A comparison between the complex symmetric based and classical computation of the singular value decomposition of normal matrices

Ferranti Micol, Le Thanh Hieu, and Vandebril Raf

Micol Ferranti
Department of Computer Science
KU Leuven, Belgium
Micol.Ferranti@cs.kuleuven.be

Hieu Le Thanh
Department of Computer Science
KU Leuven, Belgium
Hieu.LeThanh@cs.kuleuven.be

Raf Vandebril
Department of Computer Science
KU Leuven, Belgium
Raf.Vandebril@cs.kuleuven.be

Abstract

An algorithm for computing the singular value decomposition of normal matrices using intermediate complex symmetric matrices is proposed. This algorithm, as most eigenvalue and singular value algorithms, consists of two steps. It is based on combining the unitarily equivalence of normal matrices to complex symmetric tridiagonal form with the symmetric singular value decomposition of complex symmetric matrices.

Numerical experiments are included comparing several algorithms, with respect to speed and accuracy, for computing the symmetric singular value decomposition (also known as the Takagi factorization). Next we compare the novel approach with the classical Golub-Kahan method for computing the singular value decomposition of normal matrices: it is faster, consumes less memory, but on the other hand the results are significantly less accurate.

Article information

- Ferranti, Micol; Le, Thanh Hieu; Vandebril, Raf. A comparison between the complex symmetric based and classical computation of the singular value decomposition of normal matrices, Numerical Algorithms, 2013.
- The content of this article is identical to the content of the published paper, but without the final typesetting by the publisher.
- Journal's homepage: http://link.springer.com/journal/11075
- Published version: http://dx.doi.org/10.1007/s11075-013-9777-9
- KU Leuven's repository url: https://lirias.kuleuven.be/handle/123456789/427977

Numerical Algorithms manuscript No.
(will be inserted by the editor)

A comparison between the complex symmetric based and classical computation of the singular value decomposition of normal matrices

Micol Ferranti · Thanh Hieu Le · Raf Vandebril

Received: date / Accepted: date

Abstract An algorithm for computing the singular value decomposition of normal matrices using intermediate complex symmetric matrices is proposed. This algorithm, as most eigenvalue and singular value algorithms, consists of two steps. It is based on combining the unitarily equivalence of normal matrices to complex symmetric tridiagonal form with the symmetric singular value decomposition of complex symmetric matrices.

Numerical experiments are included comparing several algorithms, with respect to speed and accuracy, for computing the symmetric singular value decomposition (also known as the Takagi factorization). Next we compare the novel approach with the classical Golub-Kahan method for computing the singular value decomposition of normal matrices: it is faster, consumes less memory, but on the other hand the results are significantly less accurate.

Keywords singular value decomposition, symmetric singular value decomposition, Takagi factorization, normal matrix.

1 Introduction

Normal matrices (matrices commuting with their conjugate transpose) play a fundamental role in various applications (see, e.g., [10, 18, 23] and the references therein). Not only the link to

The research was partially supported by the Research Council KU Leuven, projects OT/11/055 (Spectral Properties of Perturbed Normal Matrices and their Applications), PFV/10/002 (Optimization in Engineering, OPTEC), by the Fund for Scientific Research–Flanders (Belgium) project G034212N (Reestablishing Smoothness for Matrix Manifold Optimization via the Resolution of Singularities), and by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office, Belgian Network DYSCO (Dynamical Systems, Control, and Optimization).

T. H. Le
Department of Computer Science, KU Leuven, Celestijnenlaan 200A, 3001 Heverlee, Belgium
E-mail: ThanhHieu.Le@cs.kuleuven.be

M. Ferranti
Department of Computer Science, KU Leuven, Celestijnenlaan 200A, 3001 Heverlee, Belgium
E-mail: Micol.Ferranti@cs.kuleuven.be

R. Vandebril
Department of Computer Science, KU Leuven, Celestijnenlaan 200A, 3001 Heverlee, Belgium
E-mail: Raf.Vandebril@cs.kuleuven.be

applications, but also their excellent numerical behavior makes them an appealing class of matrices. An extensive list of properties of normal matrices can be found in [8, 15–17, 19]. Amongst all normal matrices Hermitian, skew-Hermitian, and unitary matrices are the most common ones and as such they were studied intensively, e.g., as intermediate matrices in eigenvalue computations [4, 11, 22, 26].

Adapting and tuning the current standard algorithm for computing the singular value decomposition (SVD for short) to general normal matrices has not yet been considered. The classical Golub-Kahan SVD algorithm [2, 12, 13] is applicable to arbitrary matrices and does not exploit the normality of the matrix in any way. It first reduces the input matrix by unitary equivalences to bidiagonal form, after which the SVD of this intermediate bidiagonal matrix is computed by, e.g., a QR based method, a divide and conquer approach, a bisection based algorithm, Jacobi iterations, and other methods (see, e.g., [5] and the references therein).

It is possible to exploit the normality, however. From [25] it is known that every normal matrix is unitarily equivalent to a complex symmetric tridiagonal matrix. Making use of the complex symmetric tridiagonal matrix as intermediate matrix structure and combining this with methods for diagonalizing it, see, e.g., [1, 28, 30] a new algorithm for computing the SVD of a normal matrix is obtained.

We combine the unitary equivalence transformation to complex symmetric tridiagonal form with three algorithms for computing the symmetric singular value decomposition (abbreviated as SSVD) of a complex symmetric matrix[1]. The following SSVD algorithms are considered: a divide-and-conquer method (shortened as DAC) [28], which is a recursive approach where the original problem is split in two subproblems of half the dimension; the twisted factorization method (addressed as Twist) [30], which is inspired by the MR^3 idea to accurately retrieve the singular vectors once the singular values are known; and the last method is a modification of the QR algorithm to retain the structure of the tridiagonal complex symmetric matrix [1] (abbreviated as QR). An experimental comparison of these algorithms, with respect to timings and accuracy is presented and the best one is taken to compete with the classical QR inspired SVD algorithm. In the remainder of the text when considering the SVD, we tacitly assume it is computed by an implicit QR based chasing method [14] or the so-called Golub-Kahan approach.

The following notation is used: for a matrix A, A^T denotes its transpose; $\overline{A}$ denotes the element-wise conjugate, and $A^H = \overline{A}^T$ stands for the complex or Hermitian transpose of A. A matrix $C \in \mathbb{C}^{n\times n}$ is said to be complex symmetric if $C = C^T$. In reality one could omit the "complex" in the definition, but as often in the literature symmetric implies the matrix elements to be real, we opt to keep the "complex", indicating that the elements reside in the complex field. Every matrix A admits an SVD: $A = U\Sigma V^H$, with U and V unitary and Σ a diagonal matrix containing the singular values; every complex symmetric matrix admits an SSVD: $C = Q\Sigma Q^T$ with Q unitary and Σ a diagonal matrix having the singular values of C as diagonal elements. For our convenience, we write $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$ where $\sigma_1 \geq \ldots \geq \sigma_n$. Obviously an SSVD is an SVD, but not vice versa.

We focus on the complex symmetric based approach and the paper is organized as follows. The algorithm carrying out the unitary equivalence to tridiagonal form is presented in Section 2. Section 3 details on three common methods for computing the SSVD of complex symmetric tridiagonal matrices. In Section 4 numerical examples are presented to compare the various SSVD algorithms and to compare the SSVD based approach with the classical SVD algorithm. The conclusions are presented in Section 5.

[1] The SSVD is often also named the Takagi factorization [24].

2 Tridiagonal matrices unitarily equivalent to normal matrices

Some essential results on the equivalence between normal matrices and tridiagonal matrices (see [25] for more detail) are summarized in this section.

The equivalence tridiagonalization procedure is quite similar to the tridiagonalization procedure for Hermitian matrices and also resembles the bidiagonalization procedure [14]. In both cases reflectors (also called Householder matrices) are used to create the desired zeros. In the Hermitian case a single operator can be used to enforce zeros above and below the diagonal by a similarity. Here, however, we need equivalences and possibly differing reflectors to accomplish this, just like in the bidiagonalization procedure. A reflector, say P, is typically used for zeroing components in a vector [14, Section 5.1] and is of the form $P = I - (\frac{2}{\mathbf{v}^H\mathbf{v}})\mathbf{v}\mathbf{v}^H$, for some nonzero vector $\mathbf{v}$. The image of $\mathbf{x}$ under P is given by

$$P\mathbf{x} = \mathbf{x} - \frac{2\,\mathbf{v}^H\mathbf{x}}{\mathbf{v}^H\mathbf{v}}\mathbf{v}.$$

For every $\mathbf{x}$ one can easily construct a $\mathbf{v}$ such that $\mathbf{x}$'s image $P\mathbf{x} = \omega\|\mathbf{x}\|\mathbf{e}_1$ becomes a multiple of the identity, where ω is a unimodular factor.

The following algorithm executes the equivalence transformation of an arbitrary matrix to a tridiagonal one. For $N \in \mathbb{C}^{n\times n}$ and $k, l, s, t \in \mathbb{N}, k \le s, l \le t$, we denote by $N_{k:s,l:t}$ the submatrix of N consisting of the rows $k, \ldots, s$ and columns $l, \ldots, t$. For $k = s$ or $l = t$ we write $N_{k,l:t}$ or $N_{k:s,l}$, respectively.

Algorithm 1 *(Equivalence tridiagonalization)*

Input: A matrix $N \in \mathbb{C}^{n\times n}$.
Output: Unitary matrices U, V, and a tridiagonal matrix T such that $N = UTV^H$.
Set $U = V = I_n$, where I_n denotes the identity matrix.
for $k = 1 : n-2$,

1. Compute the reflector $P = I - \alpha\mathbf{v}\mathbf{v}^H$ to zero components in $\mathbf{x} = N_{k+1:n,k}$;
2. Set $N_{k+1:n,k:n} = P^H N_{k+1:n,k:n}$ and $U_{k+1:n,k+1:n} = U_{k+1:n,k+1:n}P$;
3. Compute the reflector $R = I - \alpha\mathbf{w}\mathbf{w}^H$ to zero components in $\mathbf{y} = N^H_{k,k+1:n}$;
4. Set $N_{k:n,k+1:n} = N_{k:n,k+1:n}R$ and $V_{k+1:n,k+1:n} = V_{k+1:n,k+1:n}R$.

end

When only the tridiagonal matrix T is desired, e.g., one is only interested in the singular values and not in the singular vectors, it is not necessary to compute U and V explicitly. In this case most of the work in each iteration goes to updating the submatrices $N_{k+1:n,k:n}$ and $N_{k:n,k+1:n}$. For each of them, take $N_{k+1:n,k:n}$, the update is executed as follows

$$N_{k+1:n,k:n} = N_{k+1:n,k:n} - \alpha\mathbf{v}\left(\mathbf{v}^H N_{k+1:n,k:n}\right). \tag{1}$$

This requires approximately (we only consider the dominant terms) $2(n-k)^2$ operations for the dot products $\mathbf{v}^H N_{k+1:n,k:n}$, $(n-k)^2$ for the outer product $(\alpha\mathbf{v})\left(\mathbf{v}^H N_{k+1:n,k:n}\right)$, and $(n-k)^2$ for the matrix subtraction. The dominant factor in the cost for computing (1) is hence $4(n-k)^2$ flops. So each update, left and right takes $4(n-k)^2$ flops. In total we get thus

$$8\sum_{k=1}^{n-2}(n-k)^2 \sim \frac{8}{3}n^3$$

floating point operations for computing the tridiagonal matrix only.

If the unitary matrices are desired the reflectors need to be accumulated. This can be done efficiently in $\frac{4}{3}n^3$ for each unitary matrix, resulting in a total of $\frac{16}{3}n^3$ flops. Comparing this to the bidiagonalization algorithm, we get an identical cost (see [5, 14]). If one, later on, wishes to apply these reflectors on another matrix one does not first accumulate them and then perform a matrix–matrix multiplication, but one applies them directly on the matrix in $2n^3$ operations.

Algorithm 1 transforms a normal matrix to a complex tridiagonal one which is not necessarily symmetric. Suppose the tridiagonal matrix T in Algorithm 1 has diagonal elements $\alpha_i, i = 1, \ldots, n$, subdiagonal elements β_i, and superdiagonal elements γ_i. We assume the sub- and superdiagonal elements different from zero. There is no loss of generality in requesting the tridiagonal matrix to be irreducible otherwise the problem can be decoupled into smaller tridiagonal matrices. It was proved in [25, Remark 1, Theorem 8] that the corresponding off-diagonal elements for a normal matrix have identical absolute values $|\beta_i| = |\gamma_i|, \forall i = 1, \ldots, n-1$. So, set $E = \operatorname{diag}(1, \frac{\beta_1}{\gamma_1}, \ldots, \frac{\beta_1 \ldots \beta_{n-1}}{\gamma_1 \ldots \gamma_{n-1}})$ then E is unitary, diagonal, and $T = SE$ is a *complex symmetric unitary decomposition* of T, i.e., S is complex symmetric and tridiagonal (see [25, Subsection 5.2] and also [9, 16]). More precisely, let $\gamma_0 = \beta_0 = 1$, the elements of S are defined by

$$S_{ij} = \begin{cases} 0 & \text{if} \quad |i-j| > 1, \\ \alpha_i \frac{\prod_{k=0}^{i-1} \gamma_k}{\prod_{k=0}^{i-1} \beta_k} & \text{if} \quad 1 \leq i = j \leq n, \\ \beta_i \frac{\prod_{k=0}^{i-1} \gamma_k}{\prod_{k=0}^{i-1} \beta_k} & \text{if} \quad |i-j| = 1. \end{cases}$$

Symmetrizing nonsymmetric tridiagonal matrices is quite often prone to numerical instabilities. We stress, however, that in this case the scaling matrix is unitary, and as such amounts to a stable scaling of the tridiagonal matrix. Moreover, every matrix is equivalent to a complex symmetric matrix, but normal matrices are unitarily equivalent. We will exploit the complex symmetry in the development of an alternative algorithm for computing the SVD of normal matrices.

3 Computing the SSVD of complex symmetric tridiagonal matrices

Take $S \in \mathbb{C}^{n \times n}$ complex symmetric and tridiagonal. We would like to find the unitary matrix Q such that $Q^T S Q = \Sigma \doteq \operatorname{diag}(\sigma_1, \ldots, \sigma_n)$, where Σ contains the singular values $\sigma_i \in \mathbb{R}, \forall i = 1, \ldots, n$ ordered as $\sigma_1 \geq \ldots \geq \sigma_n \geq 0$.

We summarize five numerical methods for computing the SSVD of complex symmetric tridiagonal matrices, amongst which we will use three: a divide-and-conquer method (DAC) [28], a QR based iteration (QR) [1, 20, 21, 29], and a hybrid algorithm: the twisted factorization method (Twist) [30]. Of course, if one separates the singular value and singular vector computations there is a whole variety of combinations. We chose to focus on those algorithms available and studied in the literature. For all these methods it holds already that they are more memory efficient as only one matrix of singular vectors needs to be stored instead of two for the standard SVD.

The basic idea of the DAC method [28] is to apply the divide-and-conquer method for computing the eigenvalues and eigenvectors of the symmetric matrix SS^H. This matrix is teared into two pentadiagonal submatrices of half the dimension whose eigenvalues are then computed recursively. The eigenvalues of the smaller matrices are then recombined to retrieve those of SS^H by using four rank-one modifications. This method needs $3n^3$ operations or a little less in case one of the core matrix–matrix multiplications can be performed more efficiently, e.g., presence of zeros, or premature deflations. The DAC approach is the fastest, but we will see, in Section 4

that it becomes numerically unreliable when some of the subdiagonal elements become relatively small [28].

The QR based approach proposed in [1, 21, 29] translates the algorithm for computing the singular values of a bidiagonal matrix to the tridiagonal setting. As one relies on the QR factorization of SS^H this results in a double shifted QR step. In these articles an implicit version is presented where the chasing relies on, and ensures at the same time the outcome to be of complex symmetric tridiagonal form again. This method needs $\mathcal{O}(n^2)$ flops for computing the singular values and $\mathcal{O}(n^3)$ flops for computing the singular vectors if all the transformation matrices are accumulated. On average, it is stated in [21] that this algorithm needs only $6n^3$ flops; the numerical experiments in Section 4.2 support this claim.

The twisted factorization method [30] focuses on computing the singular vectors assuming thereby the singular values are known. This method is inspired on the MR^3 algorithm [6, 7] for computing the eigenvectors of symmetric tridiagonal matrices up to high relative precision. Its complexity is $85n$ for computing a single eigenvector. It must be stressed, however, that this is under the assumption that all singular values are separated well-enough. We will see in Section 4, Figure 2 that this method's accuracy is highly related to the clustering of the singular values and suffers from singular values being too close.

In [20] and [3] other QR inspired approaches are presented. These algorithms execute transformations of the form Q^TSQ to retrieve the singular vectors (instead of unitary similarity in the QR case). These algorithms do, however, suffer from numerical instabilities, or are not able to tackle the general setting and will therefore not be discussed in the remainder of the text.

Combining the tridiagonalization and the SSVD computation gives us the following alternative algorithm.

Algorithm 2 *(Compute the SVD of a normal matrix)*

Input: A normal matrix N.
Output: Singular value decomposition of $N = U\Sigma V^H$.

1. Tridiagonalize $N = U_T T V_T^H$ using Algorithm 1.
2. Compute the symmetrization of $T = SE$.
3. Compute the singular values and singular vectors of S.
4. Recombine everything to obtain the singular value decomposition of N

The main cost of Algorithm 2 is the tridiagonalization and the computation of the singular vectors if required, all summing up to $\mathcal{O}(n^3)$ terms. Let us compare the SSVD and the SVD computations based on the QR algorithm in detail as these algorithms are similar in nature and, moreover, we will see in Section 4 that the QR based SSVD is the most reliable. We discuss both phases namely the tri- and bidiagonalization and the diagonalization phase separately. The bidiagonalization and tridiagonalization both take $\frac{8}{3}n^3$. We do not accumulate the matrices U_T and V_T, we will execute them on the singular vectors retrieved from the SVD and SSVD computations which costs twice $2n^3$.

A single QR sweep in the bidiagonal case only uses $\mathcal{O}(n)$ operations to modify B, which is similar to the complex symmetric tridiagonal setting. Again the dominant cost is hidden in the updating of the singular vectors. For the bidiagonal case, in each QR step $6n^2$ operations are needed to update U and a similar cost to update V, so a total of $12n^2$ for both matrices. In [14] an approximate 1.2 iterations are assumed before convergence occurs, leading to an overall complexity estimate for retrieving the SVD of a bidiagonal matrix of approximately $15n^3$.

For the complex symmetric tridiagonal matrix [21, 27] the chasing is more involved as each chase step corresponds to a double shifted QR step. Performing the operations on the tridiagonal matrix is again $\mathcal{O}(n)$ though with a larger constant than in the bidiagonal case (roughly

the double). Updating of the singular vectors takes now $11n^2$ according to a chasing based on reflectors as proposed in [21]. This $11n^2$ is only a modest improvement with respect to the $12n^2$, however, the usage of double shifts results in a significant reduction of the number of iterations. In [21] an overall cost of only $6n^3$ is claimed. Besides the lower computational cost one should not forget the reduced memory storage when comparing the SSVD with the SVD approach while running QR steps.

In total, combining both phases the singular vectors retrieved from the bi- and tridiagonal (S)SVD still need to be updated. For the SSVD we have

$$N = U_T T V_T^H = U_T S(EV_T^H) = (U_T Q)\Sigma(Q^T EV_T^H).$$

Since E is diagonal, the matrix multiplication $T = SE$ costs only $\mathcal{O}(n^2)$ flops, the remaining updates cost just as in the SVD case $4n^3$. So, in total we arrive at approximate costs of $21n^3$ for the SVD ($15n^3$ for the diagonalization, $4n^3$ for updating the singular vectors, and $8/3n^3$ for the bidiagonalization) and $12n^3$ for the SSVD ($6n^3$ for the diagonalization, $4n^3$ for updating the singular vectors, and $8/3n^3$ for the tridiagonalization) of an arbitrary normal matrix (assuming the $6n^3$ to be valid). In Section 4.2, fortran codes of both methods are compared with respect to speed and accuracy. The results confirm a speed-up of approximately 50%, but reveal on the other hand that the complex symmetric approach is significantly less accurate.

4 Numerical experiments

We have executed two sets of experiments. Section 4.1 compares the three methods for computing the SSVD. Based on the accuracy results in this section we opted to continue working with the QR based SSVD to compare against the SVD in Section 4.2. The numerical experiments ran on an Intel® Dual Core™@1.85 GHz. The algorithms in Section 4.1 were implemented in MATLAB (R2012a) and based on software (if available) from the corresponding manuscripts. The experiments in Section 4.2 were conducted in Fortran and the SSVD approached was compared with the Lapack SVD implementation.

In the experiments, a normal matrix N is randomly created as $N = Q^H DQ$ where D is a complex diagonal matrix whose diagonal elements have random normally distributed real and imaginary parts in $(0, 1)$. The unitary matrix Q is the Q-factor of the QR factorization of a random complex square matrix.

For each experiment, the relative backward error is computed as

$$\frac{||N - U\Sigma V^H||_2}{||N||_2},$$

where N is the original normal matrix and $U\Sigma V^H$ the computed SVD. The relative singular value error is given by

$$\max_{i=1,\ldots,n} \frac{|\sigma_i - \tilde{\sigma}_i|}{|\sigma_i|},$$

where $\tilde{\sigma}_i$ are the computed singular values, and σ_i are the exact ones. The exact singular values are known by construction, or are computed via the variable precision arithmetic in MATLAB.

When results related only to singular values are depicted, we do not show the performances of the QR and Twist methods separately, as the Twist algorithm relies on the QR to compute the singular values[2].

[2] This does not necessarily mean that their computed singular values are identical, as delayed convergence of the QR method results in random shifts, which typically differ every run.

4.1 Comparing three algorithms for retrieving the SSVD

We compare the QR based, DAC, and Twist method for computing the SSVD. In a first experiment their speed and accuracy are examined for computing the SSVD of random tridiagonal complex symmetric matrices. As we will see, the DAC method is not reliable in terms of accuracy, because of its sensitivity to small off-diagonal elements, although it is the most efficient in terms of speed. In a second experiment we investigate the sensitivity of the Twist method with respect to clustered singular values.

Figure 1 depicts the running times, relative singular value errosr, and relative backward errors for the QR based, the DAC based, and the Twist method when computing the SSVD of random tridiagonal complex symmetric matrices. The numerical results show that overall the QR approach delivers the most accurate results, but is also the slowest one.

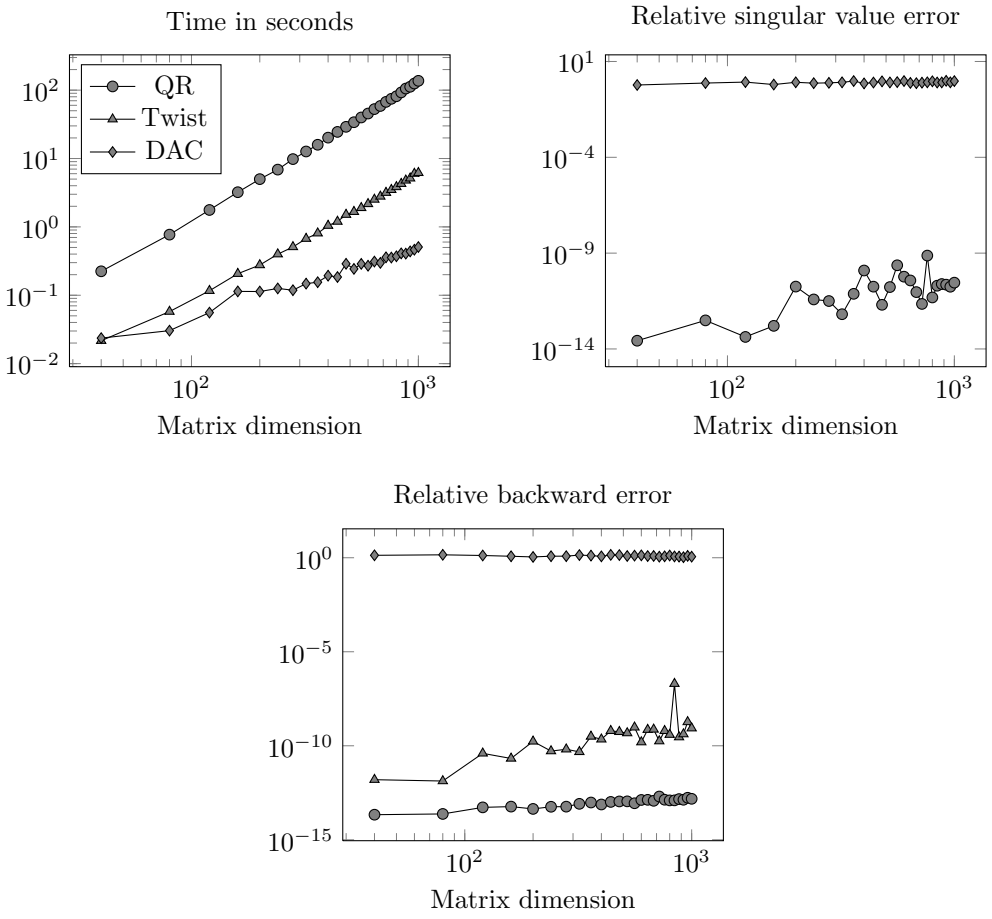

Fig. 1 Running times and relative errors of the QR based, the DAC, and Twist method when applied on a complex symmetric tridiagonal matrix (the legend of the first figure applies to all).

In a second experiment we generate matrices with clustered singular values, illustrating that the accuracy of the Twist method is affected by this. The order of the matrix is fixed at $n = 200$

and the matrix equals

$$S = \left[\begin{array}{c|c|c} S_0 & \epsilon & \\ \hline \epsilon & \ddots & \\ \hline & \epsilon & S_0 \end{array}\right]$$

where S_0 is the 20×20 symmetric tridiagonal matrix having diagonal entries equal to 2, sub- and superdiagonal entries equal to 1, and $\epsilon = 2^{-k}$ for $k = 0, 5, 10, \ldots, 55$, such that the smallest value of ϵ just passes the machine precision. In Figure 2 the results are shown, with respect to the parameter k. The QR method clearly outperforms the Twist approach when it comes to computing singular vectors accurately. For the values $k = 50, 55$, we even get NaN's (Not a Number) for the Twist algorithm.

Based on these experiments we believe that the best competitor to the classical SVD algorithm is the QR based algorithm from [1, 21]. Let us compare both approaches in detail in the Section 4.2.

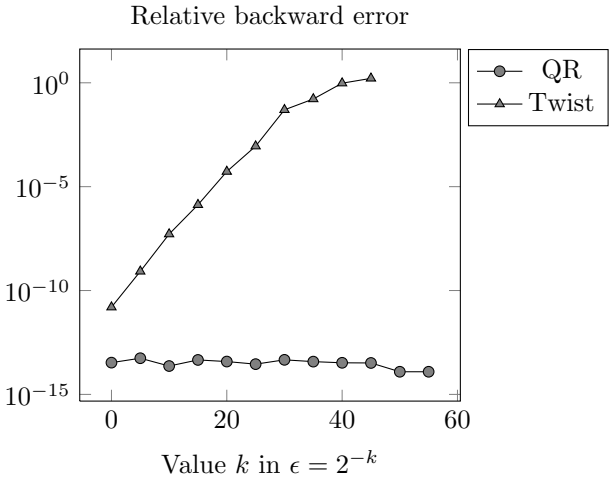

Fig. 2 Relative backward error of the QR based and Twist method when dealing with clustered singular values.

4.2 Computing the singular value decomposition of normal matrices

We analyze the speed and accuracy of the complex symmetric QR based approach with respect to the classical SVD algorithm for retrieving the singular value decomposition of normal matrices. In the remainder of the text we address these approaches as the SSVD (complex symmetric based) and the SVD methods. Algorithm 2 was encoded in Fortran and was used to compare the new approach with Lapack's SVD implementation. Each method includes the two phases: bi- or tridiagonalizing of the normal matrices, and diagonalizing the resulting bi- or tridiagonal matrices.

Size	Time SSVD	Time SVD	Error SSVD	Error SVD
100	5.4 e−02	4.8 e−02	3.2 e−13	2.5 e−15
200	3.7 e−01	3.6 e−01	3.9 e−13	3.1 e−15
300	1.2 e+00	1.4 e+00	8.7 e−13	4.0 e−15
400	2.9 e+00	3.5 e+00	7.4 e−11	4.3 e−15
500	5.8 e+00	7.1 e+00	6.1 e−13	4.8 e−15
600	1.0 e+01	1.2 e+01	1.8 e−12	5.0 e−15
700	1.6 e+01	1.9 e+01	7.1 e−12	5.5 e−15
800	2.4 e+01	2.8 e+01	1.5 e−11	5.8 e−15
900	3.5 e+01	4.2 e+01	8.5 e−12	6.1 e−15
1000	4.7 e+01	5.8 e+01	1.0 e−10	6.3 e−15
1500	1.5 e+02	1.9 e+02	2.9 e−11	7.8 e−15
2000	3.9 e+02	5.5 e+02	1.0 e−10	1.0 e−14
2500	7.8 e+02	9.7 e+02	7.6 e−11	1.1 e−14
3000	1.4 e+03	1.8 e+03	9.9 e−10	1.2 e−14
4000	4.1 e+03	9.5 e+03	5.6 e−11	1.4 e−14
5000	8.6 e+03	1.4 e+04	2.6 e−10	1.4 e−14

Table 1 Timings (in seconds) and the relative backward error of computing the singular value decomposition of normal matrices via either the complex symmetric based (SSVD) and the classical approach (SVD).

Table 1 depicts the timings of both methods in seconds as well as the relative backward errors for sizes 100 up to 5000. Each experiment was executed 3 times and the averages are depicted. The matrices under consideration were random normal constructed as explained in the beginning of the section. As soon as the matrix dimension becomes large enough, the difference in computational complexity between the SSVD ($12n^3$) and the SVD ($21n^3$) approach becomes clear. Unfortunately the table also reveals that the SSVD approach is significantly less accurate, with a deteriorating accuracy for increasing matrix sizes.

5 Conclusions

An alternative algorithm for computing the SVD of normal matrices relying on intermediate complex symmetric matrices was proposed. The results are mixed. With respect to memory consumption the SSVD approach does better, as well as for computational complexity. Further speed-up could be achieved if for instance the divide-and-conquer method could be made more reliable and if the quality of the Fortran code would pair the standards of Lapack. On the other hand, considering the accuracy of both approaches the classical SVD clearly outperforms the SSVD method.

This approach could also be used to retrieve the eigenvalues of normal matrices, as the SVD of a normal matrix closely links to the eigenvalue decomposition. If A is a normal matrix, having distinct singular values and suppose $A = U\Sigma V^H$ is a singular value decomposition then $A = U\Delta U^H$, where $\Delta = \Sigma(V^H U)$, is an eigenvalue decomposition of A. Further research is required, however, as clustered eigenvalues have a non-neglectable impact on the computations and the accuracy.

Acknowledgements

The authors wish to thank Nick Vannieuwenhoven for the stimulating discussions on this subject. Furthermore, the authors value the editor and the referees careful reading and numerous suggestions which led to major changes and greatly improved the presentation of the paper.

References

1. Bunse-Gerstner, A., Gragg, W.B.: Singular value decompositions of complex symmetric matrices. Journal of Computational and Applied Mathematics **21**, 41–54 (1988)
2. Chan, T.: An improved algorithm for computing the singular value decomposition. ACM Transactions on Mathematical Software **8**(1), 72–83 (1982)
3. Cullum, J.K., Willoughby, R.A.: A QL procedure for computing the eigenvalues of complex symmetric tridiagonal matrices. SIAM Journal on Matrix Analysis and Applications **17**(1), 83–109 (1996)
4. Delvaux, S., Van Barel, M.: Eigenvalue computation for unitary rank structured matrices. Journal of Computational and Applied Mathematics **213**(1), 268–287 (2008)
5. Demmel, J.W.: Applied Numerical Linear Algebra. SIAM, Philadelphia, Pennsylvania, USA (1997)
6. Dhillon, I.S.: A new $O(n^2)$ algorithm for the symmetric tridiagonal eigenvalue/eigenvector problem. Ph.D. thesis, Dept. of Computer Science, University of California, Berkeley (1989)
7. Dhillon, I.S., Parlett, B.N.: Multiple representations to compute orthogonal eigenvectors of symmetric tridiagonal matrices. Linear Algebra and its Applications **387**, 1–28 (2004)
8. Elsner, L., Ikramov, K.D.: Normal matrices: An update. Linear Algebra and its Applications **285**, 291–303 (1998)
9. Faßbender, H., Ikramov, K.D.: A note on an unusual type of polar decomposition. Linear Algebra and its Applications **429**, 42–49 (2008)
10. Freund, R.W.: On Conjugate Gradient type methods and polynomial preconditioners for a class of complex non-Hermitian matrices. Numerische Mathematik **57**(1), 285–312 (1990)
11. Gemignani, L.: A unitary Hessenberg QR-based algorithm via semiseparable matrices. Journal of Computational and Applied Mathematics **184**, 505–517 (2005)
12. Golub, G.H., Kahan, W.: Calculating the singular values and pseudo-inverse of a matrix. SIAM Journal on Numerical Analysis **2**, 205–224 (1965)
13. Golub, G.H., Reinsch, C.: Singular value decomposition and least squares solutions. Numerische Mathematik **14**(5), 403–420 (1970)
14. Golub, G.H., Van Loan, C.F.: Matrix Computations, third edn. Johns Hopkins University Press, Baltimore, Maryland, USA (1996)
15. Grone, R., Johnson, C.R., Sa, E.M., Wolkowicz, H.: Normal matrices. Linear Algebra and its Applications **87**, 213–225 (1987)
16. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (1985)
17. Horn, R.A., Johnson, C.R.: Topics in Matrix Analysis. Cambridge University Press, Cambridge (1991)
18. Huckle, T.: The Arnoldi method for normal matrices. SIAM Journal on Matrix Analysis and Applications **15**(2), 479–489 (1994)
19. Ikramov, K.D.: Short communications unitary-triangular factorizations of a normal matrix. Computational Mathematics and Mathematical Physics **33**(3), 407–410 (1993)
20. Luk, F., Qiao, S.: Using complex-orthogonal transformations to diagonalize a complex symmetric matrix. In: Spie proceedings the international society for optical engineering, pp. 418–425 (1997)
21. Luk, F.T., Qiao, S.: A fast singular value algorithm for Hankel matrices. In: V. Olshevsky (ed.) Fast Algorithms for Structured Matrices: Theory and Applications, *Contemporary Mathematics*, vol. 323, pp. 169–177. American Mathematical Society (2003)
22. Parlett, B.N.: The Symmetric Eigenvalue Problem, *Classics in Applied Mathematics*, vol. 20. SIAM, Philadelphia, Pennsylvania, USA (1998)
23. Reichel, L., Noschese, S.: The structured distance to normality of Toeplitz matrices with application to preconditioning. Numerical Linear Algebra with Applications (2010). DOI 10.1002/nla.735
24. Takagi, T.: On an algebraic problem related to an analytic Theorem of Carathéodory and Fejér and on an allied theorem of Landau. Japanese Journal of Mathematics **1**, 82–93 (1924)
25. Vandebril, R.: On tridiagonal matrices unitarily equivalent to normal matrices. Linear Algebra and its Applications **432**(12), 3079 – 3099 (2010)
26. Wang, T.L., Gragg, W.B.: Convergence of the unitary QR algorithm with unimodular Wilkinson shift. Mathematics of Computation **72**(241), 375–385 (2003)
27. Wax, M., Shan, T.J., Kailath, T.: Spatio-temporal spectral analysis by eigenstructure methods. IEEE Transactions on Acoustics, Speech and Signal Processing **32**(4), 817–827 (1984)

28. Xu, W., Qiao, S.: A divide-and-conquer method for the Takagi factorization. SIAM Journal on Matrix Analysis and Applications **30**(1), 142–153 (2008)
29. Xu, W., Qiao, S.: A fast symmetric svd algorithm for square hankel matrices. Linear Algebra and its Applications **428**(2-3), 550 – 563 (2008). Special Issue devoted to the Second International Conference on Structured Matrices, Second International Conference on Structured Matrices
30. Xu, W., Qiao, S.: A twisted factorization method for symmetric SVD of a complex symmetric tridiagonal matrix. Numerical Linear Algebra with Applications **16**(10), 801–815 (2009)