An Accurate SVD Algorithm for 2×2 Triangular Matrices

Vjeran Hari[*1] and **Josip Matejaš**[**2]

[1] Department of Mathematics, University of Zagreb, P.O. Box 335, 10002 Zagreb, Croatia.
[2] Faculty of Economics, University of Zagreb, Kennedyjev trg 4, 10000 Zagreb, Croatia

Received July 10, 2006

Key words singular value decomposition, triangular 2×2 matrix, accurate algorithm
Subject classification 65F15, 65G05

In this paper we present ideas that are used to define an accurate algorithm for computing the singular value decomposition of two-by-two triangular matrices and for proving the appropriate accuracy bounds. The angle formulas, originally proposed by Voevodin, are modified to become relatively accurate in floating point arithmetics. The rounding error analysis uses natural assumptions which fully comply with the IEEE floating point standards. A subtle analysis is used to express the final errors as functions of errors of some beginning or intermediate quantities. This enables to prove much sharper final error bounds, especially in the case when the initial matrix is nearly diagonal.

1 Introduction

Recently, Drmač and Veselić have proposed a compound SVD solver [3, 4] for general matrices, which is both efficient and accurate. Their algorithm first prepares the initial matrix for the iteration by using one or two QR factorizations and after that one-sided Jacobi algorithm is applied to the obtained triangular matrix. Although excellent in many aspects, their second phase of the algorithm has three minor drawbacks. First, it destroys the triangular structure obtained after the first phase. Second, it destroys frequently noticed almost diagonality of the triangular matrix. Finally, the stopping of the algorithm is expensive and based on data (which are tiny but) computed with large relative errors.

Kogbetliantz algorithm [13, 14] can also be used in the second phase, instead of the one-sided Jacobi. Under special pivot strategies [7, 10], it will not essentially destroy the triangular matrix structure. In addition, it will just further diagonalize an almost diagonal triangular matrix. And final stopping of the process will be cheap and safe. Many other excellent features, like convergence properties [5, 8, 16, 9], a posterior computation of right (or left) singular vectors [2], paralellizing and blocking of the algorithm [11], are shared with the one-sided Jacobi method. However, the main drawbacks of Kogbetliantz are speed slowdown caused by the left-hand transformations and lack of the accuracy proof. The first problem can be partly solved in the context of parallelizing the algorithm (see [10, 11]). The accuracy issue is the subject of our recent research [12]. Here we address the main subproblem: accuracy of a singular value decomposition algorithm for 2×2 triangular matrices.

This problem is not quite a new one. The routine `*LASV2` from LAPACK, the known library of linear algebra subroutines, can be used to compute SVD of triangular 2×2 matrices. Another algorithm for the same problem [1], which uses 2×2 reflectors instead of rotations, can be used too. The accuracy of the latter algorithm is proved in [15]. Our approach is based on Voevodin formulas [17] which are modified to be relatively accurate.

This report is organized as follows. In Section 1 we introduce the modified algorithm based of Voevodin formulas. In Section 2, we present assumptions that are used in the accuracy proof. In Section 3, we briefly present the essential accuracy bounds for the new algorithm and compare them with the bounds obtained for the algorithm used in `*LASV2`.

* Corresponding author: e-mail: hari@math.hr, Phone: +385 1 4605 748, Fax: +385 1 4680 335
** author: e-mail: jmatejas@efzg.hr, Phone: +385 1 238 3333,

2 Modified Voevodin Algorithm

Here we present the formulas for the rotation angles and for the transformation of the diagonal elements when computing SVD of a triangular matrix of order two. As is known (see [17, 7, 6, 11]), for each of the cases, the upper-triangular matrix and the lower-triangular matrix, there exist two pairs formulas.

We start with the case of an upper-triangular 2×2 matrix T whose non-zero elements are denoted by f, g and h. A single Kogbetliantz step diagonalizes T through the orthogonal transformation

$$T' = \begin{bmatrix} f' & 0 \\ 0 & h' \end{bmatrix} = \begin{bmatrix} c_\varphi & s_\varphi \\ -s_\varphi & c_\varphi \end{bmatrix} \begin{bmatrix} f & g \\ 0 & h \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi \\ s_\psi & c_\psi \end{bmatrix} = U^T T V.$$

Here c_φ, s_φ, c_ψ and s_ψ denote $\cos(\varphi)$, $\sin(\varphi)$, $\cos(\psi)$ and $\sin(\psi)$, respectively. We shall also use t_φ and t_ψ for $\tan(\varphi)$ and $\tan(\psi)$, respectively. There are several formulas for computing c_φ, s_φ, c_ψ and s_ψ. If the right-hand transformation is applied to T as first, we obtain (using the notation from [6, 11]) the UR (Upper-triangular, Right transformation first) algorithm. If the left transformation is applied first, we obtain the UL algorithm.

UL

$$\tan 2\varphi = \frac{2gh}{f^2 + g^2 - h^2}\,.$$

$$\tan\psi = \frac{g + ht_\varphi}{f} = \frac{ft_\varphi}{h - gt_\varphi}\,.$$

$$f' = \frac{c_\varphi}{c_\psi} f$$

$$h' = \frac{c_\psi}{c_\varphi} h$$

UR

$$\tan 2\psi = \frac{2fg}{f^2 - g^2 - h^2}\,.$$

$$\tan\varphi = \frac{ft_\psi - g}{h} = \frac{ht_\psi}{f + gt_\psi}\,.$$

The formulas in boxes are used for computing c_ψ, s_ψ, c_φ, s_φ. The formulas for f' and g' hold with both, UR and UL algorithms.

If T is lower-triangular, a single Kogbetliantz step takes form

$$T' = \begin{bmatrix} f' & 0 \\ 0 & h' \end{bmatrix} = \begin{bmatrix} c_\varphi & s_\varphi \\ -s_\varphi & c_\varphi \end{bmatrix} \begin{bmatrix} f & 0 \\ g & h \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi \\ s_\psi & c_\psi \end{bmatrix} = U^T T V.$$

Transposing this equation, one can invoke the above formulas and obtain (see [6, 11]),

LL

$$\tan 2\varphi = \frac{2fg}{f^2 - g^2 - h^2}\,,$$

$$\tan\psi = \frac{ft_\varphi - g}{h} = \frac{ht_\varphi}{f + gt_\varphi}\,,$$

$$f' = \frac{c_\psi}{c_\varphi} f$$

$$h' = \frac{c_\varphi}{c_\psi} h$$

LR

$$\tan 2\psi = \frac{2gh}{f^2 + g^2 - h^2}\,,$$

$$\tan\varphi = \frac{g + ht_\psi}{f} = \frac{ft_\psi}{h - gt_\psi}\,.$$

The formulas show that the non-negativity (positivity) of f and g implies the non-negativity (positivity) of f' and g'. Therefore, we recommend and in further exposition assume that the non-negativity of the diagonal elements are assured before the iteration begins.

As is shown in [12], it is always possible to chose between UR and UL (LR and LL) algorithms to achieve that c_ψ, s_ψ, c_φ, s_φ, f' and g' are computed with small relative error. This is the basis of the accuracy proof for the serial and modulus Kogbetliantz method. Here, we briefly describe the modified accurate algorithm.

Modified Algorithm:

- If A is upper-triangular and $f \geq h$, then **UL** algorithm is used
- If A is upper-triangular and $f < h$, then **UR** algorithm is used
- If A is lower-triangular and $f \geq h$, then **LR** algorithm is used
- If A is lower-triangular and $f < h$, then **LU** algorithm is used

In [12], we have further refined the above formulas to avoid overflows caused by finite arithmetic.

3 Rounding Error Analysis Assumptions

Since almost all today's computers comply with the IEEE standard, we shall assume that the unit round-off $\mathbf{u}$ satisfies

$$\mathbf{u} \in \left\{2^{-23},\ 2^{-24},\ 2^{-52},\ 2^{-53},\ 2^{-64}\right\}. \tag{1}$$

Here 2^{-23} corresponds to the unit round-off for single precision with mode other than rounding to the nearest. Usually, in single precision computation, the default for many compilers is rounding to nearest, i.e. $\mathbf{u} = 2^{-24}$ (in double precision $\mathbf{u} = 2^{-53}$). The case $\mathbf{u} = 2^{-64}$ corresponds to the extended precision computation. If computation is made in double precision, one can just insert 2^{-53} (or 2^{-52}) for $\mathbf{u}$ in the final estimates. If the computation is performed in quadruple precision arithmetic, one can add further terms in the set defined by (1).

To make the analysis easier to read, we have used in [12] the following notation

$\varepsilon_{subscript}$: a quantity bounded in modulus by $\mathbf{u}$,

$\boldsymbol{\epsilon}_{subscript}$: a quantity bounded in modulus by a multiple of $\mathbf{u}$.

We have used the standard model of machine arithmetic. The floating point results for the basic operation $\circ$ and for the square root are given by

$$\begin{aligned} fl(a \circ b) &= (a \circ b)(1+\varepsilon), \quad |\varepsilon| \le \mathbf{u}, \quad \circ \in \{+,-,*,/\},\ \varepsilon = \varepsilon(a,b,\circ) \quad \text{and} \\ fl(\sqrt{a}) &= \sqrt{a}(1+\varepsilon_{\surd}) \quad |\varepsilon_{\surd}| \le \mathbf{u},\ \varepsilon_{\surd} = \varepsilon_{\surd}(a), \end{aligned}$$

respectively. In the rounding error analysis, we have frequently used the following auxiliary results.

Lemma 3.1 *If* $|\varepsilon_i| \le \mathbf{u}$, $1 \le i \le n$, *then*

(i) $\displaystyle\prod_{i=1}^{n}(1+\varepsilon_i) = 1 + \boldsymbol{\epsilon}_n$, *where*

$$\begin{aligned} &|\boldsymbol{\epsilon}_2| \le 2.000\,000\,12\,\mathbf{u}, \quad |\boldsymbol{\epsilon}_3| \le 3.000\,000\,36\,\mathbf{u}, \quad |\boldsymbol{\epsilon}_4| \le 4.000\,000\,72\,\mathbf{u}, \\ &|\boldsymbol{\epsilon}_5| \le 5.000\,001\,2\,\mathbf{u}, \quad |\boldsymbol{\epsilon}_6| \le 6.000\,001\,8\,\mathbf{u} \\ &|\boldsymbol{\epsilon}_n| \le n\boldsymbol{\epsilon}\,(1 + 0.000\,000\,06\,n), \quad 7\boldsymbol{\epsilon} \le n\boldsymbol{\epsilon} \le 2^{-6}. \end{aligned}$$

(ii) $(1+\varepsilon)^{1/2} = 1 + \varepsilon_{1/2}$, $|\varepsilon_{1/2}| \le 0.500\,000\,015\,|\varepsilon|$, $|\varepsilon| \le \boldsymbol{\epsilon}$.

Lemma 3.2 *If* $|\boldsymbol{\epsilon}_1| \le p\,\mathbf{u}$, $|\boldsymbol{\epsilon}_2| \le r\mathbf{u}$, *then*

(i) $(1+\boldsymbol{\epsilon}_1)(1+\boldsymbol{\epsilon}_2) = 1 + \boldsymbol{\epsilon}_3$, $|\boldsymbol{\epsilon}_3| = |\boldsymbol{\epsilon}_1 + \boldsymbol{\epsilon}_2 + \boldsymbol{\epsilon}_1 \cdot \boldsymbol{\epsilon}_2| \ \le\ (p + r + 2^{-23} \cdot pr)\mathbf{u}$,

(ii) $\dfrac{1+\boldsymbol{\epsilon}_1}{1+\boldsymbol{\epsilon}_2} = 1 + \boldsymbol{\epsilon}_4$, $|\boldsymbol{\epsilon}_4| = \left|\dfrac{\boldsymbol{\epsilon}_1 - \boldsymbol{\epsilon}_2}{1+\boldsymbol{\epsilon}_2}\right| \le (p+r)\left(1 + \dfrac{2^{-23}\cdot r}{1 - 2^{-23}\cdot r}\right)\mathbf{u}$,

(iii) $(1+\boldsymbol{\epsilon}_1)^{1/2} = 1 + \boldsymbol{\epsilon}_5$, $|\boldsymbol{\epsilon}_5| = \dfrac{1}{1+\sqrt{1+\boldsymbol{\epsilon}_1}}\,|\boldsymbol{\epsilon}_1| \le \dfrac{p}{1+\sqrt{1-2^{-23}\cdot p}}\,\mathbf{u}$.

Lemma 3.3 *If* $\mathit{sign}(x) = \mathit{sign}(y)$, *then*

$$(1+\alpha)x + (1+\beta)y = (1+\gamma)(x+y), \quad |\gamma| \le \max\{|\alpha|,\ |\beta|\},$$

and consequently

$$(1+\alpha)x + y = (1+\gamma)(x+y), \quad \mathit{sign}(\gamma) = \mathit{sign}(\alpha), \quad |\gamma| \le \frac{x}{x+y}|\alpha| \le |\alpha|.$$

4 Accuracy Results

In the analysis, we have expressed the final rounding errors of c_φ, s_φ, c_ψ, s_ψ and q (where $f' = q \cdot f$, $h' = h/q$) via the errors of some auxiliary quantities which are computed earlier in the analysis. In particular, $\boldsymbol{\epsilon}_{c1}$ and $\boldsymbol{\epsilon}_{s1}$ are expressed as functions of $\boldsymbol{\epsilon}_\xi$, where ξ is the tangent (or cotangent) of the double angle in the above formulas. Similarly, $\boldsymbol{\epsilon}_{c2}$ and $\boldsymbol{\epsilon}_{s2}$ are expressed as functions of $\boldsymbol{\epsilon}_{t2}$ where $t2$ is tangent (or cotangent) of the second angle. Here $c1$ and $c2$ ($s1$ and $s2$) are cosines (sines) of angles which are computed as first and as second in the algorithm. A similar analysis is made for the $\star$`LASV2` algorithm. The results are displayed in the tables below. Since in applications, typically in Kogbetliantz method for n-by-n triangular matrices, most frequently the both angles will be small, we have added the bounds obtained for the case when the both tangents are smaller than 0.1.

error	**upper bounds**	
	general	$\boldsymbol{\lvert t1\rvert, \lvert t2\rvert \leq 1/10}$
$\lvert\boldsymbol{\epsilon}_{c1}\rvert$	$2.82\,\mathbf{u}$	$2.25\,\mathbf{u}$
$\lvert\boldsymbol{\epsilon}_{s1}\rvert$	$10.44\,\mathbf{u}$	$10.28\,\mathbf{u}$
$\lvert\boldsymbol{\epsilon}_{c2}\rvert$	$8.771\,\mathbf{u}$	$2.871\,\mathbf{u}$
$\lvert\boldsymbol{\epsilon}_{s2}\rvert$	$14.791\,\mathbf{u}$	$14.791\,\mathbf{u}$
$\lvert\boldsymbol{\epsilon}_{q}\rvert$	$9.4551\,\mathbf{u}$	$3.312\,\mathbf{u}$

Modified Algorithm

error	**upper bounds**	
	general	$\boldsymbol{\lvert t1\rvert, \lvert t2\rvert \leq 1/10}$
$\lvert\boldsymbol{\epsilon}_{c1}\rvert$	$30.01\,\mathbf{u}$	$11.53\,\mathbf{u}$
$\lvert\boldsymbol{\epsilon}_{s1}\rvert$	$23.01\,\mathbf{u}$	$20.06\,\mathbf{u}$
$\lvert\boldsymbol{\epsilon}_{c2}\rvert$	$20.01\,\mathbf{u}$	$3.15\,\mathbf{u}$
$\lvert\boldsymbol{\epsilon}_{s2}\rvert$	$20.01\,\mathbf{u}$	$17.03\,\mathbf{u}$
$\lvert\boldsymbol{\epsilon}_{q}\rvert$	$6.01\,\mathbf{u}$	$6.01\,\mathbf{u}$

$\star$`LASV2` **Algorithm**

We see that somewhat better accuracy bounds are obtained for the new algorithm. The best gain is obtained for the bound of $\boldsymbol{\epsilon}_{c1}$ when both angles are small. This is important since the general Kogbetliantz transformation will include the transformation of the form $x' = c1 \cdot x \pm s1 \cdot y$. For small off-diagonal elements x, y and small $s1$, the error $\boldsymbol{\epsilon}_{c1}$ will give the main contribution to the relative error of x'.

References

[1] Bojanczyk A., Ewerbring L., Luk F., and van Dooren P., An accurate product SVD algorithm, Signal Processing 25 (1991) 189–201.
[2] Drmač Z., A posteriori computation of the singular vectors in a preconditioned Jacobi SVD algorithm, IMA J. Numer. Anal. 19 (1999) 191-213.
[3] Drmač Z. and Veselić K., New fast and accurate Jacobi SVD algorithm I, LAPACK Working note 169, 2005.
[4] Drmač Z. and Veselić K., New fast and accurate Jacobi SVD algorithm II, LAPACK Working note 170, 2005.
[5] Forsythe G. E. and Henrici P., The cyclic Jacobi method for computing the principal values of a complex matrix, Trans. Amer. Math. Soc. 94 (1960) 1–23.
[6] Hari V., On the Quadratic Convergence of the Serial SVD Jacobi Methods for Triangular Matrices, SIAM J. Sci. Stat. Comput. Vol. 10, No. 6 (1989) 1076–1096.
[7] Hari V. and Veselić K., On Jacobi methods for singular value decompositions, SIAM J. Sci. Stat. Comput. Vol. 8, No. 5 (1987) 741–754.
[8] Hari V., On Sharp Quadratic Convergence Bounds for the Serial Jacobi Methods, Numer. Math. 60 (1991) 375-406.
[9] Hari V. and Matejaš J.: Quadratic Convergence of Scaled Iterates by Kogbetliantz Method, Computing [Suppl] 16 (2003) 83-105.
[10] Hari V., Butterfly Matrices and the Modulus Kogbetliantz Method, ICNAAM 2005 - International Conference on Numerical Analysis and Applied Mathematics 2005 / Edts. Simos, T. E.; Psihoyios G.; Tsitouras, G. Weinheim: Wiley-VCH, 2005, 226-229.
[11] Hari V. and Zadelj-Martić V., Parallelizing Kogbetliantz Method, Proposed for publications in Journal of Numerical Analysis, Inustrial and Applied Mathematics, pp. 1–14.
[12] Hari V. and Matejaš J., Accuracy of the Kogbetliantz method, Preprint, University of Zagreb 2006.
[13] Kogbetliantz E., Diagonalization of General Complex Matrices as a New Method for Solution of Linear Equations, Proc. Imtern. Congr. Math. Amsterdam 2 (1954) 356–357.
[14] Kogbetliantz E., Solutions of Linear Equations by Diagonalization of Coefficient Matrices, Quart. Appl. Math. 13 (1955) 123–132.
[15] Londre T. and Rhee N. H., Numerical Stability of the Parallel Jacobi Method. SIAM J. Mat. Anal. Appl. Vol. 26 No. 4 (2005) 985–1000.
[16] Matejaš J., Hari V., Scaled Iterates by Kogbetliantz Method, Proceedings of the 1st Conference on Applied Mathematics and Computations, Dubrovnik, Croatia, September 13–18, 1999; Publisher Dept. of Mathematics, University of Zagreb, 2001, pp. 1–20.
[17] Voevodin V. V., Cislennye metody linejnoj algebry, Nauka, Moscow (in Russian) 1966.