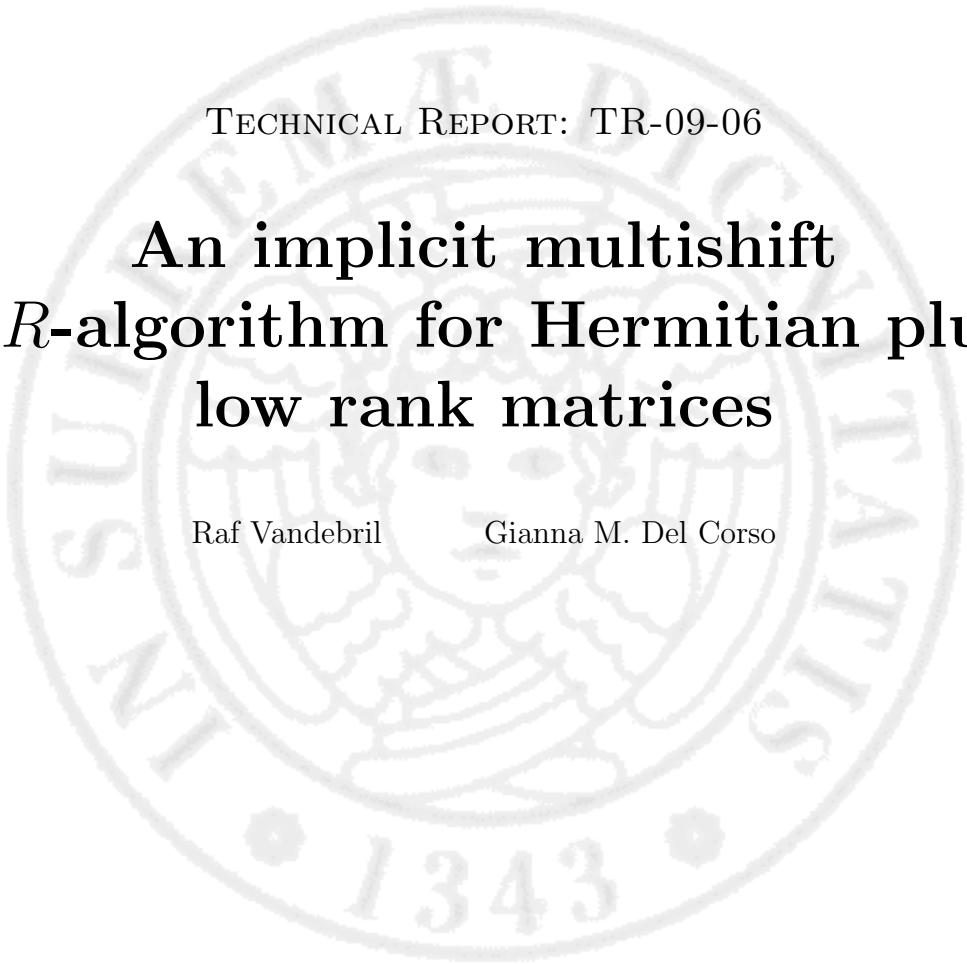
Università di Pisa
Dipartimento di Informatica



Technical Report: TR-09-06

An implicit multishift QR-algorithm for Hermitian plus low rank matrices

Raf Vandebril Gianna M. Del Corso

March 31, 2009

address: Largo B. Pontecorvo 3, 56127 Pisa, Italy. tel: +39 050 2212700 fax: +39 050 2212726

An implicit multishift QR-algorithm for Hermitian plus low rank matrices*

Raf Vandebril † Gianna M. Del Corso‡

March 31, 2009

Abstract

Hermitian plus possibly unhermitian low rank matrices can be efficiently reduced into Hessenberg form. The resulting Hessenberg matrix can still be written as the sum of a Hermitian plus low rank matrix.

In this paper we develop a new implicit multishift QR-algorithm for Hessenberg matrices, which are the sum of a Hermitian plus a possibly non-Hermitian low rank correction.

The proposed algorithm exploits both the symmetry and low rank structure to obtain a QR-step involving only $\mathcal{O}(n)$ floating point operations instead of the standard $\mathcal{O}(n^2)$ operations needed for performing a QR-step on a Hessenberg matrix. The algorithm is based on a suitable $\mathcal{O}(n)$ representation of the Hessenberg matrix. The low rank parts present in both the Hermitian and low rank part of the sum are compactly stored by a sequence of Givens transformations and few vectors.

Due to the new representation, we cannot apply classical deflation techniques for Hessenberg matrices. A new, efficient technique is developed to overcome this problem.

Some numerical experiments based on matrices arising in applications are performed. The experiments illustrate effectiveness and accuracy of both the QR-algorithm and the newly developed deflation technique.

1 Introduction

In the last few years many numerical techniques for computing eigenvalues of structured rank matrices have been proposed (e.g. [23, 3, 6] and the references therein). In particular the QR-algorithm received a great deal of this attention. Suitable representations such as quasiseparable, diagonal-subdiagonal or Givens-weight to represent the structured rank matrix are essential for the development of effective and efficient algorithms. A representation of the structured rank Hessenberg matrix in terms of $\mathcal{O}(n)$ parameters makes it possible to perform QR-steps in $\mathcal{O}(n)$ operations instead of $\mathcal{O}(n^2)$ operations required by standard methods.

This paper is concerned with the efficient computation of all the eigenvalues of an $n \times n$ Hermitian matrix perturbed by a possibly non-Hermitian low rank correction. This problem arises in different situations, for example when a perfectly Hermitian structure is corrupted by errors in only few rows or columns. Another typical situation in which this problem occurs is the eigenvalue computation of matrices with the sizes of the off-diagonal elements decaying rapidly away from the main diagonal. In this case, it can be computationally convenient to approximate, up to a desired accuracy, the original matrix by the sum of a Hermitian and a low rank matrix. In many other cases the problem to be solved is modeled naturally in this form (see the numerical

*The first author has a grant as "Postdoctoraal Onderzoeker" from the Fund for Scientific Research–Flanders (Belgium). This research was also partially supported by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office, Belgian Network DYSCO (Dynamical Systems, Control, and Optimization)

†KULeuven, Dept. of Computer Science, 3001 Leuven(Heverlee),Belgium, (raf.vandebril@cs.kuleuven.be)

‡Università di Pisa, Dept. of Computer Science, Largo Pontecorvo, 3, 56127 Pisa, Italy, (delcorso@di.unipi.it)

experiments). A typical example is the computation of roots of a polynomial expressed in a basis satisfying a three terms recurrence giving rise to a Hermitian tridiagonal matrix plus a rank-one correction. These matrices are referred to in the literature as comrade matrices [1].

More precisely, let $A = S + UV^H$, $A \in \mathbb{C}^{n\times n}$, where S is Hermitian $S = S^H$ and $U, V \in \mathbb{C}^{n\times m}$, with $m \ll n$. A standard approach to compute the eigenvalues of A consists in a preliminary reduction to Hessenberg form followed by the standard QR-algorithm. The reduction to Hessenberg form takes $\mathcal{O}(n^3)$ operations in general, but often one can take advantage of the rank structure present in the original matrix and reduce the cost to $\mathcal{O}(n^2)$ [10, 11, 16]. It can be easily seen that the resulting Hessenberg matrix H can still be expressed as the sum of a Hermitian and a low rank perturbation matrix.

In this article a new suitable $\mathcal{O}(n)$ representation based on Givens transformations of the Hessenberg matrix is first proposed. This representation allows performing implicit multishift QR-steps involving only $\mathcal{O}(n)$ operations for each iteration. Since the new representation makes deflation in the classical sense impossible a solution to this problem is also proposed.

In [9] also QR-algorithms for Hessenberg matrices which are the sum of symmetric plus low rank matrices were presented. Our article differs significantly from theirs. The authors in [9] use the quasiseparable representation, i.e. they store the low rank part by means of few vectors. This articles uses unitary 2×2 transformations for storing the low rank part. Since both representations are different the resulting QR-steps and implementations are also completely different. In [9] the standard deflation technique as present in Lapack is used. Using, however, the representation presented here, this is not possible and both the deflation technique as the successive QR-steps on submatrices need to be adapted. Similar numerical experiments were performed, making it possible to compare the two approaches.

The paper is organized as follows. Section 2 discusses some essential preliminary results such as the Givens-weight representation, the graphical schemes, similarity transforms exploiting the rank structure and information on the QR-algorithm. Section 3 introduces the structured rank representation for the Hessenberg matrix. Section 4 and 5 discuss respectively an implicit single shifted and multishifted QR-step. The modifications introduced in the algorithm for incorporating deflation are discussed in Section 6. Numerical experiments are given in Section 7.2. Finally some conclusions are presented.

2 Preliminary results

In this section we discuss preliminary results, essential for understanding this article and in particular we present the graphical schemes used for representing the QR-factorization of a Hessenberg matrix. Secondly, we discuss a similarity transformation of a Hermitian plus low rank matrix to a Hessenberg matrix, which is the sum of a Hermitian plus low rank matrix. It is shown in the next section that the number of parameters for representing this Hessenberg matrix is linear. For developing an implicit QR-algorithm, preservation of the structure is essential, this is discussed in Subsection 2.3, where the general scheme of implicit QR-iterations is given.

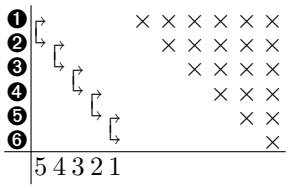
2.1 Working with Givens transformations

The algorithm presented in this paper involves matrices with a rank structure. These matrices can be represented with an adapted form of the Givens-weight representation [7, 20, 22].

Sequences of Givens transformations admit an easy understandable graphical representation. This representation will be used throughout the paper to show how the algorithm works. For efficient use of this representation, some extra tools are necessary such as the shift-through operation and the fusion. Let us first explain the Givens-weight representation by an example.

The following graphical scheme represents the QR-factorization of a 6×6 Hessenberg matrix

H, based on Givens transformations.



The scheme corresponds to the factorization $H = G_1 G_2 \dots G_5 R$. The matrix R is shown on the right by the $\times$ (assume them all different from zero) and is clearly upper triangular. The numbers on the vertical axis indicate the different rows of the matrix. The brackets with arrows depict Givens transformations. The arrows point to the rows the Givens transformations act on. The numbers on the bottom line represent a sort of timeline, describing the temporal order of application of each Givens transformation. In particular, the Givens transformation G_5 is located above 1 because is the first one to be applied to reproduce the Hessenberg matrix H. Applying G_5 on R results in a matrix having a nonzero element in position $(5, 4)$. Applying the transformations G_i gradually reestablishes the subdiagonal of the matrix H. This scheme is useful to represent the factorization of H as $G_1 G_2 \dots G_5 R$, because we can understand the precise action of the Givens transformations on R. Rewriting the formula $H = G_1 G_2 \dots G_5 R$ we have $G_5^H \dots G_1^H H = R$. Hence G_1^H was constructed such to annihilate the first subdiagonal element, G_2^H the second one and so forth.

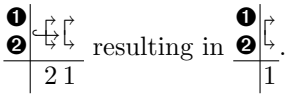
Givens transformations can also interact with each other.

Lemma 1 (Fusion) *Suppose two Givens transformations*[1] G_1 *and* G_2 *are given:*

$$G_1 = \begin{bmatrix} c_1 & -\bar{s}_1 \\ s_1 & \bar{c}_1 \end{bmatrix} \text{ and } G_2 = \begin{bmatrix} c_2 & -\bar{s}_2 \\ s_2 & \bar{c}_2 \end{bmatrix}.$$

Then we have that $G_1 G_2 = G_3$ *is again a Givens transformation. We will call this operation the fusion of Givens transformations in the remainder of the text.*

The proof is trivial. In our graphical schemes, we will depict this as follows:



The following lemma is very powerful and will give the possibility to interchange the order of Givens transformations and to obtain different patterns. Often Givens transformations of higher dimensions, say n, are considered. This means that the corresponding 2×2 Givens transformation is embedded in the identity matrix of dimension n, still acting only on two rows when applied to the left.

Lemma 2 (Shift-through) *Suppose three* 3×3 *Givens transformations* $\check{G}_1, \check{G}_2$ *and* $\check{G}_3$ *are given, such that the Givens transformations* $\check{G}_1$ *and* $\check{G}_3$ *act on the first two rows of a matrix, and* $\check{G}_2$ *acts on the second and third row (when applied on the left to a matrix).*

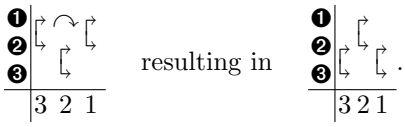
Then there exist three Givens transformations $\hat{G}_1, \hat{G}_2$ *and* $\hat{G}_3$ *such that*

$$\check{G}_1 \check{G}_2 \check{G}_3 = \hat{G}_1 \hat{G}_2 \hat{G}_3,$$

where $\hat{G}_1$ *and* $\hat{G}_3$ *work on the second and third row and* $\hat{G}_2$*, works on the first two rows.*

[1] In fact these transformations are rotations. More information on Givens rotations can be found in [2]. Throughout the entire manuscript we will assume to be working with Givens rotations. Nevertheless all results remain true when one considers 2×2 unitary transformations.

This result is well-known. The proof can be found in [21] and is based on the fact that one can factorize a 3×3 unitary matrix in different ways. Graphically we will depict this rearrangement of Givens transformations as follows.



There is also a variant in the other direction (from the right to the left scheme depicted by $\smile$). Important to remark is that the fusion of Givens transformations can be considered as a special case of the shift-through lemma.

The shift-through operation can also be applied repeatedly. This results in shift-through sequences. This is referred to as the shift-through operation of length ℓ.

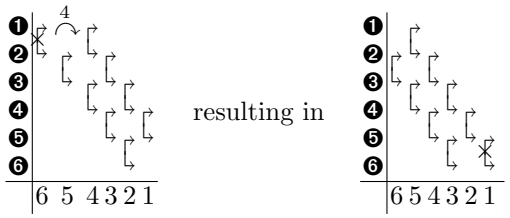
Lemma 3 *Suppose we have the following matrix product GWX, in which G denotes a Givens transformation acting on row 1 and 2. The matrices W and X are both unitary matrices consisting of a descending sequence of ℓ Givens transformations. This means that both W and X consist of ℓ successive Givens transformations. The i-th Givens transformation G_i^W of W acts on row $i+1$ and $i+2$. The i-th Givens transformation G_i^X of X acts on row i and $i+1$.*

The matrix product GWX can then be rewritten as

$$GWX = \hat{W}\hat{X}\hat{G},$$

where $\hat{G}$ is now a Givens transformation acting on row $\ell+1$ and $\ell+2$. The unitary matrices $\hat{W}$ and $\hat{X}$ are again descending sequences of ℓ Givens transformations.

Graphically, the situation described in the previous lemma is the following



where G is the first Givens acting on rows 1 and 2 in the first picture (marked with $\times$), W the leftmost chain of descending Givens, and X the rightmost chain of Givens. The symbol $\overset{\ell}{\curvearrowright}$ represents that after this operation, we will have a Givens transformation that will act on ℓ rows below the place where the initial Givens was acting. In this example, $\hat{G}$ will act on rows 5 and 6, because $\ell = 4$.

The operation depicted in the lemma is an operation "moving" a Givens rotation from top-left to the bottom-right. Similarly one can have shift-through operations of length ℓ going to the top right, top left and the bottom left. To clearly indicate where the Givens is going to and the number of single shift-trough operations, we use the symbols $\overset{\ell}{\curvearrowright}$, $\overset{\ell}{\smile}$, $\overset{\ell}{\smile}$ and $\overset{\ell}{\curvearrowleft}$.

2.2 Similarity transform to Hessenberg form

Applying a QR-method directly to a Hermitian plus rank m matrix is possible, but especially for large values of m, it is more efficient to first reduce the problem to Hessenberg form [8, 10, 4].

Suppose the initial matrix A is written as the sum of a Hermitian plus a rank m matrix, that is $A = S + UV^H$, where U and $V \in \mathbb{C}^{n \times m}$. A standard reduction of the matrix A to Hessenberg form costs $\mathcal{O}(n^3)$ operations, when one does not exploit the available low rank structure. In particular, if S is a band matrix with bandwidth b, the cost reduces to $\mathcal{O}((b+m)n^2)$ [8, 4], for transforming

the involved matrix A to Hessenberg form (see also [10, 11, 16]). The reduction to Hessenberg form preserves the eigenvalues since it is a unitary similarity transformation. This gives

$$\begin{aligned} H = Q^H AQ &= Q^H SQ + Q^H UV^H Q \\ &= \hat{S} + \hat{U}\hat{V}^H. \end{aligned}$$

Hence, the Hessenberg matrix H can be written as the sum of a Hermitian matrix $\hat{S}$ plus a rank m correction matrix $\hat{U}\hat{V}^H$. Note that, even if S was banded, $\hat{S}$ is a dense Hermitian matrix in general, but we will see that this matrix still admits an $\mathcal{O}(n)$ representation.

2.3 On the QR-algorithm

Let us denote the Hessenberg matrix we are working with as follows

$$H = S + UV^H, \tag{1}$$

with H Hessenberg, S Hermitian and U and V, two $n \times m$ matrices. We will denote by $\mathcal{F}_{n,m}$ (see [9]) the class of upper Hessenberg matrices in Equation 1, obtained as the sum of a $n \times n$ Hermitian matrix and a rank m perturbation.

As it is shown in [9] the class $\mathcal{F}_{n,m}$ is closed under QR-steps. This fact is essential for developing an efficient implicit QR-method exploiting the matrix structure. The multishift QR-algorithm, applied to matrix $H = H^{(0)} \in \mathcal{F}_{n,m}$ generates a sequence of unitarily similar matrices $\{H^{(k)}\}_k$ converging to the real canonical Schur form of H.

The multishift QR-method proceeds iteratively as follows[2], let $H^{(0)} = H$:

$$\begin{aligned} p_d^{(k)}(H^{(k)}) &= Q^{(k)} R^{(k)} \\ H^{(k+1)} &= {Q^{(k)}}^H H^{(k)} Q^{(k)}, \end{aligned} \tag{2}$$

where, for every k, $p_d^{(k)}(x)$ is a monic polynomial of degree d. For example, for $d = 1$ we have the single shift case and $p_1^{(k)}(H^{(k)}) = H^{(k)} - \mu^{(k)} I$ and for $d = 2$ we have the double-shift case, with $p_2^{(k)}(H^{(k)}) = (H^{(k)} - \mu_1^{(k)} I)(H^{(k)} - \mu_2^{(k)} I)$. The parameter $\mu^{(k)}$ for the single shift case, and parameters $\mu_1^{(k)}$ and $\mu_2^{(k)}$ for the double-shift are chosen in accordance with some shift strategy [24, 26, 17] to accelerate convergence.

Usually, instead of computing explicitly the factorization QR of the polynomial in (2), we proceed implicitly performing the transition $H^{(k)} \to H^{(k+1)}$ without even computing the products between the shifted matrices in (2) (see also [25]). The possibility of computing the iterations implicitly was firstly proposed in [13, 12]. Based on e.g. the Implicit Q-theorem [14] essential uniqueness of the matrix obtained is guaranteed at each step, once the first column of $Q^{(k)}$ has been formed.

The global flow of an implicit method is hence of the following form.

1. `Initialization step`: Determine the orthogonal transformation $\hat{Q}_{\mathcal{I}}$ (the subscript $\mathcal{I}$ refers to the initial step), such that

   $$\hat{Q}_{\mathcal{I}}^H p_d(H)\, \mathbf{e}_1 = \beta \quad \mathbf{e}_1, \beta = \|p_d(H)\mathbf{e}_1\|_2,$$

   where $p_d(H)$ is a monic polynomial of degree d, as described in (2) with suitable shifts chosen. Let $H_{\mathcal{I}} = \hat{Q}_{\mathcal{I}}^H H \hat{Q}_{\mathcal{I}}$, note that $H_{\mathcal{I}}$ is not in upper Hessenberg form since a bulge is created with the tip in position $(d+2, 1)$.

2. `Chasing steps`: Perform a unitary similarity reduction on $H_{\mathcal{I}}$ to bring it back to Hessenberg form (not altering the first column and row). Let $\hat{Q}_{\mathcal{C}}$ (the subscript $\mathcal{C}$ refers to the chasing procedure) be such that $\hat{H} = \hat{Q}_{\mathcal{C}}^H H_{\mathcal{I}} \hat{Q}_{\mathcal{C}} = \hat{Q}_{\mathcal{C}}^H \hat{Q}_{\mathcal{I}}^H H \hat{Q}_{\mathcal{I}} \hat{Q}_{\mathcal{C}}$ is in Hessenberg form. Set $\hat{Q} = \hat{Q}_{\mathcal{I}} \hat{Q}_{\mathcal{C}}$.

[2]The superscript notation $\cdot^{(i)}$ will be used only for depicting QR-steps performed on matrices. We will omit this superscript as much as possible, not to overload the notation.

It can be proved easily that $\hat{Q}$ and $\hat{H}$ are essentially identical to the matrices obtained by performing the explicit QR-algorithm. In this paper, for the sake of readability, we will consider only the single and double shift cases, that is $d = 1$ or $d = 2$, but the proposed algorithm works similarly for the general case with $d > 2$. In the single shift case, $Q_{\mathcal{I}}$ will consist of a single Givens rotation, because $p_1(H) = H - \mu I$ is still Hessenberg, while in the double shift case $p_2(H) = (H - \mu_1 I)(H - \mu_2 I)$ is a generalized Hessenberg and $Q_{\mathcal{I}}$ is obtained as the product of two Givens transformations. The size of the bulge created depends on the degree d and has $d(d+1)/2$ undesired elements. Finally, the orthogonal matrix $Q_{\mathcal{C}}$ in the chasing step is the cumulative product of $d\,(n-2)$ Givens factors.

3 A suitable $\mathcal{O}(n)$ representation

In [9] the matrices belonging to $\mathcal{F}_{n,m}$ are represented by using the diagonal and subdiagonal of H plus the two skinny matrices U and V. In this section we will deduce a representation of H based on Givens transformations similar to the one introduced for $m = 1$ in [19]. This representation allows us to store still $\mathcal{O}(n)$ elements, an has the advantage of being intuitive and stable as usual for representations based on Givens rotations (see for example [7, 18]).

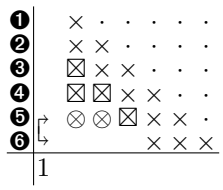
Since matrix H in (1) is the sum of a Hermitian and a low rank matrix, we know that the Hermitian matrix S has quite a large part of low rank structure. Consider $S = H - UV^H$. The Hessenberg matrix H has zeros below the subdiagonal. Hence, the matrix S needs to be of rank m below the subdiagonal. The matrix S has the following form, where the part marked with the $\boxtimes$ is coming form a rank m part.

$$S = \begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot \\ \boxtimes & \times & \times & \cdot & \cdot & \cdot \\ \boxtimes & \boxtimes & \times & \times & \cdot & \cdot \\ \boxtimes & \boxtimes & \boxtimes & \times & \times & \cdot \\ \boxtimes & \boxtimes & \boxtimes & \boxtimes & \times & \times \end{bmatrix}$$

The elements of the upper triangular part are denoted by dots. Since the matrix is Hermitian these entries are known once the lower triangular part of S is specified. We will immediately use the graphical representation. The matrix we start with is the matrix S. All subsequent graphical schemes represent factorizations of this matrix S.

3.1 A rank-1 perturbation

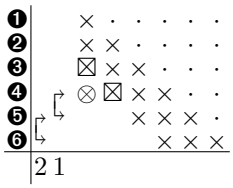
We consider the case, $m = 1$, hence, as we can see in the first scheme below, three elements (coming from a rank 1 part) in the last row of the matrix S are annihilated by a single Givens transformation acting on row 5 and 6.



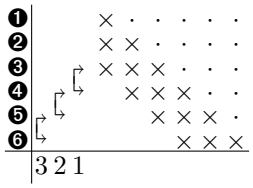
Mathematically, the scheme depicts $S = G_1 S_1$, where G_1 corresponds to the Givens transformation at time stamp 1, and S_1 is the matrix on the right having extra zeros in the last row. The scheme here presents a sort of factorization of the original Hermitian matrix S. In fact S_1 and G_1 are computed similarly as one computes a QR-factorization: $G_1^H S = S_1$, where G_1^H acts on S to create zeros, resulting in a matrix S_1.

The elements to be annihilated by the second Givens transformation are marked in the scheme with $\otimes$. In fact G_2 is computed to obtain $G_2^H G_1^H S = S_2$, where S_2 has $n-4$ zeros in row $n-1$. Rewriting $G_2^H G_1^H S = S_2$ gives us $S = G_1 G_2 S_2$ depicted in the following scheme. We see S_2 on

the right, Givens transformation G_1 above 2 and G_2 above 1. Considering $S = G_1 G_2 S_2$ we see that first G_2 needs to be re-applied to S_2 followed by G_1 to retrieve the original matrix S.



After the Givens transformation acting on row three and four is performed we have completely removed the low rank part. Removing the structured rank part introduces, however, a new subdiagonal. The matrix remaining on the right side is now a generalized Hessenberg matrix having two subdiagonals.



Let us write this factorization as $S = \tilde{V}\tilde{B}$, in which $\tilde{V}$ represents the Givens transformations on the left and the matrix $\tilde{B}$ represents the generalized Hessenberg matrix on the right. Reconsider now the original matrix H. Due to the equality between the low rank parts in the matrices $\mathbf{u}\mathbf{v}^H$ and S, we can factorize the vector $\mathbf{u}$ as $\tilde{V}\tilde{\mathbf{u}}$ where the vector $\tilde{\mathbf{u}}$ has only the first three elements different from zero. This means that we get:

$$H = S + \mathbf{u}\mathbf{v}^H = \tilde{V}\left(\tilde{B} + \tilde{\mathbf{u}}\mathbf{v}^H\right) \tag{3}$$

$$= \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot \\ & \times & \times & \times & \cdot & \cdot \\ & & \times & \times & \times & \cdot \\ & & & \times & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ \times \\ \times \\ \\ \\ \end{bmatrix} \mathbf{v}^H \right).$$

This representation was used in [19]. The disadvantage of this form is that it requires a quite complicated initial step. To simplify the QR-procedure we enlarge our matrix $\tilde{V}$ by adding two more Givens transformations to it, these transformations are constructed in such a way that they annihilate the second and third nonzero element in $\tilde{\mathbf{u}}$, hence $\mathbf{u} = V\hat{\mathbf{u}}$. The following scheme depicts the representation we will use. Since there is no correspondence between the rank structure of $\tilde{B}$ and $\tilde{\mathbf{u}}$ anymore, no additional zeros will be created in the matrix $\tilde{B}$.

$$H = V\left(B + \hat{\mathbf{u}}\mathbf{v}^H\right)$$

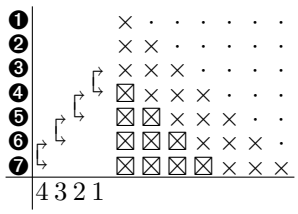
$$= \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot \\ & \times & \times & \times & \cdot & \cdot \\ & & \times & \times & \times & \cdot \\ & & & \times & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}^H \right)$$

For the ease of notation we will reuse the symbol $\mathbf{u}$, this means that in Section 4 and 5 we start from a factorization $H = V(B + \mathbf{u}\mathbf{v}^H)$ assuming $\mathbf{u}$ has only one element different from zero.

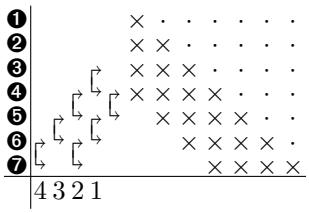
3.2 A rank-2 perturbation

We assume the perturbation to be of rank 2, the general case proceeds similarly. Decompose UV^H as $\mathbf{u}\mathbf{v}^H + \mathbf{x}\mathbf{y}^H$. The Hermitian matrix S is of the form $S = H - \mathbf{u}\mathbf{v}^H - \mathbf{x}\mathbf{y}^H$, having therefore the part below the subdiagonal of rank at most 2. Similarly as done for the rank 1 case, we can peal

off a rank 1 part. Now, this will not create zeros, because another part of rank 1 will still remain. Graphically a first sequence of Givens transformations transforms S to the following form, having the part marked with $\boxtimes$ still of rank 1. Without loss of generality we assume to be working on a 7×7 matrix.



We remark that this first sequence of Givens transformations also transforms the vector $\mathbf{u}$ into a vector with only the first three elements nonzero. To remove the remaining low rank part in the scheme above, another sequence of Givens transformations is needed. We get the following form.



This second sequence of Givens transformations will create also many zeros in the vector $\mathbf{x}$, all but the first 4 elements will become zero. This means that for the matrix H, we get the following graphical scheme.

$$H = \quad \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot \\ & \times & \times & \times & \times & \cdot & \cdot \\ & & \times & \times & \times & \times & \cdot \\ & & & \times & \times & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ \otimes_2 \\ \otimes_1 \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}^H \quad + \begin{bmatrix} \times \\ \times \\ \otimes_4 \\ \otimes_3 \\ \\ \\ \\ \end{bmatrix} \mathbf{y}^H \right)$$

Just like in the rank 1 case, we are not finished yet. The elements $\otimes$ in the vectors still need to be removed. The order of removal is indicated by their subscript. Bringing these Givens transformations outside the brackets gives us the following final representation.

$$H = \quad \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot \\ & \times & \times & \times & \times & \cdot & \cdot \\ & & \times & \times & \times & \times & \cdot \\ & & & \times & \times & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}^H \right) \tag{4}$$

This scheme corresponds to (for simplicity we reuse the symbols $\mathbf{u}$ and $\mathbf{x}$ depict the sparse vectors) $H = V^{(\mathbf{u})}\, V^{(\mathbf{x})} \left(B + \mathbf{u}\mathbf{v}^H + \mathbf{x}\mathbf{y}^H \right)$.

Remark 1 *We choose to present the construction of the sequences of Givens transformations $V^{(\mathbf{u})}$ and $V^{(\mathbf{x})}$ based on the matrix S, to show the appearance of the extra subdiagonals. In practice, however, they are entirely determined by the vectors $\mathbf{x}$ and $\mathbf{u}$ and one can immediately compute the full sequence of $n-1$ Givens transformations for $V^{(\mathbf{u})}$ and $n-2$ transformations for $V^{(\mathbf{x})}$ based on the two vectors $\mathbf{u}$ and $\mathbf{x}$.*

3.3 The generic rank $m > 2$ perturbation

Generalizing, if $H = S + UV^H$, with UV^H a rank m perturbation of a Hermitian matrix S, we can represent H as follows,

$$H = \prod_{i=1}^{m} V^{(\mathbf{u}_i)} \left(B + \sum_{i=1}^{m} \mathbf{u}_i \mathbf{v}_i^H \right),$$

where B is a generalized Hessenberg with $m+1$ subdiagonals and $\mathbf{u}_i$ are sparse vectors with only the first i entries different from zero, each $V^{(\mathbf{u}_i)}$ consists of $n-i$ Givens transformations. So, to store the representation for a Hessenberg matrix which is the sum of a Hermitian plus rank m perturbation we need to store:

- $\sum_{i=1}^{m}(n-i) = \mathcal{O}(nm+m^2)$ Givens transformations;
- $\sum_{i=1}^{m+2}(n+1-i) = \mathcal{O}(nm+m^2)$ entries for the matrix B;
- $\sum_{i=1}^{m} i = \mathcal{O}(m^2)$ entries for the vectors $\mathbf{u}_i$.

Based on this representation we will illustrate how to perform a single and double shift QR-step. We will discuss the single shift and multishift setting in two different sections, so the reader can follow more easily the algorithms. For simplicity we will restrict ourselves to the rank 2 case and the double shift case. The ideas presented are however easily generalizable to fit the more general case.

Remark 2 *In the upcoming Sections 4 and 5 we silently assume all Givens transformations in the sequences $V^{(\mathbf{u}_i)}$ to be different from the identity matrix. In this case all shift through operations are well defined, and we can run the algorithm to completion. We will come back to this in the beginning of Section 6 and discuss this in the Appendix.*

4 Single shift QR-step on higher order perturbations

In this section we explain how to perform an iteration of the single-shift QR-method, using the representation introduced in Section 3. We start with the single shift case to get the reader acquainted with the algorithm.

Let $H = H_1 = V_1^{(\mathbf{u})}V_1^{(\mathbf{x})}(B_1 + \mathbf{u}_1\mathbf{v}_1^H + \mathbf{x}_1\mathbf{y}_1^H)$. We know that for performing a single shift QR-step on a Hessenberg matrix $n-1$ Givens transformations are needed. We label the i-th of these $n-1$ Givens transformation as G_i. After this similarity transformation is performed we obtain a perturbed Hessenberg matrix $H_{i+1} = G_i^H H_i G_i$. The vectors and matrices used for representing H_i will inherit the same subscript as well as all intermediate variables with the subscript i, unless they are in final form for H_{i+1} in that case they obtain the subscript $i+1$. In particular at the end of the initialization step we will end up with matrix H_2 that is a perturbed Hessenberg with a bulge in position $(2,1)$, this is matrix $H_{\mathcal{I}}$ from Section 2.3.

4.1 Initialization

Since we are dealing with the single shift case, the initialization step consists of a single Givens transformation G_1 acting on rows 1 and 2 (corresponding to the matrix $\hat{Q}_{\mathcal{I}}$ in section 2.3), such that

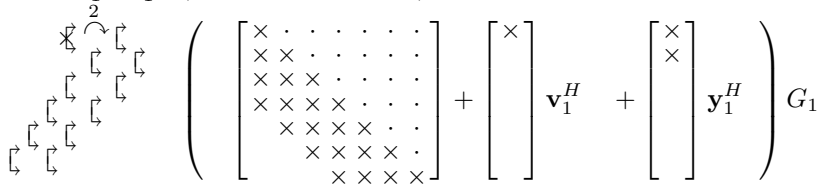
$$G_1^H(H-\mu I)\mathbf{e}_1 = \|(H-\mu I)\mathbf{e}_1\|_2\,\mathbf{e}_1.$$

Then H is transformed by a similarity transform using G_1 and $H_2 = G_1^H H_1 G_1$ is obtained which is not Hessenberg anymore. For simplicity we consider performing the left and right Givens transformation independently.

4.1.1 Transformation on the left

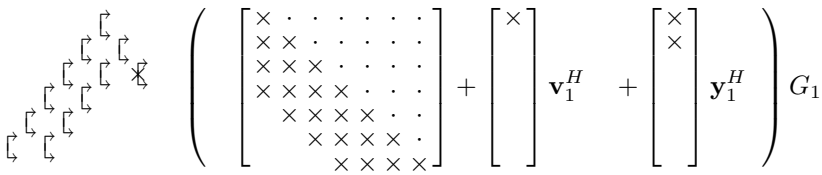
The Givens transformations marked with $\times$ in the scheme indicates the transformation G_1^H. We will start by removing this Givens transformation. We have the following situation.

$$H_2 = = G_1^H V_1^{(\mathbf{u})} V_1^{(\mathbf{x})} (B_1 + \mathbf{u}_1 \mathbf{v}_1^H + \mathbf{x}_1 \mathbf{y}_1^H) G_1$$



Performing a shift-through operation of length 2 to the bottom right: $G_1^H V_1^{(\mathbf{u})} V_1^{(\mathbf{x})} = \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} \tilde{G}_1^H$ results in the following. (The Givens transformation marked with $\times$ equals $\tilde{G}_1^H$.)

$$H_2 = \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} \tilde{G}_1^H (B_1 + \mathbf{u}_1 \mathbf{v}_1^H + \mathbf{x}_1 \mathbf{y}_1^H) G_1$$



Performing the transformation $\tilde{G}_1^H$ to the terms within the brackets gives us the following:

$$\begin{aligned} H_2 &= \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} (\tilde{G}_1^H B_1 + \tilde{G}_1^H \mathbf{u}_1 \mathbf{v}_1^H + \tilde{G}_1^H \mathbf{x}_1 \mathbf{y}_1^H) G_1 \\ &= \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} (\tilde{B}_1 + \mathbf{u}_1 \mathbf{v}_1^H + \mathbf{x}_1 \mathbf{y}_1^H) G_1 \\ &= \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} (\tilde{B}_1 + \mathbf{u}_2 \mathbf{v}_1^H + \mathbf{x}_2 \mathbf{y}_1^H) G_1. \end{aligned}$$

Due to the sparseness of the vectors $\mathbf{u}_1$ and $\mathbf{x}_1$ they don't change. Only the matrix B_1 is affected, but all the zeros in the lower left triangular corner remain intact. The vectors $\mathbf{u}_1$ and $\mathbf{x}_1$ are already in final form and hence are written as $\mathbf{u}_2$ and $\mathbf{x}_2$.

4.1.2 Transformation on the right

To conclude the initial step we need to remove the transformation G_1 depicted on the right. We remark that removing this Givens transformation is almost identical to the removal of the Givens transformations needed for the chasing as will be shown later on. Performing the transformation on the right gives us:

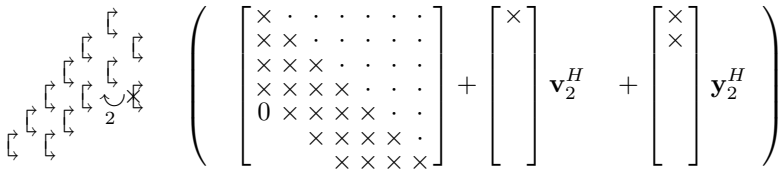
$$\begin{aligned} H_2 &= \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} (\tilde{B}_1 + \mathbf{u}_2 \mathbf{v}_1^H + \mathbf{x}_2 \mathbf{y}_1^H) G_1 \\ &= \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} (\tilde{B}_1 G_1 + \mathbf{u}_2 \mathbf{v}_1^H G_1 + \mathbf{x}_2 \mathbf{y}_1^H G_1) \\ &= \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} (\hat{B}_1 + \mathbf{u}_2 \mathbf{v}_2^H + \mathbf{x}_2 \mathbf{y}_2^H). \end{aligned}$$

The vectors $\mathbf{v}_2$ and $\mathbf{y}_2$ will not change anymore in this step. The matrix $\hat{B}_1 = \tilde{B}_1 G_1^H$ has lost one zero w.r.t. B_1, this undesired element is marked with $\otimes$ in the next scheme.

$$H_2 = \quad \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot \\ \otimes & \times & \times & \times & \times & \cdot & \cdot \\ & & \times & \times & \times & \times & \cdot \\ & & & \times & \times & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}_2^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}_2^H \right).$$

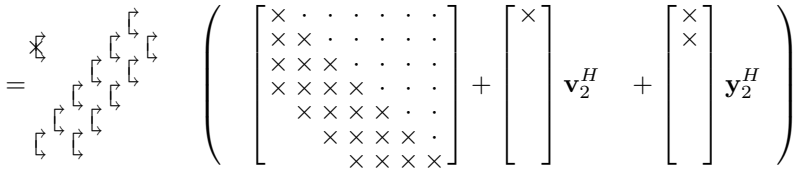
We will remove this unwanted nonzero element by a Givens transformation on the left. The upcoming shift-through operation is already depicted.

$$H_2 = \tilde{V}_1^{(\mathbf{u})}\tilde{V}_1^{(\mathbf{x})}\hat{G}_1^H(B_2 + \mathbf{u}_2\mathbf{v}_2^H + \mathbf{x}_2\mathbf{y}_2^H)$$



Performing a shift-through operation of length 2 and moving the Givens transformations $\hat{G}_1^H$ two positions up, to the top left position gives us the result of the initialization step.

$$H_2 = G_2 V_2^{(\mathbf{u})} V_2^{(\mathbf{x})}(B_2 + \mathbf{u}_2\mathbf{v}_2^H + \mathbf{x}_2\mathbf{y}_2^H)$$



The Givens transformation on the left G_2 will determine the next similarity transformation.

4.2 The Chasing

The global idea for the chasing, in order to move down a Givens transformation G_i on the left acting on rows i and $i+1$, is to apply first a similarity transformation annihilating the left transformation and creating one on the right acting on columns i and $i+1$. We then perform the transformation on the right to the matrices and drag it through so that it pops up on the left as a Givens transformation G_{i+1} acting on rows $i+1$ and $i+2$. Hence the result is a new transformation on the left, shifted down one row.

We will illustrate this step only briefly, since it is almost identical to the procedure described in the initialization step for removing the transformation on the right. Performing the similarity transformation determined by G_2 gives us:

$$\begin{aligned} H_3 = G_2^H H_2 G_2 &= V_2^{(\mathbf{u})} V_2^{(\mathbf{x})}(B_2 + \mathbf{u}_2\mathbf{v}_2^H + \mathbf{x}_2\mathbf{y}_2^H)G_2 \\ &= V_2^{(\mathbf{u})} V_2^{(\mathbf{x})}(\tilde{B}_2 + \mathbf{u}_3\mathbf{v}_3^H + \mathbf{x}_3\mathbf{y}_3^H), \end{aligned}$$

with $\mathbf{u}_2 = \mathbf{u}_3$ and $\mathbf{x}_2 = \mathbf{x}_3$. Performing the right Givens transformation creates a bulge in position $(6,2)$ in the matrix $\tilde{B}_2$. This bulge needs to be removed again by a Givens transformation, which needs to be brought to the top left by two shift-through operations.

$$\begin{aligned} H_3 &= V_2^{(\mathbf{u})} V_2^{(\mathbf{x})} \hat{G}_2(B_3 + \mathbf{u}_3\mathbf{v}_3^H + \mathbf{x}_3\mathbf{y}_3^H) \\ &= G_3 V_3^{(\mathbf{u})} V_3^{(\mathbf{x})}(B_3 + \mathbf{u}_3\mathbf{v}_3^H + \mathbf{x}_3\mathbf{y}_3^H) \end{aligned}$$

The resulting Givens transformation G_3 acts on rows 3 and 4 and has moved down one position w.r.t. G_2.

4.2.1 The last transformations

Unfortunately one cannot run the implicit algorithm fully to the end. Suppose we have the matrix H_{n-3}, and we perform the similarity transformation determined by Givens transformation G_{n-3}:

$H_{n-2} = G_{n-3}^H H_{n-3} G_{n-3}$. The following scheme depicts the matrix H_{n-2}.

$$H_{n-2} = G_{n-3}^H H_{n-3} G_{n-3} = V_{n-3}^{(\mathbf{u})} V_{n-3}^{(\mathbf{x})} (B_{n-3} + \mathbf{u}_{n-3}\mathbf{v}_{n-3}^H + \mathbf{x}_{n-3}\mathbf{y}_{n-3}^H) G_{n-3}$$

$$= \quad \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot \\ & \times & \times & \times & \times & \cdot & \cdot \\ & & \times & \times & \times & \times & \cdot \\ & & & \times & \times & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}_{n-3}^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}_{n-3}^H \right) G_{n-3}$$

Applying the Givens transformation on the right, which acts on columns $n-3$ and $n-2$ will not create a bulge in the matrix B_{n-3}. Hence, we cannot continue the chasing procedure. The algorithm breaks down in this case. To determine the final Givens transformation, we need to explicitly compute column $H_{n-3}\mathbf{e}_{n-2}$ and then $H_{n-2}\mathbf{e}_{n-1}$. Using the representation this can be achieved easily.

For completeness we will illustrate the idea on how to update the representation. Assume we have performed the similarity transformation with G_{n-3} on H_{n-3}. We obtain a matrix factorization of the following form (there is no Givens transformation G_{n-2} on the left now):

$$H_{n-2} = V_{n-2}^{(\mathbf{u})} V_{n-2}^{(\mathbf{x})} (B_{n-2} + \mathbf{u}_{n-2}\mathbf{v}_{n-2}^H + \mathbf{x}_{n-2}\mathbf{y}_{n-2}^H).$$

We know that two more Givens transformations need to be performed to complete the procedure, since $n-1$ Givens are needed to perform a QR-step on a Hessenberg matrix. Explicitly computing column $n-2$ of H_{n-2} reveals us the bulge which still needs to be removed. The Givens transformation for removing this bulge is G_{n-2}. Performing a similarity transformation on H_{n-2} with G_{n-2} leads to:

$$H_{n-1} = G_{n-2}^H V_{n-2}^{(\mathbf{u})} V_{n-2}^{(\mathbf{x})} (B_{n-2} + \mathbf{u}_{n-2}\mathbf{v}_{n-2}^H + \mathbf{x}_{n-2}\mathbf{y}_{n-2}^H) G_{n-2}$$

$$= \quad \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot \\ & \times & \times & \times & \times & \cdot & \cdot \\ & & \times & \times & \times & \times & \cdot \\ & & & \times & \times & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}_{n-2}^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}_{n-2}^H \right) G_{n-2}$$

For removing the transformation on the left, a shift-through operation is needed between G_{n-2} and the first sequence of Givens transformations $V_{n-2}^{(\mathbf{u})}$: $G_{n-2}^H V_{n-2}^{(\mathbf{u})} = V_{n-1}^{(\mathbf{u})} \tilde{G}_{n-2}^H$ (after this operation $V_{n-1}^{(\mathbf{u})}$ is in final form for this step, also $\mathbf{u}_{n-1} = \mathbf{u}_{n-2}$ and $\mathbf{x}_{n-1} = \mathbf{x}_{n-2}$ do not change):

$$H_{n-1} = V_{n-1}^{(\mathbf{u})} \tilde{G}_{n-2}^H V_{n-2}^{(\mathbf{x})} (B_{n-2} + \mathbf{u}_{n-1}\mathbf{v}_{n-2}^H + \mathbf{x}_{n-1}\mathbf{y}_{n-2}^H) G_{n-2}$$

$$= \quad \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot \\ & \times & \times & \times & \times & \cdot & \cdot \\ & & \times & \times & \times & \times & \cdot \\ & & & \times & \times & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}_{n-2}^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}_{n-2}^H \right) G_{n-2}.$$

The undesired transformation can be fused now with the last Givens transformation of the sequence $V_{n-2}^{(\mathbf{x})}$: $\tilde{G}_{n-2}^H V_{n-2}^{(\mathbf{x})} = V_{n-1}^{(\mathbf{x})}$. The Givens from the left is now removed without disturbing the representation of the matrix.

The transformation on the right can be applied without any problem. It does not perturb the structure of the matrices involved:

$$\begin{aligned} H_{n-1} &= V_{n-1}^{(\mathbf{u})} V_{n-1}^{(\mathbf{x})} (B_{n-2} + \mathbf{u}_{n-1}\mathbf{v}_{n-2}^H + \mathbf{x}_{n-1}\mathbf{y}_{n-2}^H) G_{n-1} \\ &= V_{n-1}^{(\mathbf{u})} V_{n-1}^{(\mathbf{x})} (B_{n-1} + \mathbf{u}_{n-1}\mathbf{v}_{n-1}^H + \mathbf{x}_{n-1}\mathbf{y}_{n-1}^H). \end{aligned}$$

The penultimate step is now completed. Only one Givens transformation still needs to be performed. Computed explicitly G_{n-1} determines the final similarity transformation (the factors and terms that do not change have already a changed subscript):

$$
\begin{aligned}
H_n &= G_{n-1}^H H_{n-1} G_{n-1} \\
&= G_{n-1}^H V_{n-1}^{(\mathbf{u})} V_n^{(\mathbf{x})} (B_{n-1} + \mathbf{u}_n \mathbf{v}_{n-1}^H + \mathbf{x}_n \mathbf{y}_{n-1}^H) G_{n-1} \\
&= \quad
\begin{array}{c}
\\ \\ \\ \\ \\ \\ \\
\end{array}
\left(\begin{bmatrix}
\times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\
\times & \times & \times & \cdot & \cdot & \cdot & \cdot \\
\times & \times & \times & \times & \cdot & \cdot & \cdot \\
 & \times & \times & \times & \times & \cdot & \cdot \\
 & & \times & \times & \times & \times & \cdot \\
 & & & \times & \times & \times & \times
\end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}_{n-1}^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}_{n-1}^H \right) G_{n-1} \\
&= V_n^{(\mathbf{u})} V_n^{(\mathbf{x})} (B_n + \mathbf{u}_n \mathbf{v}_n^H + \mathbf{x}_n \mathbf{y}_n^H).
\end{aligned}
$$

To complete this step a fusion between G_{n-1}^H and the last Givens transformation of $V_{n-1}^{(\mathbf{u})}$ needs to be performed: $G_{n-1}^H V_{n-1}^{(\mathbf{u})} = V_n^{(\mathbf{u})}$, furthermore simply applying the transformation on the right to B_{n-1}, $\mathbf{v}_{n-1}^H$ and $\mathbf{y}_{n-1}^H$ finishes the QR-step. One is ready to start a new iteration.

5 The multishift strategy

For simplicity we will only present the double shift case restricted to a rank 2 perturbation, the general multishift case proceeds almost identical. The double shift case is quite important since one can restrict computations to the real field in case the original matrix A is non-Hermitian but real, by simply choosing the shifts as complex conjugates. In this case one converges not to a an upper triangular matrix, but to a block upper triangular matrix, having blocks of size at most 2×2 on the diagonal. The 2×2 blocks on the diagonal will then contain the complex conjugate eigenvalues of the original real valued matrix A [24, 14].

The cost of a double shift QR-step is roughly the double of a single shift QR-step. Since, however, the convergence is assumed to be twice as fast global speed is approximately the same. Multishift algorithms have, however, many other advantages [24].

For performing a complete double-shift QR-step on a Hessenberg matrix $2(n-1)$ Givens transformations are needed. Two Givens transformations are needed in the initialization step, and in each of the chasing steps two more to move the bulge down one row. In particular, at step i we have an 'almost' Hessenberg matrix H_i. To go from H_i to H_{i+1} a similarity transformation is performed involving two Givens transformations $G_{i,2}$ and $G_{i,1}$. Transformation $G_{i,2}$ acts on rows i and $i+1$, transformation $G_{i,1}$ acts on rows $i+1$ and $i+2$. First $G_{i,1}^H$ is applied followed by $G_{i,2}^H$.

5.1 Initialization

We present now the more complicated case: the double shift QR-step, applied to matrices in $\mathcal{F}_{n,2}$ represented as in (4). As before, let $\hat{Q}_{\mathcal{I}}$ be the initial unitary transformation such that

$$\hat{Q}_{\mathcal{I}}^H (H - \mu_1 I)(H - \mu_2 I)\mathbf{e}_1 = \beta\, \mathbf{e}_1,$$

where $\beta = \|(H - \mu_1 I)(H - \mu_2 I)\mathbf{e}_1\|_2$. The orthogonal transformation $\hat{Q}_{\mathcal{I}}^H$ consists of two Givens transformations. In particular, $Q_{\mathcal{I}} = G_{1,1} G_{1,2}$ such that $G_{1,2}^H G_{1,1}^H (H - \mu_1 I)(H - \mu_2 I)\mathbf{e}_1 = \beta\, \mathbf{e}_1$. We start by applying a similarity transformation on $H = H_1$, i.e. the initial step. We obtain:

$$
\begin{aligned}
H_2 &= G_{1,2}^H G_{1,1}^H\, H_1\, G_{1,1} G_{1,2} = G_{1,2}^H G_{1,1}^H\, V_1^{(\mathbf{u})} V_1^{(\mathbf{x})} (B_1 + \mathbf{u}_1 \mathbf{v}_1^H + \mathbf{x}_1 \mathbf{y}_1^H)\, G_{1,1} G_{1,2} \\
&= \quad
\begin{array}{c}
\\ \\ \\ \\ \\ \\ \\
\end{array}
\left(\begin{bmatrix}
\times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\
\times & \times & \times & \cdot & \cdot & \cdot & \cdot \\
\times & \times & \times & \times & \cdot & \cdot & \cdot \\
 & \times & \times & \times & \times & \cdot & \cdot \\
 & & \times & \times & \times & \times & \cdot \\
 & & & \times & \times & \times & \times
\end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}_1^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}_1^H \right) G_{1,1} G_{1,2}. \qquad (5)
\end{aligned}
$$

The Givens transformations depicted on the left, marked with $\times$ are the transformations $G_{1,2}^H$ and $G_{1,1}$.

5.1.1 Transformations on the left

Similarly as in the single shift case, we will shift these transformations trough the sequences $V_1^{(\mathbf{u})}$ and $V_1^{(\mathbf{x})}$. The shift-through operation was already depicted in Scheme 5. Based on the equality $G_{1,2}^H G_{1,1}^H V_1^{(\mathbf{u})} V_1^{(\mathbf{x})} = \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} \tilde{G}_{1,2}^H \tilde{G}_{1,1}^H$, we obtain the following:

$$H_2 = \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} \tilde{G}_{1,2}^H \tilde{G}_{1,1}^H (B_1 + \mathbf{u}_1 \mathbf{v}_1^H + \mathbf{x}_1 \mathbf{y}_1^H)\, G_{1,1} G_{1,2}$$

$$= \quad \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot \\ & \times & \times & \times & \times & \cdot & \cdot \\ & & \times & \times & \times & \times & \cdot \\ & & & \times & \times & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}_1^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}_1^H \right) G_{1,1} G_{1,2}.$$

Performing the operations $\tilde{G}_{1,2}^H \tilde{G}_{1,1}^H$ on the terms in the brackets only affects the matrix B_1. Due to the zero structure of the vectors $\mathbf{u}_1$ and $\mathbf{x}_1$ they remain unaltered and are already in final form ($\mathbf{u}_2 = \mathbf{u}_1$ and $\mathbf{x}_2 = \mathbf{x}_1$):

$$H_2 = \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} (\tilde{B}_1 + \mathbf{u}_2 \mathbf{v}_1^H + \mathbf{x}_2 \mathbf{y}_1^H)\, G_{1,1} G_{1,2}$$

$$= \quad \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot \\ \hat{\otimes} & \times & \times & \times & \times & \cdot & \cdot \\ & \times & \times & \times & \times & \cdot & \\ & & \times & \times & \times & \times & \end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}_1^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}_1^H \right) G_{1,1} G_{1,2}.$$

One undesired element is created in the matrix $\tilde{B}_1$. This element is marked with $\hat{\otimes}$ since it will play a role in the chasing. Besides the Givens transformations moving to the bottom, also this bulge will move downwards as the iteration proceeds.

5.1.2 Transformations on the right

Just like in the single shift case removing the remaining transformations on the right is almost identical to the chasing procedure. Performing the transformations on the right on the terms inside the brackets gives us the following:

$$H_2 = \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} (\hat{B}_1 + \mathbf{u}_2 \mathbf{v}_2^H + \mathbf{x}_2 \mathbf{y}_2^H)$$

$$= \quad \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot \\ \otimes & \times & \times & \times & \times & \cdot & \cdot \\ \otimes & \otimes & \times & \times & \times & \times & \cdot \\ & & & \times & \times & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}_2^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}_2^H \right).$$

As mentioned before, the bulge $\hat{\otimes}$ makes part of the chasing and will move down along his subdiagonal. Hence, we only tempt to remove the undesired elements in the first column of the matrix $\hat{B}_1$. Two Givens transformations are needed.

$$H_2 = \tilde{V}_1^{(\mathbf{u})} \tilde{V}_1^{(\mathbf{x})} \hat{G}_{1,1} \hat{G}_{1,2} (B_2 + \mathbf{u}_2 \mathbf{v}_2^H + \mathbf{x}_2 \mathbf{y}_2^H)$$

$$= \quad \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot \\ & \times & \times & \times & \times & \cdot & \cdot \\ & \hat{\otimes} & \times & \times & \times & \times & \cdot \\ & & \times & \times & \times & \times & \end{bmatrix} + \begin{bmatrix} \times \\ \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{v}_2^H \quad + \begin{bmatrix} \times \\ \times \\ \\ \\ \\ \\ \end{bmatrix} \mathbf{y}_2^H \right).$$

Performing shift-through operations to bring the transformations $\hat{G}_{1,1}\hat{G}_{1,2}$ to the front gives us $\tilde{V}_1^{(\mathbf{u})}\tilde{V}_1^{(\mathbf{x})}\hat{G}_{1,1}\hat{G}_{1,2} = G_{2,1}G_{2,2}V_2^{(\mathbf{u})}V_2^{(\mathbf{x})}$, resulting:

$$H_2 = G_{2,1}G_{2,2}V_2^{(\mathbf{u})}V_2^{(\mathbf{x})}(B_2 + \mathbf{u}_2\mathbf{v}_2^H + \mathbf{x}_2\mathbf{y}_2^H), \tag{6}$$

which finalizes the initialization step.

5.2 The chasing

The global flow of the chasing is as follows. When starting the chasing procedure we have a matrix factorization of the following form (assume $i < n-3$):

$$H_i = G_{i,1}G_{i,2}V_i^{(\mathbf{u})}V_i^{(\mathbf{x})}(B_i + \mathbf{u}_i\mathbf{v}_i^H + \mathbf{x}_i\mathbf{y}_i^H),$$

where all vectors and matrices satisfy the constraints posed by the representation, only the matrix B_i has a bulge, a nonzero element in position $(i+2, i)$. One starts with a similarity transformation determined by $G_{i,1}G_{i,2}$.

$$\begin{aligned} H_{i+1} &= G_{i,2}^H G_{i,1}^H H_i G_{i,1}G_{i,2} \\ &= V_i^{(\mathbf{u})}V_i^{(\mathbf{x})}(B_i + \mathbf{u}_{i+1}\mathbf{v}_i^H + \mathbf{x}_{i+1}\mathbf{y}_i^H)G_{i,1}G_{i,2} \\ &= V_i^{(\mathbf{u})}V_i^{(\mathbf{x})}(\hat{B}_i + \mathbf{u}_{i+1}\mathbf{v}_{i+1}^H + \mathbf{x}_{i+1}\mathbf{y}_{i+1}^H). \end{aligned}$$

The vectors $\mathbf{u}_i = \mathbf{u}_{i+1}$ and $\mathbf{x}_i = \mathbf{x}_{i+1}$ are not affected by the transformations. The matrix $\hat{B}_i$ has three undesired nonzero elements in positions $(i+2, i)$, $(i+3, i)$ and $(i+3, i+1)$. The Givens transformations $\hat{G}_{i+1,1}^H$ and $\hat{G}_{i+1,2}^H$ are constructed such to annihilate respectively the elements in positions $(i+3, i)$ and $(i+2, i)$:

$$H_{i+1} = V_i^{(\mathbf{u})}V_i^{(\mathbf{x})}\hat{G}_{i,1}\hat{G}_{i,2}(B_{i+1} + \mathbf{u}_{i+1}\mathbf{v}_{i+1}^H + \mathbf{x}_{i+1}\mathbf{y}_{i+1}^H).$$

The matrix B_{i+1} has only one undesired nonzero element in position $(i+3, i+1)$, which is hence shifted down, w.r.t. the position in B_i. Applying few times the shift-through operation to bring the Givens transformations to the front gives us the result:

$$H_{i+1} = G_{i+1,1}G_{i+1,2}V_{i+1}^{(\mathbf{u})}V_{i+1}^{(\mathbf{x})}(B_{i+1} + \mathbf{u}_{i+1}\mathbf{v}_{i+1}^H + \mathbf{x}_{i+1}\mathbf{y}_{i+1}^H).$$

5.3 The final similarity transformations in the chasing

Once again, we have to deal with the last steps separately, because the implicit algorithm cannot be carried on fully to the end. The ideas applied are almost identical to the ones of Subsection 4.2.1.

6 Deflation

The QR-iterations should always be combined with a deflation technique, which allows to split the problem into two or more smaller problems with the same structure. For a Hessenberg matrix H, deflation means that the Hessenberg matrix is numerically of the following form

$$H = \left[\begin{array}{c|c} H(1:i,1:i) & \times \\ \hline 0 & H(i+1:n,i+1:n) \end{array}\right] = \left[\begin{array}{ccccc|ccccc} \times & \times & \times & \times & & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & & \times & \times & \times & \times & \times \\ & \times & \times & \times & & \times & \times & \times & \times & \times \\ & & \times & \times & & \times & \times & \times & \times & \times \\ \hline & & & 0 & & \times & \times & \times & \times & \times \\ & & & & & \times & \times & \times & \times & \times \\ & & & & & & \times & \times & \times & \times \\ & & & & & & & \times & \times & \times \\ & & & & & & & & \times & \times \end{array}\right]. \tag{7}$$

The matrix can be subdivided into two other Hessenberg matrices $H(1:i, 1:i)$ and $H(i+1:n, i+1:n)$ on which one can continue working separately.

In this paper, however, there is a strong connection between the Hermitian matrix S and the rank m part. The matrix H has a zero block, but this does not necessary mean that either S or UV^H has a zero block in the corresponding position. Only their sum must have a zero block in this position.

The coupled representation used in this manuscript makes it impossible to apply deflation in the classical sense. This means that one cannot just divide the Hessenberg matrix into two parts, each having their specific representation. In this case we maintain a global representation for the entire matrix, but the knowledge that there are deflatable blocks will make it possible to act on these different blocks separately.

This section is organized as follows. In Section 6.1 we will see how to detect deflation and how modify the representation to incorporate deflation. Sections 6.2.1 to 6.2.3 discuss how the algorithm changes in case deflation is applied. In the Appendix we treat the algorithmic changes for accounting of special structures in the low rank perturbation.

6.1 The representation in case of deflation

6.1.1 Assumptions on the Givens transformations

In the remainder of this section we assume a strict rank 2 perturbation. This means that the vectors $\mathbf{u}$ and $\mathbf{x}$ have no trailing zeros. No trailing zeros means that the sequences of Givens transformations $V^{(\mathbf{u})}$ and $V^{(\mathbf{x})}$ consist of non-identity Givens transformations. Assuming e.g. $\mathbf{u}$ to have a trailing sequence of zeros results in a leading (left-most) sequence of identity Givens transformations in $V^{(\mathbf{u})}$. This is excluded for the moment and discussed in the Appendix.

Since the shift-through operation of three non-identity Givens always results in three new non-identity Givens the chasing procedure can never create identity Givens transformations in the middle of the sequences $V^{(\mathbf{u})}$ or $V^{(\mathbf{x})}$. The creation of identity Givens transformations can only occur at the left part of the sequence, when for example a fusion is applied. We can therefore conclude that after a QR-step is performed, we can have Givens transformations equal to the identity matrix only at the bottom (left-most) part of the sequence. In this case again we refer to the Appendix.

So to conclude: in the first part of this section we explicitly assume all Givens transformations in the sequences to be different from the identity.

6.1.2 Detection of deflation

The easiest, but not fastest way to detect numerical zeros on the subdiagonal of the Hessenberg matrix is to explicitly compute these subdiagonal elements. This approach is used in [9]. Due to our specific representation there is, however, a faster way to detect deflation. First we describe the technique for a rank 2 perturbation, then we generalize to the generic rank m case. Moreover, in case $m = 0$ this coincides with the standard approach.

Suppose the matrix $H = V^{(\mathbf{u})}V^{(\mathbf{x})}\left(B + \mathbf{u}\mathbf{v}^H + \mathbf{x}\mathbf{y}^H\right)$ we are working with can be split into two independent problems. This means that the element $H(i+1) = 0$ for a specific i.

Let us denote by $W = {V^{(\mathbf{x})}}^H {V^{(\mathbf{u})}}^H H = B + \mathbf{u}\mathbf{v}^H + \mathbf{x}\mathbf{y}^H$. Since the matrices $V^{(\mathbf{u})}$ and $V^{(\mathbf{x})}$ represent sequences of ascending Givens transformations, we have $H(i+1, i) = 0$ if and only if $W(i+3, i) = 0$. Since $i + 3 \geq 4$ we have $\mathbf{u}(4:n) = 0$ and $\mathbf{x}(4:n) = 0$. This proves that, since all the Givens transformations are different from the identity, $W(i+3, i) = 0$ if and only if $B(i+3, i) = 0$. The condition $B(i+3, i) = 0$ can be checked by using a relative criterion involving the size of the neighboring elements, and it is easy to verify whether deflation can be applied.

The representation, however, doesn't reveal deflation in two other cases. In fact, we cannot determine if $H(n-1, n-2) = 0$ or $H(n, n-1) = 0$ without computing explicitly these entries retrieving the values from the representation. This problem is similar to the problem we had for computing the final steps in the chasing steps of the QR-method (see Sections 4.2.1 and 5.3).

Graphically, $H(i+1,i)=0$, corresponds to a factorizations of the following form (the dashed line depicts where deflation should be applied in the original Hessenberg matrix H):

$$H = \begin{array}{c} \ddots \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \left(\left[\begin{array}{cccc:cccc} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \hdashline & \times & \times & \times & \times & \cdot & \cdot & \cdot \\ & & \times & \times & \times & \cdot & \cdot & \cdot \\ & & & 0 & \times & \times & \cdot & \cdot \\ & & & & \times & \times & \times & \cdot \\ & & & & & \times & \times & \times \end{array}\right] + \left[\begin{array}{c} \times \\ \\ \\ \\ \\ \\ \\ \\ \end{array}\right] \mathbf{v}_2^H + \left[\begin{array}{c} \times \\ \times \\ \\ \\ \\ \\ \\ \\ \end{array}\right] \mathbf{y}_2^H \right). \tag{8}$$

We draw the attention once more to the fact that the zero indicating deflation is positioned two rows below the position where one should actually subdivide the matrix, the column, however remains the same.

Remark 3 *Considering a rank m perturbation we have $m+1$ subdiagonals. Assuming all used Givens transformations to be different from the identity (otherwise we refer to the Appendix). A zero element in the Hessenberg matrix can be detected as a zero on the $(m+1)-th$ subdiagonal. The zero resides in the same column, only shifts down m rows. The final m subdiagonal elements of the Hessenberg matrix have to be computed explicitly, their zeroness is not reflected in the representation, simply because we cannot shift down m more rows.*

Note also that in case of a standard Hessenberg matrix, which means a rank 0 perturbation, the location of the zero corresponds exactly to the zero on the subdiagonal, which brings us to the standard setting.

In standard deflation approaches, the matrix is divided into two independent subproblems with the same characteristics of the original one. On the contrary, in our case, we cannot decouple the problem into two smaller subproblems because we are using a different representation and the two parts are interacting.

Let us illustrate how this is done by using this representation. Again we assume to be working with the double shift method on a rank 2 perturbation, this illustrates the general idea.

6.2 Performing QR-steps after deflation was applied

In the previous section it was shown, that even-though deflation can be applied, both independent Hessenberg matrices (see Equation (7)) $H(1:i,1:i)$ and $H(i+1:n,i+1:n)$ still have interaction in the representation.

It is, however, possible to perform QR-steps on $H(1:i,1:i)$ and $H(i+1:n,i+1:n)$ just like in a standard QR-algorithm were one can completely separate both matrices. In the following subsections we will show how to perform QR-steps on the top parts: $H(1:i,1:i)$ bottom parts $H(i:n,i:n)$ $(1<i<n)$ and also middle parts: $H(i:j,i:j)$, $1<i<j<n$ (assuming of course suitable deflation possibilities in H for either three cases). We denote the entire matrix we act on as H, this corresponds to the matrix H from Equation (7) and (8). We will however only work on the top, bottom and middle part of this matrix H!

6.2.1 The top part

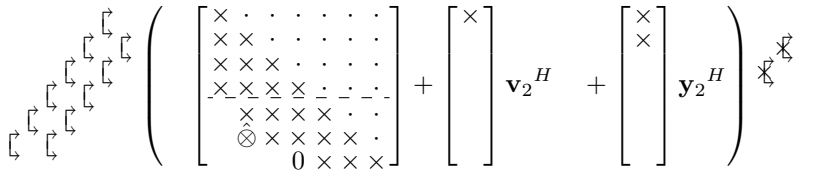
Assume deflation is possible and the following matrix factorization is given (the dashed line depicts the actual splitting between upper and lower part in the Hessenberg matrix). We start performing a QR-step on the matrix $H(1:i,1:i)$. Denote H by H_1, that is $H_1 = H$.

$$H_1 = \begin{array}{c} \ddots \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \left(\left[\begin{array}{cccc:cccc} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ \hdashline & \times & \times & \times & \times & \cdot & \cdot & \cdot \\ & & \times & \times & \times & \times & \cdot & \cdot \\ & & & 0 & \times & \times & \times & \cdot \\ & & & & & \times & \times & \times \end{array}\right] + \left[\begin{array}{c} \times \\ \\ \\ \\ \\ \\ \\ \\ \end{array}\right] \mathbf{v}_1{}^H + \left[\begin{array}{c} \times \\ \times \\ \\ \\ \\ \\ \\ \\ \end{array}\right] \mathbf{y}_1{}^H \right).$$

Let us perform an implicit QR-step on the top part of the matrix. One would expect the chasing step to break down, once the dashed line is reached. Assume the shifts are chosen with regard to rapid convergence for the upper part of the Hessenberg matrix.[3]

The initial transformations are performed similarly as in the standard case. Hence we obtain equation (6): $H_2 = G_{2,1}G_{2,2}V_2^{(\mathbf{u})}V_2^{(\mathbf{x})}(B_2 + \mathbf{u}_2\mathbf{v}_2^H + \mathbf{x}_2\mathbf{y}_2^H)$, $G_{2,1}$ and $G_{2,2}$ determining the upcoming chasing step. Performing the unitary similarity transformation with $G_{2,2}$ and $G_{2,1}$ gives us $H_3 = V_2^{(\mathbf{u})}V_2^{(\mathbf{x})}(B_2 + \mathbf{u}_2\mathbf{v}_2^H + \mathbf{x}_2\mathbf{y}_2^H)G_{2,1}G_{2,2}$, graphically depicted as follows: ($G_{2,1}$ and $G_{2,2}$ are marked with a $\times$ and shown on the right):

$H_2 =$

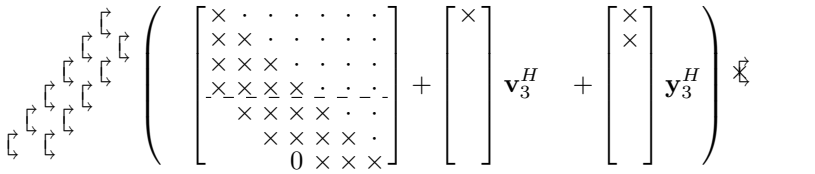


We see that since $B(7,4) = 0$ no bulge is created when transformation $G_{2,1}$ is applied to the columns; multiplication by matrix $G_{2,2}$ will leave a small bulge in position $(6,2)$ that can be removed by means of a Givens transformation acting on rows 5 and 6. This transformation can simply be dragged to the left as done before by repeatedly applying the shift-trough Lemma 2. It is interesting to remark that we have an impact below the dashed line! We obtain

$$H_3 = G_{3,1}V_3^{(\mathbf{x})}V_3^{(\mathbf{u})}\left(B_3 + \mathbf{u}_3\mathbf{v}_3^H + \mathbf{x}_3\mathbf{y}_3^H\right).$$

Another similarity transformation is now performed, involving Givens transformation $G_{3,1}$. Since the original Hessenberg matrix has the element $(5,4)$ zero, we would expect the algorithm to breakdown after this similarity transformation. Before applying $G_{3,1}$ on the right we have:

$H_3 =$



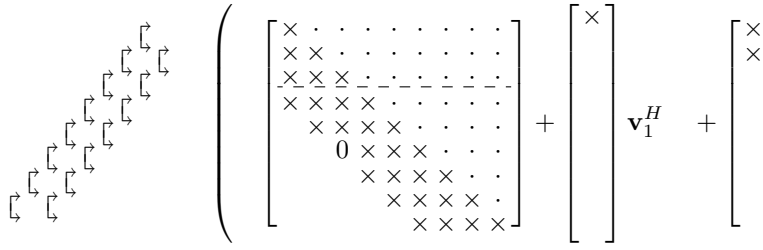
(9)

after applying $G_{3,1}$ on the right nothing essential changes. When trying to perform the chasing step, we see that no bulge in position $(7,3)$ will be created and the algorithm breaks down, as expected. One has performed an implicit QR-step on the part $H(1:i,1:i)$.

6.2.2 Bottom part

Assume we have a matrix $H = H_1$ of the following form and we would like to perform a QR-step now on the bottom part $H(i:n,i:n)$. Even though the Givens transformations acting on the top part will be of no influence, just as the vectors $\mathbf{x}$ and $\mathbf{u}$ we do depict them to omit confusion. For simplicity we restrict to a 9×9 matrix, the general case proceeds similarly.

$H_2 =$



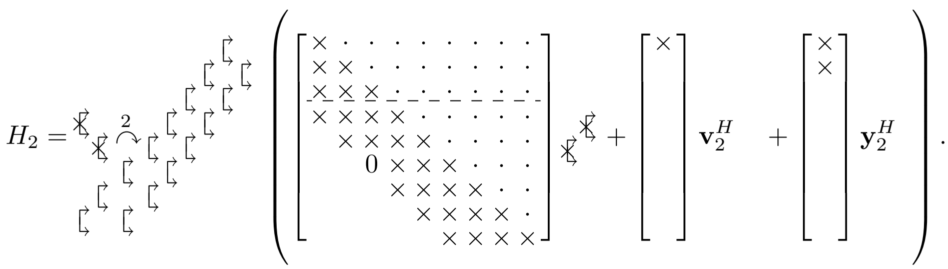
(10)

[3]The shifts are therefore computed based on the bottom right part of the matrix $H(1:i,1:i)$.

The shifts are computed and the initial Givens transformations are determined by explicitly computing the necessary columns of the matrix H. With respect to the original matrix in (7), we are now looking for two shifts μ_1 and μ_2 and two Givens transformations such that

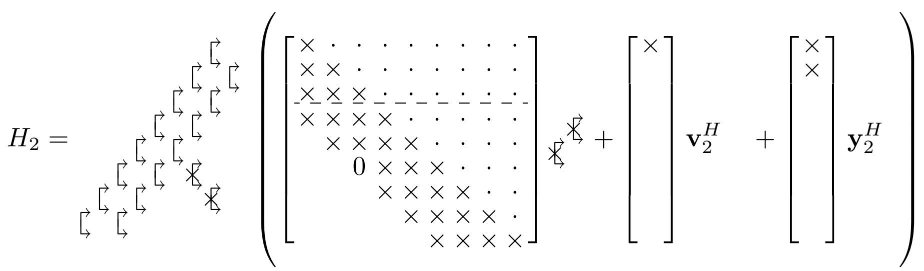
$$G_{1,2}^H G_{1,1}^H (H(i:n,i:n) - \mu_1 I)(H(i:n,i:n) - \mu_2 I)\mathbf{e}_1 = \beta \mathbf{e}_1,$$

where $\mathbf{e}_1$ has the size of $H(i:n,i:n)$. The two Givens should act on row $1,2$ and $2,3$ of the matrix $H(i:n,i:n)$, but since we are working with H that has extra rows on top, $G_{1,2}^H$, is acting on rows 4 and 5 while $G_{1,1}^H$ combines rows 5 with row 6. Graphically, we obtain (the Givens determining the similarity transformation are marked with $\times$):

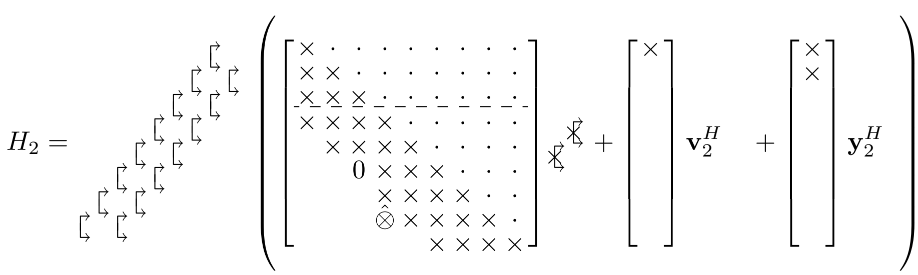


(11)

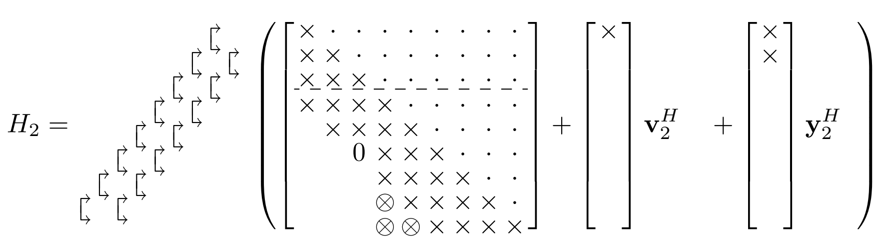
We expect that the depicted zero remains in this position after the initial similarity transformation is performed, otherwise our initial deflation was destroyed. In fact we can do almost exactly the same as in the standard initial step. We first shift through the left-most marked Givens transformations, as depicted in (11) . We obtain the following situation:



Now the zero plays an important role. Applying these two transformations on the matrix B_1, will only create one bulge. Hence the zero remains intact.



There are two Givens transformations remaining on the right. Applying then we will produce a bulge:



To remove the bulge we have to combine rows 8 and 9 and 7 and 8. These last two Givens can be brought to the front applying a series of shift-through operations (just like in Section 5.1) and one can continue the chasing procedure.

6.2.3 Middle part

When running the algorithm zeros will be created in many more instances. This will also result in cases such as for example:

$$H = \left[\begin{array}{c|c|c} H(1:i-1,1:i-1) & \times & \times \\ \hline 0 & H(i:j,i:j) & \times \\ \hline 0 & 0 & H(j+1:n,j+1:n) \end{array}\right]. \tag{12}$$

We know already how to perform QR-steps on the top part: the matrix $H(1:i-1,1:i-1)$ (standard initial step and new finishing step), on the bottom part: matrix $H(j+1:n,j+1:n)$ (standard initial step and standard finishing step). In fact, performing a QR-step on this middle part is just a combination of the standard initial step, with the new finishing step from the top part.

7 Software and experiments

7.1 The software

In this section we present some details concerning the software implementation. The code was written in Matlab R2007b and is available online[4].

As described in Section 3, the representation is based on 2×2 Givens rotations, which can be stored by either 1 or 2 parameters. So globally the memory used by the algorithm for the rank-one case, is given by $n-1$ places for the unitary 2×2 matrices[5], $3(n-1)$ places for the diagonal, subdiagonal and subsubdiagonal of the matrix B and finally n places for the product $\mathbf{u}\mathbf{v}^H$. For the rank 2 perturbation, an extra sequence of unitary transformations, an extra subdiagonal for B and an extra vector for the product $\mathbf{x}\mathbf{y}^H$ are needed.

Since, in general, we will consider non-Hermitian eigenvalue problems, the Rayleigh shift was taken, this means that for the single shift case, we chose the shift value equal to the last diagonal element while for the double shift case, both the eigenvalues of the trailing 2×2 block were taken.

It is well known that balancing can have a significant impact on the accurateness of the computed results. This means that a permuted diagonal matrix D is constructed such that the matrix $D^{-1}AD$ has the row and column norms as nearly as possible equal to each other. See, e.g., the matrix in Equation (14) in Subsection 7.2.1, which becomes Hermitian tridiagonal and hence has perfectly conditioned eigenvalues after scaling with the matrix $D = \mathrm{diag}([1, \ldots, 1, \sqrt{\alpha}])$. Although, in the general non-Hermitian setting, balancing can be performed without bothering about the structure, in our case, the Hermitian plus low rank structure needs to be maintained, hence it is not trivial to design a good balancing strategy maintaining this structure.

7.1.1 Deflation

For a generic Hessenberg matrix $H = (h_{ij})$, deflation is admitted when the following constraint is satisfied (see e.g. [26]):

$$h_{i+1,i} < \epsilon(|h_{ii}| + |h_{i+1,i+1}|), \tag{13}$$

ϵ being the machine precision.

This criterion is, however, based on the entire matrix H, which consists of a sum of a Hermitian and a low rank matrix. Both the Hermitian and the low rank matrix have, however, a non

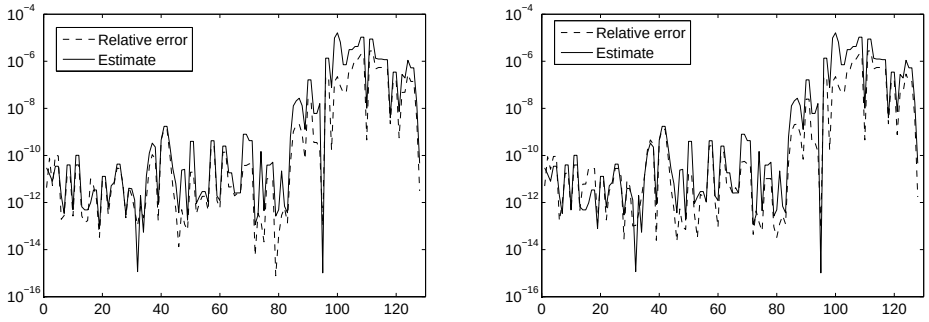
[4] http://www.cs.kuleuven.be/∼raf/

[5] In the actual implementation we used 2×2 unitary matrices instead of Givens transformations, that although more expensive from the occupation of memory are easier to handle.

neglectable low rank part below the subdiagonal. Perfect annihilation is required, in order to create zeros in the matrix H. Numerical experiments show that in case of coefficients of largely varying sizes in the matrix H, this annihilation might not be perfect anymore. Round-off introduces errors, making it difficult/impossible to accurately compute the subdiagonal elements $h_{i+1,i}$. The other matrix elements in H are computed up to high precision, but the difficulties of determining $h_{i+1,i}$ sometimes results in too early or too late termination of the QR-steps, resulting often in a loss of accuracy. This behavior was encountered in several of the numerical experiments performed.

In Section 6.1.2, it was illustrated how a zero in H can be detected in the representation by finding a corresponding zero in a subsubdiagonal of the matrix B. Let us refer to these two deflation criteria as the $\mathcal{H}$-criterion (Hessenberg) or the $\mathcal{B}$-criterion (zero detected in the matrix B). For detecting a zero in a subdiagonal of B we use Equation (13), but then applied to the subsubdiagonal elements of B.

In the following numerical experiment we compare both deflation criteria for the same matrix of size 128. In both figures all eigenvalues are plotted and their estimate of accuracy based on the condition number is also depicted. Figure 1(a) represents the eigenvalues computed via the standard $\mathcal{B}$-deflation criterion while Figure 1(b) we used the $\mathcal{H}$-criterion. Both figures look almost identical and relative accuracy of all the eigenvalues is comparable[6]. The average number of iterations is significantly different. The $\mathcal{B}$-criterion uses 4.5 iterations on average, whereas the $\mathcal{H}$-criterion due to cancellation needs on average 14 iterations. In fact the iteration can be stopped far before the 14 iterations, but one is not able to determine the subdiagonal elements accurately and hence one is not sure when to terminate the iteration process to obtain a certain level of accuracy.



(a) $\mathcal{B}$-criterion (b) $\mathcal{H}$-criterion

Figure 1: Comparison of two deflation criteria. Relative error in the eigenvalues and estimated error using the $\mathcal{B}$-deflation criterion (a) and the $\mathcal{H}$-criterion (b).

7.1.2 Computational complexity

An important operation in the algorithm is the shift-through operation. Each shift through-operation involves a QR-factorization of a 3×3 unitary matrix. Constructing the unitary matrix takes $\approx$ 18 flops, computing the Givens transformations and performing them on the unitary matrix involves another ≈ 33 operations.

Performing a simple QR-step can be divided in two main steps:

- First operation on the left. This involves a single shift through operation (51 flops), followed by applying the unitary transformation on the matrix B ($\approx$ 18 flops).
- Inside the chasing, one has to perform the unitary transformation on the right and drag it through the matrix B such that it pops up again on the left. This has to be performed $n-1$

[6]In this example all elements were constructed randomly. When sizes of the elements vary more accuracy is quite often lost in the $\mathcal{H}$-criterion.

times. To perform the operations, one needs a window of 4×3 sliding on the main diagonal of the matrix B. Computing this window takes ≈ 30 operations. Applying the unitary matrix on $\mathbf{v}$ takes 6 operations, applying it on B and factoring it again out takes ≈ 47 flops. Performing the shift through operation to bring it to the front takes ≈ 51 operations.[7]

Hence we obtain a global cost of approximately $134n - 65$ for each QR-step on a Hessenberg matrix of size n. The designed algorithm is hence clearly an $\mathcal{O}(n^2)$ method.

7.2 The numerical experiments

In this section we present the results of the numerical experiments performed on matrices belonging to the class $\mathcal{F}_{n,1}$.

We denote by $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \ldots, \lambda_n]$ the vector containing the exact eigenvalues and by $\tilde{\boldsymbol{\lambda}} = [\tilde{\lambda}_1, \tilde{\lambda}_2, \ldots, \tilde{\lambda}_n]$ the vector of the computed ones, sorted in such a way $\tilde{\lambda}_i$ is an approximation of λ_i. The error criterion for measuring accurateness are as follows:

$$E^{(abs)} = \|\boldsymbol{\lambda} - \tilde{\boldsymbol{\lambda}}\|_\infty = \max_i \left\{ |\lambda_i - \tilde{\lambda}_i| \right\},$$
$$E^{(rel)} = \max_i \left\{ \frac{|\lambda_i - \tilde{\lambda}_i|}{|\lambda_i|} \right\}.$$

To compare with the results from [9] we also use the infinity norm relative criterion defined as

$$E^{(rel2)} = \frac{\|\lambda - \tilde{\lambda}\|_\infty}{\|\lambda\|_\infty}.$$

Note that $E^{(rel)} <$ tol guarantees that all the eigenvalues have been computed with a relative error lower than the tolerance tol, while this is not true when $E^{(rel2)}$ is considered. In our numerical results we compared the eigenvalues returned by our program with the Matlab® `eig` and we used a cutting criteria of 10^{-16} as deflation tolerance.

The purpose of this experimentation is to test accuracy in the eigenvalues computation with respect to general purpose algorithms not specifically designed to deal with structured matrices. The test suite consists of different kinds of perturbed tridiagonal matrices and of Hessenberg forms of Hermitian plus low rank matrices.

7.2.1 Description of the test suite

In this section we describe the matrices used to estimate the quality of our algorithm, both from the point of view of accuracy and computational speed. Many of the matrices considered have a particular interest from an applicative point of view. In particular, we tested our software on almost tridiagonal matrices and on Hessenberg matrices that can be expressed as the sum of Hermitian and rank-one matrices. Changing slightly the notation introduced in [9], we will consider these different test matrices.

`Type I: Chebyshev-Comrade matrices` are matrices of the form $A = T_n + \mathbf{u}\mathbf{e_n}^T$, where T_n is an $n \times n$ Hermitian tridiagonal matrix associated with Chebyshev polynomials of the first kind, that is

$$T_n = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & & & & \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{2} & & & \\ & \frac{1}{2} & \ddots & \ddots & & \\ & & \ddots & \ddots & \frac{1}{2} & \\ & & & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} \\ & & & & \frac{1}{\sqrt{2}} & 0 \end{bmatrix}.$$

[7] In this rough estimate we neglect the final step which needs to be determined explicitly, since no shift through needs to be performed, the cost is roughly the same.

Table 1: Errors for Type-I matrices of size $n = 128$, and $\mathbf{u} = \alpha\,[1, \ldots, 1]^T$.

α	1	10^3	10^5	10^7	10^8	10^{11}
$E^{(abs)}$	3.0631e-13	8.6616e-12	1.4552e-10	5.5879e-09	8.9407e-08	1.7243e-06
$E^{(rel)}$	3.0631e-13	8.6722e-12	8.5210e-11	5.5943e-09	1.4885e-08	3.8406e-06
$E^{(rel2)}$	1.6396e-13	8.6555e-15	1.4552e-15	5.5879e-16	8.9407e-16	1.7243e-17
avrgit	2.6371	2.8182	2.8099	2.8099	2.7934	3.1736

These matrices arise when one wants to compute the roots of a polynomial expressed in the first kind Chebyshev polynomial basis.

`Type-II: Random tridiagonal plus random rank-one permutation.` $A = T + \mathbf{u}\mathbf{e_n}^T$, where T is a random symmetric tridiagonal matrix, and $\mathbf{u}$ is a random vector in $[0, 1]$.

`Type-III: Unsymmetric tridiagonal matrices.` We consider an almost symmetric tridiagonal matrix of the form

$$T = \begin{bmatrix} 0 & 1 & & & \\ 1 & 0 & \ddots & & \\ & \ddots & \ddots & 1 & \\ & & 1 & 0 & 1 \\ & & & \alpha & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & & & \\ 1 & 0 & \ddots & & \\ & \ddots & \ddots & 1 & \\ & & 1 & 0 & \alpha \\ & & & \alpha & 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ 1-\alpha \\ 0 \end{bmatrix} \mathbf{e_n}^T. \tag{14}$$

When α becomes very large, the dominant eigenvalues are ill-conditioned.

`Type-IV: Hamiltonian-like matrices.` We consider the Hessenberg form of Hamiltonian-like matrices which are special diagonal plus rank-one matrices from an application in transport theory[8]. The matrices considered are of size $2n \times 2n$ and are defined in terms of two parameters α and β, and two strictly positive n-vectors $\mathbf{c}, \boldsymbol{\omega}$. The diagonal plus rank-one matrices are as follows

$$A = \begin{bmatrix} D & 0 \\ 0 & -\Delta \end{bmatrix} + \begin{bmatrix} \mathbf{q} \\ \mathbf{e} \end{bmatrix} \left[\mathbf{e}^T, -\mathbf{q}^T\right],$$

where $\mathbf{e} = [1, \ldots, 1]^T$, $\mathbf{q} = [q_1, q_2, \ldots, q_n]^T$, with $q_i = c_i/(2\omega_i)$, D and Δ are two diagonal matrices with $\Delta_{ii} = (\beta\,\omega_i\,(1+\alpha))^{-1}$ and $D_{ii} = (\beta\,\omega_i\,(1-\alpha))^{-1}$. Moreover, $\sum_{i=1}^n c_i = 1$ and $0 < \omega_n < \cdots < \omega_1 < 1$.

Critical cases are those where α approaches 1, since the norm of A increases affecting the accuracy of the approximated eigenvalues. In our tests, we chose $\alpha = 0.9, \beta = 0.8$, $\mathbf{c} = [1 : n]/\mathrm{sum}([1 : n])$ and $\omega = [n : -1 : 1]/(n+1)$.

`Type-V: Hessenberg form of random matrices.` We consider the Hessenberg form of random symmetric plus random rank-one matrices.

In Table 1 we report the results for Type-I matrices choosing $\mathbf{u} = \alpha\,[1, \ldots, 1]^T$, for increasing values of α.

We note that the accuracy in the computation of the eigenvalues deteriorates as α increases, suggesting a dependence from the norm of the perturbation. Comparing the results with the ones from [9] we see that a slight improvement in absolute value is made: one digit is gained.

For Type-I matrices with randomly chosen vector $\mathbf{u}$, we are able to obtain a higher accuracy as reported in Table 2. Comparing with [9] we see, however, that we perform worse.

Results for Type-II matrices, that is random tridiagonal plus random rank-one permutation matrices are reported in Table 3. The values reported are obtained averaging over 50 instances of the problem.

[8] See [15, 9] and the references therein for more explanations about the practical interests of these matrices, and the meaning of the parameters.

Table 2: Errors for Type-I matrices of the form $T_n + \mathbf{u}\mathbf{e}_\mathbf{n}^\mathbf{T}$, with $\mathbf{u} = \text{rand}(n, 1)$. For different values of n, the absolute and relative errors are shown.

n	$E^{(abs)}$	$E^{(rel)}$	$E^{(rel2)}$
50	1.0947e-13	1.0947e-13	8.6909e-14
100	4.2141e-13	2.4800e-11	3.5758e-13
150	3.4097e-12	2.1823e-12	2.1823e-12
200	3.6731e-12	2.6981e-12	2.6981e-12
300	6.8008e-12	5.0597e-12	5.0597e-12
500	1.7089e-11	1.3405e-11	1.3405e-11

Table 3: Errors for Type-II matrices, and for different values of n. The values reported represent the average error obtained over 50 randomly generated instances.

n	$E^{(abs)}$	$E^{(rel)}$	$E^{(rel2)}$
100	2.2438e-12	1.3879e-12	1.0361e-12
150	3.4440e-10	5.6035e-10	1.5864e-10
200	5.7883e-12	4.9680e-12	2.6019e-12
300	1.5570e-12	7.6508e-12	7.2006e-13
500	4.2696e-12	3.0138e-11	1.8976e-12

Table 4: Errors for Type-III matrices, n=128, and different values of α.

α	1	10	10^2	10^3	10^5	10^7	10^8
$E^{(abs)}$	5.8842e-14	7.9892e-13	9.8765e-13	1.6662e-12	5.8321e-11	3.1127e-09	1.8700e-08
$E^{(rel)}$	5.8842e-14	4.9782e-13	4.3876e-13	3.1974e-13	5.1728e-11	1.0268e-09	9.2553e-09
$E^{(rel2)}$	2.9430e-14	2.3967e-13	9.8270e-14	5.2664e-14	1.8443e-13	9.8434e-13	1.8700e-12
avrgit	2.9098	2.9268	3.3719	3.3033	2.9016	3.0656	2.9431

Table 5: Errors for Type-IV matrices of different size. The results are obtained for $\alpha = 0.9$ and $\beta = 0.8$.

n	$E^{(abs)}$	$E^{(rel)}$	$E^{(rel2)}$
50	3.1335e-12	1.8983e-12	9.6708e-15
100	6.8070e-12	3.4106e-12	1.0694e-14
150	1.0714e-11	1.0714e-11	1.1290e-14
200	1.8701e-11	1.8701e-11	1.4825e-14
500	7.2838e-11	7.2838e-11	2.3223e-14
1000	8.0850e-10	8.0850e-10	1.2912e-13

Table 6: Type-V: Hessenberg form of randomly generated matrices.

n	$E^{(abs)}$	$E^{(rel)}$	$E^{(rel2)}$
50	2.0783e-12	2.0783e-12	3.2932e-14
100	9.5799e-12	9.5799e-12	7.6777e-14
150	5.5904e-11	6.2688e-12	2.9514e-13
200	8.2377e-11	8.2377e-11	3.2935e-13
500	9.1855e-11	3.2461e-11	1.4710e-13
1000	3.4718e-09	5.1914e-10	2.7480e-12

Matrices of Type-III are almost symmetric tridiagonal matrices that can be expressed as a rank-one perturbation of a symmetric tridiagonal matrix. Depending on the magnitude of the perturbation the extreme eigenvalues become ill-conditioned. Our algorithm is however very good in dealing with this situation (see Table 4) because for $\alpha = 10^8$ we still have an absolute error of $\mathcal{O}(10^{-8})$ and a relative accuracy of $\mathcal{O}(10^{-12})$, respect to the infinity norm. Comparing with the results provided in [9], we see that for large values of α we gained quite a lot in accuracy.

We have considered also matrices obtained after the reduction to Hessenberg form of Hamiltonian-like matrices. In Table 5 the results are reported for different sizes with $\alpha = 0.9$ and $\beta = 0.8$. Results are comparable with the ones from [9]. Unfortunately, it seems that the loss of accuracy of matrices of higher dimension depends also on the loss of accuracy when applying the unitary similarity reduction to Hessenberg form. The resulting Hessenberg matrix obtained from the reduction had a norm of the lower triangular part already equal to $10^{(-13)}$ for size 100 which is far from the expected zero.

Results for the Hessenberg form of randomly generated symmetric matrices plus random rank-one matrices are given in Table 6. We can see a loss of accuracy for higher values of n which is due to the reduction to Hessenberg form, but still acceptable when looking at the condition numbers of the individual eigenvalues.

8 Conclusions

In this article a new algorithm was developed for performing a QR-step on a Hessenberg matrix which is the sum of Hermitian plus low rank matrix. The implementation was based on the Givens-weight representation. The use of a unitary factorization to represent the matrix results in the incapability of using the standard deflation approach and hence an alternative deflation technique was developed.

Numerical experiments illustrate that this algorithm is a viable alternative to the existing technique proposed in [9]. In some experiments a significant increase in accuracy is obtained, whereas in others the technique from [9] performs better.

Future research involves a backward error analysis and a detailed investigation of the deflation criterion used. Moreover, computing explicitly the last Givens transformation when performing the QR-iteration seems an unnatural step, it would be interesting if we could get rid of this explicit computations.

References

[1] S. Barnett. *Polynomials and Linear Control Systems*. Marcel Dekker Inc, 1983.

[2] D. Bindel, J. W. Demmel, W. Kahan, and Osni A Marques. On computing Givens rotations reliably and efficiently. *ACM Transactions on Mathematical Software*, 28(2):206–238, June 2002.

[3] D.A. Bini, V. Gemignani, L. and V. Pan. Fast and stable QR eiegenvalue algorithms for generalized companion matrices and secular equations. *Numerische Mathematik*, 100:373–408, 2005.

[4] S. Chandrasekaran and M. Gu. Fast and stable eigendecomposition of symmetric banded plus semi-separable matrices. *Linear Algebra and its Applications*, 313:107–114, 2000.

[5] S. Delvaux and M. Van Barel. Rank structures preserved by the QR-algorithm: the singular case. *Journal of Computational and Applied Mathematics*, 189:157–178, 2006.

[6] S. Delvaux and M. Van Barel. Structures preserved by the QR-algorithm. *Journal of Computational and Applied Mathematics*, 187(1):29–40, March 2006.

[7] S. Delvaux and M. Van Barel. A Givens-weight representation for rank structured matrices. *SIAM Journal on Matrix Analysis and Applications*, 29(4):1147–1170, 2007.

[8] S. Delvaux and M. Van Barel. A Hessenberg reduction algorithm for rank structured matrices. *SIAM Journal on Matrix Analysis and Applications*, 29(3):895–926, 2007.

[9] Y. Eidelman, L. Gemignani and I. C. Gohberg. Efficient eigenvalue computation for quasiseparable Hermitian matrices under low rank perturbation. *Numerical Algorithms*, 47(3): 253–273, 2008.

[10] Y. Eidelman, L. Gemignani and I. C. Gohberg. On the fast reduction of a quasiseparable matrix to Hessenberg and tridiagonal forms. *Linear Algebra and its Applications*, 420(1):86–101, January 2007.

[11] D. Fasino, N. Mastronardi and M. Van Barel. Fast and stable algorithms for reducing diagonal plus semiseparable matrices to tridiagonal and bidiagonal form. In *Fast Algorithms for Structured Matrices: Theory and Applications*, volume 323 of *Contemporary Mathematics*, pages 105–118. American Mathematical Society, Providence, Rhode Island, USA, 2003.

[12] J. G. F. Francis. The QR Transformation A Unitary Analogue to the LR Transformation–Part 1. *The Computer Journal*, 4(3):265–271, 1961.

[13] J. G. F. Francis. The QR Transformation–Part 2. *The Computer Journal*, 4(4):332–345, 1962.

[14] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, Maryland, USA, third edition, 1996.

[15] J. Juang and W. W. Lin. Nonsymmetric algebraic Riccati equations and Hamiltonian-like matrices. *SIAM J. Matrix Anal. Appl.*, 20(1):228–243, 1999.

[16] N. Mastronardi, S. Chandrasekaran and S. Van Huffel. Fast and stable reduction of diagonal plus semi-separable matrices to tridiagonal and bidiagonal form. *BIT*, 41(1):149–157, 2003.

[17] B. N. Parlett. *The symmetric eigenvalue problem*. Prentice-Hall, Englewood Cliffs, N.J. 07632, 1980.

[18] R. Vandebril, M. Van Barel, and N. Mastronardi. A note on the representation and definition of semiseparable matrices. *Numerical Linear Algebra with Applications*, 12:839–858, 2005.

[19] R. Vandebril and Z. Bai. An implicit *qr*-algorithm for tridiagonal plus rank 1 matrices. Technical report, Katholieke Universiteit Leuven, Celestijnenlaan 200A, 3000 Leuven (Heverlee), Belgium, 2008.

[20] R. Vandebril, M. Van Barel, and N. Mastronardi. A note on the recursive calculation of dominant singular subspaces. Technical Report TW393, Department of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200A, 3000 Leuven (Heverlee), Belgium, May 2003.

[21] R. Vandebril, M. Van Barel, and N. Mastronardi. A parallel QR-factorization/solver of structured rank matrices. Technical Report TW474, Department of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200A, 3000 Leuven (Heverlee), Belgium, October 2006.

[22] R. Vandebril, M. Van Barel, and N. Mastronardi. *Matrix Computations and Semiseparable Matrices, Volume I: Linear Systems*. Johns Hopkins University Press, Baltimore, Maryland, USA, 2008.

[23] R. Vandebril, M. Van Barel, and N. Mastronardi. *Matrix Computations and Semiseparable Matrices, Volume II: Eigenvalue and Singular Value Methods*. Johns Hopkins University Press, 2008.

[24] D. S. Watkins. *The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods*. SIAM, Philadelphia, Pennsylvania, USA, 2007.

[25] D. S. Watkins. The QR algorithm revisited. *SIAM Review*, 50(1):133–145, 2008.

[26] J. H. Wilkinson. *The Algebraic Eigenvalue Problem*. Numerical Mathematics and Scientific Computation. Oxford University Press, New York, USA, 1999.

Appendix: Different rank combinations in the perturbation

Throughout the entire article we assumed all Givens transformations in the sequences to be different from the identity transformation. In this section we show that admitting identity transformations does not really complicate things. Identity Givens transformations are cheaper to deal with and will eventually vanish.

We know from Section 6, that identity Givens transformations can only occur as a leading part of the sequences $V^{(\mathbf{u}_i)}$, and occur only when $\mathbf{u}_i$ has trailing zeros. Let r_i denote the index of vector $\mathbf{u}_i$, such that $\mathbf{u}_i(r_i+1:n)=0$ and $\mathbf{u}_i(r_i)\neq 0$. Without loss of generality we can reorder the vectors $\mathbf{u}_i$ so that $r_1 \geq r_2 \geq \ldots \geq r_m$, which mean that $\mathbf{u}_1$ is the vector with the least number of trailing zeros, where $\mathbf{u}_n$ has the most zeros[9].

Let us gradually build up the representation. Construct $V^{(\mathbf{u}_1)}$ as a sequence of r_1 Givens transformations such that ${V^{(\mathbf{u}_1)}}^H\mathbf{u}_1$ has only the top element different from zero. Since $H = S + \sum_{i=1}^{n}\mathbf{u}_i\mathbf{v}_i^H$, when applying $V^{(\mathbf{u}_1)}$ we get

$$ {V^{(\mathbf{u}_1)}}^H H = {V^{(\mathbf{u}_1)}}^H S + \sum_{i=1}^{n} {V^{(\mathbf{u}_1)}}^H \mathbf{u}_i\mathbf{v}_i^H $$

[9]There are many possibilities for ordering the vectors and obtaining different representations. We choose the one admitting an easy flow of the chasing procedure.

Let us denote the matrices and vectors on which ${V^{(\mathbf{u}_i)}}^H$ is applied by a superscripted $(\mathbf{u}_i)$. In this way we obtain:

$$ {V^{(\mathbf{u}_1)}}^H H = S^{(\mathbf{u}_1)} + \sum_{i=1}^{n} \mathbf{u}_i^{(\mathbf{u}_1)} \mathbf{v}_i^H. $$

Considering the vectors ${V^{(\mathbf{u}_1)}}^H \mathbf{u}_i = \mathbf{u}_i^{(\mathbf{u}_1)}$, we see that the associated number $r_i^{(\mathbf{u}_1)}$ has possibly grown by 1, w.r.t. the number r_i. More precisely $r_i^{(\mathbf{u}_1)} = r_i$ for those i such that $r_i = r_1$ and if $r_i < r_1$ we obtain $r_i^{(\mathbf{u}_1)} = r_i + 1$. Continuing this procedure and constructing then $V^{(\mathbf{u}_2)}$ as a sequence of $r_2^{(\mathbf{u}_1)} - 1$ Givens transformations such that ${V^{(\mathbf{u}_2)}}^H \mathbf{u}_2^{(\mathbf{u}_1)}$ has only two elements nonzero on the top. The sequence $V^{(\mathbf{u}_2)}$ contains $r_2^{(\mathbf{u}_1)} - 1 = \min(r_1, r_2+1) - 1 < r_1$ Givens transformations. This procedure can be continued and the number of Givens transformations determining $V^{(\mathbf{u}_i)}$ can only decrease, when i increases.

The procedure above is best illustrated by an example. Assume we have a rank 2 perturbation $\mathbf{u}\mathbf{v}^H + \mathbf{x}\mathbf{y}^H$, with $\mathbf{u}(8:9) = 0$ ($r_{\mathbf{u}} = 7$), $\mathbf{x}(5:9) = 0$ ($r_{\mathbf{x}} = 4$). This means the following for the symmetric matrix $S = H - \mathbf{u}\mathbf{v}^H - \mathbf{x}\mathbf{y}^H$: the bottom part $S(8:9, 8:9)$ is of Hessenberg form; the middle part $S(5:7, 5:7)$ is of Hessenberg form plus a rank 1 perturbation; the top part $S(1:4, 1:4)$ is of Hessenberg form plus a rank 2 perturbation. This means that for the global matrix H we have that $H(1:4, 1:4)$ is the sum of a symmetric plus strict rank 2 perturbation, the part $H(5:7, 5:7)$ is a symmetric plus strict rank 1 perturbation, the bottom part $H(8:9, 8:9)$ is a symmetric plus rank 0 perturbation (hence $S(8:9, 8:9)$ is symmetric tridiagonal).

This structure implies that we need less Givens transformations to create zeros in the vectors $\mathbf{u}$ and $\mathbf{x}$. We obtain the following representation scheme where Givens transformations equal to the identity matrix are not depicted. The vector $\hat{\mathbf{u}}$ has 1 element different from zero and $\hat{\mathbf{x}}$ has 2 elements different from zero. The unitary matrix $V^{(\mathbf{u})}$ consists of 6 Givens transformations such that ${V^{(\mathbf{u})}}^H \mathbf{u} = \hat{\mathbf{u}}$. The unitary matrix ${V^{(\mathbf{x})}}^H$ is constructed to annihilate all, except the topmost 2 elements of the vector ${V^{(\mathbf{u})}}^H \mathbf{x}$, so we obtain ${V^{(\mathbf{x})}}^H {V^{(\mathbf{u})}}^H \mathbf{x} = \hat{\mathbf{x}}$. Since ${V^{(\mathbf{u})}}^H \mathbf{x}$ has lost its zero in position 5 it consists of three Givens transformations:

$$ H = V^{(\mathbf{u})} V^{(\mathbf{x})} (B + \hat{\mathbf{u}}\mathbf{v}^H + \hat{\mathbf{x}}\mathbf{y}^H) $$

$$ = \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \begin{array}{c} \vec{\curvearrowright} \\ \vec{\curvearrowright}\ \vec{\curvearrowright} \\ \vec{\curvearrowright}\ \vec{\curvearrowright}\ \vec{\curvearrowright} \\ \vec{\curvearrowright}\ \vec{\curvearrowright}\ \vec{\curvearrowright}\ \vec{\curvearrowright} \\ \vec{\curvearrowright}\ \vec{\curvearrowright}\ \vec{\curvearrowright} \\ \vec{\curvearrowright}\ \vec{\curvearrowright} \\ \vec{\curvearrowright} \end{array} \left(\begin{bmatrix} \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \times & \times & \times & \times & \cdot & \cdot & \cdot & \cdot & \cdot \\ & \times & \times & \times & \times & \cdot & \cdot & \cdot & \cdot \\ & & & \times & \times & \times & \cdot & \cdot & \cdot \\ & & & & \times & \times & \times & \cdot & \cdot \\ & & & & & & \times & \times & \cdot \\ & & & & & & & \times & \times \end{bmatrix} + \begin{bmatrix} \times \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \end{bmatrix} \mathbf{v}^H + \begin{bmatrix} \times \\ \times \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \end{bmatrix} \mathbf{y}^H \right). $$

The zeros of $\mathbf{u}$ and $\mathbf{x}$ have an impact on the representation of H in two important places. First of all the sequences of non-identity Givens transformations $V^{(\mathbf{x})}$ and $V^{(\mathbf{u})}$ are shorter than before. Second, the number of subdiagonal elements in B decreases accordingly to the number of Givens transformations.

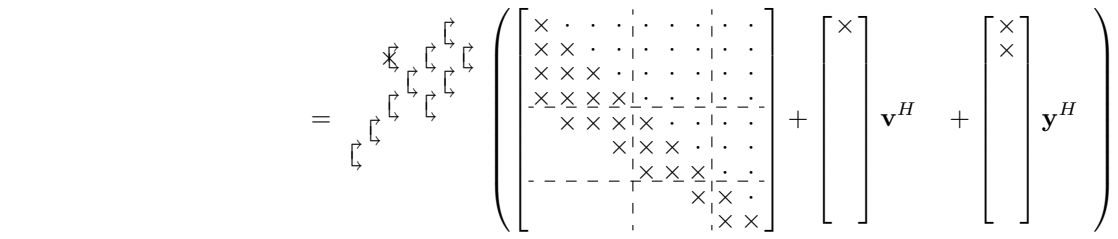
The reader can check the structure of the subdiagonal elements carefully by reconsidering exactly the same techniques as described in Section 3. We have now a matrix H, ideally partitioned into three part accordingly with a specific rank $m = 2, 1, 0$ perturbation.

Let us briefly show how natural the transition from the rank 2 to the rank 1 perturbation proceeds in case of the single shift approach, the multishift approach proceeds similarly.

The fact that many of the Givens transformations equal the identity implies that the chasing procedure becomes much simpler. The only drawback is that we cannot determine the chasing Givens transformations on the fly anymore. We need to compute them explicitly by computing a small block in the Hessenberg matrix. Once the transformation is determined, one can easily apply it on the representation, without destroying it.

Suppose we have performed the initialization step successfully and we have computed the following factorization of the matrix H_2:

$$H_2 = G_2 V_2^{(\mathbf{u})} V_2^{(\mathbf{x})} (B_2 + \mathbf{u}_2 \mathbf{v}_2^H + \mathbf{x}_2 \mathbf{y}_2^H)$$

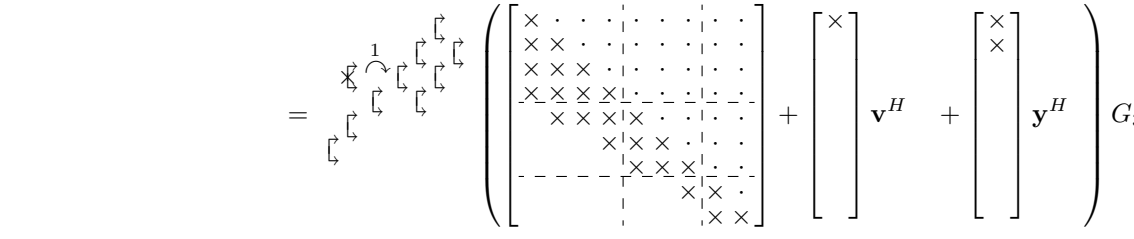


Performing the similarity transformation defined by G_2 we obtain:

$$H_3 = G_2^H H_2 G_2 = V_3^{(\mathbf{u})} V_3^{(\mathbf{x})} (B_3 + \mathbf{u}_3 \mathbf{v}_3^H + \mathbf{x}_3 \mathbf{y}_3^H) G_2.$$

Unfortunately no bulge is created, hence we cannot continue the chasing and column $H_3 \mathbf{e}_2$ needs to be computed explicitly to determine the bulge in position $(4, 2)$. Having determined G_3 explicitly we can perform the next similarity transformation:

$$H_4 = G_3^H V_3^{(\mathbf{u})} V_3^{(\mathbf{x})} (B_3 + \mathbf{u}_3 \mathbf{v}_3^H + \mathbf{x}_3 \mathbf{y}_3^H) G_3$$

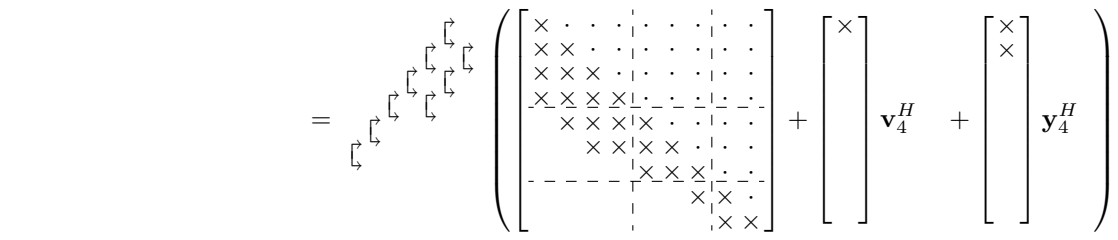


Eventhough one might expect problems now since we have again a Givens transformation on the left, we see that a simple shift-through operation $G_3^H V_3^{(\mathbf{u})} = V_4^{(\mathbf{u})} \tilde{G}_3^H$ followed by a fusion, with the last non-identity Givens transformation of $V_3^{(\mathbf{x})}$ solves this problem:

$$G_3^H V_3^{(\mathbf{u})} V_3^{(\mathbf{u})} = V_4^{(\mathbf{u})} \tilde{G}_3^H V_3^{(\mathbf{x})} = V_4^{(\mathbf{u})} V_4^{(\mathbf{x})}.$$

Applying the transformation G_3 on the right to the terms gives us (fill-in is created in position $(6, 3)$ in B_4):

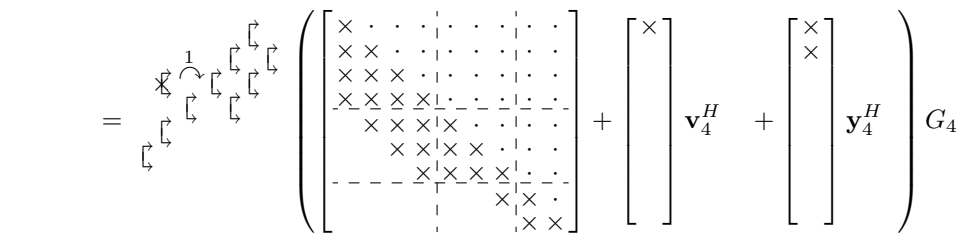
$$H_4 = V_4^{(\mathbf{u})} V_4^{(\mathbf{x})} (B_4 + \mathbf{u}_4 \mathbf{v}_4^H + \mathbf{x}_4 \mathbf{y}_4^H)$$



The created fill-in is not really a problem, it only indicates that the nice rank perturbation (the division in rank $0, 1$ and 2) is getting lost. We see now that the diagonal starts filling-up, from the next iteration we will see that also the Givens transformations start filling-up. This filling-up process is not a problem since in the end we will obtain all Givens transformations different from the identity as we assumed in the beginning of the article.

Again one cannot determine the next chasing step, G_4 also needs to be computed explicitly[10]. Performing this similarity transformation we obtain:

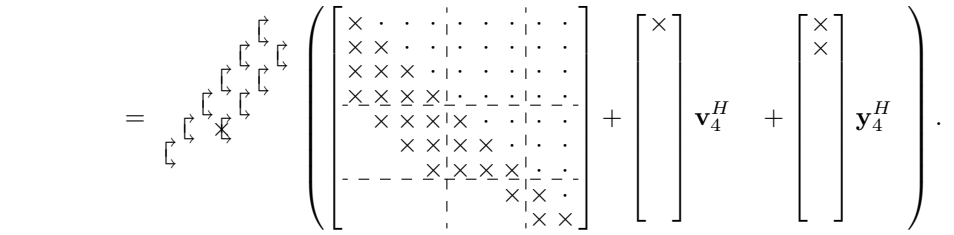
$$H_5 = G_4^H V_4^{(\mathbf{u})} V_4^{(\mathbf{x})} (B_4 + \mathbf{u}_4 \mathbf{v}_4^H + \mathbf{x}_4 \mathbf{y}_4^H) G_4$$



Performing the depicted shift-through operation will put the marked Givens transformation in an empty position: $G_4^H V_4^{(\mathbf{u})} = V_5^{(\mathbf{u})} \tilde{G}_4^H$. This transformation can then simply be added to the sequence of Givens transformation $V_4^{(\mathbf{u})}$, obtaining $V_5^{(\mathbf{x})} = \tilde{G}_4^H V_4^{(\mathbf{x})}$.

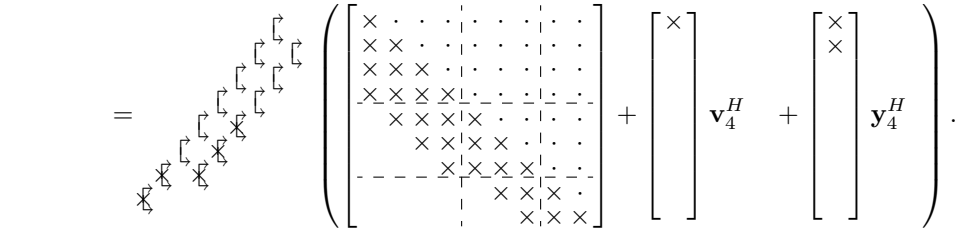
The application of transformation G_4 on the right does not have a significant impact on the structure, there is not even a new bulge created now. The new structure is depicted in the following scheme (We did denote the newly added Givens transformation with $\times$, but in this case it is not undesired in this position, it satisfies the general symmetric plus rank 2 representational constraints):

$$H_5 = V_5^{(\mathbf{u})} V_5^{(\mathbf{x})} (B_5 + \mathbf{u}_5 \mathbf{v}_5^H + \mathbf{x}_5 \mathbf{y}_5^H)$$



The global picture is clear now. Explicitly computing the remaining Givens transformations, filling up empty positions in B as well as in the sequences of Givens transformations leads to the following final scheme. The newly added Givens transformations are marked as $\times$, note that also some extra subsubdiagonal elements have been created.

$$H_n = V_n^{(\mathbf{u})} V_n^{(\mathbf{x})} (B_n + \mathbf{u}_n \mathbf{v}_n^H + \mathbf{x}_n \mathbf{y}_n^H)$$



One can see that with a single iteration step we cannot remove all the identity Givens transformations. This means that some structures will remain intact, but after the application of other QR-steps, due to the fact the matrices are not symmetric, gradually they will disappear (see e.g. [6, 5, 23]).

Remark 4 *We briefly remark that in the case some of the Givens transformations are equal to the identity, as discussed in this appendix, one has to adapt the deflation technique to deal with the particular situation of submatrices with different ranks. In particular, a combination of deflation criteria for the different m and the explicit computation of some columns should be used.*

[10] Computing these Givens transformations is in fact computationally less expensive than performing the standard chasing step, so there is no computational problem here.