilon\Theta_{12} + H\Theta_{22} \end{cases}$$

Hence

$$\begin{aligned} [B^{(1)}\ 0] &= [B \quad 0] - [A \quad 0]\Theta_{11}^{-1}\Theta_{12}\begin{bmatrix} I & \\ & 0 \end{bmatrix} \\ &= (\varepsilon\Theta_{12} + H\Theta_{22}) - (\varepsilon\Theta_{11} + H\Theta_{21})\Theta_{11}^{-1}\Theta_{12}\begin{bmatrix} I & \\ & 0 \end{bmatrix} \\ &= H(\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12})\begin{bmatrix} I & \\ & 0 \end{bmatrix} + H\Theta_{22}\begin{bmatrix} 0 & \\ & I \end{bmatrix} + \varepsilon\Theta_{12}\begin{bmatrix} 0 & \\ & I \end{bmatrix} \end{aligned}$$

so that

$$B^{(1)} = H(\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12})\begin{bmatrix} I \\ 0 \end{bmatrix}.$$

□

With equation (7*d*), we also have

$$\|B^{(1)}\| \le \|H\|. \tag{22}$$

This shows that, although J-unitary matrices may be large, this particular subspace estimate is bounded in norm by the matrix it was derived from.

4. Computation of Θ. In this section, we consider the actual construction of a J-unitary matrix Θ such that

$$[\varepsilon I \quad H]\,\Theta = [A' \quad B'], \qquad J = \begin{bmatrix} I_m & \\ & -I_n \end{bmatrix}.$$

The proof of theorem 2.1 provides a technique to compute Θ, but the construction is global and not really attractive. We are looking for algorithms that do not square the data and that allow easy updating of the factorization as more and more columns of H are included (growing n). Θ will be computed in two steps: $\Theta = \tilde{\Theta}\Pi$, where $\tilde{\Theta}$ is a $(J, \tilde{J}_2)$-unitary matrix with respect to J and an unsorted signature $\tilde{J}_2$ and is such that

$$\begin{matrix} + & - & & +/- & +/- \end{matrix}$$
$$\left[\varepsilon I_m \quad H\right]\tilde{\Theta} = \left[\begin{matrix} X & 0_{m\times n} \end{matrix}\right], \qquad X : m\times m. \tag{23}$$

Π is any permutation matrix such that $\Pi\tilde{J}_2\Pi^* = J$ is a sorted signature matrix. The latter factorization can be viewed as a hyperbolic QR factorization, in case X has a triangular form, and can be computed in a number of ways. Hyperbolic Householder transformations have

been employed for this purpose [24, 29], zeroing full rows at each step, but the most elementary way is to use elementary rotations to create one zero entry at a time, like Givens rotations for QR factorizations. Such techniques are known as (generalized) Schur algorithms, because of their similarity to the Schur method for Toeplitz matrices. In contrast to hyperbolic Householder transformations, they allow for straightforward updating and downdating. The main differences with the QR factorization, and also with the usual definite Schur algorithms (for which $\varepsilon^2 I - HH^* > 0$) are that, here, the basic operations are J-unitary elementary rotations of up to six different types, and that we have to keep track of signatures to determine which type to use.

The recursive construction of Θ in this way is not always possible, unless extra conditions on the singular values of certain submatrices of H are posed. This is a well-known complication from which all indefinite Schur and hyperbolic Householder methods suffer and that in its ultimate generality can be treated only by global matrix operations (as in [19, 20], or as in the proof of theorem 2.1, which uses an altogether different algorithm). The exceptions occurs only for specific cases, and simple pivoting schemes (column or row permutations) are virtually always adequate to eliminate this problem. We will briefly go into these aspects in section 4.5.

4.1. Elementary rotations. At an elementary level, we are looking for 2×2 matrices $\tilde{\theta}$ such that, for given scalars a, b,

$$[a \quad b]\,\tilde{\theta} \;=\; [x \quad 0]\,,$$

where x is some resulting scalar. The matrices $\tilde{\theta}$ are J-unitary, but with respect to unsorted signature matrices $\tilde{j}_1, \tilde{j}_2$:

$$(\tilde{\theta})^* \tilde{j}_1 \tilde{\theta} = \tilde{j}_2\,, \qquad \tilde{\theta}\tilde{j}_2(\tilde{\theta})^* = \tilde{j}_1\,.$$

The signature matrix $\tilde{j}_1$ is specified along with a, b and signifies the signature of $[a \;\; b]$; $\tilde{j}_2$ is a resulting signature matrix to be computed along with $\tilde{\theta}$ and x, and will be the resulting signature of $[x \;\; 0]$. There are two rules that determine $\tilde{j}_2$. From the J-unitarity of $\tilde{\theta}$, we have that

$$[a \quad b]\,\tilde{j}_1\,[a \quad b]^* \;=\; x(\tilde{j}_2)_{11}x^*$$

$$\Rightarrow\; (\tilde{j}_2)_{11} = \mathrm{sign}\Big([a \quad b]\,\tilde{j}_1\,[a \quad b]^*\Big)\,.$$

We have to assume at this point that the expression in brackets is not zero, so that $(\tilde{j}_2)_{11}$ is either $+1$ or -1. The second diagonal entry of $\tilde{j}_2$ then follows from the inertia rule: by congruence, the number of positive entries in $\tilde{j}_1$ is equal to the number of positive entries in $\tilde{j}_2$, and similarly for the negative entries.

Depending on the signatures, we choose one of the following types of elementary $(\tilde{j}_1, \tilde{j}_2)$-

$[X \quad Y] := [\varepsilon I_m \quad H]$
$\tilde{J} := \begin{bmatrix} I_m & \\ & -I_n \end{bmatrix}$
$\tilde{\Theta} = I_{m+n}$
for $k = 1$ to n and $i = 1$ to m,
  $[a \quad b] := [X(i,i) \quad Y(i,k)]$
  $\tilde{j}_1 := \begin{bmatrix} \tilde{J}(i,i) & 0 \\ 0 & \tilde{J}(m+k,m+k) \end{bmatrix}$
  Compute $\tilde{\theta}$, $\tilde{j}_2$ from $a, b, \tilde{j}_1$ s.t. $[a \;\; b]\tilde{\theta} = [* \;\; 0]$
  Embed $\tilde{\theta}$ into $\tilde{\Theta}_{(i,k)}$
  $[X \quad Y] := [X \quad Y]\tilde{\Theta}_{(i,k)}$
  $\tilde{\Theta} := \tilde{\Theta}\tilde{\Theta}_{(i,k)}$
  $\tilde{J}(i,i) := (\tilde{j}_2)_{1,1}$
  $\tilde{J}(m+k,m+k) := (\tilde{j}_2)_{2,2}$
end
$\tilde{J}_2 := \tilde{J}$

FIG. 1. *Schur algorithm to compute the factorization* $[\varepsilon I \;\; H]\tilde{\Theta} = [X \;\; 0]$ *from* H.

unitary rotations (taking $|s|^2 + |c|^2 = 1$ throughout):

1. $\tilde{j}_1 = \begin{bmatrix} 1 & \\ & -1 \end{bmatrix}$, $\tilde{j}_2 = \begin{bmatrix} 1 & \\ & -1 \end{bmatrix}$ $\Rightarrow$ $\tilde{\theta} = \begin{bmatrix} 1 & -s \\ -s^* & 1 \end{bmatrix} \frac{1}{c}$
2. $\tilde{j}_1 = \begin{bmatrix} 1 & \\ & -1 \end{bmatrix}$, $\tilde{j}_2 = \begin{bmatrix} -1 & \\ & 1 \end{bmatrix}$ $\Rightarrow$ $\tilde{\theta} = \begin{bmatrix} -s^* & 1 \\ 1 & -s \end{bmatrix} \frac{1}{c}$
3. $\tilde{j}_1 = \begin{bmatrix} -1 & \\ & 1 \end{bmatrix}$, $\tilde{j}_2 = \begin{bmatrix} 1 & \\ & -1 \end{bmatrix}$ $\Rightarrow$ $\tilde{\theta} = \begin{bmatrix} -s^* & 1 \\ 1 & -s \end{bmatrix} \frac{1}{c}$
4. $\tilde{j}_1 = \begin{bmatrix} -1 & \\ & 1 \end{bmatrix}$, $\tilde{j}_2 = \begin{bmatrix} -1 & \\ & 1 \end{bmatrix}$ $\Rightarrow$ $\tilde{\theta} = \begin{bmatrix} 1 & -s \\ -s^* & 1 \end{bmatrix} \frac{1}{c}$
5. $\tilde{j}_1 = \begin{bmatrix} 1 & \\ & 1 \end{bmatrix}$, $\tilde{j}_2 = \begin{bmatrix} 1 & \\ & 1 \end{bmatrix}$ $\Rightarrow$ $\tilde{\theta} = \begin{bmatrix} c^* & -s \\ s^* & c \end{bmatrix}$
6. $\tilde{j}_1 = \begin{bmatrix} -1 & \\ & -1 \end{bmatrix}$, $\tilde{j}_2 = \begin{bmatrix} -1 & \\ & -1 \end{bmatrix}$ $\Rightarrow$ $\tilde{\theta} = \begin{bmatrix} c^* & -s \\ s^* & c \end{bmatrix}$

The first case is the standard elementary hyperbolic rotation. The next three cases are obtained from this case by row and column permutations. Cases 5 and 6 are not hyperbolic, but ordinary elliptic rotations, but they are $(\tilde{j}_1, \tilde{j}_2)$-unitary nonetheless. These six cases are sufficient to consider, as every possible signature pair $(\tilde{j}_1, \tilde{j}_2)$ is covered. With $\tilde{j}_1$ and $\tilde{j}_2$ known, we select the appropriate type of rotation matrix, and the rotation parameters s and c follow subsequently from the equation $[a \;\; b]\tilde{\theta} = [x \;\; 0]$ as

case 1, 4 ($\|a\| > \|b\|$):	$s = b/a$,	$c = (1 - s^*s)^{1/2}$
case 2, 3 ($\|a\| < \|b\|$):	$s = a/b$,	$c = (1 - s^*s)^{1/2}$
case 5, 6:	$s = b(a^*a + b^*b)^{-1/2}$,	$c = (1 - s^*s)^{1/2}$.

4.2. Indefinite Schur algorithm. To compute the factorization (23), elementary rotations $\tilde{\theta}$ are embedded in plane rotations $\tilde{\Theta}_{(i,k)}$ which are applied to the columns of $[\varepsilon I \;\; H]$ in the same way as Givens rotations are used for computing a QR factorization. Each plane rotation produces a zero entry in H; specifically, $\tilde{\Theta}_{(i,k)}$ annihilates entry (i,k). A difference with QR is that we have to keep track of the signatures associated to the columns of the matrix to determine which type of rotations to use. The general scheme, however, goes as follows:

$$
[\varepsilon I \quad H] \quad = \quad
\begin{array}{c}
\begin{matrix} + & + & + & - & - & - & - \end{matrix} \\
\left[\begin{array}{ccc|cccc}
\underline{\varepsilon} & & 0 & \underline{\times} & \times & \times & \times \\
 & \varepsilon & & \times & \times & \times & \times \\
0 & & \varepsilon & \times & \times & \times & \times
\end{array}\right] \quad \tilde{\Theta}_{(1,1)} \rightarrow \\
\begin{matrix} - & + & + & + & - & - & - \end{matrix} \\
\left[\begin{array}{ccc|cccc}
\times & & & 0 & \times & \times & \times \\
\times & \underline{\varepsilon} & & \underline{\times} & \times & \times & \times \\
\times & & \varepsilon & \times & \times & \times & \times
\end{array}\right] \quad \tilde{\Theta}_{(2,1)} \rightarrow \\
\begin{matrix} - & + & + & + & - & - & - \end{matrix} \\
\left[\begin{array}{ccc|cccc}
\times & & & 0 & \times & \times & \times \\
\times & \times & & 0 & \times & \times & \times \\
\times & \times & \underline{\varepsilon} & \underline{\times} & \times & \times & \times
\end{array}\right] \quad \rightarrow \\
\begin{matrix} - & + & - & + & + & - & - \end{matrix} \\
\cdots \quad \overset{\tilde{\Theta}_{(m,n)}}{\rightarrow} \quad \left[\begin{array}{ccc|cccc}
\times & & & 0 & 0 & 0 & 0 \\
\times & \times & & 0 & 0 & 0 & 0 \\
\times & \times & \times & 0 & 0 & 0 & 0
\end{array}\right] = [X \quad 0],
\end{array}
$$

$$
\tilde{\Theta} \;=\; \tilde{\Theta}_{(1,1)}\tilde{\Theta}_{(2,1)}\cdots\tilde{\Theta}_{(m,1)}\cdot\tilde{\Theta}_{(1,2)}\cdots\tilde{\Theta}_{(2,2)}\cdots\tilde{\Theta}_{(m,n)}\,.
$$

(Except for the first matrix, the signatures of the columns in the above matrices are examples, as they are data dependent.) The pivot elements at each step are underlined; these entries, along with the signatures of the two columns in which they appear, determine the elementary rotation $\tilde{\theta}$ that will be used at that step, as well as the resulting signature $\tilde{j}_2$. This signature is the new signature of these two columns, after application of the rotation. The algorithm is summarized in figure 1.[3] The nulling scheme ensures that $[\varepsilon I \;\; H]\tilde{\Theta} \;=\; [X \;\; 0]$, where X is a resulting lower triangular invertible matrix; it contains the columns of A and B in some permuted order. The columns of X with a positive signature are the columns of A, the columns with a negative signature are those of B. Hence, the final step (not listed figure 1) is to sort these columns, such that $[X \;\; 0]\Pi = [A \;\; 0 \;\; B \;\; 0] = [A' \;\; B']$. Then $\Theta = \tilde{\Theta}\Pi$ is J-unitary with respect to J, and $[\varepsilon I \;\; H]\Theta = [A' \;\; B']$.

The complexity of the algorithm is similar to that of the QR factorization: about $1/2m^2n$ rotations, or $2m^2n$ flops. The Schur algorithm has a direct implementation on a systolic array of processors. This array is entirely similar to the classical Gentleman-Kung triangular Givens array [12], except that, now, all data entries have a signature associated to them, and the processors have to perform different types of rotations, depending on these signatures. We omit the details.

4.3. Updating and downdating. The Schur method is straightforward to update as more and more columns of H become known. If $[\varepsilon I \;\; H_n]\tilde{\Theta}_{(n)} = [X_n \;\; 0]$ is the factorization at point

[3] As an aside, we mention that Bojanczyk et al. [23] have developed a numerically more stable implementation of the application of hyperbolic plane rotations to vectors. This is probably of relevance in the present context.

n and $H_{n+1} = [H_n \;\; h_{n+1}]$, then, because the algorithm works column-wise,

$$[\varepsilon I \quad H_{n+1}]\tilde{\Theta}_{(n+1)} = [X_{n+1} \quad 0] \quad \Rightarrow \quad \begin{array}{l}[X_n \quad 0 \quad h_{n+1}]\,\tilde{\theta}^{(n+1)} = [X_{n+1} \quad 0 \quad 0] \\ \tilde{\Theta}_{(n+1)} = \tilde{\Theta}_{(n)}\tilde{\theta}^{(n+1)}\,,\end{array}$$

for some J-unitary matrix $\tilde{\theta}^{(n+1)}$ acting on the columns of X_n and on h_{n+1}. Hence, we can continue with the result of the factorization that was obtained at the previous step. Each update requires about $1/2m^2$ rotations.

The downdating problem is to compute the factorization for H_n with its first column h_1 removed, from a factorization of H_n. It can be converted to an updating problem, where the old column h_1 is now introduced with a positive signature,

$$[\overset{+/-}{X_n} \quad \overset{+}{h_1}]\,\tilde{\theta}^{(n+1)} = [X_{n+1} \;\; 0]\,.$$

This is possible because, implicitly, we factor $\varepsilon^2 I - H_n H_n^* + h_1 h_1^* = X_n \tilde{J} X_n^* + h_1 h_1^*$. The uniqueness of the hyperbolic QR factorization into triangular matrices with positive diagonals ([24], viz. corollary 2.2) implies that the result X_{n+1} is precisely the same as if h_1 had never been part of H_n at all.

4.4. Breakdown. In section 4.2, we had to assume that the data matrix H was such that at no point in the algorithm $[a \;\; b]\,\tilde{j}_1[a \;\; b]^*$ is equal to zero. If the expression is zero, then there is no J-unitary rotation $\tilde{\theta}$ such that $[a \;\; b]\tilde{\theta} = [* \;\; 0]$. Note that the condition in theorem 3.1 that none of the singular values of H are equal to ε does not preclude this case, but merely ascertains that there *exists* a $\tilde{\Theta}$ which will zero H. One simple example is obtained by taking $H = [1 \;\; 1]^T$, $\varepsilon = 1$. It is straightforward to show that there is no J-unitary $\tilde{\Theta}$ such that

$$\left[\begin{array}{cc|c} 1 & & 1 \\ & 1 & 1 \end{array}\right]\tilde{\Theta} = \left[\begin{array}{cc|c} \times & 0 & 0 \\ \times & \times & 0 \end{array}\right] \tag{24}$$

as the J-norms of the first row will not be equal. Hence Θ cannot be obtained by the recursive algorithm. However, a more general $\tilde{\Theta}$ does exist, such that

$$\left[\begin{array}{cc|c} \overset{+}{1} & \overset{+}{} & \overset{-}{1} \\ & 1 & 1 \end{array}\right]\tilde{\Theta} = \frac{1}{\sqrt{2}}\left[\begin{array}{cc|c} \overset{+}{1} & \overset{-}{1} & \overset{+}{0} \\ -1 & 1 & 0 \end{array}\right]$$

viz.

$$\tilde{\Theta} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 & \sqrt{2} \\ -1 & -1 & \sqrt{2} \\ 0 & 2 & -\sqrt{2} \end{bmatrix}, \qquad \tilde{J}_1 = \begin{bmatrix} 1 & & \\ & 1 & \\ & & -1 \end{bmatrix}, \qquad \tilde{J}_2 = \begin{bmatrix} 1 & & \\ & -1 & \\ & & 1 \end{bmatrix}.$$

The difference is that, in this factorization, the resulting matrix X is no longer lower triangular. Theorem 4.1 gives necessary and sufficient conditions on the singular values of H and a collection of submatrices of H, so that the Schur algorithm does not break down.

THEOREM 4.1. *Let $H : m \times n$ be a given matrix, and $\varepsilon \geq 0$. Denote by $H_{[i,k]}$ the submatrix, consisting of the first to the i-th row and the first k columns of H. The Schur algorithm does not break down if and only if none of the singular values of $H_{[i,k]}$ is equal to ε, for $i = 1, \ldots, m$ and $k = 1, \ldots, n$.*

Proof. (Necessity) When processing the k-th column of H by the Schur algorithm, we are in fact computing a triangular factorization of $[\varepsilon I_m \;\; H_{[m,k]}]$. Corollary 2.2 claims that a

suitable J-unitary operator exists if and only if $[\varepsilon I_i \;\; H_{[i,k]}]$ is J-nonsingular, for $i = 1, \ldots, m$, *i.e.*, if and only if none of the singular values of $H_{[i,k]}$ is equal to 1. The triangularization is done for $k = 1, 2, \ldots, n$ in turn.

(Sufficiency) Sufficiency at stage (i,k) follows recursively from the factorization at the previous stage and the existence and uniqueness of the factorization at the current stage. □

Similar results are known for the case where the factorization is computed via hyperbolic Householder transformations where all zeros in a row are generated at the same time. In this case there are less conditions [24], viz. theorem 2.2. It should be noted that the conditions in theorem 4.1 are quite elaborate, as only one condition (none of the singular values of H are equal to ε) suffices for the *existence* of Θ. Numerically, we might run into problems also if one of the singular values is close to ε, in which case the corresponding hyperbolic rotation has a large norm. How serious this is depends on a number of factors, and a careful numerical analysis is called for. One example where a large rotation is not fatal is the case where the singularity occurs while processing the last entry of a column ($i = m$). Although the rotation will be very large, the resulting X remains bounded and becomes singular: $X_{m,m} = 0$. Hence, the subspace information is still accurate, and X varies in a continuous way across the ε-boundary; only its signature is necessarily discontinuous. Pivoting schemes can be used to prevent large rotations, and are discussed in the next subsection.

4.5. Pivoting schemes. Because a breakdown occurs only for special values of the entries of H, we can in almost all cases employ a simple pivoting operation to avoid a large hyperbolic rotation. If such a rotation occurs at the zeroing of entry $h_{i,k}$, then the matrix $H_{[i,k]}$ has a singular value close to ε. At this point, there are a number of remedies, based on the relative freedom in the order in which zero entries are created. The simplest solution is to permute the current column with the next one, which is possible if $k < n$. We can also permute the i-th row with the $i+1$-st, if $i < m$. Instead of permutations, other, more complicated operations are also possible, such as plane rotations of two columns or rows. Finally, if $(i,k) = (m,n)$, *i.e.*, $h_{i,k}$ is the last entry to be zeroed, then H has a singular value equal to ε and there is no remedy: there is no bounded Θ. However, because it is the last rotation, X will still be bounded, but it becomes singular.

A column permutation at stage (i,k) swaps the k-th column of H with the $k+1$-st, and also swaps the corresponding rows of $\tilde{\Theta}$. Before the permutation is done, the first $i-1$ entries of h_{k+1} have to be made zero. Hence, a column permutation scheme is most easily implemented when entries of H are zeroed row by row, rather than column by column as in the algorithmic description in section 4.2. Note that it is already sufficient to create zero entries of H in an anti-diagonal fashion. This is what actually happens in a systolic array implementation, where zeros on anti-diagonals of H are created in parallel. Hence, a column pivoting scheme can be readily implemented on such an array, with only one extra buffer required at each processor (to queue entries of a second column), but without sacrificing the systolic nature of the algorithm in any sense. In column permutation schemes, X stays upper triangular and, after processing of both h_k and h_{k+1}, is the same as it would be without pivoting. $\tilde{\Theta}$ is, of course, different: unbounded in the first case, bounded in the second.

Row permutations are necessary, *e.g.*, if there is no next column ($k = n$), or if columns are to be processed one at a time. It is required that the first $k-1$ entries of the $i+1$-st row of H have already been zeroed before permuting these rows. This is automatically the case if columns are processed one by one, or requires one rotation if we use an anti-diagonal zeroing scheme. Another rotation is needed to keep X lower triangular after the permutation has been performed. This makes row pivoting computationally more expensive. We also have to keep

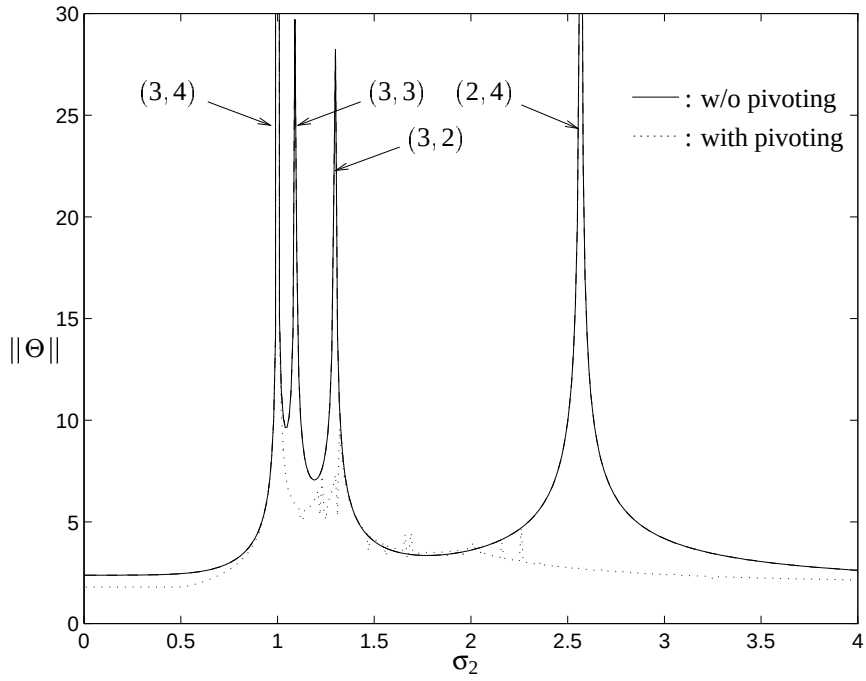

FIG. 2. *Norm of* Θ*. With pivoting,* $\|\Theta\| \to \infty$ *for certain values of* σ_2 *when the indicated entry* (i,j) *of* H *is processed. With pivoting, this only occurs when* $\sigma_2 = 1$*.*

track of the permutations: we are now in fact computing a factorization

$$\Pi[\varepsilon I \quad H]\tilde{\Theta} = [X \quad 0] \quad \Leftrightarrow \quad [\varepsilon I \quad H]\tilde{\Theta} \begin{array}{l} = \\ = \end{array} \begin{array}{l} [\Pi^* X \quad 0] \\ [X' \quad 0]\,. \end{array}$$

X is lower triangular, but the resulting X' in general not. It is possible to use any other invertible transformation of the rows instead of a permutation, such as for example a unitary plain rotation. This more general approach was suggested in [29], and provides a solution even in the special cases where permutations do not lead to bounded results, such as *e.g.*, in the case of equation (24). The resulting factorization can be viewed as a Hyperbolic URV decomposition. The added generality allows to reduce the number of hyperbolic rotations to one or two per column update, and leads to stable numerical implementations. (A discussion of this is relegated to future publications.)

5. Simulation results. In this section, we demonstrate some of the properties of the approximation scheme by means of a simple example. We take $H(\sigma_2) = U\Sigma(\sigma_2)V^*$ to be a sequence of 3×4 matrices, with U and V randomly selected constant unitary matrices, and with singular values equal to

$$(20, \sigma_2, 0.5)\,, \qquad \sigma_2 = 0, 0.01, \ldots, 3.99, 4\,.$$

The approximation tolerance is set to $\varepsilon = 1$. We compare the approximants $\hat{H}^{(0)}$ given by $S_L = 0$, $\hat{H}^{(1)}$ given by equation (18), $\hat{H}^{(2)}$ given by (21), and $\hat{H}^{(1)}$ when the factorization is computed with pivoting. The pivoting scheme consists of column permutations, except when processing the last column, in which case we switch to row permutations. The pivoting is applied in its extreme form, *i.e.*, whenever this leads to elementary rotation matrices with a

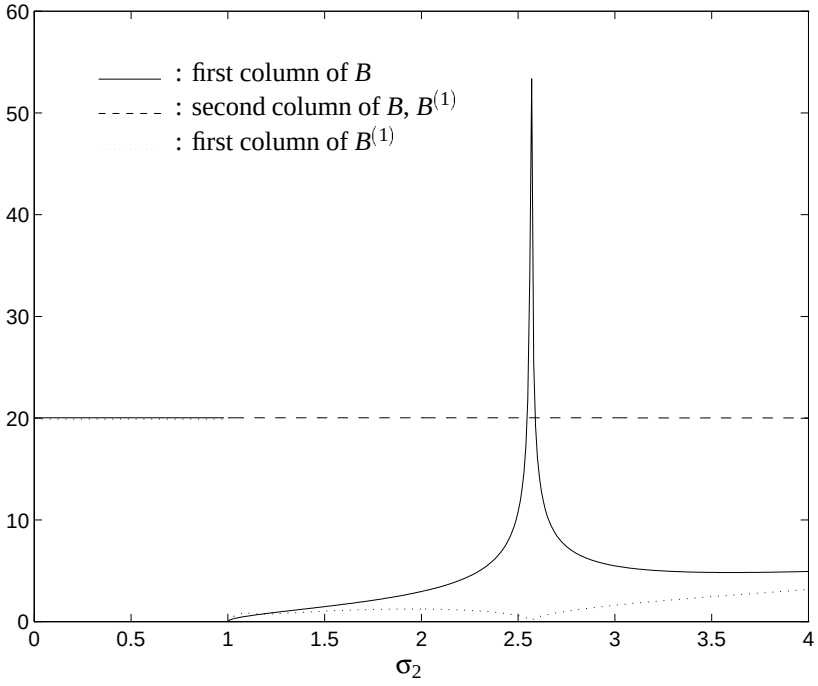

FIG. 3. *The norm of the first and second column of B and $B^{(1)}$ (no pivoting).*

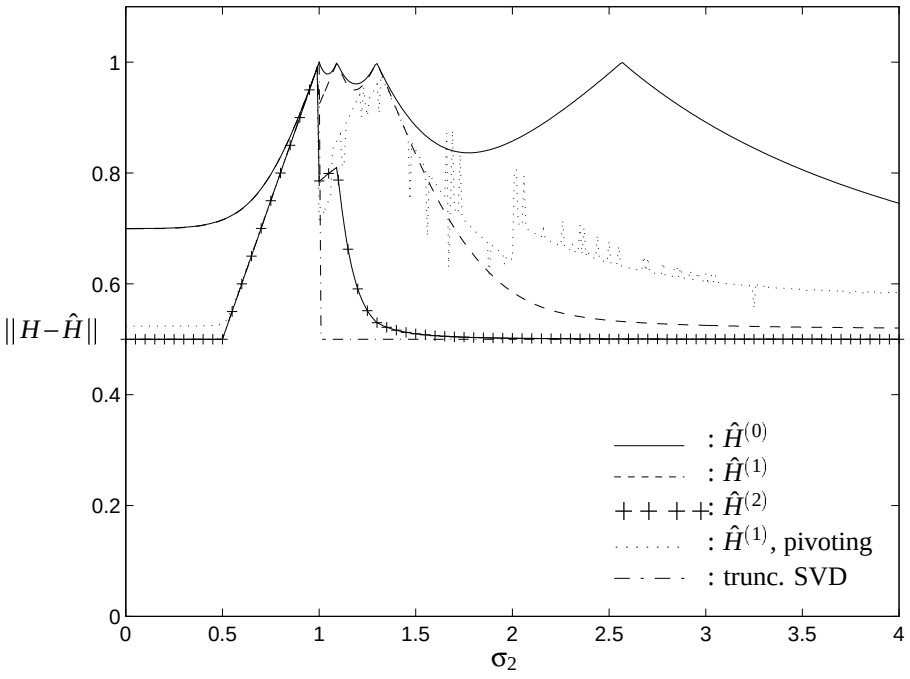

FIG. 4. *Norm of the approximation error.*

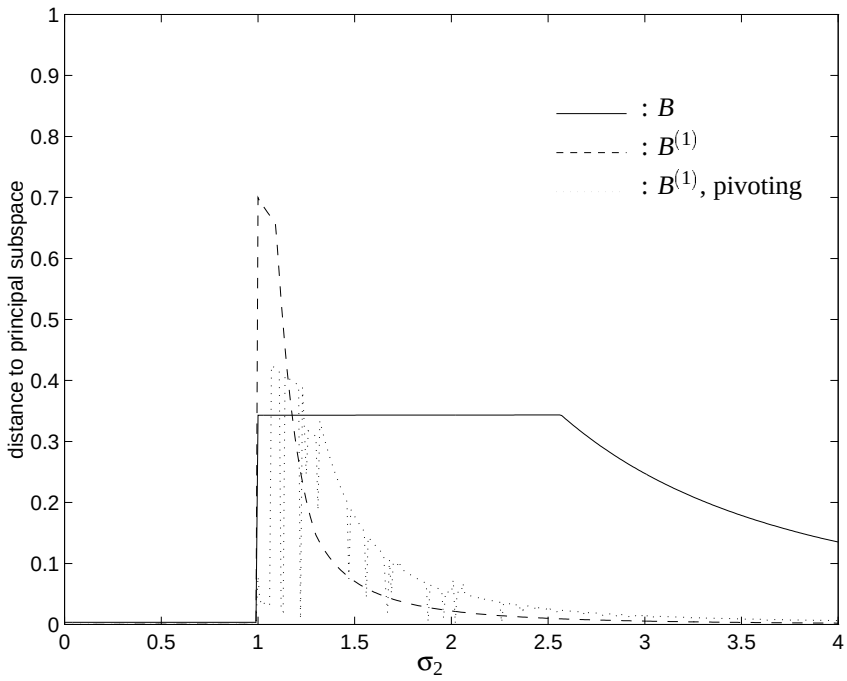

FIG. 5. *Distance between the principal and estimated subspaces.*

smaller norm. The approximants are compared on the following aspects: (*a*) $||\Theta||$, with and without pivoting; (*b*) $||H-\hat{H}||$, for each of the mentioned approximants; (*c*) the accuracy of the subspace estimates, compared to the principal subspace of H (the column span of the singular vectors with corresponding singular values larger than 1). The distance between two subspaces $\mathcal{A}$ and $\mathcal{B}$ is defined as $\text{dist}(\mathcal{A},\mathcal{B}) = ||\mathbf{P}_{\mathcal{A}}-\mathbf{P}_{\mathcal{B}}||$, where $\mathbf{P}_{\mathcal{A}}$ is the orthogonal projection onto $\mathcal{A}$ [2].

Figure 2 shows $||\Theta||$ as a function of σ_2. Without pivoting, there are a number of peaks, corresponding to the values of σ_2 where one of the submatrices $H_{[i,k]}$ has a singular value equal to 1. In the range $0 \le \sigma_2 \le 4$, this occurred for $(i,k) = (3,4)$, $(3,3)$, $(3,2)$ and $(2,4)$, respectively. When pivoting is applied, the peak at $\sigma_2 = 1$ is, necessarily, still present, but the other peaks are mostly smoothed out. Figure 3 shows the norm of the columns of B, in the scheme without pivoting. For $\sigma_2 < 1$, the rank of the approximant is 1. At $\sigma_2 = 1$, the dimension of B increases, although at first, the new column has a very small norm. For larger values of σ_2, the norm grows and the subspace becomes better defined. There is a peak at the point where $H_{[2,4]}$ has a singular value equal to 1; this peak can be removed by row pivoting but not by column pivoting. There are no peaks when $H_{(i,j)}$ has a singular value equal to 1 and $i = m$, because X becomes singular rather than unbounded when a singularity occurs at the last entry of a column. Figure 3 also shows that no peak occurs for the norm of the columns of the 'improved' subspace estimate $B^{(1)}$ of equation (19), on which both $\hat{H}^{(1)}$ and $\hat{H}^{(2)}$ are based. This is as predicted by lemma 3.4: $||B^{(1)}|| \le ||H|| = 20$. Instead of having a peak, the norm of the first column of $B^{(1)}$ dips to about 0.12.

In figure 4, the norm of $H-\hat{H}$ is shown, for the various choices of $\hat{H}$ that we discussed in section 3.2. The lowest line corresponds to the truncated SVD solution, which gives the lowest attainable error. It is seen that, for all approximants, the approximation error is always less than $\varepsilon \equiv 1$. Of the non-pivoted schemes, the approximation error for $\hat{H}^{(0)}$ is always higher than the

error for $\hat{H}^{(1)}$, $\hat{H}^{(2)}$ (but there is no *proof* that this is necessarily always the case), and the error for $\hat{H}^{(1)}$ is always higher than the error for $\hat{H}^{(2)}$, since the latter approximant minimizes this error while retaining the same subspace estimate. The approximation error for $\hat{H}^{(2)}$ is almost identically close to the theoretical minimum, except in a small region $1 \leq \sigma_2 \leq 1.5$. The errors for $\hat{H}^{(0)}$ and $\hat{H}^{(1)}$ touch a number of times on the $(\varepsilon = 1)$-line. For $\hat{H}^{(0)}$ this can be explained as follows. The error for $S_L = 0$ is given by equation (16) as $-\varepsilon\Theta_{12}\Theta_{22}^{-1}$. Because the J-unitarity of Θ implies $\Theta_{22}^{-*}\Theta_{22}^{-1} + (\Theta_{22}^{-*}\Theta_{12}^{*})(\Theta_{12}\Theta_{22}^{-1}) = I$ (viz. (6)), it follows that whenever $\| \Theta_{22} \| \rightarrow \infty$, necessarily $\| \Theta_{12}\Theta_{22}^{-1} \| \rightarrow 1$. The analysis of $\| H - \hat{H}^{(1)} \|$ from (17) is more involved and omitted at this point.

Figure 5 depicts the distance between the principal and estimated subspaces. For $\sigma_2 < 1$, this distance is very close to zero ($< .0002$) for each of the methods. The distance jumps up when σ_2 crosses 1: the subspace increases in dimension but is at first only weakly defined. For $B^{(1)}$, the distance goes down again quickly, whereas for B, it stays constant for a while before going down.

6. Conclusions. We have derived a general formula which describes all rank-d 2-norm approximants of a given matrix H. The formula relies on a factorization which exists if none of the singular values of H is equal to ε, and which can be computed by a Schur-type algorithm if additional singular value conditions are satisfied. Updating and downdating is straightforward, and the algorithm is amenable to parallel implementation. It is highly suitable for adaptive subspace estimation: some of these approximants are quite close to the truncated SVD solution (as shown by a numerical experiment), but much easier to compute. Such an application is reported in [34]. Another application is the regularization of ill-conditioned total least squares problems [35], *cf.* [36].

There are several open problems and remaining issues. Apart from the listed approximants, there might be other interesting choices, such as approximants that by construction have all their singular values larger than ε. There are applications in which an on-line computation of the approximant is required, or of its last column, instead of only its column space: an integral scheme for doing this would be interesting. As a final remark, we would like to mention that while this paper was in review, an updating version for the 'improved' approximant $B^{(1)}$ has been obtained: an orthonormal basis for this subspace can be updated using about twice the number of operations as the basic Schur updating algorithm, without the need for pivoting, and keeping the number of hyperbolic rotations as small as possible. This will be reported elsewhere.

Acknowledgements. The author wishes to extend his warm feelings of gratitude to Prof. P. Dewilde at Delft University of Technology. The approximation theory underlying the results presented in this paper stems from research in time-varying systems theory carried out in close collaboration over a number of years. The observation that the time-varying Hankel-norm approximation theory can be applied to matrices, leading to the present paper, was made by J. Götze, Technical University of Munich, while visiting Delft University in the autumn of 1993. The kind invitation by Prof. G.H. Golub for a stay at Stanford University is gratefully acknowledged.

A. Appendix: Proof of theorem 3.3. The proof of theorem 3.3 consists of two propositions. The first shows that any S_L which satisfies the constraints gives rise to a valid approximant, and the second proves the converse. Without loss of generality, we take $\varepsilon = 1$.

PROPOSITION A.1. *Let $H : m \times n$ be a given matrix, with d singular values larger than 1 and none equal to 1, and let S_L be a given matrix satisfying conditions (15) (i) and (ii). Define Θ, A', B' as in equation (12). Put*

$$S = (\Theta_{11}S_L - \Theta_{12})(\Theta_{22} - \Theta_{21}S_L)^{-1}.$$

Then S is well defined, and $\hat{H} := H - S$ is a 2-norm approximant of rank equal to d, satisfying

$$\hat{H} \;=\; (B' - A'S_L)(\Theta_{22} - \Theta_{21}S_L)^{-1}\,.$$

Proof. Let

$$\begin{bmatrix} -G_1 \\ G_2 \end{bmatrix} = \begin{bmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{bmatrix} \begin{bmatrix} -S_L \\ I \end{bmatrix}.$$

Then

$$\begin{aligned} G_1 &= \Theta_{11}S_L - \Theta_{12} \\ G_2 &= -\Theta_{21}S_L + \Theta_{22} \;=\; \Theta_{22}(I - \Theta_{22}^{-1}\Theta_{21}S_L)\,. \end{aligned}$$

Because $\|\,\Theta_{22}^{-1}\Theta_{21}\,\| < 1$ and $\|\,S_L\,\| \le 1$, G_2 is invertible, and hence $S = G_1G_2^{-1}$. The J-unitarity of Θ implies $S_L^*S_L - I = G_1^*G_1 - G_2^*G_2 = G_2^*(S^*S - I)G_2$. Since G_2 is invertible, and $S_L^*S_L - I$ is negative semidefinite, it follows that the same holds for $S^*S - I$. Hence S is contractive: $\|\,S\,\| \le 1$, and $\hat{H}$ is a 2-norm approximant of H. To derive the alternate formula for $\hat{H}$ and show that it has rank d, write

$$\begin{aligned} \hat{H} = H - S &= \begin{bmatrix} I & H \end{bmatrix} \begin{bmatrix} -S \\ I \end{bmatrix} \\ &= \begin{bmatrix} I & H \end{bmatrix} \Theta \begin{bmatrix} -S_L \\ I \end{bmatrix} (\Theta_{22} - \Theta_{21}S_L)^{-1} \\ &= \begin{bmatrix} A' & B' \end{bmatrix} \begin{bmatrix} -S_L \\ I \end{bmatrix} (\Theta_{22} - \Theta_{21}S_L)^{-1}\,. \end{aligned}$$

Hence $\hat{H} = (B' - A'S_L)(\Theta_{22} - \Theta_{21}S_L)^{-1}$. The rank of $\hat{H}$ is equal to the rank of $B' - A'S_L$. In this expression, $B' = [B\ 0]$ is of full column rank d, and $A' = [A\ 0]$, where A is of full column rank $m - d$. Because $(S_L)_{12} = 0$, it follows that $A'S_L = [A\ 0]S_L = [A(S_L)_{11}\ \ 0]$ is of rank less than or equal to d, too. Finally, $B' - A'S_L$ is precisely of rank d because the columns of A are linearly independent of the columns of B. ∎

PROPOSITION A.2. *Let $H : m \times n$ be a given matrix, with d singular values larger than 1 and none equal to 1. Define Θ, A', B' as in equation (12). Suppose that a matrix $\hat{H}$ satisfies*

$$\begin{aligned} &(a) \quad \|\,H - \hat{H}\,\| \le 1\,, \\ &(b) \quad \operatorname{rank}(\hat{H}) \le d\,. \end{aligned}$$

Then $\operatorname{rank}(\hat{H}) = d$, *and $\hat{H} = H - S$ where*

$$S = (\Theta_{11}S_L - \Theta_{12})(\Theta_{22} - \Theta_{21}S_L)^{-1}\,, \tag{25}$$

for some contractive S_L with $(S_L)_{12} = 0$.

Proof. It follows directly that S is contractive. Define matrices G_1, G_2 by

$$\begin{bmatrix} -S \\ I \end{bmatrix} \;=\; \Theta \begin{bmatrix} -G_1 \\ G_2 \end{bmatrix} \qquad \Leftrightarrow \qquad \begin{bmatrix} -G_1 \\ G_2 \end{bmatrix} = \Theta^{-1} \begin{bmatrix} -S \\ I \end{bmatrix}. \tag{26}$$

As in the proof of proposition A.1, it follows that G_2 is invertible. The J-unitarity of Θ and the contractiveness of S implies $G_1^*G_1 \le G_2^*G_2$. Hence $S_L := G_1G_2^{-1}$ is well defined and contractive, and (26) implies (25). The remaining part of the proof is technical and shows that $(S_L)_{12} = 0$. First, we define the partitionings

$$G_1 = \begin{matrix} m-d \\ d \end{matrix} \begin{bmatrix} G_{11} \\ G_{12} \end{bmatrix}, \qquad G_2 = \begin{matrix} d \\ n-d \end{matrix} \begin{bmatrix} G_{21} \\ G_{22} \end{bmatrix}, \qquad G_2^{-1} = \begin{matrix} d & n-d \end{matrix} \begin{bmatrix} (G_2^{-1})_1 & (G_2^{-1})_2 \end{bmatrix},$$

which are conform the partitionings of A' and B'. Then $(S_L)_{12} = 0 \Leftrightarrow G_{11}(G_2^{-1})_2 = 0$. To prove that, indeed, $G_{11}(G_2^{-1})_2 = 0$, we look at $[G_1^* \ G_2^*]$. Use of (26) and $\Theta^{-1} = J\Theta^* J$ gives

$$[G_1^* \ G_2^*] = [S^* \ I]\Theta = [-\hat{H}^* \ 0]\Theta + [H^* \ I]\Theta . \tag{27}$$

We also have

$$\begin{aligned}
& [I \quad H] = [A' \quad B']\Theta^{-1} \\
\Leftrightarrow \quad & \begin{bmatrix} I \\ H^* \end{bmatrix} = \Theta^{-*} \begin{bmatrix} A'^* \\ B'^* \end{bmatrix} = J\Theta J \begin{bmatrix} A'^* \\ B'^* \end{bmatrix} \\
\Rightarrow \quad & 0 = [H^* \ I_n] J \begin{bmatrix} I \\ H^* \end{bmatrix} = [H^* \ I_n]\Theta J \begin{bmatrix} A'^* \\ B'^* \end{bmatrix} \\
\Rightarrow \quad & [H^* \ I_n]\Theta = \begin{bmatrix} (0_{n\times(m-d)} & *) & (0_{n\times d} & *) \end{bmatrix} ,
\end{aligned}$$

where '$*$' stands for some quantity whose precise value is not of interest. In the last step, we used the fact that $[A \ B]$ is of full rank. Inserting this result in (27) shows that

$$\mathcal{R}(G_{11}^*) \subset \mathcal{R}(\hat{H}^*) , \qquad \mathcal{R}(G_{21}^*) \subset \mathcal{R}(\hat{H}^*) .$$

G_2 is invertible, hence $\mathcal{R}(G_{21}^*)$ is of full dimension d. Since the rank of $\hat{H}$ is less than or equal to d, it follows that the rank of $\hat{H}$ is precisely equal to d, and that actually $\mathcal{R}(G_{21}^*) = \mathcal{R}(\hat{H}^*)$. This implies $\mathcal{R}(G_{11}^*) \subset \mathcal{R}(G_{21}^*)$, so that there is some matrix M such that $G_{11} = MG_{21}$. Hence

$$\begin{aligned}
& G_2(G_2)^{-1} = I \\
\Leftrightarrow \quad & \begin{bmatrix} G_{21} \\ G_{22} \end{bmatrix} [(G_2^{-1})_1 \quad (G_2^{-1})_2] = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \\
\Rightarrow \quad & G_{21}(G_2^{-1})_2 = 0 \\
\Rightarrow \quad & G_{11}(G_2^{-1})_2 = MG_{21}(G_2^{-1})_2 = 0 .
\end{aligned}$$

□

REFERENCES

[1] G.W. STEWART, *An updating algorithm for subspace tracking*, IEEE Trans. Signal Processing, 40 (1992), pp. 1535–1541.

[2] G. GOLUB AND C.F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, 1989.

[3] L.V. FOSTER, *Rank and null space calculations using matrix decomposition without column interchanges*, Lin. Alg. Appl., 74 (1986), pp. 47–71.

[4] T.F. CHAN, *Rank revealing QR factorizations*, Lin. Alg. Appl., 88/89 (1987), pp. 67–82.

[5] T.F. CHAN AND P.C. HANSEN, *Computing truncated singular value decomposition least squares solutions by rank revealing QR-factorizations*, SIAM J. Sci. Stat. Comput., 11 (1990), pp. 519–530.

[6] ———, *Some applications of the rank revealing QR-factorization*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 727–741.

[7] C.H. BISCHOF AND G.M. SHROFF, *On updating signal subspaces*, IEEE Trans. Signal Processing, 40 (1992), pp. 96–105.

[8] S. CHANDRASEKARAN AND I.C.F. IPSEN, *On rank-revealing factorisations*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 592–622.

[9] J.R. BUNCH AND C.P. NIELSEN, *Updating the singular value decomposition*, Numerische Mathematik, 31 (1978), pp. 111–129.

[10] M. MOONEN, P. VAN DOOREN, AND J. VANDEWALLE, *An SVD updating algorithm for subspace tracking*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 1015–1038.

[11] M. MOONEN, P. VAN DOOREN, AND F. VANPOUCKE, *On the QR algorithm and updating the SVD and URV decomposition in parallel*, Lin. Alg. Appl., 188/189 (1993), pp. 549–568.

[12] W.M. GENTLEMAN AND H.T. KUNG, *Matrix triangularization by systolic arrays*, Proc. SPIE, Real Time Signal Proc. IV, 298 (1981), pp. 19–26.

[13] I. SCHUR, *Uber Potenzreihen, die im Innern des Einheitskreises beschränkt sind, I*, J. Reine Angew. Math., 147 (1917), pp. 205–232. Eng. Transl. *Operator Theory: Adv. Appl.*, vol. 18, pp. 31-59, Birkhäuser Verlag, 1986.

[14] H. DYM, *J-Contractive Matrix Functions, Reproducing Kernel Hilbert Spaces and Interpolation*, no. 71 in CBMS reg. conf. ser., American Math. Soc., Providence, 1989.

[15] J.A. BALL, I. GOHBERG, AND L. RODMAN, *Interpolation of Rational Matrix Functions*, vol. 45 of Operator Theory: Advances and Applications, Birkhäuser Verlag, 1990.

[16] V.M. ADAMJAN, D.Z. AROV, AND M.G. KREIN, *Analytic properties of Schmidt pairs for a Hankel operator and the generalized Schur-Takagi problem*, Math. USSR Sbornik, 15 (1971), pp. 31–73. (transl. of *Iz. Akad. Nauk Armjan. SSR Ser. Mat. 6 (1971)*).

[17] D. PAL AND T. KAILATH, *Fast triangular factorization and inversion of Hermitian, Toeplitz, and related matrices with arbitrary rank profiles*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 1016–1042.

[18] P. DEWILDE AND E. DEPRETTERE, *The generalized Schur algorithm: Approximation and hierarchy*, in Operator Theory: Advances and Applications, vol. 29, Birkhäuser Verlag, 1988, pp. 97–116.

[19] K. DIEPOLD AND R. PAULI, *Schur parametrization of symmetric indefinite matrices based on a network theoretic model*, Archiv für Elektronik und Übertragungstechnik AEÜ, 44 (1990), pp. 92–96.

[20] ———, *A recursive algorithm for lossless embedding of non-passive systems*, in Challenges of a Generalized System Theory, P. Dewilde e.a., ed., Essays of the Royal Dutch Academy of Sciences, North-Holland, Amsterdam, The Netherlands, 1993.

[21] PH. DELSARTE, Y. GENIN, AND Y. KAMP, *A method of matrix inverse triangularization, based on contiguous principal submatrices*, Lin. Alg. Appl., 31 (1980), pp. 199–212.

[22] J.-M. DELOSME AND I.C.F. IPSEN, *Parallel solution of symmetric positive definite systems with hyperbolic rotations*, Lin. Alg. Appl., 77 (1986), pp. 75–111.

[23] A.W. BOJANCZYK, R.P. BRENT, P. VAN DOOREN, AND F.R. DE HOOG, *A note on downdating the Cholesky factorization*, SIAM J. Sci. Stat. Comp., 8 (1987), pp. 210–221.

[24] A. BUNSE-GERSTNER, *An analysis of the HR algorithm for computing the eigenvalues of a matrix*, Lin. Alg. Appl., 35 (1981), pp. 155–173.

[25] C.M. RADER AND A.O. STEINHARDT, *Hyperbolic Householder transformations*, IEEE Trans. Acoust. Speech, Signal Proc., 34 (1986), pp. 1589–1602.

[26] S.T. ALEXANDER, C.-T. PAN, AND R.J. PLEMMONS, *Analysis of a recursive least squares hyperbolic rotation algorithm for signal processing*, Lin. Alg. Appl., 98 (1988), pp. 3–40.

[27] G. CYBENKO AND M. BERRY, *Hyperbolic Householder algorithms for factoring structured matrices*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 499–520.

[28] C.H. BISCHOF, C.-T. PAN, AND P.T.P. TANG, *A Cholesky up- and downdating algorithm for systolic and SIMD architectures*, SIAM J. Sci. Stat. Comput., 14 (1993), pp. 670–676.

[29] R. ONN, A.O. STEINHARDT, AND A.W. BOJANCZYK, *The hyperbolic singular value decomposition and applications*, IEEE Trans. Signal Proc., 39 (1991), pp. 1575–1588.

[30] P.M. DEWILDE AND A.J. VAN DER VEEN, *On the Hankel-norm approximation of upper-triangular operators and matrices*, Integral Eq. Operator Th., 17 (1993), pp. 1–45.

[31] A.J. VAN DER VEEN AND P.M. DEWILDE, *On low-complexity approximation of matrices*, Linear Algebra and its Applications, 205/206 (1994), pp. 1145–1202.

[32] K. GLOVER, *All optimal Hankel norm approximations of linear multi-variable systems and their L^∞-error bounds*, Int. J. Control, 39 (1984), pp. 1115–1193.

[33] D.J.N. LIMEBEER AND M. GREEN, *Parametric interpolation, H_∞-control and model reduction*, Int. J. Control, 52 (1990), pp. 293–318.

[34] J. GÖTZE AND A.J. VAN DER VEEN, *On-line subspace estimation using a Schur-type method*, to appear in IEEE Trans. Signal Proc., (1994).

[35] A.J. VAN DER VEEN, *Schur method for low-rank matrix approximation*, in Proc. SPIE, "Advanced Signal Processing Algorithms, Architectures, and Implementations V", vol. 2296, San Diego, CA, July 1994.

[36] R.D. FIERRO, G.H. GOLUB, P.C. HANSEN, AND D.P. O'LEARY, *Regularization by truncated total least squares*, in Proc. 5-th SIAM Conf. on Applied Lin. Alg., J.G. Lewis, ed., Snowbird, UT, June 1994, pp. 250–254.