On Deflations in Extended QR Algorithms

Thomas Mach *Raf Vandebril*

Report TW 634, September 2013

KU Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

On Deflations in Extended QR Algorithms

Thomas Mach *Raf Vandebril*

Report TW 634, September 2013

Department of Computer Science, KU Leuven

Abstract

In this paper we discuss the deflation criterion used in the extended QR algorithm based on the chasing of rotations. We show that this deflation criterion can be considered to be optimal with respect to absolute and relative perturbation of the eigenvalues.

Further, we present a generalization of aggressive early deflation to the extended QR algorithms. Aggressive early deflation is the key technique for the identification and deflation of already converged, but hidden, eigenvalues. Often these possibilities for deflation are not detected by the standard technique. We present numerical results underpinning the power of aggressive early deflation also in the context of extended QR algorithms. We further generalize these ideas by the transcription of middle deflations.

Keywords : extended QR algorithms, deflation, perturbation bounds, aggressive early deflations, middle deflations, rotations, extended Hessenberg matrices
MSC : Primary : 65F15, Secondary : 15A18.

ON DEFLATIONS IN EXTENDED QR ALGORITHMS*

THOMAS MACH† AND RAF VANDEBRIL†

Abstract. In this paper we discuss the deflation criterion used in the extended QR algorithm based on the chasing of rotations. We show that this deflation criterion can be considered to be optimal with respect to absolute and relative perturbation of the eigenvalues.

Further, we present a generalization of aggressive early deflation to the extended QR algorithms. Aggressive early deflation is the key technique for the identification and deflation of already converged, but hidden, eigenvalues. Often these possibilities for deflation are not detected by the standard technique. We present numerical results underpinning the power of aggressive early deflation also in the context of extended QR algorithms. We further generalize these ideas by the transcription of middle deflations.

Key words. extended QR algorithms, deflation, perturbation bounds, aggressive early deflations, middle deflations, rotations, extended Hessenberg matrices

AMS subject classifications. 65F15, 15A18

1. Introduction. In a series of papers [2, 20–22] Aurentz, Watkins, and Vandebril present a generalization of Francis's implicitly shifted QR algorithm [9, 10] and its derivatives. They replace the Hessenberg matrix by its QR decomposition, where the unitary factor Q is a product of $n-1$ rotations, with n the dimension of the matrix under consideration. These rotators are not necessarily ordered in the same way as for Hessenberg matrices and thus the matrices are called *extended Hessenberg matrices*. The usage of different orderings, which can be adaptively changed during the QR algorithm, can reduce the number of iterations by almost 50% in special cases, see [20]. This huge reduction origins from the fact that the Krylov subspace determining the convergence of Francis's QR algorithm is replaced by a rational Krylov subspace [21].

The replacement of the Hessenberg matrix by the QR decomposition of an extended Hessenberg matrix has the consequence that the standard deflation criterion in Francis's QR algorithm,

$$|H(k+1,k)| \leq \epsilon \|H\|_F \quad \text{or} \quad |H(k+1,k)| \leq \epsilon \left(|H(k,k)| + |H(k+1,k+1)|\right),$$

see, e.g., [1], cannot be applied anymore. Watkins and Vandebril use a deflation criterion based on almost diagonal rotators. An almost diagonal rotator is replaced by a diagonal matrix. In the consequence the product of rotators and upper triangular matrices has a zero block below the diagonal and thus the problem can be split into two smaller problems. So far there is no theoretical explanation for the usage of this criterion. In Section 2 we prove that this is indeed a very good strategy, since the relative perturbation of (small) eigenvalues is bounded by the conditioning of the eigenvalue.

*The research was partially supported by the Research Council KU Leuven, projects CREA-13-012 Can Unconventional Eigenvalue Algorithms Supersede the State of the Art (CUASSA), OT/11/055 Spectral Properties of Perturbed Normal Matrices and their Applications, CoE EF/05/006 Optimization in Engineering (OPTEC), and fellowship F+/13/020 Exploiting unconventional QR-algorithms for fast and accurate computations of roots of polynomials, by the DFG research stipend MA 5852/1-1, by the Fund for Scientific Research–Flanders (Belgium) project G034212N Reestablishing Smoothness for Matrix Manifold Optimization via Resolution of Singularities, and by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office, Belgian Network DYSCO (Dynamical Systems, Control, and Optimization).

†Department of Computer Science, KU Leuven, 3001 Leuven (Heverlee), Belgium. ({thomas.mach,raf.vandebril}@cs.kuleuven.be).

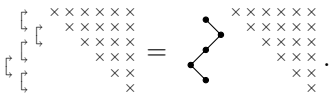
The evolution of the QR algorithm in the last 60 years have brought us new ideas for deciding when and how to deflate the matrix. In particular the aggressive early deflation [6] has lead to reductions of runtime between 5% and 25% on average and up to 75% for special cases [13, 14]. The effect of aggressive early deflation on the convergence was investigated in [15]. The aggressive early deflation has been also used within the QZ algorithm [12], for palindromic eigenvalue problems [16], and more recently for accelerating dqds, a solver for tridiagonal eigenvalue problems [18]. In Section 3 we show how to use the concept of aggressive early deflation in the setting of extended QR algorithms. In Section 4 we present numerical results showing that aggressive early deflation can reduce the number of iterations and the runtime of the extended QR algorithm.

There have been attempts to use the ideas of aggressive early deflation for middle deflations [5]. We also apply the concept of middle deflations to the extended QR algorithm, see Section 5. We present an adaptive determination of deflation window and demonstrate its success for random matrices in numerical experiments. Even if this approach fails for general matrices, we believe that extended Hessenberg matrices are a valuable contribution to the, so far unsuccessful, concept of middle deflations and thus this is of interest for the reader, too.

The article assumes that the reader is familiar with Francis's QR and extended QR algorithms, we do, however, repeat some of the important facts on handling rotations in Subsection 1.2 and give a short sketch of the extended QR algorithm in Subsection 1.3.

1.1. Notation. In this paper we use the following notation: matrices are denoted by upper case letters A, the lower case Greek letters λ, μ stand for eigenvalues, upper case Greek letters Λ for sets of eigenvalues, with $\lambda(A)$ the eigenvalues of A, and lower case letters for complex scalars. In Section 2 we use the lower case letters v, w, x, and y also for eigenvectors. The Hermitian conjugate of a matrix A is denoted by a superscripted H: A^H. The colon notation is used to select submatrices ranging from rows i to j and columns k to ℓ: $A(i : j, k : \ell)$; and the shorthand notation $G(i : j) := G(i : j, i : j)$. If we want to denote the sparsity pattern of a matrix, then we use $\times$ for arbitrary in general non-zero entries and $\otimes$ to highlight particular entries.

1.2. Preliminaries. The extended QR algorithm is based on the QR factorization, where the Q is factorized in $n-1$ rotators. Each of these rotators is a identity matrix, where we replaced a 2×2 submatrix on the diagonal by a unitary matrix $\left[\begin{smallmatrix} c & s \\ -\bar{s} & \bar{c} \end{smallmatrix}\right]$, with $|c|^2 + |s|^2 = 1$. These matrices are often called *Givens* or *Jacobi rotations* [11], for brevity we call them *rotators*. We denote these rotators by $\curvearrowright$, where the arrows point to the rows where we have embedded the 2×2 unitary matrix. The arrows also indicate the rows of a matrix on the right of the rotator that are changed by applying the rotator to this matrix. If the rotators consumes too much space in this notation we compress it: A line shows the zigzag-shape and dot $\bullet$ represents a rotator acting on the neighboring rows, e.g.,



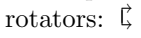
The ordering of the rotators, e.g., symbolized through the black line in the right diagram, is called the *zigzag-shape* or *zigzag-pattern* of the matrix. In this example the pattern contains, from top to bottom, a short *descending* sequence of only two

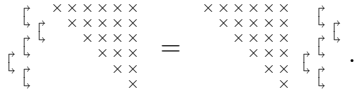
rotators, followed by an *ascending* sequence of three rotators and a final descending sequence of two rotators. The pattern above is also described by the selection vector $\begin{bmatrix}\ell & r & r & \ell\end{bmatrix}$. The first ℓ represents that the rotator above of the second is on the left-hand side of the second. The next r means that the second rotator is on the right-hand side of the third, and so forth.

We further define that the rotator G_i acts on row i and $i+1$, with

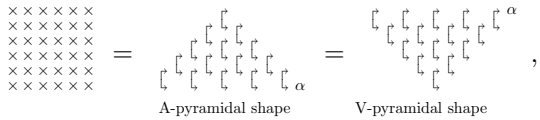
$$G_i(i:i+1) = \begin{bmatrix} c_i & s_i \\ -\overline{s_i} & \overline{c_i} \end{bmatrix}.$$

If a matrix A has a QR decomposition of the form $A = G_{\tau(1)}G_{\tau(2)}\cdots G_{\tau(n-1)}R$, with τ a permutation of $(1,\ldots,n-1)$ and R an upper triangular matrix, then we call this matrix an *extended Hessenberg matrix* because of the link to extended Krylov subspaces, see, e.g., [17]. Further we call A reduced if at least one G_i is a diagonal matrix, meaning $\exists i \in \{1,\ldots,n-1\} :\ s_i = 0$.

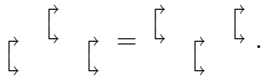
The factorized QR representation of a matrix is a powerful representation, since we can perform several operations with the rotators. We will now briefly mention the most important ones. The product of two rotations is again a rotation; this means that we can fuse two rotators:  , where the tiny arrow marks the fused rotation. If one applies the rotator G_i from the left to an upper triangular matrix, then we generate a bulge in position $(i+1,i)$. By applying a rotation $\tilde{G}_i$ from the right, which acts on the columns i and $i+1$, one can eliminate this bulge. In this manner a whole zigzag-shape can be passed one by one through the upper triangular matrix, e.g.,



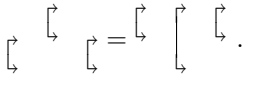
Every unitary matrix $Q \in \mathbb{C}^{n\times n}$ can be factored in $\frac{1}{2}n(n-1)$ rotators. These rotators can be arranged in, e.g., two pyramidal shape [20]:

$Q =$ 

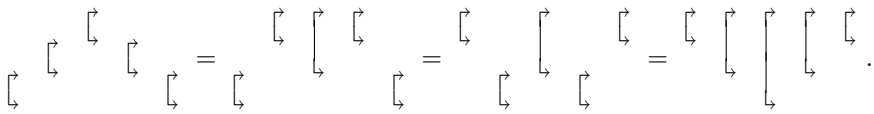
where α stands for an identity matrix where one diagonal entry is replaced by $\alpha = det(Q)$ with $|\alpha| = 1$. The α is not part of the pyramidal shape. If the pyramidal shape points up, then we called it an A-pyramidal shape, otherwise a V-pyramidal shape. The special case for $n = 3$ provides an important operation; we can turnover a pattern of three rotations, like



In this paper we will also use another transformation of three rotations, which has not been used in the context of extended QR algorithms. We will use that



Applying this identity twice yields



If we have a unitary matrix

$$\begin{bmatrix} \times & a & b \\ \times & \times & c \\ \times & \times & \times \end{bmatrix},$$

then we use the first rotator to annihilate c, the second for the updated b and the last one for the updated a. The remaining matrix is a unitary, upper triangular matrix and thus diagonal. We will call this *wedge-to-broadhead* transformation. Analog to the pyramidal shapes we call the wedge above an *A-wedge* and the broadhead an *V-broadhead*, so that the transformation is an A-wedge-to-V-broadhead one.

1.3. Extended QR Algorithm. In this subsection we will provide a short sketch of the extended QR algorithm. For details we refer the reader to [20] and [22]. The first step in the extended multishift QR algorithm is to bring the matrix in extended Hessenberg form. Then we perform implicit QR steps until all eigenvalues are found. Such an implicit QR step consists of the bulge generation, the bulge chasing, and finally the removal of the bulge, by merging it into the zigzag-pattern.

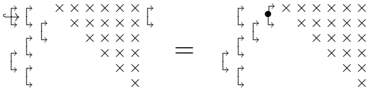
For the bulge generation we fix a polynomial, e.g.,

$$p(A) = (A - \rho_1 I)(I - \rho_2 A^{-1}) \cdots (A - \rho_s I),$$

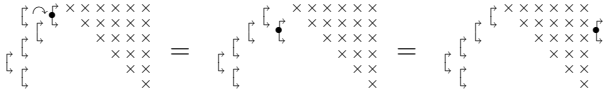
where we have to choose between positive and negative powers of A depending on the zigzag-shape. The s shifts can be determined from the trailing $s \times s$ submatrix of A. We then compute $x = \beta p(A) e_1$, with e_1 the first column of the identity matrix. Based on x we compute s rotators $S_1, \ldots, S_s$ such that

$$S_1^{-1} \cdots S_s^{-1} x = \gamma e_1.$$

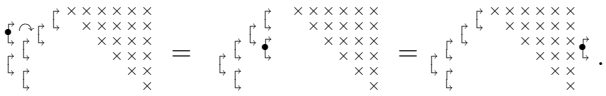
For simplicity we will restrict the further explanation to the single shift case, $s = 1$. Applying the bulge generating similarity transformation to the matrix A gives:

$$S_1^{-1} A S_1 = S_1^{-1}$$  $$S_1 =$$  ,

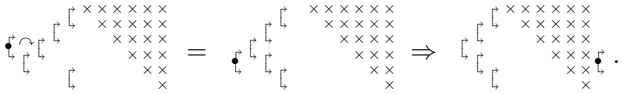
of which the left rotator gets fused and the right is brought to the left and become the bulge, marked by a dot. This bulge is now chased to the bottom of the zigzag-shape, by turnovers, similarity transformations, and passing rotators through the upper triangular matrix. In our example this means:

$$S_1^{-1} A S_1 =$$  .

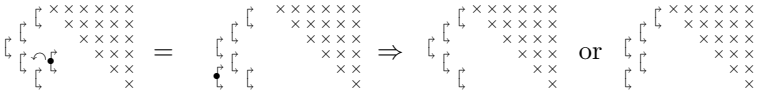
By a similarity transformation we bring the bulge to the other side, do again a turnover, and a pass though operation:



In the next row the bulge reaches the second bend, where the zigzag-shape changes direction. We will see that bends move up one row every time a bulge rotator passes them. This means that the extended QR algorithm preserves the structure of an extended Hessenberg matrix in a weak sense, since the next iterate is again of extended Hessenberg form, but with a different zigzag-pattern. Now the next chasing steps:



In the next row we have to make a last turnover. Afterwards, we merge the two rotators in the last row to let the bulge disappear. Here we have the freedom to choose where we want to place the fused rotator: on the left or right side of the zigzag-shape. We have



The decision where to merge the last rotator influences the convergence properties of the algorithm. In [21] it has been shown that depending on the decision the convergence rate is either

$$\frac{|\lambda_n - \rho_1|}{|\lambda_{n-1} - \rho_1|} \quad \text{or} \quad \frac{\left|\lambda_n^{-1} - \rho_1^{-1}\right|}{\left|\lambda_{n-1}^{-1} - \rho_1^{-1}\right|}.$$

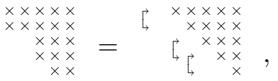
Using the optimal shape can result in almost halving the number of iterations for certain examples, see [20]. After we have done several QR iterations some of the eigenvalues are converged. Of course we do not want to use shifts approximating already converged eigenvalues. Thus we need to deflate these eigenvalues from the matrix. This will be explained in details in the next section.

2. Standard Deflation. Infinitely many QR steps lead to convergence to a block upper triangular matrix. To reduce the matrix size and to free converged shifts one has to deflate (converged) eigenvalues from the matrix. A reduced extended Hessenberg matrix allows to split the eigenvalue problem into two smaller ones. Since our matrix converges to a reduced Hessenberg matrix, but stays often unreduced, small perturbations are necessary to deflate. In Francis's QR algorithm one searches for small subdiagonal entries in the Hessenberg matrix H, which are set to zero. Typical conditions are

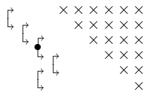
$$|H(k+1,k)| \leq \epsilon \left\|H\right\|_F \quad \text{or} \quad |H(k+1,k)| \leq \epsilon \left(|H(k,k)| + |H(k+1,k+1)|\right), \quad (2.1)$$

where the latter is tighter and shows better relative accuracy for small eigenvalues, see, e.g., [13].

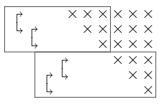
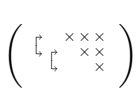
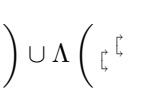
In the extended QR algorithm the extended Hessenberg matrix is only available in the factorized form. This makes it expensive to test for small subdiagonal entries or submatrices $H(k:n, 1:k)$ of small norm as one has to compute these elements or submatrices explicitly. Since the QR decomposition of a reduced Hessenberg matrix is for instance

$A =$  ,

with $G_2 = I$, we can search for almost diagonal G_i. In [20] this deflation criterion is used, but without presenting a theoretical explanation why it provides good relative accuracy. Suppose we have an extended Hessenberg matrix in the form

 ,

where the dot-marked rotator is the identity, then we can deflate this eigenvalue problem into two smaller problems

$\Lambda\left(\right.$  $\left.\right) = \Lambda\left(\right.$  $\left.\right) \cup \Lambda\left(\right.$  $\left.\right).$

In the extended QR algorithm one searches for almost diagonal rotators. These rotators are scaled by a diagonal matrix so that they are almost equal to the identity. Then the matrix is changed by setting the off diagonal entries to zero. This means we replace

$$A = G_{\tau(1)} G_{\tau(2)} \cdots G_{\tau(n-1)} R \tag{2.2}$$

with

$$\tilde{A} = G_{\tau(1)} G_{\tau(2)} \cdots G_{\tau(i-1)} G_{\tau(i+1)} \cdots G_{\tau(n-1)} R \tag{2.3}$$

if $\left\| I - G_{\tau(i)} \right\|_2 \leq \epsilon$. Of course this perturbs the eigenvalues, but this perturbation to the spectrum of A can be bounded. The following series of lemmas, theorems and corollaries contain bounds that are increasingly tight for well conditioned eigenvalues of small magnitude. The first two should be consider as a "training camp" for the following ones. We start with a simple application of the Bauer-Fike theorem.

LEMMA 2.1. *Let A and $\tilde{A}$ be defined as in (2.2) and (2.3). Let further A be diagonalizable, $A = X\Lambda X^{-1}$, and λ an eigenvalue of $\tilde{A}$. Then there exists an eigenvalue μ of A, so that*

$$|\lambda - \mu| \leq \kappa_2(X) \|G_{\tau(i)} - I\|_2 \|A\|_2, \tag{2.4}$$

where $\|\cdot\|_2$ is the spectral norm and $\kappa_2(X)$ the condition number of X and hence the condition number of the eigenvalue problem in the 2-norm.

Proof. As shown in [3], known as the Bauer-Fike theorem, we have

$$\min_{\mu \in \Lambda(A)} |\lambda - \mu| \leq \kappa_2(X) \|A - \tilde{A}\|_2,$$

with $X^{-1}AX$ the Jordan normal form of A. By using the structure of A and $\tilde{A}$ we get

$$\begin{aligned} \|A - \tilde{A}\|_* &\leq \|G_{\tau(1)} \cdots G_{\tau(i-1)}\|_* \, \|G_{\tau(i)} - I\|_* \, \|G_{\tau(i+1)} \cdots G_{\tau(n-1)}\|_* \, \|R\|_* \\ &\leq \|G_{\tau(i)} - I\|_* \, \|R\|_* \\ &\leq \|G_{\tau(i)} - I\|_* \, \|A\|_*, \end{aligned} \tag{2.5}$$

where $\|\cdot\|_*$ is a unitarily invariant norm, e.g., $\|\cdot\|_2$. Combining both inequalities completes the proof. ∎

Hence we have a good absolute perturbation bound. But with a little additional effort we can improve this to a relative perturbation bound as the following lemma shows.

LEMMA 2.2. *Let A and $\tilde{A}$ be defined as in (2.2) and (2.3). Let further A be nonsingular and diagonalizable $A = X\Lambda X^{-1}$ and λ be an eigenvalue of $\tilde{A}$. Then there exists an eigenvalue μ of A, so that*

$$\frac{|\lambda - \mu|}{|\mu|} \leq \kappa_2(X) \left\| G_{\tau(i)} - I \right\|_2, \tag{2.6}$$

where $\|\cdot\|_2$ is the spectral norm and $\kappa_2(X)$ the condition number of X and hence the condition number of the eigenvalue problem of A in the 2-norm.

Proof. In [8, Theorem 2.3] it has been shown that for diagonalizable A, with $A = A_1 A_2$, it holds that

$$\min_i \frac{|\lambda - \mu_i|}{|\mu_i|} \leq \kappa_2(W) \left\| A_1^{-1} E A_2^{-1} \right\|_2, \tag{2.7}$$

where W is the eigenvector matrix of $A_2 A_1$ and $E = A - \tilde{A}$. We set $A_1 = G_{\tau(1)} \cdots G_{\tau(i)}$ and $A_2 = G_{\tau(i+1)} \cdots G_{\tau(n-1)} R$ and thus have $A_1^{-1} E A_2^{-1} = I - G_{\tau(i)}^{-1}$. The unitarily invariance of the 2-norm ensures that

$$\|I - G_{\tau(i)}^{-1}\|_2 = \|G_{\tau(i)}^{-1}(G_{\tau(i)} - I)\|_2 \leq \|G_{\tau(i)} - I\|_2.$$

Since $A_1 A_2 = A = X\Lambda X^{-1}$ we have

$$A_2 A_1 = A_1^{-1} A_1 A_2 A_1 = A_1^{-1} X \Lambda X^{-1} A_1$$

and hence $W = A_1^{-1} X$. Since A_1^{-1} is unitary $\kappa_2(W) = \kappa_2(X)$. ∎

The estimation in Lemma 2.2 is a relative bound, but still the error estimator can be entirely wrong for matrices with well conditioned small eigenvalues and bad conditioned large eigenvalues. The bad conditioned eigenvalues lead to a bad conditioning of the whole eigenvalue problem. This weakness will be healed with the following theorem. We use the same approach as in [8, Corollary 2.2 and Theorem 2.3].

It is well known that the perturbation of single eigenvalues can be estimated with the following linearization, see, e.g., [11] or [23].

THEOREM 2.3. *Let μ be an eigenvalue of A with multiplicity 1, having right eigenvector x, and left eigenvector y. Let E be a perturbation of the form ϵF, with $\|F\|_2 = 1$. For $\mu = \mu(\epsilon) \in \Lambda(A + E)$ with $\mu(0) = \lambda$, we have*

$$|\mu - \lambda| \leq \frac{\|E\|_2}{y^H x} + \mathcal{O}(\|E\|_2^2).$$

In (2.5) we have seen that for our deflation perturbation we have

$$\|E\| = \|A - \tilde{A}\|_2 \leq \left\| G_{\tau(i)} - I \right\|_2 \|A\|_2,$$

and thus

$$|\mu - \lambda| \leq \frac{\left\| G_{\tau(i)} - I \right\|_2 \|A\|_2}{y^H x} + \mathcal{O}\left(\left\| G_{\tau(i)} - I \right\|_2^2 \|A\|_2^2 \right).$$

Analog to [8, Corollary 2.2] we make a relative estimator out of Theorem 2.3.

THEOREM 2.4. *Let A be invertible and λ an eigenvalue of A with multiplicity 1, right eigenvector x, and left eigenvector y. Let further E be a perturbation of the form ϵF, with $\|F\|_2 = 1$. For $\mu = \mu(\epsilon) \in \Lambda(A+E)$ with $\mu(0) = \lambda$, we have*

$$\frac{|\mu - \lambda|}{|\lambda|} \leq \frac{\left\|A^{-1}E\right\|_2}{y^H x} + \mathcal{O}(\left\|A^{-1}E\right\|_2^2).$$

Proof. Let $(\hat{x}, \mu)$ be an eigenpair of $A+E$, so that $(A+E)\hat{x} = \mu\hat{x}$. This can be reformulated as $\hat{x} = \mu A^{-1}\hat{x} - A^{-1}E\hat{x}$. We set $\overline{A} = \mu A^{-1}$ and $\overline{E} = -A^{-1}E$. Hence we have

$$(\overline{A} + \overline{E})\hat{x} = \hat{x}.$$

Thus 1 is an eigenvalue of $\overline{A} + \overline{E}$. The eigenvalues of $\overline{A}$ are μ/λ_i with the same eigenvectors as A. We apply Theorem 2.3 to $\overline{A}$, $\overline{A} + \overline{E}$, and the eigenvalue 1. ∎

In the next theorem we use the argumentation from [8, Theorem 2.3] to get a similar result.

THEOREM 2.5. *We make the same assumptions as in Theorem 2.4. If $A = A_1 A_2$ and A_1 and A_2 invertible, then*

$$\frac{|\mu - \lambda|}{|\lambda|} \leq \frac{\|A_1^{-1}EA_2^{-1}\|_2}{y^H x} + \mathcal{O}\left(\|A_1^{-1}EA_2^{-1}\|_2^2\right).$$

Proof. We set $\hat{A} = A_2 A A_2^{-1}$ and $\hat{E} = A_2 E A_2^{-1}$. The eigenvalue μ is also an eigenvalue of $\hat{A}$ with right eigenvector $v = A_2 x$ and left eigenvector $w^H = y^H A_2^{-1}$. We apply Theorem 2.4 and get

$$\frac{|\mu - \lambda|}{|\lambda|} \leq \frac{\|\hat{A}^{-1}\hat{E}\|_2}{w^H v} + \mathcal{O}\left(\|\hat{A}^{-1}\hat{E}\|_2^2\right).$$

Since $\hat{A}^{-1}\hat{E} = A_2 A^{-1} E A_2^{-1} = A_1^{-1}EA_2^{-1}$ and $w^H v = y^H A_2^{-1} A_2 x = y^H x$ we have thus proved the theorem. ∎

We now apply this to the same A_1 and A_2 as in Lemma 2.2.

COROLLARY 2.6. *Let A and $\tilde{A}$ be defined as in (2.2) and (2.3). Let further A be nonsingular, and let λ be a single eigenvalue of $\tilde{A}$. Then there exists an eigenvalue μ of A, so that*

$$\frac{|\mu - \lambda|}{|\lambda|} \leq \frac{\|I - G_{\tau(i)}\|_2}{x^H y} + \mathcal{O}\left(\|I - G_{\tau(i)}\|_2^2\right),$$

with x and y as above.

Proof. We set $A_1 = G_{\tau(1)} \cdots G_{\tau(i)}$ and $A_2 = G_{\tau(i+1)} \cdots G_{\tau(n-1)} R$ and thus have $A_1^{-1} E A_2^{-1} = I - G_{\tau(i)}^{-1}$. We finally apply Theorem 2.5. ∎

Lemma 2.2 and Corollary 2.6 provide us optimal relative perturbation bounds and estimations respectively for deflations based on almost diagonal rotators. The bounds depend only on the norm of the perturbation and a condition number of the individual eigenvalue. Hence we now know that the deflation criterion based on almost diagonal rotators is not perturbing the eigenvalues more than necessary.

This deflation procedure is simple and additionally provides perfect relative accuracy. Hence it is not necessary to take other rotators, R, or even A into consideration.

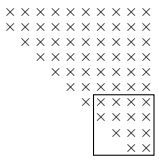
For the standard deflation procedure in Francis's QR algorithm based on the deflation criteria (2.1) one can also provide bounds on the perturbation matrix E. These bounds can be used to assess the absolute accuracy of the eigenvalues by applying the Bauer-Fike theorem or Theorem 2.3. However, [8, Theorem 2.3] leading to the relative bound in (2.7) and Theorem 2.5 cannot be applied here, since in general A, $\tilde{A}$, and $E = A - \tilde{A}$ do not have a common factorization differing only in one factor.

Finally, we would like to draw the attention of the reader to the fact that in dqds and MRRR one is also replacing a matrix, here a tridiagonal, by its factorization, here a LDU, LDL^T, or LU factorization, to improve the relative accuracy of the computed eigenvalues, see, e.g., [19]. In the extended QR algorithm we do somehow the same. We also replace the matrix, here an extended Hessenberg matrix, by its factorization, here the QR decomposition. The primary aim was not the improvement of the accuracy but of the convergence rate. However, the theorems above show that the relative accuracy is not reduced by the deflation.

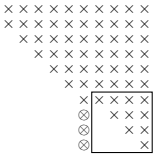
3. Aggressive Early Deflation. In state-of-the-art implementations of Francis's multishift QR algorithm one uses besides standard deflations also aggressive early deflations, which reduce the required CPU time on average by about 15% [13]. In this section we will transcribe the ideas of aggressive early deflation to extended QR algorithms. Therefore we start with a short description of aggressive early deflation for Hessenberg matrices. A detailed description can be found in [6].

3.1. Aggressive Early Deflation for Hessenberg Matrices. The concept of aggressive early deflation is, among others, based on the observation that a series of small but non-negligible subdiagonal entries may lead to deflatable, converged eigenvalues, even if none of the subdiagonal entries is below the deflation threshold. In the Hessenberg form every row is coupled with the neighboring rows by the subdiagonal element. The aggressive early deflation replace this coupling between neighbors of the last, say m rows by a coupling with a single, namely the $(n-m)$th row, making it easier to deflate rows in the trailing $m \times m$-submatrix.

Assume an upper Hessenberg matrix

$A =$  ,

the marked rectangle ($m = 4$) is the deflation window. The marked submatrix is $S := A(n-m+1:n)$ and has the Schur factorization $S = VTV^H$. We embed the unitary matrix V in an identity matrix and call the large matrix Q. Applying Q to A yields

$A' = Q^H A Q =$  with $Q = \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & & V \end{bmatrix}$.

Obviously this does not affect the eigenvalues. The matrix A' is of *upper Hessenberg plus spike* form, where the circled entries form the spike. This spike makes the main difference between A and A', since in A' the last four rows are only coupled with the

last row in front of the deflation window. This allows to remove any of these rows if the entry in the spike is small. Note that a small entry in the spike corresponds to a converged eigenvalue. The main advantage is that one deflates earlier. Thus the next iteration is cheaper since the new matrix is smaller. Furthermore already converged shifts will not be used again in the next iteration and new shifts can be used to force the convergence to other eigenvalues. This is in particular important in multishift implementations.

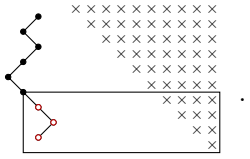
Essentially the aggressive early deflation consists of the following five steps:

1. Choose and compute a trailing submatrix S of A.
2. Compute the Schur decomposition of $S = V^H T V$.
3. Apply V to A and compute the "spike".
4. Deflate within the "spike".
5. Restore the original form.

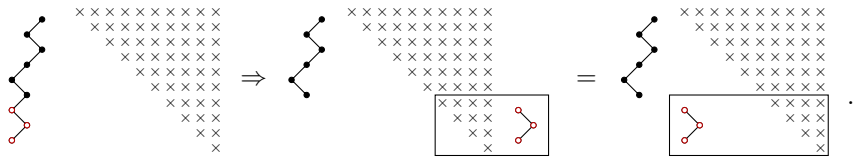
Aggressive early deflation is more expensive compared to standard deflation. Applying the matrix V from the Schur decomposition to A and restoring the original form involve updates of the last m columns of A by an $m \times m$ matrix, leading to a complexity of $\mathcal{O}(nm^2)$.

3.2. Aggressive Early Deflation for Extended Hessenberg Matrices. We will now transcribe the five steps of aggressive early deflation one-by-one. The first steps are almost the same for extended Hessenberg matrices. Computing the Schur decomposition of a small matrix in QR decomposition is done recursively by an extended QR algorithm. The spike will be replaced by a special pattern revealing the coupling properties of the spike and allowing to easily deflate the converged eigenvalues.

1. Computing S. Assume our matrix A has the form

$A =$  .

The rectangle is again the deflation window. The rotators inside the deflation window are marked with red dots, ○. If the trailing rotators are on the left of the last black rotator, then one can perform a similarity transformation to bring the trailing rotators to the other side followed by passing them through the upper triangular. This is shown in this example:

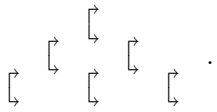
$A =$  .

Thus one can easily ensure that the trailing rotators are between the previous rotator and the upper triangular matrix.

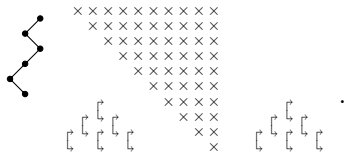
2. Schur Decomposition of S. Compute the Schur decomposition for this submatrix by using the extended QR algorithm without aggressive early deflation:

$$S = \begin{array}{c} \curvearrowright \\ \\ \curvearrowright \end{array} \begin{array}{c} \curvearrowright \\ \\ \\ \end{array} \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix} = VTV^H.$$

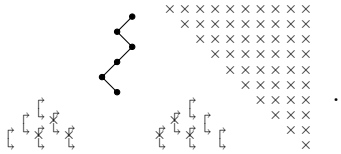
As mentioned in Subsection 1.2 the matrix V can be factored in rotators arranged in a pyramidal shape:

$V =$ 

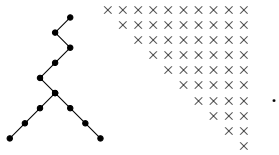
3. Computing the Spike. Replace now the lower right submatrix by its Schur decomposition, and write A as

$A =$ 

By a similarity transformation one can bring the pyramid on the right-hand side to the left,

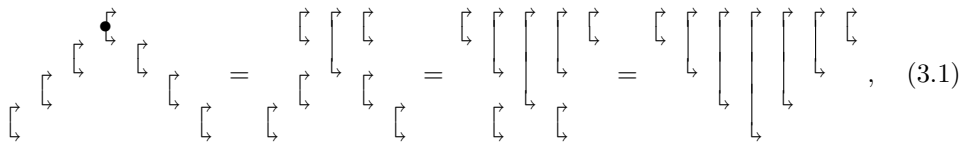
$A' =$ 

Since the right pyramid is the Hermitian transpose of the left pyramid the rotators marked with a cross annihilate each other. We finally end up with the following scheme

$A' =$ 

The remaining most outer rotators of V and V^H form a large A-wedge and would vanish if the last rotator in front of the deflation window would be the identity. So they are the analog to the spike in the aggressive early deflation in the Hessenberg QR algorithm, which vanishes if the subdiagonal entry at the top of the spike is zero. The rows are still coupled with their neighbors, but this will be changed now.

We use the A-wedge-to-V-broadhead transformation recursively and end up with



(3.1)

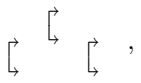
where the last rotator partially outside the deflation window is marked with a dot. The middle rotation becomes often relatively soon close to an identity[1] and thus can be deflated. We will call the large broadhead at the end also a V-broadhead and the complete transformation also an A-wedge-to-V-broadhead transformation.

The advantage of the V-broadhead is that the trailing rows are again linked only with the top row and no longer with their neighbors.

[1] A side effect is that for an efficient implementation it is necessary to store and handle $c_i - 1$, with c_i the diagonal entry of this rotation, separately to avoid cancellation.

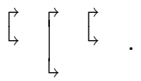
4. Deflate within the "Spike". We now have a broadhead, our replacement for the spike, where the trailing rows of A are sufficiently uncoupled to easily deflate the corresponding eigenvalues. We start with the rotator in the middle of the broadhead shape. If this rotation is almost diagonal, then we deflate this rotator and merge the next inner rotators. This is repeated until the rotator in the middle is no longer almost diagonal. We now test the other rotators. Since the kth rotation is close to diagonal if and only if the $q+1-k$th rotation, q the number of rotators left, is close to diagonal, as Lemma 3.1 shows, it is sufficient to test half of the remaining rotators. This means that searching for converged rows in the broadhead shape is not more expensive then in the spike in the Hessenberg case.

LEMMA 3.1. *Assume three rotators are given*



(3.2)

where the left rotation is the Hermitian transpose of the right one. These rotators can be transform into



(3.3)

The left and right rotation of this new decomposition of the unitary matrix are both diagonal or none of the rotations is diagonal.

Proof. Let the rotations in (3.2) be

$$\begin{bmatrix} 1 & & \\ & c & s \\ & -\overline{s} & \overline{c} \end{bmatrix} \begin{bmatrix} d & t & \\ -\overline{t} & \overline{d} & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & \\ & \overline{c} & -s \\ & \overline{s} & c \end{bmatrix}$$

and

$$\begin{bmatrix} e & u & \\ -\overline{u} & \overline{e} & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & v \\ & 1 & \\ -\overline{v} & & \overline{f} \end{bmatrix} \begin{bmatrix} g & w & \\ -\overline{w} & \overline{g} & \\ & & 1 \end{bmatrix}$$

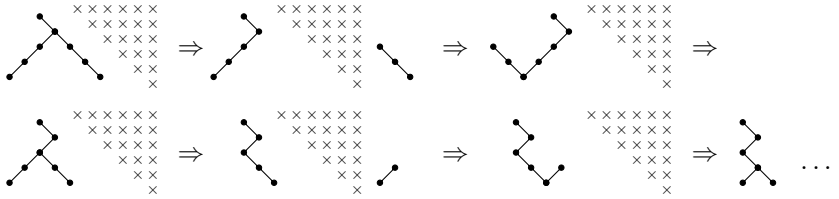
the rotations in (3.3). We have two factorizations of the same matrix. Multiplying them out yields $e = \overline{g}$. If $|e| = |g| = 1$, then $|u| = |w| = 0$, since both factors are rotators. ∎

Removing almost diagonal rotators reduces the size of the eigenvalue problem. The perturbation error is bounded in the same way as described in Section 2.

For small m we observed in almost all examples that the middle rotator is diagonal if at least one rotator is diagonal. This makes the reduction easy, since we can start in the middle. After removing the middle rotator the next two are merged and the product is tested again. This is different for larger m. For large m we sometimes observe that other rotators are diagonal and the middle one is not. This makes reordering of the eigenvalues in the Schur factorization and a recomputing of the wedge and the broadhead shaped rotators necessary. This is expensive and has not been implemented for the numerical experiments in the following section.

5. Restore the Original Form. After we have deflated all diagonal rotators we have to restore the original pattern of rotators. Therefore we transform the broadhead back into the wedge form. This is simply the reverse process from the one in step 3.

The wedge consist of an ascending and a descending sequence of rotators. These two sequences have to be merged into one zigzag-shape according to the selection vector. This is now explained by an example, which is also visualized in the next diagram. Let the selection vector of the trailing rotations start with an r. Then we have to pass the descending sequence through the upper triangular and bring the rotators by similarity transformation to the other side. The rotators in the last row of both sequences are merged. This reduces the length of the descending sequence by one. Afterwards we pass the descending sequence through the ascending sequence. We now have fixed the first of the trailing rotators in the correct position. Let the next entry in the selection vector be an ℓ. Then we bring the ascending sequence to the other side, pass it through the upper triangular, merge the last rotators, and bring the ascending sequence into the original position by turnovers. These steps are summarized in the following diagram, which only shows the last $m+1$ rows/columns of the matrix:



The description above shows that for extended Hessenberg matrices some steps of aggressive early deflation are more complicated than for Hessenberg matrices. However, the complexity is the same. The most expensive computations are the application of V to A and the restoring of the zigzag-shape in the trailing rotators. The application of V updates the trailing m columns with an $m \times m$ matrix, which takes $\mathcal{O}(nm^2)$ operations. During the restoring one has to transfer up to $\frac{1}{2}m(m-1)$ rotators through the upper triangular matrix R. Each of these transfers changes $\mathcal{O}(n)$ entries in R. Hence the costs are also bounded by $\mathcal{O}(nm^2)$. If the eigenvectors are needed, then we have to update a matrix storing all unitary transformation while we bring the rotators to the other side, this can also be done in $\mathcal{O}(nm^2)$ operations. All the other computations involve at most m^2 rotators and the trailing $m \times m$ submatrix of R. Under these computations the Schur decomposition and the decomposition of V into $\frac{1}{2}m(m-1)$ rotators are the most expensive, both $\mathcal{O}(m^3)$.

3.3. Predicting Deflations. The costs of aggressive early deflation are bounded by $\mathcal{O}(nm^2)$. This is compared to the $\mathcal{O}(n^2)$ operations per iteration cheap, but, nevertheless, we want to perform these computations only if the probability of a deflation is high enough. We will now briefly repeat the argumentation from [6, Section 2.8] for the Hessenberg case, which delivers the absolute value of the last entry in the spike. Based on this argumentation we show a heuristic to predict how many deflations one can expect within a given deflation window.

We summarize the most important steps in this paragraph. First, one takes the Schur decomposition of $S = VTV^H$. One can show that the last entry of the spike is $z = A(n-m+1, n-m)V(1, m)$. Let $T(n, n) = \lambda$ be an eigenvalue of S. Then $S-\lambda I$ is a singular, unreduced Hessenberg matrix. Further, let UR be the QR decomposition of $S - \lambda I$. The U is also an unreduced Hessenberg matrix and can be written as a descending series of rotators $G_1^S \cdots G_{m-1}^S$. Since the last column of U and the last

column of V are both normalized left eigenvectors of S, we have $|U(1,k)| = |V(1,k)|$. Thus the absolute value of z is

$$|z| = |A(n-m+1, n-m)| \left| \prod_{i=1}^{m-1} s_i \right|,$$

where s_i is the off-diagonal entry of G_i^S. If $\left|\prod_{i=1}^{m-1} s_i\right|$ is small, say smaller than machine precision, then one can deflate the λ from A.

In [6] the argumentation is now continued to link the product to the Hessenberg matrix A. This is not useful for our purpose, since in the extended QR algorithm we have a QR decomposition of S, but not of $S-\lambda I$, available. The following heuristic is based on the QR decomposition of S, since the computation of the QR decomposition of $S-\lambda I$ is too expensive. We take the QR decomposition of A, which gives us also a QR decomposition of S, and compute

$$p = \prod_{i=n-m}^{n-1} |s_i| \,. \tag{3.4}$$

We observe that aggressive early deflation is successful for small values of p. For ϵ equal to double precision accuracy, $n = 200$, and $m = 25$, we observed for $p >$1e$-$13 never an additional deflation opportunity. However, $p <$1e$-$18 was almost every time followed by a deflation. In the range in between we observed that deflations become rarer with growing p. These indicating values of p might depend, among others, on n, m, and the deflation accuracy ϵ.

We have no complete theoretical explanation for the use of p, however, we want to give the following heuristic explanation: Let $A(n-m+1:n)$ be the trailing submatrix we investigate. Every rotator has the form

$$G_i = \begin{bmatrix} c_i & s_i \\ -\overline{s_i} & \overline{c_i} \end{bmatrix}.$$

This means we multiply a row with a factor of absolute value $|c_i|$ and add an $|s_i|$ multiple of the neighboring row. Thus the value p is an indicator for the coupling of the last row with the $(n-m)$th row. During the aggressive early deflation procedure we change the matrix in a way that the rows from $n-m+1$ to n are only coupled with row $n-m$. Thus the value p tells us when this coupling is weak and exactly such a weak coupling is the basis for the deflation.

4. Numerical Experiments. In this section we use the MATLAB® implementation of the multishift QR algorithm that has been used in [21]. We check for standard deflation in all rotations with a threshold of 4.4409e$-$15. The aggressive early deflation additionally tests the trailing $m \times m$ part for deflation opportunities. The used random matrices are generated by the MATLAB command `rand(n,n)+sqrt(-1)*rand(n,n)`. These matrices have random complex entries with real and imaginary part between 0 and 1. We choose a random zigzag-shape for the rotators, where the direction is changed with a probability of 30%. The MATLAB function `schur` is used to compute the Schur decomposition in the aggressive early deflation. As soon as the deflation makes the matrix smaller than a predefined size we stop the process and compute the remaining eigenvalues using a plain double shift implementation of the extended

QR algorithm. The finally remaining 2×2 blocks on the diagonal are triangularized using the MATLAB function `eig`. The tests have been performed on an Intel® Core™i5-3570 (3.40GHz). All the presented results are averages over three runs.

It is known that extended QR algorithms suffer from shift blurring analog to the shift blurring observed in the QR algorithm, see [21]. For testing the aggressive early deflation we need multishift steps with a large number of shifts, since the aggressive early deflation benefits from simultaneous convergence to many eigenvalues. To avoid shift blurring we compute s shifts at once, but chase only bulges of size 6 or smaller. Hence we chase in each iteration $s/6$ times a bulges of size 6. Further implementations might improve this by packing the bulges tighter and using blocked updates as it is done in recent implementations of Francis's QR algorithm.

We test our algorithm with different matrix dimensions. For $n = 50$ we did not observe an advantage using aggressive early deflation. For $n = 500$ we stop the usage of aggressive early deflation by a size of 250. In Figure 4.1(a) the number of iterations before aggressive early deflation is turned off multiplied with the number of shifts is shown for different pairs of m, the number of shifts, and s, the size of the deflation window. This product is a measure for the number of bulges that are chased through the matrix. One can see that more shifts in each step increase this number, while aggressive early deflation reduces the number. Chasing more bulges requires more time and so we see this effect also in Figure 4.1(b), where the runtime is plotted. Larger deflation windows have two effects, they reduce the number of iterations but increase the costs of the deflation. The point with the minimal number of iterations corresponds to one of the highest runtimes. We have marked the points from Table 4.1 with black asterisks. The size m of the deflation windows should be chosen between s and $2s$, where s is the number of shifts, to achieve good performance.

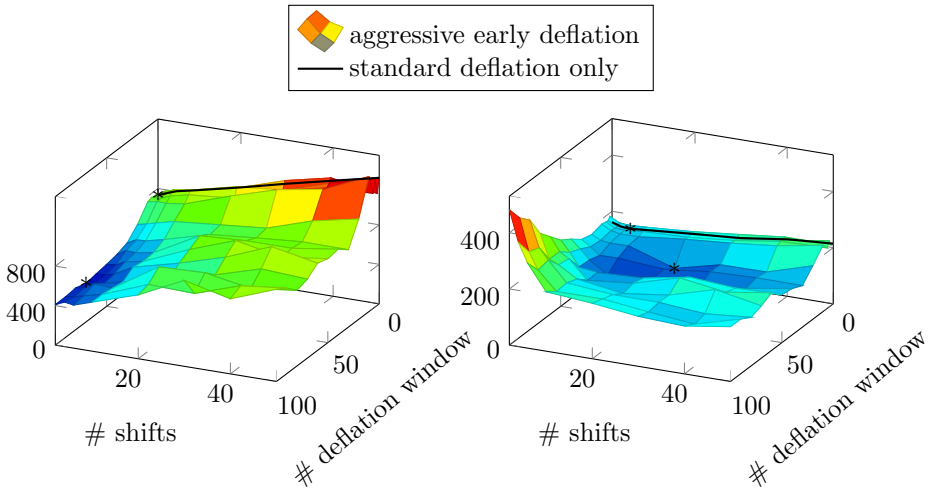
For $n = 100$ we observed a similar behavior as for $n = 500$, see Figure 4.2. The advantage of aggressive early deflation is, however, smaller. This is in accordance with the description of the LAPACK 3.1 implementation of aggressive early deflation for Hessenberg matrices [7]. In this implementation aggressive early deflation is turned off for matrices of size 75×75 or smaller.

The shortest runtime and the lowest number of iteration without aggressive early deflation is compared with the best results with aggressive early deflation in Table 4.1, where s is the number of shifts and m the size of the deflation window. In the best cases of aggressive early deflation we can save up to a third of the runtime. Since we used a MATLAB implementation partially using much more optimized functions, such as `schur`, these results have to be considered preliminary. A high level implementation of the whole algorithm might show a smaller advantage of aggressive early deflation.

5. Middle Deflation.

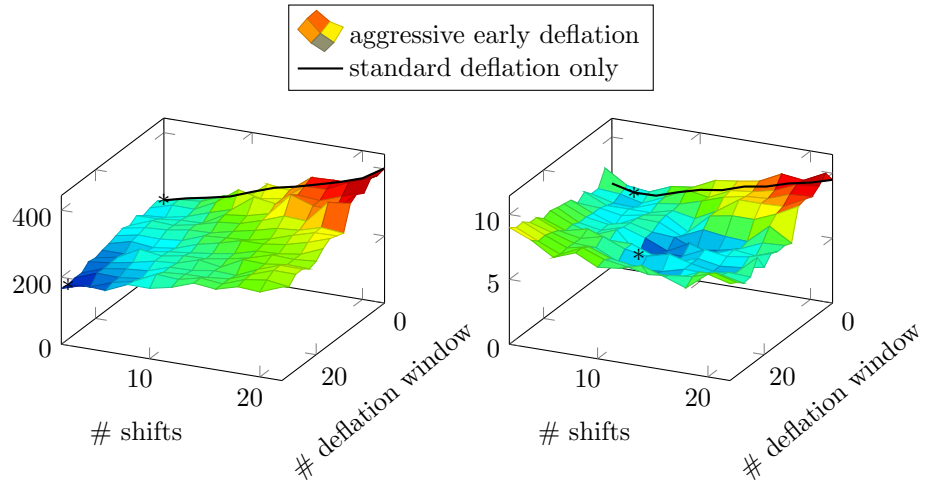
5.1. Middle Deflation for Hessenberg Matrices. The ideas of aggressive early deflation have also been applied to middle deflations [5]. We start again with a short description of the middle deflation for Hessenberg matrices. The core ideas are the same as for aggressive early deflation, but the possible gain is much higher. The main advantage is that a middle deflation splits the matrix into two smaller ones of comparable size. This makes a middle deflation much more precious than a deflation at the end of a matrix, since the complexity of the QR algorithm is cubic. Hence, halving the problem reduces the costs of the remaining problems from $\mathcal{O}(n^3)$ to $2 \cdot \mathcal{O}((n/2)^3)$ and thus by a factor of 4.

In the Hessenberg case one takes out a submatrix from the middle of the matrix, computes the Schur decomposition, and applies the similarity transformation from



(a) Number of Iterations · shifts. (b) Runtime in s

Fig. 4.1: The effect of aggressive early deflation for $n = 500$.



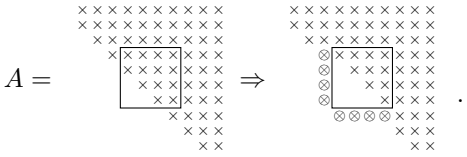
(a) Number of Iterations · shifts. (b) Runtime in s

Fig. 4.2: The effect of aggressive early deflation for $n = 100$.

	best without early deflation			best with early deflation				savings
n	s	t	#it·s	s	m	t	#it·s	
100	4	6.00 s	215	10	18	5.62 s	223	6% time
100	2	6.42 s	198	2	28	8.60 s	163	18% iterations
500	6	148.49 s	803	20	20	100.45 s	727	32% time
500	2	161.38 s	732	2	70	314.85 s	391	46% iterations

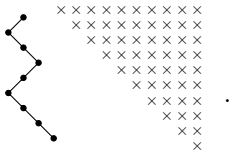
Table 4.1: The effect of aggressive early deflation.

the unitary factors of the Schur decomposition to the whole matrix. This destroys also the Hessenberg structure of the matrix and we get the following shape with two spikes:



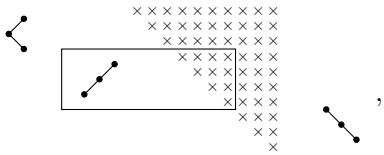
The two spikes are formed by the circled entries $\otimes$ of the matrix. If in our example at least four of these entries are small enough, then we can deflate the problem to two smaller ones. Unfortunately, such deflations are rare and hard to detect. Also shift strategies enforcing the occurrence of middle deflation are, to the authors' best knowledge, unknown. Nevertheless, we will now present a transcription of middle deflation to extended QR algorithms based on rotators.

5.2. Middle Deflation for Extended Hessenberg Matrices. Assume we have given an extended Hessenberg matrix in its QR decomposition, e.g.,



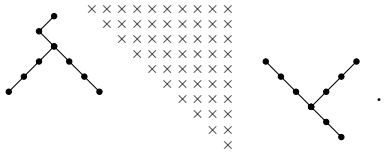
We have to do the same five steps as for the aggressive early deflation, see Section 3.1.

In the first step we have to choose the submatrix S. For details on the selection of the deflation window see the next section describing our preliminary numerical experiments. In our example the middle three rotators can not be applied to R since they are blocked by the last three rotators. After passing the trailing three rotators through the upper triangular we are in the following situation:



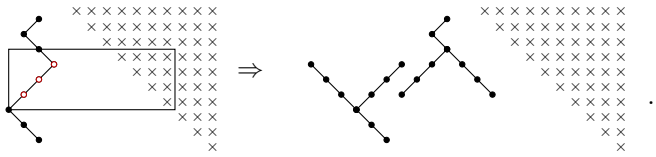
where the rectangle marks the deflation window and thus S. We now use the extended QR algorithm recursively to compute the Schur decomposition of S, $S = V^H T V$. Applying V and V^H to the upper triangular matrix R and to the whole matrix yields

where the matrices V and V^H are embedded in identity matrices. Every unitary matrix of size $m \times m$ can be decomposed in $\frac{1}{2}m(m-1)$ rotators in a pyramid shape. For the unitary matrices on the left-hand side we use an A-pyramid shape and for the matrices on the right-hand side a V-pyramid shape. Again most of the rotators annihilate each other. Hence we and up with two wedges, an A-wedge on the left-hand side and a V-wedge on the right-hand side,

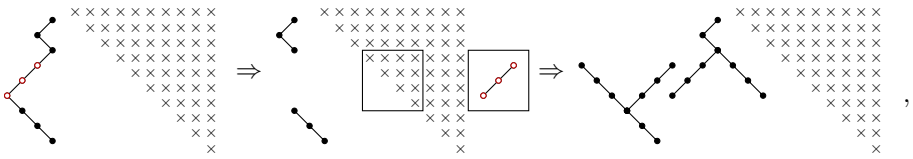


That is still a situation where all the rows and columns respectively are coupled with their neighbors. We use again the wedge-to-broadhead transformation from (3.1) to bring them in a shape, where they are only coupled with the nearest row/column outside the deflation window. On the left-hand side we get again a V-broadhead and on the right-hand side an A-broadhead. One can now remove the diagonal rotators starting with the middle rotation of the broadhead shape. As in middle deflations for Hessenberg matrices we have to remove m rotators, otherwise we cannot deflate the problem in two smaller ones. It might also be necessary to reorder eigenvalues in the Schur decomposition.

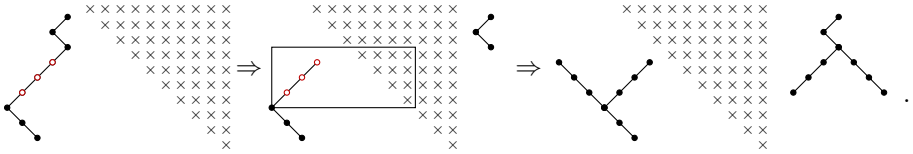
If the rotators are not ordered as in our example, then we have to do other steps to bring them into a suitable shape. If we are in the situation that the first and the last rotators are on the left-hand side of the middle rotators (red dots ○), then we do not have to do any transformation of the shape. We only get the two broadheads on the same side, e.g.,



If the first and the last rotators are on the right-hand side of the middle ones, then we have to bring the middle rotators by similarity transformation to the other side of R, e.g,



and get again the two broadheads on the same side. The last possible situation is that the first rotators are on the right-hand side of the middle rotators and the last are on the left-hand side. In this situation one passes the first rotators through the upper triangular matrix and one gets:



In the next section we present preliminary numerical results for the special case of random matrices.

5.3. Preliminary Numerical Experiments. Let us make the simplifying assumption that the (extended) QR algorithm for an eigenvalue problem of size n costs cn^3 flops. A successful middle deflation splits the matrix $A \in \mathbb{C}^{n\times n}$ in $A_1 \in \mathbb{C}^{n_1\times n_1}$ and $A_2 \in \mathbb{C}^{n_2\times n_2}$, with $n_1 + n_2 \leq n$ and $n_1 \approx n_2$. To achieve this splitting we have to solve an eigenvalue problem of size m. If $n_1 = n_2 = \frac{1}{2}n$, then we save flops for

$m \leq \sqrt[3]{-0.75} \cdot n \approx 0.9085 \cdot n$.

Since the middle deflation procedure involve further computations, we head for a deflation window of size $m \approx 0.75 \cdot n$. We choose the deflation window $f : g$ based on a variation of (3.4), f and g fulfill

$$p_{f,n/2} = \prod_{i=f}^{\frac{n}{2}} |s_i| \leq \epsilon \qquad \text{and} \qquad p_{n/2,g} = \prod_{i=\frac{n}{2}}^{g} |s_i| \leq \epsilon.$$

Since the solution of the eigenvalue problem in the middle deflation is a very expensive step it has to be avoided that the middle deflations fail. Thus we add a safety margin and enlarge the deflation window by 15 indices on both sides.

For the following example we turn off aggressive early deflation.

EXAMPLE 5.1. Let $A \in \mathbb{C}^{1500\times 1500}$ a be a random matrix. After three multishift steps with 150 shifts each the first and the last row of A is deflated. Further the middle deflation prediction suggest that there is a deflation if we use the deflation window 378:1011, which is enlarged to 363:1026. Thus we have to solve a eigenvalue problem of size 723. The middle deflation is successful resulting in two smaller eigenvalues problems of size 664 and 621. Based on the above assumption that the extended QR algorithm costs cn^3 flops, the solution of the three eigenvalue problems is more than 70% cheaper than the solution of the problem of size 1498. Based on the more realistic estimation of $2n^3 + 3n^2 + n$ flops we would save 58% ignoring the additional computations in the middle deflation procedure. We observed similar behavior for other random matrices of different size as long as the number of shifts is about $0.1 \cdot n$.

After this successful test we run the same code for two matrices from the matrix market [4], `tols2000`$\in \mathbb{R}^{2000\times 2000}$ and `pde2961`$\in \mathbb{R}^{2961\times 2961}$.

EXAMPLE 5.2. After the first iteration with `tols2000` the deflation prediction returns the window 996:1005, which is enlarged to 981:1020. Thus we have to find 40 deflatable rotations in the both broadheads together, but there are only three.

The observations with `pde2961` are similar. The deflation prediction returns the deflation window 1429:1533, with safety margin 1414:1548, after the first iteration, but we do not find any deflatable rotator.

Hence the deflation prediction fails completely for both matrices.

The choice of deflation windows is the crucial step for middle deflations. The heuristic prediction investigated here is only partially successful and requires further investigations.

6. Conclusions. The standard deflation criterion in extended QR algorithms preserves eigenvalues to high relative accuracy. We have further seen that the ideas of aggressive early deflation can be transcribed to extended QR algorithms. As in the classical Hessenberg case this leads to earlier deflations and to a significant reduction of the number of iterations. Our numerical results show that aggressive early deflation can reduce also the runtime of the extended QR algorithm. Finally we have seen that in special case also middle deflations can be used in extended QR algorithms.

Acknowledgemends. We thank David S. Watkins (Washington State University) for his useful comments.

REFERENCES

[1] M. Ahues and F. Tisseur, *A new deflation criterion for the QR algorithm*, LAPACK Working Note 122, 1997.

[2] J. Aurentz, R. Vandebril, and D. S. Watkins, *Fast computation of the zeros of a polynomial via factorization of the Companion matrix*, SIAM Journal on Scientific Computing, 35 (2013), pp. A255–A269.

[3] F. L. Bauer and C. T. Fike, *Norms and exclusion theorems*, Numerische Mathematik, 2 (1960), pp. 137–141.

[4] R. F. Boisvert, R. Pozo, K. Remington, R. F. Barrett, and J. J. Dongarra, *Matrix market: A web resource for test matrix collections*, in The Quality of Numerical Software: Assessment and Enhancement, Chapman & Hall, 1997, pp. 125–137.

[5] K. Braman, *Middle deflations in the QR algorithm*, in Talk at Householder Symposium XVII, 2008.

[6] K. Braman, R. Byers, and R. Mathias, *The multishift QR algorithm. part II: Aggressive early deflation*, SIAM Journal on Matrix Analysis and Applications, 23 (2002), pp. 948–973.

[7] R. Byers, *LAPACK 3.1 xHSEQR: Tuning and implementation notes on the small bulge multi-shift QR algorithm with aggressive early deflation*, LAPACK Working Note 187, 2007.

[8] S. C. Eisenstat and I. C. F. Ipsen, *Relative perturbations techiques for singular value problems*, SIAM Journal on Numerical Analysis, 32 (1995).

[9] J. G. F. Francis, *The QR transformation. A unitary analogue to the LR transformation – part 1*, The Computer Journal, 4 (1961), pp. 265–271.

[10] ———, *The QR transformation – part 2*, The Computer Journal, 4 (1962), pp. 332–345.

[11] G. H. Golub and C. F. Van Loan, *Matrix computations*, Johns Hopkins University Press, Baltimore, MD, USA, 4th ed., 2013.

[12] B. Kågström and D. Kressner, *Multishift variants of the QZ algorithm with aggressive early deflation*, SIAM Journal on Matrix Analysis and Applications, 29 (2006), pp. 199–227.

[13] D. Kressner, *Numerical methods for general and structured eigenvalue problems*, vol. 46 of LNCSE, Springer, 2005.

[14] ———, *On the use of larger bulges in the QR algorithm*, Electronic Transactions on Numerical Analysis, 20 (2005), pp. 50–63.

[15] ———, *The effect of aggressive early deflation on the convergence of the QR algorithm*, SIAM Journal on Matrix Analysis and Applications, 30 (2008), pp. 805–821.

[16] D. Kressner, C. Schröder, and D. S. Watkins, *Implicit QR algorithms for palindromic and even eigenvalue problems*, Numerical Algorithms, 51 (2009), pp. 209–238.

[17] T. Mach, M. S. Pranić, and R. Vandebril, *Computing approximate extended Krylov subspaces without explicit inversion*, Report TW 623, KU Leuven, February 2013. Submitted.

[18] Y. Nakatsukasa, K. Aishima, and I. Yamazaki, *dqds with aggressive early deflation*, SIAM Journal on Matrix Analysis and Applications, 33 (2012), pp. 22–51.

[19] B. N. Parlett, *For tridiagonals T replace T with LDL^t*, Journal of Computational and Applied Mathematics, 123 (2000), pp. 117–130. Numerical Analysis 2000. Vol. III: Linear Algebra.

[20] R. Vandebril, *Chasing bulges or rotations? A metamorphosis of the QR-algorithm*, SIAM Journal on Matrix Analysis and Applications, 32 (2011), pp. 217–247.

[21] R. Vandebril and D. S. Watkins, *A generalization of the multishift QR algorithm*, SIAM Journal on Matrix Analysis and Applications, 33 (2012), pp. 759–779.

[22] ———, *An extension of the QZ algorithm beyond the Hessenberg-upper triangular pencil*, Electronic Transactions on Numerical Analysis, 40 (2013), pp. 17–35.

[23] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford University Press, New York, USA, 1965.