A CS decomposition for orthogonal matrices with application to eigenvalue computation

D. Calvetti* L. Reichel† H. Xu‡

Abstract

We show that a Schur form of a real orthogonal matrix can be obtained from a full CS decomposition. Based on this fact a CS decomposition-based orthogonal eigenvalue method is developed. We also describe an algorithm for orthogonal similarity transformation of an orthogonal matrix to a condensed product form, and an algorithm for full CS decomposition. The latter uses mixed shifted and zero-shift iterations for high accuracy. Numerical examples are presented.

Keywords. Orthogonal matrix, Eigenvalue problem, Full CS decomposition, High accuracy
AMS subject classification. 65F15, 15A23, 15A18, 15B10, 65G50, 65F35

1 Introduction

The eigenvalue problem for unitary and orthogonal matrices has many applications, including time series analysis, signal processing, and numerical quadrature; see, e.g., [2, 7, 13, 14] for discussions. These applications, as well as the elegant theory, have spurred the development of numerical methods for these eigenvalue problems. Both QR algorithms and divide-and-conquer methods have been developed; see [1, 3, 8, 9, 10, 11, 12, 15, 18, 19]. Both types of methods require initial unitary similarity transformation of a given unitary matrix $Q \in \mathbb{C}^{2n\times 2n}$ to an upper Hessenberg matrix H represented in product form,

$$H = H_1 H_2 \dots H_{2n-1} H_{2n},$$

where

$$H_k = \operatorname{diag}\left[I_{k-1}, \begin{bmatrix} -\gamma_k & \sigma_k \\ \sigma_k & \overline{\gamma_k} \end{bmatrix}, I_{2n-k} \right], \quad |\gamma_k|^2 + \sigma_k^2 = 1, \quad \gamma_k \in \mathbb{C}, \quad \sigma_k > 0, \tag{1}$$

is a complex Householder matrix for $1 \le k \le 2n-1$, and $H_{2n} = \operatorname{diag}\left[I_{2n-1}, -\gamma_{2n} \right]$, with $\gamma_{2n} \in \mathbb{C}$ unimodular, is a truncated complex Householder matrix. The Hessenberg matrix H can be transformed further by unitary similarity transformation into the product form $H_o H_e$, where

$$H_o = H_1 H_3 \dots H_{2n-1}, \quad H_e = H_2 H_4 \dots H_{2n}. \tag{2}$$

The latter transformation was first described by Ammar et al. in [1] for orthogonal matrices of even order $Q \in \mathbb{R}^{2n\times 2n}$, and they applied it to determine the eigenvalues and, if so desired, eigenvectors of Q. Orthogonal matrices of odd order can be reduced to orthogonal matrices of even order by deflation; see below. Subsequently, Bunse–Gerstner and Elsner [4] developed a

*Department of Mathematics, Case Western Reserve University, Cleveland, OH 44106, USA. E-mail: dxc57@case.edu.

†Department of Mathematical Sciences, Kent State University, Kent, OH 44242, USA. E-mail: reichel@math.kent.edu. Research supported in part by NSF grant DMS-1115385.

‡Department of Mathematics, University of Kansas, Lawrence, KS 66045, USA. E-mail: feng@ku.edu. Research supported in part by Fudan University Key Laboratory Senior Visiting Scholar Project.

QZ-type algorithm for the matrix pencil $\{H_o^*, H_e\}$ by exploiting the quasi-diagonal structure of the matrices H_o and H_e. Here and below the superscript * denotes transposition and complex conjugation. A superscript ^{T} will denote transposition only.

This paper describes how an orthogonal matrix Q can be brought into the form H_oH_e directly by an orthogonal similarity transformation. This transformation is related to the transformation used by Bunse–Gerstner and Elsner [4], but with different reduction targets and reduction order. In this way, we obtain a new approach for computing the spectral factorization of an orthogonal matrix based on the product form H_oH_e.

Real eigenvalues, i.e., eigenvalues ± 1, of an orthogonal matrix Q can be removed by deflations; see [1, 8] for discussions. Therefore throughout this paper we assume that the orthogonal matrix Q is of even order, $2n \times 2n$, and does not have real eigenvalues. Then we can apply an orthogonal similarity transformation to the expression H_oH_e to obtain a matrix of the form

$$\Sigma Z \Sigma Z^T, \quad \Sigma = \mathrm{diag}[I_n, -I_n], \quad Z = \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{bmatrix},$$

where the blocks $Z_{11}, Z_{21}, Z_{12}^T, Z_{22}^T$ are $n \times n$ and upper bidiagonal.

Suppose that

$$Z = \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} \begin{bmatrix} \Phi & \Psi \\ \Psi & -\Phi \end{bmatrix} \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix}^T$$

is a full CS decomposition, where U_1, U_2, V_1, V_2 are orthogonal,

$$\Phi = \mathrm{diag}[\phi_1, \ldots, \phi_n], \qquad \Psi = \mathrm{diag}[\psi_1, \ldots, \psi_n],$$

and the diagonal entries satisfy $\phi_i^2 + \psi_i^2 = 1$ and $\phi_i, \psi_i > 0$ for $i = 1, \ldots, n$. Then

$$\Sigma Z \Sigma Z^T = \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} \begin{bmatrix} \Phi^2 - \Psi^2 & 2\Phi\Psi \\ -2\Phi\Psi & \Phi^2 - \Psi^2 \end{bmatrix} \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix}^T,$$

from which we obtain a real Schur form of Q after a simple permutation. We develop an algorithm to compute a full CS decomposition of the matrix Z and determine a real Schur form of Q from this decomposition. Sutton [16] describes an algorithm for computing a full CS decomposition of a unitary matrix in a 2×2 block form. His algorithm is more reliable than other CS decomposition methods. Our algorithm follows the approach described in [17] for SVD iterations.

An orthogonal matrix is perfectly well-conditioned and all its eigenvalues are unimodular. Therefore, any backward stable method will compute all eigenvalues with high accuracy. However, this does not mean that small real and imaginary parts of the eigenvalues always are computed with high relative accuracy. For this reason, we apply the Demmel–Kahan zero-shift SVD iteration technique [5] to accurately compute small singular values of blocks of Z.

When an orthogonal similarity transformation needs to be generated for the orthogonal eigenproblem, a CS decomposition-based eigenvalue method only requires a pair of orthogonal matrices $\{U_1, U_2\}$ or $\{V_1, V_2\}$ to be determined. This results in an efficient algorithm. Moreover, differently from QR-type algorithms, which normally use unimodular shifts, the CS decomposition-based method uses standard shifts.

We remark that the initial reduction method can be applied to complex unitary matrices as well. The factorization $\Sigma Z \Sigma Z^T$ provides a new condensed form for real orthogonal matrices and therefore is of independent interest. The full CS decomposition algorithm can be applied to solve real orthogonal eigenvalue problems, as well as to compute a CS decomposition of a unitary matrix.

This paper is organized as follows. Section 2 describes the initial orthogonal transformation that determines the representation H_oH_e of the given matrix Q. We explain in Section 3 how a real orthogonal matrix is orthogonally similar to $\Sigma Z \Sigma Z^T$ and why a full CS decomposition of Z gives a Schur form of $\Sigma Z \Sigma Z^T$. Section 4 contains a full CS decomposition algorithm and some implementation details, and Section 5 presents a new algorithm for the orthogonal eigenvalue problem based on the full CS decomposition. This section also contains a few computed examples. Section 6 contains concluding remarks.

In the remainder of this paper, I denotes an identity matrix, e_j is the jth column of I, and $\|\cdot\|_2$ stands for the Euclidean vector norm or associated induced matrix norm. We use MATLAB inspired notation to define a submatrix. For instance for a matrix $A = [a_{ij}]$, we let

$$A(i:j,k) := [a_{ik}, \ldots, a_{jk}]^T$$

denote the submatrix that consists of the vector made up of the entries of column k of A in rows i through j. Similarly,

$$A(k, i:j) := [a_{ki}, \ldots, a_{kj}]$$

stands for the vector consisting of the entries of row k and columns i through j.

2 Initial reduction to condensed form

This section describes a process for reducing a $2n \times 2n$ unitary matrix Q by unitary similarity transformation to the form $H_o H_e$, where both H_o and H_e are in product form; see (2). Details of the process are shown in the algorithm below, which uses the following elementary matrices:

(a) $H_k(\gamma, \sigma)$ or simply H_k $(1 \le k \le p-1)$ denotes a $p \times p$ Householder matrix analogous to (1). Note that for a vector $x = [x_1, x_2]^T \neq 0$ with $x_2 \ge 0$, we have

$$\gamma = -\frac{x_1}{\sqrt{|x_1|^2 + x_2^2}}, \quad \sigma = \frac{x_2}{\sqrt{|x_1|^2 + x_2^2}} \ge 0.$$

Then

$$\begin{bmatrix} -\gamma & \sigma \\ \sigma & \bar{\gamma} \end{bmatrix}^* x = \begin{bmatrix} \|x\|_2 \\ 0 \end{bmatrix}.$$

For $k = p$, we let $H_p(\gamma) = \mathrm{diag}[I_{p-1}, -\gamma]$. When σ is real, $[H_k(\gamma, \sigma)]^* = \overline{H_k(\gamma, \sigma)}$.

(b) $H_{\ge i}(x)$ is a modified $p \times p$ Householder matrix associated with a vector $x \in \mathbb{C}^{p-i+1}$. It is defined by

$$H_{\ge i}(x) = \mathrm{diag}[I_{i-1}, \tilde{H}]\,\mathrm{diag}[I_{i-1}, \beta, I_{p-i}],$$

where $\tilde{H}$ is a Householder matrix such that $\tilde{H}x = \beta\|x\|_2 e_1$ and $|\beta| = 1$. Therefore,

$$[H_{\ge i}(x)]^* \begin{bmatrix} 0 \\ x \end{bmatrix} = \|x\|_2 e_i.$$

When $i = p$, x is a scalar and $H_{\ge p}(x) = \mathrm{diag}[I_{p-1}, \beta]$, where $\beta = 1$ if $x = 0$; otherwise $\beta = x/|x|$.

For a given vector $x = [y^T, x_i, z^T]^T$ with $y \in \mathbb{C}^{i-1}, z \in \mathbb{C}^{p-i}$, and $[x_i, z^T]^T \neq 0$, we can determine a Householder matrix $H_i(\gamma, \sigma)$ with $\sigma \ge 0$ and a matrix $H_{\ge i+1}(z)$ such that

$$[H_i(\gamma, \sigma)]^* [H_{\ge i+1}(z)]^* x = [y, \alpha, 0]^T, \quad \alpha = \sqrt{|x_i|^2 + \|z\|_2^2} > 0.$$

Algorithm 1. *Given a $2n \times 2n$ unitary matrix Q, the algorithm computes a unitary matrix Θ_0 such that $\Theta_0^* Q \Theta_0 = H_o H_e$.*

0. Set $\Theta_0 = I_{2n}$.

1. For $k = 1, 2, \ldots, n-1$

(a) Determine $H_{\ge 2k} := H_{\ge 2k}(Q(2k:2n, 2k-1))$

(b) Update $Q \leftarrow H_{\ge 2k}^* Q H_{\ge 2k}$, $\Theta_0 \leftarrow \Theta_0 H_{\ge 2k}$

(c) *Determine* $H_{2k-1} := H_{2k-1}(\gamma_{2k-1}, \sigma_{2k-1})$ *with* γ_{2k-1} *and* $\sigma_{2k-1} \geq 0$ *satisfying*
$$\begin{bmatrix} -\gamma_{2k-1} & \sigma_{2k-1} \\ \sigma_{2k-1} & \bar{\gamma}_{2k-1} \end{bmatrix}^* \begin{bmatrix} q_{2k-1,2k-1} \\ q_{2k,2k-1} \end{bmatrix} = e_1$$

(d) *Update* $Q \leftarrow H_{2k-1}^* Q$

(e) *Determine* $H_{\geq 2k+1} := H_{\geq 2k+1}(Q(2k, 2k+1:2n)^*)$

(f) *Update* $Q \leftarrow H_{\geq 2k+1}^* Q H_{\geq 2k+1}$, $\Theta_0 \leftarrow \Theta_0 H_{\geq 2k+1}$

(g) *Determine* $H_{2k} := H_{2k}(\gamma_{2k}, \sigma_{2k})$ *with* γ_{2k} *and* $\sigma_{2k} \geq 0$ *satisfying*
$$\begin{bmatrix} -\gamma_{2k} & \sigma_{2k} \\ \sigma_{2k} & \bar{\gamma}_{2k} \end{bmatrix}^* \begin{bmatrix} q_{2k,2k} \\ q_{2k,2k+1} \end{bmatrix} = e_1$$

(h) *Update* $Q \leftarrow Q H_{2k}^*$

End For

2. (a) *Determine* $H_{\geq 2n} := H_{\geq 2n}(Q(2n, 2n-1))$

(b) *Update* $Q \leftarrow H_{\geq 2n}^* Q H_{\geq 2n}$, $\Theta_0 \leftarrow \Theta_0 H_{\geq 2n}$

(c) *Determine* $H_{2n-1} := H_{2n-1}(\gamma_{2n-1}, \sigma_{2n-1})$ *with* γ_{2n-1} *and* $\sigma_{2n-1} \geq 0$ *satisfying*
$$\begin{bmatrix} -\gamma_{2n-1} & \sigma_{2n-1} \\ \sigma_{2n-1} & \bar{\gamma}_{2n-1} \end{bmatrix}^* \begin{bmatrix} q_{2n-1,2n-1} \\ q_{2n,2n-1} \end{bmatrix} = e_1$$

(d) *Update* $Q \leftarrow H_{2n-1}^* Q$

(e) *Determine* $H_{2n} := H_{2n}(\gamma_{2n})$ *with* $\gamma_{2n} = -q_{2n,2n}$

We illustrate the process with a 6×6 unitary matrix. For $k = 1$, we determine $H_{\geq 2}, H_1, H_{\geq 3}, H_2$ such that

$$Q = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \end{bmatrix} \overset{H_{\geq 2}^* Q H_{\geq 2}}{\longrightarrow} \begin{bmatrix} x & x & x & x & x & x \\ + & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \end{bmatrix}$$

$$\overset{H_1^* H_{\geq 2}^* Q H_{\geq 2}}{\longrightarrow} \begin{bmatrix} 1 & \otimes & \otimes & \otimes & \otimes & \otimes \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \end{bmatrix} \overset{H_{\geq 3}^* H_1^* H_{\geq 2}^* Q H_{\geq 2} H_{\geq 3}}{\longrightarrow} \begin{bmatrix} 1 & \otimes & \otimes & \otimes & \otimes & \otimes \\ 0 & x & + & 0 & 0 & 0 \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \end{bmatrix}$$

$$\overset{H_{\geq 3}^* H_1^* H_{\geq 2}^* Q H_{\geq 2} H_{\geq 3} H_2^*}{\longrightarrow} \begin{bmatrix} 1 & \otimes & \otimes & \otimes & \otimes & \otimes \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & \otimes & x & x & x & x \\ 0 & \otimes & x & x & x & x \\ 0 & \otimes & x & x & x & x \\ 0 & \otimes & x & x & x & x \end{bmatrix},$$

where "0" denotes a zero introduced by transformation, "+" stands for a nonnegative entry, the entries "1" are due the fact that the matrices H_i and Q are unitary, and "$\otimes$" is a zero entry that arises because we work with unitary matrices.

For $k = 2$, we repeat the procedure with the trailing 4×4 submatrix to obtain

$$H_{\geq 5}^* H_3^* H_{\geq 4}^* H_{\geq 3}^* H_1^* H_{\geq 2}^* Q H_{\geq 2} H_{\geq 3} H_2^* H_{\geq 4} H_{\geq 5} H_4^* = \begin{bmatrix} 1 & \otimes & \otimes & \otimes & \otimes & \otimes \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & \otimes & 1 & \otimes & \otimes & \otimes \\ 0 & \otimes & 0 & 1 & 0 & 0 \\ 0 & \otimes & 0 & \otimes & x & x \\ 0 & \otimes & 0 & \otimes & x & x \end{bmatrix}.$$

Finally, application of $H_{\geq 6}$, H_5, and H_6 yields

$$H_5^* H_{\geq 6}^* H_{\geq 5}^* H_3^* H_{\geq 4}^* H_{\geq 3}^* H_1^* H_{\geq 2}^* Q H_{\geq 2} H_{\geq 3} H_2^* H_{\geq 4} H_{\geq 5} H_4^* H_{\geq 6} H_6^* = I_6.$$

In general, we have

$$H_{2n-1}^* H_{\geq 2n}^* H_{\geq 2n-1}^* \dots H_{\geq 3}^* H_1^* H_{\geq 2}^* Q H_{\geq 2} H_{\geq 3} H_2^* \dots H_{\geq 2n-1} H_{2n-2}^* H_{\geq 2n} H_{2n}^* = I.$$

Using the fact that H_k^* commutes with $H_{\geq j}$ and $H_{\geq j}^*$ for any $j > k+1$, and letting

$$\Theta_0 = H_{\geq 2} H_{\geq 3} \dots H_{\geq 2n-1} H_{\geq 2n},$$

the above equation can be written as

$$(H_{2n-1}^* H_{2n-3}^* \dots H_3^* H_1^*) \Theta_0^* Q \Theta_0 (H_2^* H_4^* \dots H_{2n-2}^* H_{2n}^*) = I,$$

which gives

$$\Theta_0^* Q \Theta_0 = H_o H_e, \quad H_o = H_1 H_3 \dots H_{2n-1}, \quad H_e = H_{2n} \dots H_4 H_2 = H_2 H_4 \dots H_{2n}. \tag{3}$$

Several comments are in order:

1. Algorithm 1 is similar to the algorithm described in [4] for unitary matrices. The main difference is that the latter takes H_o as the reduction target. Our algorithm computes the matrices $H_1, \dots, H_{2n-1}$ explicitly in the elimination process. With H_{2k-1} being involved explicitly, Algorithm 1 avoids the decision-making step for determining the matrices $H_{\geq 2k+1}$, and is more straightforward to implement and potentially more reliable.

2. Algorithm 1 follows the same approach used in [17] for the initial reduction of a full CS decomposition. The procedure in [17] differs in that it applies two-sided non-similar block diagonal unitary transformations.

3. If H_k is diagonal for some k during the computations, the unitary matrix $H_o H_e$ decouples and yields two or more submatrices.

4. If Q is real orthogonal, because $\det H_k = -1$ for all $k = 1, \dots, 2n-1$, it follows that $\det Q = -\det H_{2n}(\gamma) = \gamma$. In this case, if $\gamma = -1$, then 1 and -1 must be eigenvalues of Q. These eigenvalues can be deflated; see [1, 8].

5. Algorithm 1 requires about $64n^3/3$ flops for computing H_o, H_e. If the unitary matrix Θ_0 has to be stored, additional $32n^3/3$ flops are needed. Here, one flop stands for one of the arithmetic floating-point operations $+, -, \times, \div, \sqrt{}$.

We briefly turn to the stability of Algorithm 1. Let $\hat{\Theta}_0$, $\hat{H}_k$ $(k = 1, \dots, 2n)$ be the computed matrices determined by the reduction process. It follows from standard error analysis [6, Chapter 5] that there are unitary matrices $\tilde{\Theta}_0$ and $\tilde{H}_k$ $(k = 1, \dots, 2n)$ such that

$$I + \Delta = (\tilde{H}_1 \tilde{H}_3 \dots \tilde{H}_{2n-1})^* \tilde{\Theta}_0^* (Q + E_1) \tilde{\Theta}_0 (\tilde{H}_2 \tilde{H}_4 \dots \tilde{H}_{2n-2} \tilde{H}_{2n})^*$$

with

$$\|E_1\|_2, \|\tilde{\Theta}_0 - \hat{\Theta}_0\|_2, \|\tilde{H}_1 - \hat{H}_1\|, \dots, \|\tilde{H}_{2n} - \hat{H}_{2n}\|_2 = O(\mu),$$

where $I + \Delta$ is the computed matrix obtained without enforcing the diagonal entries to be 1 and the entries with $\otimes$ in the illustration to be zero; μ denotes machine precision. From

$$\|(I + \Delta)^*(I + \Delta) - I_{2n}\|_2 = \|(Q + E_1)^*(Q + E_1) - I_{2n}\|_2 = O(\mu)$$

and the zero pattern of Δ as shown in the illustration, one can show similarly as in [17] that

$$\|\Delta\|_2 = O(\mu).$$

It is easily seen that

$$\hat{H}_1\hat{H}_3\ldots\hat{H}_{2n-1} = \tilde{H}_1\tilde{H}_3\ldots\tilde{H}_{2n-1}(I+F_o), \quad \hat{H}_2\hat{H}_4\ldots\hat{H}_{2n} = \tilde{H}_2\tilde{H}_4\ldots\tilde{H}_{2n}(I+F_e)$$

with $\|F_o\|_2, \|F_e\|_2 = O(\mu)$. Then for the computed matrices

$$\hat{H}_o := \hat{H}_1\hat{H}_3\ldots\hat{H}_{2n-1}, \qquad \hat{H}_e := \hat{H}_2\hat{H}_4\ldots\hat{H}_{2n},$$

one has

$$\hat{H}_o\hat{H}_e = \tilde{\Theta}_0^*(Q+E)\tilde{\Theta}_0, \quad \|E\|_2 = O(\mu).$$

This shows backward stability of Algorithm 1.

3 Transformation of H_oH_e to $\Sigma Z\Sigma Z^T$

From now on, we assume that $Q \in \mathbb{R}^{2n\times 2n}$ is orthogonal and that there is a real orthogonal matrix Θ_0 such that

$$\Theta_0^T Q\Theta_0 = H_oH_e,$$

where

$$\begin{aligned} H_o &= H_1H_3\ldots H_{2n-1} = \operatorname{diag}\left[\begin{bmatrix} -\gamma_1 & \sigma_1 \\ \sigma_1 & \gamma_1 \end{bmatrix}, \begin{bmatrix} -\gamma_3 & \sigma_3 \\ \sigma_3 & \gamma_3 \end{bmatrix}, \ldots, \begin{bmatrix} -\gamma_{2n-1} & \sigma_{2n-1} \\ \sigma_{2n-1} & \gamma_{2n-1} \end{bmatrix}\right], \\ H_e &= H_2H_4\ldots H_{2n} = \operatorname{diag}\left[1, \begin{bmatrix} -\gamma_2 & \sigma_2 \\ \sigma_2 & \gamma_2 \end{bmatrix}, \ldots, \begin{bmatrix} -\gamma_{2n-2} & \sigma_{2n-2} \\ \sigma_{2n-2} & \gamma_{2n-2} \end{bmatrix}, -1\right], \end{aligned}$$

and $\sigma_k > 0$ for $k = 1, \ldots, 2n-1$. Note that all $\gamma_1, \ldots, \gamma_{2n-1}$ are real.

Define for any $\gamma, \sigma \in \mathbb{R}$ with $\sigma^2 + \gamma^2 = 1$ and $\sigma > 0$, the numbers $a, b \in \mathbb{R}$ with $a, b > 0$ by

$$(a,b) = \begin{cases} \left(\frac{\sigma}{\sqrt{2(1+\gamma)}}, \frac{\sqrt{2(1+\gamma)}}{2}\right), & \gamma \geq 0, \\ \left(\frac{\sqrt{2(1-\gamma)}}{2}, \frac{\sigma}{\sqrt{2(1-\gamma)}}\right), & \gamma < 0. \end{cases}$$

Then one obtains the spectral decompositions

$$\begin{bmatrix} a & b \\ b & -a \end{bmatrix}^T \begin{bmatrix} -\gamma & \sigma \\ \sigma & \gamma \end{bmatrix} \begin{bmatrix} a & b \\ b & -a \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{4}$$

and

$$\begin{bmatrix} b & a \\ -a & b \end{bmatrix}^T \begin{bmatrix} -\gamma & \sigma \\ \sigma & \gamma \end{bmatrix} \begin{bmatrix} b & a \\ -a & b \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{5}$$

In fact, if $\sigma = \sin\theta$ and $\gamma = \cos\theta$ for some $\theta \in (0, \pi)$, then $a = \cos(\theta/2)$ and $b = \sin(\theta/2)$.

Application of (4) to each block of H_o yields an orthogonal matrix

$$\Theta_o = \operatorname{diag}\left[\begin{bmatrix} a_1 & b_1 \\ b_1 & -a_1 \end{bmatrix}, \begin{bmatrix} a_2 & b_2 \\ b_2 & -a_2 \end{bmatrix}, \ldots, \begin{bmatrix} a_n & b_n \\ b_n & -a_n \end{bmatrix}\right]$$

with $a_k, b_k > 0$ and $a_k^2 + b_k^2 = 1$ for $k = 1, \ldots, n$, such that

$$\Theta_o^T H_o\Theta_o = \operatorname{diag}[1, -1, 1, -1, \ldots, 1, -1].$$

Similarly, by applying (5) to each 2×2 block in H_e we can determine an orthogonal matrix

$$\Theta_e = \operatorname{diag}\left[1, \begin{bmatrix} f_1 & g_1 \\ -g_1 & f_1 \end{bmatrix}, \ldots, \begin{bmatrix} f_{n-1} & g_{n-1} \\ -g_{n-1} & f_{n-1} \end{bmatrix}, 1\right]$$

with $f_k, g_k > 0$ and $f_k^2 + g_k^2 = 1$ for $k = 1, \ldots, n-1$, such that

$$\Theta_e^T H_e \Theta_e = \mathrm{diag}[1, -1, 1, -1, \ldots, 1, -1].$$

Introduce

$$P = [e_1, e_3, \ldots, e_{2n-1}, e_2, e_4, \ldots, e_{2n}], \qquad \Sigma = \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix}.$$

Then

$$(\Theta_o P)^T H_o (\Theta_o P) = (\Theta_e P)^T H_e (\Theta_e P) = \Sigma.$$

Now define

$$Z := (\Theta_o P)^T (\Theta_e P) = (P^T \Theta_o P)^T (P^T \Theta_e P) =: Z_o Z_e,$$

where

$$Z_o = \left[\begin{array}{ccc|ccc} a_1 & & & b_1 & & \\ & a_2 & & & b_2 & \\ & & \ddots & & & \ddots \\ & & a_n & & & b_n \\ \hline b_1 & & & -a_1 & & \\ & b_2 & & & -a_2 & \\ & & \ddots & & & \ddots \\ & & b_n & & & -a_n \end{array}\right], \tag{6}$$

$$Z_e = \left[\begin{array}{ccc|ccc} 1 & & & 0 & & \\ & f_1 & & -g_1 & 0 & \\ & & \ddots & \ddots & \ddots & \\ & & f_{n-1} & & -g_{n-1} & 0 \\ \hline 0 & g_1 & & f_1 & & \\ & \ddots & \ddots & & \ddots & \\ & & 0 \quad g_{n-1} & & f_{n-1} & \\ & & 0 & & & 1 \end{array}\right].$$

Then it follows from

$$\Theta_0^T Q \Theta_0 = H_o H_e = \Theta_o P \Sigma (\Theta_o P)^T (\Theta_e P) \Sigma (\Theta_e P)^T$$

that

$$(\Theta_0 \Theta_o P)^T Q (\Theta_0 \Theta_o P) = \Sigma Z \Sigma Z^T. \tag{7}$$

Because Z is an orthogonal matrix, it has a CS decomposition, see, e.g., [6],

$$\begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix}^T Z \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix} = \begin{bmatrix} \Phi & \Psi \\ \Psi & -\Phi \end{bmatrix}, \tag{8}$$

where U_1, U_2, V_1, $V_2 \in \mathbb{R}^{n \times n}$ are orthogonal and

$$\Phi = \mathrm{diag}[\phi_1, \phi_2, \ldots, \phi_n], \quad \Psi = \mathrm{diag}[\psi_1, \psi_2, \ldots, \psi_n]$$

with $\phi_k^2 + \psi_k^2 = 1$ and $\phi_k, \psi_k > 0$ for $k = 1, \ldots, n$. Finally, let

$$\Theta := \Theta_0 \Theta_o P \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} P. \tag{9}$$

Then we have the real Schur decomposition

$$\Theta^T Q \Theta = \mathrm{diag}\left[\begin{bmatrix} \phi_1^2-\psi_1^2 & 2\phi_1\psi_1 \\ -2\phi_1\psi_1 & \phi_1^2-\psi_1^2 \end{bmatrix}, \ldots, \begin{bmatrix} \phi_n^2-\psi_n^2 & 2\phi_n\psi_n \\ -2\phi_n\psi_n & \phi_n^2-\psi_n^2 \end{bmatrix}\right]. \tag{10}$$

Note that for

$$\tilde{\Theta} = \Theta_0 \Theta_e P \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix} P,$$

we have another real Schur decomposition

$$\tilde{\Theta}^T Q \tilde{\Theta} = \Theta^T Q^T \Theta = \mathrm{diag}\left[\begin{bmatrix} \phi_1^2-\psi_1^2 & -2\phi_1\psi_1 \\ 2\phi_1\psi_1 & \phi_1^2-\psi_1^2 \end{bmatrix}, \ldots, \begin{bmatrix} \phi_n^2-\psi_n^2 & -2\phi_n\psi_n \\ 2\phi_n\psi_n & \phi_n^2-\psi_n^2 \end{bmatrix}\right].$$

In order to generate the orthogonal matrices Θ or $\tilde{\Theta}$, one has to compute either the matrix pairs $\{U_1, U_2\}$ or $\{V_1, V_2\}$, but not necessarily both of them.

We remark that in [1] an SVD-based method was proposed for solving the eigenvalue problem for Q. However, the approach there is different from ours. We finally note that the factorizations of this section do not apply to complex unitary matrices, because their eigenvalues might not appear in complex conjugate pairs.

4 Computation of a full CS decomposition of Z

We turn to the problem of computing a full CS decomposition of Z. The product form $Z = Z_o Z_e$ with Z_o, Z_e given by (6) yields

$$Z = \left[\begin{array}{cccc|ccc} a_1 & b_1 g_1 & & & b_1 f_1 & & \\ & a_2 f_1 & \ddots & & -a_2 g_1 & \ddots & \\ & & \ddots & b_{n-1} g_{n-1} & & \ddots & b_{n-1} f_{n-1} \\ & & & a_n f_{n-1} & & & -a_n g_{n-1} \quad b_n \\ \hline b_1 & -a_1 g_1 & & & -a_1 f_1 & & \\ & b_2 f_1 & \ddots & & -b_2 g_1 & \ddots & \\ & & \ddots & -a_{n-1} g_{n-1} & & \ddots & -a_{n-1} f_{n-1} \\ & & & b_n f_{n-1} & & & -b_n g_{n-1} \quad -a_n \end{array}\right]$$

$$= \left[\begin{array}{cccc|ccc} \alpha_{11} & \beta_{11} & & & \alpha_{13} & & \\ & \alpha_{21} & \ddots & & -\beta_{13} & \ddots & \\ & & \ddots & \beta_{n-1,1} & & \ddots & \alpha_{n-1,3} \\ & & & \alpha_{n1} & & & -\beta_{n-1,3} \quad \alpha_{n3} \\ \hline \alpha_{12} & -\beta_{12} & & & -\alpha_{14} & & \\ & \alpha_{22} & \ddots & & -\beta_{14} & \ddots & \\ & & \ddots & -\beta_{n-1,2} & & \ddots & -\alpha_{n-1,4} \\ & & & \alpha_{n2} & & & -\beta_{n-1,4} \quad -\alpha_{n4} \end{array}\right] =: \left[\begin{array}{c|c} Z_{11} & Z_{12} \\ \hline Z_{21} & Z_{22} \end{array}\right]. \tag{11}$$

The further discussion requires the following elementary orthogonal matrices:

(a) For $x = [x_1, x_2]^T \neq 0$, we define Householder matrices $H_{ij}(x) \in \mathbb{R}^{2n\times 2n}$ and Givens matrices $G_{ij}(x) \in \mathbb{R}^{n\times n}$ by

$$\begin{aligned} H_{ij}(x) &= I_{2n} + (\gamma-1)e_i e_i^T - (\gamma+1)e_j e_j^T + \sigma(e_i e_j^T + e_j e_i^T), \quad 1 \le i < j \le 2n, \\ G_{ij}(x) &= I_n + (\gamma-1)(e_i e_i^T + e_j e_j^T) + \sigma(e_j e_i^T - e_i e_j^T), \quad 1 \le i < j \le n, \end{aligned}$$

where

$$\gamma = \frac{x_1}{\sqrt{x_1^2 + x_2^2}}, \quad \sigma = \frac{x_2}{\sqrt{x_1^2 + x_2^2}}$$

satisfy

$$\begin{bmatrix} \gamma & \sigma \\ \sigma & -\gamma \end{bmatrix} x = \begin{bmatrix} \gamma & -\sigma \\ \sigma & \gamma \end{bmatrix}^T x = \begin{bmatrix} \|x\|_2 \\ 0 \end{bmatrix}.$$

Note that $\gamma, \sigma \geq 0$ when $x_1, x_2 \geq 0$. For clarity, we sometimes will write $H_{ij}(x)$ and $G_{ij}(x)$ as $H_{ij}(\gamma, \sigma)$ and $G_{ij}(\gamma, \sigma)$, respectively.

(b) For $x = [x_1, x_2]^T \neq 0$ and $1 \leq i \leq n < j \leq 2n$, we define $2n \times 2n$ Givens matrices

$$G_{ij}(x) = I_{2n} + (\gamma - 1)(e_i e_i^T + e_j e_j^T) + \sigma(e_j e_i^T - e_i e_j^T),$$

where

$$\gamma = \frac{x_2}{\sqrt{x_1^2 + x_2^2}}, \qquad \sigma = \frac{x_1}{\sqrt{x_1^2 + x_2^2}}$$

are such that

$$\begin{bmatrix} \gamma & -\sigma \\ \sigma & \gamma \end{bmatrix} x = \begin{bmatrix} 0 \\ \|x\|_2 \end{bmatrix}.$$

Again, $\gamma, \sigma \geq 0$ when $x_1, x_2 \geq 0$.

In [16], Sutton proposed a full SVD iteration process that applies the bidiagonal iteration to all four bidiagonal blocks of Z simultaneously. For one simultaneous bidiagonal iteration, it determines real orthogonal matrices $U_1, U_2, V_1, V_2 \in \mathbb{R}^{n \times n}$ such that

$$\begin{bmatrix} U_1 & \\ & U_2 \end{bmatrix}^T Z \begin{bmatrix} V_1 & \\ & V_2 \end{bmatrix} = \tilde{Z}$$

with $\tilde{Z}$ of the same form as Z. The iteration is based on the fact that for any pair of real numbers μ_1, μ_2 satisfying $\mu_1^2 + \mu_2^2 = 1$, we have

$$\begin{aligned} Z_{11}^T Z_{11} - \mu_1^2 I &= \mu_2^2 I - Z_{21}^T Z_{21}, \\ Z_{11} Z_{11}^T - \mu_1^2 I &= \mu_2^2 I - Z_{12} Z_{12}^T, \\ Z_{22}^T Z_{22} - \mu_1^2 I &= \mu_2^2 I - Z_{12}^T Z_{12}, \\ Z_{22} Z_{22}^T - \mu_1^2 I &= \mu_2^2 I - Z_{21} Z_{21}^T. \end{aligned}$$

In exact arithmetic the simultaneous bidiagonal iteration is equivalent to one bidiagonal iteration on Z_{11} (or Z_{22}) with a shift μ_1^2, or on Z_{12} (or Z_{21}) with a shift μ_2^2. Note that one bidiagonal iteration on a block, say Z_{11} with a shift μ_1^2, is equivalent to one QR iteration on $Z_{11}^T Z_{11}$ with the same shift μ_1^2 [6, Sec. 8.6]: if $\tilde{Z}_{11} = U_1^T Z_{11} V_1$ is upper bidiagonal with U_1, V_1 real orthogonal and $V_1 e_1$ parallel to $(Z_{11}^T Z_{11} - \mu_1^2 I)e_1$, then $\tilde{Z}_{11}^T \tilde{Z}_{11} = V_1^T (Z_{11}^T Z_{11}) V_1$ is symmetric tridiagonal generated from $Z_{11}^T Z_{11}$ using a similarity transformation with V_1. Here we describe a related simultaneous iteration procedure that is based on the alternative initial reduction procedure considered in [17]. The main difference is that in Algorithm 2 below one does not need to decide which row or column will be used for generating a Givens matrix in each step of the bulge-chasing process. Another difference is that Algorithm 2 computes $\tilde{Z}_o, \tilde{Z}_e$ of the same forms as Z_o, Z_e defined in (6) by reducing Z to I, and then determines $\tilde{Z} = \tilde{Z}_o \tilde{Z}_e$ after the reduction process.

Algorithm 2 [Full CS bidiagonal iteration]. *Given a $2n \times 2n$ real orthogonal matrix Z of the form (11), the algorithm computes orthogonal matrices $U_1, U_2, V_1, V_2 \in \mathbb{R}^{n \times n}$ such that*

$$\begin{bmatrix} U_1 & \\ & U_2 \end{bmatrix}^T Z \begin{bmatrix} V_1 & \\ & V_2 \end{bmatrix} = \tilde{Z}$$

by applying one QR iteration step to $Z_{11}^T Z_{11}$ with shift μ_1^2. The orthogonal matrices U_1, U_2, V_1, V_2 are determined by updating available matrices.

0. Compute $x = (Z_{11}^T Z_{11} - \mu_1^2 I)(1:2,1)$ *and* $F_{12} := G_{12}(x)$

Update $Z_{11} \leftarrow Z_{11}F_{12}, \quad Z_{21} \leftarrow Z_{21}F_{12}$

Update $V_1 \leftarrow V_1 F_{12}$

1. For $k = 1, 2, \ldots, n-1$

(a) Determine $G_{k,k+1} := G_{k,k+1}(Z_{11}(k:k+1,k))$ *and* $\tilde{G}_{k,k+1} := G_{k,k+1}(Z_{21}(k:k+1,k))$

(b) Update $Z_{11} \leftarrow G_{k,k+1}^T Z_{11}$, $Z_{12} \leftarrow G_{k,k+1}^T Z_{12}$, $U_1 \leftarrow U_1 G_{k,k+1}$,
$Z_{21} \leftarrow \tilde{G}_{k,k+1}^T Z_{21}$, $Z_{22} \leftarrow \tilde{G}_{k,k+1}^T Z_{22}$, $U_2 \leftarrow U_2 \tilde{G}_{k,k+1}$

(c) Determine $H_{k,n+k} := H_{k,n+k}([Z_{11}(k,k), Z_{21}(k,k)]^T)$

(d) Update $Z \leftarrow H_{k,n+k} Z$

(e) Determine $F_{k+1,k+2} := G_{k+1,k+2}(Z_{21}(k, k+1:k+2)^T)$ *(and* $F_n := F_{n,n+1}$
$:= \mathrm{diag}[I_{n-1}, \mathrm{sign}(Z_{21}(n-1,n))]$*) and* $\tilde{F}_{k,k+1} := G_{k,k+1}(Z_{22}(k, k:k+1)^T)$

(f) Update $Z_{11} \leftarrow Z_{11}F_{k+1,k+2}$, $Z_{21} \leftarrow Z_{21}F_{k+1,k+2}$, $V_1 \leftarrow V_1 F_{k+1,k+2}$,
$Z_{12} \leftarrow Z_{12}\tilde{F}_{k,k+1}$, $Z_{22} \leftarrow Z_{22}\tilde{F}_{k,k+1}$, $V_2 \leftarrow V_2 \tilde{F}_{k,k+1}$

(g) Determine $G_{k+1,n+k} := G_{k+1,n+k}([Z_{21}(k,k+1), Z_{22}(k,k)]^T)$

(h) Update $Z \leftarrow Z G_{k+1,n+k}^T$

End For

2. (a) Determine $G_n := \mathrm{diag}[I_{n-1}, \mathrm{sign}(Z_{11}(n,n))]$ *and* $\tilde{G}_n := \mathrm{diag}[I_{n-1}, \mathrm{sign}(Z_{21}(n,n))]$

(b) Update $Z_{11} \leftarrow G_n^T Z_{11}$, $Z_{12} \leftarrow G_n^T Z_{12}$, $U_1 \leftarrow U_1 G_n$,
$Z_{21} \leftarrow \tilde{G}_n^T Z_{21}$, $Z_{22} \leftarrow \tilde{G}_n^T Z_{22}$, $U_2 \leftarrow U_2 \tilde{G}_n$

(c) Determine $H_{n,2n} := H_{n,2n}([Z_{11}(n,n), Z_{21}(n,n)]^T)$

(d) Update $Z \leftarrow H_{n,2n} Z$

(e) Determine $\tilde{F}_n := \mathrm{diag}[I_{n-1}, \mathrm{sign}(Z_{22}(n,n))]$

(f) Update $Z_{12} \leftarrow Z_{12}\tilde{F}_n$, $Z_{22} \leftarrow Z_{22}\tilde{F}_n$, $V_2 \leftarrow V_2 \tilde{F}_n$

3. Form $\tilde{Z} = H_{1,n+1} H_{2,n+2} \ldots H_{n,2n} G_{2,n+1} G_{3,n+2} \ldots G_{n,2n-1}$

The sign function used in the algorithm is defined for real x by

$$\mathrm{sign}(x) = \begin{cases} x/|x|, & \text{if } x \neq 0, \\ 1, & \text{otherwise.} \end{cases}$$

We illustrate the computations of Algorithm 2 with a 6×6 matrix Z:

$$Z = \left[\begin{array}{ccc|ccc} x & x & 0 & x & 0 & 0 \\ 0 & x & x & x & x & 0 \\ 0 & 0 & x & 0 & x & x \\ \hline x & x & 0 & x & 0 & 0 \\ 0 & x & x & x & x & 0 \\ 0 & 0 & x & 0 & x & x \end{array}\right] \quad \overset{F_{12}}{\longrightarrow} \quad \left[\begin{array}{ccc|ccc} x & x & 0 & x & 0 & 0 \\ f & x & x & x & x & 0 \\ 0 & 0 & x & 0 & x & x \\ \hline x & x & 0 & x & 0 & 0 \\ f & x & x & x & x & 0 \\ 0 & 0 & x & 0 & x & x \end{array}\right]$$

$$\overset{G_{12}, \tilde{G}_{12}}{\longrightarrow} \quad \left[\begin{array}{ccc|ccc} + & x & f & x & f & 0 \\ 0 & x & x & x & x & 0 \\ 0 & 0 & x & 0 & x & x \\ \hline + & x & f & x & f & 0 \\ 0 & x & x & x & x & 0 \\ 0 & 0 & x & 0 & x & x \end{array}\right] \quad \overset{H_{14}}{\longrightarrow} \quad \left[\begin{array}{ccc|ccc} 1 & \otimes & \otimes & \otimes & \otimes & 0 \\ 0 & x & x & x & x & 0 \\ 0 & 0 & x & 0 & x & x \\ \hline 0 & x & x & x & x & 0 \\ 0 & x & x & x & x & 0 \\ 0 & 0 & x & 0 & x & x \end{array}\right]$$

$$
\begin{array}{c}
F_{23},\tilde{F}_{12} \\ \longrightarrow
\end{array}
\left[\begin{array}{ccc|ccc}
1 & \otimes & \otimes & \otimes & \otimes & 0 \\
0 & x & x & x & x & 0 \\
0 & f & x & f & x & x \\
\hline
0 & + & 0 & + & 0 & 0 \\
0 & x & x & x & x & 0 \\
0 & f & x & f & x & x
\end{array}\right]
\begin{array}{c}
G_{24}^T \\ \longrightarrow
\end{array}
\left[\begin{array}{ccc|ccc}
1 & \otimes & \otimes & \otimes & \otimes & 0 \\
0 & x & x & \otimes & x & 0 \\
0 & x & x & \otimes & x & x \\
\hline
0 & 0 & 0 & 1 & 0 & 0 \\
0 & x & x & \otimes & x & 0 \\
0 & x & x & \otimes & x & x
\end{array}\right]
$$

$$
\begin{array}{c}
G_{23},\tilde{G}_{23} \\ \longrightarrow
\end{array}
\left[\begin{array}{ccc|ccc}
1 & \otimes & \otimes & \otimes & \otimes & 0 \\
0 & + & x & \otimes & x & f \\
0 & 0 & x & \otimes & x & x \\
\hline
0 & 0 & 0 & 1 & 0 & 0 \\
0 & + & x & \otimes & x & f \\
0 & 0 & x & \otimes & x & x
\end{array}\right]
\begin{array}{c}
H_{25} \\ \longrightarrow
\end{array}
\left[\begin{array}{ccc|ccc}
1 & \otimes & \otimes & \otimes & \otimes & 0 \\
0 & 1 & \otimes & \otimes & \otimes & \otimes \\
0 & 0 & x & \otimes & x & x \\
\hline
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & x & \otimes & x & x \\
0 & 0 & x & \otimes & x & x
\end{array}\right]
$$

$$
\begin{array}{c}
F_3,\tilde{F}_{23} \\ \longrightarrow
\end{array}
\left[\begin{array}{ccc|ccc}
1 & \otimes & \otimes & \otimes & \otimes & \otimes \\
0 & 1 & \otimes & \otimes & \otimes & \otimes \\
0 & 0 & x & \otimes & x & x \\
\hline
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & + & \otimes & + & 0 \\
0 & 0 & x & \otimes & x & x
\end{array}\right]
\begin{array}{c}
G_{35}^T \\ \longrightarrow
\end{array}
\left[\begin{array}{ccc|ccc}
1 & \otimes & \otimes & \otimes & \otimes & \otimes \\
0 & 1 & \otimes & \otimes & \otimes & \otimes \\
0 & 0 & x & \otimes & \otimes & x \\
\hline
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & \otimes & 1 & 0 \\
0 & 0 & x & \otimes & \otimes & x
\end{array}\right]
$$

$$
\begin{array}{c}
G_3,\tilde{G}_3 \\ \longrightarrow
\end{array}
\left[\begin{array}{ccc|ccc}
1 & \otimes & \otimes & \otimes & \otimes & \otimes \\
0 & 1 & \otimes & \otimes & \otimes & \otimes \\
0 & 0 & + & \otimes & \otimes & x \\
\hline
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & \otimes & 1 & 0 \\
0 & 0 & + & \otimes & \otimes & x
\end{array}\right]
\begin{array}{c}
H_{36} \\ \longrightarrow
\end{array}
\left[\begin{array}{ccc|ccc}
1 & \otimes & \otimes & \otimes & \otimes & \otimes \\
0 & 1 & \otimes & \otimes & \otimes & \otimes \\
0 & 0 & 1 & \otimes & \otimes & \otimes \\
\hline
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & \otimes & 1 & 0 \\
0 & 0 & 0 & \otimes & \otimes & x
\end{array}\right]
$$

$$
\begin{array}{c}
\tilde{F}_3 \\ \longrightarrow
\end{array}
\left[\begin{array}{ccc|ccc}
1 & \otimes & \otimes & \otimes & \otimes & \otimes \\
0 & 1 & \otimes & \otimes & \otimes & \otimes \\
0 & 0 & 1 & \otimes & \otimes & \otimes \\
\hline
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & \otimes & 1 & 0 \\
0 & 0 & 0 & \otimes & \otimes & 1
\end{array}\right] = I_6,
$$

where "0" denotes zero, "+" stands for a nonnegative entry, "1" is one, "$\otimes$" is a zero obtained because Z is orthogonal, and elements marked by "f" are obtained by fill-in. They are generically nonvanishing.

Because $H_{k,n+k}$ commutes with the block diagonal matrix $\mathrm{diag}[G_{j,j+1}, \tilde{G}_{j,j+1}]$ and $G_{k+1,n+k}^T$ commutes with $\mathrm{diag}[F_{j+1,j+2}, \tilde{F}_{j,j+1}]$ for any $j > k$, we obtain

$$
H_{36}H_{25}H_{14}\begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix}^T Z \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix} G_{24}^T G_{35}^T = I_6.
$$

In general, one has

$$
H_{n,2n}\dots H_{2,n+2}H_{1,n+1}\begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix}^T Z \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix} G_{2,n+1}^T \dots G_{n,2n-1}^T = I_{2n}.
$$

Therefore

$$
\begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix}^T Z \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix} = \tilde{Z}_o\tilde{Z}_e = \tilde{Z},
$$

where

$$
\tilde{Z}_o = H_{1,n+1}H_{2,n+2}\dots H_{n,2n}, \quad \tilde{Z}_e = G_{2,n+1}G_{3,n+2}\dots G_{n,2n-1}.
$$

Algorithm 2 requires about $130n$ flops. If the matrices U_1, U_2, V_1, V_2 have to be updated, then additional $24n^2$ flops are needed during the whole iteration process. One has to determine a shift μ_1^2 for the rotation F_{12}. For instance, one may choose the Wilkinson shift, i.e., an eigenvalue of the trailing 2×2 submatrix of $Z_{11}Z_{11}^T$ that is closest to the last diagonal entry of $Z_{11}Z_{11}^T$; see [16] for details on the choice of shift. One can show that Algorithm 2 is backward stable similarly as we did for Algorithm 1. Alternatively, one can proceed in the same manner as in [17].

For the eigenvalue problem of an orthogonal matrix, any backward stable algorithm is also forward stable. Hence, the eigenvalues can be computed accurately with a backward stable method. This, however, does not guarantee that small real and imaginary parts of the eigenvalues will be computed accurately. In our case, this is equivalent to the fact that small singular values of the blocks of Z might not be computed accurately. We therefore describe another full CS decomposition iteration based on the Demmel–Kahan zero-shift SVD iteration [5] to achieve high accuracy of small singular values. The iteration procedure from Z to $\tilde{Z}$ is the standard bulge-chasing process. Since $\mu_1 = 0$, the iteration formulas can be explicitly derived.

Theorem 1 *Let $\tilde{Z}$ be the matrix generated from Z in (11) with the zero-shift full CS iteration. Then*

$$\begin{aligned} &\tilde{\alpha}_{k1} = r_k, && \tilde{\alpha}_{k2} = \frac{\alpha_{k3}s_k}{p_k}, && \tilde{\alpha}_{k3} = \frac{\tilde{\alpha}_{k2}q_k}{r_k}, && \tilde{\alpha}_{k4} = q_k, \\ &\tilde{\beta}_{k1} = s_{k,1}p_{k+1}, && \tilde{\beta}_{k2} = \frac{\beta_{k3}p_{k+1}}{s_k}, && \tilde{\beta}_{k3} = \frac{\tilde{\beta}_{k2}r_{k+1}}{q_k}, && \tilde{\beta}_{k4} = z_{k,2}s_{k+1}, \end{aligned} \tag{12}$$

where, for $k = 1, 2, \ldots, n$,

$$\begin{aligned} p_k^2 &= (t_{k-1,1}\alpha_{k1})^2 + \beta_{k1}^2, & t_{k,1} &= \frac{t_{k-1,1}\alpha_{k1}}{p_k}, & z_{k,1} &= \frac{\beta_{k1}}{p_k}, \\ r_k^2 &= (c_{k-1,1}p_k)^2 + (\alpha_{k+1,1}z_{k,1})^2, & c_{k,1} &= \frac{c_{k-1,1}p_k}{r_k}, & s_{k,1} &= \frac{\alpha_{k+1,1}z_{k,1}}{r_k}, \\ s_k^2 &= (c_{k-1,2}\alpha_{k4})^2 + \beta_{k4}^2, & c_{k,2} &= \frac{c_{k-1,2}\alpha_{k4}}{s_k}, & s_{k,2} &= \frac{\beta_{k4}}{s_k}, \\ q_k^2 &= (t_{k-1,2}s_k)^2 + (s_{k,2}\alpha_{k+1,4})^2, & t_{k,2} &= \frac{t_{k-1,2}s_k}{q_k}, & z_{k,2} &= \frac{s_{k,2}\alpha_{k+1,4}}{q_k}, \end{aligned} \tag{13}$$

with $t_{0,1} = c_{0,1} = c_{0,2} = t_{0,2} = 1$ and $\beta_{nj} = 0$ for $j = 1, 2, 3, 4$.

Proof. A proof is provided in Appendix A. □

We note that the block Z_{21} is not involved in the iteration. The four sets of parameters in (13) stem from four Givens matrices, each of which can be computed with the following algorithm [6].

Algorithm 3 [Givens matrix] *Given a real vector $x = [x_1, x_2]^T \neq 0$, the algorithm computes γ, σ and $r = \|x\|_2$ such that* $\begin{bmatrix} \gamma & \sigma \\ -\sigma & \gamma \end{bmatrix} x = r \begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

If $|x_1| > |x_2|$

Compute $t = x_2/x_1$, $s = \sqrt{1+t^2}$

Compute $\gamma = 1/s$, $\sigma = t/s$, $r = |x_1|s$

else

Compute $t = x_1/x_2$, $s = \sqrt{1+t^2}$

Compute $\gamma = t/s$, $\sigma = 1/s$, $r = |x_2|s$

end

Theorem 1 suggests the following iterative method.

Algorithm 4 [Zero-shift CS bidiagonal iteration]. *Given a $2n \times 2n$ real orthogonal matrix Z of the form (11), the algorithm computes orthogonal matrices $U_1, U_2, V_1, V_2 \in \mathbb{R}^{n\times n}$ such that*

$$\begin{bmatrix} U_1 & \\ & U_2 \end{bmatrix}^T Z \begin{bmatrix} V_1 & \\ & V_2 \end{bmatrix} = \tilde{Z}$$

based on one QR iteration step applied to $Z_{11}^T Z_{11}$ with zero shift. The orthogonal matrices U_1, U_2, V_1, V_2 are determined by updating the available matrices.

0. Set $t_{0,1} = t_{0,2} = c_{0,1} = c_{0,2} = 1$ and $\beta_{nj} = 0$ for $j = 1, 2, 3, 4$.

1. For $k = 1, \ldots, n-1$

(a) Compute p_k and $G_{k,k+1}(t_{k,1}, z_{k,1})$ with $x = [t_{k-1,1}\alpha_{k1}, \beta_{k1}]^T$

(b) Compute r_k and $G_{k,k+1}(c_{k,1}, s_{k,1})$ with $x = [c_{k-1,1}p_k, \alpha_{k+1,1}z_{k,1}]^T$

(c) Compute s_k and $G_{k,k+1}(c_{k,2}, s_{k,2})$ with $x = [c_{k-1,2}\alpha_{k4}, \beta_{k4}]^T$

(d) Compute q_k and $G_{k,k+1}(t_{k,2}, z_{k,2})$ with $x = [t_{k-1,2}s_k, \alpha_{k+1,4}s_{k,2}]^T$

(e) If $k > 1$

$$\text{Compute } \tilde{\beta}_{k-1,1} = s_{k-1,1}p_k,\ \tilde{\beta}_{k-1,2} = \beta_{k-1,3}p_k/s_{k-1},$$
$$\tilde{\beta}_{k-1,3} = \tilde{\beta}_{k-1,2}r_k/q_{k-1},\ \tilde{\beta}_{k-1,4} = z_{k-1,2}s_k$$

End if

(f) Compute $\tilde{\alpha}_{k1} = r_k$, $\tilde{\alpha}_{k2} = \alpha_{k3}s_k/p_k$, $\tilde{\alpha}_{k3} = \tilde{\alpha}_{k2}q_k/r_k$, $\tilde{\alpha}_{k4} = q_k$

(g) Update $U_1 \leftarrow U_1 G_{k,k+1}(c_{k,1}, s_{k,1}), U_2 \leftarrow U_2 G_{k,k+1}(c_{k,2}, s_{k,2})$ and $V_1 \leftarrow V_1 G_{k,k+1}(t_{k,1}, z_{k,1})$, $V_2 \leftarrow V_2 G_{k,k+1}(t_{k,2}, z_{k,2})$

2. (a) Compute $p_n = \alpha_{n1}t_{n-1,1}$, $r_n = c_{n-1,1}p_n$, $s_n = \alpha_{n4}c_{n-1,2}$, $q_n = t_{n-1,2}s_n$

(b) Compute $\tilde{\beta}_{n-1,1} = s_{n-1,1}p_n$, $\tilde{\beta}_{n-1,2} = \beta_{n-1,3}p_n/s_{n-1}$,
$$\tilde{\beta}_{n-1,3} = \tilde{\beta}_{n-1,2}r_n/q_{n-1},\ \tilde{\beta}_{n-1,4} = z_{n-1,2}s_n$$

(c) Compute $\tilde{\alpha}_{n1} = r_n$, $\tilde{\alpha}_{n2} = \alpha_{n3}s_n/p_n$, $\tilde{\alpha}_{n3} = \tilde{\alpha}_{n2}q_n/r_n$, $\tilde{\alpha}_{n4} = q_n$

This algorithm requires $44(n-1)+4$ flops. Additional $24n(n-1)$ flops are needed to update U_1, U_2, V_1, V_2.

The factored form $\tilde{Z}_o\tilde{Z}_e$ is easily derived from $\tilde{Z}$ if it is required. One may simply use the entries of $\tilde{Z}_{11}$ and $\tilde{Z}_{21}$ and apply the following code:

0. $\tilde{a}_1 = \tilde{\alpha}_{11}, \quad \tilde{b}_1 = \tilde{\alpha}_{12}$

1. For $k = 1, \ldots, n-1$

(a) Determine $\tilde{f}_k, \tilde{a}_{k+1}, \tilde{b}_{k+1}$ by applying Algorithm 3 to $x := [\tilde{\alpha}_{k+1,1}, \tilde{\alpha}_{k+1,2}]^T$

(b) Determine $\tilde{g}_k$ by applying Algorithm 3 to $x := [\tilde{\beta}_{k1}, \tilde{\beta}_{k2}]^T$

End for

This code demands $12n-12$ flops. We remark that it follows from (11) that one may also use the formulas $\tilde{g}_k = \tilde{\beta}_{k1}/\tilde{b}_k$ or $\tilde{g}_k = \tilde{\beta}_{k2}/\tilde{a}_k$ to compute $\tilde{g}_k$. The reason for using the formula in the code is to enforce the relation $\tilde{f}_k^2 + \tilde{g}_k^2 = 1$ explicitly.

The following first order error analysis, which is based on the results in [5], shows that highly accurate singular values can be computed with Algorithm 4. The floating-point arithmetic is assumed to satisfy

$$fl(\alpha \circ \beta) = (\alpha \circ \beta)(1+\delta_1) = (\alpha \circ \beta)/(1+\delta_2), \qquad |\delta_1|, |\delta_2| \le \mu,$$

where $\circ \in \{+, -, \times, \div, \sqrt{}\}$ and μ is the machine precision.

Theorem 2 *Suppose that Algorithm 4 is applied to the data* $\{\alpha_{kj}\}$ *and* $\{\beta_{kj}\}$ *to compute* $\{\tilde{\alpha}_{kj}\}$ *and* $\{\tilde{\beta}_{kj}\}$ *on a computer with machine precision* μ*. Let* $\{\hat{\alpha}_{kj}\}$ *and* $\{\hat{\beta}_{kj}\}$ *be the computed data. Then*

$$\hat{\alpha}_{kj} = \tilde{\alpha}_{kj}(1+\epsilon_{\alpha_{kj}}), \quad \hat{\beta}_{kj} = \tilde{\beta}_{kj}(1+\epsilon_{\beta_{kj}})$$

and

$$\begin{aligned}
&|\epsilon_{\alpha_{k1}}| \le \frac{69k-48}{2}\mu,\ |\epsilon_{\alpha_{k2}}| \le \frac{25k-8}{2}\mu,\ |\epsilon_{\alpha_{k3}}| \le \frac{163k-75}{2}\mu,\ |\epsilon_{\alpha_{k4}}| \le \frac{69k-48}{2}\mu,\\
&|\epsilon_{\beta_{k1}}| \le \frac{50k+13}{4}\mu,\ |\epsilon_{\beta_{k2}}| \le \frac{50k+9}{4}\mu,\ |\epsilon_{\beta_{k3}}| \le \frac{326k-75}{4}\mu,\ |\epsilon_{\beta_{k4}}| \le \frac{188k-83}{4}\mu. \quad (14)
\end{aligned}$$

Furthermore, if during the iterations $s_{k,1}^2, s_{k,2}^2, z_{k,1}^2, z_{k,2}^2 \le \tau < 1$ *for all* k*, then*

$$\begin{aligned}
|\epsilon_{\alpha_{k1}}| &\le \frac{88-38\tau}{4(1-\tau)^2}\mu, & |\epsilon_{\beta_{k1}}| &\le \frac{44-19\tau}{2(1-\tau)}\mu,\\
|\epsilon_{\alpha_{k2}}| &\le \frac{42-17\tau}{2(1-\tau)}\mu, & |\epsilon_{\beta_{k2}}| &\le \frac{42-17\tau}{2(1-\tau)}\mu,\\
|\epsilon_{\alpha_{k3}}| &\le \frac{134-105\tau+21\tau^2}{2(1-\tau)^2}\mu, & |\epsilon_{\beta_{k3}}| &\le \frac{134-105\tau+21\tau^2}{2(1-\tau)^2}\mu,\\
|\epsilon_{\alpha_{k4}}| &\le \frac{88-38\tau}{4(1-\tau)^2}\mu, & |\epsilon_{\beta_{k4}}| &\le \frac{65-36\tau-4\tau^2}{2(1-\tau)^2}\mu.
\end{aligned} \quad (15)$$

Proof. A proof is given in Appendix B. □

It is pointed out in [5] that typically the β_{kj}'s decease during the iterations. Therefore, the bounds in (15) will be more realistic in practice.

We may use Algorithms 3 and 4 together to compute a full CS decomposition of Z, where we apply the latter algorithm to compute small singular values and use the former algorithm to compute the other ones. Slight modifications of the deflation and stopping criteria proposed in [5] can be used. For an $n \times n$ bidiagonal matrix B with diagonal entries $\{\alpha_j\}_{j=1}^n$ and super(sub)-diagonal entries $\{\beta_j\}_{j=1}^{n-1}$, we apply the following algorithms, cf. [5]:

Algorithm A

$\lambda_n = |\alpha_n|$

For $j = n-1$ *to* 1

Compute $\lambda_j = |\alpha_j|/(1+|\beta_j|/\lambda_{j+1})$

End

Algorithm B

$\nu_1 = |\alpha_1|$

For $j = 1$ *to* $n-1$

Compute $\nu_{j+1} = |\alpha_{j+1}|/(1+|\beta_j|/\nu_j)$

End

Then $\|B^{-1}\|_\infty^{-1} = \min_j \lambda_j$ and $\|B^{-1}\|_1^{-1} = \min_j \nu_j$.

For each of the blocks of Z, we can apply the above algorithms to compute a lower bound for the smallest singular value:

$$\begin{aligned}
\underline{\sigma}_{11} &= \min\{\|Z_{11}^{-1}\|_1^{-1}, \|Z_{11}^{-1}\|_\infty^{-1}\}, \quad \underline{\sigma}_{21} = \min\{\|Z_{21}^{-1}\|_1^{-1}, \|Z_{21}^{-1}\|_\infty^{-1}\},\\
\underline{\sigma}_{12} &= \min\{\|Z_{12}^{-1}\|_1^{-1}, \|Z_{12}^{-1}\|_\infty^{-1}\}, \quad \underline{\sigma}_{22} = \min\{\|Z_{22}^{-1}\|_1^{-1}, \|Z_{22}^{-1}\|_\infty^{-1}\}.
\end{aligned}$$

Let *tol* be a selected tolerance. We use the following criterion to switch from zero-shift iteration to shifted iteration:

If $n \min\{\underline{\sigma}_{11}, \underline{\sigma}_{22}\} tol \le \mu$

run zero-shift iteration with Z

elseif $n \min\{\underline{\sigma}_{21}, \underline{\sigma}_{12}\} tol \le \mu$

run zero-shift iteration with a permuted Z

else

run shifted iteration

end

The permuted matrix Z is given by

$$\begin{bmatrix} \hat{P} & 0 \\ 0 & -\hat{P} \end{bmatrix}^T Z \begin{bmatrix} 0 & \hat{P} \\ \hat{P} & 0 \end{bmatrix} = \left[\begin{array}{cccc|cccc} \alpha_{n3} & \beta_{n-1,3} & & & \alpha_{n1} & & & \\ & \alpha_{n-1,3} & \ddots & & -\beta_{n-1,1} & \ddots & & \\ & & \ddots & \beta_{13} & & \ddots & \alpha_{21} & \\ & & & \alpha_{13} & & & -\beta_{11} & \alpha_{11} \\ \hline \alpha_{n4} & -\beta_{n-1,4} & & & -\alpha_{n2} & & & \\ & \alpha_{n-1,4} & \ddots & & -\beta_{n-1,2} & \ddots & & \\ & & \ddots & -\beta_{14} & & \ddots & -\alpha_{22} & \\ & & & \alpha_{14} & & & -\beta_{12} & -\alpha_{12} \end{array}\right],$$

where $\hat{P} = [e_n, -e_{n-1}, \ldots, (-1)^{n-1} e_1]$. This matrix has the same pattern as Z. In the situation that $n \min\{\underline{\sigma}_{11}, \underline{\sigma}_{22}\} tol > \mu$ and $n \min\{\underline{\sigma}_{21}, \underline{\sigma}_{12}\} tol \le \mu$, the blocks Z_{11}, Z_{22} have no small singular values, while Z_{12}, Z_{21} do. In this case we apply the zero-shift iteration to the permuted Z to compute these small singular values. The orthogonal matrices have to be changed accordingly:

$$U_1 \leftarrow U_1 \hat{P}, \quad U_2 \leftarrow -U_2 \hat{P}, \quad V_1 \leftarrow V_2 \hat{P}, \quad V_2 \leftarrow V_1 \hat{P}.$$

For pre-iteration deflation, we use the following deflation criterion when applying Algorithms A and B to blocks of Z:

if for some j, $|\beta_j/\nu_j|$ or $|\beta_j/\lambda_{j+1}| \le tol$ for all four blocks, set $\beta_{jk} = 0$ for $k = 1, 2, 3, 4$.

For the shifted iteration we use the deflation criterion:

$$\text{if } \max_{1 \le k \le 4} \beta_{n-1,k}/\alpha_{n,k} \le tol, \quad \text{set } \beta_{n-1,k} = 0, \quad \text{for } k = 1, 2, 3, 4. \tag{16}$$

Algorithm 5 [Overall CS decomposition] *Given a $2n \times 2n$ matrix Z defined in (11), the algorithm computes a full CS decomposition (8).*

Initial Step. *Choose tol and M, where M is the maximum number of full CS iterations.*
Set $U_1 = U_2 = V_1 = V_2 = I_n$, or the already existing ones.

Iteration Step.
(a) Apply Algorithms A and B to each of the four blocks of Z
If for some j, $|\beta_j|/\nu_j < tol$ or $\beta_j/\lambda_{j+1} < tol$ for all four blocks
Set $\beta_{j1} = \beta_{j2} = \beta_{j3} = \beta_{j4} = 0$
Decouple Z to form submatrices of the same form as Z
end if
(b) For each submatrix (still denoted by Z) of size $2p \times 2p$, goto (c)
(c) Apply Algorithms A and B to compute $\underline{\sigma}_{11}, \underline{\sigma}_{12}, \underline{\sigma}_{21}, \underline{\sigma}_{22}$
If $p \max\{\underline{\sigma}_{11}, \underline{\sigma}_{22}\} tol < \mu$

(d) *While* $\max\left\{\frac{\beta_{p-1,1}}{\underline{\sigma}_{11}}, \frac{\beta_{p-1,2}}{\underline{\sigma}_{21}}, \frac{\beta_{p-1,3}}{\underline{\sigma}_{12}}, \frac{\beta_{p-1,4}}{\underline{\sigma}_{22}}\right\} \geq tol$ *and # of iterations* $\leq M$
Apply Algorithm 4 to Z
End while
If # of iterations is larger than M*, then report divergence*
otherwise, set $\beta_{p-1,k} = 0$ *for* $k = 1, 2, 3, 4$
and repeat (c) with the reduced matrix Z
Elseif $p \max\{\underline{\sigma}_{12}, \underline{\sigma}_{21}\} tol < \mu$
Repeat (d) with the permuted Z
Else
(e) *While* $\max_{1 \leq k \leq 4} \beta_{p-1,k}/\alpha_{pk} \geq tol$ *and # of iterations* $\leq M$
Apply Algorithm 3 to Z
End while
If # of iterations is larger than M*, then report divergence*
otherwise, set $\beta_{p-1,k} = 0$ *for* $k = 1, 2, 3, 4$
and repeat (e) with the reduced matrix Z
End if

We let $tol = n\mu$ and $M = 3n^2/4$ in the algorithm.

Underflow may occur during the iteration process. Vanishing elements $\underline{\sigma}_{11}$, $\underline{\sigma}_{12}$, $\underline{\sigma}_{21}$, or $\underline{\sigma}_{22}$, is a good indication. Since Z is orthogonal, we may scale up the matrix if underflow takes place. The graded structure of Z may cause loss of accuracy. This difficulty can be reduced by chasing the bulge in suitable direction (up or down) as described in [5].

5 The Schur form and numerical examples

The computations for determining the Schur form of a real orthogonal matrix Q are summarized by the following algorithm:

Algorithm 6 [Orthogonal Schur form] *Given a* $2n \times 2n$ *real orthogonal matrix* Q *without real eigenvalues, the algorithm computes the real Schur form (10).*

Step 1. *Apply Algorithm 1 to* Q *to compute the factorization (3)*

Step 2. *Compute the factorization (7)*

Step 3. *Apply Algorithm 5 to* Z *to compute the full CS decomposition (8) (without computing* V_1, V_2*)*

Step 4 *Form the matrix* Θ *using (9) and determine (10)*

With proper choices of the tolerances, the algorithm is backward stable. The main cost is the initial reduction step when fewer than $O(n^2)$ zero-shift CS decomposition iterations are carried out. This is the typical situation. We tested the eigenvalue algorithm with several examples. The main purpose of the numerical experiments is to illustrate the accuracy achieved with the algorithm. We remark that when computing the eigenvalues of a general orthogonal matrix, zero-shift full CS iteration typically will not improve the accuracy. This is because the matrix Z is generated after the initial reduction process, and rounding errors from this process pollute Z. All the experiments were carried out with MATLAB version 7.10.0 on an iMac 8.1 computer with an Intel Core 2 Duo 2.4 GHz processor.

Example 1. We generated 40 real orthogonal matrices Q of size 30×30 using the MATLAB command `[Q,R]=qr(randn(30))`. For each Q we computed the eigenvalues by using Algorithm 6 in two ways. Method I is simply Algorithm 6. Method II is essentially the same as Algorithm

6, but in Step 3, we use the shifted CS iterations only. Also, for Method II the deflation criterion (16) is changed to

$$\max_{1\le k\le 4} 2\beta_{n-1,k}/(\alpha_{n-1,k}+\alpha_{nk}) \le tol.$$

For each method, we report the maximum and minimum eigenvalue errors for each matrix. The results are shown in Figure 1, where the matrices (labeled in horizontal direction) are sorted according to the magnitude of the maximum errors from Method I. The "exact" eigenvalues are computed by using `eig` from the MATLAB Symbolic Toolbox. For comparison we also display the extreme errors of the eigenvalues computed by the standard MATLAB function `eig`. The broken lines in Figure 1 indicate that the minimum errors are numerically zero.

Figure 1: Maximum and minimum eigenvalue errors for 40 random orthogonal matrices of Example 1.

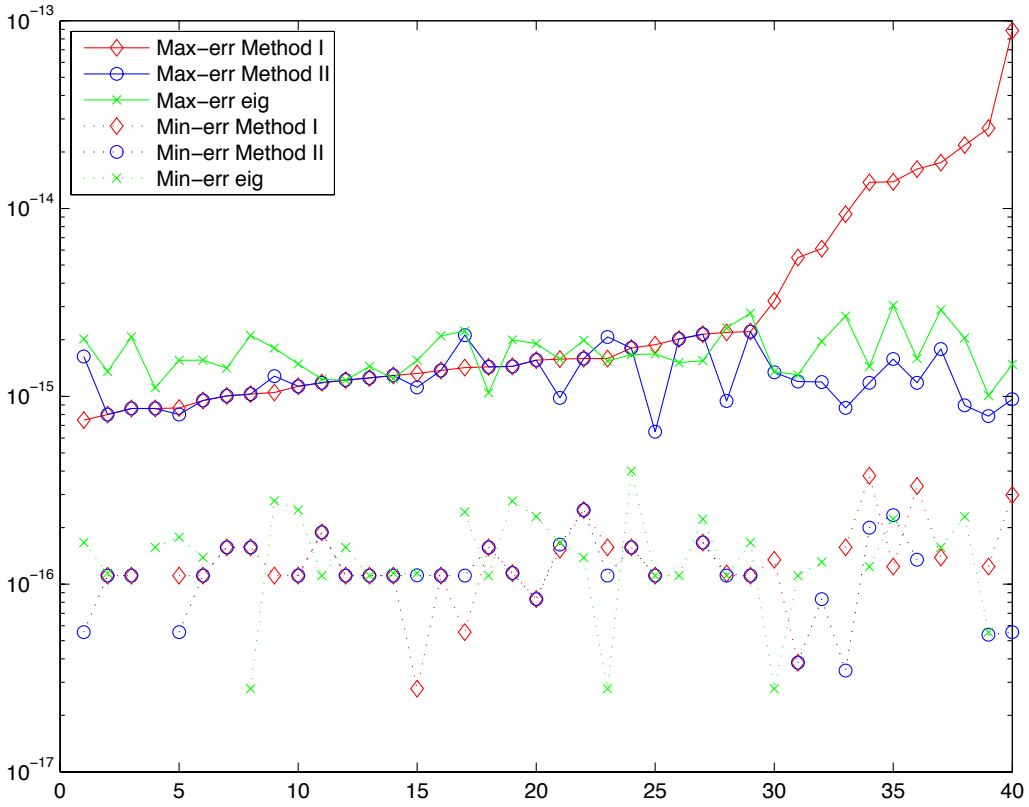


Figure 1 shows that Method II computes the eigenvalues at least as accurately as `eig`, while Method I may give less accurate eigenvalues. A closer look reveals that the less accurately computed eigenvalues are the ones with small imaginary part, and the large errors are caused by the zero-shift CS iterations. This is because slow convergence demands many more iterations, and this increases the error in the real part, while accuracy of the imaginary part cannot be improved.

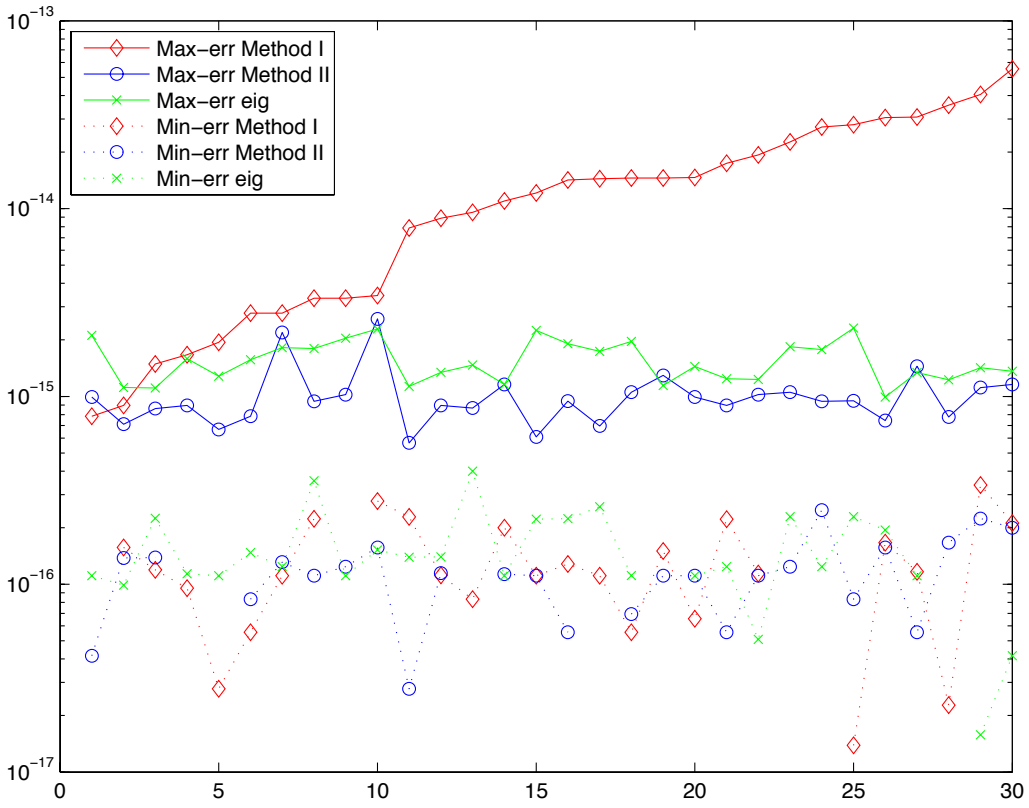
Example 2. In this example, we tested thirty 20×20 orthogonal matrices of the form

$$Q = UDU^T,$$

where D is quasi-diagonal containing 10 complex conjugate eigenvalue pairs, among which are two pairs $-\sqrt{1-10^{-8}} \pm i10^{-4}$ and $\sqrt{1-10^{-14}} \pm i10^{-7}$. The other pairs are of the form $\pm\sqrt{1-d_i^2} \pm id_i$, where d_i is a positive random number generated with the MATLAB function `rand`, and the sign for the real part also is randomly generated. The 30 orthogonal matrices are determined by the (fixed) matrix D and 30 randomly generated orthogonal matrices U. For each matrix Q the eigenvalues

are computed by Methods I and II, as well as by `eig`, similarly as in Example 1. For each Q the zero-shift CS iterations for both the regular and permuted versions are used with Method I. The eigenvalue errors are reported in Figure 2. We observe that similarly as in Example 1, the zero-shift CS iteration yields the largest errors.

Figure 2: Maximum and minimum eigenvalue errors for orthogonal matrices of Example 2.



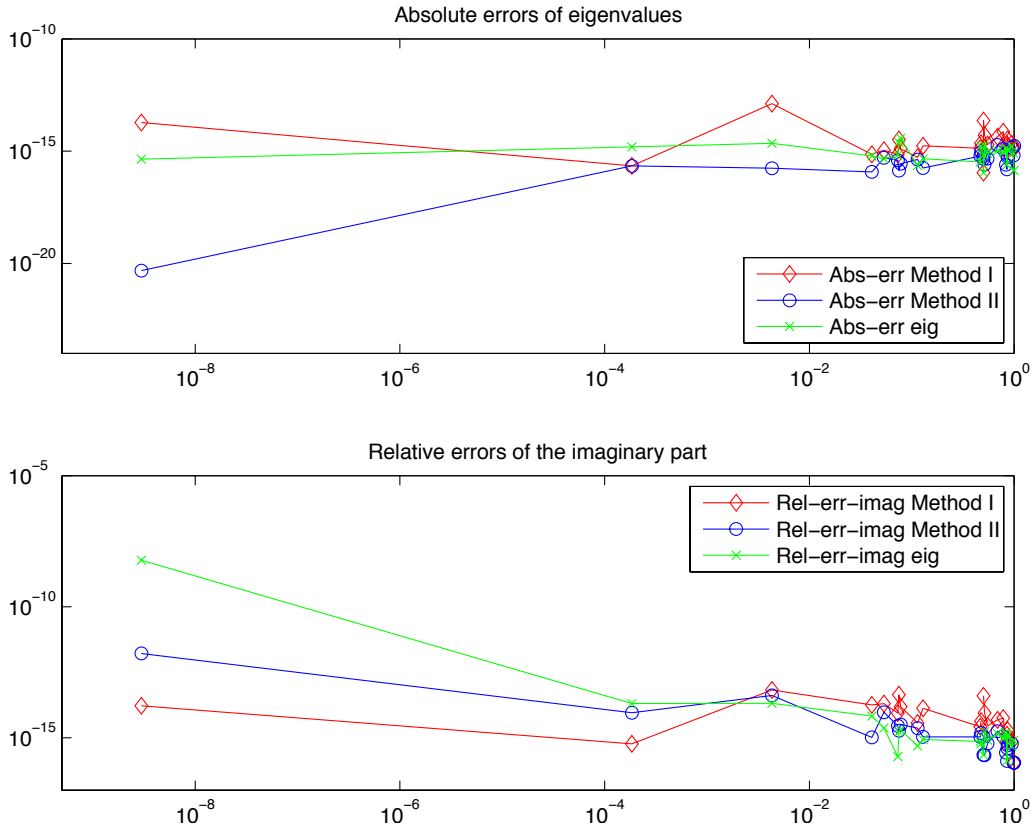
Example 3. In this example we tested the zero-shift CS decomposition iterations to see if the small imaginary parts of orthogonal eigenvalues can be computed accurately. To this end, we constructed orthogonal matrices of the form H_oH_e, and for each H_k defined in (1), we let $\gamma_k = \cos\theta_k$ and $\sigma_k = \sin\theta_k$ with a random $\theta_k \in (0, \pi)$. We tested ten such randomly generated orthogonal matrices of size 50×50 and observed that the relative errors of the small imaginary parts of the eigenvalues computed with Method I are smaller than for eigenvalues computed with the other methods, although the corresponding eigenvalue errors are slightly larger. The results of a typical orthogonal matrix are shown in Figure 3, where the errors are arranged according to the magnitude of the imaginary parts (in horizontal direction).

6 Conclusions and open problems

We have shown that the eigenvalue problem for a real orthogonal matrix can be formulated as a full CS decomposition problem with a simple transformation, and based on this fact we have developed a backward stable eigenvalue method. It is interesting that for real orthogonal matrices, the eigenvalue problem and the CS decomposition merge in such a way.

There are still many open problems. For instance, H_oH_e is orthogonally similar to the matrix $\Sigma Z \Sigma Z^T$, which is actually a 2×2 block matrix with all four tridiagonal blocks. Is it possible to develop an algorithm that uses this structure directly? Also, there are several implementation options that can be explored. For instance, in the CS decomposition iterations one could keep Z

Figure 3: Eigenvalue errors for a typical orthogonal matrix in Example 3.



in product form $Z_o Z_e$, which might improve speed and accuracy somewhat because the product form is determined by only $4n-2$ parameters while Z requires $8n-4$ parameters.

Appendix A. Proof of Theorem 1

We would like to use the form of Z in (11) with $\{a_k\}, \{b_k\}, \{f_k\}, \{g_k\}$. Note that (13) is equivalent to

$$\begin{aligned}
p_k^2 &= \left(\frac{a_{1\to k} f_{1\to k-1}}{p_{1\to k-1}}\right)^2 + (b_k g_k)^2, & c_{k,1} &= \frac{a_{1\to k} f_{1\to k-1}}{p_{1\to k}}, & \frac{b_k g_k}{p_k},\\
r_k^2 &= \left(\frac{p_{1\to k}}{r_{1\to k-1}}\right)^2 + \left(\frac{a_{k+1} b_k f_k g_k}{p_k}\right)^2, & c_{k,1} &= \frac{p_{1\to k}}{r_{1\to k}}, & s_{k,1} &= \frac{a_{k+1} b_k f_k g_k}{p_k r_k},\\
s_k^2 &= \left(\frac{a_{1\to k} f_{1\to k}}{s_{1\to k-1}}\right)^2 + (b_{k+1} g_k)^2, & c_{k,2} &= \frac{a_{1\to k} f_{1\to k}}{s_{1\to k}}, & s_{k,2} &= \frac{b_{k+1} g_k}{s_k},\\
q_k^2 &= \left(\frac{s_{1\to k}}{q_{1\to k-1}}\right)^2 + \left(\frac{a_{k+1} b_{k+1} f_{k+1} g_k}{s_k}\right)^2, & t_{k,2} &= \frac{s_{1\to k}}{q_{1\to k}}, & s_{k,2} &= \frac{a_{k+1} b_{k+1} f_{k+1} g_k}{s_k q_k}
\end{aligned} \tag{17}$$

with $f_{1\to 0} = p_{1\to 0} = r_{1\to 0} = s_{1\to 0} = q_{1\to 0} = 1$ and $f_n = 1, g_n = 0$. Here, for a number set $\{x_1, \ldots, x_n\}$ and $1 \le k \le n$,

$$x_{1\to k} := x_1 x_2 \ldots x_k.$$

We first need the following result.

Lemma 3

$$s_{1\to k}^2 = (a_{1\to k+1}f_{1\to k})^2 + (b_{k+1}p_{1\to k})^2, \tag{18}$$
$$p_{1\to k}^2 = (a_{1\to k}f_{1\to k})^2 + (g_k s_{1\to k-1})^2 = s_{1\to k}^2 + (a_{k+1}g_k s_{1\to k-1})^2, \tag{19}$$

for $k = 1, \ldots, n$

Proof. We show (18) by induction. When $k = 1$, using $f_1^2 + g_1^2 = 1$ and $p_1^2 = a_1^2 + (b_1 g_1)^2 = (a_1 f_1)^2 + g_1^2$ gives

$$(a_1 a_2 f_1)^2 + (b_2 p_1)^2 = (a_1 a_2 f_1)^2 + b_2^2((a_1 f_1)^2 + g_1^2) = (a_1 f_1)^2 + (b_2 g_1)^2 = s_1^2.$$

Assume that (18) holds for $k-1$. Since

$$p_{1\to k}^2 = (a_{1\to k}f_{1\to k-1})^2 + (b_k g_k p_{1\to k-1})^2,$$

and from the assumption $s_{1\to k-1}^2 = (a_{1\to k}f_{1\to k-1})^2 + (b_k p_{1\to k-1})^2$, one has

$$\begin{aligned}
s_{1\to k}^2 &= (a_{1\to k}f_{1\to k})^2 + (b_{k+1}g_k s_{1\to k-1})^2 \\
&= (a_{1\to k}f_{1\to k})^2 + (b_{k+1}g_k)^2((a_{1\to k}f_{1\to k-1})^2 + (b_k p_{1\to k-1})^2) \\
&= (a_{1\to k}f_{1\to k})^2(a_{k+1}^2 + b_{k+1}^2) + (a_{1\to k}b_{k+1}f_{1\to k-1}g_k)^2 + (b_k b_{k+1}g_k p_{1\to k-1})^2 \\
&= (a_{1\to k+1}f_{1\to k})^2 + (a_{1\to k}b_{k+1}f_{1\to k-1})^2 + (b_k b_{k+1}g_k p_{1\to k-1})^2 \\
&= (a_{1\to k+1}f_{1\to k})^2 + b_{k+1}^2((a_{1\to k}f_{1\to k-1})^2 + (b_k g_k p_{1\to k-1})^2) \\
&= (a_{1\to k+1}f_{1\to k})^2 + (b_{k+1}p_{1\to k})^2.
\end{aligned}$$

The relation (18) now follows by induction.

From (18), for any k, one has

$$\begin{aligned}
(b_{k+1}p_{1\to k})^2 &= s_{1\to k}^2 - (a_{1\to k+1}f_{1\to k})^2 = (a_{1\to k}f_{1\to k})^2 + (b_{k+1}g_k s_{1\to k-1})^2 - (a_{1\to k+1}f_{1\to k})^2 \\
&= b_{k+1}^2((a_{1\to k}f_{1\to k})^2 + (g_k s_{1\to k-1})^2).
\end{aligned}$$

Hence, by dividing b_{k+1}^2 on both sides and using

$$s_{1\to k}^2 = (a_{1\to k}f_{1\to k})^2 + (b_{k+1}g_k s_{1\to k-1})^2,$$

we obtain (19). □

Proof of Theorem 1. We show (12) and (13) by induction, following the bulge-chasing process. The properties $a_k^2 + b_k^2 = 1$ and $f_k^2 + g_k^2 = 1$ are used throughout the bulge-chasing process.

Since $\mu_1 = 0$, from (11) the first column of $Z_{11}^T Z_{11}$ is parallel to $x = [a_1, b_1 g_1]^T$. From this x we determine a Givens rotation $G_{12}(t_{1,1}, z_{1,1})$ with $p_1, t_{1,1}, z_{1,1}$ defined in (17). By post-multiplying this rotation to Z_{11}, Z_{21}, we create a bulge in the $(2,1)$ entry for each of these two blocks. Since the first round of bulge-chasing only involves the first three rows and columns of four blocks of Z, we focus on the window

$$Z_1 := \left[\begin{array}{ccc|ccc}
p_1 & & & b_1 f_1 & & \\
a_2 b_1 f_1 g_1 p_1^{-1} & a_1 a_2 f_1 p_1^{-1} & b_2 g_2 & -a_2 g_1 & b_2 f_2 & \\
 & & a_3 f_2 & & -a_3 g_2 & b_3 f_3 \\
\hline
a_1 b_1 f_1^2 p_1^{-1} & -g_1 p_1^{-1} & & -a_1 f_1 & & \\
b_1 b_2 f_1 g_1 p_1^{-1} & a_1 b_2 f_1 p_1^{-1} & -a_2 g_2 & -b_2 g_1 & -a_2 f_2 & \\
 & & b_3 f_2 & & -b_3 g_2 & -a_3 f_3
\end{array}\right].$$

Let $G_{12}(c_{1,1}, s_{1,1})$ and $G_{12}(c_{1,2}, s_{1,2})$ be the Givens rotations with $r_1, c_{1,1}, c_{1,1}$ and $s_1, c_{1,2}, s_{1,2}$ defined in (17). By pre-multiplying $G_{12}^T(c_{1,1}, s_{1,1})$ to Z_{11} and Z_{12}, and $G_{12}^T(c_{1,2}, s_{1,2})$ to Z_{21} and

Z_{22}, simple calculations yield

$$Z_1 \leftarrow \left[\begin{array}{ccc|cc|c} r_1 & \frac{a_{1\to 2}a_2b_1f_1^2g_1}{r_1p_1^2} & \frac{a_2b_{1\to 2}f_1g_{1\to 2}}{r_1p_1} & \frac{b_1f_1s_1^2}{r_1p_1} & \frac{a_2b_{1\to 2}f_{1\to 2}g_1}{r_1p_1} & \\ & \frac{a_{1\to 2}f_1}{r_1} & \frac{b_2g_2p_1}{r_1} & -\frac{a_2g_1}{r_1p_1} & \frac{b_2f_2p_1}{r_1} & \\ & & a_3f_2 & & -a_3g_2 & b_3f_3 \\ \hline \frac{b_1f_1s_1}{p_1} & -\frac{a_{1\to 2}a_2f_1g_1}{s_1p_1} & -\frac{a_2b_2g_{1\to 2}}{s_1} & -s_1 & \frac{a_2b_2f_2g_1}{s_1} & \\ & \frac{b_2p_1}{s_1} & -\frac{a_{1\to 2}f_1g_2}{s_1} & & -\frac{a_{1\to 2}f_{1\to 2}}{s_1} & \\ & & b_3f_2 & & -b_3g_2 & -a_3f_3 \end{array}\right].$$

Let $G_{12}(t_{1,2}, z_{1,2})$ and $G_{23}(t_{2,1}, z_{2,1})$ be the Givens rotations with $q_1, t_{1,2}, z_{1,2}$ and $p_2, t_{2,1}, z_{2,1}$ defined in (17). By post-multiplying $G_{12}(t_{1,2}, z_{1,2})$ to Z_{12}, Z_{22}, and $G_{23}(t_{2,1}, z_{2,1})$ to Z_{11}, Z_{21}, one obtains

$$Z_1 \leftarrow \left[\begin{array}{ccc|cc|c} r_1 & \frac{a_2b_1f_1g_1p_2}{r_1p_1} & & \frac{b_1f_1q_1s_1}{r_1p_1} & & \\ & \frac{p_{1\to 2}}{r_1} & & -\frac{a_2g_1p_{1\to 2}p_2}{r_1q_1s_1} & \frac{b_2f_2r_1p_1}{q_1s_1} & \\ & \frac{a_3b_2f_2g_2}{p_2} & \frac{a_{1\to 3}f_{1\to 2}}{p_{1\to 2}} & -\frac{a_2a_3b_2f_2g_{1\to 2}}{q_1s_1} & -\frac{a_3g_2s_1}{q_1} & b_3f_3 \\ \hline \frac{b_1f_1s_1}{p_1} & -\frac{a_2g_1p_2}{s_1} & & -q_1 & & \\ & \frac{a_{1\to 2}b_2f_{1\to 2}f_2}{s_1p_2} & -\frac{g_2p_1(p_2^2+(b_2f_2)^2)}{s_1p_2} & -\frac{a_{1\to 2}a_2b_2f_{1\to 2}f_2g_1}{q_1s_1^2} & -\frac{a_{1\to 2}f_{1\to 2}}{q_1} & \\ & \frac{b_2b_3f_2g_2}{p_2} & \frac{a_{1\to 2}b_3f_{1\to 2}}{p_{1\to 2}} & -\frac{a_2b_2b_3f_2g_{1\to 2}}{q_1s_1} & -\frac{b_3g_2s_1}{q_1} & -a_3f_3 \end{array}\right].$$

Now the initial bulge is chased from the $(2,1)$ to the $(3,2)$ position in Z_{11} and Z_{21}, and we are ready for the next bulge-chasing step.

Assume that k steps of bulge-chasing have been carried out. Now the zoom-in window is formed by the columns and rows $k+1, k+2, k+3$ of Z_{11}, Z_{21}, denoted by $Z_{k+1}^{(1)}$, and the columns of $k, k+1, k+2$ and rows $k+1, k+2, k+3$ of Z_{12} and Z_{22}, denoted by $Z_{k+1}^{(2)}$:

$$Z_{k+1}^{(1)} = \left[\begin{array}{ccc} \frac{p_{1\to k+1}}{r_{1\to k}} & & \\ \frac{a_{k+2}b_{k+1}f_{k+1}g_{k+1}}{p_{k+1}} & \frac{a_{1\to k+2}f_{1\to k+1}}{p_{1\to k+1}} & b_{k+2}g_{k+2} \\ & & a_{k+3}f_{k+2} \\ \hline \frac{a_{1\to k+1}b_{k+1}f_{1\to k+1}f_{k+1}}{s_{1\to k}p_{k+1}} & -\frac{g_{k+1}p_{1\to k}(p_{k+1}^2+(b_{k+1}f_{k+1})^2)}{s_{1\to k}p_{k+1}} & \\ \frac{b_{k+1}b_{k+2}f_{k+1}g_{k+1}}{p_{k+1}} & \frac{a_{1\to k+1}b_{k+2}f_{1\to k+1}}{p_{1\to k+1}} & -a_{k+2}g_{k+2} \\ & & b_{k+3}f_{k+2} \end{array}\right],$$

$$Z_{k+1}^{(2)} = \left[\begin{array}{ccc} -\frac{a_{k+1}g_kp_{1\to k+1}p_{k+1}}{q_ks_kr_{1\to k}} & \frac{b_{k+1}f_{k+1}r_{1\to k}p_{1\to k}}{q_{1\to k}s_{1\to k}} & \\ -\frac{a_{k+1}a_{k+2}b_{k+1}f_{k+1}g_kg_{k+1}}{q_ks_k} & -\frac{a_{k+2}g_{k+1}s_{1\to k}}{q_{1\to k}} & b_{k+2}f_{k+2} \\ & & -a_{k+3}g_{k+2} \\ \hline -\frac{a_{1\to k+1}a_{k+1}b_{k+1}f_{1\to k+1}f_{k+1}g_k}{q_ks_ks_{1\to k}} & -\frac{a_{1\to k+1}f_{1\to k+1}}{q_{1\to k}} & \\ -\frac{a_{k+1}b_{k+1}b_{k+2}f_{k+1}g_kg_{k+1}}{q_ks_k} & -\frac{b_{k+2}g_{k+1}s_{1\to k}}{q_{1\to k}} & -a_{k+2}f_{k+2} \\ & & -b_{k+3}g_{k+2} \end{array}\right].$$

At step $k+1$, in order to annihilate the $(k+2, k+1)$ entry of Z_{11} and Z_{21} as well as the $(k+2, k)$ entry of Z_{12} and Z_{22}, we use the Givens rotations $G_{k+1,k+2}(c_{k+1,1}, s_{k+1,1})$ and $G_{k+1,k+2}(c_{k+1,2}, s_{k+1,2})$ with $r_{k+1}, c_{k+1,1}, s_{k+1,1}$ and $s_{k+1}, c_{k+1,2}, s_{k+1,2}$ defined in (17). After the transformations, it is

straightforward to show that the zoom-in windows are given by

$$Z_{k+1}^{(1)} \leftarrow \begin{bmatrix} r_{k+1} & \frac{a_{1\to k+2}a_{k+2}b_{k+1}f_{1\to k+1}f_{k+1}g_{k+1}}{r_{k+1}p_{k+1}p_{1\to k+1}} & \frac{a_{k+2}b_{k+1}b_{k+2}f_{k+1}g_{k+1}g_{k+2}}{r_{k+1}p_{k+1}} \\ & \frac{a_{1\to k+2}f_{1\to k+1}}{r_{1\to k+1}} & \frac{b_{k+2}g_{k+2}p_{1\to k+1}}{r_{1\to k+1}} \\ \hline \frac{b_{k+1}f_{k+1}s_{k+1}}{p_{k+1}} & -\frac{a_{1\to k+2}a_{k+2}f_{1\to k+1}g_{k+1}}{s_{k+1}p_{1\to k+1}} & a_{k+3}f_{k+2} \\ & \frac{b_{k+2}p_{1\to k+1}}{s_{1\to k+1}} & -\frac{a_{k+2}b_{k+2}g_{k+1}g_{k+2}}{s_{k+1}} \\ & & -\frac{a_{1\to k+2}f_{1\to k+1}g_{k+2}}{s_{1\to k+1}} \\ & & b_{k+3}f_{k+2} \end{bmatrix},$$

$$Z_{k+1}^{(2)} \leftarrow \begin{bmatrix} -\frac{a_{k+1}g_kr_{k+1}p_{k+1}}{q_ks_k} & \frac{b_{k+1}f_{k+1}s_{1\to k+1}s_{k+1}}{r_{k+1}p_{k+1}q_{1\to k}} & \frac{a_{k+2}b_{k+1}b_{k+2}f_{k+1}f_{k+2}g_{k+1}}{r_{k+1}p_{k+1}} \\ & -\frac{a_{k+2}g_{k+1}p_{1\to k}((b_{k+1}f_{k+1}r_{1\to k})^2+(s_{1\to k}p_{k+1})^2)}{r_{1\to k+1}p_{k+1}q_{1\to k}s_{1\to k}} & \frac{b_{k+2}f_{k+2}p_{1\to k+1}}{r_{1\to k+1}} \\ \hline -\frac{a_{k+1}b_{k+1}f_{k+1}g_ks_{k+1}}{q_ks_k} & -\frac{s_{1\to k+1}}{q_{1\to k}} & -a_{k+3}g_{k+2} \\ & & -\frac{a_{k+2}b_{k+2}f_{k+2}g_{k+1}}{s_{k+1}} \\ & & -\frac{a_{1\to k+2}f_{1\to k+2}}{s_{1\to k+1}} \\ & & -b_{k+3}g_{k+2} \end{bmatrix},$$

where we used the identity (18) to simplify the $(k+1,k+2)$ entry of Z_{21}.

Next, we annihilate simultaneously the $(k+1,k+3)$ entry in both Z_{11} and Z_{21}, and the $(k+1,k+2)$ entry in both Z_{12} and Z_{22} with the Givens rotations $G_{k+2,k+3}(t_{k+2,1},z_{k+2,1})$ and $G_{k+1,k+2}(t_{k+1,2},z_{k+1,2})$, respectively, with $p_{k+2},t_{k+2,1},z_{k+2,1}$ and $q_{k+1},t_{k+1,2},z_{k+1,2}$ defined in (17). After these transformations, one has

$$Z_{k+1}^{(1)} \leftarrow \begin{bmatrix} r_{k+1} & \frac{a_{k+2}b_{k+1}f_{k+1}g_{k+1}p_{k+2}}{r_{k+1}p_{k+1}} & \\ & \frac{p_{1\to k+2}}{r_{1\to k+1}} & \\ & \frac{a_{k+3}b_{k+2}f_{k+2}g_{k+2}}{p_{k+2}} & \frac{a_{1\to k+3}f_{1\to k+2}}{p_{1\to k+2}} \\ \hline \frac{b_{k+1}f_{k+1}s_{k+1}}{p_{k+1}} & -\frac{a_{k+2}g_{k+1}p_{k+2}}{s_{k+1}} & \\ & \frac{a_{1\to k+2}b_{k+2}f_{1\to k+2}f_{k+2}}{s_{1\to k+1}p_{k+2}} & -\frac{g_{k+2}p_{1\to k+1}(p_{k+2}^2+(b_{k+2}f_{k+2})^2)}{s_{1\to k+1}p_{k+2}} \\ & \frac{b_{k+2}b_{k+3}f_{k+2}g_{k+2}}{p_{k+2}} & \frac{a_{1\to k+2}b_{k+3}f_{1\to k+2}}{p_{1\to k+2}} \end{bmatrix},$$

$$Z_{k+1}^{(2)} \leftarrow \begin{bmatrix} -\frac{a_{k+1}g_kr_{k+1}p_{k+1}}{q_ks_k} & \frac{b_{k+1}f_{k+1}q_{k+1}s_{k+1}}{r_{k+1}p_{k+1}} & \\ & -\frac{a_{k+2}g_{k+1}p_{1\to k+1}p_{k+2}^2}{q_{k+1}s_{k+1}r_{1\to k+1}} & \frac{b_{k+2}f_{k+2}r_{1\to k+1}p_{1\to k+1}}{q_{1\to k+1}s_{1\to k+1}} \\ & -\frac{a_{k+2}a_{k+3}b_{k+2}f_{k+2}g_{k+1}g_{k+2}}{q_{k+1}s_{k+1}} & -\frac{a_{k+3}g_{k+2}s_{1\to k+1}}{q_{1\to k+1}} \\ \hline -\frac{a_{k+1}b_{k+1}f_{k+1}g_ks_{k+1}}{q_ks_k} & -q_{k+1} & \\ & -\frac{a_{1\to k+2}a_{k+2}b_{k+2}f_{1\to k+2}f_{k+2}g_{k+1}}{s_{1\to k+1}q_{k+1}s_{k+1}} & -\frac{a_{1\to k+2}f_{1\to k+2}}{q_{1\to k+1}} \\ & -\frac{a_{k+2}b_{k+2}b_{k+3}f_{k+2}g_{k+1}g_{k+2}}{q_{k+1}s_{k+1}} & -\frac{b_{k+3}g_{k+2}s_{1\to k+1}}{q_{1\to k+1}} \end{bmatrix},$$

where, in order to get the expressions of the $(k+1,k+1)$, $(k+1,k+2)$, and $(k+2,k+2)$ entries of Z_{12} we have used (19).

After $n-2$ steps, the zoom-in window of Z is the submatrix formed by the last two rows and columns of Z_{11} and Z_{21}, denoted by $Z_{n-1}^{(1)}$, and the submatrix formed by the last two rows and

three columns of Z_{12} and Z_{22}, denoted by $Z_{n-1}^{(2)}$:

$$Z_{n-1}^{(1)} = \begin{bmatrix} \frac{p_{1\to n-1}}{r_{1\to n-2}} & \\ \frac{a_n b_{n-1} f_{n-1} g_{n-1}}{p_{n-1}} & \frac{a_{1\to n} f_{1\to n-1}}{p_{1\to n-1}} \\ \frac{a_{1\to n-1} b_{n-1} f_{1\to n-1} f_{n-1}}{s_{1\to n-2} p_{n-1}} & -\frac{g_{n-1} p_{1\to n-2}(p_{n-1}^2+(b_{n-1}f_{n-1})^2)}{s_{1\to n-2} p_{n-1}} \\ \frac{b_{n-1} b_n f_{n-1} g_{n-1}}{p_{n-1}} & \frac{a_{1\to n-1} b_n f_{1\to n-1}}{p_{1\to n-1}} \end{bmatrix},$$

$$Z_{n-1}^{(2)} = \begin{bmatrix} -\frac{a_{n-1} g_{n-2} p_{1\to n-2} p_{n-1}^2}{q_{n-2} s_{n-2} r_{1\to n-2}} & \frac{b_{n-1} f_{n-1} r_{1\to n-2} p_{1\to n-2}}{q_{1\to n-2} s_{n-2}} & \\ -\frac{a_{n-1} a_n b_{n-1} f_{n-1} g_{n-2} g_{n-1}}{q_{n-2} s_{n-2}} & -\frac{a_n g_{n-1} s_{1\to n-2}}{q_{1\to n-2}} & b_n \\ -\frac{a_{1\to n-2} a_{n-1}^2 b_{n-1} f_{1\to n-2} f_{n-1}^2 g_{n-2}}{s_{1\to n-2} q_{n-2} s_{n-2}} & -\frac{a_{1\to n-1} f_{1\to n-1}}{q_{1\to n-2}} & \\ -\frac{a_{n-1} b_{n-1} b_n f_{n-1} g_{n-2} g_{n-1}}{q_{n-2} s_{n-2}} & -\frac{b_n g_{n-1} s_{1\to n-2}}{q_{1\to n-2}} & -a_n \end{bmatrix}.$$

Similarly, using the Givens rotations $G_{n-1,n}(c_{n-1,1}, s_{n-1,1})$ and $G_{n-1,n}(c_{n-1,2}, s_{n-1,2})$ with $r_{n-1}, c_{n-1,1}, s_{n-1,1}$ and with $s_{n-1}, c_{n-1,2}, s_{n-1,2}$ defined in (17), we annihilate the $(n, n-1)$ entry of Z_{11}, Z_{21} as well as the $(n, n-2)$ entry of Z_{12}, Z_{22}. These transformations yield

$$Z_{n-1}^{(1)} \leftarrow \begin{bmatrix} r_{n-1} & \frac{a_n b_{n-1} f_{n-1} g_{n-1} p_n}{r_{n-1} p_{n-1}} \\ & r_n \\ \frac{b_{n-1} f_{n-1} s_{n-1}}{p_{n-1}} & -\frac{a_n g_{n-1} p_n}{s_{n-1}} \\ & \frac{b_n p_{1\to n}}{p_n s_{1\to n-1}} \end{bmatrix},$$

where

$$p_n := \frac{a_{1\to n} f_{1\to n-1}}{p_{1\to n-1}}, \qquad r_n := \frac{p_{1\to n}}{r_{1\to n-1}},$$

and

$$Z_{n-1}^{(2)} \leftarrow \begin{bmatrix} -\frac{a_{n-1} g_{n-2} r_{n-1} p_{n-1}}{q_{n-2} s_{n-2}} & \frac{b_{n-1} f_{n-1} s_{1\to n-2} s_{n-1}^2}{r_{n-1} p_{n-1} q_{1\to n-2}} & \frac{a_n b_{n-1} b_n f_{n-1} g_{n-1}}{r_{n-1} p_{n-1}} \\ & -\frac{a_n g_{n-1} p_{1\to n-2}((b_{n-1} f_{n-1} r_{1\to n-2})^2+(s_{1\to n-2} p_{n-1})^2)}{r_{1\to n-1} p_{n-1} q_{1\to n-2} s_{1\to n-2}} & \frac{b_n p_{1\to n-1}}{r_{1\to n-1}} \\ -\frac{a_{n-1} b_{n-1} f_{n-1} g_{n-2} s_{n-1}}{q_{n-2} s_{n-2}} & -\frac{s_{1\to n-1}}{q_{1\to n-2}} & -\frac{a_n b_n g_{n-1}}{s_{n-1}} \\ & & -\frac{a_{1\to n} f_{1\to n-1}}{s_{1\to n-1}} \end{bmatrix}.$$

Finally, in order to annihilate the $(n-1, n)$ entry in both Z_{12}, Z_{22}, we need the Givens rotation $G_{n-1,n}(t_{n-1,2}, z_{n-1,2})$ with $q_{n-1}, t_{n-1,2}, z_{n-1,2}$ defined in (17). After the transformation, $Z_{n-1}^{(2)}$ becomes

$$Z_{n-1}^{(2)} \leftarrow \begin{bmatrix} -\frac{a_{n-1} g_{n-2} r_{n-1} p_{n-1}}{q_{n-2} s_{n-2}} & \frac{b_{n-1} f_{n-1} q_{n-1} s_{n-1}}{r_{n-1} p_{n-1}} & \\ & -\frac{a_n g_{n-1} p_{1\to n-1} p_n^2}{q_{n-1} s_{n-1} r_{1\to n-1}} & \frac{b_n r_{1\to n-1} p_{1\to n-1}}{q_{1\to n-1} s_{1\to n-1}} \\ -\frac{a_{n-1} b_{n-1} f_{n-1} g_{n-2} s_{n-1}}{q_{n-2} s_{n-2}} & -q_{n-1} & \\ & -\frac{a_n b_n g_{n-1} p_{1\to n}}{s_{1\to n-1} q_{n-1} s_{n-1}} & -\frac{p_{1\to n}}{q_{1\to n-1}} \end{bmatrix}.$$

With $g_n = 0$, one has

$$p_{1\to n} = r_{1\to n} = s_{1\to n} = q_{1\to n} = a_{1\to n} f_{1\to n-1}.$$

Eventually, $Z \to \tilde{Z}$ with

$$\begin{bmatrix} \tilde{Z}_{11} \\ \hline \tilde{Z}_{21} \end{bmatrix} \leftarrow \left[\begin{array}{ccccc} r_1 & \frac{a_2 b_1 f_1 g_1 p_2}{r_1 p_1} & & & \\ & r_2 & \ddots & & \\ & & \ddots & \frac{a_{n-1} b_{n-2} f_{n-2} g_{n-2} p_{n-1}}{r_{n-2} p_{n-2}} & \\ & & & r_{n-1} & \frac{a_n b_{n-1} f_{n-1} g_{n-1} p_n}{r_{n-1} p_{n-1}} \\ & & & & r_n \\ \hline \frac{b_1 f_1 s_1}{p_1} & -\frac{a_2 g_1 p_2}{s_1} & & & \\ & \frac{b_2 f_2 s_2}{p_2} & \ddots & & \\ & & \ddots & -\frac{a_{n-1} g_{n-2} p_{n-1}}{s_{n-2}} & \\ & & & \frac{b_{n-1} f_{n-1} s_{n-1}}{p_{n-1}} & -\frac{a_n g_{n-1} p_n}{s_{n-1}} \\ & & & & \frac{b_n f_n s_n}{p_n} \end{array}\right],$$

$$\begin{bmatrix} \tilde{Z}_{12} \\ \hline \tilde{Z}_{22} \end{bmatrix} \leftarrow \left[\begin{array}{ccccc} \frac{b_1 f_1 q_1 s_1}{r_1 p_1} & & & & \\ -\frac{a_2 g_1 r_2 p_2}{q_1 s_1} & \frac{b_2 f_2 q_2 s_2}{r_2 p_2} & & & \\ & \ddots & \ddots & & \\ & & -\frac{a_{n-1} g_{n-2} r_{n-1} p_{n-1}}{q_{n-2} s_{n-2}} & \frac{b_{n-1} f_{n-1} q_{n-1} s_{n-1}}{r_{n-1} p_{n-1}} & \\ & & & -\frac{a_n g_{n-1} r_n p_n}{q_{n-1} s_{n-1}} & \frac{b_n q_n s_n}{r_n p_n} \\ \hline -q_1 & & & & \\ -\frac{a_2 b_2 f_2 g_1 s_2}{q_1 s_1} & -q_2 & & & \\ & \ddots & \ddots & & \\ & & -\frac{a_{n-1} b_{n-1} f_{n-1} g_{n-2} s_{n-1}}{q_{n-2} s_{n-2}} & -q_{n-1} & \\ & & & -\frac{a_n b_n g_{n-1} s_n}{q_{n-1} s_{n-1}} & -q_n \end{array}\right].$$

It is easily shown that the matrix $\tilde{Z}$ can be expressed as

$$\tilde{Z} = \left[\begin{array}{ccccc|ccccc} r_1 & \frac{\alpha_{21}\beta_{11}p_2}{r_1 p_1} & & & & \frac{\alpha_{13} q_1 s_1}{r_1 p_1} & & & & \\ & r_2 & \ddots & & & -\frac{\beta_{13} r_2 p_2}{q_1 s_1} & \ddots & & & \\ & & \ddots & \frac{\alpha_{n1}\beta_{n-1,1} p_n}{r_{n-1} p_{n-1}} & & & \ddots & \frac{\alpha_{n-1,3} q_{n-1} s_{n-1}}{r_{n-1} p_{n-1}} & & \\ & & & r_n & & & & -\frac{\beta_{n-1,3} r_n p_n}{q_{n-1} s_{n-1}} & \frac{\alpha_{n3} q_n s_n}{r_n p_n} \\ \hline \frac{\alpha_{13} s_1}{p_1} & -\frac{\beta_{13} p_2}{s_1} & & & & -q_1 & & & & \\ & \frac{\alpha_{23} s_2}{p_2} & \ddots & & & -\frac{\alpha_{24}\beta_{14} s_2}{q_1 s_1} & \ddots & & & \\ & & \ddots & -\frac{\beta_{n-1,3} p_n}{s_{n-1}} & & & \ddots & -q_{n-1} & & \\ & & & \frac{\alpha_{n3} s_n}{p_n} & & & & -\frac{\alpha_{n4}\beta_{n-1,4} s_n}{q_{n-1} s_{n-1}} & -q_n \end{array}\right].$$

The formulas in (12) can be derived using this matrix expression and (13). □

The sets $\{p_k\}, \{r_k\}, \{s_k\}, \{q_k\}$ have the following properties.

Lemma 4 *(a) For all* $1 \le k \le n$,

$$0 < p_k, r_k, s_k, q_k < 1.$$

(b) For $k = 1, 2, \ldots, n$,

$$1 > r_{1\to k} > p_{1\to k} > s_{1\to k} \ge a_{1\to n} f_{1\to n-1}, \quad 1 > r_{1\to k} > q_{1\to k} > s_{1\to k} \ge a_{1\to n} f_{1\to n-1}.$$

Proof. (a) Using the fact that $r_1, \ldots, r_n$ are the diagonal entries of $\tilde{Z}_{11}$, that $-q_1, \ldots, -q_n$ are the diagonal entries of $\tilde{Z}_{22}$, and that $\tilde{Z}$ is orthogonal, one has $0 < r_k, q_k < 1$. From the formulas in (13),

$$0 < p_k \le \sqrt{\alpha_{k1}^2 + \beta_{k1}^2} < 1, \quad 0 < s_k \le \sqrt{\alpha_{k4}^2 + \beta_{k4}^2} < 1.$$

(b) Using the product form $\tilde{Z} = \tilde{Z}_o \tilde{Z}_e$, one obtains

$$\tilde{f}_k = \frac{q_{1\to k}}{r_{1\to k}}.$$

Because $\tilde{f}_k < 1$, we have $q_{1\to k} < r_{1\to k}$. The inequalities $r_{1\to k} > p_{1\to k}$ and $q_{1\to k} > s_{1\to k}$ follow from the formulas in (17). The inequality $p_{1\to k} > s_{1\to k}$ is from (19). Finally, because $0 < s_k < 1$ for any k, one has $s_{1\to k} \ge s_{1\to n} = a_{1\to n} f_{1\to n-1}$. □

Appendix B. Error analysis for Algorithm 4

In the following, δ with a subscript is a tiny number of size $O(\mu)$. Let $\hat{\alpha}$ be the computed value of α (or $\tilde{\alpha}$). We use the notation $\hat{\alpha} = \alpha(1+\epsilon_\alpha)$ (or $\hat{\alpha} = \tilde{\alpha}(1+\epsilon_\alpha)$) and consider first order errors. The following error analysis for the computations of a Givens rotation is from [5, Lemma 5].

Lemma 5 *Suppose that γ, σ and $\rho = \|x\|_2$ are the values computed by Algorithm 3 with $x = [x_1, x_2]^T$ in exact arithmetic, and let $\hat{\gamma}, \hat{\sigma}, \hat{\rho}$ be the computed values from a slight perturbed x with $[x_1(1+\epsilon_{x_1}), x_2(1+\epsilon_{x_2})]^T$. Then*

$$\hat{\gamma} = \gamma(1+\epsilon_\gamma), \quad \hat{\sigma} = \sigma(1+\epsilon_\sigma), \quad \hat{\rho} = \rho(1+\epsilon_\rho),$$

where

$$\begin{aligned}
\epsilon_\rho &= \epsilon_{x_1}\gamma^2 + \epsilon_{x_2}\sigma^2 + \delta_\rho, && |\delta_\rho| \le \frac{13}{4}\mu, \\
\epsilon_\gamma &= (\epsilon_{x_1} - \epsilon_{x_2})\sigma^2 + \delta_\gamma, && |\delta_\gamma| \le \frac{21}{4}\mu, \\
\epsilon_\sigma &= (\epsilon_{x_2} - \epsilon_{x_1})\gamma^2 + \delta_\sigma, && |\delta_\sigma| \le \frac{21}{4}\mu.
\end{aligned}$$

Proof of Theorem 2. Consider the computed values at the kth iteration of Algorithm 4. We have

$$\hat{\beta}_{kj} = \tilde{\beta}_{kj}(1+\epsilon_{\beta_{kj}}), \quad \hat{\alpha}_{kj} = \tilde{\alpha}_{kj}(1+\epsilon_{\alpha_{kj}}), \quad j = 1,2,3,4,$$

where

$$\begin{array}{ll}
\epsilon_{\alpha_{k1}} = \epsilon_{r_k}, & \epsilon_{\beta_{k1}} = \epsilon_{s_{k,1}} + \epsilon_{p_{k+1}} + \delta_{\beta_{k1}}, \\
\epsilon_{\alpha_{k2}} = \epsilon_{s_k} - \epsilon_{p_k} + \delta_{\alpha_{k2}}, & \epsilon_{\beta_{k2}} = \epsilon_{p_{k+1}} - \epsilon_{s_k} + \delta_{\beta_{k2}}, \\
\epsilon_{\alpha_{k3}} = \epsilon_{\alpha_{k2}} + \epsilon_{q_k} - \epsilon_{r_k} + \delta_{\alpha_{k3}}, & \epsilon_{\beta_{k3}} = \epsilon_{\beta_{k2}} + \epsilon_{r_{k+1}} - \epsilon_{q_k} + \delta_{\beta_{k3}}, \\
\epsilon_{\alpha_{k4}} = \epsilon_{q_k}, & \epsilon_{\beta_{k4}} = \epsilon_{z_{k,2}} + \epsilon_{s_{k+1}} + \delta_{\beta_{k4}},
\end{array} \tag{20}$$

and

$$|\delta_{\beta_{k1}}|, |\delta_{\beta_{k4}}| \le \mu, \quad |\delta_{\beta_{k2}}|, |\delta_{\beta_{k3}}|, |\delta_{\alpha_{k2}}|, |\delta_{\alpha_{k3}}| \le 2\mu.$$

Define $x_{k-1,1} := \alpha_{k1} t_{k-1,1}$. Then

$$\hat{x}_{k-1,1} = x_{k-1,1}(1+\epsilon_{x_{k-1,1}}).$$

By Lemma 5, we have

$$\hat{p}_k = p_k(1+\epsilon_{p_k}), \quad \hat{t}_{k,1} = t_{k,1}(1+\epsilon_{t_{k,1}}), \quad \hat{z}_{k,1} = z_{k,1}(1+\epsilon_{z_{k,1}}),$$

where

$$\begin{aligned}
\epsilon_{p_k} &= \epsilon_{x_{k-1,1}} t_{k,1}^2 + \delta_{p_k}, \quad |\delta_{p_k}| \le \frac{13}{4}\mu, && (21)\\
\epsilon_{t_{k,1}} &= \epsilon_{x_{k-1,1}} z_{k,1}^2 + \delta_{t_{k,1}}, \quad |\delta_{t_{k,1}}| \le \frac{21}{4}\mu, && (22)\\
\epsilon_{z_{k,1}} &= -\epsilon_{x_{k-1,1}} t_{k,1}^2 + \delta_{z_{k,1}}, \quad |\delta_{z_{k,1}}| \le \frac{21}{4}\mu. && (23)
\end{aligned}$$

Since

$$\hat{x}_{k,1} = x_{k,1}(1+\epsilon_{x_{k,1}}),$$

by (22),

$$\epsilon_{x_{k,1}} = \epsilon_{t_{k,1}} + \delta_1 = \epsilon_{x_{k-1,1}} z_{k,1}^2 + \delta_2, \quad |\delta_2| \le \frac{25}{4}\mu. \tag{24}$$

Similarly,

$$\hat{r}_k = r_k(1+\epsilon_{r_k}), \quad \hat{c}_{k,1} = c_{k,1}(1+\epsilon_{c_{k,1}}), \quad \hat{s}_{k,1} = s_{k,1}(1+\epsilon_{s_{k,1}}),$$

where, by (21), (23), and using $c_{k,1}^2 + s_{k,1}^2 = 1$,

$$\begin{aligned}
\epsilon_{r_k} &= \epsilon_{x_{k-1,1}} t_{k,1}^2 (c_{k,1}^2 - s_{k,1}^2) + \epsilon_{c_{k-1,1}} c_{k,1}^2 + \delta_{r_k}, \quad |\delta_{r_k}| \le \frac{21}{2}\mu,\\
\epsilon_{c_{k,1}} &= (2\epsilon_{x_{k-1,1}} t_{k,1}^2 + \epsilon_{c_{k-1,1}} + \delta_4) s_{k,1}^2 + \delta_{c_{k,1}}, \quad |\delta_4| \le \frac{21}{2}\mu, \quad |\delta_{c_{k,1}}| \le \frac{21}{4}\mu, && (25)\\
\epsilon_{s_{k,1}} &= -(2\epsilon_{x_{k-1,1}} t_{k,1}^2 + \epsilon_{c_{k-1,1}} + \delta_4) c_{k,1}^2 + \delta_{s_{k,1}}, \quad |\delta_{s_{k,1}}| \le \frac{21}{4}\mu.
\end{aligned}$$

Combining (24) with (25), one has

$$\begin{bmatrix} |\epsilon_{x_{k,1}}| \\ |\epsilon_{c_{k,1}}| \end{bmatrix} \le \begin{bmatrix} z_{k,1}^2 & 0 \\ 2t_{k,1}^2 s_{k,1}^2 & s_{k,1}^2 \end{bmatrix} \begin{bmatrix} |\epsilon_{x_{k-1,1}}| \\ |\epsilon_{c_{k-1,1}}| \end{bmatrix} + \begin{bmatrix} \frac{25}{4} \\ \frac{21+42s_{k,1}^2}{4} \end{bmatrix} \mu. \tag{26}$$

Using the same trick as in [5, Lemma 7], we get

$$\begin{bmatrix} |\epsilon_{x_{k,1}}| \\ |\epsilon_{c_{k,1}}| \end{bmatrix} \le \left(\sum_{i=1}^{k} \begin{bmatrix} \prod_{j=i}^{k} z_{j,1}^2 & 0 \\ 2s_{i,1}^2 \left(1 - \prod_{j=i}^{k} z_{j,1}^2\right) & \prod_{j=i}^{k} s_{j,1}^2 \end{bmatrix} \right) \begin{bmatrix} \frac{25}{4} \\ \frac{63}{4} \end{bmatrix} \mu.$$

Hence,

$$|\epsilon_{x_{k,1}}| \le \frac{25k}{4}\mu, \quad |\epsilon_{c_{k,1}}| \le \frac{113k}{4}\mu,$$

and, therefore,

$$|\epsilon_{p_k}| \le \frac{25k-12}{4}\mu, \quad |\epsilon_{t_{k,1}}|,\ |\epsilon_{z_{k,1}}| \le \frac{25k-4}{4}\mu, \quad |\epsilon_{r_k}| \le \frac{138k-96}{4}\mu, \quad |\epsilon_{s_{k,1}}| \le \frac{163k-100}{4}\mu.$$

In the same way, for the errors in $s_k, c_{k,2}, s_{k,2}, q_k, t_{k,2}, z_{k,2}$, with $x_{k-1,2} = c_{k-1,2}\alpha_{k4}$, we obtain

$$\begin{bmatrix} |\epsilon_{x_{k,2}}| \\ |\epsilon_{t_{k,2}}| \end{bmatrix} \le \begin{bmatrix} s_{k,2}^2 & 0 \\ 2c_{k,2}^2 z_{k,2}^2 & z_{k,2}^2 \end{bmatrix} \begin{bmatrix} |\epsilon_{x_{k-1,2}}| \\ |\epsilon_{t_{k-1,2}}| \end{bmatrix} + \begin{bmatrix} \frac{25}{4} \\ \frac{21+42z_{k,2}^2}{4} \end{bmatrix}. \tag{27}$$

Similarly, one has $|\epsilon_{x_{k,2}}| \le \frac{25k}{4}\mu$, and for

$$\begin{aligned}
&\hat{s}_k = s_k(1+\epsilon_{s_k}), \quad \hat{c}_{k,2} = c_{k,2}(1+\epsilon_{c_{k,2}}), \quad \hat{s}_{k,2} = s_{k,2}(1+\epsilon_{s_{k,2}});\\
&\hat{q}_k = q_k(1+\epsilon_{q_k}), \quad \hat{t}_{k,2} = t_{k,2}(1+\epsilon_{t_{k,2}}), \quad \hat{z}_{k,2} = z_{k,2}(1+\epsilon_{z_{k,2}}),
\end{aligned}$$

we have

$$|\epsilon_{s_k}| \le \frac{25k-12}{4}\mu, \quad |\epsilon_{c_{k,2}}|,\ |\epsilon_{s_{k,2}}| \le \frac{25k-4}{4}\mu,$$
$$|\epsilon_{q_k}| \le \frac{138k-96}{4}\mu, \quad |\epsilon_{t_{k,2}}| \le \frac{113k}{4}\mu, \quad |\epsilon_{z_{k,2}}| \le \frac{163k-100}{4}\mu.$$

Substituting these bounds into (20) yields (14).

If $z_{k,1}^2, z_{k,2}^2, s_{k,1}^2, s_{k,2}^2 \le \tau < 1$ for all k, then from (26) and (27), we obtain that

$$\begin{bmatrix} |\epsilon_{x_{k,1}}| \\ |\epsilon_{c_{k,1}}| \end{bmatrix} \le \begin{bmatrix} \tau & 0 \\ 2\tau & \tau \end{bmatrix} \begin{bmatrix} |\epsilon_{x_{k-1,1}}| \\ |\epsilon_{c_{k-1,1}}| \end{bmatrix} + \begin{bmatrix} \frac{25}{4} \\ \frac{21+42\tau}{4} \end{bmatrix} \mu,$$
$$\begin{bmatrix} |\epsilon_{x_{k,2}}| \\ |\epsilon_{t_{k,2}}| \end{bmatrix} \le \begin{bmatrix} \tau & 0 \\ 2\tau & \tau \end{bmatrix} \begin{bmatrix} |\epsilon_{x_{k-1,2}}| \\ |\epsilon_{t_{k-1,2}}| \end{bmatrix} + \begin{bmatrix} \frac{25}{4} \\ \frac{21+42\tau}{4} \end{bmatrix} \mu.$$

Following [5, Lemma 8],

$$|\epsilon_{x_{k,1}}|,\ |\epsilon_{x_{k,2}}| \le \frac{25}{4(1-\tau)}\mu, \quad |\epsilon_{c_{k,1}}|,\ |\epsilon_{t_{k,2}}| \le \left(\frac{50\tau}{4(1-\tau)^2} + \frac{21(2\tau+1)}{4(1-\tau)}\right)\mu.$$

Using the same derivations now yields (15). □

References

[1] G. S. Ammar, W. B. Gragg, and L. Reichel, On the eigenproblem for orthogonal matrices, in *Proceedings of the 25th Conference on Decision and Control*, IEEE, Piscataway, pp. 1963–1966, 1986.

[2] G. S. Ammar, W. B. Gragg, and L. Reichel, Determination of Pisarenko frequency estimates as eigenvalues of an orthogonal matrix, in *Advanced Algorithms and Architectures for Signal Processing II*, ed. F. T. Luk, Proceedings of the Society of Photo-Optical Instrumentation Engineers (SPIE), vol. 826, The International Society for Optical Engineering, Bellingham, WA, pp. 143–145, 1987.

[3] G. S. Ammar, L. Reichel, and D. C. Sorensen, Algorithm 730: An implementation of a divide and conquer algorithm for the unitary eigenproblem, *ACM Trans. Math. Software*, 18:292–307, 1992, and 20:161, 1994.

[4] A. Bunse–Gerstner and L. Elsner, Schur parameter pencils for the solution of the unitary eigenproblem, *Linear Algebra Appl.*, 154/156:741–778, 1991.

[5] J. Demmel and W. Kahan, Accurate singular values of bidiagonal matrices, *SIAM J. Sci. Stat. Comput.*, 11:873–912, 1990.

[6] G. H. Golub and C. F. Van Loan, *Matrix Computations,* 3*rd ed.*, Johns Hopkins University Press, Baltimore, 1996.

[7] W. B. Gragg, Positive definite Toeplitz matrices, the Arnoldi process for isometric operators, and Gaussian quadrature on the unit circle, *J. Comput. Appl. Math.*, 46:183-198, 1993. This is a slightly edited version of a paper published in Russian in *Numerical Methods in Linear Algebra*, Moscow University Press, ed. E. S. Nikolaev, Moscow University Press, Moscow, pp. 16–32, 1982.

[8] W. B. Gragg, The QR algorithm for unitary Hessenberg matrices, *J. Comput. Appl. Math.*, 16:1–8, 1986.

[9] W. B. Gragg, Stabilization of the UHQR-algorithm, in *Advances in Computational Mathematics*, eds. Z. Chen, Y. Li, C. A. Micchelli, and Y. Xu, Marcel Dekker, New York, pp. 139–154, 1999.

[10] W. B. Gragg and L. Reichel, A divide and conquer method for unitary eigenproblems, in *Hypercube Multiprocessors 1987*, ed. M. T. Heath, SIAM, Philadelphia, pp. 639–647, 1987.

[11] W. B. Gragg and L. Reichel, A divide and conquer method for unitary and orthogonal eigenproblems, *Numer. Math.*, 57:695–718, 1990.

[12] M. Gu, R. Guzzo, X.-B. Chi, and X.-Q. Cao, A stable divide and conquer algorithm for the unitary eigenproblem, *SIAM J. Matrix Anal. Appl.*, 25:385–404, 2003.

[13] W. B. Jones, O Njåstad, and W. J. Thron, Moment theory, orthogonal polynomials, quadrature and continued fractions associated with the unit circle, *Bull. London Math. Soc.*, 21:113–152, 1989.

[14] L. Reichel and G. S. Ammar, Fast approximation of dominant harmonics by solving an orthogonal eigenvalue problem, in *Mathematics in Signal Processing II*, ed. J. G. McWhirter, Oxford University Press, Oxford, pp. 575–591, 1990.

[15] M. Stewart, An error analysis of a unitary Hessenberg QR algorithm, *SIAM J. Matrix. Anal. Appl.*, 28:40–67, 2006.

[16] B. D. Sutton, Computing the complete CS decomposition, *Numer. Algorithms* 50:33–65, 2009.

[17] B. D. Sutton, Stable computation of the CS decomposition: simultaneous bidiagonalization, *SIAM J. Matrix Anal. Appl.*, 33:1–21, 2012.

[18] T.-L. Wang and W. B. Gragg, Convergence of the shifted QR algorithm for unitary Hessenberg matrices, *Math. Comp.*, 71:1473–1476, 2001.

[19] T.-L. Wang and W. B. Gragg, Convergence of the unitary QR algorithm with a unimodular Wilkinson shift, *Math. Comp.*, 72:375–385, 2002.