Algorithm 977: A QR–Preconditioned QR SVD Method for Computing the SVD with High Accuracy

ZLATKO DRMAČ, Faculty of Science, University of Zagreb

A new software for computing the singular value decomposition (SVD) of real or complex matrices is proposed. The method implemented in the code xGESVDQ is essentially the QR SVD algorithm available as xGESVD in LAPACK. The novelty is an extra step, the QR factorization with column (or complete row and column) pivoting, also already available in LAPACK as xGEQP3. For experts in matrix computations, the combination of the QR factorization and an SVD computation routine is not new. However, what seems to be new and important for applications is that the resulting procedure is numerically superior to xGESVD and that it is capable of reaching the accuracy of the Jacobi SVD. Further, when combined with pivoted Cholesky factorization, xGESVDQ provides numerically accurate and fast solvers (designated as xPHEVC, xPSEVC) for the Hermitian positive definite eigenvalue problem. For instance, using accurately computed Cholesky factor, xPSEVC computes all eigenvalues of the 200×200 Hilbert matrix (whose spectral condition number is greater that 10^{300}) to nearly full machine precision. Furthermore, xGESVDQ can be used for accurate spectral decomposition of general (indefinite) Hermitian matrices.

CCS Concepts: • **Mathematics of computing → Computations on matrices;**

Additional Key Words and Phrases: Accuracy, condition number, Jacobi method, pivoting, SVD

ACM Reference Format:
Zlatko Drmač. 2017. Algorithm 977: A QR–preconditioned QR SVD method for computing the SVD with high accuracy. ACM Trans. Math. Softw. 44, 1, Article 11 (July 2017), 30 pages.
DOI: http://dx.doi.org/10.1145/3061709

1. INTRODUCTION AND PRELIMINARIES

In this report, we propose a new way to compute the singular value decomposition (SVD) of general matrices using the LAPACK library [Anderson et al. 1999]. The main features of the new software are the following:

(1) Simple modular design, based on the existing LAPACK subroutines xGESVD (or xGESDD), xGEQP3, and xGEQRF. It allows simple replacement of xGESVD with another (bidiagonalization-based) SVD subroutine, so it can be considered as a universal framework, or an expert driver routine, for enhancing numerical robustness of the existing highly optimized codes that are already available in high-performance numerical libraries such as -the MKL [Intel 2015]. With respect to runtime efficiency, as compared to xGESVD and xGESDD alone, the new code has an overhead of one QR factorization with pivoting, which is used as a preconditioner.
(2) Numerical robustness, in which the new code not only provably matches the classical backward stability of xGESVD (and xGESDD) but exceeds it to the extent of

This work was supported by the Croatian Science Foundation under grant HRZZ–IP-11-2013-9345.
Author's address: Z. Drmač, Department of Mathematics, Faculty of Science, Bijenicka 30, 10000 Zagreb, Croatia; email: drmac@math.hr.

DOI: http://dx.doi.org/10.1145/3061709

reaching the high accuracy of the Jacobi SVD method. This claim is supported by overwhelming numerical evidence and partial theoretical insights. For instance, the new code computes all singular values of the accurately computed Cholesky factor (see Demmel [1999]) of the Hilbert matrix of dimension $n = 200$, with the spectral condition number above 10^{300}, to 14 digits of accuracy in 16-digit arithmetic. Providing a theoretical basis for this exceptional accuracy is a new challenging problem for numerical analysis of the proposed method.

The seemingly simple design of the new code is built on rather subtle details with far-reaching consequences, yielding a superior mathematical software for the key matrix decomposition. Theoretical foundations that motivate and intuitively explain numerical robustness of the algorithm are already available in the literature, and in this report we build our case upon them. In essence, we provide no new theory, but we do provide new insights and new technology for the matrix computations software development. This report is intended as accompanying blueprints of the software, and it can also serve as a mini-tutorial on finite precision SVD computation.

To set the scene, in the rest this section we briefly describe the two main approaches to SVD computation—the bidiagonalization-based methods and the Jacobi SVD method—and we point out the key differences in the structures of their backward and forward errors. We prefer informal and less technical discussion with the intention to provide the key insights that lead to more accurate SVD computation. Interested readers will be able to fill in the details using the provided references. Section 2 contains the new proposed algorithm, with a discussion and intuitive explanation of the increased numerical accuracy that is illustrated on the 200×200 Hilbert matrix. The results of systematic testing of the new software are presented in Section 3. In our opinion, they build a strong case for recommending the new subroutine as an enhancement of the existing LAPACK subroutines `xGESVD` and `xGESDD`. Finally, in Section 4, we show how the new technique applies to solving Hermitian positive definite eigenvalue problems. For the reader's convenience, in Section 5 we use error analysis and simple case study examples to illustrate why orthogonality of the deployed transformations may not be enough to compute the smallest singular values to high relative accuracy.

1.1. Bidiagonalization-Based SVD Computation

Let A be an $m \times n$ complex matrix. Without loss of generality, we assume that $m \geq n$. Currently, the most efficient numerical methods for computing the SVD $A = U\left(\begin{smallmatrix}\Sigma\\0\end{smallmatrix}\right)V^*$ first reduce A to bidiagonal form [Golub and Kahan 1965],

$$\begin{pmatrix} B \\ 0 \end{pmatrix} = X^* A Y, \quad X, Y \ \text{ unitary}, \quad B = \begin{pmatrix} \alpha_1 & \beta_1 & & \\ & \alpha_2 & \ddots & \\ & & \ddots & \beta_{n-1} \\ & & & \alpha_n \end{pmatrix}, \tag{1}$$

and in the next step, the SVD of B is computed as $B = W\Sigma Z^*$. The SVD of A is then assembled as

$$A = X\begin{pmatrix} W & 0 \\ 0 & I \end{pmatrix}\begin{pmatrix} \Sigma \\ 0 \end{pmatrix}(YZ)^* \equiv U\begin{pmatrix} \Sigma \\ 0 \end{pmatrix}V^*, \quad \Sigma = \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{pmatrix}, \quad \sigma_1 \geq \cdots \geq \sigma_n.$$

Since only unitary transformations are involved, the preceding procedure is deemed backward stable, and the backward stability is usually measured in the (unitarily invariant) matrix norms $\|\cdot\|_F$ and $\|\cdot\|_2$. More precisely, if the computed matrices are

marked with tildes ($\widetilde{X}$, $\widetilde{Y}$, $\widetilde{B}$, $\widetilde{W}$, $\widetilde{Z}$, $\widetilde{\Sigma}$), then there exists a backward error ΔA such that

$$A + \Delta A = \widetilde{X}\begin{pmatrix} \widetilde{B} \\ 0 \end{pmatrix}\widetilde{Y}^*, \quad \widetilde{X}, \widetilde{Y} \text{ numerically unitary}, \quad \frac{\|\Delta A\|_F}{\|A\|_F} \leq \epsilon_1. \tag{2}$$

Furthermore, the computed SVD of $\widetilde{B}$ can be modeled as

$$\widetilde{B} + \Delta\widetilde{B} = \widetilde{W}\widetilde{\Sigma}\widetilde{Z}^*, \quad \widetilde{W}, \widetilde{Z} \text{ numerically unitary}, \quad \frac{\|\Delta\widetilde{B}\|_F}{\|\widetilde{B}\|_F} \leq \epsilon_2. \tag{3}$$

Here, *numerically unitary* means that the $m \times m$ matrix $\widetilde{X}$ satisfies $\widetilde{X}^*\widetilde{X} = I_m + E$ with $\|E\|_2 \leq O(m)\boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon}$ denotes the roundoff unit. Note that in this case, $\|\widetilde{X}\|_2 \approx 1 + O(m)\boldsymbol{\varepsilon}$, $\|\widetilde{X}^{-1}\|_2 \approx 1 + O(m)\boldsymbol{\varepsilon}$, and $\widetilde{X}^{-1} = \widetilde{X}^* + O(m)\boldsymbol{\varepsilon}$. Both ϵ_1 and ϵ_2 are bounded by the roundoff times' modestly growing functions of the dimensions. The composite backward error of (2) and (3) can be written as

$$A + \underbrace{\Delta A + \widetilde{X}\begin{pmatrix} \Delta\widetilde{B} \\ 0 \end{pmatrix}\widetilde{Y}^*}_{\widetilde{\Delta A}} = \widetilde{X}\begin{pmatrix} \widetilde{W} & 0 \\ 0 & I \end{pmatrix}\begin{pmatrix} \widetilde{\Sigma} \\ 0 \end{pmatrix}(\widetilde{Y}\widetilde{Z})^*, \tag{4}$$

and it is bounded in matrix norm as

$$\|\widetilde{\Delta A}\|_F \leq (\epsilon_1 + \epsilon_2\|\widetilde{X}^{-1}\|_2\|\widetilde{Y}^{-1}\|_2 + \epsilon_1\epsilon_2\|\widetilde{X}^{-1}\|_2\|\widetilde{Y}^{-1}\|_2)\|A\|_F. \tag{5}$$

The two most popular methods of this type are implemented in LAPACK as `xGESVD` (the QR method) and `xGESDD` (the divide and conquer method). Relations (4) and (5) express the fact that the SVD computation is mixed stable: the computed decomposition $A + \widetilde{\Delta A} = \widetilde{U}\left(\begin{smallmatrix} \widetilde{\Sigma} \\ 0 \end{smallmatrix}\right)\widetilde{V}^*$ is close to an SVD of $A + \widetilde{\Delta A}$. Strictly speaking, this is not an SVD because $\widetilde{U} \equiv \mathit{computed}(\widetilde{X}\left(\begin{smallmatrix} \widetilde{W} & 0 \\ 0 & I \end{smallmatrix}\right))$ and $\widetilde{V} \equiv \mathit{computed}(\widetilde{Y}\widetilde{Z})$ are only numerically unitary.

Remark 1.1. If A is a tall and skinny ($m \gg n$), then it is more efficient to first compute the QR factorization of A, $A = Q\left(\begin{smallmatrix} R \\ 0 \end{smallmatrix}\right)$ and then to apply the preceding two-step procedure to the upper triangular matrix R and adjust the left singular vectors using the unitary factor Q (see Chan [1982]). The optimal ratio m/n for this strategy can be tuned for a particular algorithm and a computing environment (hardware, compiler); in `xGESVD`, the crossover point is obtained by calling, for example, `ILAENV( 6, 'CGESVD', JOBU // JOBVT, M, N, 0, 0 )`.

It is important to note that in `xGESVD`, the SVD (3) is computed, by `xBDSQR` (implicit zero-shift QR algorithm [Demmel and Kahan 1990]), to high relative accuracy: the values $\widetilde{\sigma}_i = \widetilde{\Sigma}_{ii}$, arranged as $\widetilde{\sigma}_1 \geq \cdots \geq \widetilde{\sigma}_n$, are, for all i's, to nearly machine precision accurate approximations of the exact singular values $\widehat{\sigma}_1 \geq \cdots \geq \widehat{\sigma}_n$ of $\widetilde{B}$. More precisely, the computed values satisfy rather sharp forward error bound:[1]

$$|\widetilde{\sigma}_i - \widehat{\sigma}_i| \leq \gamma n\boldsymbol{\varepsilon}\widehat{\sigma}_i, \quad i = 1, \ldots, n. \tag{6}$$

Further, the singular vectors are also accurate to the limits dictated by the perturbation theory. It has been conjectured [Demmel and Kahan 1990] (based on overwhelming numerical evidence and perturbation theory) that the angular errors in the computed singular vectors $\widetilde{w}_i$, $\widetilde{z}_i$, as compared to the exact singular vectors $\widehat{w}_i$, $\widehat{z}_i$ of $\widetilde{B}$, obey the

[1]Here, γ is a moderate factor that depends on the number of QR iterations. For detailed analysis, we refer the reader to Demmel and Kahan [1990].

bound

$$\max\{\theta(\widetilde{w}_i, \widehat{w}_i), \theta(\widetilde{z}_i, \widehat{z}_i)\} \leq \frac{p(n)\boldsymbol{\varepsilon}}{gap_i}, \quad gap_i = \min\left\{\frac{\widehat{\sigma}_{i-1} - \widehat{\sigma}_i}{\widehat{\sigma}_i}, \frac{\widehat{\sigma}_i - \widehat{\sigma}_{i+1}}{\widehat{\sigma}_i}\right\}. \tag{7}$$

Unfortunately, the backward error ΔA in (2) may be such that the SVD of $A+\Delta A$ is not necessarily an accurate approximation of the SVD of A. This is particularly the case if one considers the errors in the singular values σ_i of A that are much smaller than $\|A\|_2$. Namely, by the classical perturbation result (see Mirsky's theorem in Section 4.3 of [1990]), $|\sigma_i(A+\Delta A) - \sigma_i(A)| \leq \|\Delta A\|_2$—that is, for each $\sigma_i(A) \neq 0$,

$$\frac{|\sigma_i(A+\Delta A) - \sigma_i(A)|}{\sigma_i(A)} \leq \frac{\|\Delta A\|_2}{\|A\|_2}\frac{\|A\|_2}{\sigma_i(A)} \leq \kappa_2(A)\frac{\|\Delta A\|_2}{\|A\|_2}, \tag{8}$$

where $\kappa_2(A) = \|A\|_2\|A^\dagger\|_2$ is the spectral condition number of A. Obviously, for $\sigma_i(A) \ll \|A\|_2$, the backward error $\|\Delta A\|_2/\|A\|_2$ may be amplified by a potentially large factor. What is guaranteed is that for all i,

$$\frac{|\sigma_i(A+\Delta A) - \sigma_i(A)|}{\|A\|_2} \leq \frac{\|\Delta A\|_2}{\|A\|_2}. \tag{9}$$

If A is given with an initial uncertainty $\Delta_0 A$ such that the only available information is that $\|\Delta_0 A\|_F \leq \epsilon\|A\|_F$, then are acceptable accuracy and the singular values are computed within the accuracy warranted by the data.

1.2. Preconditioned Jacobi SVD Method

Jacobi's method is known as numerically robust and more accurate than QR [Demmel and Veselić 1992]. An improved version of the method [Drmač and Veselić 2008a], Drmač and Veselić [2008b] uses the QR factorization with pivoting as a preconditioner that facilitates faster convergence of the one-sided Jacobi process. We will describe a simplified version of the method to point out the key features that make the method more accurate. For detailed description and numerical analysis, we refer the reader to Demmel and Veselić [1992], Drmač and Veselić [2008a], Drmač and Veselić [2008b], and the LAPACK codes `xGESVJ`, `xGEJSV`. Let

$$\Pi_r A \Pi_c = Q\begin{pmatrix} R \\ 0 \end{pmatrix}, \quad Q^*Q = QQ^* = I_m, \quad R = \begin{pmatrix} r_{11} & \cdots & r_{1n} \\ & \ddots & \vdots \\ & & r_{nn} \end{pmatrix} \tag{10}$$

be the QR factorization, where we can choose, besides $\Pi_r = I_m$ and $\Pi_c = I_n$, one of the following pivot strategies that determine the permutation matrices Π_r, Π_c:

(1) $\Pi_r = I_m$, Π_c determined by the Businger-Golub pivoting [Businger and Golub 1965]. In LAPACK, this factorization is implemented in xGEQP3 as a BLAS 3 level routine. (See Drmač and Bujanović [2008] for a robust implementation.)
(2) Π_r is the permutation matrix that sorts the rows of A in decreasing order in the ℓ_∞ norm (Björck's row pivoting), and Π_c is column permutation matrix determined by the Businger-Golub pivoting in the QR factorization of $\Pi_r A$.
(3) Π_r and Π_c are determined in a complete pivoting [Powell and Reid 1969]. For a list of various other pivot strategies, see Section 3.3.2.

Now consider the one-sided Jacobi SVD applied to $L = R^*$. A sequence of Jacobi rotations $\mathcal{J}_1, \mathcal{J}_2, \ldots$ is applied from the right, $L\mathcal{J}_1\mathcal{J}_2\cdots$, giving in the limit $LJ = W\Sigma$,

$J = \mathcal{J}_1\mathcal{J}_2\cdots$, and the SVD of R^* is $R^* = W\Sigma J^*$. The computed SVD of A reads

$$A = (\Pi_r^T Q)\begin{pmatrix} J & 0 \\ 0 & I \end{pmatrix}\begin{pmatrix} \Sigma \\ 0 \end{pmatrix}(\Pi_c W)^* \equiv U\begin{pmatrix} \Sigma \\ 0 \end{pmatrix}V^*. \tag{11}$$

Consider now the backward error in the preceding procedure. To ease the notation, assume that the matrix A has been prepermuted, so without loss of generality, $\Pi_r = I_m$, $\Pi_c = I_n$. In finite precision, we have the following: the computed QR factorization in the first step satisfies [Drmač 1994]

$$A + \delta A = \widetilde{Q}\begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}, \quad \|\delta A(:,i)\|_2 \le \boldsymbol{\varepsilon}_{qr}\|A(:,i)\|_2,\ i = 1,\ldots,n, \tag{12}$$

where $\boldsymbol{\varepsilon}_{qr}$ is a moderate multiple of the roundoff $\boldsymbol{\varepsilon}$. Note that the backward error in each column of A is small relative to the length of that column. This means that even the tiniest columns are preserved under backward perturbation, and it is due to the fact that in the Householder or Givens QR factorization, the columns are independently multiplied by a sequence of unitary matrices. In the next step, the application of the Jacobi rotations to $\widetilde{L} = \widetilde{R}^*$ can be represented as a transformation of backward perturbed $\widetilde{L} + \delta\widetilde{L}$:

$$(\widetilde{L} + \delta\widetilde{L})\widetilde{J} = \widetilde{W}\widetilde{\Sigma}, \quad \|\delta\widetilde{L}(i,:)\|_2 \le \boldsymbol{\varepsilon}_J\|\widetilde{L}(i,:)\|_2,\ i = 1,\ldots,n, \tag{13}$$

where $\widetilde{W}$, $\widetilde{J}$ are numerically unitary, $\widetilde{\Sigma} = \mathrm{diag}(\widetilde{\sigma}_i)_{i=1}^n$, and $\boldsymbol{\varepsilon}_J \le O(n)\boldsymbol{\varepsilon}$. Again, this row-wise backward error bound is due to the fact that postmultiplications by Jacobi rotations, although designed to orthogonalize the columns, can be considered as independent transformations of matrix rows that produce in each row backward error small relative to the length of that row.[2] If we set $\delta\widetilde{R} = (\delta\widetilde{L})^*$, then the two steps combined read

$$A + \underbrace{\delta A + \widetilde{Q}\begin{pmatrix} \delta\widetilde{R} \\ 0 \end{pmatrix}}_{\Delta A} = \widetilde{Q}\begin{pmatrix} \widetilde{J}^{-*} & 0 \\ 0 & I \end{pmatrix}\begin{pmatrix} \widetilde{\Sigma} \\ 0 \end{pmatrix}\widetilde{W}^*, \tag{14}$$

where $\widetilde{J}^{-*} \approx \widetilde{J}$ because of numerical orthogonality, and

$$\|\Delta A(:,i)\|_2 \le \boldsymbol{\varepsilon}_{qr}\|A(:,i)\|_2 + \boldsymbol{\varepsilon}_J\|\widetilde{Q}\|_2\|\widetilde{R}(:,i)\|_2 \tag{15}$$

$$\le \|A(:,i)\|_2\boldsymbol{\varepsilon}_{qr}^J, \quad \boldsymbol{\varepsilon}_{qr}^J = \boldsymbol{\varepsilon}_{qr} + \boldsymbol{\varepsilon}_J\|\widetilde{Q}\|_2\|\widetilde{Q}^{-*}\|_2(1+\boldsymbol{\varepsilon}_{qr}). \tag{16}$$

Clearly, $\boldsymbol{\varepsilon}_{qr}^J$ is a moderate multiple of the roundoff $\boldsymbol{\varepsilon}$. Now set $A = CD$, where $D = \mathrm{diag}(\|A(:,i)\|_2)_{i=1}^n$ and C has unit norm columns. In addition, set $\Delta A = (\Delta C)D$. To estimate the change of the singular values due to ΔA, write $A + \Delta A = (I + \Delta AA^\dagger)A$, and by the multiplicative perturbation theory (e.g., see Eisenstat and Ipsen [1995]),

$$\frac{|\sigma_i(A+\Delta A) - \sigma_i(A)|}{\sigma_i(A)} \le \|\Delta AA^\dagger\|_2 = \|\Delta CC^\dagger\|_2 \tag{17}$$

$$\le \|\Delta C\|_2\|C^\dagger\|_2 = \frac{\|\Delta C\|_2}{\|C\|_2}\kappa_2(C). \tag{18}$$

[2]See Section 5.1.

Now note the following two important facts:

(1) Since $\kappa_{scaled}(A) \equiv \kappa_2(C) \leq \sqrt{n} \min_{D=\mathrm{diag}} \kappa_2(AD) \leq \sqrt{n}\kappa_2(A)$ (see van der Sluis [1969]), $\kappa_{scaled}(A)$ is never much larger and it can be much smaller than $\kappa_2(A)$.
(2) $1 \leq \|C\|_2 \leq \sqrt{n}$ and, by (16), $\|\Delta C\|_2 \leq \sqrt{n}\boldsymbol{\varepsilon}_{qr}^J$.

The advantage of (18) over (8) and (9) is obvious. In particular, the error bound (18) is independent of $\kappa_2(D)$, so ill conditioning caused by column scaling has no influence on the accuracy of the Jacobi SVD, although it can seriously damage the accuracy of the bidiagonalization-based SVD method. In fact, the larger the spread among the column norms of A, the faster will be the convergence of the Jacobi iterations applied to R^*. In the extreme case, one sweep of one-sided Jacobi rotations numerically behaves as the Gram-Schmidt orthogonalization [Drmač 1997], thus yielding swift convergence.

Remark 1.2. It is an important fact that the backward error bounds (12) and (13) and the composite bound (16) remain valid for tiled, blocked, and parallel implementations of the QR factorization and of the one-sided Jacobi SVD algorithm, independent of the parallel pivot strategies and actual order of execution. The reason is simple: unitary transformations are always from the side that is opposite to scaling, so the large and the small columns (or rows) are never mixed; instead, they are transformed concurrently and independently. And, actually, even if we do the analysis in the column space, the Jacobi rotation, as the right singular vector matrix of the pair of pivot columns, will be gentle to the smaller column. This is an important aspect of the Jacobi algorithm, and for the reader's convenience we illustrate it in detail in Section 5.

Remark 1.3. For deeper insight into how the rank revealing QR factorization may impact an SVD numerical algorithm, we refer the reader to Stewart [1997a, 1997b] and Drmač and Veselić [2008a, 2008b].

Remark 1.4. Here, *preconditioner* is not an oxymoron. In the one-sided Jacobi SVD process $LJ = W\Sigma$, in the limit we have $\kappa_{scaled}(LJ) = 1$, and $L = R^*$ usually has moderate $\kappa_{scaled}(L)$ and $\kappa_{scaled}(L) \ll \kappa_{scaled}(A)$ if $\kappa_{scaled}(A) \ll \kappa_2(A)$. Hence, the use of this term is not in the traditional and narrow sense of reducing the spectral condition number. The key here is that this step reduces the scaled condition number, and the classical spectral condition number remains unchanged.

1.2.1. The Structure of R. For the sake of completeness and for the reader's convenience, we briefly review the key role of the rank revealing QR factorization (10). For more details, we refer the reader to Section 2 of Drmač and Veselić [2008a, 2008b] and Section 6 of Drmač and Bujanović [2008]. The column pivoting [Businger and Golub 1965] results in a triangular factor R that has particularly strong diagonal dominance:

$$|R_{ii}| \geq \sqrt{\sum_{k=i}^{j} |R_{kj}|^2}, \quad \text{for all } \ 1 \leq i \leq j \leq n. \tag{19}$$

For our purposes, this structure yields two important facts. The first is that the one-sided Jacobi SVD applied to R^* will tend to use rotations with smaller angles and that the right singular vectors of R^* (infinite product of Jacobi rotations) have their dominant components distributed in a particular way. This is related to the structure of the eigenvectors of quasidefinite matrices. To illustrate, introduce the block partition $R = \left(\begin{smallmatrix} R_{[11]} & R_{[12]} \\ 0 & R_{[22]} \end{smallmatrix}\right)$, where the diagonal blocks are $k \times k$ and $(n-k) \times (n-k)$, and assume that $\sigma_{\min}(R_{[11]} \ R_{[12]}) > \sigma_{\max}(R_{[22]})$. If A has strongly graded columns, such a strong separation of the diagonal blocks is likely for many values of $k \in \{1, \ldots, n\}$. As a

consequence, the cross-product matrix

$$RR^* = \left(\begin{array}{c|c} R_{[11]}R_{[11]}^* + R_{[12]}R_{[12]}^* & R_{[12]}R_{[22]}^* \\ \hline R_{[22]}R_{[12]}^* & R_{[22]}R_{[22]}^* \end{array}\right) \equiv H = \left(\begin{array}{c|c} H_{[11]} & H_{[12]} \\ \hline H_{[12]}^* & H_{[22]} \end{array}\right)$$

can be shifted with $\zeta \in (\lambda_{\max}(H_{[22]}), \lambda_{\min}(H_{[11]}))$ so that $H - \zeta I_n$ is quasidefinite. Hence, in the eigenvector matrix of H, partitioned in the same way, the diagonal blocks dominate the off-diagonal ones in the Löwner partial order [George et al. 2000]. Since this is likely to hold for many k's, the absolutely largest values in the eigenvector matrix of RR^* (the left singular vector matrix of R) will be closer to the diagonal.

The second important fact is that a very strong preconditioning effect takes place. Namely, if we write $R = D_r R_r$, where $D_r = \mathrm{diag}(|\mathrm{R_{ii}}|)$ or $D_r = \mathrm{diag}(\|\mathrm{R(i,:)}\|_2)$, then $\kappa_2(R_r)$ is moderate, possibly much smaller than $\kappa_2(C)$. In fact, it is possible to bound $\kappa_2(R_r)$ with a function of n, independent of A (see Drmač [1999] and Drmač and Veselić [2008a, 2008b]).

1.3. Trade-Off Between Accuracy and Speed

Intensive research and development of bidiagonalization routines and bidiagonalization-based codes that exploit multiprocessor architectures, multicore processors, memory hierarchies, blocking, tiling, and other techniques have resulted in amazingly increased efficiency of the reduction to bidiagonal form and of the bidiagonal SVD subroutines (e.g., see Ltaief et al. [2013]). In highly optimized libraries such as Intel's Math Kernel Library (MKL), bidiagonalization-based codes are faster than the Jacobi SVD, except in cases of particular classes of matrices.

In this situation, the QR method (`xGESVD`) is not the most accurate, and it is not the fastest; in Ltaief et al. [2013], it is even deemed deprecated. However, multiprocessor/multicore-based optimization of the Jacobi SVD methods remains an important task for future work, and its high accuracy appears to be more expensive compared to the bidiagonalization-based methods in high-performance libraries (e.g., MKL) whose efficiency seems hard to beat. This may change in the future; recent work [Bečka et al. 2015] shows promising advances (e.g., a ScaLAPACK-type implementation of Jacobi SVD outperforms `PxGESVD`).

With ever-increasing demand for efficient numerical codes capable of handling large-scale computational tasks, numerical reliability seems to be a less attractive feature—code optimization on new computing hardware, using modern software tools, programming paradigms, and runtime optimization techniques are in the focus of several strong research groups. However, numerical accuracy, robustness, and reliability of mathematical software are important properties that should carry a lot of weight when assessing the quality of the code. It would be ideal to get the highest accuracy warranted by the data in the shortest possible runtime. Unfortunately, the trade-off between accuracy and speed seems unavoidable, and speed is outrunning the accuracy. In Kahan's formulation [Kahan 2008] of Gresham's law for computing, the fast drives out the slow even if the fast is wrong.

2. A PROPOSAL FOR MODIFICATION OF XGESVD

Our goal is set to exploit the tremendous efforts and excellent results of the work on efficient bidiagonalization-based methods and to improve their numerical properties at an acceptable price in runtime overhead. In particular, `xGESVD` uses an accurate bidiagonal SVD subroutine `xBDSQR`, and its overall accuracy essentially depends on the bidiagonalization (`xGEBRD`). Providing more accurate bidiagonalization would result in better overall SVD computation, and other methods, such asthose of Großer and Lang [2003], might also benefit from our proposed approach.

2.1. Back to Bidiagonalization: Modifications

In his Algorithm 6.1, Barlow [2002] was the first to attempt and partially succeed to make bidiagonalization provably backward stable in a stronger sense that would facilitate a more accurate SVD. The key step of the new bidiagonalization is the initial reduction to triangular form via the QR factorization with complete pivoting (10). Then, a specially designed sequence of Givens rotations reduces $L = R^*$ to bidiagonal form. The analysis of the proposed procedure is rather complicated and the final error bounds are better than (2), weaker than for the Jacobi SVD, but still provide improved theoretical insight. The accuracy of the SVD based on the new bidiagonalization is illustrated by successfully computing the singular values of the Cholesky factors of Hilbert matrices of dimensions up to 90. It should be noted that the new and more accurate bidiagonalization requires the use of Givens rotations that include pivot search, and that the estimated flops overhead is $O(n^3)$. This means that its performance is not nearly as high as of the `xGEBRD` subroutine in LAPACK.

Ralha [2003] introduced the concept of one-sided bidiagonalization, which was further analyzed in Barlow et al. [2005] and Bosner and Drmač [2005]. The one-sided bidiagonalization-based SVD is shown to possess the potential of computing to high relative accuracy, with the proviso that certain orthogonality issues are properly resolved.

Unfortunately, none of these potentially accurate SVD methods have matured to the level of a library subroutine suitable for including in LAPACK, for example. However, it is clear that there is a potential for producing a bidiagonalization that, combined with an accurate bidiagonal SVD, yields a more accurate SVD. For a matrix computations researcher, this is an exciting challenge.

2.2. The QR-Preconditioned xGESVD

The method proposed in this report offers a simple but far-reaching modification of the existing tuned code `xGESVD`. We just add one more elementary building block that already exists as a high-performance LAPACK subroutine—the QR factorization with column pivoting, implemented in `xGEQP3`, and, optionally, we deploy an initial Björck's row sorting as an efficient emulation of the full Powell-Reid pivoting [Powell and Reid 1969]. Thus, the new version of `xGESVD` can be described as follows:

> *Compute the QR factorization (10) with pivoting as implemented in `xGEQP3`, apply `xGESVD` to R (or R^*, whichever is more convenient or can be better optimized in the software), and assemble the SVD of A in the straightforward way.*

There seems to be nothing new in this. As pointed out in Remark 1.1, the QR factorization is often used to improve the efficiency of the bidiagonalization in the case of a tall and skinny matrix; here, pivoting is required. Further, (10) has been used in Barlow [2002] as the key preprocessing step for the new pivoted Givens bidiagonalization. The point we make here is that (10) alone is enough to facilitate more accurate bidiagonalization. In other words, a numerically superior version of `xGESVD` is readily available, and it only requires (10) as a preconditioner.[3] This is important for matrix computations software development and to our best knowledge is new.

A generic description of the preconditioned `xGESVD` is given in Algorithm 1. For efficiency and because of the availability of `xGEQP3`, the initial row sorting $A := \Pi_r A$ followed by the Businger-Golub column pivoting is used instead of full pivoting.

[3]After the factorization (10), the ensuing bidiagonalization of R or R^* is conditioned to exhibit certain numerical behavior.

Further, we leave two options: to apply xGESVD to R or R^* to facilitate experimenting with numerical accuracy and runtime efficiency (e.g., convergence of QR iterations).

If $m \gg n$, we allow to first compute the QR factorization without pivoting and then compute another QR of the $n \times n$ triangular factor, but with pivoting. This is a small technical detail, for simplicity not included in Algorithm 1, if the implementer wants to reduce the overhead due to pivoting. It is straightforward to show that this would not change the backward stability. However, it is important to note that in the cases of high $\kappa_{scaled}(A)$ (e.g., if $\kappa_{scaled}(A) > 1/\varepsilon$), for best results the first contact of the method with the matrix A should be through row and column pivoted QR factorization.

ALGORITHM 1: $(U, \Sigma, V) =$ xGESVDQ$(A,$ method$)$ $(A \in \mathbb{C}^{m\times n}, m \geq n)$

1: $(\Pi_r A)\Pi_c = Q\begin{pmatrix} R \\ 0 \end{pmatrix}$ {Initial row sorting in order of decreasing ℓ_∞ norm, and QR factorization with pivoting (e.g., xGEP3) or full pivoting.}
2: Determine the numerical rank ρ of R and set $R_\rho = R(1:\rho, 1:n)$.
3: **if** $\rho = n$ and condition estimate needed **then**
4: $\kappa \approx \|(R_\rho \mathrm{diag}(1/\|R_\rho(:,i)\|_2))^{-1}\|_2$ {Use, for example, xPOCON and adjust to the norm $\|\cdot\|_2$.}
5: **end if**
6: **if** method = "upper" **then**
7: Compute the SVD $R_\rho = \widehat{U}\begin{pmatrix}\widehat{\Sigma} & 0_{\rho,n-\rho}\end{pmatrix}\widehat{V}^*$. {Use xGESVD}
8: The SVD of A is $A = \left[\Pi_r^T Q\begin{pmatrix}\widehat{U} & 0 \\ 0 & I_{m-\rho}\end{pmatrix}\right]\begin{pmatrix}\widehat{\Sigma} & 0_{\rho,n-\rho} \\ 0_{m-\rho,\rho} & 0_{m-\rho,n-\rho}\end{pmatrix}(\Pi_c\widehat{V})^*$
9: **else**
10: Compute the SVD $R_\rho^* = \widehat{U}\begin{pmatrix}\widehat{\Sigma} \\ 0_{m-\rho,\rho}\end{pmatrix}\widehat{V}^*$. {Use xGESVD}
11: The SVD of A is $A = \left[\Pi_r^T Q\begin{pmatrix}\widehat{V} & 0 \\ 0 & I_{m-\rho}\end{pmatrix}\right]\begin{pmatrix}\widehat{\Sigma} & 0_{\rho,n-\rho} \\ 0_{m-\rho,\rho} & 0_{m-\rho,n-\rho}\end{pmatrix}(\Pi_c\widehat{U})^*$
12: **end if**

Remark 2.1. We have analogously developed xGESDDQ, a preconditioned version of xGESDD. However, since in this report we focus on high relative accuracy, xGESVDQ seems more attractive and we drop the discussion of xGESDD, which will be available in a separate report. (The less satisfactory numerics of xGESDD are due not only to the bidiagonalization but also to the bidiagonal SVD implemented in xBDSDC.) In principle, the modification that we propose here improves the numerical accuracy of the bidiagonalization, and any bidiagonalization-based algorithm will benefit from it (e.g., Großer and Lang [2003]) and particularly if it computes the bidiagonal SVD with high accuracy.

2.2.1. On Bidiagonalizing R. In Section 5.3, we provide examples that identify one source of loss of accuracy during the bidiagonalization—a large rotation angle may increase the scaled condition number that will provoke cancellations in later steps. They also illustrate how simultaneous use of column and row scalings with the corresponding condition numbers may allow computing even the smallest singular values to high accuracy. Hence, chances for more accurate bidiagonalization increase if both the column-scaled and the row-scaled condition number of the input matrix are moderate, and if the bidiagonalizing orthogonal (unitary) transformations are in a sense gentle when mixing the columns or rows of different norms.

Intuitively, if the bidiagonalizing matrices X and Y were such that their largest entries (in absolute value) were closer to the diagonal and if A were correspondingly graded, then some intermediate jumps of the relevant scaled condition numbers could be avoided. The key observation is that this is very likely to happen if we replace A with its triangular factor from the pivoted QR factorization.

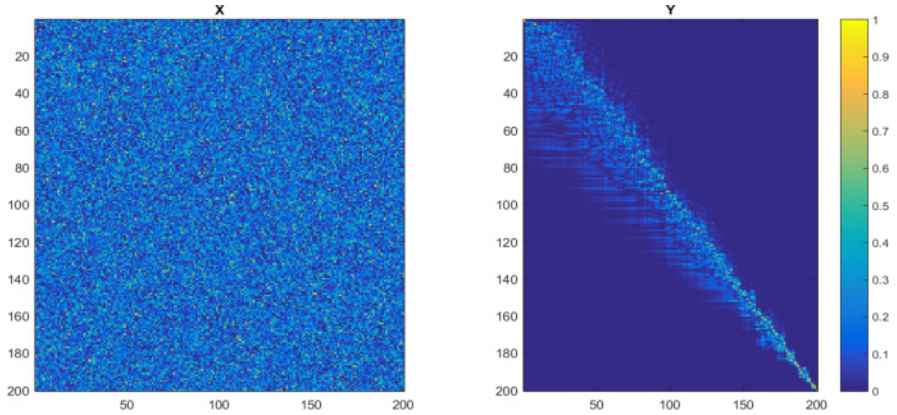

Fig. 1. (Example 2.2) The structure of the orthogonal matrices X and Y in the bidiagonalization $A(:,\zeta) = XBY^*$, where ζ is the permutation that sorts the columns of A in descending order in Euclidean norm.

Let $A \equiv \Pi_r A \Pi_c = CD$ as before, and let $R = X_1 B_1 Y_1^*$ be the bidiagonalization of R. If we write $R = TD$, then T has unit columns. From $X_1^* T = B_1 Y_1^* D^{-1}$, it follows that

$$\left| \frac{(B_1)_{ii}(Y_1)_{ji}}{D_{jj}} + \frac{(B_1)_{i,i+1}(Y_1)_{j,i+1}}{D_{jj}} \right| \leq 1. \tag{20}$$

If both A and D are highly ill conditioned, then (20) indicates that for $\sigma_i \gg D_{jj}$ (in which case is likely $|(B_1)_{ii}| \gg D_{jj}$), we should expect that $|(Y_1)_{ji}|$ is accordingly small.

Similarly, write $R = D_1 T_1$, where D_1 is diagonal and T_1 has unit rows. Alternatively, we may take $D_1 = \mathrm{diag}(|R_{ii}|)_{i=1}^n$. It is well known that the $|R_{ii}|$ (and also $(D_1)_{ii}$) and then the $|(B_1)_{ii}|$ will be distributed similarly to the singular values of A. As before, conclude that $|\frac{(X_1)_{i,j-1}(B_1)_{j-1,j}}{(D_1)_{ii}} + \frac{(X_1)_{ij}(B_1)_{jj}}{(D_1)_{ii}}| \leq \sqrt{n}$, and the corresponding entries of X_1 are expected to be small in modulus. Further note that with any block partition

$$\begin{pmatrix} R_{[11]} & R_{[12]} \\ 0 & R_{[22]} \end{pmatrix} = \begin{pmatrix} (X_1)_{[11]} & (X_1)_{12} \\ (X_1)_{[21]} & (X_1)_{[22]} \end{pmatrix} \begin{pmatrix} (B_1)_{[11]} & (B_1)_{12} \\ 0 & (B_1)_{[22]} \end{pmatrix} \begin{pmatrix} (Y_1)_{[11]} & (Y_1)_{12} \\ (Y_1)_{[21]} & (Y_1)_{[22]} \end{pmatrix}^*$$

of the bidiagonalization $R = X_1 B_1 Y_1^*$, we have (under certain assumptions)

$$(X_1)_{[21]} = R_{[22]}(Y_1)_{[21]}(Y_1)_{[11]}^{-1}(I + R_{[11]}^{-1}R_{[12]}(Y_1)_{[21]}(Y_1)_{[11]}^{-1})^{-1}R_{[11]}^{-1}(X_1)_{[11]} \tag{21}$$

and we see that $\|(X_1)_{[21]}\|_F \leq const \cdot (\sigma_{\max}(R_{[22]})/\sigma_{\min}(R_{[11]})) \cdot \|(Y_1)_{[21]}\|_F$. Note that rank reveling pivoting tends to compute R such that $\sigma_{\min}(R_{[11]}) > \sigma_{\max}(R_{[22]})$, and this particularly is the case if A is strongly graded.

Example 2.2. We illustrate the preceding using a random 200×200 matrix $A = CD$, where $\kappa_2(D) \approx 7 \cdot 10^{32}$ and $\kappa_2(C) \approx 2.8 \cdot 10^3$. We use Matlab's `imagesc(abs(.))` to visualize the structures of bidiagonalizing matrices (Figures 1 and 2). The color coding of `imagesc(abs(.))` clearly visualizes how introducing order (column sorting in Figure 1 and pivoted QR factorization in Figure 2) moves the absolutely larger entries (light colored) of the bidiagonalizing orthogonal matrices closer to the diagonal. For the importance of such a structure during bidiagonalization, see Section 5.

2.2.2. Is `xGESVDQ` *Better?* The new code `xGESVDQ` is provably always as good as `xGESVD` because the additional QR factorization introduces backward error that is bounded in the same way as (2), and it is actually better structured (see (12)). Hence, in the sense

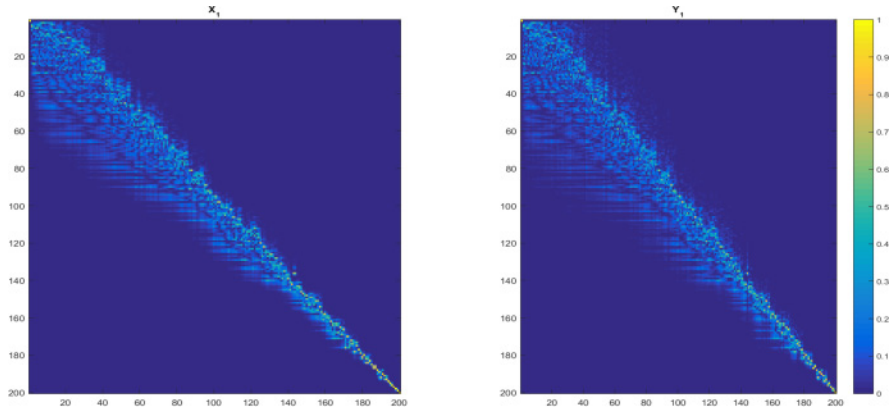

Fig. 2. (Example 2.2) The structure of the orthogonal matrices in the bidiagonalization $R = X_1 B_1 Y_1^*$.

of the analysis presented in Section 1.1, both routines have very similar accuracy and the smallest singular values will be recovered if $\kappa_2(A)$ is moderate (i.e., below $1/\boldsymbol{\varepsilon}$).

However, if $A = CD$ and $\kappa_2(A)$ is large (e.g., $\kappa_2(A) \gg 1/\boldsymbol{\varepsilon}$), we expect that better accuracy is feasible if $\kappa_2(C) \ll \kappa_2(A)$—that is, the ill conditioning of A is exposed in the diagonal scaling matrix $D = \mathrm{diag}(\|A(:,i)\|_2)_{i=1}^n$, which then has large $\kappa_2(D)$. Since D can be computed in $O(mn)$ flops, we can easily check whether D is ill conditioned. If $\kappa_2(D)$ is mild, then switching to `xGESVDQ` may not bring substantial improvement. If $\kappa_2(D)$ is large, then our thesis is that `xGESVDQ` will perform much better. We have only partial understanding of the exceptional accuracy of this modified QR SVD, based on intuition and computational experience combined with some theoretical insights. To support the claim about the advantage of using `xGESVDQ`, we give an extreme and instructive example.

Example 2.3. Take the Hilbert matrix of dimension $n = 200$, factor it as $H_{200} = GDG^*$ using Gaussian elimination with pivoting as in Demmel [1999] and set $A = G\sqrt{D}$ (i.e., A is a row permuted pivoted Cholesky factor). Here, $\kappa_2(G) \approx 138.35$ and $\kappa_2(\sqrt{D}) > 10^{151}$. Hence, $\kappa_{scaled}(A) \approx \kappa_2(G)$ is moderate, and A has, up to a permutation, the structure of the transposed R from Section 1.2.1. The computed matrices $\widetilde{G}$ and $\widetilde{D}$ are computed in a forward stable way, with entry-wise small relative errors. This means that in the working precision with roundoff $\boldsymbol{\varepsilon} \approx 2.2 \cdot 10^{-16}$, the Jacobi SVD method can *provably* compute all singular values of A, even the tiniest ones, to nearly 14 decimal digits of accuracy [Demmel 1999; Demmel et al. 1999].

We use Matlab function `svd()` (which is based on `xGESVD` from LAPACK) and compute the singular values $\sigma_i(A)$, and we also compute the bidiagonalization $A = XBY^T$. Then we compute the QR factorization with complete pivoting $A\Pi_c = QR$ and bidiagonalize the triangular factor R in two ways: as $R = X_1 B_1 Y_1^T$ and as $R^T = X_2 B_2 Y_2^T$. Then we compute (again using `svd`) the singular values $\sigma_i(B_1)$, $\sigma_i(B_2)$ of B_1, B_2, respectively. Remarkably, despite the range of the singular values from $\sigma_1 \approx 1.417$ to $\sigma_{200} \approx 1.046 \cdot 10^{-152}$, the three sets of values coincide to nearly the machine precision so that they are indistinguishable on the graph in Figure 3. In light of the discussion in Sections 2.2.1 and 5.3, it is instructive to inspect the structure of orthogonal matrices from the bidiagonalizations of A, R, and R^T (Figures 4 and 5).

We continue with the experiment as follows. Let ζ be a random permutation of $1, \ldots, n$. Let us replace A with $A(\zeta, :)$ and repeat the experiment by computing the singular values of A, R, B_1, and B_2. The accuracy of the computed singular values

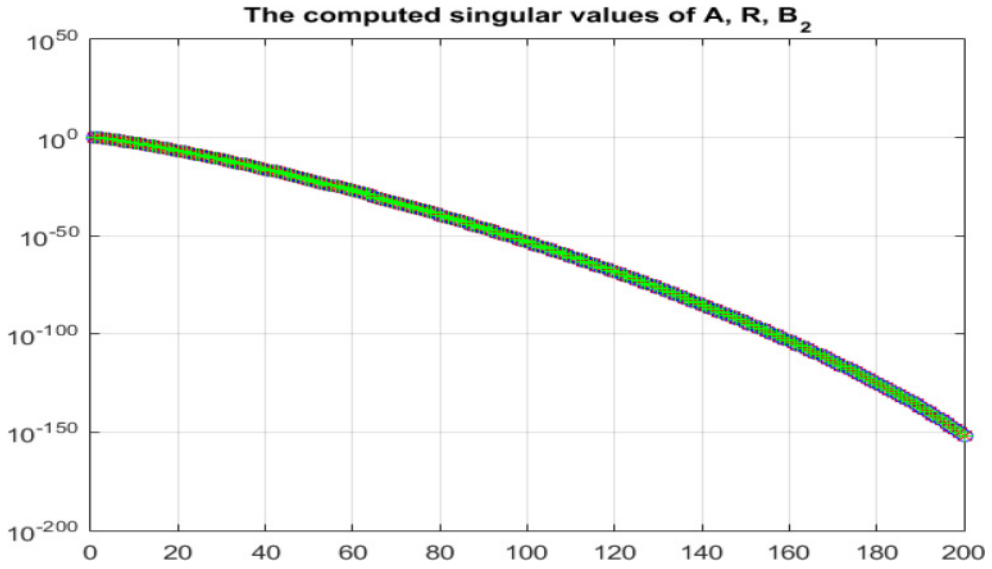

Fig. 3. (Example 2.3) The singular values of A (+), R (o), and B_2 (x), as computed by `svd` in Matlab. They all agree (including the singular values of B_1 not shown on this plot) to roughly 14 decimal digits, and all also agree with the values computed by the Jacobi SVD to nearly 14 digits.

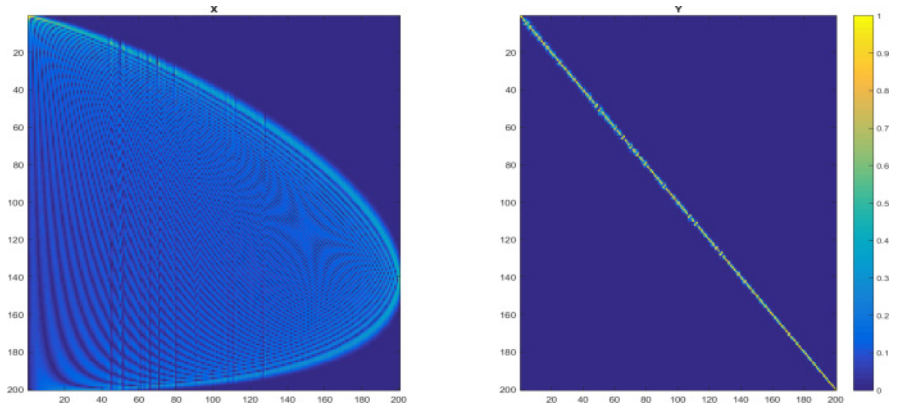

Fig. 4. (Example 2.3) The structure of the orthogonal matrices in the bidiagonalization $A = XBY^T$.

did not change and the plot would be almost identical to the one in Figure 3. The structure of the orthogonal matrices that reduce $A(\zeta, :)$ to bidiagonal form has changed in a predictable way, shown in Figure 6. Similar conclusions about the accuracy of the computed singular values and the structure of bidiagonalizing orthogonal matrices hold if, instead of a row permutation, we premultiply A with a random orthogonal matrix (Figure 7).

Next, the columns of A are permuted and the SVD of $A(:, \zeta)$ is computed; the results are shown in Figure 8 . The singular values of R and B_2 (and also the singular values of B_1 not shown on this plot) again agree to roughly 14 decimal digits, and all also agree with the values computed by the Jacobi SVD to nearly 14 digits. The singular values of $A(:, \zeta)$ that are below $\epsilon \|A\|_2$ are computed with relatively large upward bias.

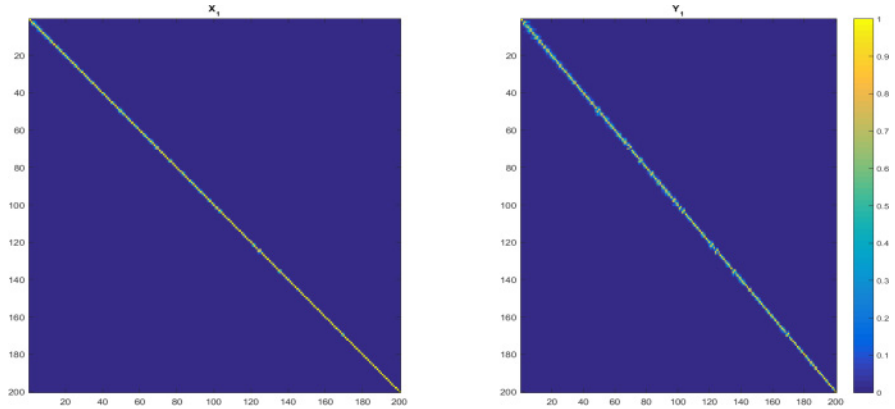

Fig. 5. (Example 2.3) The structure of the orthogonal matrices in the bidiagonalization $R = X_1 B_1 Y_1^T$. (A similar structure is present in X_2, Y_2 in the bidiagonalization $R^T = X_2 B_2 Y_2^T$, which is not shown here to save space.)

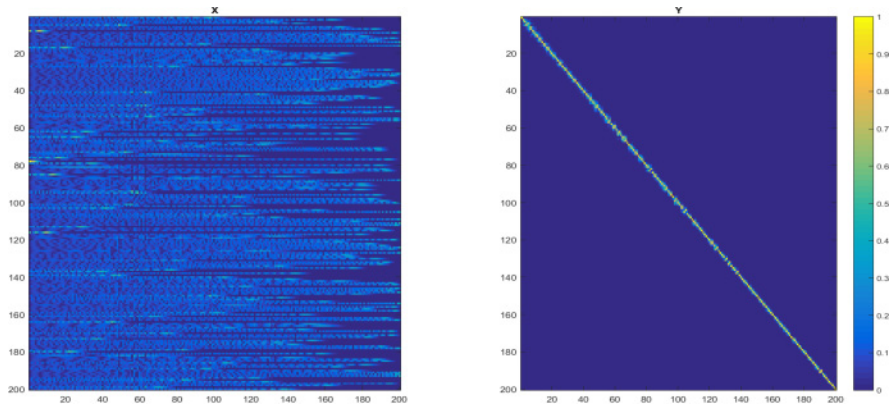

Fig. 6. (Example 2.3) The structure of the orthogonal matrices in the bidiagonalization $A(\zeta, :) = XBY^T$.

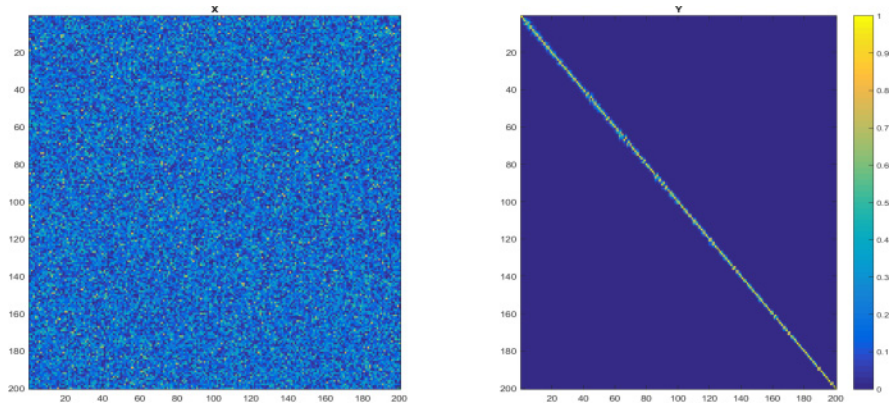

Fig. 7. (Example 2.3) The structure of the orthogonal matrices in the bidiagonalization $\Omega_{random} A = XBY^T$, where Ω_{random} is random (Haar-distributed) orthogonal matrix.

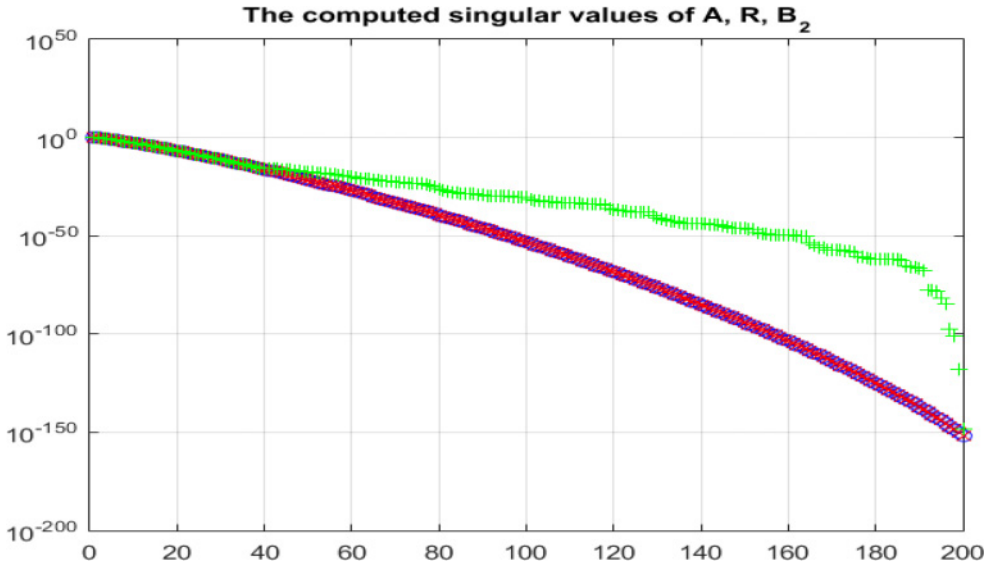

Fig. 8. (Example 2.3) The singular values of $A \equiv A(:, \zeta)$ (+), R (o), and B_2 (x), as computed by `svd` in Matlab.

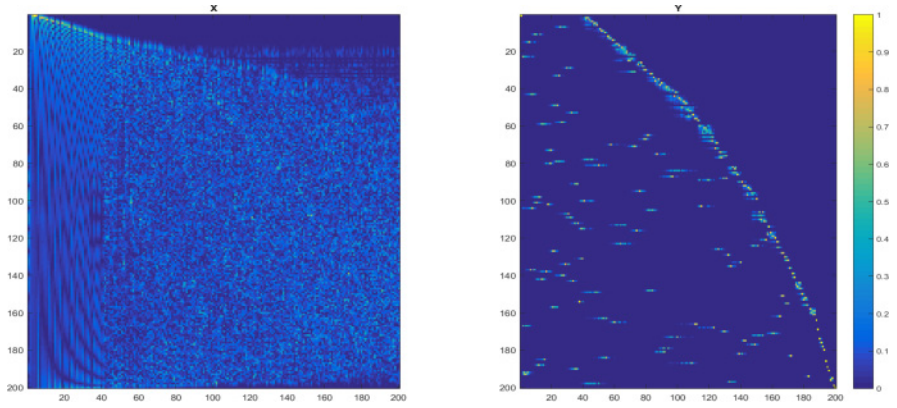

Fig. 9. (Example 2.3) The structure of the orthogonal matrices in the bidiagonalization $A(:, \zeta) = XBY^T$.

It is again instructive to inspect the structure of the bidiagonalizing matrices (Figure 9).

Remark 2.4. Finally, it is an interesting question for future research how this preprocessing with rank revealing QR factorization influences the structure of the bidiagonal factor and the convergence of the QR iterations.

2.2.3. Technical Detail: Rank Revealing QR Factorization. Line 2 of Algorithm 1 authorizes the pivoted (rank revealing) QR factorization from line 1 to determine the numerical rank of A, based on the computed triangular factor R. In all three options for pivoting, listed in Section 1.2, the computed upper triangular $R = (R_{ij})_{i,j=1}^{n}$ has the diagonal dominance structure

$$|R_{ii}|^2 \geq \sum_{k=i}^{j} R_{kj}^2, \quad 1 \leq i < j \leq n. \tag{22}$$

With careful implementation [Drmač and Bujanović 2008], the computed $\widetilde{R} = (\widetilde{R}_{ij})_{i,j=1}^{n}$ has the structure (22) up to small rounding error.

If A is close to a matrix of lower rank, and if the rank revealing pivoting performs well (which is usually the case), the computed factor can be partitioned as

$$\tilde{R} = \begin{pmatrix} \widetilde{R}_{[11]} & \widetilde{R}_{[12]} \\ 0 & \widetilde{R}_{[22]} \end{pmatrix}, \quad \text{where } \widetilde{R}_{[22]} \in \mathbb{R}^{(n-\rho)\times(n-\rho)} \text{ is } \textit{sufficiently small}. \tag{23}$$

Setting $\widetilde{R}_{[22]}$ to zero and computing the SVD of $(\widetilde{R}_{[11]} \;\; \widetilde{R}_{[12]})$ is equivalent to working with backward perturbed A as follows:

$$\left(A + \Delta A - \widetilde{Q} \begin{pmatrix} 0 & 0 \\ 0 & \widetilde{R}_{[22]} \\ 0 & 0 \end{pmatrix} \Pi_c^T \right) \Pi_c = \widetilde{Q} \begin{pmatrix} \widetilde{R}_{[11]} & \widetilde{R}_{[12]} \\ 0 & 0 \\ 0 & 0 \end{pmatrix}. \tag{24}$$

Hence, with given tolerance threshold ϵ, the quotient $\|\widetilde{R}_{[22]}\|_F / \|A\|_F$ is compared against ϵ (e.g., $\epsilon \approx m\boldsymbol{\varepsilon}$), and we have truncation with controlled backward error in the matrix norm sense. If such a level of accuracy is acceptable, it will be indicated by the corresponding job parameter on entry to `xGESVDQ`.

The index ρ can be determined also with a less aggressive strategy, such as by looking for a gap between two consecutive diagonals of $\tilde{R}$ (i.e., $|\widetilde{R}_{\rho+1,\rho+1}| \le \epsilon |\widetilde{R}_{\rho\rho}|$). In that case, $\|\widetilde{R}_{[22]}\|_2 \le \sqrt{n-\rho}\,\epsilon |\widetilde{R}_{\rho\rho}|$. For the corresponding error bonds in this case, see Drmač and Veselić [2008a].

The option of determining the numerical rank is attractive in the case of large matrices of low numerical rank ($\rho \ll n$), and if the accuracy of the form (9) is sufficient. Indeed, in that case, the bidiagonalization runs on a $\rho \times n$ matrix and the QR iterations run in the dimension $\rho \times \rho$. In fact, in such cases, `xGESVDQ` has the potential for outperforming both `xGESVD` and `xGESDD`.

Finally, if high accuracy is desired even in the tiniest singular values, the rank revealing QR factorization will not be authorized to determine the numerical rank. That task is left to the routine that computes the singular values.

2.2.4. Other Technical Details of the Implementation and Further Development. For a library routine, `xGESVDQ` must be equipped with a simple device that prevents overflow of the largest singular value—that is, if needed, A is replaced with sA, where s is a suitably determined scaling factor. This requires an extra pass through the array A. The same task is performed in `xGESVD`, and in an optimal implementation, one would integrate `xGESVD` into the framework provided by `xGESVDQ` and drop the second search for absolutely largest entries and possible scaling. It is a challenging task to modify the bidiagonal SVD part of `xGESVD` to allow computation of the singular values in the full range of the floating point numbers.

There are other details that contribute to numerical robustness and better theoretical understanding of the computation. For instance, the row pivoting[4] may greatly increase the set of matrices for which high accuracy in the SVD is feasible. In some cases, it suffices if A can be written as $A = D_1 C D_2$, where C is well conditioned and D_1 D_2 are diagonal matrices with particularly ordered diagonal entries. For more details see Cox and Higham [1998], Drmač [1999, 2000], Higham [2000], and Drmač and Veselić [2008a, 2008b].

Next, in the square case, it might make a difference to choose to implicitly diagonalize AA^* instead of A^*A. Hence, it would be useful to make an informed decision on whether

[4]Initial row sorting is already available as an option in `xGESVDQ`.

to use A or A^* as input to xGESVDQ. See Section 3 of Drmač and Veselić [2008a] and the LAPACK routine xGEJSV for more details.

3. NUMERICAL TESTING OF THE IMPLEMENTATION

Our code [Drmač 2016] is implemented in all four data types as SGESVDQ, DGESVDQ, CGESVDQ, and ZGESVDQ, and thoroughly tested.[5] In this section, for the sake of brevity, we only show the results of CGESVDQ. The goal of the test is to examine to what extent the new routine computes the SVD of A with forward error that only depends on $\kappa_{scaled}(A) = \kappa_2(C)$, where $A = CD$, as explained in Section 1.2. The working precision of CGESVDQ is $\boldsymbol{\varepsilon}$=SLAMCH('Epsilon')$\approx 5.96 \cdot 10^{-8}$. The test matrices and the reference values are computed in working precision with $\boldsymbol{\varepsilon}_{double}$=DLAMCH('Epsilon')$\approx 2.22 \cdot 10^{-16}$.

3.1. Test Matrices

The test matrices are generated in eight classes, $i = 1, \ldots, 8$; in the i-th class the scaled condition number is $\kappa_{scaled}(A) \approx 10^i$, and in each class with fixed $\kappa_{scaled}(A)$ the diagonal scaling D increases with factor 10^2, from 10^2 to 10^{16} in eight steps. To generate a matrix $A = CD$ with $\kappa_{scaled}(A) \equiv \kappa_2(C) = k_1$ and $\kappa_2(D) = k_2$, we use Algorithm 2.

ALGORITHM 2: $A =$ ZMGEN(k_1, k_2, μ_1, μ_2) $(A \in \mathbb{C}^{m \times n}, m \geq n)$

1: Generate random diagonal matrix C with $\kappa_2(C) = k_1$ and with C_{ii} distributed as determined by the mode parameter μ_1. For example, use DLATM1.
2: Set $C = \Omega_1 C \Omega_2$, where Ω_1, Ω_2 are Haar-distributed random unitary matrices (e.g., ZLAROR, which implements Stewart [1980]).
3: Use a sequence of plane rotations Ψ_j such that $C\Psi_1\Psi_2\cdots$ has equilibrated columns. Use P 8.5.3 and P 8.5.4 of Golub and Van Loan [1989].
4: Generate random diagonal matrix D with $\kappa_2(D) = k_2$ and with D_{ii} distributed as determined by the mode parameter μ_2. For example, use DLATM1.
5: Set $A = CD$.

With fixed value of the scaled condition number $k_1 \approx 10^i$, by varying k_2, μ_1, μ_2, 1,352 matrices are generated in DOUBLE COMPLEX precision, giving the total of 10,816 test cases. We fix the dimensions to $m = 1{,}000$, $n = 700$.

3.2. Reference Values

To test CGESVDQ, we first compute reference values that are sufficiently accurate for the purpose of the test. To that end, we can use the singular values computed by ZGESVJ or ZGEJSV because both methods have provably accurate results. As an illustration of the compliance with theory, we compare the singular values computed by the two subroutines. The (measured) relative differences between the singular values computed by ZGEJSV and ZGESVJ, shown in Figure 10, are approximately bounded by $\max(m, \kappa_{scaled}(A))\boldsymbol{\varepsilon}$.

The staircase shape of the measured differences in Figure 10 is due to the ordering of the test matrices: they are grouped with increasing $\kappa_{scaled}(A)$ from 10 to 10^8. For test indices at the i-th stair, perturbation theory predicts uncertainty in the singular values of the order of $10^i\boldsymbol{\varepsilon} \approx 10^{i-16}$, and this is just about how much the results of the two routines differ. Since in our test examples the scaled condition is at most of the order of 10^8, we have about eight correct digits for reference. Before the matrices are given to CGESVDQ, they are first rounded to COMPLEX precision, thus with initial

[5] Beware of bugs in the preceding code; I have only proved it correct, not tried it [Knuth 1977].

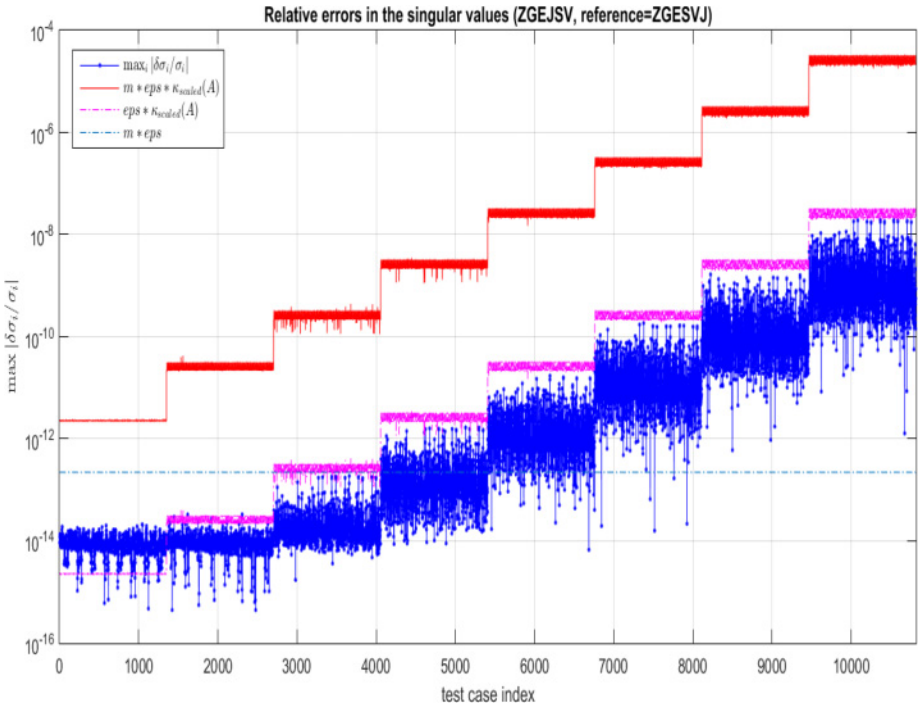

Fig. 10. Maximal relative differences between the singular values computed by ZGESVJ and ZGEJSV over 10,816 test matrices of dimension $1{,}000 \times 700$, generated as explained in Section 3.1. Note the dependence on $\boldsymbol{\varepsilon}\kappa_{scaled}(\cdot)$, in agreement with error analysis and perturbation theory.

entrywise uncertainty of order 10^{-8} and with exact singular values that coincide with the original DOUBLE COMPLEX matrix to roughly $8-i$ digits for $k_1 \approx 10^i$, $i = 1, \ldots, 8$.

3.3. Test Results

The following tests were performed on the generated 10,816 examples:

(1) *The residual* $\|A - \widetilde{U}\widetilde{\Sigma}\widetilde{V}^*\|_F/\|A\|_F$: It should be bounded by $f(m,n)\boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon}$ is the roundoff unit of the working precision and $f(m,n)$ is moderate polynomial in m, n. Such a bound follows from the backward error analysis, and it usually considerably overestimates the actual residual (i.e., the backward error). In practice, we expect the residual to be bounded by $O(m)\boldsymbol{\varepsilon}$.

(2) *Orthogonality of the singular vectors*: $\max_{i\neq j} |\widetilde{u}_i^*\widetilde{u}_j|$ (should be below $cm\boldsymbol{\varepsilon}$) and $\max_{i\neq j} |\widetilde{v}_i^*\widetilde{v}_j|$ (it should be below $cn\boldsymbol{\varepsilon}$). Here c is a small constant, $1 \leq c \leq 10$ (say).

(3) *The relative errors in the computed singular values*: If $\widetilde{\sigma}_1 \geq \widetilde{\sigma}_2 \geq \cdots \geq \widetilde{\sigma}_n$ are the computed and $\widehat{\sigma}_1 \geq \widehat{\sigma}_2 \geq \cdots \geq \widehat{\sigma}_n$ the reference values. From the perturbation theory and error analysis of the method, it follows that the maximal relative error in the computed singular values should be bounded by $g(m,n)\boldsymbol{\varepsilon}\kappa_{scaled}(A)$, where $g(m,n)$ is moderate. Hence, for any full column rank A, we have

$$\eta \equiv \max_{i=1:n} \frac{|\widetilde{\sigma}_i - \widehat{\sigma}_i|}{\widehat{\sigma}_i} \leq g(m,n)\boldsymbol{\varepsilon}\kappa_{scaled}(A). \tag{25}$$

To estimate the actual behavior of the right-hand side in (25), we compare the measured error η with $\boldsymbol{\varepsilon}\kappa_{scaled}(A)$ and $m\boldsymbol{\varepsilon}\kappa_{scaled}(A)$.

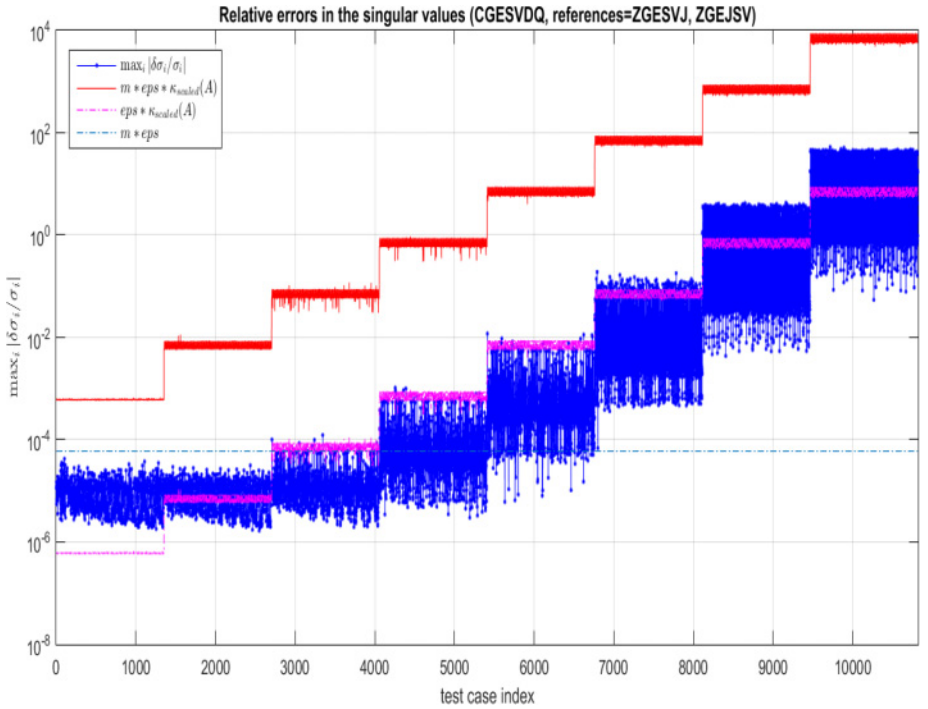

Fig. 11. Accuracy in the singular values computed by CGESVDQ. The reference values are justified by the perturbation theory and double checked as shown in Figure 10.

(4) *The errors in the computed singular vectors*: If the computed decomposition satisfies $A + \Delta A \equiv (I + \Delta A A^{\dagger})A = \widetilde{U}\widetilde{\Sigma}\widetilde{V}^{*}$, then Theorem 3.3. of Eisenstat and Ipsen [1995] implies that the difference between the i-th computed right singular vector $\widetilde{v}_i$ and its reference value $\widehat{v}_i$ (computed by ZGESVJ) can be estimated as

$$\|\widetilde{v}_i - \widehat{v}_i(\widehat{v}_i^{*}\widetilde{v}_i)\|_2 \leq \frac{O(\|\Delta A A^{\dagger}\|_2)}{gap_i}, \quad gap_i = \min\left\{2, \min_{j\neq i}\frac{|\widehat{\sigma}_i - \widehat{\sigma}_j|}{\widehat{\sigma}_i}\right\}. \tag{26}$$

An analogous statement holds true for the left singular vectors. In the case of backward stability discussed in Section 1.2, it holds that $O(\|\Delta A A^{\dagger}\|_2) \leq g(m,n)\boldsymbol{\varepsilon}\kappa_{scaled}(A)$. Hence, in the case that the computed vectors obey these bounds, the measured values of $\|\widetilde{v}_i - \widehat{v}_i(\widehat{v}_i^{*}\widetilde{v}_i)\|_2 \cdot gap_i$ should be bounded by $g(m,n)\boldsymbol{\varepsilon}\kappa_{scaled}(A)$, i.e. by $O(\boldsymbol{\varepsilon})\kappa_{scaled}(A)$. See Section 5.2 of Demmel [1997] for more details.

The first two tests are routine checking of an SVD software implementation, and the code under examination excelled in both of them. The new challenge posed to the code are the last two tests, and xGESVDQ proved to be up to the task.

In Figure 11, we show the relative differences between the singular values computed by CGESVDQ and the reference values from ZGESVJ. It is striking how CGESVDQ computes the singular values to nearly the same level of accuracy as CGEJSV, and how the maximal measured relative errors behave as $\boldsymbol{\varepsilon}\kappa_{scaled}(A)$. In fact, one could argue (with a proof in the case of CGEJSV) that most of the error is caused in the first step, the QR factorization. Further, the accuracy of the right singular vectors is similar to the Jacobi SVD. In Figure 12, we display the measured error in the right singular vectors multiplied by the relative gap: note how the errors follow and stay below $\boldsymbol{\varepsilon}\kappa_{scaled}(A)$. The analogous results for the left singular vectors are omitted for the sake of brevity.

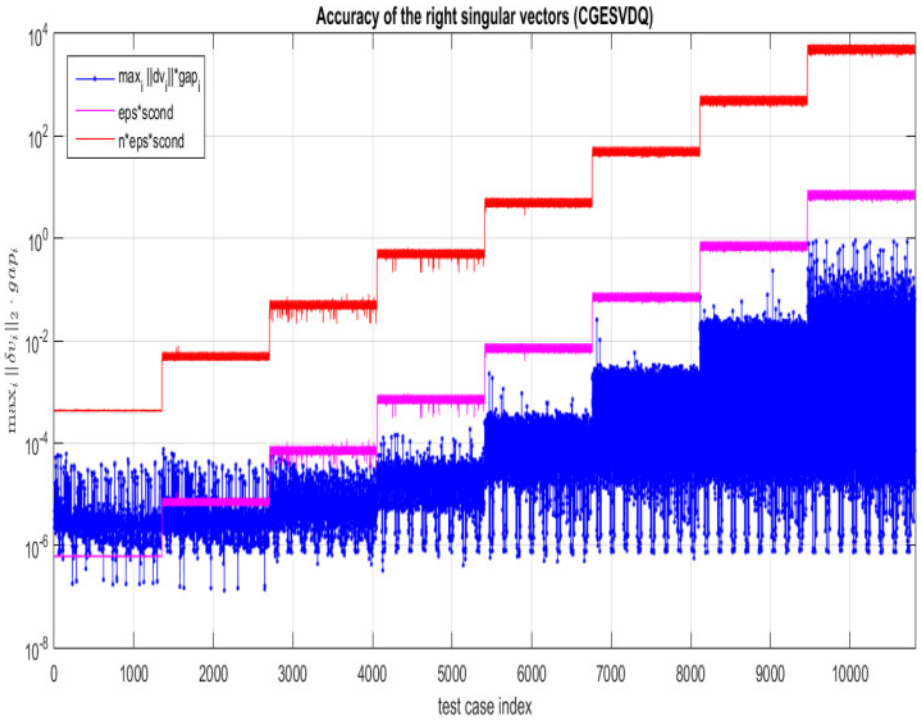

Fig. 12. Accuracy in the right singular vectors computed by CGESVDQ as compared to the reference values from ZGESVJ. The measured values $\|\widetilde{v}_i - \widehat{v}_i(\widehat{v}_i^*\widetilde{v}_i)\|_2 \cdot gap_i$ are, up to a factor, bounded by $\boldsymbol{\varepsilon}\kappa_{scaled}(\cdot)$ as in (26).

The gain of the new routine CGESVDQ over the original CGESVD is illustrated in Figure 13. It is obvious that high $\kappa_2(A)$, $A = CD$, induced by diagonal scaling D, causes large relative errors in the smallest singular values computed by CGESVD.

Example 3.1. In Drmač [2015], we presented a Jacobi SVD–based algorithm for accurate computation of Factored Hankel matrices. In one numerical experiment in Matlab, we replaced the Jacobi SVD with the combination of the QR factorization with column pivoting and the function `svd()`. The method was successful in computing the singular values to nearly machine precision despite the condition number $\sigma_{\max}/\sigma_{\min}$ bigger than 10^{200}. The result of that experiment triggered our development of xGESVDQ.

3.3.1. Runtime Efficiency. Researchers focused on software optimization and runtime efficiency will immediately complain that the initial pivoted QR factorization kills the performance, and unfortunately they will be right. A trade-off between accuracy and speed seems unavoidable. The QR with pivoting is less efficient than bidiagonalization just because of pivoting. However, the overhead is clearly seen (the time for xGEQP3), and it can be estimated a priori relative to the time of xGESVD, up to possible difference in the number of QR iterations.

It should be pointed out that the initial QR factorization not only improves the numerical accuracy of the method but also provides a reliable error bound with a negligible overhead. Namely, once we have computed R, it takes only $O(n^2)$ flops to scale its columns and estimate $\kappa_{scaled}(A) = \kappa_{scaled}(R)$ using LAPACK's xPOCON.

For the matrices used for accuracy tests, whose results are shown in Figures 11 and 13, the corresponding runtimes are shown on Figure 14. On average, CGESVDQ was about 25% slower than CGESVD. It is then up to the user to decide whether that extra

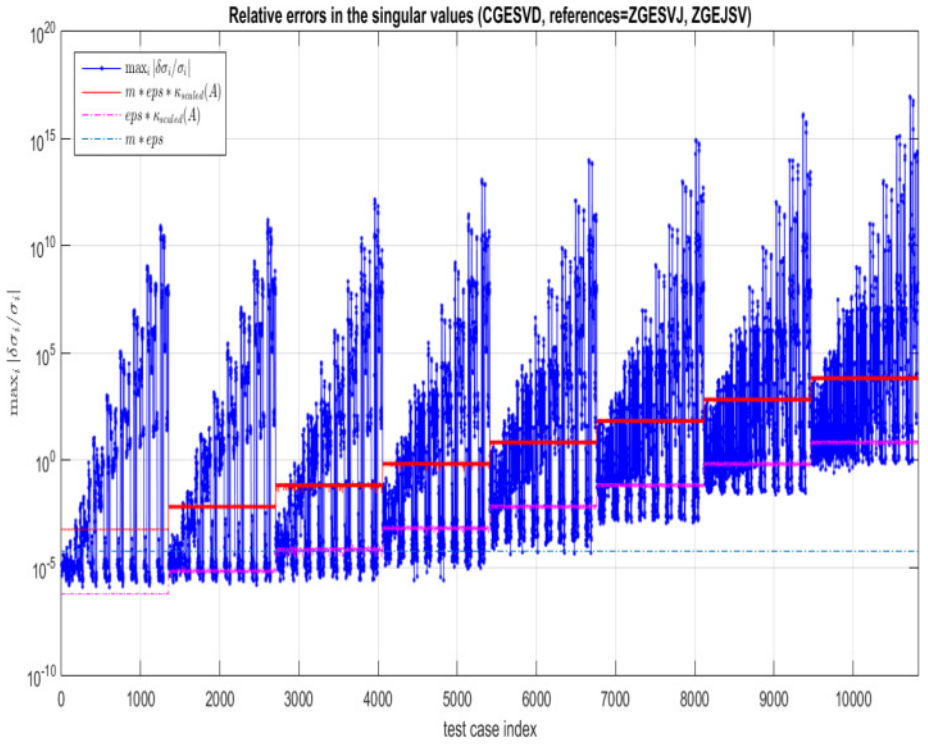

Fig. 13. Accuracy in the singular values computed by CGESVD. Large relative errors indicate that in the case of large $\kappa_2(A)$, the smallest singular values are computed entirely wrong, even if $\kappa_{scaled}(A)$ is moderate.

time is worth having the potentially better results with a realistic error estimate, as shown in Figure 11 (in contrast to those in Figure 13) and Figure 12, and as illustrated in Examples 2.3 and 3.1.

For best performance in the case when the standard accuracy analogous to the one delivered by xGESDD suffices, one would use a modified version of xGEQP3 with two additional input parameters that will specify the cutoff method and the corresponding threshold and break the factorization leaving the truncated part not factored at all. More precisely, after elimination of k columns, we have[6] (see Section 2.2.3):

$$(A + \Delta_k A)\Pi_k = \widetilde{Q}_k \begin{pmatrix} \widetilde{R}_{[11]} & \widetilde{R}_{[12]} \\ 0 & \widetilde{X}_{[22]} \end{pmatrix}, \quad \widetilde{R}_{[11]} \in \mathbb{C}^{k\times k}, \quad \widetilde{X}_{[22]} \in \mathbb{C}^{(m-k)\times(n-k)},$$

where $\widetilde{X}_{[22]}$ needs to be factored in the ensuing $n-k$ steps. However, in the course of pivoting, the largest column of $\widetilde{X}_{[22]}$ will be determined and its norm will become the modulus of $\widetilde{R}_{k+1,k+1}$. This means that the truncation strategies discussed in Section 2.2.3 can be immediately deployed, and if a cutoff test is positive, the factorization can stop with the backward error analogous to (24), where $\widetilde{X}_{[22]}$, left unfactored, replaces $\begin{pmatrix} \widetilde{R}_{[22]} \\ 0 \end{pmatrix}$.

3.3.2. Other Choices of Pivoting. Possible modifications toward improved implementation will benefit from faster rank revealing QR factorization algorithms, such as in Tomas et al. [2012], or from using, for example, localized pivoting, such as xGEQPX,

[6]Here we use subscript k to denote intermediate values after the k-th step.

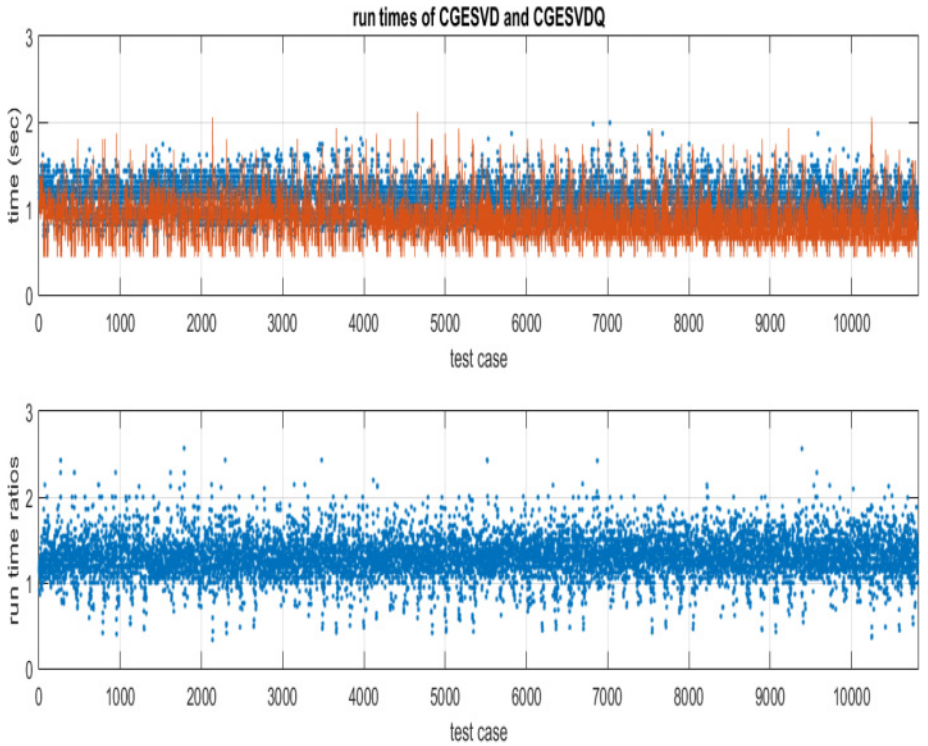

Fig. 14. Runtime ratios *time*(CGESVDQ)/*time*(CGESVD) for computing the SVD with n left singular vectors, the singular values, and the n right singular vectors. The computation is done using an Intel MKL 5.4.1 on an Intel Core i7-4810MQ CPU @ 2.80GHz and with 16GB RAM.

xGEQPY from Bischof and Quintana-Orti [1998a, 1998b]. A new great potential to reduce the overhead due to pivoting is the recent development of the communication avoiding tournament pivoting [Demmel et al. 2015] and a randomized sampling approach [Duersch and Gu 2015] that nearly reaches the efficiency of the QR factorization without pivoting (xGEQRF). In the proposed software xGESVDQ, we opted for xGEQP3 to provide a (LAPACK-based) ready-to-deploy enhancement of xGESVD that will automatically benefit from high-performance implementations of LAPACK in vendor-optimized libraries such as MKL [Intel 2015]. Further, with the modular structure of the xGESVDQ code, we provide a platform for further development: xGEQP3 can be replaced with a more efficient code, once it becomes available. However, numerical analysis that would explain and prove the exceptional accuracy remains a challenging problem; for a formal proof, the structure of the triangular factor returned by xGEQP3 is a good starting point [Barlow 2002].

3.3.3. Using Static Pivoting: Initial (Row and) Column Sorting. A simple replacement for dynamic column pivoting could even be just initial column permutation to sort the columns in nonincreasing sequence with respect to their lengths, designated as xGESVDS. In fact, already such a simple strategy greatly improves the accuracy of xGESVD, and it can be used in practice, optionally enhanced with a fast localized pivoting. However, a further study of the numerical potential of such relaxed strategies is necessary. We have repeated the previous test with the code xGESVDQS that has optional row sorting, and instead of dynamic pivoting, the columns are presorted in decreasing sequence with respect to their Euclidean norms. The measured relative errors in

the computed singular values behave similarly as shown in Figure 11. The runtime overhead is essentially the time of xGEQRF.

4. APPLICATION TO HERMITIAN/SYMMETRIC DEFINITE EIGENVALUE PROBLEM

When spectral decomposition of a real symmetric[7] positive definite matrix H is needed to high relative accuracy with special attention to the tiniest eigenvalues, then the best strategy is not to form the matrix H at all. Very often, if it is known a priori that H is positive definite, then we most likely can directly assemble a matrix A such that $H = A^TA$. Any spectral information on H can be obtained from the SVD of A, and since $\kappa_2(A) = \sqrt{\kappa_2(H)}$, the numerical advantage of this implicit diagonalization of H is obvious. In some case, working with A instead of H is the key step in the computation.

Similarly as in the case of the SVD, there is difference in accuracy between the tridiagonalization-based methods such as the QR and the divide and conquer algorithms and the Jacobi algorithm [Demmel and Veselić 1992]. In Drmač and Veselić [2000], the eigenvectors of faster but less accurate tridiagonalization-based methods are used as a preconditioner for the Jacobi algorithm, with an idea to achieve both runtime efficiency and numerical accuracy. In this section, we show that a QR-based method can be competitive in accuracy with the Jacobi algorithm.

4.1. Preliminaries: Cholesky Factorization and the Implicit Jacobi SYPDEV Algorithm

If a (presumably) positive definite real symmetric H is already stored in the machine memory and if its spectral decomposition is required to high accuracy, then the best initial step in a numerical algorithm is the Cholesky factorization $H = LL^T$, which is a kind of "litmus test" for positive definiteness. A justification for this follows from error analysis and the corresponding perturbation theory. Therefore, for the reader's convenience, in this section we briefly review the elements of perturbation theory relevant to floating point computation of eigenvalues of positive definite matrices [Barlow and Demmel 1990; Demmel 1989; Demmel and Veselić 1992; Demmel et al. 1999].

For positive definite H, the key for accurate spectral decomposition is the diagonally scaled matrix $H_s = D^{-1}HD^{-1}$, where $D = \mathrm{diag}(\sqrt{H_{11}}, \ldots, \sqrt{H_{nn}})$. Since $\kappa_2(H_s) \leq n \min_{D=\mathrm{diag}} \kappa_2(DHD) \leq n\kappa_2(H)$, and since it is possible that $\kappa_2(H_s) \ll \kappa_2(H)$, perturbation estimates with $\kappa_2(H_s)$ instead of $\kappa_2(H)$ are expected to give stronger results.

THEOREM 4.1 [DEMMEL 1989]. *Let H be an $n \times n$ symmetric matrix with positive diagonal entries, stored in the machine memory. Let H be input matrix in the Cholesky factorization algorithm. Then the following holds:*

(*i*) *If the Cholesky algorithm successfully completes all operations and computes lower triangular matrix $\tilde{L}$, then there exists a symmetric backward perturbation δH such that $\tilde{L}\tilde{L}^T = H + \delta H$ and*

$$|\delta H_{ij}| \leq \eta_C \sqrt{H_{ii}H_{jj}}, \quad \eta_C = \frac{c(n)\boldsymbol{\varepsilon}}{1 - 2c(n)\boldsymbol{\varepsilon}}, \quad c(n) = \max\{3, n\}. \tag{27}$$

(*ii*) *If $\lambda_{\min}(H_s) > n\eta_C$, then the Cholesky algorithm will succeed and compute $\tilde{L}$.*
(*iii*) *If $\lambda_{\min}(H_s) < \boldsymbol{\varepsilon}$, then there exist simulation of rounding errors in which the Cholesky algorithm fails to complete all operations.*
(*iv*) *If $\lambda_{\min}(H_s) \leq -n\eta_C$, then it is certain that the Cholesky algorithm will fail.*[8]

[7]For simplicity, we only consider the real symmetric case; the Hermitian case is treated analogously.
[8]It is possible that our matrix H is theoretically positive definite and that errors in computing its entries as functions of some parameters cause the stored matrix (in the computer memory) to be indefinite. Additionally,

From Theorem 4.1, it follows that in the case of no additional assumptions on the structure of H, numerical computation of all eigenvalues of H is feasible only if $\lambda_{\min}(H_s) > \boldsymbol{\varepsilon}$ (i.e., if $\|H_s^{-1}\|_2 < 1/\boldsymbol{\varepsilon}$). Further, the particular structure of the backward error yields the following sharp forward error bound for the eigenvalues of $\widetilde{L}\widetilde{L}^T$.

THEOREM 4.2. *Let $\lambda_1 \geq \cdots \geq \lambda_n$ and $\widetilde{\lambda}_1 \geq \cdots \geq \widetilde{\lambda}_n$ be the eigenvalues of H and of $\widetilde{H} = H + \delta H = \widetilde{L}\widetilde{L}^T$, respectively. If $\|L^{-1}\delta H L^{-T}\|_2 < 1$, then*

$$\max_i \left| \frac{\widetilde{\lambda}_i - \lambda_i}{\lambda_i} \right| \leq \|H_s^{-1}\|_2 \left\| \left[\frac{\delta H_{ij}}{\sqrt{H_{ii}H_{jj}}} \right]_{i,j=1}^{n} \right\|_2 . \tag{28}$$

(Recall the classical Weyl's theorem: $\max_i |\frac{\widetilde{\lambda}_i - \lambda_i}{\lambda_i}| \leq \kappa_2(H)\frac{\|\delta H\|_2}{\|H\|_2}$.*)*

PROOF. Let $Y = \sqrt{I + L^{-1}\delta H L^{-T}}$. Then $H + \delta H = L(I + L^{-1}\delta H L^{-T})L^T = LYY^TL^T$ is similar with Y^TL^TLY, and we can equivalently compare the eigenvalues $\lambda_i(L^TL) = \lambda_i(H)$ and $\lambda_i(Y^TL^TLY) = \lambda_i(H + \delta H)$.

Now recall Ostrowski's theorem: if $\widetilde{M} = Y^TMY$, then for all i, $\lambda_i(\widetilde{M}) = \lambda_i(M)\xi_i$, where $\lambda_{\min}(Y^TY) \leq \xi_i \leq \lambda_{\max}(Y^TY)$.

Since $Y^TY = I + L^{-1}\delta H L^{-T}$, we have $|\lambda_i(H) - \lambda_i(\widetilde{H})| \leq \lambda_i(H)\|L^{-1}\delta H L^{-T}\|_2$, where

$$\begin{aligned} \|L^{-1}\delta H L^{-T}\|_2 &= \|L^{-1}D(D^{-1}\delta H D^{-1})DL^{-T}\|_2 = \|L^{-1}D(\delta H_s)DL^{-T}\|_2 \\ &\leq \|L^{-1}D\|_2^2\|\delta H_s\|_2 = \|DL^{-T}L^{-1}D\|_2\|\delta H_s\|_2 \\ &= \|(D^{-1}HD^{-1})^{-1}\|_2\|\delta H_s\|_2 = \|H_s^{-1}\|_2\|\delta H_s\|_2. \end{aligned}$$

The claim (28) follows since $\delta H_s = (\delta H_{ij}/\sqrt{H_{ii}H_{jj}})$. □

If $L = U\Sigma V^T$ is the SVD of L, then $H = LL^T = U\Sigma^2U^T$ is the spectral decomposition of H. Hence, once we have computed the Cholesky factor, the problem is reduced to computing its SVD. Since $\kappa_2(L) = \sqrt{\kappa_2(H)}$, even the methods for computing the SVD with the accuracy described in Section 1.1 become competitive for accurate computation of the eigenvalues of H. The Jacobi SVD method fits best because it will compute the SVD with the condition number $\sqrt{\kappa_2(H_s)}$ and with the backward error of the same type and size as in Theorem 4.1. Such a scheme was introduced by Veselić and Hari [1989].

ALGORITHM 3: $(\lambda, U) = \texttt{xSYPDEVJ}(H)$

$P^THP = LL^T$ {Cholesky factorization with pivoting}
if L computed successfully **then**
  $X = L$, $X_\infty = X\langle V_x\rangle$ {One-sided Jacobi SVD on X}
  $\lambda_i = X_\infty(:,i)^TX_\infty(:,i), \quad i = 1, \ldots, n$; $\lambda = (\lambda_1, \ldots, \lambda_n)$;
  $U(:,i) = \dfrac{1}{\sqrt{\lambda_i}}PX_\infty(:,i), \quad i = 1, \ldots, n$;
else
  Red flag: H is not numerically positive definite. If the Cholesky factorization succeeded to compute k columns of L, compute the SVD of $L(1:n, 1:k)$ as above earlier.
end if

it is possible that, due to rounding errors, the algorithm succeeds in computing a triangular factor of such an indefinite matrix.

PROPOSITION 4.3. *Let* $\widetilde{L}$, $\widetilde{X}_\infty$, $\widetilde{U}$, $\widetilde{\lambda} = (\widetilde{\lambda}_1, \ldots, \widetilde{\lambda}_n)$ *be the computed approximations of* L, X_∞, U, $\lambda = (\lambda_1, \ldots, \lambda_n)$, *respectively. Let* $\widetilde{\Lambda} = \mathrm{Diag}(\widetilde{\lambda})$. *Then* $\widetilde{U}\widetilde{\Lambda}\widetilde{U}^T = H + \Delta H$ *with*

$$\max_{i,j} \frac{|\Delta H_{ij}|}{\sqrt{H_{ii}H_{jj}}} \leq \widetilde{\eta}_H \equiv \eta_C + (1+\eta_C)(2\eta_J + O(\boldsymbol{\varepsilon}) + O(\boldsymbol{\varepsilon}^2)).$$

PROOF. We know that $\widetilde{L}\widetilde{L}^T = H + \delta H \equiv \tilde{H}$ with $|\delta H_{ij}| \leq \eta_C \sqrt{H_{ii}H_{jj}}$ for all i, j. Further, we can write $\widetilde{X}_\infty = (\widetilde{L} + \delta\widetilde{L})\widehat{V}$, where $\widehat{V}$ is orthogonal and $\|\delta\widetilde{L}(i,:)\| \leq \eta_J \|\widetilde{L}(i,:)\|$ for all i. Let $\widetilde{\Sigma} = \mathrm{diag}(\sqrt{\widetilde{\lambda}_1}, \ldots, \sqrt{\widetilde{\lambda}_n})$. Simple calculation shows that we can write $\widetilde{U}\widetilde{\Sigma} = \widetilde{X}_\infty + \delta\widetilde{X}_\infty$, where $|\delta\widetilde{X}_\infty| \leq \epsilon_\lambda |\widetilde{X}_\infty|$, $0 \leq \epsilon_\lambda \leq 3\boldsymbol{\varepsilon}$. Now it holds that $\widetilde{U}\widetilde{\Sigma}^2\widetilde{U}^T = H + \delta H + E$, where for all $i, j |E_{ij}| \leq 2((\eta_J + \epsilon_\lambda(1+\eta_J)) + (\eta_J + \epsilon_\lambda(1+\eta_J))^2)\sqrt{\widetilde{H}_{ii}\widetilde{H}_{jj}}$. □

4.2. New Subroutines `xPHEVC`, `XPSEVC`

Based on the discussion in Section 4.1 and the demonstrated accuracy of the SVD code `xGESVDQ`, we have developed the eigensolvers `xPHEVC`, `XPSEVC`, based on the scheme[9] outlined in Algorithm 4.

ALGORITHM 4: $(\lambda, U) =$ `xPSEVC`(H) $(H = H^T \in \mathbb{R}^{n\times n}$ positive definite)

$P^T H P = LL^T$ {Cholesky factorization with pivoting, such as using `xPSTRF`}
if L computed successfully **then**
  `CALL xGESVD[Q]`{Compute the singular values Σ and the left singular vectors that overwrite L}
  $\lambda_i = \sigma_i^2, \quad i = 1, \ldots, n$; $\lambda = (\lambda_1, \ldots, \lambda_n)$; $U \equiv PL$;
else
  Red flag: H is not numerically positive definite. If the Cholesky factorization succeeded to compute k columns of L, `CALL xGESVD[Q]` to compute the SVD of $L(1:n, 1:k)$ as earlier.
end if

The numerical experiments presented in Example 2.3 should be convincing enough to indicate that this approach has great potential. It is astonishing that the bidiagonalization-based QR algorithm, when applied to a carefully computed Cholesky factor, is capable of computing the eigenvalues of a 200×200 Hilbert matrix with a condition number greater than 10^{300} to nearly 14 digits of accuracy working only in the standard IEEE double precision with the roundoff $\boldsymbol{\varepsilon} \approx 2.2 \cdot 10^{-16}$. Of course, the structure of the pivoted Cholesky factor is precisely the same as in the Businger-Golub pivoted triangular R factor, so this computation is implicitly `xGESVDQ`.

Here, too, one is tempted to reduce the pivoting overhead and replace it with initial sorting of the diagonal entries of H into a decreasing sequence. Such an idea is not new. For instance, Veselić [1996] pointed out that the eigenvalues of real symmetric matrices can be computed by the QR algorithm if the initial matrix H is transformed by similarity $H_1 = P^T H P$, where P is a suitable permutation matrix. Unfortunately, determining P seems to be difficult task, and it may involve searching in the permutation group. As reported in Veselić [1996], in experiments with small dimension matrices, there was always a "magic" permutation that improved the accuracy of QR, but there was no evidence that it is a sorting permutation.

[9]We proposed a similar scheme in Algorithm 4.1 of Bosner and Drmač [2005], where the SVD is based on the one-sided bidiagonalization.

However, here we do not tridiagonalize $P^T HP$, but we compute the Cholesky factor and proceed with bidiagonalization, which yields better results. In addition, as discussed previously, even for the SVD computation of a general rectangular A, the accuracy of xGESVD may greatly improve by mere sorting of columns and, optionally, the rows in decreasing sequences in norm.

Finally, let us point out that xGESVDQ can be applied for accurate numerical spectral decomposition of general Hermitian (indefinite) matrices as the core SVD routine in the method [Dopico et al. 2003].

5. APPENDIX: A COMPARISON OF JACOBI SVD AND GIVENS BIDIAGONALIZING ROTATIONS

In the framework of the error analysis presented in Section 1.1, Jacobi rotations, Givens rotations, and Householder reflectors all share the same backward stability property for any choice of the corresponding defining parameters. This is simply due to the fact that those are orthogonal (or unitary) transformations. However, different SVD methods, based on those transformations, do not share same accuracy properties. For the reader's convenience, we discuss this important issue and, using simple case study examples, point out a few subtle details.

5.1. Plane Rotation in Floating Point Arithmetic

Let $\widetilde{s}, \widetilde{c}$ be the computed values that define a plane rotation, and consider its application to two vectors x and y in computer arithmetic. Assume that $\|x\|_2 \geq \|y\|_2$. It holds that[10]

$$(x' \;\; y') \;=\; (x \;\; y) * \begin{pmatrix} \widetilde{c} & \widetilde{s} \\ -\widetilde{s} & \widetilde{c} \end{pmatrix} = (x \;\; y)\begin{pmatrix} \widetilde{c} & \widetilde{s} \\ -\widetilde{s} & \widetilde{c} \end{pmatrix} + E, \quad |E| \leq \epsilon(|x| \;\; |y|)\begin{pmatrix} |\widetilde{c}| & |\widetilde{s}| \\ |\widetilde{s}| & |\widetilde{c}| \end{pmatrix}, \tag{29}$$

$$= ((x \;\; y) + (\delta x \;\; \delta y))\begin{pmatrix} \widetilde{c} & \widetilde{s} \\ -\widetilde{s} & \widetilde{c} \end{pmatrix}, \quad \text{where } (\delta x \;\; \delta y) = E\begin{pmatrix} \widetilde{c} & \widetilde{s} \\ -\widetilde{s} & \widetilde{c} \end{pmatrix}^{-1}, \quad \text{and} \tag{30}$$

$$|(\delta x \;\; \delta y)| \;\leq\; \frac{\epsilon}{\widetilde{c}^2 + \widetilde{s}^2}(|x| \;\; |y|)\begin{pmatrix} \widetilde{c}^2 + \widetilde{s}^2 & 2|\widetilde{s}\widetilde{c}| \\ 2|\widetilde{s}\widetilde{c}| & \widetilde{c}^2 + \widetilde{s}^2 \end{pmatrix}, \quad \widetilde{c}^2 + \widetilde{s}^2 = 1 + O(\boldsymbol{\varepsilon}). \tag{31}$$

Note that $2|\widetilde{s}\widetilde{c}| \leq 1 + O(\boldsymbol{\varepsilon})$, $0 \leq \epsilon \leq O(\boldsymbol{\varepsilon})$, and that

$$\|(\delta x \;\; \delta y)\|_F \leq 2\epsilon \|(x \;\; y)\|_F (1 + O(\boldsymbol{\varepsilon})). \tag{32}$$

Note that (32) also holds for any subset of rows of $(\delta x \;\; \delta y)$. In particular, for any row index i, we have $\sqrt{|\delta x_i|^2 + |\delta y_i|^2} \leq 2\epsilon\sqrt{|x_i|^2 + |y_i|^2}(1 + O(\boldsymbol{\varepsilon}))$.

Further, the backward error in each individual vector can be estimated as follows:

$$\|\delta x\|_2 \leq \epsilon\left(\|x\|_2 + \frac{2|\widetilde{s}\widetilde{c}|}{\widetilde{c}^2 + \widetilde{s}^2}\|y\|_2\right), \quad \|\delta y\|_2 \leq \epsilon\left(\|y\|_2 + \frac{2|\widetilde{s}\widetilde{c}|}{\widetilde{c}^2 + \widetilde{s}^2}\|x\|_2\right). \tag{33}$$

Clearly, if $\|x\|_2 \gg \|y\|_2$ and if $|\widetilde{s}\widetilde{c}|$ is not correspondingly small, the backward error in y can be large—so large that it can completely wipe out the information contained in y.

Remark 5.1. Relation (32) is the main ingredient for the backward error estimate (13) as a general conclusion: multiplication of an $m \times n$ A from the right (left) by a numerically orthogonal or unitary matrix U can induce only small backward error in each individual row (column) of a matrix. This can be shown as follows. From $A * U = AU + E$, $|E| \leq O(n)\boldsymbol{\varepsilon}|A||U|$, for any row index i, $(A * U)(i,:) = (A(i,:) + E(i,:)U^{-1})U$, where it holds that $\|E(i,:)U^{-1}\|_2 \leq O(n)\boldsymbol{\varepsilon}\|A(i,:)\|_2\||U|\|_2\|U^{-1}\|_2$. Since U is numerically orthogonal, the factor $\||U|\|_2\|U^{-1}\|_2$ can be bounded by $\sqrt{n}(1 + O(n)\boldsymbol{\varepsilon})$.

[10]We use an asterisk (*) to denote standard matrix multiplication in finite precision arithmetic. Absolute values and inequalities involving vectors and matrices are understood entrywise.

5.2. One-Sided Jacobi Rotation

Now consider simple one-sided Jacobi rotation applied to the columns of an $m \times n$ real matrix A. We do not assume that A has been preconditioned by the QR factorization.

Let $(x\ \ y)$ be the pair of pivotal columns in the one-sided Jacobi SVD algorithm applied to $A = (\cdots\ x\ \cdots\ y\ \cdots)$, and let $\|x\|_2 \gg \|y\|_2$. The Jacobi rotation $\mathcal{J}_{xy}$ with the angle ϕ is the eigenvector matrix of $\left(\begin{smallmatrix} h_{xx} & h_{xy} \\ h_{yx} & h_{yy} \end{smallmatrix}\right) = (x\ \ y)^T(x\ \ y)$. In the nontrivial case $h_{xy} \neq 0$, an easy calculation gives the formulas

$$\mathcal{J}_{xy} = \frac{1}{\sqrt{1+t^2}}\begin{pmatrix} 1 & t \\ -t & 1 \end{pmatrix}, \quad \text{where } t = \frac{\operatorname{sign}(\zeta)}{|\zeta| + \sqrt{1+\zeta^2}}, \quad \text{and } \zeta \equiv \cot 2\phi = \frac{h_{yy} - h_{xx}}{2h_{xy}}. \tag{34}$$

When implemented in floating point arithmetic, a few fine details are necessary for robust numerical performance [Drmač 1997]. Here we want to point out a particular structure of the transformation in the case $\|x\|_2 \gg \|y\|_2$. To that end, note that in the case $\|x\| > \|y\|/\sqrt{\boldsymbol{\varepsilon}}$, we can estimate with error $O(\boldsymbol{\varepsilon})$ that

$$\cot 2\phi = \frac{\|y\|_2/\|x\|_2 - \|x\|_2/\|y\|_2}{2y^Tx/\|x\|_2/\|y\|_2} \approx \frac{-\|x\|_2/\|y\|_2}{2y^Tx/\|x\|_2/\|y\|_2}, \quad |\cot 2\phi| \geq \frac{1}{2\sqrt{\boldsymbol{\varepsilon}}},$$

and with relative error of order $\boldsymbol{\varepsilon}$ that

$$\tan\phi \approx \frac{1}{2}\frac{1}{\cot 2\phi} \approx -\frac{y^Tx}{\|x\|^2} = -\frac{y^Tx}{\|x\|_2\|y\|_2}\frac{\|y\|_2}{\|x\|_2}. \tag{35}$$

Thus, $|\tan\phi| \leq \sqrt{\boldsymbol{\varepsilon}}$, $\sin\phi \approx \tan\phi$ and $\cos\phi$ will be computed as one. Note that for $\|y\|_2 \ll \|x\|_2$, $|\tan\phi|$ will be correspondingly small. After the transformation, the pivot columns will be mutually numerically orthogonal. For the backward error, using (35) in (33) we see that even in the smaller column y, the backward error is small relative to $\|y\|_2$. In terms of the global matrix A, the backward error δA of this rotation has only two nonzero columns, $\delta A = (0\ \ldots\ 0\ \delta x\ 0\ \ldots\ 0\ \delta y\ 0\ \ldots\ 0)$.

5.3. Givens Rotation in Bidiagonalization

Next, let us analyze the Givens rotation as a bidiagonalization tool. (Similar analysis can be done for the Householder reflectors; we skip the details for the sake of brevity.) The best way to understand the problem is through instructive case study examples.

Example 5.2. Now consider the matrix

$$A = \begin{pmatrix} 1 & \mu & \mu \\ 0 & 1 & \mu \\ 0 & 1 & -\mu \end{pmatrix}, \quad \mu \text{ small real parameter (e.g., } |\mu| < \boldsymbol{\varepsilon}). \tag{36}$$

It can be written as $A = CD$ with diagonal D and $\kappa_2(C) < 2$. Hence, small relative perturbation in the columns of A can induce only equally small relative errors in its singular values. In particular, even the smallest singular value of A (which is below $\sqrt{3}|\mu|$) is perfectly well determined by μ, no matter how small $|\mu|$ is.

In a bidiagonalization scheme for A, the next step could be to annihilate the position $(1,3)$ using the Givens rotation $\mathcal{G} = (1/\sqrt{2})\left(\begin{smallmatrix} 1 & -1 \\ 1 & 1 \end{smallmatrix}\right)$. The result is

$$A' = \begin{pmatrix} 1 & \sqrt{2}\mu & 0 \\ 0 & \frac{1+\mu}{\sqrt{2}} & \frac{-1+\mu}{\sqrt{2}} \\ 0 & \frac{1-\mu}{\sqrt{2}} & \frac{-1-\mu}{\sqrt{2}} \end{pmatrix} \approx \widetilde{A}' = \begin{pmatrix} 1 & \sqrt{2}\mu(1+\epsilon_{12}) & 0_{\downarrow} \\ 0 & \frac{1}{\sqrt{2}}(1+\epsilon_{22}) & \frac{-1}{\sqrt{2}}(1+\epsilon_{23}) \\ 0 & \frac{1}{\sqrt{2}}(1+\epsilon_{32}) & \frac{-1}{\sqrt{2}}(1+\epsilon_{33}) \end{pmatrix}, \tag{37}$$

where $|\epsilon_{ij}| \lessapprox \boldsymbol{\varepsilon}$ are small rounding errors, and where we introduce the notation $0_{\downarrow}$ to denote that the zero is the result of "set to zero" rather than computed zero. Note that the last two columns of A are numerically orthogonal (up to machine precision), whereas the last two columns of A' and of its perfectly accurately computed value $\widetilde{A}'$ are nearly collinear (up to machine precision). As a consequence, $\widetilde{A}'$ cannot be written as $\widetilde{A}' = \widetilde{C}'\widetilde{D}'$ with well-conditioned $\widetilde{C}'$ and diagonal $\widetilde{D}'$. The transformation $\mathcal{G}$, focused to annihilating the position $(1, 3)$, acts locally, but globally it does not see the big picture.

Further, in the next step, another rotation, from the left, will annihilate the position $(3, 2)$ in $\widetilde{A}'$, and at the same time, the position $(3, 3)$ will suffer a severe cancellation, resulting in a noisy result of the order of $\boldsymbol{\varepsilon}$. The smallest singular value of the order of $|\mu|$ has been irreparably lost.

Since the parameters of the rotation $\mathcal{G}$ are both $O(1)$, and since the third column is much smaller in norm than the second one, relation (33) indicates that this step is not backward stable in the stronger columnwise sense: the smaller of the two columns will be perturbed by a contribution of the larger one. However, as it can be seen from (37), this step is entrywise forward stable, but, alas, the resulting matrix has become severely ill conditioned.

For the purpose of numerical illustration, take $\mu = \boldsymbol{\varepsilon}^2$. Matlab's `svd()` function computes the smallest singular value of A as $\min(svd(A)) = 5.836378470777888e-17$. However, up to $O(\boldsymbol{\varepsilon})$ error, the true value is $\sigma_{\min}(A) \approx 6.972611193684197e-32$. Interestingly, this is the value also returned by $\min(svd(A'))$ (i.e., when A^T s given as input to `svd()`). Additionally, the same value is obtained as the inverse of the largest singular value of A^{-T}, as computed by `svd()`.

Remark 5.3. The backward error ΔA of the computation (37) is small in each row of A (see Remark 5.1), but the row-scaled condition number is large. If we write $A + \Delta A = A(I + A^{-1}\Delta A)$, and if $D_r = \mathrm{diag}(\|A(i,:)\|_2)_{i=1}^3$, the relative error in the singular values is bounded by $\|A^{-1}\Delta A\|_2 = \|(D_r^{-1}A)^{-1}(D_r^{-1}\Delta A)\|_2 \leq \|(D_r^{-1}A)^{-1}\|_2\|D_r^{-1}\Delta A\|_2$. Here, $\|D_r^{-1}\Delta A\|_2$ must be small by Remark 5.1, but $\|(D_r^{-1}A)^{-1}\|_2$ must be large, as the last two rows of A are nearly collinear.

Example 5.4. (Example 5.2, continued.) Now assume that before annihilating $A(1, 3)$ in (36), we annihilate the position $A(3, 2)$. Since $\kappa_{scaled}(A) < 2$, this transformation (multiplication by an orthogonal matrix from the left) will preserve the smallest singular values. With the appropriate orthogonal matrix G, the result is

$$G * A = \begin{pmatrix} 1.000000000000000e+00 & 4.930380657631324e-32 & 4.930380657631324e-32 \\ 0 & 1.414213562373095e+00 & 0 \\ 0 & 0 & -6.972611193684197e-32 \end{pmatrix}.$$

An interesting thing has happened: small $\kappa_{scaled}(A)$ has been a safeguard against damaging small singular values during the floating point multiplication $G * A$, but this step has produced a new matrix $(G * A)$ whose rows can be scaled to give a well-conditioned row-scaled matrix. (Note that the last two rows are now numerically mutually orthogonal. In fact, by perturbation theory, the diagonal of $G*A$ represents its singular values to machine precision.) That new condition number will then be a safeguard that will allow annihilation of the position $(1, 3)$ without damaging small singular values. If $G^{(1,3)}$ is the corresponding Givens rotation, then

$$(G * A) * G^{(1,3)} = \begin{pmatrix} 1.000000000000000e+00 & 6.972611193684197e-32 & 0 \\ 0 & 9.999999999999998e-01 & -9.999999999999998e-01 \\ 0 & -4.930380657631323e-32 & -4.930380657631323e-32 \end{pmatrix}.$$

The last two columns of the result are again numerically parallel, but the rows are numerically orthogonal—recall Remark 5.3. Completing the bidiagonalization gives

$$\widetilde{B} = \begin{pmatrix} -1.000000000000000e+00 & 6.972611193684197e-32 & 0 \\ 0 & 9.999999999999998e-01 & 9.999999999999998e-01 \\ 0 & 0 & -9.860761315262645e-32 \end{pmatrix},$$

and the two smallest singular values are computed as

$$\sigma_2(\widetilde{B}) \approx 1.000000000000000e+00, \quad \sigma_3(\widetilde{B}) \approx 6.972611193684196e-32.$$

Example 5.5. (Example 5.2, continued.) The matrix $B = A^{-T}$ is also instructive. Only its largest computed singular value $\sigma_1(A^{-T}) \approx 1.434182936954525e+31$ is accurate; the remaining two are computed by `svd()` as

$$\sigma_2(A^{-T}) \approx 1.669893362298930e+15, \quad \sigma_3(A^{-T}) \approx 3.100161857773022e-17. \tag{38}$$

For comparison, if $B^T = A^{-1}$ is given as input to `svd()`, the corresponding singular values are computed as

$$\sigma_2(A^{-1}) \approx 1.000000000000000e+00, \quad \sigma_3(A^{-1}) \approx 7.187500000000000e-01. \tag{39}$$

Accurate approximations of σ_2 and σ_3 are $\sigma_2(A^{-1}) \approx 1.000000000000000e+00$, $\sigma_3(A^{-1}) \approx 7.071067811865475e-01$. We see that the singular values much smaller than $\|A^{-1}\|_2$ can be severely overestimated as well as underestimated. The fact that merely transposing the matrix may result in completely differently computed singular values clearly shows that the usual statements that deem an algorithm numerically acceptable if it is backward stable in matrix norm are not a guarantee for high accuracy (see (38) and (39)).

Remark 5.6. The problems illustrated here are also responsible for similar loss of accuracy when the bidiagonalization is done using the Householder reflectors. Indeed, it is easily seen that in the formula $A' = A(I - vv^T) = A - (Av)v^T$, small columns of A may be changed so that the condition number of the columns scaled A' becomes extremely large, similarly as in Example 5.2.

Remark 5.7. For another example on how the rotation angle may impact the accuracy, and for enlightening discussion, see Stewart [1995].

ACKNOWLEDGMENTS

The author thanks Jesse Barlow (Penn State), Jim Demmel (Berkeley), Beresford Parlett (Berkeley), and Krešimir Veselić (Hagen) for fruitful discussions and encouraging comments. He is also indebted to the anonymous referees and the algorithm section editor Tim Hopkins for constructive criticism that helped improve the presentation of the method and the software implementation.

REFERENCES

E. Anderson, Z. Bai, C. Bischof, L. S. Blackford, J. Demmel, J. J. Dongarra, J. Du Croz, S. Hammarling, A. Greenbaum, A. McKenney, and D. Sorensen. 1999. *LAPACK Users' Guide* (3rd ed.). Society for Industrial and Applied Mathematics, Philadelphia, PA.

J. Barlow. 2002. More accurate bidiagonal reduction for computing the singular value decomposition. *SIAM Journal on Matrix Analysis and Applications* 23, 761–798.

J. Barlow and J. Demmel. 1990. Computing accurate eigensystems of scaled diagonally dominant matrices. *SIAM Journal on Numerical Analysis* 27, 3, 762–791.

J. L. Barlow, N. Bosner, and Z. Drmač. 2005. A new stable bidiagonal reduction algorithm. *Linear Algebra and Its Applications* 397, 35–84.

M. Bečka, G. Okša, and M. Vajteršic. 2015. New dynamic orderings for the parallel one-sided block-Jacobi SVD algorithm. *Parallel Processing Letters* 25, 1550003-1–1550003-19.

C. H. Bischof and G. Quintana-Orti. 1998a. Algorithm 782: Codes for rank-revealing QR factorizations of dense matrices. *ACM Transactions on Mathematical Software* 24, 2, 254–257.

C. H. Bischof and G. Quintana-Orti. 1998b. Computing rank-revealing QR factorizations of dense matrices. *ACM Transactions on Mathematical Software* 24, 2, 226–253.

N. Bosner and Z. Drmač. 2005. On accuracy properties of one-sided bidiagonalization algorithm and its applications. In *Applied Mathematics and Scientific Computing*, Z. Drmač, M. Marušić, and Z. Tutek, (Eds.). Springer, 141–150.

P. A. Businger and G. H. Golub. 1965. Linear least squares solutions by householder transformations. *Numerische Mathematik* 7, 269–276.

T. F. Chan. 1982. An improved algorithm for computing the singular value decomposition. *ACM Transactions on Mathematical Software* 8, 72–83.

A. J. Cox and N. J. Higham. 1998. Stability of Householder QR factorization for weighted least squares problems. In *Numerical Analysis (Dundee)*. Pitman Research Notes in Mathematics, Vol. 380. Longman, Harlow, England, 57–73.

J. Demmel. 1989. *On Floating Point Errors in Cholesky*. LAPACK Working Note 14. Computer Science Department, University of Tennessee.

J. Demmel. 1997. *Applied Numerical Linear Algebra*. SIAM.

J. Demmel. 1999. Accurate singular value decompositions of structured matrices. *SIAM Journal on Matrix Analysis and Applications* 21, 2, 562–580.

J. Demmel, L. Grigori, M. Gu, and H. Xiang. 2015. Communication avoiding rank revealing QR factorization with column pivoting. *SIAM Journal on Matrix Analysis and Applications* 36, 1, 55–89.

J. Demmel, M. Gu, S. Eisenstat, I. Slapničar, K. Veselić, and Z. Drmač. 1999. Computing the singular value decomposition with high relative accuracy. *Linear Algebra and Its Applications* 299, 21–80.

J. Demmel and W. Kahan. 1990. Accurate singular values of bidiagonal matrices. *SIAM Journal on Scientific and Statistical Computing* 11, 5, 873–912.

J. Demmel and K. Veselić. 1992. Jacobi's method is more accurate than QR. *SIAM Journal on Matrix Analysis and Applications* 13, 4, 1204–1245.

F. M. Dopico, J. M. Molera, and J. Moro. 2003. An orthogonal high relative accuracy algorithm for the symmetric eigenproblem. *SIAM Journal on Matrix Analysis and Applications* 25, 2, 301–351.

Z. Drmač. 1994. *Computing the Singular and the Generalized Singular Values*. Ph.D. Dissertation, Lehrgebiet Mathematische Physik, Fernuniversität Hagen, Germany.

Z. Drmač. 1997. Implementation of Jacobi rotations for accurate singular value computation in floating point arithmetic. *SIAM Journal on Scientific Computing* 18, 1200–1222.

Z. Drmač. 1999. A posteriori computation of the singular vectors in a preconditioned Jacobi SVD algorithm. *IMA Journal of Numerical Analysis* 19, 191–213.

Z. Drmač. 2000. On principal angles between subspaces of Euclidean space. *SIAM Journal on Matrix Analysis and Applications* 22, 173–194.

Z. Drmač. 2015. SVD of Hankel matrices in Vandermonde-Cauchy product form. *Electronic Transactions on Numerical Analysis* 44, 593–623.

Z. Drmač. 2016. *xGESVDQ: A Software Implementation of the QR–Preconditioned QR SVD Method for Computing the Singular Value Decomposition User Guide*. Technical Report. Department of Mathematics, Faculty of Science, University of Zagreb, Croatia.

Z. Drmač and Z. Bujanović. 2008. On the failure of rank-revealing QR factorization software—a case study. *ACM Transactions on Mathematical Software* 35, 2, 12:1–12:28.

Z. Drmač and K. Veselić. 2000. Approximate eigenvectors as preconditioner. *Linear Algebra and Its Applications* 309, 13, 191–215.

Z. Drmač and K. Veselić. 2008a. New fast and accurate Jacobi SVD algorithm: I. *SIAM Journal on Matrix Analysis and Applications* 29, 4, 1322–1342.

Z. Drmač and K. Veselić. 2008b. New fast and accurate Jacobi SVD algorithm: II. *SIAM Journal on Matrix Analysis and Applications* 29, 4, 1343–1362.

J. A. Duersch and M. Gu. 2015. *True BLAS-3 Performance QRCP Using Random Sampling*. ArXiv e-prints.

S. C. Eisenstat and I. C. F. Ipsen. 1995. Relative perturbation techniques for singular value problems. *SIAM Journal on Numerical Analysis* 32, 6, 1972–1988.

A. George, K. Ikramov, and A. B. Kucherov. 2000. Some properties of symmetric quasi-definite matrices. *SIAM Journal on Matrix Analysis and Applications* 21, 4, 1318–1323.

G. H. Golub and W. Kahan. 1965. Calculating the singular values and pseudo-inverse of a matrix. *SIAM Journal on Numerical Analysis* 2, 2, 205–224.

G. H. Golub and C. F. Van Loan. 1989. *Matrix Computations* (2nd ed.). Johns Hopkins University Press, Baltimore, MD.

B. Großer and B. Lang. 2003. An $O(n^2)$ algorithm for the bidiagonal SVD. *Linear Algebra and Its Applications* 358, 13, 45–70.

N. J. Higham. 2000. QR factorization with complete pivoting and accurate computation of the SVD. *Linear Algebra and Its Applications* 309, 153–174.

Intel. 2015. Intel® Math Kernel Library 11.2 Update 3. Intel.

W. Kahan. 2008. Why Can I Debug Some Numerical Programs That You Can't? Retrieved June 6, 2017, from https://www.cs.berkeley.edu/~wkahan/Stnfrd50.pdf.

D. Knuth. 1977. The Correspondence Between Donald E. Knuth and Peter van Emde Boas on Priority Deques During the Spring of 1977. Retrieved June 6, 2017, from https://staff.fnwi.uva.nl/p.vanemdeboas/knuthnote.pdf.

H. Ltaief, P. Luszczek, and J. Dongarra. 2013. High-performance bidiagonal reduction using tile algorithms on homogeneous multicore architectures. *ACM Transactions on Mathematical Software* 39, 3, 16:1–16:22.

M. J. D. Powell and J. K. Reid. 1969. On applying Householder transformations to linear least squares problems. In *Information Processing 68: Proceedings of the International Federation of Information Processing Congress, Edinburgh, 1968*. 122–126.

R. Ralha. 2003. One-sided reduction to bidiagonal form. *Linear Algebra and Its Applications* 358, 13, 219–238.

G. W. Stewart. 1980. The efficient generation of random orthogonal matrices with an application to condition estimators. *SIAM Journal of Numerical Analysis* 17, 3, 403–409.

G. W. Stewart. 1995. *QR Sometimes Beats Jacobi*. Technical Report TR–3434. Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park.

G. W. Stewart. 1997a. *A Gap-Revealing Matrix Decomposition*. Technical Report TR–3771. Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park.

G. W. Stewart. 1997b. *The QLP Approximation to the Singular Value Decomposition*. Technical Report TR–97–75. Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park.

G. W. Stewart and J. Sun. 1990. *Matrix Perturbation Theory*. Academic Press, Boston, MA.

A. Tomas, Z. Bai, and V. Hernandez. 2012. Parallelization of the QR decomposition with column pivoting using column cyclic distribution on multicore and GPU processors. In *High Performance Computing for Computational Science—VECPAR 2012*. Lecture Notes in Computer Science, Vol. 7851. Springer, 50–58.

A. van der Sluis 1969. Condition numbers and equilibration of matrices. *Numerische Mathematik* 14, 14–23.

K. Veselić. 1996. A note on the accuracy of symmetric eigenreduction algorithms. *Electronic Transactions on Numerical Analysis* 4, 37–45.

K. Veselić and V. Hari. 1989. A note on a one-sided Jacobi algorithm. *Numerische Mathematik* 56, 627–633.

Received February 2016; revised February 2017; accepted February 2017