Technische Universität Chemnitz-Zwickau

Sonderforschungsbereich 393

Numerische Simulation auf massiv parallelen Rechnern

Brahim Benhammouda

RANK-REVEALING "TOP-DOWN" ULV FACTORIZATIONS

Preprint SFB393/97-02

Key words: ULV and URV factorizations, Orthogonal factorizations, Rank-revealing factorizations, Numerical rank, Differential-algebraic equations

AMS subject classifications: 65F25, 15A03, 65F35.

Author's address:

Brahim Benhammouda
Fakultät für Mathematik
TU Chemnitz-Zwickau
D-09107 Chemnitz, FRG
e-mail: brahim@mathematik.tu-chemnitz.de

This work has been supported by the Deutsche Forschungsgemeinschaft under the grant no. Me 790/5-2 "Differentiell-algebraische Gleichungen".

Preprint-Reihe des Chemnitzer SFB 393

SFB393/97-02 **January 1997**

Abstract

Rank-revealing ULV and URV factorizations are useful tools to determine the rank and to compute bases for null-spaces of a matrix. However, in the practical ULV (resp. URV) factorization each left (resp. right) null vector is recomputed from its corresponding right (resp. left) null vector via triangular solves. Triangular solves are required at initial factorization, refinement and updating. As a result, algorithms based on these factorizations may be expensive, especially on parallel computers where triangular solves are expensive. In this paper we propose an alternative approach. Our new rank-revealing ULV factorization, which we call "top-down" ULV factorization ($TDULV$-factorization) is based on right null vectors of lower triangular matrices and therefore no triangular solves are required. Right null vectors are easy to estimate accurately using condition estimators such as incremental condition estimator (ICE). The $TDULV$ factorization is shown to be equivalent to the URV factorization with the advantage of circumventing triangular solves.

1. Introduction. Recent numerical integration methods for differential-algebraic equations ($DAEs$) [17, 18, 19] require at each time integration step the computation of the numerical rank and bases for null-spaces of very large matrices. These matrices are obtained by a recursive differentiation algorithm which appends new rows to the previous matrices. The process of incorporating a new row or column in a matrix is called updating. Other applications are the solution of underdetermined rank-deficient least squares problems [12, 14, 20], subset selection problems [13, 14] and information retrieval [2].

The singular value factorization (SVD) [14, p. 246] is known to be an extremely reliable tool for computing the numerical rank and bases for the null-spaces of a matrix. However, the SVD is "too expensive" when it comes to recursive algorithms or real-time applications, since its computation requires $\mathcal{O}(n^3)$ flops[1] and the SVD is difficult to update [1, 6]. Therefore alternative algorithms that are nearly as accurate as the SVD, cheaper and easier to update are desired.

Recently, Stewart [25, 26, 27, 29] proposed two rank-revealing factorizations, called ULV and URV factorizations. These two factorizations are

[1]Here, a flop is either an addition or a multiplication

effective in exhibiting the numerical rank and bases for the null-spaces. The ULV and the URV factorizations can be updated in $\mathcal{O}(n^2)$ flops, sequentially and in $\mathcal{O}(n)$ flops on an array of n processors [26, 27]. Recent work related to the URV and ULV factorization both in theory and implementation may be found in [8, 9, 10, 11, 22, 23]. The rank-revealing ULV and the URV algorithms are iterative and require estimates of the condition number of some triangular submatrices at every iteration step of initial factorization, refinement and updating. In the URV and the ULV factorizations small singular values and associated null vectors are estimated by means of conditions estimators [3, 4, 5, 15, 24, 30]. A survey of condition estimators is given in [16].

In the practical ULV (resp. URV) factorization, however, each left (resp. right) null vector is recomputed from its corresponding right (resp. left) null vector via triangular solves. Triangular solves are required for the initial factorization, the refinement and updating. For some applications triangular solves have to be performed many times in order to achieve a required accuracy. Therefore algorithms based on the usual ULV and URV factorizations may be very expensive on parallel computers, where triangular solves are expensive.

For this reason we introduce an alternative rank-revealing ULV factorization, called "top-down" ULV factorization ($TDULV$-factorization). This new factorization relies on right null vectors of lower triangular matrices which are accurately estimated using condition estimators. This results in circumventing triangular solves required in the usual rank-revealing ULV and URV factorizations. Our $TDULV$ factorization is essentially equivalent to the URV with the advantage of avoiding triangular solves, thus it is more suitable for parallel implementations. Furthermore the $TDULV$ uses the null vectors obtained from condition estimators in a straithforward way.

In this paper we describe the $TDULV$ factorization, give an algorithm to compute it and show how this algorithm can be implemented, refined and updated efficiently. The remainder of this paper is organised as follows. In section 2, we briefly review the usual rank-revealing ULV and URV factorizations. Our new $TDULV$ factorization method is proposed in section 3. In section 4 we give details of the $TDULV$ factorization algorithm. The new algorithm is presented in section 5. Finally, we draw a conclusion in section 6.

2. ULV and URV factorizations. In this section we review the rank-revealing ULV and URV factorizations introduced by Stewart [25, 26, 27, 29]. We first introduce the concept of numerical rank of a matrix. Given a matrix $A \in \mathcal{R}^{m\times n}$ $(m \geq n)$ a singular value factorization (SVD) (see [14, § 2.5]) of A has the form

$$A = U\Sigma V^T, \tag{1}$$

where $U = [u_1, \cdots, u_m]$ and $V = [v_1, \cdots, v_m]$ are orthogonal matrices and $\Sigma = diag(\sigma_1, \cdots, \sigma_n)$ is an $m \times n$ diagonal matrix whose entries, the *singular values* of A, are ordered such that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$. Then the *numerical rank* of A with respect to a threshold $\epsilon > 0$ is defined as the number of singular values of A strictly larger than ϵ, *i.e.*,

$$\sigma_1 \geq \cdots \geq \sigma_k > \epsilon \geq \sigma_{k+1} \geq \cdots \geq \sigma_n. \tag{2}$$

ϵ is a threshold below which a singular value of the matrix A is declared to be numerically null or negligeable. The ratio σ_{r+1}/σ_k estimates the "gap" between "large" and "small" singular values of A. The numerical rank is well defined whenever the gap is sufficiently large. Ways for choosing the threshold ϵ may be found in [28].

For $i = k+1, \cdots, n$ the columns v_i of V satisfy $\|Av_i\| \leq \epsilon$ and therefore are called *numerical right null vectors* ($\|\cdot\|$ denotes the matrix 2-norm). In the same way columns $u_{k+1}, \cdots, u_n$ of U are called *numerical left null vectors*, since they satisfy $\|u_i^T A\| \leq \epsilon$ for $i = k+1, \cdots, n$.

The *numerical right null-space* of A is defined by

$$\mathcal{N}_k^r := \text{span}\{v_{k+1}, \ldots, v_n\}. \tag{3}$$

in the same way we define the *numerical left null-space* of A by

$$\mathcal{N}_k^l := \text{span}\{u_{k+1}, \ldots, u_n\}. \tag{4}$$

Given a matrix $A \in \mathcal{R}^{m\times n}$, a ULV [29] factorization of A has the form

$$A = U \begin{pmatrix} L_k & 0 \\ H & E \end{pmatrix} V^T, \tag{5}$$

with orthogonal matrices $U \in \mathcal{R}^{m\times m}$, $V \in \mathcal{R}^{n\times n}$ and $L_k \in \mathcal{R}^{k\times k}$, $E \in \mathcal{R}^{(m-k)\times(m-k)}$ lower triangular matrices, $H \in \mathcal{R}^{(m-k)\times k}$.

Such a factorization is said to be *rank-revealing* if $\|[H\ E]\| = \mathcal{O}(\sigma_{k+1})$ and L_k is well-conditioned, *i.e.*, $\sigma_k(L_k)/\sigma_1(L_k) \geq c$, where $c > 0$ is some given tolerance.

Similarly, a URV factorization [25, 29] of A has the form

$$A = U \begin{pmatrix} R_k & F \\ 0 & G \end{pmatrix} V^T, \tag{6}$$

where $R_k \in \mathcal{R}^{k \times k}$, $G \in \mathcal{R}^{(m-k)\times(n-k)}$ are upper triangular matrices and where $F \in \mathcal{R}^{k\times(n-k)}$.
Such a factorization is said to be a *rank-revealing* if R_k is well-conditioned and $\|[F^T\ G^T]^T\| = \mathcal{O}(\sigma_{k+1})$.

In factorizations (5) and (6) the numerical rank of A is revealed by the dimension of the submatrices L_k and R_k, respectively. Orthonormal left and right bases for the null-spaces of A are revealed by the matrix U and V, respectively. More precisely, columns $k+1$ through n of U and V span the left and right null-spaces of A, respectively.

Factorization (5) and (6) are based on estimating small singular values of the middle factors L and R and the associated left and right null vectors, respectively. Then deflating small singular values from the bottom of matrices L and R, factorizations (5) and (6) are obtained. Adaptive versions of the ULV and URV algorithms and results concerning the effect of estimated null vectors on the size of off-diagonal blocks H and F are discussed in [10]. There, it is shown that the sizes of H and F depend strongly on approximations of the null vectors. The norms of H and F in turn affect the accuracy of the approximated null-spaces. A refinement method for the URV factorization was presented and analysed in Stewart [25].

The usual way to compute a ULV factorization (5) of a matrix A is first to compute an ordinary QL factorization of A [31, p. 140] and then to "peel-off" small singular values one by one from the bottom of the matrix L [10, 27]. This requires approximations of left null vectors of the triangular matrix L at each iteration step of factorization, refinement and updating. In the practical rank-revealing ULV factorizations, left null vectors are usually obtained from the corresponding right null vectors via triangular solves. For very large problems, however, this results in an extra cost and may lead to loss of accuracy in the subspaces. In the next section we present a more efficient ULV factorization that avoids triangular solves by working with right null vectors of lower triangular matrices. This reduces the computational work needed for the triangular solves.

3. TDULV factorization. In this section we present the rank-revealing $TDULV$ factorization. The idea of our factorization is to compute first any QL factorization of A (for example by using the LAPACK routine xGEQLF[2]) then to "peel-off" small singular values of L one after the other from the top of the matrix L in a sequence of deflation steps until a large singular value is detected. This is achieved by estimating small singular values of L and associated right null vectors using condition estimators (for example by using the incremental condition estimator (ICE)[5] implemeted in LAPACK routine xLAIC1). This process leads to the so called "top-down" ULV factorization $TDULV$.

$$A = U \begin{pmatrix} E & 0 \\ H & L_k \end{pmatrix} V^T, \tag{7}$$

where $L_k \in \mathcal{R}^{k \times k}$, $E \in \mathcal{R}^{(m-k)\times(n-k)}$ are lower triangular matrices and where $H \in \mathcal{R}^{k \times (n-k)}$.
We call such a factorization rank-revealing $TDULV$ factorization if L_k is well-conditioned and $\|[E^T \; H^T]^T\| = \mathcal{O}(\sigma_{k+1})$.
In the $TDULV$ factorization the rank of the matrix A is revealed by the dimension of the right bottom submatrix L_k. The first $n-k$ columns of the orthogonal matrices U and V furnish orthonormal left and right bases for null-spaces of A, respectively.

We show in the appendix that the rank-revealing $TDULV$ factorization (7) and the rank-revealing URV factorization (6) are mathematically equivalent. The advantage of the $TDULV$ over the URV is that the $TDULV$ works with singular vectors computed by condition estimators in a straithfoward way.

4. Outline of the rank-revealing TDULV Algorithm. In this section, we discuss the implementation of the $TDULV$ algorithm. We show how to refine the factorization to make it rank-revealing. We then discuss the updating of the factorization. The rank-revealing $TDULV$ factorization process

[2]Here, the prefix x is S or D

begins with any QL factorization of A followed by an iteration with makes the factor L rank-revealing. The matrix L is declared to be numerically rank deficient with respect to a threshold ϵ if L has at least one singular value $\sigma \leq \epsilon$. Small singular values σ of L and associated null vectors v of norm one are estimated efficiently using condition estimators. If the matrix L is rank deficient then we transform it to an equivalent lower triangular matrix $P^T LQ$ by means of Givens rotations. The orthogonal matrix Q is formed as the product of Givens rotations such that components of v are annihilated one at a time to obtain the canonical unit vector e_1, *i.e.* we have $Q^T v = e_1$. We postmultiply L by the orthogonal matrix Q. Then we triangularize LQ by premultiplying it by an orhogonal matrix P^T where P is again formed as product of Givens rotations. It follows that

$$\epsilon \geq \sigma = \|Lv\| = \|P^T LQQ^T v\| = \|P^T LQe_1\|, \tag{8}$$

and hence the first column of the triangular matrix $P^T LQ$ is small. This way of proceeding is called deflation and applying it repeatedly, the $TDULV$ is computed. However, to obtain accurate null-spaces one may have to delay the deflation and refine the factorization until the required accuray is achieved.

4.1 Refinement. Factorization (7) reveals the numerical rank of A by the dimension of the matrix L_k. Bases for approximate left and right null-spaces of A are given by the first $n-k$ columns of U and V respectively. To obtain an accurate basis for the left null-space one may need to refine the factorization by bringing the matrix E to near diagonal form. This is achieved by Givens rotations. Suppose we have obtained the partial factorization

$$A = U \begin{pmatrix} e & 0 \\ h & L_{n-1} \end{pmatrix} V^T, \tag{9}$$

where the matrix in the middle is assumed to be rank deficient.
The aim of the refinement is to make the norm $\|h\| \leq \tau$, where τ is some deflation parameter. This leads to accurate bases for null-spaces of A and A^T. The first step in the refinement is to compute an orthogonal matrix Q such that off-diagonal elements of the first column of LQ vanish, *i.e.*,

$$LQ = \begin{pmatrix} e' & h'^T \\ 0 & L'_{n-1} \end{pmatrix}. \tag{10}$$

This is achieved by zeroing successively elements of h by means of Givens rotations. The matrix Q is the product of these Givens rotations applied to

L from the right. Nonzero elements then appear in the first row of LQ. The second step in the refinement is to determine an orthogonal matrix P such that element of h'^T are annihilated by premultiplying LQ by P^T. This is done by zeroing successively the elements of h'^T by means of Givens rotations. We then obtain the following lower triangular matrix

$$P^T LQ = \begin{pmatrix} e'' & 0 \\ h'' & L''_{n-1} \end{pmatrix}. \tag{11}$$

After these two steps of refinement, elements of h'' have become smaller. If $\|h''\| < \tau$, then we deflate the first row and column in (11). To maintain null-spaces, transformations P and Q in these two steps of refinement are also applied to U and V. At this point, the factorization of A is given by

$$A = (UP) \begin{pmatrix} e'' & 0 \\ h'' & L''_{n-1} \end{pmatrix} (VQ)^T. \tag{12}$$

In this fashion the matrix E in (7) is made "closer" to a diagonal matrix.

4.2 TDULV-Updating. The $TDULV$ factorization can be updated when a new row is incorporated at the bottom of the matrix A. Assume that after having computed a rank-revealing factorization (7) of A, we wish to include a new row in A. The aim of the updating is to compute a rank-revealing factorization of the updated matrix from that of A at a low computational cost namely $\mathcal{O}(n^2)$ or less. This should be done without destroying small elements of E and F. The row-updating of the rank-revealing $TDULV$ is described as follows

$$\begin{pmatrix} A \\ a^T \end{pmatrix} = \begin{pmatrix} U & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} E & 0 \\ H & L_k \\ x^T & y^T \end{pmatrix} V^T, \tag{13}$$

where a^T is the appended row and where $(x^T \; y^T) = a^T V$.

In the first phase of updating we annihilate the first $n-k-1$ components of x^T, while maintaining the triangular form of E. This is performed by applying a sequence of interleaved right and left Givens rotations. In the process each right rotation introduces above the diagonal of E a nonzero element which is zeroed out by left rotation. In this annihilation process of x^T only rows of E and the first $n-k$ columns of the middle matrix in (13) are involved. In this fashion "smallness" of matrices E and H is preserved. The reduction procedure, where only E and x^T are shown, is illustrated in $Fig.$ 1 (In all figures, vertical arrows point out the columns involved in a postmultiplication by a rotation. Horizontal arrows point out the rows involved in a premultiplication by a rotation. A check over an element indicates the element to be eliminated.).

$$
\begin{array}{cccc}
\downarrow & \downarrow & & \\
e & & & \\
e & e & & \\
e & e & e & \\
e & e & e & e \\
\check{x} & x & x & x
\end{array}
\Longrightarrow
\begin{array}{ccccc}
 & & & & \\
\rightarrow & e & \check{e} & & \\
\rightarrow & e & e & & \\
 & e & e & e & \\
 & e & e & e & e \\
 & 0 & x & x & x
\end{array}
\Longrightarrow
\begin{array}{cccc}
 & \downarrow & \downarrow & \\
e & & & \\
e & e & & \\
e & e & e & \\
e & e & e & e \\
0 & \check{x} & x & x
\end{array}
\Longrightarrow
$$

$$
\begin{array}{ccccc}
 & & & & \\
 & e & & & \\
\rightarrow & e & e & \check{e} & \\
\rightarrow & e & e & e & \\
 & e & e & e & e \\
 & 0 & 0 & x & x
\end{array}
\Longrightarrow
\begin{array}{cccc}
 & & \downarrow & \downarrow \\
e & & & \\
e & e & & \\
e & e & e & \\
e & e & e & e \\
0 & 0 & \check{x} & x
\end{array}
\Longrightarrow
\begin{array}{ccccc}
 & & & & \\
 & e & & & \\
 & e & e & & \\
\rightarrow & e & e & \check{e} & \\
\rightarrow & e & e & e & e \\
 & 0 & 0 & 0 & x
\end{array}
\Longrightarrow
\begin{array}{cccc}
 & & & \\
e & & & \\
e & e & & \\
e & e & e & \\
e & e & e & e \\
0 & 0 & 0 & x
\end{array}
$$

Fig. 1 Annihilation of components of x^T

The second phase is to triangularize the following trapezoidal matrix by annihilating the last row

$$
\begin{array}{ccccc}
e & & & & \\
h & l & & & \\
h & l & l & & \\
h & l & l & l & \\
h & l & l & l & l \\
x & y & y & y & y
\end{array}
$$

This is performed by means of Givens rotations as follows

$$
\begin{array}{ccccc}
 & & & \downarrow & \downarrow \\
e & & & & \\
h & l & & & \\
h & l & l & & \\
h & l & l & l & \\
h & l & l & l & l \\
x & y & y & y & \check{y}
\end{array}
\Longrightarrow
\begin{array}{cccccc}
 & e & & & & \\
 & h & l & & & \\
 & h & l & l & & \\
\rightarrow & h & l & l & l & \check{l} \\
\rightarrow & h & l & l & l & l \\
 & x & y & y & y & 0
\end{array}
\Longrightarrow
\begin{array}{ccccc}
 & & \downarrow & \downarrow & \\
e & & & & \\
h & l & & & \\
h & l & l & & \\
h & l & l & l & \\
h & l & l & l & l \\
x & y & y & \check{y} & 0
\end{array}
\Longrightarrow
$$

$$
\begin{array}{cccccc}
 & e & & & & \\
 & h & l & & & \\
\rightarrow & h & l & l & \check{l} & \\
\rightarrow & h & l & l & l & \\
 & h & l & l & l & l \\
 & x & y & y & 0 & 0
\end{array}
\Longrightarrow
\begin{array}{ccccc}
 & \downarrow & \downarrow & & \\
e & & & & \\
h & l & & & \\
h & l & l & & \\
h & l & l & l & \\
h & l & l & l & l \\
x & y & \check{y} & 0 & 0
\end{array}
\Longrightarrow
\begin{array}{cccccc}
 & e & & & & \\
\rightarrow & h & l & \check{l} & & \\
\rightarrow & h & l & l & & \\
 & h & l & l & l & \\
 & h & l & l & l & l \\
 & x & y & 0 & 0 & 0
\end{array}
\Longrightarrow
$$

$$
\begin{array}{cccccc}
 & e & & & & \\
\rightarrow & h & l & & & \\
 & h & l & l & & \\
 & h & l & l & l & \\
 & h & l & l & l & l \\
\rightarrow & x & \check{y} & 0 & 0 & 0
\end{array}
\Longrightarrow
\begin{array}{cccccc}
\rightarrow & e & & & & \\
 & x & l & & & \\
 & h & l & l & & \\
 & h & l & l & l & \\
 & h & l & l & l & l \\
\rightarrow & \check{x} & 0 & 0 & 0 & 0
\end{array}
\Longrightarrow
\begin{array}{ccccc}
x & & & & \\
x & l & & & \\
h & l & l & & \\
h & l & l & l & \\
h & l & l & l & l \\
0 & 0 & 0 & 0 & 0
\end{array}
$$

Fig. 2 Triangularization

The triangularization process replaces the zeros in the last matrix of $Fig.1$ by some small elements h. These small elements can be eliminated or neglected.

5. TDULV-Algorithm.

The rank-revealing $TDULV$ factorization is summarized in the following algorithm:

Input:

- Matrix $A \in \mathcal{R}^{m \times n}$ $(m \geq n)$ to be decomposed.
- Threshold ϵ for singular values of A.
- Deflation tolerance τ for $\|H\|$.

- Maximum number of iterations N_τ for the refinement.

Output:

- Numerical rank k.
- Orthogonal matrices $U \in \mathcal{R}^{m\times m}$, $V \in \mathcal{R}^{n\times n}$ and a lower triangular matrix $L \in \mathcal{R}^{n\times n}$.

1. Compute a QL factorization of A: $A = Q\binom{0}{L}$, where Q is orthogonal and $L \in \mathcal{R}^{n\times n}$ is lower triangular (*e.g.*, using the LAPACK routine xGEQLF).
2. Initialization: $U \leftarrow Q$, $V \leftarrow I_n$, $k \leftarrow n$ and $itstep \leftarrow 0$.
3. Compute the smallest singular value σ_n of L and the associated right null vector $v^n \in \mathcal{R}^n$ of norm one (*e.g.*, by using the incremental condition estimator (ICE)[5] implemented in LAPACK routines xLAIC1).
4. **While** ($\sigma_k < \epsilon$ and $k \geq 2$) **do**
   While ($itstep \leq N_\tau$) **do**
   For $j = n-1, \cdots, n-k+1$ **do**
   Determine a Givens rotation $Q_{j,\,j+1} \in \mathcal{R}^{k\times k}$, so that premultiplication of v^k by $Q^T_{j,\,j+1}$ zeroes component v^k_{j+1} of v^k using v^k_j. Update L and V
   $L \leftarrow L\begin{pmatrix} I_{n-k} & 0 \\ 0 & Q_{j,\,j+1}\end{pmatrix}$ and $V \leftarrow V\begin{pmatrix} I_{n-k} & 0 \\ 0 & Q_{j,\,j+1}\end{pmatrix}$.
   Determine a Givens rotation $P_{j,\,j+1} \in \mathcal{R}^{k\times k}$, so that premultiplication of $L_k := L(n-k+1:n, n-k+1:n)$ by $P^T_{j,\,j+1}$ zeroes $l_{j,\,j+1}$ using $l_{j+1,\,j+1}$. Update L and U
   $L \leftarrow \begin{pmatrix} I_{n-k} & 0 \\ 0 & P^T_{j,\,j+1}\end{pmatrix} L$ and $U \leftarrow U\begin{pmatrix} I_{m-k} & 0 \\ 0 & P_{j,\,j+1}\end{pmatrix}$.
   Enddo
   If ($\|L(n-k+2:n, n-k+1)\| < \tau$ or $itstep > N_\tau$) **then**
   Deflation: Set $k \leftarrow k-1$ and $itstep \leftarrow 0$.
   Else
   Refinement: Set $itstep \leftarrow itstep + 1$.
   Determine a sequence of Givens rotations $\bar{Q}_{n+1-k,\,n+2-k}, \ldots, \bar{Q}_{n+1-k,\,n}$ so that postmultiplication of L_k by $\bar{Q}_k := \bar{Q}_{n+1-k,\,n+2-k} \cdots \bar{Q}_{n+1-k,\,n}$ zeroes the

elements $L(n-k+2:n, n-k+1)$. Update L and V
$L \leftarrow L \begin{pmatrix} I_{n-k} & 0 \\ 0 & \bar{Q}_k \end{pmatrix}$ and $V \leftarrow V \begin{pmatrix} I_{n-k} & 0 \\ 0 & \bar{Q}_k \end{pmatrix}$.
Determine a sequence of Givens rotations
$\bar{P}_{n+1-k,\, n}, \ldots, \bar{P}_{n+1-k,\, n+2-k}$ so that premultiplication of L_k by $\bar{P}_k^T$, where $\bar{P}_k := \bar{P}_{n+1-k,\, n} \cdots \bar{P}_{n+1-k,\, n+2-k}$, zeroes the elements $L(n-k+1, n-k+2:n)$. Update L and U
$L \leftarrow \begin{pmatrix} I_{n-k} & 0 \\ 0 & \bar{P}_k^T \end{pmatrix} L$ and $U \leftarrow U \begin{pmatrix} I_{m-k} & 0 \\ 0 & \bar{P}_k \end{pmatrix}$.

Endif

Compute the smallest singular value σ_k of L_k and the associated right null vector $v^k \in \mathcal{R}^k$ of norm one.

Endwhile

Endwhile

End of TDULV – Algorithm

The algorithm terminates if a lower bound $\sigma_k > \epsilon$ is computed. The dimension k of the bottom right matrix L_k is equal to the numerical rank. Bases for left and right null-spaces are given by the first $n+1-k$ columns of U and V, respectively.

Example 1

We now describe the steps for the $(n+1-k)th$ stage of the algorithm and illustrate it for the case $k=4$. At this stage already $(n-4)$ deflations have been performed and the matrix L has the form (7) where the right bottom submatrix L_4 has dimension $k=4$. According to the above algorithm, only the matrix L_4 is involved in the coming steps, we therefore sketch only this matrix. The next step in our algorithm is to compute σ and $v \in \mathcal{R}^4$, approximations of the smallest singular value of L_4 and the associated right null vector. Then we annihilate successively the $4th, 3rd$ and the $2nd$ component of v using Givens rotations Q_{12}^T, Q_{23}^T, Q_{34}^T so that

$$Q_{12}^T Q_{23}^T Q_{34}^T v = (1,0,0,0)^T.$$

The sketch below clarifies the effect of carring out successive transformations

$Q_{i,\,i+1}$ on v.

$$
\begin{array}{c}\\ \\ \rightarrow \\ \rightarrow\end{array}\begin{pmatrix} v \\ v \\ v \\ v \end{pmatrix} \xmapsto{Q_{34}^T} \begin{array}{c}\\ \rightarrow \\ \rightarrow \\ \\ \end{array}\begin{pmatrix} v \\ v \\ v \\ 0 \end{pmatrix} \xmapsto{Q_{23}^T} \begin{array}{c}\rightarrow \\ \rightarrow \\ \\ \\ \end{array}\begin{pmatrix} v \\ v \\ 0 \\ 0 \end{pmatrix} \xmapsto{Q_{12}^T} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.
$$

Fig. 3 Reduction of v

We must then postmultiply L_4 by these rotations. This multiplication by $Q_{i,\,i+1}$ produces a nonzero $(i, i+1)$ entry in L_4. To restore the triangular form to L_4, we premultiply it by some appropriate plane rotation $P_{i,\,i+1}^T$. For $i = 3, 2, 1$ we then have

$$
L_4 \rightarrow L_4 Q_{34} \rightarrow P_{34}^T L_4 Q_{34} \rightarrow P_{34}^T L_4 Q_{34} Q_{23} \rightarrow P_{12}^T P_{23}^T P_{34}^T L_4 Q_{34} Q_{23} Q_{12}.
$$

The triangular form changes as follows

$$
\begin{array}{cccc} & & \downarrow & \downarrow \\ l & & & \\ l & l & & \\ l & l & l & \\ l & l & l & l \end{array}
\Longrightarrow
\begin{array}{ccccc} & & & & \\ & l & & & \\ & l & l & & \\ \rightarrow & l & l & l & \check{l} \\ \rightarrow & l & l & l & l \end{array}
\Longrightarrow
\begin{array}{cccc} & \downarrow & \downarrow & \\ l & & & \\ l & l & & \\ l & l & l & \\ l & l & l & l \end{array}
\Longrightarrow
\begin{array}{ccccc} & & & & \\ & l & & & \\ \rightarrow & l & l & \check{l} & \\ \rightarrow & l & l & l & \\ & l & l & l & l \end{array}
\Longrightarrow
$$

$$
\begin{array}{cccc} \downarrow & \downarrow & & \\ l & & & \\ l & l & & \\ l & l & l & \\ l & l & l & l \end{array}
\Longrightarrow
\begin{array}{ccccc} \rightarrow & l & \check{l} & & \\ \rightarrow & l & l & & \\ & l & l & l & \\ & l & l & l & l \end{array}
\Longrightarrow
\begin{array}{cccc} e & & & \\ h & l & & \\ h & l & l & \\ h & l & l & l \end{array}
$$

Fig. 4 Deflation procedure

The elements h and e in the first column of L_4 indicate that small elements have been generated in this column. To see this, consider the norm of $L_4 v$,

$$
\epsilon \geq \sigma_4 = \|L_4 v\| = \|P^T L_4 Q Q^T v\| = \|P^T L_4 Q e_1\|,
$$

where $P = P_{34} P_{23} P_{12}$ and $Q = Q_{34} Q_{23} Q_{12}$. Thus the first column of the triangular matrix $P^T L_4 Q$ is small. One can stop at this point and take the computed factorization as rank-revealing factorization. However, to obtain the correct numerical rank and accurate null-spaces, one has to bring the matrix E to near diagonal form by reducing the norm of H. This is acomplished by reducing the size of elements h in the first column of $P^T L_4 Q$ in each deflation step as follows

$$
\begin{array}{cccc}
\downarrow & \downarrow & & \\
e & & & \\
\check{h} & l & & \\
h & l & l & \\
h & l & l & l
\end{array}
\Longrightarrow
\begin{array}{cccc}
\downarrow & & \downarrow & \\
e & h & & \\
0 & l & & \\
\check{h} & l & l & \\
h & l & l & l
\end{array}
\Longrightarrow
$$

$$
\begin{array}{cccc}
\downarrow & & & \downarrow \\
e & h & h & \\
0 & l & & \\
0 & l & l & \\
\check{h} & l & l & l
\end{array}
\Longrightarrow
\begin{array}{cccc}
e & h & h & h \\
0 & l & & \\
0 & l & l & \\
0 & l & l & l
\end{array}
$$

Fig. 5 Zeroing off-diagonal elements of the first column

We reduce now the first row of the matrix L using left rotations as follows

$$
\begin{array}{ccccc}
\rightarrow & e & h & h & \check{h} \\
 & 0 & l & & \\
 & 0 & l & l & \\
\rightarrow & 0 & l & l & l
\end{array}
\Longrightarrow
\begin{array}{ccccc}
\rightarrow & e & h & \check{h} & \\
 & 0 & l & & \\
\rightarrow & 0 & l & l & \\
 & h & l & l & l
\end{array}
\Longrightarrow
$$

$$
\begin{array}{ccccc}
\rightarrow & e & \check{h} & & \\
\rightarrow & 0 & l & & \\
 & h & l & l & \\
 & h & l & l & l
\end{array}
\Longrightarrow
\begin{array}{cccc}
e & & & \\
h & l & & \\
h & l & l & \\
h & l & l & l
\end{array}
$$

Fig. 6 Zeroing off-diagonal elements of the first row

Example 2

Let A be the lower triangular matrix with 1 on the diagonal and -1 as off-diagonal elements. For $m = n = 3$ and $\epsilon = 1.5$ the rank is 2 and the middle matrix L is

$$
\begin{pmatrix}
.3472963553338607 & 0 & 0 \\
9.302245467261437e-14 & 1.53607859485413 & 0 \\
4.855638724371323e-14 & 7.799425873704058e-2 & -1.87450385105148
\end{pmatrix}.
$$

Tables 1 and 2 show that the null-spaces computed from the $TDULV$ closely approximate the null-spaces computed from the SVD.

U_{TDULV}	U_{SVD}
.8440296287459917	.8440296287459852
.4490987851112734	.4490987851112867
.2931284138572732	.2931284138572721

Table 1: Bases for the left null-space

V_{TDULV}	V_{SVD}
.2931284138573261	.2931284138572723
.4490987851112454	.4490987851112868
.8440296287459887	.8440296287459852

Table 2: Bases for the right null-space

6. Conclusion. In this paper we have proposed a new ULV factorization called $TDULV$ factorization and an algorithm to compute it. This factorization is based on right null vectors of lower triangular matrices rather than left null vectors as in the ULV factorizations. First, this has resulted in avoiding triangular solves, which may be expensive on parallel computers and reducing the cost related to these solves especially if they have to be performed many times with very large matrices. Second, this avoids including parameters related to triangular solves. Furthermore our method uses null vectors computed by condition estimators in a straightforward way. Therefore it may be more accurate than the URV in exhibing the numerical rank and bases for null-spaces.

Appendix.

Lemma 1

Rank-revealing URV factorizations and $TDULV$ factorizations of a matrix A are equivalent.

Proof

Let A have the rank-revealing $TDULV$ factorization

$$A = U\begin{pmatrix} E & 0 \\ H & L_k \end{pmatrix}V^T.$$

Then we can write

$$A = (UJ_m)\, J_m \begin{pmatrix} E & 0 \\ H & L_k \end{pmatrix} J_n\, (VJ_n)^T ,$$

where

$$J_m = \begin{pmatrix} 0 & J_k \\ J_{m-k} & 0 \end{pmatrix}, \qquad J_n = \begin{pmatrix} 0 & J_{n-k} \\ J_k & 0 \end{pmatrix}$$

and where J_k denotes the $k \times k$ flip matrix. Therefore

$$A = (UJ_m) \begin{pmatrix} J_k L_k J_k & J_k H J_{n-k} \\ 0 & J_{m-k} E J_{n-k} \end{pmatrix} (VJ_n)^T ,$$

which is of the form

$$A = \hat{U} \begin{pmatrix} R & F \\ 0 & G \end{pmatrix} \hat{V}^T$$

with

$$G = J_{m-k} E J_{n-k}, \quad F = J_k H J_{n-k}, \quad R = J_k L_k J_k, \quad \hat{U} = UJ_m, \quad \hat{V} = VJ_n.$$

Note that the matrices G and R are upper triangular and $\hat{U}$ and $\hat{V}$ are orthogonal. Furthermore we have

$$\|G\| = \|E\|, \qquad \|F\| = \|H\|, \qquad \|R_k^{-1}\| = \|L_k^{-1}\|,$$

therefore the $TDURV$ factorization is rank revealing if and only if the URV is rank-revealing. The bases obtained from the $TDULV$ are given by the first columns of the orthogonal matrices UJ_m and VJ_n.

References

[1] M. W. Berry, S. Dumais And G. O'Brien, *The computational complexity of alternative updating approaches for an SVD-encoded indexing scheme*, In Proceedings of the Seventh SIAM Conference of Parallel Processing for Sci. Comput., SIAM, Philadelphia, 1995.

[2] M. W. Berry And R. D. Fierro, *Low-rank orthogonal decomposition for retrieval applications*, Num. Linear Alg. Appl. 3, (1996), 301-327.

[3] C. H. Bischof, *Incremental condition estimator*, SIAM J. Matrix Anal. Appl. 11, (1990), 312-322.

[4] C. H. Bischof, J. G. Lewis And D. J. Pierce, *Incremental condition estimation for sparse matrices*, SIAM J. Matrix Anal. Appl. 11, (1990), 644-659.

[5] C. H. Bischof And P. T. P. Tang, *Generalizing incremental condition estimation*, J. Numer. Linear Algebra Appl. 1, (1992), 149-163.

[6] J. R. Bunch And N. P. Nielsen, *Updating the SVD*, Numer. Math. 31, (1978), 111-129.

[7] T. F. Coleman, *Large sparse numerical optimization*, Lecture Notes in Computer Science 165, Sringer-Verlag, New York, (1984), 331-346.

[8] R. D. Fierro, *Perturbation analysis for tow-sided (or complete) orthogonal decompositions*, SIAM J. Matrix Anal. Appl. 17, (1996), 383-400.

[9] R. D. Fierro And P. C. Hansen, *Low rank revealing two- sided orthogonal decompositions*, Tech. Report 94-09, Department of Mathematics, California State Univerisity, San Marcos, CA, October 1994.

[10] R. D. Fierro And J. R. Bunch, *Bounding the subspaces from rank revealing two-sided orthogonal decompositions*, SIAM J. Matrix Anal. Appl. 16, (1995), 743-759.

[11] R. D. Fierro And P. C. Hansen, *Accuracy of TSVD solutions computed from rank-revealing decompositions*, Numer. Math. 70, (1995) 453-471.

[12] G. H. Golub, *Numerical methods for solving linear least squares problems*, Numer. Math. 7, (1965), 206-216.

[13] G. H. Golub, V. Klema And G. W. Stewart, *Rank degeneracy and least squares problems*, Tech. Report TR-456, Department of Computer Science, University of Mayland, 1976.

[14] G. H. Golub And C. V. Loan, *Matrix computations*, John-Hopkins, Baltimore, Second Edition, 1998.

[15] W. W. Hager, *Condition estimates*, SIAM J. Sci. Statist. Comput. 29, (1987), 311-316.

[16] N. J. Higham, *A survey of condition number estimation for triangular matrices*, SIAM Review 29, (1987), 575-596.

[17] P. Kunkel, V. Mehrmann, *A new class of discretization methods for the solution of linear differential-algebraic equations with variable coefficients*, SIAM J. Numer. Anal. 33, (1996), 1941-1961.

[18] P. Kunkel, V. Mehrmann, *Local and global invariants of linear differential-algebraic equations and their relations*, Electr. Trans. Numer. Analysis, to appear.

[19] P. Kunkel, V. Mehrmann W. Rath And J. Weickert, *A new software package for linear differential-algebraic equations*, SIAM J. Sci. Comput. 18, (1997), 115-138.

[20] C. L. Lawson And R. J. Hanson, *Solving least squares problems*, Prentice-Hall, Englewood Cliffs, NJ, 1974.

[21] K. J. R. Liu, D. P. O'Leary, G. W. Stewart And Y.-J. J. Wu, *URV ESPRIT for tracking time-varying signals*, IEEE Trans. Signal Proc. 42, (1994), 3441-3449.

[22] R. Mathias And G. W. Stewart, *A block QR algorithm and the singular value decomposition*, Linear Algebra Appl. 182, (1993), 91-100.

[23] H. Park And L. Eldén, *Downdating the rank-revealing URV decompositions*, SIAM J. Matrix Anal. Appl. 16, (1995), 138-155.

[24] D. J. Pierce And R. J. Plemmons, *Fast adaptive condition estimation*, SIAM J. Matrix Anal. Appl. 13, (1992), 274-291.

[25] G. W. Stewart, *On algorithm for refining a rank-revealing URV decomposition and a perturbation theorem for singular values*, Tech. Report CS-TR 2626, University of Mayland, Department of Computer Science, College Park, MD, 1991.

[26] G. W. Stewart, *An updating algorithm for for subspace tracking*, IEEE Trans. Signal Proc. 40, (1992), 1535-1541.

[27] G. W. Stewart, *Updating a rank-revealing ULV decomposition*, SIAM J. Matrix Anal. Appl. 14, (1993), 494-499.

[28] G. W. Stewart, *Determining rank in presence of errors*, pp. 275-291 in M. S. Moonen, G. H. Golub And B. L. R. De Moor (Eds.), Linear Algebra for Large Scale and Real-time Applications, NATO ASI Series, Kluwer, Dordrecht, 1993.

[29] G. W. Stewart, *Updating URV decompositions in parallel*, Parallel Comp. 20, (1994), 151-172.

[30] P. T. P. Tang, *Dynamic condition estimation and Rayleigh-Ritz approximation*, SIAM J. Matrix Anal. Appl. 15, (1994), 331-346.

[31] B. N. PARLETT, *The symmetric eigenvalue problem*, Prentice-Hall, Englewood Cliffs, N.J., 1980.