Numerische Mathematik (2020) 144:23–53
https://doi.org/10.1007/s00211-019-01080-4

Numerische Mathematik

Fast QR iterations for unitary plus low rank matrices

Roberto Bevilacqua[1] · **Gianna M. Del Corso**[1] · **Luca Gemignani**[1]

Received: 8 February 2019 / Revised: 2 September 2019 / Published online: 23 October 2019

Abstract
Some fast algorithms for computing the eigenvalues of a (block) companion matrix have recently appeared in the literature. In this paper we generalize the approach to encompass unitary plus low rank matrices of the form $A = U + XY^H$ where U is a general unitary matrix. Three important cases for applications are U unitary diagonal, U unitary block Hessenberg and U unitary in block CMV form. Our extension exploits the properties of a larger matrix $\hat{A}$ obtained by a certain embedding of the Hessenberg reduction of A suitable to maintain its structural properties. We show that $\hat{A}$ can be factored as product of lower and upper unitary Hessenberg matrices possibly perturbed in the first k rows, and, moreover, such a data-sparse representation is well suited for the design of fast eigensolvers based on the QR iteration. The resulting eigenvalue algorithm is fast and backward stable.

Mathematics Subject Classification 65F15

1 Introduction

The design of specialized algorithms that compute the eigenvalues of unitary matrices is so far a classical topic in structured numerical linear algebra (compare [16] and the references given therein). The major applications that stimulate research in this area lie in signal processing [2], in time series analysis [1], in Gaussian quadrature on

The research of the last two authors was partially supported by GNCS project "Analisi di matrici sparse e data-sparse: metodi numerici ed applicazioni"and by the project sponsored by University of Pisa under the Grant PRA-2017-05.

✉ Gianna M. Del Corso
gianna.delcorso@unipi.it

Roberto Bevilacqua
roberto.bevilacqua@unipi.it

Luca Gemignani
luca.gemignani@unipi.it

[1] Dipartimento di Informatica, Università di Pisa, Largo Bruno Pontecorvo 3, 56127 Pisa, Italy

the unit circle [33] and in trigonometric approximation theory [23]. In the last years many authors have dealt with the issue of efficiently computing the eigenvalues of unitary matrices perturbed by low rank corrections (see the books [22,41] for general overviews of these developments). Motivations come from (matrix) polynomial root-finding problems [3,11,12,14,29] and generally from eigenvalue problems associated with finite truncations of large (block) unitary matrices arising in the aforementioned applications [39] as well as in certain statistical methods for the analysis of complex systems [27]. Typically in these applications large unitary matrices are represented in condensed form using the (block) Hessenberg [32] or the (block) CMV shape [4,36].

The papers [13,17] presented the first fast and numerically reliable eigensolvers for certain low rank perturbations of unitary matrices, while in [40] the analogous case of low rank perturbation of Hermitian structure was addressed. Since then two challenging issues have attracted much work: (1) the search of numerical algorithms that are computationally efficient with respect to the size both of the matrix and of the perturbation and (2) a formal proof of the backward stability of these algorithms. Very recently numerical methods which combine all these two features have been proposed in [5,7] for computing the eigenvalues of companion and block companion matrices, respectively. These methods incorporate some techniques that are specifically adjusted to exploit the properties of companion and block companion forms. In particular, the Hessenberg reduction of a block companion matrix is found by relying upon the decomposition of the matrix as product of scalar companion matrices which provides the factored representation of the Hessenberg reduction to be used in the QR iterative process.

In this paper we generalize the approach pursued in [5,7] to deal with input matrices of the form $A_i = U + XY^H \in \mathbb{C}^{n\times n}$ where U is a general $n \times n$ unitary matrix and $X, Y \in \mathbb{C}^{n\times k}$ with $k \leq n$. Eigenvalue computation is customarily a two-step process. Firstly the input matrix A_i is reduced in Hessenberg form by unitary similarity, that is, $A_i \rightarrow A_f := QA_iQ^H$, where the final A_f is Hessenberg and Q unitary. Then the QR iteration is applied to the Hessenberg reduction A_f for computing its Schur form. Each iterate generated by a fast adaptation of the QR scheme inherits the condensed representation of the initial matrix A_f obtained at the end of the Hessenberg reduction. By setting $U := QUQ^H$, $X := QX$ and $Y = QY$ it is found that this matrix $A_f = U + XY^H$ is still unitary plus low rank in Hessenberg form.

The efficient computation of a condensed representation of unitary plus low rank matrices in Hessenberg form is the subject of the papers [10,30]. There it is shown that A_i can be embedded into a larger matrix $\hat{A}_i$ which is converted by unitary similarity in the Hessenberg matrix $\hat{A}_f$. Both $\hat{A}_i$ and $\hat{A}_f$ are specified in factored form as the product of three factors, that is $\hat{A}_i = L_i \cdot F_i \cdot R_i$ and $\hat{A}_f = L_f \cdot F_f \cdot R_f$ with L_i and L_f unitary lower k-Hessenberg matrices, R_i and R_f unitary upper k-Hessenberg matrices and F_i, F_f unitary upper Hessenberg matrix perturbed in the first k rows. Then a bulge chasing technique to carry out the transformations $L_i \rightarrow L_f$, $R_i \rightarrow R_f$ and $F_i \rightarrow F_f$ using $O(n^2k)$ operations is derived. The construction greatly simplifies when the matrix A_i is a unitary (block) Hessenberg or CMV matrix modified in the first/last rows/columns since in these cases the three factors L_i, F_i and R_i are easily and

cheaply obtained. In particular, this is the case of block companion matrices, for which we can compute the factored Hessenberg representation with $O(n^2k)$ operations.

Our present work aims at designing a fast version of the implicit QR eigenvalue method [26] for unitary plus low rank Hessenberg matrices $\hat{A} = \hat{A}_f = U + XY^H$, $U \in \mathbb{C}^{n\times n}$ unitary and $X, Y \in \mathbb{C}^{n\times k}$, represented in compressed form as $\hat{A} = L \cdot F \cdot R$, where L is the product of k unitary lower Hessenberg matrices, R is the product of k unitary upper Hessenberg matrices and the middle factor F is a unitary upper Hessenberg matrix perturbed in the first k rows. The representation is data-sparse since it involves $O(nk)$ data storage consisting of $O(k)$ vectors of length n and $O(nk)$ Givens rotations. Specifically, the main results are:

1. The development of a bulge-chasing technique for performing one step of the implicit QR algorithm applied to a matrix specified in the LFR format by returning as output the updated factored LFR representation of the new iterate.
2. A careful look at the structural properties of Hessenberg matrices given in the LFR format by implying that, under some auxiliary assumptions on the properness of the factors L and R, the middle matrix F is reducible iff the same holds for the Hessenberg matrix. It follows that the deflation in the Hessenberg iterate can be revealed in the middle factor converging to an upper triangular matrix in the limit.
3. A cost and error analysis of the resulting adaptation of the implicit QR algorithm. We prove that one single QR iteration requires $O(nk)$ ops only and it is backward stable.

The paper is organized as follows. In Sect. 2 we recall some preliminary material about the structural properties of possibly perturbed unitary matrices. Section 3 gives the theoretical foundations of our algorithm which is presented and analyzed in Sect. 4. In Sect. 5 the backward stability of the algorithm is formally proved. Finally, in Sect. 6 we show the results of numerical experiments followed by some conclusions and future work in Sect. 7.

2 Preliminaries

We first recall some basic properties of unitary matrices which play an important role in the derivation of our methods.

Lemma 1 *Let U be a unitary matrix of size n. Then*

$$\operatorname{rank}(U(\alpha, \beta)) = \operatorname{rank}(U(J\backslash\alpha, J\backslash\beta)) + |\alpha| + |\beta| - n$$

where $J = \{1, 2, \ldots, n\}$ and α and β are subsets of J. If $\alpha = \{1, \ldots, h\}$ and $\beta = J\backslash\alpha$, then we have

$$\operatorname{rank}(U(1:h, h+1:n)) = \operatorname{rank}(U(h+1:n, 1:h)), \quad \textit{for all} \quad h = 1, \ldots, n-1.$$

Proof This well known symmetry in the rank-structure of unitary matrices follows by a straightforward application of the nullity theorem [25]. □

Lemma 2 *Let U be a unitary matrix of size n, and let $\alpha, \beta \subseteq \{1, 2, \ldots, n\}$, such that $|\alpha| = |\beta|$, then*

$$|\det(U(\alpha, \beta))| = |\det(U(J\backslash\alpha, J\backslash\beta))|.$$

Proof See Gantmacher [28], Property 2 on page 21. □

Definition 1 A matrix H is called *k-upper Hessenberg* if $h_{ij} = 0$ when $i > j + k$. Similarly, H is called *k-lower Hessenberg* if $h_{ij} = 0$ when $j > i + k$. In addition, when H is *k-upper Hessenberg* (*k-lower Hessenberg*) and the outermost entries are non-zero, that is, $h_{j+k,j} \neq 0$ ($h_{j,j+k} \neq 0$), $1 \leq j \leq n - k$, then the matrix is called *proper*.

Note that for $k = 1$, that is when the matrix is in Hessenberg form, the notion of properness coincides with that of being unreduced. Also, a k-upper Hessenberg matrix $H \in \mathbb{C}^{n\times n}$ is proper iff $\det(H(k + 1 : n, 1 : n - k)) \neq 0$. Similarly a k-lower Hessenberg matrix H is proper iff $\det(H(1 : n - k, k + 1 : n)) \neq 0$. For $k < 0$ a k-Hessenberg matrix is actually a strictly triangular matrix with $-k$ vanishing diagonals.

It is well known [42] that, given a non-zero n-vector x we can build a zero-creating matrix from a product of $n - 1$ Givens matrices $\mathcal{G}_1 \cdots \mathcal{G}_{n-1}$, where $\mathcal{G}_i = I_{i-1} \oplus G_i \oplus I_{n-i-1}$ and G_i is a 2×2 complex Givens rotations of the form $\begin{bmatrix} c & -s \\ s & \bar{c} \end{bmatrix}$ such that $|c|^2 + s^2 = 1$, with $s \in \mathbb{R}$, $s \geq 0$. The subscript index i indicates the active part of the matrix $\mathcal{G}_i$. The descending sequence of Givens rotations $H = \mathcal{G}_1 \cdots \mathcal{G}_{n-1}$ turns out to be a unitary upper Hessenberg matrix such that $H\mathrm{x} = \alpha \mathrm{e}_1$, and $|\alpha| = \|\mathrm{x}\|_2$. Note that H is proper if and only if all the Givens matrices appearing in its factorization are non trivial, i. e. $s \neq 0$ [7]. Generalizing this result we obtain the following lemma.

Lemma 3 *Let $X \in \mathbb{C}^{m\times k}$, $k < m$, be of full rank. Then*

1. *there exist a unitary k-upper Hessenberg matrix H and an upper triangular matrix $T \in \mathbb{C}^{m\times k}$, $T = \begin{bmatrix} T_k \\ 0 \end{bmatrix}$ with $T_k \in \mathbb{C}^{k\times k}$ nonsingular such that*

$$HX = T. \tag{2.1}$$

2. *The product of the outermost entries of H is given by*

$$\prod_{i=1}^{m-k} |h_{i+k,i}| = \frac{|\det(X(m - k + 1 : m, 1 : k))|}{\prod_{i=1}^{k} \sigma_i(X)}, \tag{2.2}$$

   where $\sigma_1(X), \sigma_2(X), \ldots, \sigma_k(X)$ are the singular values of X.
3. *Let s be the maximum index such that* $\mathrm{rank}(X(s : m, :)) = k$, *then $h_{i+k,i} \neq 0$ for $i = 1, \ldots, s - 1$.*

Proof The existence of H is proved by construction. Relation (2.1) defines a QR decomposition of the matrix X. The unitary factor H can be determined as product of k unitary upper Hessenberg matrices $H = H_k \cdots H_1$ such that $H_h = I_{h-1} \oplus \tilde{H}_h$, $H_0 = I_m$ and $\tilde{H}_h X^{(h-1)}(h : m, h) = t_{hh}\mathbf{e}_1$ where $X^{(h-1)} = H_{h-1} \cdots H_1 H_0 \cdot X$, $1 \le h \le k$.

Now, let us split H into blocks

$$H = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix},$$

where H_{12} is $k \times k$ and H_{21} is $(m-k) \times (m-k)$, upper triangular. The product of the outermost entries of H is given by $\det(H_{21}) = \prod_{i=1}^{m-k} h_{i+k,i}$. Since H is unitary, and $X = H^H T$ we have

$$\det(X(m-k+1 : m, :)) = \det(H_{12}^H)\det(T_k).$$

From Lemma 2 we have

$$|\det(H_{21})| = |\det(H_{12})| = \frac{|\det(X((m-k+1 : m, :))|}{|\det(T_k)|}.$$

We get relation (2.2) observing that if $X = P\Sigma Q^H$ is the SVD decomposition of X, $(HP)\Sigma Q^H$ is the SVD decomposition of T and hence $\sigma_i(T_k) = \sigma_i(X)$, $i = 1, \ldots, k$.

Finally, let s be the maximum index such that $\operatorname{rank}(X(s : m, 1 : k)) = k$. Then $s \le m-k+1$ and, moreover, from $X(s : m, 1 : k) = H^H(s : m, 1 : k)T_k$ we obtain that $\operatorname{rank}(H(1 : k, s : m)) = k$ since T_k is nonsingular. Using Lemma 1 we have $k = \operatorname{rank}(H^H(s : m, 1 : k)) = \operatorname{rank}(H(k+1 : m, 1 : s-1) + k + (m-s+1) - m$, meaning that $H(k+1 : m, 1 : s-1)$ has full rank equal to $s-1$. Since $H(k+1 : m, 1 : s-1)$ is upper triangular we have that $h_{i+k,i} \neq 0$ for $i = 1, \ldots, s-1$. □

Remark 1 From the proof of Lemma 3 we know that H can be written as a product of k upper Hessenberg matrices, i.e., $H = H_k H_{k-1} \cdots H_1$. The jth of these Hessenberg matrices is the one annihilating the j-th column of $X^{(j-1)}$ from row $j+1$ to row m. Then each H_j can be factored as the product of $m-j$ Givens rotations. From this observation we get that $H_j = \mathcal{G}_j^{(j)} \cdots \mathcal{G}_{m-1}^{(j)}$ where each $\mathcal{G}_i^{(j)}$ is a Givens rotation acting on rows $i, i+1$. This decomposition of H corresponds to annihilate progressively the lower subdiagonals of H by means of rotations working on the left. Alternatively, we can proceed by zeroing the lower subdiagonals of H by means of rotations working on the right and acting on the columns of H. In this way we find a different factorization of the form $H = D\hat{H}_k \hat{H}_{k-1} \cdots \hat{H}_1$ where $\hat{H}_j = \hat{\mathcal{G}}_1^{(j)} \hat{\mathcal{G}}_2^{(j)} \cdots \hat{\mathcal{G}}_{m-k+j-1}^{(j)}$ and D is unitary diagonal.

3 Representation

Generalizing the approach discussed in [7] for the companion matrix, it is useful to embed the unitary plus low-rank matrix A into a larger matrix to guarantee the properness of some factors of the representation that we are going to introduce.

Theorem 1 *Let $A \in \mathbb{C}^{n\times n}$ be such that $A = U + XY^H$, with U unitary and X, Y $n \times k$ full rank matrices. We can construct an $N \times N$ matrix $\hat{A}$, $N = n + k$, such that $\hat{A} = \hat{U} + \hat{X}\hat{Y}^H$, with $\hat{U}$ unitary, $\hat{X}, \hat{Y}$ $N \times k$ full rank matrices, $\hat{X}(n+1 : N, :) = -I_k$, and such that*

$$\hat{A} = \begin{bmatrix} A & B \\ 0_{k,n} & 0_k \end{bmatrix}, \quad \text{for a suitable} \quad B \in \mathbb{C}^{n\times k}. \tag{3.1}$$

Proof The proof is constructive. We first compute the economy size QR decomposition of matrix Y, $Y = QR$ where $Q \in \mathbb{C}^{n\times k}$ and $R \in \mathbb{R}^{k\times k}$. Set $\tilde{Y} = Q$ and $\tilde{X} = XR^H$. We still have $XY^H = \tilde{X}\tilde{Y}^H$ but now Y has orthonormal columns, i.e., $\tilde{Y}^H\tilde{Y} = I_k$. Define

$$\hat{U} = \begin{bmatrix} U - U\tilde{Y}\tilde{Y}^H & B \\ \tilde{Y}^H & 0_k \end{bmatrix}, \tag{3.2}$$

where $B = U\tilde{Y}$ and

$$\hat{X} = \begin{bmatrix} \tilde{X} + B \\ -I_k \end{bmatrix} \quad \hat{Y} = \begin{bmatrix} \tilde{Y} \\ 0_k \end{bmatrix}. \tag{3.3}$$

Note that $\hat{U} + \hat{X}\hat{Y}^H$ has the structure described in (3.1) and, moreover by direct calculation we can verify that $\hat{U}$ is unitary. □

From now on we denote by $N = n + k$ the dimension of the matrix $\hat{A}$. It is worth pointing out that in view of the block triangular structure in (3.1) the Hessenberg reduction of the original matrix A can be easily adjusted to specify the Hessenberg reduction of the larger matrix $\hat{A}$. Thus, in the following it is always assumed that both A and $\hat{A}$ are in upper Hessenberg form.

Theorem 2 *Let $\hat{A} = \hat{U} + \hat{X}\hat{Y}^H \in \mathbb{C}^{N\times N}$ be the upper Hessenberg matrix obtained by embedding an $n \times n$ proper Hessenberg matrix A as described in Theorem 1. Then we can factorize $\hat{A}$ as follows*

$$\hat{A} = L \cdot F \cdot R, \quad \text{where} \tag{3.4}$$

L is a proper unitary k-lower Hessenberg matrix.
R is a unitary k-upper Hessenberg matrix. Moreover, the leading $n - 1$ entries in the outermost diagonal of R, $r_{i+k,i}$, $i = 1, \ldots, n - 1$, are nonzero.
$F = Q + TZ^H$, where Q is a block diagonal unitary Hessenberg matrix, $Q = \left[\begin{array}{c|c} I_k & \\ \hline & \hat{Q} \end{array}\right]$, with $\hat{Q}$ proper, $T = \begin{bmatrix} T_k \\ 0_{n,k} \end{bmatrix}$ with $T_k \in \mathbb{C}^{k\times k}$ upper triangular, and $Z \in \mathbb{C}^{N\times k}$, with full rank.

If in addition A is nonsingular then R is proper.

Proof First note that from the properness of A it follows that $\text{rank}(A) \geq n-1$. From Theorem 1 we have that $\hat{X}$ has full rank, and $\det(\hat{X}(n+1:N,:)) = \det(-I_k) \neq 0$, hence, by Lemma 3 we can find a proper L^H and a nonsingular square triangular T_k such that $L^H\hat{X} = T$, with $T = [T_k^T, 0_{k,n}]^T$. For the properness of L^H and A, we get that $L^H\hat{A}$ is a proper $(k+1)$-upper Hessenberg matrix and moreover the matrix $V = L^H\hat{U} = L^H\hat{A} - T\hat{Y}^H$, is unitary and still a proper $(k+1)$-upper Hessenberg matrix because $T\hat{Y}^H$ is null under the kth row.

Now the matrix V can be factored as $V = QR$, where R is unitary k-upper Hessenberg, and Q^H is the unitary lower Hessenberg matrix obtained as the product of the $n-1$ Givens rotations annihilating from the top entries in the outermost diagonal of V, i.e., $Q^H = \mathcal{G}_{N-1}\ldots\mathcal{G}_{k+2}\mathcal{G}_{k+1}$, where $\mathcal{G}_i$ acts on rows $i, i+1$. Since the first k rows are not involved, the matrix Q has the structure $Q = \left[\begin{array}{c|c} I_k & \\ \hline & \hat{Q} \end{array}\right]$, where $\hat{Q}$ is unitary $n \times n$ Hessenberg. Moreover, since V is proper, $\hat{Q}$ is proper as well.

From the definitions of V, Q and R we have:

$$\hat{A} = LV + LT\hat{Y}^H = L(Q + TZ^H)R,$$

where $Z = RY$. The matrix Z is full rank, since R is unitary and Y is full rank.

Now let us consider the submatrices $R(k+1:N, 1:j)$, for $j = n-1$ and $j = n$. In both cases, from the relation $R = Q^H(L^H\hat{A} - TY^H)$ and the structural properties of the matrices involved therein, we have that

$$\begin{aligned}\text{rank}(R(k+1:N,1:j)) &= \text{rank}(\hat{Q}^H L^H(k+1:N,1:n)\hat{A}(1:n,1:j)) \\ &= \text{rank}(A(1:n,1:j)).\end{aligned}$$

For $j = n-1$, since A is proper, the rank of that submatrix is $n-1$. This implies that the entries $r_{i+k,i}$, $i = 1,\ldots,n-1$, are nonzero. For $j = n$, if A is nonsingular, then the rank is n, so $r_{N,n}$ is nonzero. □

The following theoremproves that the product of the factors L, F, R having the properties stated in Theorem 2 is indeed an upper Hessenberg matrix with the last k rows equal to zero. It reveals also that deflation can be performed only when one of the subdiagonal entries of Q approaches zero.

Theorem 3 *Let $L, R \in \mathbb{C}^{N\times N}$, where L is a proper unitary k-lower Hessenberg matrix and R is a unitary k-upper Hessenberg matrix. Let Q be a block diagonal unitary upper Hessenberg matrix of the form $Q = \left[\begin{array}{c|c} I_k & \\ \hline & \hat{Q} \end{array}\right]$, with $\hat{Q}$ $n \times n$ unitary Hessenberg and T_k a $k \times k$ nonsingular upper triangular matrix. Then*

1. $L(n+1:N, 1:k) = -T_k^{-1}$.
2. *Setting $Z^H = L(n+1:N,:)Q$, $T = [T_k^H, 0]^T$, and $F = Q + TZ^H$, we have that*

(a) *the matrix* $\hat{A} = LFR$ *is upper Hessenberg, with* $\hat{A}(n+1:N,:) = 0$*, that is*

$$\hat{A} = \begin{bmatrix} A & * \\ 0_{k,n} & 0_{k,k} \end{bmatrix},$$

(b) $\hat{A}$ *is a unitary plus rank* k *matrix.*

3. *If* R *is proper then the upper Hessenberg matrix* $A \in \mathbb{C}^{n\times n}$ *is nonsingular. In this case* A *is proper if and only if* $\hat{Q}$ *is proper.*

Proof To prove part 1, note that $\hat{X} = LT$, and $\hat{X}(n+1:N,:) = -I_k$, hence $-I_k = L(n+1:N,:)T = L(n+1:N,1:k)T_k$, and then we have $L(n+1:N,1:k) = -T_k^{-1}$.

For part 2, let us consider the matrix $C = L\,Q$. This matrix is unitary with a k-quasiseparable structure below the k-th upper diagonal. In fact, for any $h, h = 2,\ldots n+1$ we have

$$\begin{aligned} C(h:N,1:h+k-2) &= L(h:N,:)\,Q(:,1:h+k-2) \\ &= L(h:N,1:h+k-1)\,Q(1:h+k-1,1:h+k-2). \end{aligned}$$

Applying Lemma 1 we have $\operatorname{rank}(L(h:N,1:h+k-1)) = k$, implying that $\operatorname{rank}(C(h:N,1:h+k-2)) \le k$.

Since $C(n+1:N,1:k) = L(n+1:N,1:k)$ is nonsingular, we conclude that $\operatorname{rank}(C(h:N,1:h+k-2)) = k$. From this observation we can then find a set of generators $P, S \in \mathbb{C}^{(N\times k)}$ and a $(1-k)$-upper Hessenberg matrix U_k such that $U_k(1,k) = U_k(n,N) = 0$ so that $C = PS^H + U_k$ (see [9,22]). Moreover, we have $C(n+1:N,1:k) = L(n+1:N.:)Q(:,1:k) = M$. Then we can recover the rank k correction PS^H from the left-lower corner of C obtaining

$$PS^H = -C(:,1:k)T_kC(n+1:N,:).$$

Since $C(:,1:k) = LQ(:,1:k) = L(:,1:k)$, we get that $PS^H = -LTZ^H$, and hence $U_k = L(Q + TZ^H) = L\,F$. We conclude the proof of part (b), by noticing that $\hat{A} = U_k\,R$ is upper Hessenberg as it is the product of a $(1-k)$-upper Hessenberg matrix by a k-upper Hessenberg matrix. Moreover, we find that $\hat{A}(n+1:N,:) = U_k(n+1:N,:)R = 0$ since $U_k(n+1:N,:) = 0$.

To prove part 3, as already observed in the proof of Theorem 2, we use the rank equation

$$\operatorname{rank}(R(k+1:N,1:n)) = \operatorname{rank}(\hat{A}(k+1:n,1:n)) = \operatorname{rank}(A),$$

thus, if R is proper then A is nonsingular. In this case, from the properness of L and noticing that

$$a_{i+1,i} = q_{i+k+1,i+k}\,r_{i+k,i}/\bar{l}_{i+1,i+k+1}, \quad i = 1,\ldots,n-1, \tag{3.5}$$

we get that $a_{i+1,i} = 0$ iff $q_{i+k+1,i+k} = 0$. □

Remark 2 From the previous Theorem, one sees that when a matrix $\hat{A}$ is represented in the LFR form where L, F and R have the structural properties required, then A is nonsingular if and and only if R is proper. Moreover, from (3.5) one deduces that one of the outermost entries $a_{i+1,i}$ can be zero only if we have either $q_{i+k+1,i+k} = 0$ or $r_{i,i+k} = 0$. Vice-versa, we can have that $r_{N,n} = 0$ without any subdiagonal entry of A being equal to zero. This is the only case where A is proper and singular.

The next theorem shows that the compressed representation $\hat{A} = LFR$ is eligible to be used under the QR eigenvalue algorithm for computing the eigenvalues of $\hat{A}$ and, a fortiori, of A.

Theorem 4 *Let $\hat{A} = \hat{U} + \hat{X}\hat{Y}^H \in \mathbb{C}^{N\times N}$, $N = n+k$ be a Hessenberg matrix obtained by embedding a proper $n \times n$ Hessenberg matrix $A = U + XY^H$ as described in Theorem 1. Let P be the unitary factor of the QR factorization of $p_d(\hat{A})$, where $p_d(x)$ is a monic polynomial of degree d. Let $\hat{A}^{(1)} = P^H \hat{A} P$ be the matrix obtained by applying a step of the multi-shifted QR algorithm to the matrix $\hat{A}$ with shifts being the roots of $p_d(x)$. Then, we have that*

$$\hat{A}^{(1)} = \begin{bmatrix} A^{(1)} & B^{(1)} \\ 0_{k,n} & 0_k \end{bmatrix},$$

where $A^{(1)}$ is the matrix generated by applying one step of the multi-shifted QR algorithm to the matrix A with shifts being the roots of $p_d(x)$. Both $\hat{A}^{(1)}$ and $A^{(1)}$ are upper Hessenberg and if $A^{(1)}$ is proper then the factorization of $\hat{A}^{(1)} = L^{(1)} F^{(1)} R^{(1)}$ exists and has still the same properties stated in Theorem 2; in particular, $L^{(1)}$ is proper and, if A is nonsingular also $R^{(1)}$ is proper.

Proof From (3.1) we have

$$\hat{A} = \begin{bmatrix} A & B \\ 0_{k,n} & 0_k \end{bmatrix}.$$

Since $p_d(\hat{A})$ is also block triangular, we can take

$$P = \begin{bmatrix} P_1 & 0_{n,k} \\ 0_{k,n} & P_2 \end{bmatrix}, \tag{3.6}$$

where P_1 and P_2 are unitary. Hence,

$$\hat{A}^{(1)} = \begin{bmatrix} A^{(1)} & B^{(1)} \\ 0_{k,n} & 0_k \end{bmatrix},$$

where $A^{(1)}$ is the matrix generated by applying one step of the multi-shifted QR algorithm to the matrix A with shifts being the roots of $p_d(x)$. We have $\hat{A}^{(1)} = P^H \hat{A} P = P^H \hat{U} P + P^H \hat{X}\hat{Y}^H P = U_1 + \hat{X}_1 \hat{Y}_1^H$, setting $U_1 = P^H \hat{U} P$ and $\hat{X}_1 = P^H \hat{X}$, $\hat{Y}_1 = P^H \hat{Y}$. Because P_2 is unitary, we have that $|\det(\hat{X}_1(n+1:N,:))| =$

$|\det(\hat{X}(n+1:N,:))| \neq 0$, then the conditions given by Lemma 3 are satisfied and we can conclude that $L^{(1)}$ is proper. We note that $\hat{A}^{(1)}$ and $A^{(1)}$ are upper Hessenberg for the well known properties of the shifted QR algorithm. When $A^{(1)}$ is proper then we can apply Theorem 2 which guarantees the existence of the representation of $\hat{A}^{(1)}$. □

The algorithm we propose is an implicitly shifted QR method, and hence the factors $L^{(1)}$, $F^{(1)}$, $R^{(1)}$ are obtained by manipulating Givens rotations. In Sect. 4 we describe the algorithm and we show that the factors obtained with the implicit procedure agree with the requirements given in Theorem 3. The implicit Q-Theorem [31] guarantees that the matrix obtained after an implicit step is basically the same matrix one get with an explicit one. The next result gives a quantitative measure of the properness of matrices L and R generated along the QR iterative method.

Corollary 1 *Let $\hat{U}$, $\hat{X}$, $\hat{Y}$ as described in Theorem 1 and let $\hat{A} = L\,F\,R$ as in Theorem 2. Let $K = 1/\prod_{i=1}^{k}\sigma_i(X)$, where $\sigma_i(X)$ are the singular values of X. We have:*

1. *the module of the product of the outermost entries of L, is such that $\prod_{i=1}^{n}|l_{i,i+k}| = K$ and is constant over QR steps. Moreover for each outermost entry of L we have $K \leq |l_{i,i+k}| \leq 1$.*
2. *the module of the product of the outermost entries of R is $\prod_{i=1}^{n}|r_{i+k,i}| = K|\det A|$ and is constant over QR steps. Moreover for each outermost entry of R we have $K|\det(A)| \leq |r_{i+k,i}| \leq 1$.*

Proof To prove part 1 we first observe that $|\det(\hat{X}(n+1:N,:))| = 1$, because $\hat{X}(n+1:N,:)) = -I_k$ by construction. To prove that the product of the outermost entries remains unchanged over QR steps, we use Theorem 4 observing that $|\det(\hat{X}(n+1:N,:))| = |\det(\hat{X}_1((n+1:N,:))|$ and that $\hat{X}$ and $\hat{X}_1$ have the same singular values. We get the thesis applying part 2 of Lemma 3.

We can also see that $0 < |l_{j,j+k}| \leq 1$ and that $|l_{j,j+k}| = K/|\prod_{i=1,i\neq j}^{n} l_{i,i+k}|$. Since $|\prod_{i=1,i\neq j}^{n} l_{i,i+k}| \leq 1$ we have $|l_{j,j+k}| \geq K$.

The relation on $\prod_{i=1}^{n}|r_{i+k,i}|$ is similarly deduced by applying Binet rule to equality $L(k+1:N,1:n)A = \hat{Q}R(k+1:N,1:n)$. After a QR step the first k rows of $V_1 = L^{(1)H}\hat{U}^{(1)}$ are orthonormal and, moreover, the $k\times k$ submatrix in the right upper corner of V_1 satisfies

$$\begin{aligned}|\det(V_1(1:k,n+1:N))| &= \det(V_1(k+1:N,1:n))| \\ &= |\det(L^{(1)}(1:n,k+1:N))|\,|\det(A^{(1)})| = K|\det(A)|.\end{aligned}$$

□

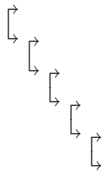
Remark 3 As observed in [5,7] also for our representation it is possible to recover the structure of the $N\times k$ matrix Z from the representation (3.4). In fact, we have $\hat{A}(n+1:N,:) = L(n+1:N,:)(Q+TZ^H)R = 0_{k,N}$. Since R is nonsingular, and $L(n+1:N,1:k) = -T_k^{-1}$ we have that

$$Z^H = L(n+1:N,:)Q. \tag{3.7}$$

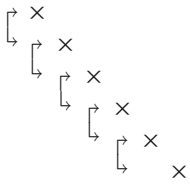
4 The algorithm

In this section we show how to carry out a single step of Francis's implicitly shifted QR algorithm acting directly on the representation of the matrix described in Sect. 3. In the sequel we assume R to be a proper k-upper Hessenberg matrix. In the view of the previous sections this means that A is nonsingular. If, otherwise, A is singular then we can perform a step of the QR algorithm with zero shift to remove the singularity. In this way the parametrization of R is automatically adjusted to specify a proper matrix in its active part.

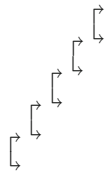
It is convenient to describe the representation and the algorithm using a pictorial representation already introduced in several papers (compare with [6] and the references given therein). Specifically, the action of a Givens rotation acting on two consecutive rows of the matrix is depicted as . A chain of descending two-pointed arrows as below



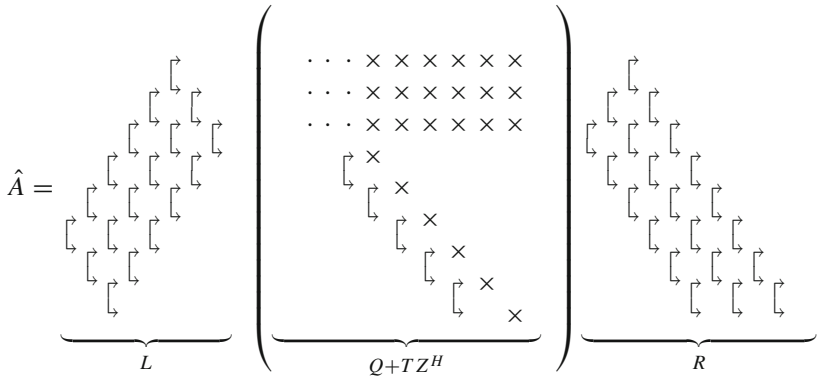
represents a unitary upper Hessenberg matrix (in this case a 6×6 matrix). Vice versa, since any unitary Hessenberg matrix H of size n can be factored as $H = \mathcal{G}_1(c_1)\mathcal{G}_2(c_2)\cdots\mathcal{G}_{n-1}(c_{n-1})\,D$, where $\mathcal{G}_k(c_k) = I_{k-1} \oplus G_k(c_k) \oplus I_{n-k-1}$, $G_k(c_k) = \begin{bmatrix} c_k & s_k \\ -s_k & \bar{c}_k \end{bmatrix}$, with $|c_k|^2 + s_k^2 = 1$, $s_k \geq 0$, and D is unitary diagonal, we have that H can be represented as follows



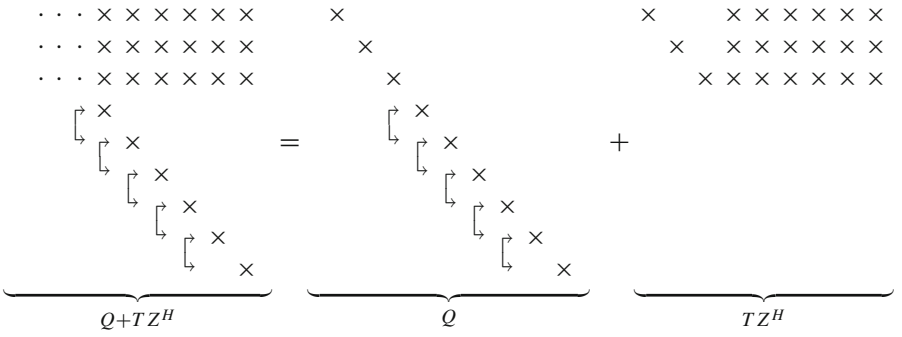
where the $\times$ represent the entries of the diagonal phase matrix D. Similarly the chain



represents a unitary lower Hessenberg matrix. As observed in Remark 1 the k-Hessenberg matrices L and R appearing in the representation of $\hat{A}$ can be factored as the product of k unitary Hessenberg matrices, and any unitary Hessenberg can be represented through their Schur parametrization [32] by ascending or descending chains of Givens rotations times a unitary diagonal matrix. In our case the unitary diagonal matrices that would be necessary to get the Schur parametrization in terms of Givens factors, can all be accumulated in the unitary factor Q. In the light of Theorem 2 the careful reader will not be surprised by the shape of the chains of Givens rotations in the factorization of factors L and R where some of the Givens rotations are missing. Hence, using our pictorial representations we can exemplify the case $n = 6, k = 3$, $N = n + k = 9$, as follows



where the central matrix can be expressed as

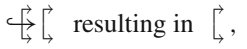


and the $\cdot$ represent zeros. These zeros are obtained summing the contribution of the $k \times k$ principal blocks of Q and of TZ^H which sums up to zero. We have used the fact that

$$Q = \left[\begin{array}{c|c} I_k & \\ \hline & \hat{Q} \end{array}\right] D = \mathcal{G}_{k+1} \cdots \mathcal{G}_{N-1} \hat{D},$$

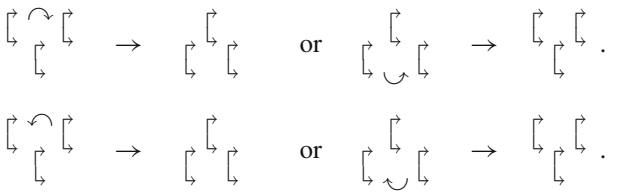
wherte $\mathcal{G}_i$ are Givens matrices acting on rows $i, i+1$ and $\hat{D}$ is a unitary diagonal matrix. Furthermore, in the lower left corner of the Schur parametrization of L we have trivial Givens rotations since $X(n+1:N,:) = -I_k$. The description of the bulge chasing algorithm in Sect. 4.1 will make it clear why this structure is not modified.

Givens transformations can also interact with each other by means of the *fusion* or the *turnover* operations (see [41], pp.112–115). The fusion operation will be depicted as follows:



and consists of the concatenation of two Givens transformations acting on the same rows. The turnover operation allows to rearrange the order of some Givens transformations (see [41]).

Graphically we will depict this rearrangement of Givens transformations as follows:



When we apply a sequence of r consecutive turnover operations we will use the same symbol surmounted by the number of turnovers such as $\overset{r}{\curvearrowright}$.

Each fusion and turnover has a constant cost since consists in the operations involving 2×2 or 3×3 matrices. Note that while the fusion between two Givens rotations can result in a trivial rotation, this is never the case when we perform a turnover between three non-trivial rotations.

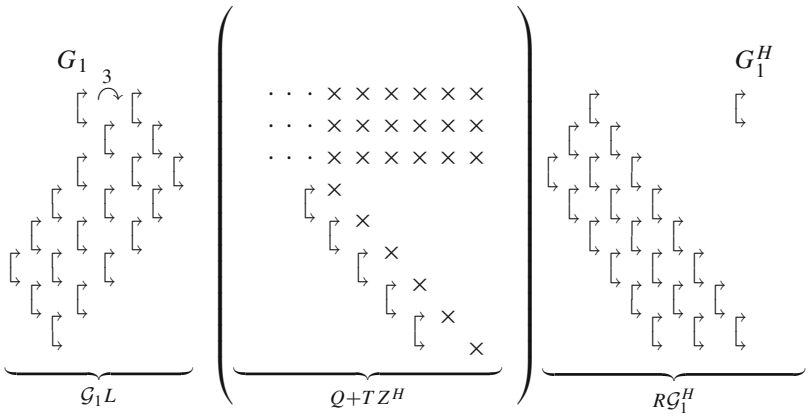
4.1 Initialization and bulge chasing

As observed in Remark 3 we do not have to perform the Givens transformations on the rank k part since the matrix Z can be recovered at the end of the QR process and the matrix T is not affected by the transformations which act on rows $k+1$ to N. As we will explain in Sect. 5 we prefer to store explicitly the vectors Z rather then recovering them at the end of the process because in this way we are able to prove a tighter bound for the backward error of the method.

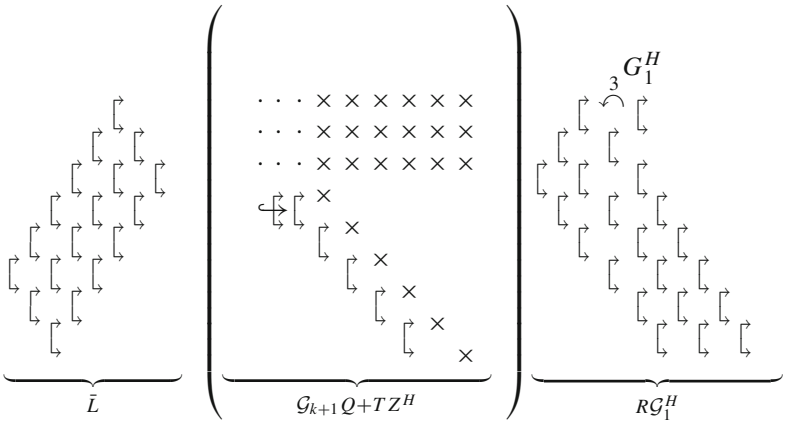
The implicit QR algorithm starts with the computation of the shift. Using a Wilkinson shift strategy we need to reconstruct the 2×2 lower-right hand corner of $\hat{A}$. This can be done by operating on the representation and it requires $O(k)$ flops. Once the shift μ is computed, we retrieve the first two components of the first column of $\hat{A}$, i.e., $\hat{a}_{11}, \hat{a}_{21}$ and we compute the 2×2 Givens rotation G_1 such that

$$G_1 \begin{bmatrix} \hat{a}_{11} - \mu \\ \hat{a}_{21} \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}.$$

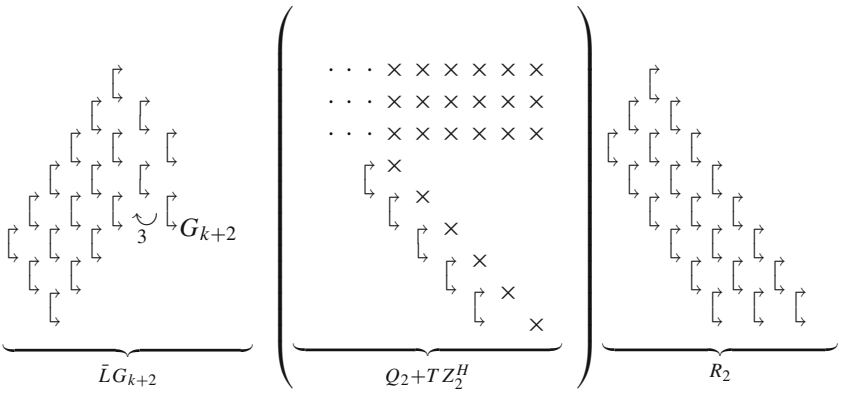
Let $\mathcal{G}_1 = G_1 \oplus I_{N-2}$, we have that matrix $\mathcal{G}_1 \hat{A} \mathcal{G}_1^H$ becomes



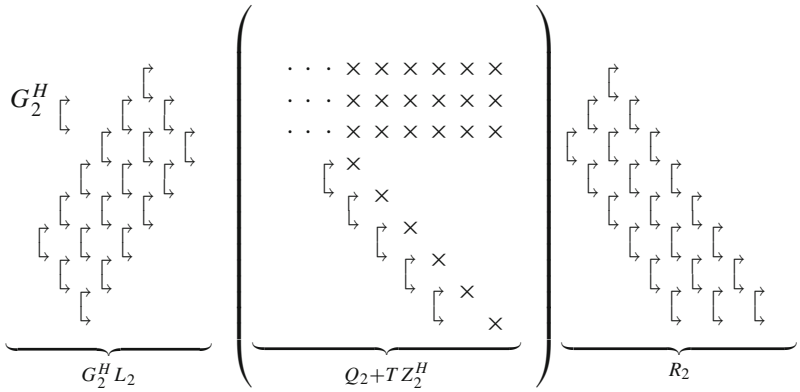
Applying a series of k turnovers operations we can pass G_1 through the ascending sequence of Givens transformations, and a new Givens transformation G_{k+1} acting on rows $k+1$ and $k+2$, will appear before the bracket, and then is fused with the first nontrivial rotation of Q.



Similarly the Givens rotation G_1^H on the right is shifted through the sequence of Givens transformations representing R and applied to the columns of Z^H and on the right of $\mathcal{G}_{k+1} Q$. Then another turnover operation is applied giving

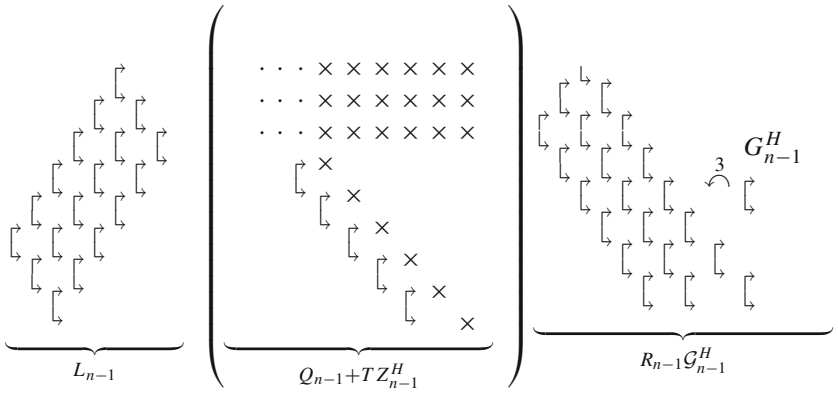


At the end of the initialization phase the Givens rotation G_{k+2} on the right of $\bar{L}$ can be brought on the left giving rise to the bulge represented by a Givens rotation G_2 acting on rows 2 and 3, namely $\bar{L}G_{k+2} = G_2^H L_2$. We have



The bulge needs to be chased down.

At this point we have $\mathcal{G}_1 \hat{A} \mathcal{G}_1^H = \mathcal{G}_2^H L_2 (Q_2 + T Z_2^H) R_2$, where $\mathcal{G}_2 = 1 \oplus G_2 \oplus I_{N-3}$. Performing a similarity transformation to get rid of $\mathcal{G}_2^H$, we have that the matrix G_2^H on the right can be brought to the left applying turnover operations. Repeating the same reasoning $n-1$ times, we have finally to remove a Givens rotation acting on columns $n-1$ and n.



With the application of k turnover operations, we get that $R_{n-1}\mathcal{G}_{n-1}^H = \mathcal{G}_{n+k-1}R_n$, where $\mathcal{G}_{n+k-1} = I_{N-2} \oplus G_{N-1}$. The Givens rotation G_{N-1} acts on the last two columns and will modify the last two columns of Z_{n-1}^H and then fuses with matrix Q_{n-1}. At this point the Hessenberg structure is restored, and the implicit step ends.

The graphical representation of the algorithm corresponds to the following updating of the matrices involved in the representation for suitable P, S, V

$$L^{(1)} = P^H L S, \quad Q^{(1)} = S^H Q V, \quad T^{(1)} = S^H T,$$
$$Z^{(1)} = V^H Z, \quad \text{and} \quad R^{(1)} = V^H R P.$$

In particular in P are gathered the $n-1$ rotations needed to restore the Hessenberg structure of the full matrix, so that there are no operations involving the last k rows of L, meaning that we can assume $P_2 = P(n+1:N, n+1:N) = I_k$. S is the product of the Givens rotations that have shifted through the factor L when turnover operations are performed, and similarly V is the product of the Givens matrices shifted through R from the right.

To show that this corresponds actually to a QR step it is sufficient to verify that we are under the hypothesis of Theorem 3, i.e., that $L^{(1)}$ and $R^{(1)}$ are unitary k-Hessenberg matrices, $T^{(1)}$ is still of the form $T^{(1)} = [T_k^T, 0]^T$ and that $Z^{(1)}$ has the structure described in point 2 of Theorem 3. From the description of the algorithm we can see that the matrices S and V are block diagonal with the leading block of size k equal to the identity matrix since the turnover operations shift down of k rows the rotations acting on rows and columns of L and R respectively. We note that at the end of the chasing steps the k-Hessenberg structure of $L^{(1)}$ and $R^{(1)}$ is restored, and $T^{(1)} = [T_k^H, 0]^T$ because $S(1:k, 1:k) = I_k$. Moreover, $L^{(1)}$ is still proper since the turnover operations cannot introduce trivial rotations. Matrix $Q^{(1)}$ is still block-diagonal with the leading block $k \times k$ unitary diagonal, and the tailing block with Hessenberg structure. For $Z^{(1)} = V^H Z$ we need to prove that $Z^{(1)H} = L^{(1)}(n+1:N, :)Q^{(1)}$. From (3.6), and observing that $P_2 = I_k$ we have $L^{(1)}(n+1:N, :) = L(n+1:N, :)S$. Substituting we get

$$Z^{(1)H} = Z^H V = L(n+1:N,:)QV = L(n+1:N,:)SS^H QV$$
$$= L^{(1)}(n+1:N,:)Q^{(1)} \tag{4.1}$$

as required. To apply the implicit Q-Theorem we need to observe that the first column of A is only affected by the first rotation during the initialization step and is never changed after that.

4.2 Computational cost

The reduction of a generic matrix to Hessenberg form requires in the general case $O(n^3)$ flops, but as we observed in the introduction, in special cases the reduction can be achieved with $O(n^2k)$ operations. In [10] is proposed an $O(n^2k)$ algorithm to reduce a unitary-plus-low-rank matrix to Hessenberg form and obtain directly the LFR factorization suitable as starting point of the QR method we just described in this paper. The algorithm in [10] can be applied when

1. $A = D + UV^H$, $U, V \in \mathbb{C}^{n\times k}$, and D is unitary block diagonal with block size $k < n$.
2. $A = H + [I_k, 0]^T Z^H$, $Z \in \mathbb{C}^{n\times k}$, and H is unitary block upper Hessenberg with block size $k < n$;
3. $A = G + [I_k, 0]^T Z^H$, $Z \in \mathbb{C}^{n\times k}$, and G is unitary block CMV with block size $k < n$;

These three cases cover the most interesting structures of low-rank perturbation of unitary matrices. In the general case of unitary matrices, where the spectral factorization of the unitary part is not known, in general we cannot expect to recover the eigenvalues even of the unitary part in $o(n^3)$.

Unitary matrices are always diagonalizable, so we fall in case (1) if we know the eigen-decomposition of the unitary part. Block companion matrices belong to case (2), and applying the algorithm in [10] to reduce them in Hessenberg form we get directly the factored representation.

We assume hence that A is already in Hessenberg form and we know that the embedding preserves this structure. If we are not in cases (1)–(3) it is necessary to compute the matrices required for embedding A in $\hat{A}$ which can be performed using $O(n^2k)$ operations. Similarly the cost of the representation is $O(n^2k)$ operations, since we need to compute $O(nk)$ Givens rotations and apply them to $N \times N$ matrices.

The key ingredients of the algorithm are turnover or fusion operations. Each of such operations can be performed with a bounded number of operations since they involve only 2×2 or 3×3 matrices. Each QR step consists of an inizialization phase, requiring $3k+1$ turnovers and a fusion as well as the updating of two rows of the $n \times k$ matrix Z in the case we choose to update it at each step. Each of the remaining $n-2$ chasing steps consists of $2k+1$ turnovers and the possible update of Z. In the final step we have k turnovers and a fusion with the last Givens rotation in Q. Overall the cost of an implicit QR step is $O(nk)$, and assuming as usual that deflation happens after a constant number of steps, we get an overall cost of $O(n^2k)$ arithmetic operations for retrieving all the eigenvalues. Comparing the cost of this algorithm with the cost of

the unstructured QR method, which requires $O(n^2)$ flops per iteration and then a total cost of $O(n^3)$, shows the advantage of using this approach.

4.3 Deflation

Deflation techniques are based on equation (3.5) which shows that the possibility of performing deflation can be recognized by direct inspection on the representation without reconstructing the matrix A. In practice it is equivalent to check the subdiagonal entries of the factor Q.

Lemma 4 *Assume that the QR iteration applied to the matrix $\hat{A}$ is convergent to an upper triangular matrix. Denote by $\hat{A}^{(s)} = L^{(s)}(Q^{(s)} + TZ^{(s)H})R^{(s)}$ the matrix obtained after s steps of the QR algorithm. Then, for $i = 1, \ldots, n-1$, we have $\lim_{s\to\infty} q^{(s)}_{i+k+1,i+k} = 0$, and moreover, for any prescribed tolerance τ, if s is such that $|q^{(s)}_{i+k+1,i+k}| < \tau K$, then $|a^{(s)}_{i+1,i}| < \tau$.*

Proof From relation (3.5) we have that

$$|q^{(s)}_{i+k+1,i+k}| = |a^{(s)}_{i+1,i}| \frac{\left|\bar{l}^{(s)}_{i+1,i+k+1}\right|}{|r^{(s)}_{i+k,i}|},$$

where $a^{(s)}_{i,j}$ are the entries of the matrix $A^{(s)}$ defined according to Theorem 4. Using Corollary 1 and the convergence of the QR algorithm we have $\lim_{s\to\infty} q^{(s)}_{i+k+1,i+k} = 0$.

From Corollary 1 we have $|r^{(s)}_{i+k,i}| \leq 1$ and $\left|\bar{l}^{(s)}_{i+1,i+k+1}\right| \geq K$. Hence

$$|a^{(s)}_{i+1,i}| = |q^{(s)}_{i+k+1,i+k}| \frac{|r^{(s)}_{i+k,i}|}{\left|\bar{l}^{(s)}_{i+1,i+k+1}\right|} \leq \tau.$$

□

Remark 4 Lemma 4 suggests to use as deflation criteria the condition $|q^{(s)}_{i+k+1,i+k}| < \varepsilon K$, where ε is the machine precision. The value of K as described in Corollary 1 can be computed as $1/|\det(T_k)|$ for the upper triangular matrix T_k given in the proof of Theorem 2. Note that, as we represent the matrix Q in terms of Givens rotations $\begin{bmatrix} c & s \\ -s & \bar{c} \end{bmatrix}$, with $s \in \mathbb{R}$, $s \geq 0$, we can simply check when a sine value is smaller than εK. When this condition is satisfied we replace the corresponding Givens transformation with the 2×2 unitary diagonal matrix $D_j = \begin{bmatrix} c/|c| & \\ & \bar{c}/|c| \end{bmatrix}$. In [37] it is shown, in a more general setting, that the eigenvalues of the matrix obtained by replacing a Givens transformation by a 2×2 identity matrix are accurate approximations of the original ones when the Givens rotation is close to the identity. This is a consequence of the Bauer-Fike Theorem. Applying the same idea to our framework it is immediate to see

that the absolute error introduced in a single eigenvalue is bounded by $\kappa_2 \|G_j - D_j\|_2$ where κ_2 is the condition number of the eigenvector matrix (assuming to work with diagonalizable matrices) and $G_j = \begin{bmatrix} c & s \\ -s & \bar{c} \end{bmatrix}$, $j = i + k + 1$ is the Givens rotation such that $|q_{i+k+1,i+k}| = s \leq \varepsilon K$. Moreover we can bound $\|G_j - D_j\|_2$ by $\sqrt{2}\varepsilon K$.

5 Backward error analysis

In this section we bound the backward error of the shifted QR algorithm presented in Sect. 4. The algorithm basically can be described by the following steps:

1. *Preliminary phase*: the input unitary-plus-low-rank Hessenberg matrix A is embedded into a larger Hessenberg matrix $\hat{A}$;
2. *Initialization phase*: a compressed LFR-type representation of $\hat{A}$ is computed, $\hat{A} = L^{(0)} F^{(0)} R^{(0)}$;
3. *Iterative phase*: at each step h, given the representation of the matrix $\hat{A}^{(h)} = L^{(h)} F^{(h)} R^{(h)}$ we perform a shifted QR iteration with the proposed algorithm, and return $A^{(h+1)} = L^{(h+1)} F^{(h+1)} R^{(h+1)}$.

As suggested in the introduction the initialization phase can be determined in several different ways depending on the additional features of the input matrix. In this section to be consistent with the approach pursued in the previous sections we only consider the case where the representation is found by a sequence of QR factorizations.

Concerning the preliminary phase we notice that as in the proof of Theorem 1 we can first compute the economy size QR decomposition of the full rank matrix Y. If we set $Y = QR$ and then rename the components of $A = U + XY^H$ as follows $U \leftarrow U$, $X \leftarrow XR^H$ and $Y \leftarrow Q$ we find that $Y^H Y = I_k$. In this way the embedding is performed at a negligible cost by introducing a small error perturbation of order $\tilde{\gamma}_k \|A\|_2$ where $\tilde{\gamma}_k = ck\varepsilon/(1 - ck\varepsilon)$ [34] and c and ε denote a small integer constant and the machine precision, respectively. For the sake of simplicity it is assumed that $\|X\|_2 = \|X\|_2 \|Y\|_2 \approx \|A\|_2$, and we refer to the notation used in [34].

5.1 Backward stability of the representation

In this section we prove that our representation is backward stable. The ingredients of the representation essentially are: the k-lower Hessenberg matrix L, the upper Hessenberg matrix Q, the $k \times k$ upper triangular matrix T, matrix Z^H and the k-upper Hessenberg matrix R. In particular, given un upper Hessenberg matrix $\hat{A} = \hat{U} + \hat{X}\hat{Y}^H$ we would like to show that the exact representation of $\hat{A} = L(Q + TZ^H)R$ differs from the computed one $\tilde{A} = \tilde{L}(\tilde{Q} + \tilde{T}\tilde{Z}^H)\tilde{R}$ by an amount proportional to $\|\hat{A}\|_2 \approx \|A\|_2$ and to the machine precision ε.

The computation proceeds by the following steps.

– Computation of T and L. We note that L and T, $T = [T_k^T; 0]^T$ in exact arithmetics are respectively the Q and the R factors of the QR factorization of matrix $\hat{X}$. From Theorem 19.4 of [34] and consequent considerations there exists a perturbation

Δ_X such that

$$(\hat{X} + \Delta_X) = \tilde{L}\tilde{T}, \tag{5.1}$$

where $\|\Delta_X(:, j)\|_2 \leq \sqrt{k}\,\tilde{\gamma}_{Nk}\|\hat{X}(:, j)\|_2$, $\tilde{L} = L + \Delta_L$, $\|\Delta_L(:, j)\|_2 \leq \tilde{\gamma}_{Nk}$.

– Computation of Q. The computed matrix $\tilde{Q}$ is obtained starting from matrix $\tilde{B} = \tilde{L}^H(\hat{U} + \Delta_U) \doteq L^H\hat{U} + \Delta_L{}^H\hat{U} + L^H\Delta_U$, and $\|\Delta_U\|_2 \lesssim \tilde{\gamma}_{N^2}$, which derives from the backward analysis of the product of two unitary factors. The computed factor is

$$\tilde{Q} = \left[\begin{array}{c|c} I_k & \\ \hline & \tilde{Q}_1 \end{array}\right],$$

where Q_1 is obtained applying the QR factorization to $\tilde{B}(k + 1 : N, 1 : n)$. Reasoning similarly as done before we have

$$\tilde{B}(k + 1 : N, 1 : n) + \Delta_B^{(1)} = \tilde{Q}_1\tilde{R}_1,$$

where $\|\Delta^{(1)}_{B(:,j)}\|_2 \leq \tilde{\gamma}_{nk}$, and $\tilde{Q} = Q + \Delta_Q$ with $\|\Delta_Q(:, j)\|_2 \leq \tilde{\gamma}_{nk}$.

– Computation of R. Similarly the computed $\tilde{R}$ is such that $\tilde{B} + \Delta_B^{(2)} = \tilde{Q}\tilde{R}$, where $||\Delta_B^{(1)}\|_2$ is small and $\tilde{R} = R + \Delta_R$ with $\|\Delta_R(:, j)\|_2 \leq \tilde{\gamma}_{Nk}$.

– Computation of Z^H. As we have seen we can perform QR iterations without computing Z explicitly because we can retrieve the matrix Z at convergence using formula 3.7. However for the stability of the all process we prefer to work with explicit Z, updating it at each step. This will affect the computational cost of a factor $O(nk)$ at each step but results in a more stable method (see Fig. 2 and the related discussion).

Note that in exact arithmetic $Z = RY$. Since the product by a small perturbation of a unitary matrix is backward stable, we get

$$\tilde{Z} = \tilde{R}(Y + \Delta Y), \quad \text{where} \quad \|\Delta Y\|_2 \lesssim \tilde{\gamma}_{N^2}.$$

We can conclude that

$$\tilde{A} = \tilde{L}(\tilde{Q} + \tilde{T}\tilde{Z}^H)\tilde{R} = \tilde{L}(\tilde{B} + \Delta_B^{(2)}) + \tilde{L}\tilde{T}\tilde{Z}^H\tilde{R} \tag{5.2}$$

$$= \tilde{L}(\tilde{L}^H(\hat{U} + \Delta_U) + \Delta_B^{(2)}) + (\hat{X} + \Delta_X)(\hat{Y} + \Delta_Y)^H\tilde{R}^H\tilde{R}. \tag{5.3}$$

Since $\tilde{L}\tilde{L}^H = I + \Delta_1$, and $\tilde{R}^H\tilde{R} = I + \Delta_2$ with $\|\Delta_1\|_2$ and $\|\Delta_2\|_2$ bounded by a constant times $\tilde{\gamma}_{N^2}$, we have

$$\tilde{A} \doteq \hat{U} + \hat{X}\hat{Y}^H + E = \hat{A} + E$$

where $E = \Delta_1\hat{U} + \Delta_U + L\Delta_B^{(2)} + \hat{X}\hat{Y}^H\Delta_2 + \Delta_X\hat{Y}^H + \hat{X}\Delta_Y^H$, and $\|E\|_2$ is bounded by a constant times $\|\hat{X}\|_2\|\hat{Y}\|_2 \approx \|A\|_2$.

5.2 Backward stability of the implicit QR steps

Matrix $A^{(1)}$, computed by a QR step applied to matrix $\hat{A}$, is such that $A^{(1)} = P^H \hat{A} P$ with P unitary. In this section we want to show that there exists a perturbation matrix E_A such that the computed matrix $\tilde{A}^{(1)} = P^H(\hat{A}+E_A)P$ where $\|E_A\|_2$ is proportional to $\|A\|_2$ and to the machine precision ε.

Let $\hat{A} = L(Q + TZ^H)R$ be the representation of $\hat{A}$ in floating point arithmetic (note that for not overloading the notation we dropped the superscripts). Similarly the new representation of the matrix $A^{(1)} = P^H \hat{A} P = P^H L(Q + TZ^H)R\,P$ is $A^{(1)} = L^{(1)}(Q^{(1)} + T^{(1)}Z^{(1)H})R^{(1)}$, where

$$L^{(1)} = P^H L S, \quad Q^{(1)} = S^H Q V, \quad T^{(1)} = S^H T,$$
$$Z^{(1)} = V^H Z, \quad \text{and} \quad R^{(1)} = V^H R P,$$

and the unitary matrices S and V are those originating from the turnovers on the Givens rotations composing L and R.

Theorem 5 *After one step of the implicit QR method where the operations are performed in floating point arithmetic we have*

$$\tilde{L}^{(1)} = P^H(L + E_L)S, \quad \tilde{Q}^{(1)} = S^H(Q + E_Q)V, \quad T^{(1)} = S^H T,$$
$$Z^{(1)} = V^H(Z + E_Z), \quad \textit{and} \quad R^{(1)} = V^H(R + E_R)P,$$

where $\|E_L\|_2, \|E_Q\|_2, \|E_R\|_2$ are bounded by a small multiple of the machine precision ε, while $\|E_Z\|_2$ is a bounded by $\tilde{\gamma}_N$, where $\tilde{\gamma}_N = cN\varepsilon/(1 - cN\varepsilon)$, and c is a small constant.

Proof The backward analysis of the error in the unitary factors, L, R and Q is similar to the one performed in Theorem 7.1 in [7]. To prove that the backward error in $T^{(1)}$ is zero we note that, because of the structure of S and of T, the product $S^H T$ is in practice never computed since, $S^H T = T$. The computation of $Z^{(1)}$ is the result of the product of a unitary factor and the rectangular matrix Z whose columns are orthonormal. For this reason $\|E_Z\|_2$ is bounded by $\tilde{\gamma}_N \|Y\|_2 = \tilde{\gamma}_N$. □

Summarizing we obtain the following result.

Theorem 6 *Let $\tilde{A}^{(1)}$ be the result computed in floating point arithmetic of a QR step applied to the matrix $\hat{A}$. Then there exists a perturbation Δ_A such that $\tilde{A}^{(1)} = P^H(\hat{A}+\Delta_A)P$, and $\|\Delta_A\|_2 \le \tilde{\gamma}_{N^2}\|\hat{A}\|_2$, where $\tilde{\gamma}_{N^2} = cN^2\varepsilon/(1 - cN^2\varepsilon)$, and c is a small constant.*

Proof The proof follows easily by using the results proved in Theorem 5 by assembling together the contributions of each error in the factors of A. □

6 Numerical experiments

We tested our algorithm on several classes of matrices. The purpose of the experimentation is to show that the algorithm is indeed backward stable as proved in Sect. 5 and

Table 1 In the upper part of the table scalar polynomials whose roots are known

#	Description, roots	Degree
1	Wilkinson polynomial $1, 2, \ldots n$	n
2	Scaled and shifted Wilkinson polynomial $-2.1, -1.9, \ldots, 1.7$	n
3	Reverse Wilkinson polynomial $1, 1/2, \ldots, 1/n$	n
4	Prescribed roots $2^{-m}, 2^{-(m-1)}, \ldots, 2^m$	$2m+1$
5	Prescribed roots shifted $2^{-m} - 3, \ldots, 2^m - 3$	$2m+1$
6	Chebyshev polynomial $\cos\left(\frac{(2j-1)\pi}{2n}\right)$	n
7	$\sum_{i=0}^{n} x^i \quad \cos\left(\frac{2\pi j}{n+1}\right) + i \sin\left(\frac{2\pi j}{n+1}\right)$	n
8	Bernoulli polynomial –	n
9	$p_1(z) = 1 + (m/(m+1) + (m+1)/m)\, z^m + z^{2m}$ –	$2m$
10	$p_2(z) = \frac{1}{m}\left(\sum_{j=0}^{m-1}(m+j)z^j + (m+1)z^m + \sum_{j=0}^{m-1} z^{2m-j}\right)$ –	$2m$
11	$p_3(z) = (1-\lambda)z^{m+1} - (\lambda-1)z^m + (\lambda+1)z + (1-\lambda)\,,\ \lambda = 0.999$ –	$m+1$

These polynomials have also been tested in [7,17], at the bottom of the table polynomials with particular structures, tested also in [12,35]

to confirm that the computation of all the eigenvalues by our method requires $O(n^2k)$ operations as proved theoretically.

The test suite consists of the following:

- Companion matrices associated with scalar polynomials whose roots are known (see description in Table 1).
- Companion matrices associated with scalar polynomials whose roots are unknown (see description in Table 1).
- Random fellow matrices [41] with a prescribed norm.
- Block companion matrices associated with matrix polynomials from the NLEVP collection [8].
- Random unitary plus rank-k matrices.
- Random unitary-diagonal-plus-rank-k matrices.
- Fiedler penta-diagonal companion matrices [24,38]. The associated polynomials are the scalar polynomials in Table 1 and we have then associated the same reference number.

In [20] the backward stability of polynomial rootfinding using Fiedler matrices different from the companion matrix is analyzed. The analysis shows that, when some coefficients of the monic polynomial are large, it is preferable to use the standard companion form. However, when the coefficients can be bounded by a moderate number, the algorithms using Fielder matrices are as backward stable as the ones using the companion. The purpose of our experiments with Fiedler pentadiagonal matrices is not to suggest to use these matrices for polynomial rootfinding, but to show that the backward stability of the proposed method is not affected by a larger k since for Fiedler pentadiagonal matrices we have $k = n/2$.

The algorithm is implemented in Matlab and the code is available upon request. In order to check the accuracy of the output we compare the computed approximations

with the actual eigenvalues of the matrix, in the case these are known. Otherwise we consider the values returned by the internal Matlab function `eig` applied to the initial matrix A already in Hessenberg form and with the balancing option on. Specifically, we match the two lists of approximations and then find the average error as

$$\mathrm{fw_{err}} = \frac{1}{n}\sum_{j=1}^{n} err_j,$$

where err_j is the relative error in the computation of the jth eigenvalue. The eigenvalues are retrieved reconstructing the matrix at convergence and taking the diagonal entries. Note that unitary factor in F has reduced to a unitary diagonal matrix, plus the the rank k part confined in the first k rows.

To validate the results provided in Sect. 5 we show the behavior of the backward error on the computed Schur form. Let P be the accumulated unitary similarity transformation obtained applying steps of the implicit QR algorithm as described in Sect. 4 to the augmented matrix $\hat{A}$. Because the last k rows of $\hat{A}$ are null according to Theorem 4, P is block diagonal

$$P = \begin{bmatrix} P_1 & \\ & P_2 \end{bmatrix},$$

where $P_1 \in \mathbb{C}^{n\times n}$, $P_2 \in \mathbb{C}^{k\times k}$ are unitary matrices. We can set $P_2 = I_k$ since no rotations act on the last k rows of the enlarged matrix. Assume that m is the number of iterations needed to reach convergence and that $\tilde{A}^{(m)}$ is the matrix reconstructed from the computed factors $\tilde{L}_m, \tilde{F}_m, \tilde{R}_m$ produced by performing m steps of the implicit algorithm. Not to overload the notation we denote with the same symbol $\tilde{A}^{(m)}$ its $n \times n$ leading principal submatrix. As in [17] we consider as a measure of the backward stability the relative error

$$\mathrm{bw_{err}}(A) = \frac{\|P_1^T A P_1 - \tilde{A}^{(m)}\|_\infty}{\|A\|_\infty}. \tag{6.1}$$

In order to compare the stability of our algorithm with that of algorithm tailored for polynomial rootfinding [7,15] we computed also the backward error in terms of the coefficient of the polynomial. In particular, let $p(x) = \sum_{i=0}^{n} p_i x^i = \prod_{i=1}^{n}(x-\lambda_i)$ be our monic test polynomial with roots λ_i, and denote by $\tilde{\lambda}_i$ the computed roots obtained with our algorithm applied to the companion matrix A. We denote by $\hat{p}(x)$ the polynomial having $\tilde{\lambda}_i$ as exact roots, i.e., $\hat{p}(x) = \prod(x-\tilde{\lambda}_i)$. Using the extended precision arithmetic of Matlab we computed the coefficients $\hat{p}_i$ of $\hat{p}(x)$ in the monomial basis. We define the backward error in terms of the coefficients of the polynomial as follows

$$\mathrm{bw_{err}}(p) = \max_i \frac{|p_i - \hat{p}_i|}{\|x\|_\infty}, \tag{6.2}$$

$x = (1, p+n-1, \ldots, p_0)$.

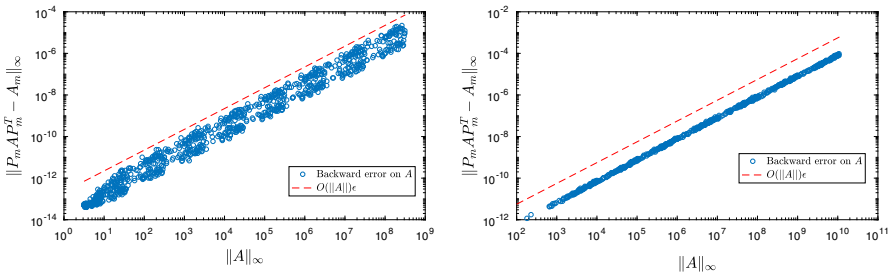


Fig. 1 Absolute backward error in the computation of the eigenvalues respect to the norm of the matrix. On the left the results obtained from thousand random unitary-plus-rank-5 matrices of size 50 that were generated as explained in Theorem 3. On the right the absolute backward error is plotted against the norm of the matrix for one thousand unitary diagonal-plus-rank-5 matrices of size 100. The dashed lines represent a reference line for the theoretical backward stability

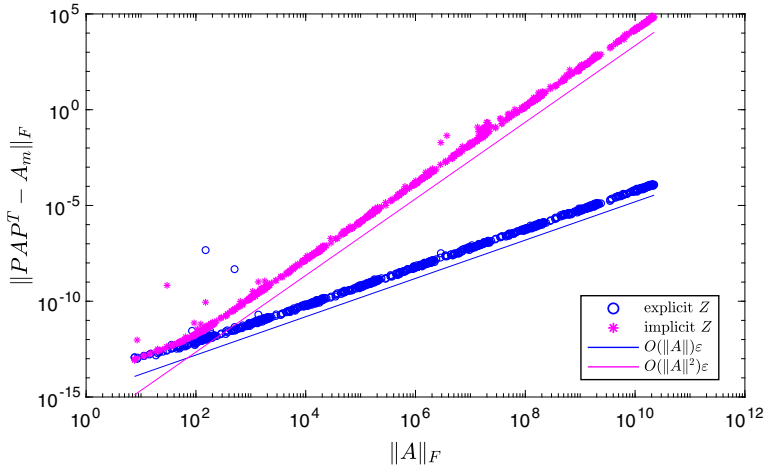


Fig. 2 Comparison on the backward errors of the algorithm with an explicit and implicit Z. We plotted the absolute backward error respect to the norm of the matrix. The results are obtained from a thousand random matrix polynomials of degree 10 where the coefficients are 5×5 matrices, whose norms range from 1 to 10^9. The solid lines represent a reference lines to show that in the case Z is explicitly computed the absolute backward error behaves as $O(\|A\|)\varepsilon$ while, keeping Z implicit, the backward error increases as $O(\|A\|^2)\varepsilon$

To confirm experimentally the stability of the algorithm we measured the backward error for matrices with prescribed norms. In particular, in Fig. 1 for matrices in the class i.e., generic unitary-plus-rank-5, and unitary diagonal-plus-rank-5, we report the results obtained on one thousand matrices of size 50 with norm ranging from 1 to 10^{13}, and we plot the absolute backward error $\|P_1^T A P_1 - \tilde{A}^{(m)}\|_\infty$ versus $\|A\|_\infty$. The dashed lines represent a slope proportional to $\epsilon\|A\|_\infty$ and as we can see the plots agree with the results proved in Sect. 5.

In Fig. 2, for matrix polynomials, we compare backwards stability of the algorithm using implicit Z or updating explicitly Z at each iteration. We observe that the absolute backward error behaves as $\varepsilon\|A\|$ when Z is updated at each iterations, while it b ehaves as $\varepsilon\|A\|^2$ when Z is retrieved only at the end of the computations. This shows in a

Table 2 Results on scalar companion matrices from Table 1

#	n	$\|A\|_\infty$	$\text{bw}_{\text{err}}(A)$	$\text{bw}_{\text{err}}(p)$	AMVW	BEGG	ZHSEQR
1	10	$1.93e{+}07$	$1.68e{-}15$	$6.31e{-}15$	$5.12e{-}15$	$1.55e{-}15$	$4.11e{-}16$
1	15	$9.62e{+}12$	$1.00e{-}15$	$8.90e{-}15$	$3.96e{-}15$	$3.45e{-}15$	$1.33e{-}15$
1	20	$2.28e{+}19$	$2.03e{-}15$	$5.28e{-}14$	$1.04e{-}14$	$3.35e{-}01$	$4.47e{-}15$
2	20	$6.69e{+}02$	$1.55e{-}15$	$1.36e{-}14$	$8.21e{-}16$	$2.17e{-}15$	$1.09e{-}15$
3	20	$1.03e{+}01$	$3.58e{-}15$	$8.08e{-}15$	$2.23e{-}15$	$2.93e{-}14$	$1.24e{-}15$
4	20	$1.93e{+}14$	$1.55e{-}15$	$4.98e{-}14$	$2.70e{-}15$	$4.32e{-}14$	$4.44e{-}16$
5	20	$1.33e{+}18$	$1.44e{-}15$	$4.41e{-}14$	$2.72e{-}14$	$5.42e{-}01$	$4.38e{-}15$
6	20	$2.02e{+}01$	$1.63e{-}15$	$1.70e{-}14$	$1.54e{-}15$	$1.79e{-}14$	$2.52e{-}15$
7	20	$6.32e{+}00$	$3.41e{-}15$	$1.81e{-}14$	$2.00e{-}15$	$2.67e{-}14$	$2.85e{-}15$
8	20	$6.76e{+}04$	$1.86e{-}15$	$2.50e{-}14$	$2.17e{-}15$	$1.70e{-}14$	$4.56e{-}15$
9	40	$6.71e{+}00$	$7.98e{-}15$	$1.87e{-}13$	$9.95e{-}17$	$7.07e{-}01$	$3.52e{+}13$
10	40	$1.14e{+}01$	$5.00e{-}15$	$3.10e{-}14$	$4.99e{-}15$	$3.92e{-}14$	$6.11e{-}15$
10	20	$7.99e{+}00$	$2.89e{-}15$	$1.27e{-}14$	$2.94e{-}15$	$1.96e{-}14$	$4.85e{-}15$
11	31	$2.83e{+}03$	$1.91e{-}15$	$4.64e{-}13$	$4.89e{-}15$	$2.43e{-}13$	$1.03e{-}14$

We see that the backward error is always very small, the forward error (with is not shown) is dependent on the conditioning of the problem. In the last three column we report [7] the backward error in terms of the polynomial coefficients of the unbalanced, single shifted version of the algorithm AMVW [7], of BEGG [15], and the LAPACK routine ZHSEQR

Table 3 Results on the NLEVP collection. On the top part of the table results for quadratic problems with $k = n/2$

Name	n	k	Degree	$\|\text{ceig(A)}\|_\infty$	$\|A\|_\infty$	forw_{err}	back_{err}
Acousticwave1d	20	10	2	5.96e+01	1.37e+01	1.42e−15	1.02e−14
Bicycle	4	2	2	5.70e+02	9.62e+03	2.60e−15	8.29e−16
Cdplayer	120	60	2	4.50e+03	2.67e+07	5.17e−16	5.85e−15
Closedloop	4	2	2	9.00e+00	3.00e+00	8.99e−16	1.42e−15
Dirac	160	80	2	2.11e+03	1.38e+03	5.24e−14	1.39e−13
Hospital	48	24	2	4.49e+01	1.11e+04	7.84e−13	2.57e−14
Metalstrip	18	9	2	1.71e+02	3.48e+02	7.78e−16	2.42e−15
Omnicam1	18	9	2	5.04e+15	1.73e+05	4.03e−07	2.60e−15
Omnicam2	30	15	2	4.66e+17	6.22e+07	1.16e−02	4.90e−15
Powerplant	24	8	2	1.72e+05	3.73e+07	7.13e−08	2.68e−15
qep2	6	3	2	1.80e+16	4.00e+00	3.31e−09	3.65e−16
Sign1	162	81	2	3.29e+09	1.53e+01	4.10e−09	5.84e−14
Sign2	162	81	2	9.61e+02	5.63e+01	4.27e−13	3.54e−14
Spring	10	5	2	2.33e+00	8.23e+01	3.00e−16	1.93e−15
Wiresaw1	20	10	2	1.57e+01	1.42e+03	6.00e−14	4.20e−15
Butterfly	240	64	4	2.97e+01	5.18e+01	5.15e−14	1.29e−13
Orrsommerfeld	40	10	4	1.88e+06	9.67e+00	1.83e−14	6.35e−15
Plasmadrift	384	128	3	6.64e+04	3.24e+02	1.02e−13	4.86e−14

On the bottom matrix polynomials with degree greater than 2

very clear way that it is better to update the rank-k part at each iteration. In order to explain these discrepancies theoretically we recall that at the beginning of our error analysis in the previous section we assume that the matrix $\hat{A}$ is upper Hessenberg. However the actual matrix obtained at the end of the Hessenberg reduction process only satisfies this requirement up to a backward error of order $\varepsilon\|A\|$. The different behavior of the explicit and the implicit algorithm depends on the propagation of this error. Specifically we can show that in the explicit variant the error propagates additively whereas in the implicit counterpart the error increases by a factor of order $\|A\|$. Similar error bounds have appeared in [5] where a backward stable method for eigenvalues and eigenvectors approximation of matrix polynomials is proposed. The algorithm is a variant of Francis's implicitly shifted QR algorithm applied on the companion pencil, and the rank correction is not explicitly computed but it is computed only once at the end of the computation when retrieving the eigenvalues. The authors of [5] proved that on the unscaled pencil (S, T) the computed Schur form is the exact Schur form of a perturbed pencil $(S + \delta_S, T + \delta_T)$, where $\|\delta_S\| \leq \varepsilon\|S\|^2$ and $\|\delta_T\| \leq \varepsilon\|T\|^2$. Working with the pencil they are able to remove the dependence from the norm by scaling the pencil. In our case it is not possible to scale A without destroying the unitary plus low rank structure, but we prove that the absolute error is $O(\|A\|)\varepsilon$ keeping Z explicit. In specific cases as for polynomial rootfinding where the Hessenberg structure of $\hat{A}$ is determined exactly we achieve very good results also when keeping Z implicit. In Table 2 we report the backward errors in the scalar polynomials described in Table 1. We report both the backward error in terms of the matrix coefficients and of the coefficients of the polynomial and we see that the tests confirm the backward stability of the algorithm. Edelman and Murakami [21] proved that the analysis of the backward error in terms of the polynomial coefficients might introduce an additional factor proportional to $\|A\|$ but Table 2 revels that we do better than expected because $\mathrm{bw}_{\mathrm{err}}(p) = \varepsilon\, O(\|A\|)$, and not $\varepsilon\, O(\|A\|^2)$. We report also the values of $\mathrm{bw}_{\mathrm{err}}(p)$ obtained on the same tests by two specialized algorithms for polynomial rootfinding, namely AMVW [7] and BEGG [15] and by ZHSEQR, the LAPACK routine for computing the eigenvalues of a Hessenberg matrix without any further structure. We obtain better results than those one gets using BEGG method, but sometimes we lose a digit of precision compared to AMVW. We think that this is mostly due to differences in shift and in deflation criteria or in the retrieving, in hight precision, the coefficients of the polynomial $\hat{p}(x)$ from the computed roots. Our method provides a unified framework to treat a larger class of matrices that contains companions and block companions but also perturbations of CMV shapes, or unitary diagonal plus low rank, and so on. See [27] for some real world applications different from scalar/matrix polynomials computation.

Table 3 reports the results obtained for several problems form the NLEVP collection [8], which contains polynomial eigenvalue problems from real-life applications. To apply our method we needed to invert the coefficient corresponding to the higher degree of the polynomial so not all the problems of the collection were suitable for our algorithm. In the collection we find mainly quadratic polynomial and a few examples of polynomial of degree ≥ 3. Table 3 reportsthe degree d of the polynomials, the size k of the coefficients, and $n = kd$ that is the size of the matrix of the linearization. We cannot compare directly with the method proposed in [5] since the authors of that

Table 4 Top: Random polynomials of low degree with different norm sizes

n	k	Degree	$\|A\|_\infty$	$\|\text{ceig(A)}\|_\infty$	forw_{err}	back_{err}
50	2	25	2.46e+01	1.77e+01	3.63e−15	1.10e−14
50	2	25	1.42e+06	2.85e+01	2.65e−12	9.37e−15
50	5	10	2.34e+01	2.32e+01	2.26e−15	8.11e−15
50	5	10	2.23e+06	3.86e+01	7.73e−12	8.31e−15
50	10	5	2.75e+01	3.24e+01	1.78e−15	9.25e−15
50	10	5	3.16e+06	3.90e+01	1.08e−11	7.78e−15
100	5	20	1.02e+06	3.43e+01	7.05e−13	9.88e−15
200	5	40	1.96e+06	3.73e+01	2.62e−13	1.93e−14
400	5	80	3.90e+06	1.01e+02	1.44e−13	3.19e−14
750	5	130	7.09e+06	9.79e+01	5.59e−13	5.30e−14
1000	5	200	9.57e+07	2.60e+04	8.49e−11	7.53e−14

We see that, in agreement with the theoretical results, the relative backward error is not affected by the norm of the matrix. Bottom: Random polynomials with higher degree and moderately high norm. We see that also in the larger example the stability is not compromised. The figures for the larger tests are the average over 10 runs

paper work on a pencil (S, T) and then were able to scale each matrix of the pencil by a factor $\alpha = \max(\|S\|, \|T\|)$ to remove the dependence of the error on the norm. In principle the algorithm BEGG [15], based on quasiseparable representation as well as other methods based on Givens weight and Givens vector representation, can be extended to deal with these matrices but with a cost of order at least $O(n^2k^3)$ which is not competitive for $k = O(n)$. The results of our algorithm for higher degree random matrix polynomials are reported in Table 4 where also the forward and backward errors for different values of the norm of the coefficients of the polynomials are shown. Each line refers to the average value over 50 tests on generalized companion matrices associated to matrix polynomials of size k and degree $d = n/k$. For each pair (k, d) we performed experiments varying the norm of the resulting generalized companion matrix. We see that as expected, for matrices with larger norm, we may have a loss of accuracy in the computed solutions.

Figure 3 shows, for a set of random matrices, that the cost of the algorithm is linear in k and quadratic in n. Tables 5, 6, and 7, contain the results for random unitary-plus-low-

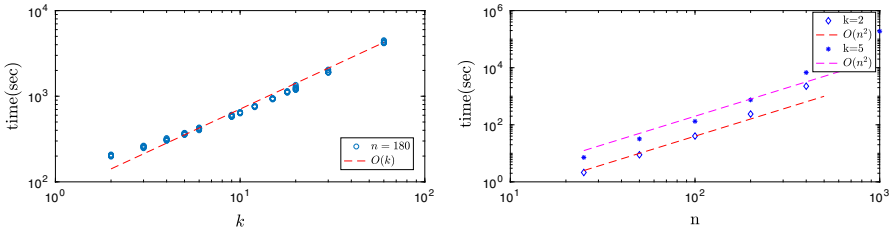


Fig. 3 On the right the double logarithmic plot for random matrices of size 180 that are unitary-plus-rank-k with k ranging from 1 to 60. The reference line shows the linear dependence on k. On the right, for $k = 2$ and $k = 5$ and matrices of size ranging from 25 to 1000. The dashed lines represent the $O(n^2)$ slope

Table 5 Unitary plus low rank random matrices, with different sizes, rank of the correction and norm of the matrix

n	k	$\|A\|_\infty$	$\|\text{ceig(A)}\|_\infty$	forw_{err}	back_{err}
50	1	7.14e+00	4.19e+00	7.33e−15	9.45e−15
50	1	9.70e+04	3.52e+01	1.70e−16	3.10e−15
50	2	7.19e+00	3.78e+00	7.51e−15	9.92e−15
50	2	9.83e+04	3.27e+01	1.97e−16	2.91e−15
50	25	8.27e+00	1.00e+15	5.36e−15	1.10e−14
50	25	9.98e+04	7.85e+14	1.95e−16	3.15e−15
100	1	1.00e+01	1.37e+01	1.46e−14	1.79e−14
100	1	9.85e+04	2.04e+02	2.46e−16	4.86e−15
100	2	1.01e+01	1.97e+01	1.52e−14	1.89e−14
100	2	9.91e+04	1.80e+02	2.60e−16	4.78e−15
100	25	1.10e+01	2.18e+07	1.30e−14	2.07e−14
100	25	9.99e+04	1.79e+06	3.30e−16	5.21e−15

For each n and k we tested two cases $\|A\|_\infty = O(1)$ and $\|A\|_\infty = O(10^4)$. Each result reported is the average over 50 random tests

Table 6 Unitary diagonal plus low rank random matrices, with different rank of the correction and norm of the matrix

n	k	$\|A\|_\infty$	$\|\text{ceig(A)}\|_\infty$	forw_{err}	back_{err}
50	1	3.46e+01	2.25e+00	2.66e−16	2.78e−15
50	1	3.34e+06	2.25e+00	3.11e−17	2.32e−15
50	2	5.96e+01	1.40e+01	1.57e−16	2.78e−15
50	2	5.86e+06	2.18e+01	8.83e−14	2.63e−15
50	25	6.37e+02	5.59e+01	7.22e−17	2.58e−15
50	25	6.36e+07	9.34e+01	8.63e−13	2.21e−15

For each n and k we tested two cases $\|A\|_\infty = O(1)$ and $\|A\|_\infty = O(10^4)$. Each result reported is the average over 50 random tests

rank matrices, for perturbed unitary diagonal matrices and for Fiedler pentadiagonal matrices. In all the cases, and independently on the matrix norm, we get very good results for the backward stability. Note that when the actual eigenvalues are unknown the results for the forward error show that the computed approximations agree with those returned by Matlab `eig` command.

7 Conclusions

In this paper we have presented a novel algorithm for eigenvalue computation of unitary-plus-low-rank Hessenberg matrices. The algorithm is computationally efficient with respect to both the size of the matrix and the size of the perturbation. Further, the algorithm is shown to be backward stable. At the core of the algorithm is a compressed data-sparse representation of the matrix as a product of three factors. The outermost factors are unitary generalized Hessenberg matrices whereas the factor in

Table 7 Results on Fiedler pentadiagonal matrices [38] associated to scalar polynomials

#	n	k	$\|\text{ceig}(A)\|_\infty$	$\|A\|_\infty$	forw_{err}	back_{err}
1	20	10	1.87e+21	2.30e+19	7.17e−01	2.88e−15
2	20	10	2.08e+04	9.48e+02	2.80e−01	3.68e−15
3	20	10	2.45e+15	1.52e+01	2.45e−01	4.39e−15
4	20	10	3.19e+15	2.67e+14	7.70e−02	5.71e−15
5	20	10	4.72e+20	1.66e+18	7.02e−02	2.17e−15
6	30	15	1.11e+18	1.05e+02	3.08e−01	1.31e−14
7	20	10	4.90e+00	5.18e+00	1.97e−15	7.31e−15
8	20	10	2.21e+15	7.10e+04	7.03e−11	5.16e−15
9	40	3	8.06e+15	4.04e+00	1.28e−03	1.02e−14
10	30	15	3.83e+01	1.06e+01	3.51e−15	1.08e−14
11	29	15	2.37e+00	4.00e+00	3.03e−15	1.88e−14

As proved in [18,19] the rank-correction for dense polynomials is in general $k = \lceil n/2 \rceil$ but it can be lower in the case the polynomial is sparse

the middle is a unitary upper Hessenberg matrix corrected by a low rank perturbation located in the first rows. In particular cases it is possible to obtain the data-sparse Hessenberg form with cost $O(n^2k)$ flops instead of the customary $O(n^3)$ flops. It is shown that deflation and convergence of the QR iteration can be checked directly from the representation by greatly simplifying the resulting fast scheme. Future work is concerned with the analysis of efficient procedures for computing the factored representation of the initial matrix as well as the design of a fast QZ iteration for matrix pencils.

Acknowledgements We would like to thank the anonymous reviewers whose helpful comments allowed to improve the quality of the presentation.

References

1. Ammar, G., Calvetti, D., Reichel, L.: Computing the poles of autoregressive models from the reflection coefficients. In: Proceedings of 31st Annual Allerton Conference on Communication, Control, and Computing, pp. 255–264 (1993)
2. Ammar, G., Gragg, W., Reichel, L.: Direct and inverse unitary eigenproblems in signal processing: an overview. In: De Moor, B.L.R., Moonen, F.T., Golub, G.H. (eds.) Linear Algebra for Large Scale and Real-Time Applications. Springer, New York (1993)
3. Ammar, G.S., Calvetti, D., Reichel, L.: Continuation methods for the computation of zeros of Szegö polynomials. Linear Algebra Appl. **249**, 125–155 (1996)
4. Ammar, G.S., Gragg, W.B., Reichel, L.: On the eigenproblem for orthogonal matrices. In: 1986 25th IEEE Conference on Decision and Control, pp. 1963–1966 (1986)
5. Aurentz, J., Mach, T., Robol, L., Vandebril, R., Watkins, D.S.: Fast and backward stable computation of the eigenvalues and eigenvectors of matrix polynomials. Math. Comput. **88**, 313–347 (2019)
6. Aurentz, J., Mach, T., Robol, L., Vandebril, R., Watkins, D.S.: Core-Chasing Algorithms for the Eigenvalue Problem. Fundamentals of Algorithms. SIAM, Philadelphia (2018)
7. Aurentz, J.L., Mach, T., Vandebril, R., Watkins, D.S.: Fast and backward stable computation of roots of polynomials. SIAM J Matrix Anal. Appl. **36**(3), 942–973 (2015)

8. Betcke, T., Higham, N.J., Mehrmann, V., Schröder, C., Tisseur, F.: NLEVP: a collection of nonlinear eigenvalue problems. ACM Trans. Math. Softw. **39**(2), 7:1–7:28 (2013)
9. Bevilacqua, R., Del Corso, G.M.: Structural properties of matrix unitary reduction to semiseparable form. Calcolo **41**(4), 177–202 (2004)
10. Bevilacqua, R., Del Corso, G.M., Gemignani, L.: On computing efficient data-sparse representations of unitary plus low-rank matrices. Technical report (2019)
11. Bindel, D., Chandresekaran, S., Demmel, J., Garmire, D., Gu, M.: A fast and stable nonsymmetric eigensolver for certain structured matrices. Technical report (2005)
12. Bini, D.A., Daddi, F., Gemignani, L.: On the shifted QR iteration applied to companion matrices. Electron. Trans. Numer. Anal. **18**(electronic), 137–152 (2004)
13. Bini, D.A., Eidelman, Y., Gemignani, L., Gohberg, I.: Fast QR eigenvalue algorithms for Hessenberg matrices which are rank-one perturbations of unitary matrices. SIAM J. Matrix Anal. Appl. **29**(2), 566–585 (2007)
14. Bini, D.A., Gemignani, L., Pan, V.Y.: Fast and stable QR eigenvalue algorithms for generalized companion matrices and secular equations. Numer. Math. **100**(3), 373–408 (2005)
15. Boito, P., Eidelman, Y., Gemignani, L., Gohberg, I.: Implicit QR with compression. Indagationes Mathematicae **23**(4), 733–761 (2012)
16. Bunse-Gerstner, A., Elsner, L.: Schur parameter pencils for the solution of the unitary eigenproblem. Linear Algebra Appl. **154**(156), 741–778 (1991)
17. Chandrasekaran, S., Gu, M., Xia, J., Zhu, J.: A fast QR algorithm for companion matrices. In: Ball, J.A., Eidelman, Y., Helton, J.W., Olshevsky, V., Rovnyak, J. (eds.) Recent Advances in Matrix and Operator Theory. Operator Theory: Advances and Applications, vol. 179, pp. 111–143. Birkhäuser, Basel (2007)
18. Del Corso, G.M., Poloni, F., Robol, L., Vandebril, R.: When is a matrix unitary or hermitian plus low rank? Numer. Linear Algebra Appl. (To appear) (2019)
19. Del Corso, G.M., Poloni, F., Robol, L., Vandebril, R.: Factoring block Fiedler companion matrices. Springer INdAM Ser. **30**, 129–155 (2019)
20. De Terán, F., Dopico, F.M., Pérez, J.: Backward stability of polynomial root-finding using Fiedler companion matrices. IMA J. Numer. Anal. **36**(1), 133–173 (2016)
21. Edelman, A., Murakami, H.: Polynomial roots from companion matrix eigenvalues. Math. Comput. **64**(210), 763–776 (1995)
22. Eidelman, Y., Gohberg, I., Haimovici, I.: Separable type representations of matrices and fast algorithms. In: Eigenvalue method, Volume 235 of Operator Theory: Advances and Applications, vol. 2, Birkhäuser/Springer, Basel (2014)
23. Fassbender, H.: On numerical methods for discrete least-squares approximation by trigonometric polynomials. Math. Comput. **66**(218), 719–741 (1997)
24. Fiedler, M.: A note on companion matrices. Linear Algebra Appl. **372**, 325–331 (2003)
25. Fiedler, M., Markham, T.L.: Completing a matrix when certain entries of its inverse are specified. Linear Algebra Appl. **74**, 225–237 (1986)
26. Francis, J.G.F.: The QR transformation-part 2. Comput. J. **4**(4), 332–345 (1962)
27. Fyodorov, Y.V., Sommers, H.-J.: Random matrices close to Hermitian or unitary: overview of methods and results. J. Phys. A Math. Gen. **36**(12), 3303–3347 (2003)
28. Gantmacher, F.R.: The Theory of Matrices. Number v. 1 in the Theory of Matrices. Chelsea Pub. Co. (1960)
29. Gemignani, L.: A unitary Hessenberg QR-based algorithm via semiseparable matrices. J. Comput. Appl. Math. **184**(2), 505–517 (2005)
30. Gemignani, L., Robol, L.: Fast Hessenberg reduction of some rank structured matrices. SIAM J. Matrix Anal. Appl. **38**(2), 574–598 (2017)
31. Golub, G.H., Van Loan, C.F.: Matrix Computations. Johns Hopkins Studies in the Mathematical Sciences, 3rd edn. Johns Hopkins University Press, Baltimore (1996)
32. Gragg, W.B.: The QR algorithm for unitary Hessenberg matrices. J. Comput. Appl. Math. **16**, 1–8 (1986)
33. Gragg, W.B.: Positive definite Toeplitz matrices, the Arnoldi process for isometric operators, and Gaussian quadrature on the unit circle. J. Comput. Appl. Math. **46**(1–2), 183–198 (1993). **(Computational complex analysis)**
34. Higham, N.J.: Accuracy and Stability of Numerical Algorithms, 2nd edn. Society for Industrial and Applied Mathematics (SIAM), Philadelphia (2002)

35. Jenkins, M.A., Traub, J.F.: Principles for testing polynomial zerofinding programs. ACM Trans. Math. Softw. **1**(1), 26–34 (1975)
36. Kimura, H.: Generalized Schwarz form and lattice-ladder realizations of digital filters. IEEE Trans. Circuits Syst. **32**(11), 1130–1139 (1985)
37. Mach, T., Vandebril, R.: On deflations in extended QR algorithms. SIAM J. Matrix Anal. Appl. **35**(2), 559–579 (2014)
38. Moler, C.: Fiedler Companion Matrix. Cleve's Corner (2013)
39. Sinap, A., Van Assche, W.: Orthogonal matrix polynomials and applications. In: Proceedings of the Sixth International Congress on Computational and Applied Mathematics (Leuven, 1994), vol. 66, pp. 27–52 (1996)
40. Vandebril, R., Del Corso, G.M.: An implicit multishift QR-algorithm for Hermitian plus low rank matrices. SIAM J. Sci. Comput. **32**(4), 2190–2212 (2010)
41. Vandebril, R., Van Barel, M., Mastronardi, N.: Matrix Computations and Semiseparable Matrices, vol. I, II. Johns Hopkins University Press, Baltimore (2008)
42. Watkins, D.S.: The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods, 1st edn. Society for Industrial and Applied Mathematics, Philadelphia (2007)