MULTISHIFT VARIANTS OF THE QZ ALGORITHM WITH AGGRESSIVE EARLY DEFLATION LAPACK WORKING NOTE 173*

BO KÅGSTRÖM† AND DANIEL KRESSNER†‡

Abstract. New variants of the QZ algorithm for solving the generalized eigenvalue problem are proposed. An extension of the small-bulge multishift QR algorithm is developed, which chases chains of many small bulges instead of only one bulge in each QZ iteration. This allows the effective use of level 3 BLAS operations, which in turn can provide efficient utilization of high performance computing systems with deep memory hierarchies. Moreover, an extension of the aggressive early deflation strategy is proposed, which can identify and deflate converged eigenvalues long before classic deflation strategies would. Consequently, the number of overall QZ iterations needed until convergence is considerably reduced. As a third ingredient, we reconsider the deflation of infinite eigenvalues and present a new deflation algorithm, which is particularly effective in the presence of a large number of infinite eigenvalues. Combining all these developments, our implementation significantly improves existing implementations of the QZ algorithm. This is demonstrated by numerical experiments with random matrix pairs as well as with matrix pairs arising from various applications.

Key words. Generalized eigenvalue problem, generalized Schur form, QZ algorithm, multishifts, aggressive early deflation, blocked algorithms.

AMS subject classifications. 65F15, 15A18, 15A22, 47A75

1. Introduction. The QZ algorithm is a numerically backward stable method for computing generalized eigenvalues and deflating subspaces of small- to medium-sized regular matrix pairs (A, B) with $A, B \in \mathbb{R}^{n\times n}$. It goes back to Moler and Stewart in 1973 [35] and has undergone only a few modifications during the next 25 years, notably through works by Ward [45, 46], Kaufman [27], Dackland and Kågström [11]. Non-orthogonal variants of the QZ algorithm include the LZ algorithm by Kaufman [26] and the AB algorithm for pencils by Kublanovskaya [32].

The purpose of the QZ algorithm is to compute a *generalized Schur decomposition* of (A, B), i.e., orthogonal matrices Q and Z so that $S = Q^T AZ$ is quasi-upper triangular with 1×1 and 2×2 blocks on the diagonal, while the matrix $T = Q^T BZ$ is upper triangular. This decomposition provides almost everything needed to solve the generalized nonsymmetric eigenvalue problem (GNEP). *Generalized eigenvalues*, defined as root pairs (α, β) of the bivariate polynomial $\det(\beta A - \alpha B)$, can be directly computed from the diagonal blocks of S and T, although some care must be taken to implement this computation in a safe manner, see [35, 43]. Moreover, the leading k columns of the orthogonal matrices Z and Q span a pair of *deflating subspaces* [39] if the $(k+1, k)$ subdiagonal entry of the matrix S vanishes. A reordering of the diagonal blocks of S and T can be used to compute other deflating subspaces, see [21, 25, 42].

The eigenvalues of (A, B) are read off from (S, T) as follows. The 2×2 diagonal blocks correspond to pairs of complex conjugate eigenvalues. The real eigenvalues are given in pairs (s_{ii}, t_{ii}) corresponding to the 1×1 diagonal blocks of (S, T). The

*Also appeared as technical report UMINF-05.11, Department of Computing Science, Umeå University, Umeå, Sweden. This work was supported by the DFG Emmy Noether fellowship KR 2950/1-1 and by the *Swedish Research Council* under grant VR 621-2001-3284 and by the *Swedish Foundation for Strategic Research* under grant A3 02:128. This research was conducted using the resources of the High Performance Computing Center North (HPC2N).

†`{bokg, kressner}@cs.umu.se`, Department of Computing Science and HPC2N, Umeå University, S-901 87 Umeå, Sweden

‡`kressner@math.hr`, Department of Mathematics, Bijenička 30, 10000 Zagreb, Croatia.

finite eigenvalues are s_{ii}/t_{ii}, where $t_{ii} \neq 0$. An infinite eigenvalue is represented as $(s_{ii}, 0)$ with $s_{ii} \neq 0$. If $(s_{ii}, t_{ii}) \neq (0, 0)$ for all i, then (A, B) is a *regular matrix pair* or equivalently $\beta A - \alpha B$ is a regular matrix pencil. Otherwise, the matrix pair is *singular* and at least one (s_{ii}, t_{ii}) equals $(0, 0)$. These situations need extra caution, and so called staircase-type algorithms can be used for identifying singular cases by computing a generalized upper triangular (GUPTRI) form of (A, B) (e.g., see Demmel and Kågström [12, 13]).

Three ingredients make the QZ algorithm work effectively. First, the matrix pair (A, B) is reduced to Hessenberg-triangular form, i.e., orthogonal matrices Q and Z are computed so that $H = Q^T AZ$ is upper Hessenberg and $T = Q^T BZ$ is upper triangular. Second, a sequence of so called implicit shifted QZ iterations is applied to (H, T), in order to bring H closer to (block) upper triangular form while preserving the Hessenberg-triangular form of (H, T). Each of these iterations can be seen as chasing a pair of bulges from the top left to the bottom right corner along the subdiagonals of H and T; a point of view that has been emphasized by Watkins and Elsner [51]. The third ingredient is deflation, which aims at splitting the computation of the generalized Schur form (S, T) into smaller subproblems. This paper describes improvements for the latter two ingredients, QZ iterations and deflations.

Inspired by the works of Braman, Byers, Mathias [6], and Lang [34] for the QR algorithm, we propose multishift QZ iterations that chase a tightly coupled chain of bulge pairs instead of only one bulge pair per iteration. This allows the effective use of level 3 BLAS operations [14, 23, 24] during the bulge chasing process, which in turn can provide efficient utilization of today's high performance computing systems with deep memory hierarchies. Tightly coupled bulge chasing has also successfully been used in the reduction of a matrix pair (H_r, T) in block Hessenberg-triangular form, where H_r has r subdiagonals, to Hessenberg-triangular form (H, T) [11].

Recently, Braman, Byers, and Mathias [7] also presented a new, advanced deflation strategy, the so called aggressive early deflation. Combining this deflation strategy with multishift QR iterations leads to a variant of the QR algorithm, which may, for sufficiently large matrices, require less than 10% of the computing time needed by the LAPACK [2] implementation. We will show that this deflation strategy can be extended to the QZ algorithm, resulting in similar time savings.

A (nearly) singular matrix B often implies that the triangular matrix T of the corresponding Hessenberg-triangular form has one or more diagonal entries close to zero. Each of these diagonal entries admits the deflation of an infinite eigenvalue. Some applications, such as semi-discretized Stokes equations [40], lead to matrix pairs that have a large number of infinite eigenvalues. Consequently, a substantial amount of computational work in the QZ algorithm is spent for deflating these eigenvalues. We will provide a discussion on this matter including preprocessing techniques and we propose windowing techniques that lead to more efficient algorithms for deflating infinite eigenvalues within the QZ algorithm. This approach is conceptually close to blocked algorithms for reordering eigenvalues in standard and generalized Schur forms [30].

The rest of this paper is organized as follows. In Section 2, we review and extend conventional multishift QZ iterations, and provide some new insight into their numerical backward stability. Multishift variants that are based on chasing a tightly coupled chain of bulge pairs are described in Section 3. In Section 4, a thorough discussion on dealing with infinite eigenvalues is presented that include preprocessing and efficient methods for deflating such eigenvalues within the QZ algorithm. Aggressive early deflation for the QZ algorithm and its connection to the distance of uncontrollability for descriptor systems are studied in Section 5. Computational experiments, presented in Section 6, demonstrate the effectiveness of our newly developed multishift QZ algorithm with advanced deflation techniques. Finally, some concluding remarks are summarized in Section 7.

2. Conventional multishift QZ iterations. Throughout the rest of this paper we assume that the matrix pair under consideration, which will be denoted by (H, T), is already in Hessenberg-triangular form. Efficient algorithms for reducing a given matrix pair to this form can be found in [11, 29]. For the moment, we also assume that (H, T) is an *unreduced matrix pair*, i.e., all subdiagonal entries of H as well as all diagonal entries of T are different from zero. The latter condition implies that only finite eigenvalues are considered.

A QZ iteration relies on a fortunate choice of m shifts (or shift pairs) (μ_1, ν_1), $(\mu_2, \nu_2), \ldots, (\mu_m, \nu_m)$ with $\mu_i \in \mathbb{C}$ and $\nu_i \in \mathbb{R}$, giving rise to the *shift polynomial*

$$p(HT^{-1}) = (\nu_1 HT^{-1} - \mu_1 I_n)(\nu_2 HT^{-1} - \mu_2 I_n) \cdots (\nu_m HT^{-1} - \mu_m I_n). \tag{2.1}$$

If x denotes the first column of this matrix polynomial, then the first step of an implicit shifted QZ iteration consists of choosing an orthogonal matrix Q_1 such that $Q_1^T x$ is a multiple of the first unit vector e_1. The rest of the QZ iteration consists of reducing the updated matrix pair $(Q_1^T H, Q_1^T T)$ back to Hessenberg-triangular form, without modifying the first rows of $Q_1^T H$ and $Q_1^T T$ by transformations from the left.

In the original formulation of the QZ algorithm [35], this reduction to Hessenberg-triangular form was described for $m \leq 2$, based on combinations of Givens rotations and Householder matrices. This approach has the negative side-effect that one QZ iteration with $m = 2$ shifts requires more *flops* (floating point operations) than two QZ iterations with $m = 1$ shift. Partly avoiding this increase of flops, Ward [45] proposed the so called *combination shift QZ algorithm* which uses $m = 1$ for real shifts and $m = 2$ for complex conjugate pairs of shifts. Later on, Watkins and Elsner [51] proposed a variant that is solely based on Householder matrices, which requires roughly 27% less flops than the original formulation and may employ an arbitrary number m of shifts. This variant is currently implemented in the LAPACK subroutine `DHGEQZ`. A curiosity of this subroutine is that it still uses Ward's combination shift strategy despite the fact that two single shift QZ iterations now require roughly 9% more flops than one double shift iteration.

2.1. Householder-based variants. In the following, we describe the Householder-based variant by Watkins and Elsner in more detail. To simplify the notation, we make use of the following convention.

DEFINITION 2.1. *A Householder matrix which maps the last $n - j$ elements of a given vector $x \in \mathbb{R}^n$ to zero without modifying the leading $j - 1$ elements is denoted by $\mathcal{H}_j(x)$.*

Let us illustrate the first few steps of an implicit QZ iteration for $n = 6, m = 2$. First, a Householder matrix $\mathcal{H}_1(x)$ is used to map x, the first column of the shift polynomial defined in (2.1), to a multiple of e_1. Note that only the leading three elements of x are nonzero. Hence, if $\mathcal{H}_1(x)$ is applied from the left to H and T, only the first three rows (denoted by the symbols $\hat{h}$ and $\hat{t}$ below) are affected while the remaining rows stay unchanged (denoted by h and t):

$$(H,T) \leftarrow \left(\begin{bmatrix} \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ 0 & 0 & h & h & h & h \\ 0 & 0 & 0 & h & h & h \\ 0 & 0 & 0 & 0 & h & h \end{bmatrix}, \begin{bmatrix} \hat{t} & \hat{t} & \hat{t} & \hat{t} & \hat{t} & \hat{t} \\ \hat{t} & \hat{t} & \hat{t} & \hat{t} & \hat{t} & \hat{t} \\ \hat{t} & \hat{t} & \hat{t} & \hat{t} & \hat{t} & \hat{t} \\ 0 & 0 & 0 & t & t & t \\ 0 & 0 & 0 & 0 & t & t \\ 0 & 0 & 0 & 0 & 0 & t \end{bmatrix} \right). \tag{2.2}$$

Next, to avoid further fill-in in the factor T, the newly introduced entries $(2,1)$ and $(3,1)$ must be eliminated. Recall that we are not allowed to change the first row of T by applying a transformation from the left. However, it is still possible to achieve these eliminations by applying a Householder matrix using the following simple fact.

LEMMA 2.2 ([51]). *Let $T \in \mathbb{R}^{n\times n}$ be an invertible matrix then the first column of $T\,\mathcal{H}_1(T^{-1}e_1)$ is a scalar multiple of e_1.*

Applying a Householder matrix from the *right* to eliminate several elements in one *column* (instead of one row) is somewhat opposite to their standard use. This motivates us to call such a matrix an *opposite Householder matrix.* Applying $\mathcal{H}_1(T^{-1}e_1)$ from the right yields the following diagram:

$$(H,T) \leftarrow \left(\begin{bmatrix} \hat{h} & \hat{h} & \hat{h} & h & h & h \\ \hat{h}_b & \hat{h}_b & \hat{h}_b & h & h & h \\ \hat{h}_b & \hat{h}_b & \hat{h}_b & h & h & h \\ \hat{h}_b & \hat{h}_b & \hat{h}_b & h & h & h \\ 0 & 0 & 0 & h & h & h \\ 0 & 0 & 0 & 0 & h & h \end{bmatrix}, \begin{bmatrix} \hat{t} & \hat{t} & \hat{t} & t & t & t \\ \hat{0}_b & \hat{t}_b & \hat{t}_b & t & t & t \\ \hat{0}_b & \hat{t}_b & \hat{t}_b & t & t & t \\ 0_b & 0_b & 0_b & t & t & t \\ 0 & 0 & 0 & 0 & t & t \\ 0 & 0 & 0 & 0 & 0 & t \end{bmatrix} \right). \tag{2.3}$$

Here, we have used the subscript b to designate entries that belong to the so called *bulge pair.* The rest of the QZ iteration can be seen as pushing this bulge pair along the subdiagonals down to the bottom right corners until it vanishes. The next two steps consist of applying the Householder matrix $\mathcal{H}_2(He_1)$ from the left and the opposite Householder matrix $\mathcal{H}_2(T^{-1}e_2)$ from the right:

$$(H,T) \leftarrow \left(\begin{bmatrix} h & h & h & h & h & h \\ \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ \hat{0} & \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ \hat{0} & \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ 0 & 0 & 0 & h & h & h \\ 0 & 0 & 0 & 0 & h & h \end{bmatrix}, \begin{bmatrix} t & t & t & t & t & t \\ 0 & \hat{t} & \hat{t} & \hat{t} & \hat{t} & \hat{t} \\ 0 & \hat{t} & \hat{t} & \hat{t} & \hat{t} & \hat{t} \\ 0 & \hat{t} & \hat{t} & \hat{t} & \hat{t} & \hat{t} \\ 0 & 0 & 0 & 0 & t & t \\ 0 & 0 & 0 & 0 & 0 & t \end{bmatrix} \right),$$

$$(H,T) \leftarrow \left(\begin{bmatrix} h & \hat{h} & \hat{h} & \hat{h} & h & h \\ h & \hat{h} & \hat{h} & \hat{h} & h & h \\ 0 & \hat{h}_b & \hat{h}_b & \hat{h}_b & h & h \\ 0 & \hat{h}_b & \hat{h}_b & \hat{h}_b & h & h \\ 0 & \hat{h}_b & \hat{h}_b & \hat{h}_b & h & h \\ 0 & 0 & 0 & 0 & h & h \end{bmatrix}, \begin{bmatrix} t & \hat{t} & \hat{t} & \hat{t} & t & t \\ 0 & \hat{t} & \hat{t} & \hat{t} & t & t \\ 0 & \hat{0}_b & \hat{t}_b & \hat{t}_b & t & t \\ 0 & \hat{0}_b & \hat{t}_b & \hat{t}_b & t & t \\ 0 & 0_b & 0_b & 0_b & t & t \\ 0 & 0 & 0 & 0 & 0 & t \end{bmatrix} \right). \tag{2.4}$$

For general m and n, the implicit shifted QZ iteration based on (opposite) Householder matrices is described in Algorithm 1. Here, the *colon notation* $A(i_1 : i_2, j_1 : j_2)$ is used to designate the submatrix of a matrix A defined by rows i_1 through i_2 and columns j_1 through j_2.

Note that the shifts employed in Algorithm 1 are based on the generalized eigenvalues of the bottom right $m \times m$ submatrix pair, a choice which is sometimes called *generalized Francis shifts* and which ensures quadratic local convergence [51]. If $m \ll n$, a proper implementation of this algorithm requires $2(4m+3)n^2 + \mathcal{O}(n)$ flops for updating H and T. In addition, $(4m+3)n^2 + \mathcal{O}(n)$ flops are required for updating each of the orthogonal factors Q and Z.

Algorithm 1 Implicit shifted QZ iteration based on Householder matrices

Input: An $n \times n$ matrix pair (H, T) in unreduced Hessenberg-triangular form, an integer $m \in [2, n]$.

Output: Orthogonal matrices $Q, Z \in \mathbb{R}^{n \times n}$ so that $Q^T(H, T)Z$ is the Hessenberg-triangular matrix pair obtained after applying a QZ iteration with m shifts. The matrix pair (H, T) is overwritten by $Q^T(H, T)Z$.

Compute (μ_1, ν_1), $(\mu_2, \nu_2), \dots, (\mu_m, \nu_m)$ as generalized eigenvalues of the matrix pair

$$(H(n-m+1:n, n-m+1:n), T(n-m+1:n, n-m+1:n)).$$

Set $x = ((\nu_1 HT^{-1} - \mu_1 I_n)(\nu_2 HT^{-1} - \mu_2 I_n) \cdots (\nu_m HT^{-1} - \mu_m I_n))e_1$.
$(H, T) \leftarrow \mathcal{H}_1(x) \cdot (H, T)$
$Q \leftarrow \mathcal{H}_1(x), \quad Z \leftarrow \mathcal{H}_1(T^{-1}e_1)$
$(H, T) \leftarrow (H, T) \cdot Z$
for $j \leftarrow 1, 2, \dots, n-2$ **do**
  $\tilde{Q} \leftarrow \mathcal{H}_{j+1}(He_j)$
  $(H, T) \leftarrow \tilde{Q} \cdot (H, T)$
  $Q \leftarrow Q\tilde{Q}$
  $\tilde{Z} \leftarrow \mathcal{H}_{j+1}(T^{-1}e_{j+1})$
  $(H, T) \leftarrow (H, T) \cdot \tilde{Z}$
  $Z \leftarrow Z\tilde{Z}$
end for

2.2. Error analysis of opposite Householder matrices. Some authors have raised concerns that the use of opposite Householder matrices could introduce numerical instabilities in the QZ algorithm, see, e.g., [11, p. 444]. Such instabilities could arise if some entries that should be zero after the application of an opposite Householder matrix are non-negligible in finite-precision arithmetic. In the following, we provide a brief error analysis showing that such an event may not occur if some care is taken.

Without loss of generality, we can restrict the analysis to an opposite Householder matrix of the form $\mathcal{H}_1(T^{-1}e_1)$ for some nonsingular matrix $T \in \mathbb{R}^{n \times n}$. Although an ill-conditioned T may severely affect the data representing $\mathcal{H}_1(T^{-1}e_1)$, it has almost no effect on the purpose of $\mathcal{H}_1(T^{-1}e_1)$, which is the introduction of zero entries. To explain this, assume that a numerically backward stable method is employed to solve the linear system $Tx = e_1$, yielding a computed solution $\hat{x}$. This implies that $\hat{x}$ is the exact solution of a slightly perturbed system

$$(T + F)\hat{x} = e_1, \quad \|F\|_2 \le c_T \|T\|_2, \tag{2.5}$$

where c_T is not much larger than the unit roundoff $\mathbf{u}$ [20]. Now, consider the Householder matrix $\mathcal{H}_1(\hat{x}) = I - \tilde{\beta}\tilde{v}\tilde{v}^T$, where $\tilde{\beta} \in \mathbb{R}, \tilde{v} \in \mathbb{R}^n$, such that $(I - \tilde{\beta}\tilde{v}\tilde{v}^T)\hat{x} = \tilde{\gamma}e_1$ for some scalar $\tilde{\gamma}$. The computation of the quantities $\tilde{\beta}, \tilde{v}$ defining $\mathcal{H}_1(\hat{x})$ is also subject to roundoff errors. Using standard computational methods, the computed quantities $\hat{v}, \hat{\beta}$ satisfy

$$|\hat{\beta} - \tilde{\beta}| \le c_\beta |\tilde{\beta}| \approx (4n + 8)\mathbf{u}|\tilde{\beta}|, \quad \|\hat{v} - \tilde{v}\|_2 \le c_v \|\tilde{v}\|_2 \approx (n + 2)\mathbf{u}\|\tilde{v}\|_2,$$

see [20, p. 365]. It follows that

$$\begin{aligned}\|T \cdot (I - \hat{\beta}\hat{v}\hat{v}^T)e_1 - 1/\tilde{\gamma} \cdot e_1\|_2 \le{}& \|T \cdot (I - \tilde{\beta}\tilde{v}\tilde{v}^T)e_1 - 1/\tilde{\gamma} \cdot e_1\|_2 \\ &+ (2c_\beta + 4c_v)\|T\|_2 + \mathcal{O}(\mathbf{u}^2)\end{aligned}$$

$$\leq (c_T + 2c_\beta + 4c_v)\|T\|_2 + \mathcal{O}(\mathbf{u}^2).$$

This shows that if $\hat{x}$ is computed by a backward stable method, then the last $n-1$ elements in the first column of $T(I - \beta\hat{v}\hat{v}^T)$ can be set to zero without spoiling the backward stability of the QZ algorithm.

In this paper, we favour the following method for constructing opposite Householder matrices. Let $T = RQ$ be an RQ decomposition, i.e., the matrix $R \in \mathbb{R}^{n\times n}$ is upper triangular and $Q \in \mathbb{R}^{n\times n}$ is orthogonal. If T is invertible, then $Q^T e_1$ is a scalar multiple of $T^{-1}e_1$ implying that $\mathcal{H}_1(Q^T e_1)$ is an opposite Householder matrix. Even if T is singular, it can be shown that the first column of $T \cdot \mathcal{H}_1(Q^T e_1)$ is mapped to a multiple of e_1:

$$T \cdot \mathcal{H}_1(Q^T e_1) = R \cdot [Q \cdot \mathcal{H}_1(Q^T e_1)] = \begin{bmatrix} r_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \begin{bmatrix} \tilde{q}_{11} & 0 \\ 0 & \tilde{Q}_{22} \end{bmatrix}$$
$$= \begin{bmatrix} r_{11}\tilde{q}_{11} & R_{12}\tilde{Q}_{22} \\ 0 & R_{22}\tilde{Q}_{22} \end{bmatrix}.$$

RQ decompositions enjoy a favourable backward error analysis, the constant c_T in (2.5) can be bounded by roughly $n^2\mathbf{u}$, see, e.g., [20, Thm. 18.4].

2.3. Bulge pairs and shift blurring. Convergence in the implicit shifted QZ iteration typically becomes manifest in the bottom right corner of the matrix pair, often the m^{th}-last subdiagonal entry of H converges to zero. As a QZ iteration can be interpreted as chasing a bulge pair from the top left corner to the bottom right corner of (H,T), the question arises how the information contained in the shifts is passed during this chasing process. Watkins [50] discovered a surprisingly simple relationship; the indented shifts are the finite eigenvalues of the bulge pairs.

To explain this in more detail, suppose that the implicit shifted QZ iteration with m shifts, Algorithm 1, is applied to $(H,T) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n}$ with $n > m$. As before, we assume that (H,T) is in unreduced Hessenberg-triangular form but we do not assume that T is nonsingular, only the part used for the shift computation (the trailing $m\times m$ principal submatrix of T) and the part involved in the introduction of the bulge pair (the leading $m \times m$ principal submatrix of T) are required to be nonsingular. Let x be a multiple of the first column of the shift polynomial defined in (2.1). The *initial bulge pair* is the matrix pair $\big(B_0^{(H)}, B_0^{(T)}\big)$, where

$$B_0^{(H)} = [x(1:m+1), H(1:m+1:1:m)] = \begin{bmatrix} x_1 & h_{11} & \cdots & h_{1m} \\ x_2 & h_{21} & \ddots & \vdots \\ \vdots & & \ddots & h_{mm} \\ x_{m+1} & 0 & & h_{m+1,m} \end{bmatrix},$$

$$B_0^{(T)} = [0, T(1:m+1:1:m)] = \begin{bmatrix} 0 & t_{11} & \cdots & t_{1m} \\ 0 & 0 & \ddots & \vdots \\ \vdots & & \ddots & t_{mm} \\ 0 & 0 & \cdots & 0 \end{bmatrix}.$$

THEOREM 2.3 ([50]). *If the leading $m\times m$ principal submatrix of T is nonsingular, then the shifts $(\sigma_1, 1), \ldots, (\sigma_m, 1)$ are the finite eigenvalues of the initial bulge pair $\big(B_0^{(H)}, B_0^{(T)}\big)$.*

During the course of a QZ iteration, a bulge pair is first created at the top left corners and then chased down to the bottom right corners. Let $\big(H^{(j)}, T^{(j)}\big)$ denote the updated matrix pair (H, T) obtained after the bulge pair has been chased $j-1$ steps, which amounts to applying $j-1$ loops of Algorithm 1. Then, the j^{th} *bulge pair* $\big(B_j^{(H)}, B_j^{(T)}\big)$ is given by

$$\begin{array}{rcl} B_j^{(H)} & = & H^{(j)}(j+1:j+m+1, j:j+m+1), \\ B_j^{(T)} & = & T^{(j)}(j+1:j+m+1, j:j+m+1), \end{array} \tag{2.6}$$

which corresponds to the submatrices designated by the subscript b in (2.3)– (2.4).

THEOREM 2.4 ([50]). *If the m^{th} leading principal submatrix of T is nonsingular, then the shifts $\sigma_1, \ldots, \sigma_m$ are the finite eigenvalues of the j^{th} bulge pair $\big(B_j^{(H)}, B_j^{(T)}\big)$.*

Note that the definition of a bulge pair is only possible for $j \leq n-m-1$, since otherwise (2.6) refers to entries outside of $H^{(j)}$ and $T^{(j)}$. This issue can be resolved; by adding virtual rows and columns to the matrix pair $(H^{(j)}, T^{(j)})$, see [50], Theorem 2.4 can be extended to the case $j > n-m-1$.

Early attempts to improve the performance of the QR algorithm focused on using shift polynomials of high degree [4], leading to medium-order Householder matrices during the QR iteration and enabling the efficient use of WY representations. This approach, however, has proved disappointing due to the fact that the convergence of such a large-bulge multishift QR algorithm is severely affected by roundoff errors [15]. This effect is caused by *shift blurring*: with increasing m the eigenvalues of the bulge pairs, which should represent the shifts in exact arithmetic, often become extremely sensitive to perturbations [48, 49, 31]. Already for moderate m, say $m \geq 15$, the shifts may be completely contaminated by round-off errors during the bulge chasing process. Not surprisingly, we made similar observations in numerical experiments with implicit shifted QZ iterations, which also suffer from shift blurring.

3. Multishift QZ iterations based on tightly coupled tiny bulge pairs. The trouble with shift blurring can be avoided by developing variants of the implicit shifted QZ algorithm that still rely on a large number of simultaneous shifts but chase several tiny bulge pairs instead of one large bulge pair. Such ideas have already been successfully applied to the QR algorithm, see, e.g., [6, 34] and the references therein. In this section, we describe an extension of the work by Braman, Byers, and Mathias [6] to the QZ algorithm.

For the purpose of describing this new tiny-bulge multishift QZ algorithm, let m denote the number of simultaneous shifts to be used in each QZ iteration and let n_s denote the number of shifts contained in each bulge pair. It is assumed that m is an integer multiple of n_s. To avoid shift blurring phenomena we use tiny values for n_s, say $n_s = 2$ or $n_s = 4$.

Our algorithm performs an implicit shifted QZ iteration with m generalized Francis shifts to a Hessenberg-triangular matrix pair (H, T) and consists of three stages, which are described in more detail below. First, a tightly coupled chain of m/n_s bulge pairs is bulge-by-bulge introduced in the top left corners of H and T. Second, the whole chain at once is chased down along the subdiagonal until the bottom bulge pair reaches the bottom right corners of H and T. Finally, all bulge pairs are bulge-by-bulge chased off this corner.

3.1. Introducing a chain of bulge pairs. The tiny-bulge multishift QZ algorithm begins with introducing m/n_s bulge pairs in the top left corner of the matrix

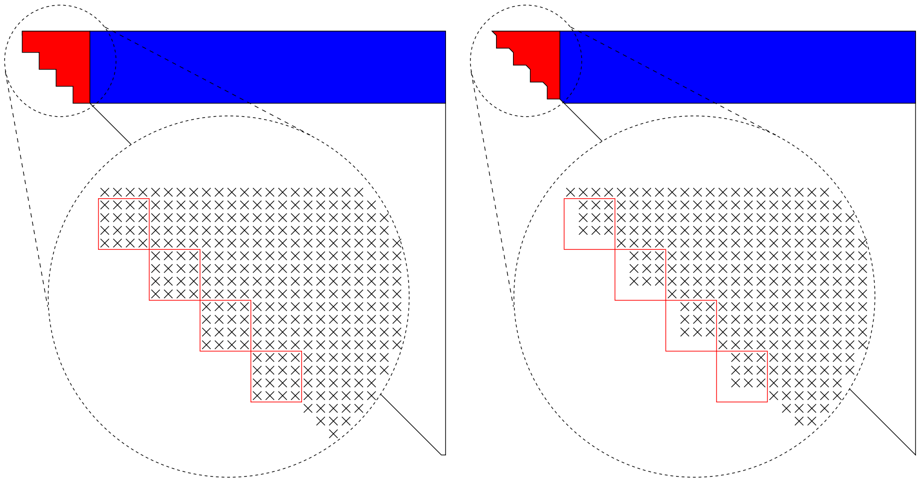


FIG. 3.1. *Introducing a chain of $m/n_s = 4$ tightly coupled bulge pairs, each of which contains $n_s = 3$ shifts.*

pair (H, T). Every bulge pair contains a set of n_s shifts. It is assumed that the $((m/n_s)(n_s + 1) - 1)^{\text{th}}$ leading principal submatrix of T is nonsingular. The first bulge pair is introduced by applying an implicit QZ iteration with n_s shifts and interrupting the bulge chasing process as soon as the bottom right corner of the bulge in H touches the $(p_h - 1, p_h)$ subdiagonal entry of H, where $p_h = (m/n_s)(n_s + 1) + 1$. The next bulge pair is chased until the bottom right corner of the bulge in H touches the $(p_h - n_s - 2, p_h - n_s - 1)$ subdiagonal entry. This process is continued until all m/n_s bulge pairs are introduced, see Figure 3.1. Note that only the submatrices marked red in Figure 3.1 must be updated during the bulge chasing process. To update the remaining parts (marked blue), all orthogonal transformations from the left are accumulated into a $p_h \times p_h$ matrix U and applied in terms of general matrix-matrix multiply (GEMM) operations:

$$\begin{aligned} H(1 : p_h, (p_h + 1) : n) &\leftarrow U^T \cdot H(1 : p_h, (p_h + 1) : n), \\ T(1 : p_h, (p_h + 1) : n) &\leftarrow U^T \cdot T(1 : p_h, (p_h + 1) : n). \end{aligned}$$

3.2. Chasing a chain of bulge pairs. In each step of the tiny-bulge multishift QZ algorithm, the chain of bulge pairs is chased k steps downwards. Before the first step, this chain resides in columns/rows $p_l : p_h$ with $p_l = 1, p_h = (m/n_s)(n_s + 1) + 1$ as above. Before the next step, we have $p_l = 1 + k, p_h = (m/n_s)(n_s + 1) + 1 + k$, and so on.

The whole chain is chased in a bulge-by-bulge and bottom-to-top fashion. One such step is illustrated in Figure 3.2. Again, only the principal submatrices marked red in Figure 3.2 must be updated during the bulge chasing process. All transformations from the left and from the right are accumulated into orthogonal matrices U and V, respectively. Then, GEMM operations can be used to update the rest of the matrix pair (marked blue in Figure 3.2):

$$\begin{aligned} H(p_l : p_h + k, (p_h + 1) : n) &\leftarrow U^T \cdot H(p_l : p_h + k, (p_h + 1) : n), \\ T(p_l : p_h + k, (p_h + 1) : n) &\leftarrow U^T \cdot T(p_l : p_h + k, (p_h + 1) : n), \\ H(1 : p_l - 1, p_l : p_h + k) &\leftarrow H(1 : p_l - 1, p_l : p_h + k) \cdot V, \end{aligned}$$

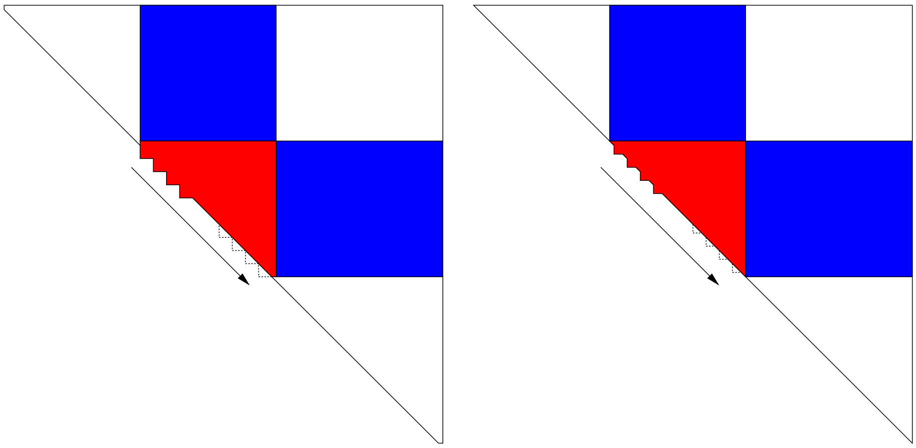


FIG. 3.2. *Chasing a chain of $m/n_s = 4$ tightly coupled bulge pairs.*

$$T(1 : p_l - 1, p_l : p_h + k) \leftarrow T(1 : p_l - 1, p_l : p_h + k) \cdot V.$$

Note that both matrices, U and V, have the following block structure:

$$\begin{array}{c} \\ 1 \\ l_1 \\ l_2 \end{array}\begin{array}{c} \begin{array}{ccc} 1 & l_2 & l_1 \end{array} \\ \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & \square & \triangle \\ 0 & \triangle & \square \end{array}\right] \end{array}, \tag{3.1}$$

where $l_1 = (m/n_s)(n_s+1)-n_s$ and $l_2 = k+n_s$. Exploiting this triangular block structure may have a positive effect on the efficiency of matrix-matrix multiply operations involving U or V.

As for the tiny-bulge multishift QR algorithm, we have to be aware of so called *vigilant deflations* [6, 47], i.e., zero or tiny subdiagonal elements in H that arise during the chasing process. In order to preserve the information contained in the bulge pairs, the chain of bulge pairs must be reintroduced in the row in which the zero appears. Fortunately, we have not to be aware of zero or tiny subdiagonal elements in T, since the bulge pairs are properly passed through infinite eigenvalues, see Section 4.

After a certain number of steps, the bottom bulge pair of the chain reaches the bottom right corners of the matrix pair. As soon as this happens, the whole chain is bulge-by-bulge chased off this corner, similarly to the introduction of bulge pairs.

3.3. Classic deflation of finite eigenvalues. The goal of (multishift) QZ iterations is to drive the subdiagonal entries of the Hessenberg matrix in (H, T) to zero while preserving the upper triangular shape of T. Once a subdiagonal entry $h_{k+1,k}$ is considered zero, the problem is deflated into two smaller problems:

$$\left(\begin{bmatrix} H_{11} & H_{12} \\ 0 & H_{22} \end{bmatrix}, \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \right).$$

Afterwards, the (multishift) QZ iteration is applied separately to the $k \times k$ and $(n - k) \times (n - k)$ matrix pairs (H_{11}, T_{11}) and (H_{22}, T_{22}), respectively.

In the original formulation of the QZ algorithm [35] and the current implementation in LAPACK, a subdiagonal entry $h_{k+1,k}$ is considered zero if

$$|h_{k+1,k}| \leq \mathbf{u}\,\|H\|_F. \tag{3.2}$$

A more conservative criterion, in the spirit of the LAPACK implementation of the QR algorithm, is to consider $h_{k+1,k}$ zero if

$$|h_{k+1,k}| \leq \mathbf{u}\,(|h_{k,k}| + |h_{k+1,k+1}|). \tag{3.3}$$

It is known for standard eigenvalue problems that, especially in the presence of graded matrices, the use of the criterion (3.3) gives higher accuracy in the computed eigenvalues [38]. We have observed similar accuracy improvements for the QZ algorithm when using (3.3) in favour of (3.2). We have also encountered examples where both criteria give similar accuracy but with slightly shorter execution times for (3.2) due to earlier deflations.

4. Dealing with infinite eigenvalues. If the degree p of the polynomial $\det(\beta A - \alpha B)$ is less than n then the matrix pair (A, B) is said to have $n-p$ infinite eigenvalues. The relationship between infinite eigenvalues and the QZ algorithm is subtle and calls for caution. In finite-precision arithmetic, the QZ algorithm may utterly fail to correctly identify infinite eigenvalues, especially if the *index* of the matrix pair, defined as the size of the largest Jordan block associated with an infinite eigenvalue [17], is larger than one [35]. In the context of differential-algebraic equations (DAE), the index of (A, B) corresponds to the index of the DAE $B\dot{x} = Ax + f$. Many applications, such as multibody systems and electrical circuits, lead to DAEs with index at least two, see, e.g., [8, 37, 41].

If the matrix pair (A, B) has an infinite eigenvalue then the matrix B is singular. This implies that at least one of the diagonal entries in the upper triangular matrix T in the Hessenberg-triangular form (H, T) and in the generalized Schur form (S, T) is zero, and vice versa. In finite-precision arithmetic, zero diagonal entries are spoiled by roundoff errors. While a tiny zero diagonal entry of T implies that T is numerically singular, the converse is generally *not* true. There are well known examples of upper triangular matrices that are numerically singular but have diagonal entries that are not significantly smaller than the norm of the matrix [18, Ex. 5.5.1].

In such cases, much more reliable decisions on the nature of infinite eigenvalues can be met using algorithms that reveal Kronecker structures, such as GUPTRI [12, 13]. In some cases, infinite eigenvalues can be cheaply and reliably deflated by exploiting the structure of A and B.

4.1. Preprocessing deflation of infinite eigenvalues. Given a regular matrix pair (A, B) with infinite eigenvalues corresponding to several Jordan blocks, the QZ algorithm will typically compute eigenvalue pairs (α_i, β_i) with β_i nonzero. Moreover, otherwise well-conditioned eigenvalues can be affected by perturbations from these defective infinite eigenvalues, e.g., they may coincide or appear in clusters of eigenvalues corresponding to computed infinite as well as finite eigenvalues. In the following, we briefly describe two preprocessing techniques for handling such situations.

Exploiting staircase algorithms. Without having any knowledge of the Jordan structure of the infinite eigenvalue, in principle, the only reliable and robust way to identify all infinite eigenvalues is to apply a preprocessing step with a staircase-type of algorithm.

By applying the GUPTRI algorithm [12, 13, 22] to a regular pair (A, B) with infinite eigenvalues, we get

$$U^T(A,B)V = \left(\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{\text{inf}} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{\text{inf}} \end{bmatrix} \right), \tag{4.1}$$

where U and V are orthogonal transformation matrices, $(A_{\text{inf}}, B_{\text{inf}})$ reveals the Jordan structure of the infinite eigenvalue, and (A_{11}, B_{11}) is a matrix pair with only finite eigenvalues.

Let us illustrate the GUPTRI form (4.1) with a small example. We consider a 7×7 pair (A, B) with three finite eigenvalues and an infinite eigenvalue of multiplicity four corresponding to two nilpotent Jordan blocks N_1 and N_3. The infinite eigenvalue is both *derogatory* and *defective*, since it has more than one eigenvector (two Jordan blocks) but lacks a full setting of eigenvectors (four Jordan blocks). Then $(A_{\text{inf}}, B_{\text{inf}})$ has the following schematic staircase form:

$$(A_{\text{inf}}, B_{\text{inf}}) = \left(\left[\begin{array}{c|c|cc} \mathbf{z} & y & x & x \\ \hline 0 & \mathbf{y} & x & x \\ \hline 0 & 0 & \mathbf{x} & \mathbf{x} \\ 0 & 0 & 0 & \mathbf{x} \end{array} \right], \left[\begin{array}{c|c|cc} 0 & \mathbf{y} & x & x \\ \hline 0 & 0 & \mathbf{x} & \mathbf{x} \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] \right).$$

The bold numbers $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ in A_{inf} represent diagonal blocks of full rank, and the $\mathbf{x}$ and $\mathbf{y}$ in B_{inf} represent superdiagonal blocks of full row rank. Outgoing from the bottom right corner of B_{inf}, the sizes of the diagonal blocks (stairs) $w = (2, 1, 1)$ are the *Weyr characteristics* of the infinite eigenvalue. These indices relate to the dimensions of the nullspaces $\mathcal{N}(B^j)$ such that $\sum_{k=1}^{j} w_k = \dim \mathcal{N}(B^j)$ for $j = 1, 2, 3$. In other words, w_j is the number of Jordan blocks of size $\geq j$. Now, the infinite Jordan structure can be read off from w giving the *Segre characteristics* $s = (3, 1)$, where s_1 is the size of the largest Jordan block, s_2 is the size of the second largest block and so on. Both w and s sum up to the *algebraic multiplicity* and w_1 is the *geometric multiplicity* of the infinite eigenvalue.

After such a preprocessing deflation of the infinite eigenvalues of (A, B), we apply the QZ algorithm to the matrix pair (A_{11}, B_{11}) in (4.1). For more introductory material on singular matrix pairs and the GUPTRI form see [22] and the references therein.

Exploiting knowledge of structure. In some cases, infinite eigenvalues can be reliably deflated by taking into account knowledge on the structure of the matrices A and B. If this is feasible by orthogonal transformations, this is the recommended way of dealing with infinite eigenvalues, as the decision which eigenvalues are considered infinite is not affected by roundoff error. In the context of DAEs, several frameworks have been developed that can help identify and exploit such structures, see, e.g., [19, 33]. The following example is closely related to work by Stykel [40], in which (A, B) arises from a semi-discretized Stokes equation.

EXAMPLE 4.1. *Consider* $A = \left[\begin{smallmatrix} K & L \\ L^T & 0 \end{smallmatrix}\right]$ *and* $B = \left[\begin{smallmatrix} M & 0 \\ 0 & 0 \end{smallmatrix}\right]$*, where L is an $m\times(n-m)$ matrix of full column rank ($n \leq 2m$) and M is an $m \times m$ symmetric positive definite matrix. By a QR decomposition of L we may transform the matrix pair (A, B) to*

$$(A,B) \leftarrow \left(\begin{bmatrix} K_{11} & K_{12} & L_1 \\ K_{21} & K_{22} & 0 \\ L_1^T & 0 & 0 \end{bmatrix}, \begin{bmatrix} M_{11} & M_{12} & 0 \\ M_{21} & M_{22} & 0 \\ 0 & 0 & 0 \end{bmatrix} \right),$$

where L_1 is an $m \times m$ invertible matrix. The submatrix M_{22} is again symmetric positive definite, which in particular yields its invertibility. By a simple block permutation, A and B can be transformed to block upper triangular form,

$$(A, B) \leftarrow \left(\begin{bmatrix} L_1 & K_{12} & K_{11} \\ 0 & K_{22} & K_{21} \\ 0 & 0 & L_1^T \end{bmatrix}, \begin{bmatrix} 0 & M_{12} & M_{11} \\ 0 & M_{22} & M_{21} \\ 0 & 0 & 0 \end{bmatrix} \right).$$

Thus, the eigenvalues of the matrix pair (K_{22}, M_{22}) constitute the finite eigenvalues of (A, B).

4.2. Deflation of infinite eigenvalues within the QZ algorithm. Although preprocessing is the preferable way of dealing with infinite eigenvalues, there can be good reasons to let the QZ algorithm do this job, particularly if the reliable detection of infinite eigenvalues is not a major concern. One reason is that computing a GUPTRI form is quite a costly procedure [13]. This and the following two subsections are concerned with existing and new approaches to deflate infinite eigenvalues that are signaled by tiny diagonal entries of the matrix T in a Hessenberg-triangular matrix pair (H, T).

For testing the smallness of a diagonal entry t_{jj} we may, similar to (3.2)–(3.3), either use the norm-wise criterion

$$|t_{jj}| \le \mathbf{u} \cdot \|T\|_F, \tag{4.2}$$

as implemented in the LAPACK routine `DHGEQZ`, or the neighbor-wise criterion $|t_{jj}| \le \mathbf{u} \cdot (|t_{j-1,j}| + |t_{j,j+1}|)$. The latter criterion might help avoid artificial infinite eigenvalues caused by a poor scaling of the matrix pair. Let us briefly sketch the procedure developed by Moler and Stewart [35] for deflating an infinite eigenvalue after t_{jj} has been set to zero, for the case $n = 5$ and $j = 3$:

$$(H, T) = \left(\begin{bmatrix} h & h & h & h & h \\ h & h & h & h & h \\ 0 & h & h & h & h \\ 0 & 0 & h & h & h \\ 0 & 0 & 0 & h & h \end{bmatrix}, \begin{bmatrix} t & t & t & t & t \\ 0 & t & t & t & t \\ 0 & 0 & 0 & t & t \\ 0 & 0 & 0 & t & t \\ 0 & 0 & 0 & 0 & t \end{bmatrix} \right).$$

First, a Givens rotation is applied to columns 2 and 3 to annihilate t_{22}, followed by a Givens rotation acting on rows 3 and 4 to annihilate the newly introduced nonzero entry h_{42}:

$$(H, T) \leftarrow \left(\begin{bmatrix} h & \hat{h} & \hat{h} & h & h \\ h & \hat{h} & \hat{h} & h & h \\ 0 & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ 0 & \hat{0} & \hat{h} & \hat{h} & \hat{h} \\ 0 & 0 & 0 & h & h \end{bmatrix}, \begin{bmatrix} t & \hat{t} & \hat{t} & t & t \\ 0 & \hat{0} & \hat{t} & t & t \\ 0 & 0 & 0 & \hat{t} & \hat{t} \\ 0 & 0 & 0 & \hat{t} & \hat{t} \\ 0 & 0 & 0 & 0 & t \end{bmatrix} \right).$$

In a similar manner, the two zero diagonal entries in T are pushed one step upwards:

$$(H, T) \leftarrow \left(\begin{bmatrix} \hat{h} & \hat{h} & h & h & h \\ \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ \hat{0} & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ 0 & 0 & h & h & h \\ 0 & 0 & 0 & h & h \end{bmatrix}, \begin{bmatrix} \hat{0} & \hat{t} & t & t & t \\ 0 & 0 & \hat{t} & \hat{t} & \hat{t} \\ 0 & 0 & \hat{t} & \hat{t} & \hat{t} \\ 0 & 0 & 0 & t & t \\ 0 & 0 & 0 & 0 & t \end{bmatrix} \right).$$

Finally, a Givens rotation acting on rows 1 and 2 is used to deflate the infinite eigenvalue at the top left corner:

$$(H, T) \leftarrow \left(\left[\begin{array}{c|cccc} \hat{h} & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ \hline \hat{0} & \hat{h} & \hat{h} & \hat{h} & \hat{h} \\ 0 & h & h & h & h \\ 0 & 0 & h & h & h \\ 0 & 0 & 0 & h & h \end{array} \right], \left[\begin{array}{c|cccc} 0 & \hat{t} & \hat{t} & \hat{t} & \hat{t} \\ \hline 0 & \hat{t} & \hat{t} & \hat{t} & \hat{t} \\ 0 & 0 & t & t & t \\ 0 & 0 & 0 & t & t \\ 0 & 0 & 0 & 0 & t \end{array} \right] \right).$$

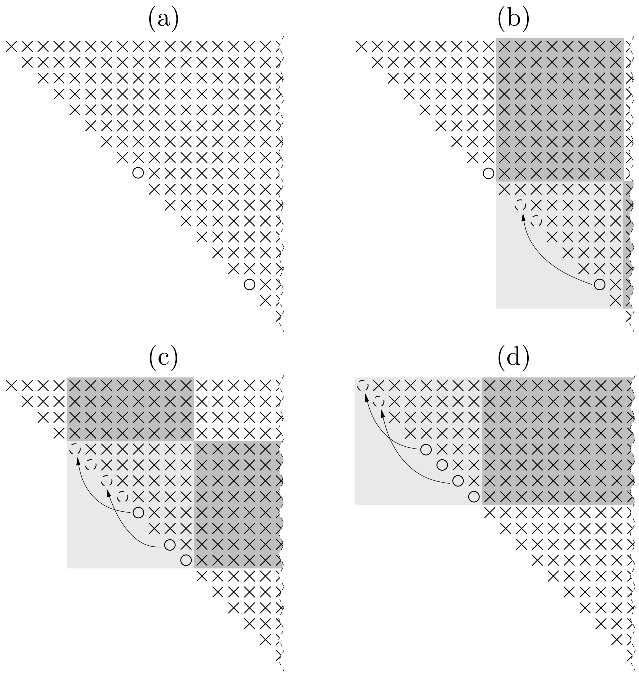


FIG. 4.1. *Illustration of windowing algorithm for deflating two infinite eigenvalues.*

The outlined procedure requires roughly $6jn$ flops for updating each of the factors H, T, Q, and Z. If $j > n/2$, it is cheaper to push the infinite eigenvalue to the bottom right corner.

4.3. Windowing techniques for deflating infinite eigenvalues. The algorithm described in the previous subsection performs $\mathcal{O}(jn)$ flops while accessing $\mathcal{O}(jn)$ memory, making it costly in terms of execution time on computing systems with deep memory hierarchies. If the dimension of the matrix pair is large and many infinite eigenvalues are to be deflated, this degrades the overall performance of the multishift QZ algorithm. A higher computation/communication ratio can be attained by using windowing techniques similar to those proposed in [5, 11, 30]. In the following, we illustrate such an algorithm, conceptually close to a recently presented block algorithm for reordering standard and generalized Schur forms [30].

Consider a matrix pair (H, T) in Hessenberg-triangular form, where the 9^{th} and the 16^{th} diagonal entries of T are zero, see Figure 4.1(a). Both zero entries will be pushed simultaneously in a window-by-window fashion to the top left corner. The first step consists of pushing the lower zero diagonal entry to the top left corner of the 8-by-8 window marked by the light-gray area in Figure 4.1(b). This creates zero diagonal entries at positions 11 and 12. Note that it makes little sense to push one step further; the leading zero at position 10 would be destroyed when pushing the zero diagonal entry at position 9. During this procedure, only the entries of H and T that reside within the window are updated and the data representing the performed Givens rotations is pipelined, see [30] for more details. Afterwards, the pipelined transformations are applied to the parts outside the window marked by dark-gray areas in Figure 4.1(b) as well as to the corresponding parts of the transformation matrices Q and Z. To maintain locality of the memory reference pattern, rows are updated in stripes of n_b columns (in the computational environments we considered, choosing $n_b = 32$ was nearly optimal). The next window contains the diagonal positions $5, \dots, 12$, see Figure 4.1(c). The zeros at positions 9 and 11 are subsequently pushed to positions 5 and 7, respectively. Again, the update of parts outside the window as well as the update of the transformation matrices are delayed as described above. The last 8-by-8 window resides in the top left corner and yields the deflation of two infinite eigenvalues, see Figure 4.1(d).

Note that we have only provided the generic picture; pushing a zero diagonal entry in T may leave "traces" in the form of additional zero diagonal entries. A proper implementation of the windowing algorithm has to take care of such events. Moreover, to achieve optimal performance, the number of zero diagonal entries to be simultaneously pushed and the window size should be significantly larger than those chosen in the descriptive example, see Section 6.2.

4.4. Infinite eigenvalues and multishift QZ iterations. An important observation made in [50] is that the shift transmission mechanism works even if T is singular, provided that this singularity does not affect the generalized Francis shifts or the definition of the first column of the shift polynomial. In fact, Theorem 2.4 only assumes that the intended shifts are finite and that the m^{th} leading principal submatrix of T is nonsingular.

Hence, zero diagonal entries at positions $m+1, \dots, n-m$ in T do not affect the information contained in the bulge pairs and do consequently not affect the convergence of the QZ iteration. What happens to such a zero diagonal entry if a bulge pair passes through it? This question has been addressed by Ward [44, 45] for $m \le 2$, and by Watkins for general m [50]. The answer is that the zero diagonal entry moves m positions upwards along the diagonal. Note that although it is assumed in [50] that the multishift QZ iteration is based on Givens rotations, the same answer holds for a QZ iteration based on (opposite) Householder matrices, see [29].

These results imply that infinite eigenvalues need only be deflated if they correspond to zero diagonal entries at positions $1, \dots, m$ and $n-m+1, \dots, n$ of T. Other zero diagonal entries will be automatically moved upwards in the course of the QZ algorithm to the top diagonal positions, where they then can be deflated. Note, however, that this "transport" of zero diagonal elements only holds under the assumption of exact arithmetic; it can be severely affected by roundoff error.

EXAMPLE 4.2. *Consider the* 10×10 *matrix pair*

$$(H,T) = \left(\begin{bmatrix} 3 & 3 & \cdots & \cdots & 3 \\ 1 & 3 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \vdots \\ & & \ddots & 3 & 3 \\ & & & 1 & 3 \end{bmatrix}, \begin{bmatrix} 1 & 1 & \cdots & \cdots & 1 \\ & 0 & \ddots & & \vdots \\ & & \ddots & \ddots & \vdots \\ & & & 0 & 1 \\ & & & & 1 \end{bmatrix} \right)$$

in Hessenberg-triangular form. It can be shown that this matrix pair has four infinite eigenvalues. Applying a single-shift QZ iteration, Algorithm 1 with $m = 1$*, once to* (H,T) *leads to an updated triangular matrix* T *with the leading diagonal entry being exactly zero. None of the other diagonal entries, however, satisfies the deflation criterion (4.2). Also in the course of further QZ iterations applied to the deflated matrix pair no other infinite eigenvalue can be detected. After convergence, the three remaining infinite eigenvalues of* (H,T) *have been perturbed to finite eigenvalues of magnitude* $\approx 1.9 \times 10^5$*. On the other hand, if all entries of* T *satisfying (4.2) are subsequently deflated* before *any QZ iteration, then all four infinite eigenvalues are detected.*

Example 4.2 reveals that not taking care of all (nearly) zero diagonal entries in T increases the chances that infinite eigenvalues go undetected. Besides the obvious disadvantages, failing to detect infinite eigenvalues may have an adverse effect on the convergence of the QZ algorithm [27, 50]. There is no simple cure for the effects observed in Example 4.2. Setting a diagonal entry, which is known to be zero in exact arithmetic but does not satisfy (4.2), explicitly to zero would spoil the backward stability of the QZ algorithm. We therefore recommend to take care of *all* nearly zero diagonal entries in T before applying a QZ iteration.

5. Aggressive early deflation applied to the QZ algorithm. The idea behind the aggressive early deflation strategy in the QZ algorithm is to enhance the deflation strategy described in Section 3.3 by taking advantage of perturbations outside the subdiagonal entries of the Hessenberg matrix, as in the QR algorithm [7]. This gives the possibility to identify and deflate converged eigenvalues much earlier than either of the deflation criteria (3.2) and (3.3) would do, which results in fewer QZ iterations and thereby has the potential to save both floating point operations and execution time.

5.1. Pairs of reducing perturbations. For simplicity, we consider an $n \times n$ *complex* unreduced Hessenberg-triangular matrix pair (H, T). Let P_H and P_T be complex perturbation matrices. Suppose there exist a unitary matrix Q of the form $Q = \left[\begin{smallmatrix} 1 & 0 \\ 0 & \tilde{Q} \end{smallmatrix}\right]$ and a unitary matrix Z such that the transformed perturbed matrix pair,

$$(\hat{H}, \hat{T}) \equiv Q^H (H + P_H, T + P_T) Z, \tag{5.1}$$

is in reduced Hessenberg-triangular form:

$$\hat{H} = \begin{bmatrix} \hat{H}_{11} & \hat{H}_{12} \\ 0 & \hat{H}_{22} \end{bmatrix}, \qquad \hat{T} = \begin{bmatrix} \hat{T}_{11} & \hat{T}_{12} \\ 0 & \hat{T}_{22} \end{bmatrix}. \tag{5.2}$$

Then, in analogy to the matrix case, (P_H, P_T) is called a *reducing perturbation pair.* If the norm of (P_H, P_T) is tiny, the equivalence transformation above has split the problem of computing the eigenvalues of (H, T) in two (or more) subproblems of smaller size without affecting the backward stability of the QZ algorithm.

In the following, we derive results that characterize and identify pairs of reducing perturbations, which are extensions of similar results for the matrix case [7].

Lemma 5.1. *Let (H, T) with $H, T \in \mathbb{C}^{n\times n}$ be in unreduced Hessenberg-triangular form and $P_H, P_T \in \mathbb{C}^{n\times n}$. Assume that $T + P_T$ is invertible. Then (P_H, P_T) is a* reducing perturbation pair *for (H, T),* if and only if *the regular matrix pair $(H + P_H, T + P_T)$ has a left (generalized) eigenvector $y \in \mathbb{C}^n$ with a zero first component, $y_1 = 0$.*

Proof. Assume (P_H, P_T) is a reducing perturbation pair for (H, T), i.e., there exist a unitary matrix $Q = \left[\begin{smallmatrix} 1 & 0 \\ 0 & \tilde{Q} \end{smallmatrix}\right]$ and a unitary matrix Z such that $(\hat{H}, \hat{T})$ defined by (5.1) is in reduced block-triangular form (5.2). This implies that $(\hat{H}, \hat{T})$ has a left (generalized) eigenvector $\hat{y}$ with $\hat{y}_1 = 0$; indeed, the first $\dim(\hat{H}_{11})$ components equal to zero. Since Q has block diagonal structure, it follows that $y = Q\hat{y}$ is a left eigenvector of $(H + P_H, T + P_T)$ with $y_1 = \hat{y}_1 = 0$.

In the opposite direction, assume that $(H + P_H, T + P_T)$ has a left (generalized) eigenvector $y \in \mathbb{C}^n$ with a zero first component, $y_1 = 0$, associated with the eigenvalue pair $(\alpha, \beta) \in \mathbb{C}^2$, i.e., $\beta y^H (H + P_H) = \alpha y^H (T + P_T)$. By replacing the initial QR factorization of B in the standard algorithm for reducing a matrix pair (A, B) to Hessenberg-triangular form [18, Alg. 7.7.1] by an RQ factorization of B, we construct unitary matrices $Q = \left[\begin{smallmatrix} 1 & 0 \\ 0 & \tilde{Q} \end{smallmatrix}\right]$ and Z such that $Q^H(H + P_H, T + P_T)Z = (\hat{H}, \hat{T})$ is in Hessenberg-triangular form. It follows that $\hat{y} = Q^H y$ is a left (generalized) eigenvector of $(\hat{H}, \hat{T})$ and $\hat{y}_1 = y_1 = 0$ due to the fact that Q is block diagonal. Let k be the smallest index for which $\hat{y}_k \neq 0$ and partition $\hat{y}^H = [0, z]^H$ with $z \in \mathbb{C}^{n-k+1}$. If $\hat{H}$ and $\hat{T}$ are conformally partitioned,

$$\hat{H} = \left[\begin{array}{cc} \hat{H}_{11} & \hat{H}_{12} \\ h_{k,k-1}e_1 e_{k-1}^T & \hat{H}_{22} \end{array}\right], \qquad \hat{T} = \left[\begin{array}{cc} \hat{T}_{11} & \hat{T}_{12} \\ 0 & \hat{T}_{22} \end{array}\right],$$

then $\beta\, \hat{y}^H \hat{H} = \alpha\, \hat{y}^H \hat{T}$ yields

$$\beta\, [\hat{h}_{k,k-1}\hat{y}_k e_{k-1}^T, z^H \hat{H}_{22}] = \alpha\, [0, z^H \hat{T}_{22}].$$

The nonsingularity of $T + P_T$ implies $\beta \neq 0$, which in turn gives $\hat{h}_{k,k-1} = 0$, i.e., $\hat{H}$ is in reduced Hessenberg form. ∎

Note that the second part of the proof of Lemma 5.1 also shows how orthogonal matrices Q and Z yielding a deflated matrix pair (5.2) can be obtained by a slightly modified form of Hessenberg-triangular reduction. In the context of aggressive deflation, a useful reducing perturbation pair (P_H, P_T) must have enough zero structure so that relatively little work is needed to retransform $(H + P_H, T + P_T)$ to Hessenberg-triangular form. By restricting P_H and P_T to Hessenberg and triangular matrices, respectively, there will be no extra work.

LEMMA 5.2. *(P_H, P_T) is a* reducing perturbation pair *for (H, T) of minimal Frobenius norm in the set of Hessenberg-triangular pairs,* if and only if *P_T is the zero matrix and P_H is zero except for some subdiagonal entry.*

Proof. Let (P_H, P_T) be a reducing perturbation pair in Hessenberg-triangular form. Decompose $P_H = P_H^{(s)} + P_H^{(u)}$ in its subdiagonal part $P_H^{(s)}$ and its upper triangular part $P_H^{(u)}$. Then $(P_H^{(s)}, 0)$ is a reducing perturbation pair of smaller norm. ∎

This choice leads to the small-subdiagonal deflation strategy for the QZ algorithm described in Section 3.3.

In order to reach a more aggressive deflation strategy, we must allow more general perturbations, where (P_H, P_T) is not necessarily in Hessenberg-triangular form. Extending the matrix case, we consider small perturbations P_H and P_T that are nonzero only in its last k rows and $k+1$ columns. Now, if $k \ll n$, the cost is small (compared to a QZ iteration) to retransform the perturbed matrix pair to Hessenberg-triangular form, see also Section 5.2.

Let the matrix pair (H, T) be in unreduced Hessenberg-triangular form with $H, T \in \mathbb{C}^{n\times n}$ and partitioned as follows:

$$H = \left[\begin{array}{ccc} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ 0 & H_{32} & H_{33} \end{array}\right], \qquad T = \left[\begin{array}{ccc} T_{11} & T_{12} & T_{13} \\ 0 & T_{22} & T_{23} \\ 0 & 0 & T_{33} \end{array}\right], \tag{5.3}$$

where the block rows from top to bottom (and block columns from left to right) have $n - n_w - 1, 1$, and n_w rows (columns), respectively. Let the perturbation pair (P_H, P_T)

be partitioned conformally with (H, T), but with the following nonzero structure

$$(5.4) \qquad P_H = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & P_{32}^{(H)} & P_{33}^{(H)} \end{bmatrix}, \qquad P_T = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & P_{33}^{(T)} \end{bmatrix}.$$

A special choice of such perturbations is given in the following lemma.

LEMMA 5.3. *If (α, β) is an eigenvalue pair of $(P_{33}^{(H)}, P_{33}^{(T)})$ with left eigenvector y, normalized such that $\|y\|_2 = 1$, then $(P_H, 0)$ partitioned as in (5.4) with $P_{32}^{(H)} = -(y^H H_{32})y$ and $P_{33}^{(H)} = 0$ is a reducing perturbation pair.*

Proof. We have

$$\beta\,[0, 0, y^H](H + P_H) = \beta\,[0, 0, y^H H_{33}] = \alpha\,[0, 0, y^H T_{33}] = \alpha\,[0, 0, y^H]T.$$

This shows that $[0, 0, y^H]^H$ is a left eigenvector of the perturbed matrix pair $(H + P_H, T + P_T)$ with $P_T = 0$, which together with Lemma 5.1 concludes the proof. ☐

To search for reducing perturbation pairs, we can choose from all, generically n, possible perturbations in the sense of Lemma 5.3. Although this strategy will generally only yield a reducing perturbation of *approximately* minimal Frobenius norm among all pairs of the form (5.4), the perturbations of Lemma 5.3 have the major advantage of being effectively computed and tested. Finding the minimum among all reducing perturbations of the form (5.4) is closely related to finding the distance to uncontrollability of a descriptor system [9]. This connection along with numerical methods for computing the distance to uncontrollability will be studied in a forthcoming paper. However, in preliminary numerical experiments with the multishift QR algorithm we observed that rarely any extra deflations can be gained from using perturbations more general than those of Lemma 5.3.

To illustrate the effectiveness of Lemma 5.3, let us consider the following matrix pair, which has been considered in [1] as an extension of the motivating example in [7]:

$$(5.5) \quad (H, T) = \left(\begin{bmatrix} 6 & 5 & 4 & 3 & 2 & 1 \\ 0.001 & 1 & 0 & 0 & 0 & 0 \\ & 0.001 & 2 & 0 & 0 & 0 \\ & & 0.001 & 3 & 0 & 0 \\ & & & 0.001 & 4 & 0 \\ & & & & 0.001 & 5 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ & 1 & 0 & 0 & 0 & 0 \\ & & 1 & 0 & 0 & 0 \\ & & & 1 & 0 & 0 \\ & & & & 1 & 0 \\ & & & & & 1 \end{bmatrix} \right).$$

Let us consider a partitioning of the form (5.3) for $n_w = 5$. Then the eigenvalues of (H_{33}, T_{33}) are given by the (α, β) pairs $(1, 1), (2, 1), \ldots, (5, 1)$ with $\lambda = \alpha/\beta$. Each of these eigenvalues yields a reducing perturbation pair in the sense of Lemma 5.3. The respective norms of $\|P_{32}^{(H)}\|$ are as follows:

$$1:\ 1.0 \times 10^{-3},\ 2:\ 1.0 \times 10^{-6},\ 3:\ 5.0 \times 10^{-10},\ 4: 1.7 \times 10^{-13},\ 5: 4.2 \times 10^{-17}.$$

In double precision, the eigenvalue 5 can thus be safely deflated. In single precision, even three eigenvalues $(3, 4,$ and $5)$ correspond to a reducing perturbation of norm below machine precision. See also Section 6.3, where this example is studied for larger matrices.

5.2. Implementation aspects. In the following, we describe an efficient method which puts the aggressive early deflation motivated by Lemma 5.3 into practice. For this purpose, we consider a partition of (H, T) of the form (5.3) and focus on the $n_w \times (n_w + 1)$ submatrix pair $([H_{32},\ H_{33}], [0,\ T_{33}])$, which defines the deflation window.

First, by means of the QZ algorithm, orthogonal matrices Q_1 and Z_1 resulting in a generalized Schur decomposition of (H_{33}, T_{33}) are computed. This admits the following partitioning:

$$Q_1^T([H_{32},\, H_{33}], [0,\, T_{33}]) \begin{bmatrix} 1 & 0 \\ 0 & Z_1 \end{bmatrix} = \left(\begin{bmatrix} s_3 & \tilde{H}_{33} & \tilde{H}_{34} \\ s_4 & 0 & \tilde{H}_{44} \end{bmatrix}, \begin{bmatrix} 0 & \tilde{T}_{33} & \tilde{T}_{34} \\ 0 & 0 & \tilde{T}_{44} \end{bmatrix} \right),$$

where s_3, s_4 are column vectors of appropriate size, and $(\tilde{H}_{44}, \tilde{T}_{44})$ is either 1×1, representing a real eigenvalue of (H_{33}, T_{33}), or 2×2, representing a complex conjugate pair of eigenvalues. If $(\tilde{H}_{44}, \tilde{T}_{44})$ represents a real eigenvalue then the corresponding reducing perturbation in the sense of Lemma 5.3 is obtained by setting the scalar s_4 to zero. Similarly, if $(\tilde{H}_{44}, \tilde{T}_{44})$ is 2×2, a reducing perturbation is obtained by setting the two entries of s_4 to zero. This is, strictly speaking, not a perturbation in the sense of Lemma 5.3 and it may happen that one of the two complex conjugate eigenvalues of $(\tilde{H}_{44}, \tilde{T}_{44})$ considered individually yields a reducing perturbation which is significantly smaller than $\|s_4\|_2$. However, deflating this eigenvalue alone is not possible without leaving the realm of real matrices.

There are several possible choices for criteria under which $\|s_4\|_2$, the norm of the reducing perturbation described above, can be considered negligible. A liberal deflation criterion, which just preserves numerical backward stability, is given by

$$\|s_4\|_2 \le \mathbf{u}\|H\|_F. \tag{5.6}$$

A more conservative criterion in the spirit of (3.3) is given by

$$\|s_4\|_2 \le \begin{cases} \mathbf{u}|\tilde{H}_{44}|, & \text{if } \tilde{H}_{44} \text{ is } 1 \times 1, \\ \mathbf{u}\sqrt{|\det(\tilde{H}_{44})|}, & \text{otherwise.} \end{cases} \tag{5.7}$$

This is more preferable for reasons explained in Section 3.3. A range of other criteria can be found in [7, Sec. 2.4].

If $\|s_4\|_2$ satisfies the chosen deflation criterion, we mark $(\tilde{H}_{44}, \tilde{T}_{44})$ as deflatable and apply the described process again to the reduced matrix pair $([s_3,\, \tilde{H}_{33}], \tilde{T}_{33}])$. Otherwise, we mark $(\tilde{H}_{44}, \tilde{T}_{44})$ as undeflatable and reorder the generalized Schur decomposition of (H_{33}, T_{33}) to construct orthogonal matrices Q_2 and Z_2 such that

$$(Q_1Q_2)^T([H_{32},\, H_{33}], [0,\, T_{33}]) \begin{bmatrix} 1 & 0 \\ 0 & Z_1Z_2 \end{bmatrix} = \left(\begin{bmatrix} \bar{s}_3 & \bar{H}_{33} & \bar{H}_{34} \\ \bar{s}_4 & 0 & \bar{H}_{44} \end{bmatrix}, \begin{bmatrix} 0 & \bar{T}_{33} & \bar{T}_{34} \\ 0 & 0 & \bar{T}_{44} \end{bmatrix} \right),$$

where $(\bar{H}_{33}, \bar{T}_{33})$ is of the same order and has the same eigenvalues as $(\tilde{H}_{44}, \tilde{T}_{44})$. In this case, the described process is applied again to the matrix pair $([\bar{s}_4,\, \bar{H}_{44}], \bar{T}_{44}])$. The whole procedure is repeated until the matrix pair vanishes, i.e., $n_w - k$ undeflatable and k deflatable eigenvalues have been found yielding a decomposition of the form

$$Q^T([H_{32},\, H_{33}], [0,\, T_{33}]) \begin{bmatrix} 1 & 0 \\ 0 & Z \end{bmatrix} = \left(\begin{bmatrix} \check{s}_3 & \check{H}_{33} & \check{H}_{34} \\ \check{s}_4 & 0 & \check{H}_{44} \end{bmatrix}, \begin{bmatrix} 0 & \check{T}_{33} & \check{T}_{34} \\ 0 & 0 & \check{T}_{44} \end{bmatrix} \right),$$

where $(\check{H}_{44}, \check{T}_{44})$ is $k \times k$ and contains all deflatable eigenvalues. Moreover, we have $\|\check{s}_4\|_2 \le \sqrt{k}\mathbf{u}\|H\|_F$ no matter whether (5.6) or (5.7) is used. Hence, $\check{s}_4$ can be safely set to zero and the QZ algorithm is continued with the $(n-k) \times (n-k)$ matrix pair

$$(\tilde{H}, \tilde{T}) = \left(\begin{bmatrix} H_{11} & H_{12} & H_{13}Z \\ H_{21} & H_{22} & H_{23}Z \\ 0 & \check{s}_3 & \check{H}_{33} \end{bmatrix}, \begin{bmatrix} T_{11} & T_{12} & T_{13}Z \\ 0 & T_{22} & T_{23}Z \\ 0 & 0 & \check{T}_{33} \end{bmatrix} \right).$$

Note that the matrix pair $(\tilde{H}, \tilde{T})$ is not in Hessenberg-triangular form due to the "spike" $\check{s}_3$. If we apply a Householder matrix $\mathcal{H}_1(\check{s}_3) = I - \beta vv^T$ to the last $n_w - k$ rows of $(\tilde{H}, \tilde{T})$, we have

$$\mathcal{H}_1(\check{s}_3)\check{T}_{33} = \check{T}_{33} - \beta v(\check{T}_{33}^T v)^T.$$

Hence, $\mathcal{H}_1(\check{s}_3)\check{T}_{33}$ is a rank-one update of a triangular matrix. Similar to updating algorithms for the QR decomposition [18, Sec. 12.5], we can construct an orthogonal matrix Z_3 as a sequence of $n_w - k - 1$ Givens rotations such that $Z_3^T \check{T}_{33}^T v = \gamma e_n$ for some $\gamma \in \mathbb{R}$. Consequently,

$$\mathcal{H}_1(\check{s}_3)\check{T}_{33}Z_3 = \check{T}_{33}Z_3 - \beta\gamma v e_n^T$$

is an upper Hessenberg matrix. By another sequence of $n_w - k - 1$ Givens rotations the subdiagonal elements of $\mathcal{H}_1(\check{s}_3)\check{T}_{33}Z_3$ can be eliminated so that $\mathcal{H}_1(\check{s}_3)\check{T}_{33}Z_3Z_4$ becomes upper triangular. The described algorithm requires $\mathcal{O}((n_w - k)^2)$ flops which is favourable compared to the $\mathcal{O}((n_w - k)^3)$ flops needed for computing a RQ factorization of $\mathcal{H}_1(\check{s}_3)\check{T}_{33}$ from scratch. Finally, the standard reduction algorithm [18, Alg. 7.7.1] without the initial QR factorization is applied to the matrix pair $\mathcal{H}_1(\check{s}_3)(\check{H}_{33}, \check{T}_{33})Z_3Z_4$ in order to compute orthogonal matrices $Q_3 = \left[\begin{smallmatrix} 1 & 0 \\ 0 & \tilde{Q}_3 \end{smallmatrix}\right]$ and Z_5 such that $Q_3^T \mathcal{H}_1(\check{s}_3)(\check{H}_{33}, \check{T}_{33})Z_3Z_4Z_5$ is Hessenberg-triangular. Setting

$$\tilde{Q} = \begin{bmatrix} I_{n-n_w} & 0 \\ 0 & \mathcal{H}_1(\check{s}_3)Q_3 \end{bmatrix}, \quad \tilde{Z} = \begin{bmatrix} I_{n-n_w} & 0 \\ 0 & Z_3Z_4Z_5 \end{bmatrix},$$

yields a Hessenberg-triangular matrix pair $\tilde{Q}^T(\tilde{H}, \tilde{T})\tilde{Z}$ from which the multishift QZ algorithm can be continued. Note that before continuing with a multishift QZ iteration it can be beneficial to apply aggressive early deflation again if sufficiently many eigenvalues have been deflated, i.e., if the ratio k/n_w is above a certain threshold, which has been set to 40% in our experiments (parameter #3 in Table 6.1).

6. Computational experiments. To assess their performance and robustness, we have implemented the newly developed variants of the QZ algorithm in Fortran 77 and performed several experiments to be described in the following subsections.

6.1. Computational platform(s). The experiments are carried out on two of the HPC2N clusters, `seth` and `sarek`, which have advanced memory systems with different characteristics.

`seth` has a total of 240 processors and 120 Gbyte of memory. The cluster consists of 120 nodes, dual Athlon MP2000+ (1.667Ghz) with 1 GB memory per node and connected with a Wulfkit3 SCI high speed interconnect in a $4 \times 5 \times 6$ grid. Athlon MP2000+ has a 64 kB instruction and 64 kB data L1 Cache and a 256 kB of integrated L2 cache. Software used: Debian GNU/Linux 3.0, Portland F90 6.0, ATLAS BLAS 3.5.9.

`sarek` has a total of 384 processors and 1.54 Tbyte of memory: The cluster consists of 190 HP DL145 nodes, with dual AMD Opteron 248 (2.2GHz) and 8 GB memory per node, connected in a Myrinet 2000 high speed interconnect. AMD Opteron 248 has a 64 kB instruction and 64 kB data L1 Cache (2-way associative) and a 1024 kB unified L2 Cache (16-way associative). Software used: Debian GNU/Linux 3.1, Portland F90 6.0, Goto BLAS 0.94.

All results reported are run on one processor of the clusters in double precision real arithmetic ($\epsilon_{mach} \approx 2.2 \times 10^{-16}$).

		`seth`	`sarek`
#1	Minimal (sub)matrix pair dimension for multishift QZ iterations	300	300
#2	Minimal (sub)matrix pair dimension for aggressive early deflation	300	300
#3	Minimal success rate for repeated aggressive early deflation	40%	40%
#4	Window size for simultaneous deflation of infinite eigenvalues	60	84
#5	Number of infinite eigenvalues to be deflated simultaneously	20	28

TABLE 6.1
Default values for some parameters of the multishift QZ algorithm with aggressive early deflation.

6.2. Random matrix pairs. The described multishift QZ iterations and deflation algorithms depend on various parameters, which all have some influence on the overall execution time of the QZ algorithm. We have performed numerical experiments with randomly generated matrix pairs and numerous sets of parameters to measure the influence of each individual parameter. In the following, we focus on the three parameters that have been observed to have the largest impact on the execution time and therefore require particular attention:

m – number of simultaneous shifts used in each multishift QZ iteration (integer multiple of n_s),
n_s – number of shifts contained in each bulge during multishift QZ iterations,
n_w – aggressive early deflation window size.

All other parameters turned out to have less influence on the performance and have been set in a heuristic manner. The default values displayed in Table 6.1 yielded good performance for matrix pairs of size $500, \ldots, 3000$. If the order of an active submatrix pair in the course of the multishift QZ algorithm described in Section 3 becomes smaller than parameter #1, it is more efficient to resort to double-shift QZ iterations. Similarly, if the order is smaller than parameter #2, aggressive early deflation is turned off. It is best to choose #1 not smaller than #2. If aggressive early deflation yielded the deflation of k eigenvalues and the ratio n_w/k exceeds parameter #3, another search for early deflations is immediately performed on the deflated matrix pair before applying a (multishift) QZ iteration. Finally, the parameters #4 and #5 represent the window size and the maximal number of infinite eigenvalues to be pushed simultaneously in the block algorithm for deflating infinite eigenvalues described in Section 4.3. It is necessary to choose #4 larger than two times #5; we found choosing #4 three times larger nearly optimal.

To make the new implementation better comparable to the LAPACK implementation, we used throughout all experiments the liberal deflation criteria (3.2), (4.2), and (5.6). The use of the more conservative deflation criteria (3.3) and (5.7) may result in more accurate eigenvalues but may also lead to slightly more QZ iterations.

6.2.1. Influence of m and n_s. To measure the influence of the parameters m and n_s on the performance of the multishift QZ algorithm *without* aggressive early deflation, we generated $n \times n$ matrices A and B having pseudo-random entries uniformly distributed in the interval $[-1, 1]$ and reduced them to Hessenberg-triangular form by applying the LAPACK routine `DGGHRD`. Figures 6.1 and 6.2 display the measured execution times for the following implementations of the QZ algorithm:

`DHGEQZ`: LAPACK implementation as described in the original paper by Moler and Stewart [35] with some of the modifications proposed in [27, 45, 51], see also Section 2.1.

`KDHGEQZ`: Blocked variant of `DHGEQZ`, developed by Dackland and Kågström [11].

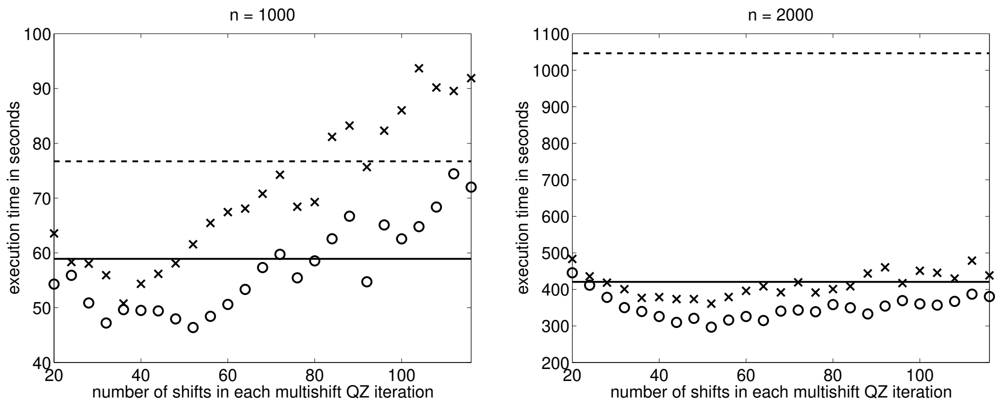


Fig. 6.1. *Random matrix pairs: Execution times on* `seth` *of* `DHGEQZ` *(dashed line),* `KDHGEQZ` *(solid line), and* `MULTIQZ` *<u>without</u> aggressive early deflation for $n_s = 2$ (crosses) and $n_s = 4$ (circles).*

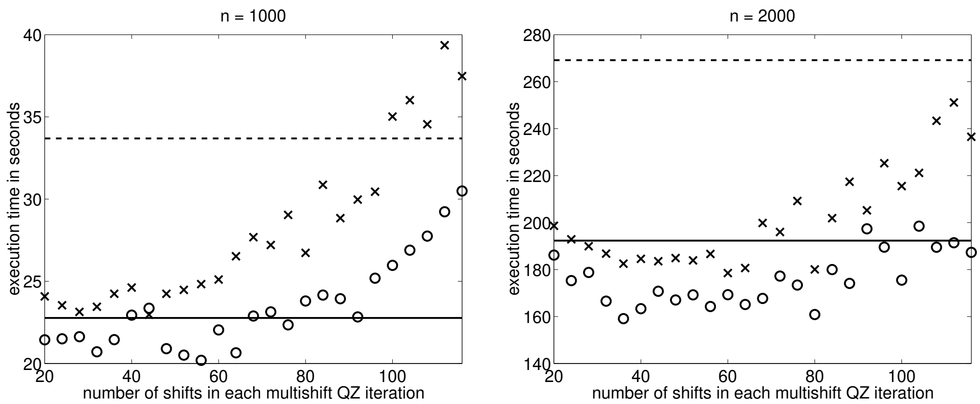


Fig. 6.2. *Random matrix pairs: Execution times on* `sarek` *of* `DHGEQZ` *(dashed line),* `KDHGEQZ` *(solid line), and* `MULTIQZ` *<u>without</u> aggressive early deflation for $n_s = 2$ (crosses) and $n_s = 4$ (circles).*

`MULTIQZ`: Multishift QZ algorithm based on tightly coupled chains of tightly bulges as described in Section 3.

The graphs in Figures 6.1 and 6.2 show the sensitivity of the measured execution times for $n = 1000$ and $n = 2000$ as a function of m, the degree of the multishift polynomial used in `MULTIQZ`, where m is varying between 20 and 116 with step size 4. On `seth` it can be observed that the optimal time for `MULTIQZ` is significantly lower for both $n_s = 2$ and $n_s = 4$ than the time needed by `DHGEQZ` and `KDHGEQZ`. On `sarek` the gained savings are less substantial. In fact, for $n_s = 2$ and $n = 1000$ even with the optimal m, `MULTIQZ` requires slightly more execution time than `KDHGEQZ`.

6.2.2. Influence of n_w. Similar results for the three implementations of the QZ algorithm *with* aggressive early deflation are displayed in Figures 6.3 and 6.4. The graphs show the sensitivity of the measured execution times with respect to n_w, the size of the deflation window, for $n = 1000$ and $n = 2000$. For `DHGEQZ` and `KDHGEQZ` aggressive early deflation has not been performed after each QZ iteration but only after every $m/2$ (double-shift) QZ iterations, meaning that an overall number of m shifts is applied before each search for early deflations. For all three implementations,

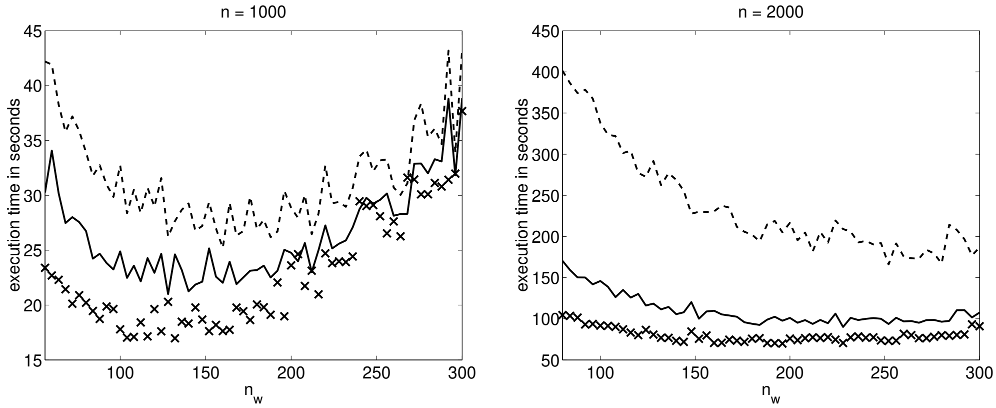


Fig. 6.3. *Random matrix pairs: Execution times on* `seth` *of* `DHGEQZ` *with aggressive early deflation (dashed line),* `KDHGEQZ` *with aggressive early deflation (solid line), and* `MULTIQZ` *with aggressive early deflation (crosses).*

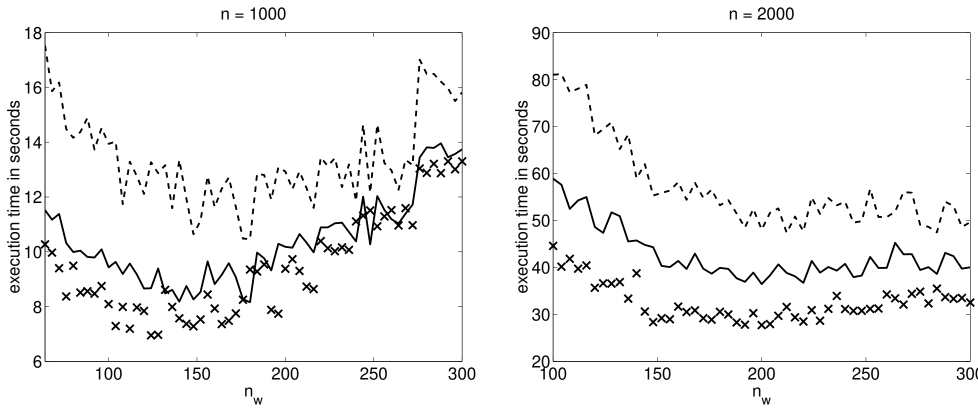


Fig. 6.4. *Random matrix pairs: Execution times on* `sarek` *of* `DHGEQZ` *with aggressive early deflation (dashed line),* `KDHGEQZ` *with aggressive early deflation (solid line), and* `MULTIQZ` *with aggressive early deflation (crosses).*

we have chosen the optimal value for m in the set $\{20, 24, \ldots, 116\}$. Moreover, we have set $n_s = 4$ for `MULTIQZ`. Again, it can be seen that `MULTIQZ` outperforms LAPACK's `DHGEQZ`, but is also faster than `KDHGEQZ`.

6.2.3. Infinite eigenvalues. To generate matrix pairs having plenty of infinite eigenvalues, we generated Hessenberg-triangular matrix pairs (H, T) in the same manner as in the previous two subsections and set each diagonal element of T with probability 0.5 to zero. For $n = 2000$, this resulted in a matrix pair (H, T) with roughly 1000 zero entries on the diagonal of T. Either implementation of the QZ algorithm detected 656 infinite eigenvalues. If only those infinite eigenvalues that correspond to nearly zero diagonal entries at the top left and bottom right corners of T are deflated in the course of the QZ algorithm, see Section 4.4, then a significant portion remains undetected. For example, when using this strategy together with `DHGEQZ` only 399 infinite eigenvalues were detected, which confirms the findings of Example 4.2. On the other hand, this strategy significantly lowers the time spent for dealing with infinite

Deflation strategy	seth	sarek
All ∞ eigenvalues at top left corner (unblocked)	204.8 (0.70)	71.6 (0.66)
All ∞ eigenvalues at top left corner (blocked)	142.1 (0.56)	59.8 (0.54)
All ∞ eigenvalues at nearest corner (unblocked)	137.0 (0.60)	43.5 (0.55)
All ∞ eigenvalues at nearest corner (blocked)	91.6 (0.45)	37.1 (0.44)
Necessary ∞ eigenvalues at nearest corner (unblocked)	99.6 (0.11)	43.9 (0.13)
Necessary ∞ eigenvalues at nearest corner (blocked)	94.1 (0.05)	42.8 (0.11)

Table 6.2

Infinite eigenvalues: Execution times in seconds on `seth` *and* `sarek` *of* `MULTIQZ` *with aggressive early deflation for a* 2000×2000 *random matrix pair with* 656 *infinite eigenvalues. The numbers shown in brackets correspond to the part (number between 0 and 1) of the execution time that was spent for deflating infinite eigenvalues.*

eigenvalues. This can be seen in the last two rows of Table 6.2, which lists execution times for various strategies used in `MULTIQZ` with $n_s = 4$ and the optimal values for m and n_w obtained from Section 6.2.2. Note, however, that failing to detect infinite eigenvalues adversely affects the convergence of the QZ algorithm; the time spent for QZ iterations increases from 61 to 89 seconds on `seth`. We remark that the use of the windowing technique described in Section 4.3 is denoted by "(blocked)" in Table 6.2. There are other interesting observations in the figures of this table. Deflating infinite eigenvalues at the nearest corner of the matrix T (and not at only one corner as it is done in `DHGEQZ`) is a simple means to significantly lower the execution time. Roughly the same portion of time can be saved by using the windowing technique. The most efficient strategy, which detects all 656 infinite eigenvalues, is a combination of both techniques, deflation at the nearest corner combined with windowing.

6.3. Aggressive early deflation at its best. In exceptional cases, aggressive early deflation can have a dramatic positive effect on the computational time. Such a case are matrix pairs of the form (5.5), for which rarely any QZ iterations are needed to deflate eigenvalues. The graphs in Figure 6.5 show the measured execution times of the three implementations of the QZ algorithm with and without aggressive early deflation for $n = 600$ to 3000 (`seth`) or 4000 (`sarek`) with step size 200.

We remark that since aggressive early deflation works so well, the time spent for (multishift) QZ iterations is negligible compared to the overall time. In fact, the timings for `DHGEQZ`, `KDHGEQZ` and `MULTIQZ` with aggressive early deflation are virtually identical and orders of magnitude better than without early deflation. For example, for $n = 4000$ the time of `DHGEQZ` is reduced from nearly one hour to less than 7 seconds.

6.4. Examples from applications. The purpose of this section is to summarize the performance of the multishift QZ algorithm with aggressive early deflation for matrix pairs that arise from practically relevant applications. We have selected 16 matrix pairs from the Matrix Market collection [3], 6 matrix pairs from model reduction benchmark collections [10, 28], and 4 matrix pairs arising from the computation of corner singularities of elliptic PDEs [36]. A more detailed description of the selected matrix pairs along with individual performance results can be found in Appendix A. In the following, we summarize these results and sort the matrix pairs into groups according to their order n as shown in the following table.

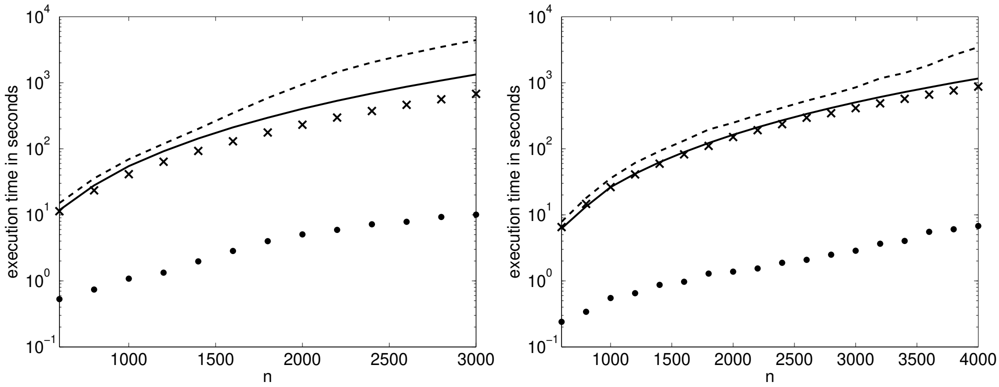


Fig. 6.5. *BBM/ADK example [1, 7]: Execution times (in logarithmic scale) on* `seth` *(left figure) and* `sarek` *(right figure) of* `DHGEQZ` *(dashed line),* `KDHGEQZ` *(solid line),* `MULTIQZ` *(crosses) without aggressive early deflation, and* `DHGEQZ/KDHGEQZ/MULTIQZ` *with aggressive early deflation (dots).*

Group	G1	G2	G3	G4
Order	$n \in [485, 1000]$	$n \in [1001, 1500]$	$n \in [1501, 2000]$	$n \in [2001, 3600]$
#pairs	6	8	5	7

Matrix pairs arising from applications differ in many aspects from random matrix pairs. An aspect which can particularly affect the performance of the QZ algorithm is that the matrix entries in most of the pairs from the Matrix Market collection differ wildly in magnitude. Bad scaling makes the performance of the QZ algorithm erratic and much less predictive than for random matrix pairs. For example, LAPACK's `DHGEQZ` requires 338 seconds for a 1900×1900 matrix pair arising from the discretized heat equation (see Section A.19) but less than 25 seconds for the 1922×1922 matrix pair consisting of the matrices BCSSTK26 and BCSSTM26 from the Matrix Market collection (see Section A.12). Balancing can remedy bad scaling but we have decided not to make us of it since this preprocessing step is by default turned off in most major software environments such as Matlab.

We have tested `DHGEQZ`, `KDHGEQZ`, and `MULTIQZ` for all possible combinations of the parameters n_w (aggressive early deflation window size), m (number of shifts before each aggressive early deflation) and n_s (number of shifts per bulge) satisfying $n_w \in \{40, 60, 80, \ldots, 400\}$, $m \in \{24, 32, 40, \ldots, 160\}$ and $n_s \in \{2, 4\}$. Due to the memory limitations of `seth`, all numerical experiments described in the following have only been performed on `sarek`. Also, to limit the variety of parameters, we have turned off the blocked algorithms for deflating infinite eigenvalues. Column 3 of Table 6.3 shows for each of the four groups the average times in seconds of `DHGEQZ` and `KDHGEQZ` without aggressive early deflation. The fourth column displays the average computing times for all three implementations with aggressive early deflation obtained by choosing m and n_w optimally for *each* matrix pair in the group. The fifth column displays similar times obtained by choosing m and n_w optimally to yield best average performance for *all* matrix pairs together in each group. The corresponding choices of m and n_w are listed in columns 6 and 7, respectively. The difference between the figures of columns 4 and 5 is roughly 10%, which demonstrates that nearly optimal average performance can be obtained without having to optimize m and n_w for each matrix pair individually. Ideally, m and n_w should be chosen adaptively within the QZ

Group	Implementation	w/o agg.	optimal	ave. opt.	m	n_w
G1	`DHGEQZ+AGG`	8.86	4.61	5.03	24	120
	`KDHGEQZ+AGG`	7.07	3.62	3.78	24	100
	`MULTIQZ`($n_s = 2$)`+AGG`	–	3.36	3.45	24	60
	`MULTIQZ`($n_s = 4$)`+AGG`	–	3.40	3.56	24	40
G2	`DHGEQZ+AGG`	55.5	23.7	26.5	56	180
	`KDHGEQZ+AGG`	41.4	17.2	18.9	40	140
	`MULTIQZ`($n_s = 2$)`+AGG`	–	15.3	16.5	72	160
	`MULTIQZ`($n_s = 4$)`+AGG`	–	15.1	16.3	88	180
G3	`DHGEQZ+AGG`	130.3	53.6	56.4	48	220
	`KDHGEQZ+AGG`	89.9	36.5	38.8	72	200
	`MULTIQZ`($n_s = 2$)`+AGG`	–	30.3	30.7	56	200
	`MULTIQZ`($n_s = 4$)`+AGG`	–	27.5	30.0	88	200
G4	`DHGEQZ+AGG`	479	157	170	48	340
	`KDHGEQZ+AGG`	271	97	104	40	220
	`MULTIQZ`($n_s = 2$)`+AGG`	–	80	85	72	220
	`MULTIQZ`($n_s = 4$)`+AGG`	–	115	124	80	220

TABLE 6.3

Application examples: Summary of measured execution times and choice of parameters m and n_w that give optimal average performance.

algorithm, but it is not clear how such a strategy can be effectively realized.

On average, the multishift QZ algorithm with aggressive early deflation (`MULTIQZ +AGG`, $n_s = 2$) is between 2.6 and 6 times faster than the original LAPACK implementation (`DHGEQZ`). Surprisingly, the block version of Dackland and Kågström is, when equipped with aggressive early deflation (`KDHGEQZ+AGG`), only 10% to 20% slower than the multishift QZ algorithm. There is little justification for setting n_s, the number of shifts per bulge, to $n_s = 4$ in favour of $n_s = 2$, in contrast to the results for random matrix pairs.

We have also tested the backward stability of the new variants of the QZ algorithm by measuring the residual $\|(\hat{Q}^T A\hat{Z} - \hat{S}, \hat{Q}^T B\hat{Z} - \hat{T})\|_F$ of the computed Schur decomposition $(\hat{S}, \hat{T})$ as well as the orthogonality $\|\hat{Q}^T\hat{Q} - I\|_F, \|\hat{Z}^T\hat{Z} - I\|_F$ of the computed transformation matrices $\hat{Q}$ and $\hat{Z}$. The obtained results are of the same order as those obtained using the LAPACK implementation.

7. Conclusions. In this paper, we have developed new multishift variants of the QZ algorithm using advanced deflation techniques, which significantly improve upon the performance compared to all existing implementations. It is planned that an implementation of our multishift QZ algorithm with aggressive early deflation is included in a coming release of LAPACK. The ideas presented here are currently applied to the development of a distributed memory QZ algorithm. Future work also includes the investigation of extending the described results to even more general versions of the QR algorithm, such as the periodic QR and QZ algorithms.

8. Final Remarks and Acknowledgments. The work presented in this article is based on preliminary results derived in [1, 29]. The authors are greatly indebted to Ralph Byers and David Watkins for several discussions on the odds and ends of multishift QR and QZ algorithms. The computational experiments in Section 6 were performed using facilities of the High Performance Computing Center North (HPC2N) in Umeå.

REFERENCES

[1] B. Adlerborn, K. Dackland, and B. Kågström. Parallel and blocked algorithms for reduction of a regular matrix pair to Hessenberg-triangular and generalized Schur forms. In J. Fagerholm et al., editor, *PARA 2002*, LNCS 2367, pages 319–328. Springer-Verlag, 2002.

[2] E. Anderson, Z. Bai, C. H. Bischof, S. Blackford, J. W. Demmel, J. J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. C. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, PA, third edition, 1999.

[3] Z. Bai, D. Day, J. W. Demmel, and J. J. Dongarra. A test matrix collection for non-Hermitian eigenvalue problems (release 1.0). Technical Report CS-97-355, Department of Computer Science, University of Tennessee, Knoxville, TN, USA, March 1997. Also available online from `http://math.nist.gov/MatrixMarket`.

[4] Z. Bai and J. W. Demmel. On a block implementation of the Hessenberg multishift QR iterations. *Internat. J. High Speed Comput.*, 1:97–112, 1989.

[5] C. H. Bischof, B. Lang, and X. Sun. A framework for symmetric band reduction. *ACM Trans. Math. Software*, 26(4):581–601, 2000.

[6] K. Braman, R. Byers, and R. Mathias. The multishift QR algorithm. I. Maintaining well-focused shifts and level 3 performance. *SIAM J. Matrix Anal. Appl.*, 23(4):929–947, 2002.

[7] K. Braman, R. Byers, and R. Mathias. The multishift QR algorithm. II. Aggressive early deflation. *SIAM J. Matrix Anal. Appl.*, 23(4):948–973, 2002.

[8] K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical solution of initial-value problems in differential-algebraic equations*, volume 14 of *Classics in Applied Mathematics*. SIAM, Philadelphia, PA, 1996. Revised and corrected reprint of the 1989 original.

[9] R. Byers. The descriptor controllability radius. In U. Helmke, R. Mennicken, and J. Saurer, editors, *Proceedings of the Conference on the Mathematical Theory of Networks and Systems, MTNS '93*, pages 85–88. Akademie Verlag, Berlin, 1994.

[10] Y. Chahlaoui and P. Van Dooren. A collection of benchmark examples for model reduction of linear time invariant dynamical systems. SLICOT working note 2002-2, WGS, 2002.

[11] K. Dackland and B. Kågström. Blocked algorithms and software for reduction of a regular matrix pair to generalized Schur form. *ACM Trans. Math. Software*, 25(4):425–454, 1999.

[12] J. W. Demmel and B. Kågström. The generalized Schur decomposition of an arbitrary pencil $A - \lambda B$: robust software with error bounds and applications. I. Theory and algorithms. *ACM Trans. Math. Software*, 19(2):160–174, 1993.

[13] J. W. Demmel and B. Kågström. The generalized Schur decomposition of an arbitrary pencil $A - \lambda B$: robust software with error bounds and applications. II. Software and applications. *ACM Trans. Math. Software*, 19(2):175–201, 1993.

[14] J. J. Dongarra, J. Du Croz, I. S. Duff, and S. Hammarling. A set of level 3 basic linear algebra subprograms. *ACM Trans. Math. Software*, 16:1–17, 1990.

[15] A. A. Dubrulle. The multishift QR algorithm – is it worth the trouble? TR 6320-3558, IBM Scientific Center, Palo Alto, CA, 1991.

[16] I. Duff, R. G. Grimes, and J. G. Lewis. Users' guide for the Harwell-Boeing sparse matrix collection. Technical report TR/PA/92/86, CERFACS, 1992.

[17] F.R. Gantmacher. *The Theory of Matrices*. Chelsea, New York, 1960.

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

[19] E. Griepentrog and R. März. *Differential-Algebraic Equations and Their Numerical Treatment*. Teubner Texte zur Mathematik. Teubner-Verlag, Leipzig, 1986.

[20] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia, PA, 1996.

[21] B. Kågström. A direct method for reordering eigenvalues in the generalized real Schur form of a regular matrix pair (A, B). In M. S. Moonen, G. H. Golub, and B. L. R. De Moor, editors, *Linear algebra for large scale and real-time applications (Leuven, 1992)*, volume 232 of *NATO Adv. Sci. Inst. Ser. E Appl. Sci.*, pages 195–218. Kluwer Acad. Publ., Dordrecht, 1993.

[22] B. Kågström. Singular matrix pencils. In Z. Bai, J. W. Demmel, J. J. Dongarra, A. Ruhe, and H. van der Vorst, editors, *Templates for the Solution of Algebraic Eigenvalue Problems*, pages 260–277. SIAM, Philadelphia, PA, 2000.

[23] B. Kågström, P. Ling, and C. F. Van Loan. GEMM-based level 3 BLAS: Algorithms for the model implementations. *ACM Trans. Math. Software*, 24(3):268–302, 1999.

[24] B. Kågström, P. Ling, and C. F. Van Loan. GEMM-based level 3 BLAS: High-performance model implementations and performance evaluation benchmark. *ACM Trans. Math. Software*, 24(3):303–316, 1999.

[25] B. Kågström and P. Poromaa. Computing eigenspaces with specified eigenvalues of a regular

matrix pair (A, B) and condition estimation: theory, algorithms and software. *Numer. Algorithms*, 12(3-4):369–407, 1996.

[26] L. Kaufman. The LZ-algorithm to solve the generalized eigenvalue problem. *SIAM J. Numer. Anal.*, 11:997–1024, 1974.

[27] L. Kaufman. Some thoughts on the QZ algorithm for solving the generalized eigenvalue problem. *ACM Trans. Math. Software*, 3(1):65–75, 1977.

[28] J. G. Korvink and B. R. Evgenii. Oberwolfach benchmark collection. In P. Benner, V. Mehrmann, and D. C. Sorensen, editors, *Dimension Reduction of Large-Scale Systems*, volume 45 of *Lecture Notes in Computational Science and Engineering*, pages 311–316. Springer, Heidelberg, 2005.

[29] D. Kressner. *Numerical Methods and Software for General and Structured Eigenvalue Problems*. PhD thesis, TU Berlin, Institut für Mathematik, Berlin, Germany, 2004.

[30] D. Kressner. Block algorithms for reordering standard and generalized Schur forms. LAPACK working note 171, September 2005.

[31] D. Kressner. On the use of larger bulges in the QR algorithm. *Electron. Trans. Numer. Anal.*, 20:50–63, 2005.

[32] V. N. Kublanovskaya. AB-algorithm and its modifications for the spectral problems of linear pencils of matrices. *Numer. Math.*, 43(3):329–342, 1984.

[33] P. Kunkel and V. Mehrmann. *Differential-Algebraic Equations. Analysis and Numerical Solution*. EMS Publishing House, Zürich, Switzerland, 2006. To appear.

[34] B. Lang. Effiziente Orthogonaltransformationen bei der Eigen- und Singulärwertzerlegung. Habilitationsschrift, 1997.

[35] C. B. Moler and G. W. Stewart. An algorithm for generalized matrix eigenvalue problems. *SIAM J. Numer. Anal.*, 10:241–256, 1973.

[36] C. Pester. CoCoS – computation of corner singularities. Preprint SFB393/05-03, Technische Universität Chemnitz, 2005.

[37] P. J. Rabier and W. C. Rheinboldt. *Nonholonomic motion of rigid mechanical systems from a DAE viewpoint*. SIAM, Philadelphia, PA, 2000.

[38] G. W. Stewart. On the eigensystems of graded matrices. *Numer. Math.*, 90(2):349–370, 2001.

[39] G. W. Stewart and J.-G. Sun. *Matrix Perturbation Theory*. Academic Press, New York, 1990.

[40] T. Stykel. Balanced truncation model reduction for semidiscretized Stokes equation. Technical report 04-2003, Institut für Mathematik, TU Berlin, 2003.

[41] C. Tischendorf. *Solution of index-2-DAEs and its application in circuit simulation*. Dissertation, Humboldt-Univ. zu Berlin, 1996.

[42] P. Van Dooren. Algorithm 590: DSUBSP and EXCHQZ: Fortran subroutines for computing deflating subspaces with specified spectrum. *ACM Trans. Math. Softw.*, 8:376–382, 1982.

[43] C. F. Van Loan. *Generalized Singular Values with Algorithms and Applications*. PhD thesis, The University of Michigan, 1973.

[44] R. C. Ward. *A numerical solution to the generalized eigenvalue problem*. PhD thesis, University of Virginia, Charlottesville, Va., 1974.

[45] R. C. Ward. The combination shift QZ algorithm. *SIAM J. Numer. Anal.*, 12(6):835–853, 1975.

[46] R. C. Ward. Balancing the generalized eigenvalue problem. *SIAM J. Sci. Statist. Comput.*, 2(2):141–152, 1981.

[47] D. S. Watkins. Shifting strategies for the parallel QR algorithm. *SIAM J. Sci. Comput.*, 15(4):953–958, 1994.

[48] D. S. Watkins. Forward stability and transmission of shifts in the QR algorithm. *SIAM J. Matrix Anal. Appl.*, 16(2):469–487, 1995.

[49] D. S. Watkins. The transmission of shifts and shift blurring in the QR algorithm. *Linear Algebra Appl.*, 241/243:877–896, 1996.

[50] D. S. Watkins. Performance of the QZ algorithm in the presence of infinite eigenvalues. *SIAM J. Matrix Anal. Appl.*, 22(2):364–375, 2000.

[51] D. S. Watkins and L. Elsner. Theory of decomposition and bulge-chasing algorithms for the generalized eigenvalue problem. *SIAM J. Matrix Anal. Appl.*, 15:943–967, 1994.

Sec.	Name	n	Brief description
A.1	**BCSST08**	1074	TV studio
A.2	**BCSST09**	1083	Clamped square plate
A.3	**BCSST10**	1086	Buckling of a hot washer
A.4	**BCSST11**	1473	Ore car – lumped mass
A.5	**BCSST12**	1473	Ore car – consistent mass
A.6	**BCSST13**	2003	Fluid flow
A.7	**BCSST19**	817	Part of a suspension bridge
A.8	**BCSST20**	485	Frame within a suspension bridge
A.9	**BCSST21**	3600	Clamped square plate
A.10	**BCSST23**	3134	Part of a 3D globally triangularized building
A.11	**BCSST24**	3562	Calgary Olympic Saddledome arena
A.12	**BCSST26**	1922	Seismic analysis, nuclear power station
A.13	**BCSST27**	1224	Buckling analysis, engine inlet from Boeing jetliner
A.14	**BFW782**	782	Discretized Maxwell's equation
A.15	**DWA512**	512	Square dielectric waveguide
A.16	**MHD3200**	3200	Alfven spectra in magnetohydrodynamics

TABLE A.1
Selected set of matrix pairs from the Matrix Market collection.

Appendix A. Benchmark Suite. This section contains a detailed description of the application-oriented numerical experiments from Section 6.4. We have used matrix pairs from three different sources:

Matrix Market. The Matrix Market collection [3] contains a number of generalized eigenvalue problems, which mostly represent structural engineering problems from the Harwell-Boeing sparse matrix collection [16]. We have selected all real generalized eigenvalue problems of dimension 485 . . . 3600, see Table A.1 for a summary.

Descriptor systems. A linear time-invariant descriptor system takes the form $E\dot{x}(t) = Ax(t)+Fu(t)$, where E, A are $n\times n$ matrices and F is an $n\times m$ matrix. Computing the open loop poles of this system amounts to computing the eigenvalues of the matrix pair (A, E). Table A.2 summarizes our selection of such matrix pairs from the Oberwolfach [28] and SLICOT [10] model reduction benchmark collections.

Corner singularities. The computation of corner singularities of elliptic equations in polyhedral corners leads to quadratic eigenvalue problems of the form

$$(\lambda^2 M + \lambda G + K)x = 0, \tag{A.1}$$

where M, K are symmetric positive definite and G is skew-symmetric, see [36] and the references therein. A standard linearization turns (A.1) into a generalized eigenvalue problem associated with the matrix pair

$$(A, B) = \left(\begin{bmatrix} K & 0 \\ 0 & I \end{bmatrix}, \begin{bmatrix} -G & -M \\ I & 0 \end{bmatrix} \right).$$

Table A.3 provides an overview of a selected set of such matrix pairs taken from [36].

In the following, each subsection is devoted to one of the selected matrix pairs. For each matrix pair, we provide a table of "optimal" execution times of various variants of the QZ algorithm and two plots aiming to provide some intuition on the sensitivity of these execution times with respect to the parameters n_w and m.

Sec.	Name	n	Brief description
A.17	**BEAM**	1992	Linear beam with damping
A.18	**FILTER**	1668	2D model of a tunable optical filter
A.19	**HEAT**	1900	Heat conduction through a beam
A.20	**STEEL**	1357	Optimal cooling of steel profiles
A.21	**MNA1**	578	Modified nodal analysis model
A.22	**MNA4**	980	Modified nodal analysis model

Table A.2
Selected set of matrix pairs from model reduction benchmark collections.

Sec.	Name	n	Brief description
A.23	**CIRC90**	2166	Circular cone, opening angle 90 degrees
A.24	**CIRC120**	1626	Circular cone, opening angle 120 degrees
A.25	**FICH669**	1338	Fichera corner, Dirichlet boundary conditions
A.26	**FICH1227**	2454	Fichera corner, Dirichlet boundary conditions

Table A.3
Matrix pairs arising in the computation of corner singularities.

Each table displays the execution times required by the three implementations of the QZ algorithm explained in more detail in Section 6.2, without and with aggressive early deflation (the latter is denoted by `+AGG`) for each individual matrix pair. We have tested all possible combinations of the parameters n_w (aggressive early deflation window size), m (number of shifts before each aggressive early deflation) and n_s (number of shifts per bulge) satisfying $n_w \in \{40, 60, 80, \ldots, 400\}$, $m \in \{24, 32, 40, \ldots, 160\}$ and $n_s \in \{2, 4\}$. From all these combinations only the minimal execution time and the corresponding parameter setting are displayed in each table.

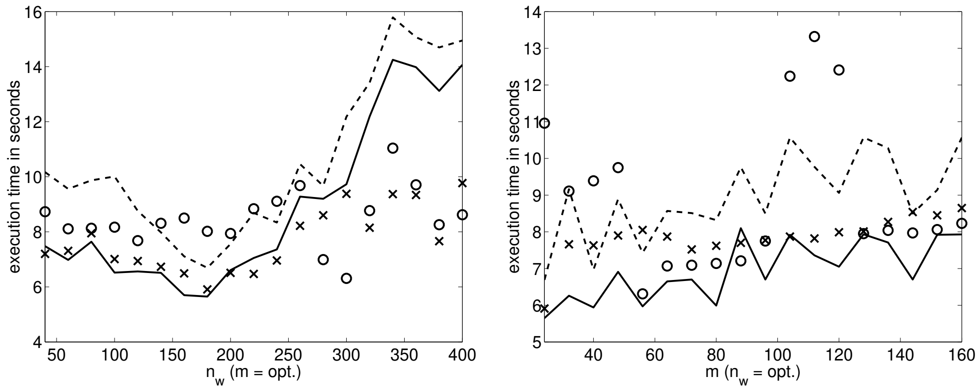
Each pair of plots displays the execution times of `DHGEQZ+AGG` (dashed line), `KDHGEQZ+AGG` (solid line), and `MULTIQZ+AGG` with $n_s = 2$ (crosses) and $n_s = 4$ (circles). The measured times for $n_w \in \{40, 60, 80, \ldots, 400\}$ and fixed, optimally chosen m are shown in the left plot while the times for optimally chosen n_w and $m \in \{24, 32, 40, \ldots, 160\}$ are shown in the right plot.

The grouping of matrix pairs described in Section 6.4 is as follows:

G1 = {A.7, A.8, A.14, A.15, A.21, A.22},
G2 = {A.1, A.2, A.3, A.4, A.5, A.13, A20, A25},
G3 = {A.12, A.17, A.18, A.19, A.24},
G4 = {A.6, A.9, A.10, A.11, A.16, A.23, A.26}.

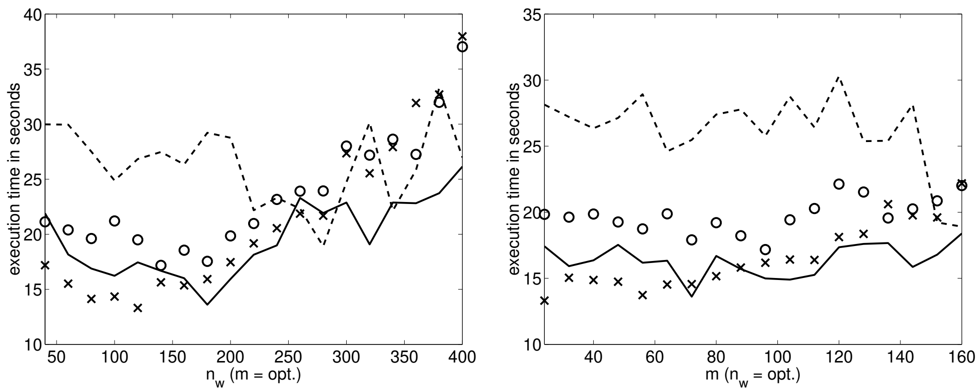
A.1. BCSST08. The 1074×1074 matrix pair BCSST08 consists of the matrices BCSSTK08 and BCSSTM08 from the Matrix Market collection and represents the structural engineering analysis of a TV studio. The QZ algorithm detected no infinite eigenvalues.

BCSST08	opt. time	opt. n_w	opt. m	opt. n_s
`DHGEQZ`	11.1 sec			
`KDHGEQZ`	9.9 sec			
`DHGEQZ+AGG`	6.7 sec	180	24	
`KDHGEQZ+AGG`	5.7 sec	180	24	
`MULTIQZ+AGG`	5.9 sec	180	24	2



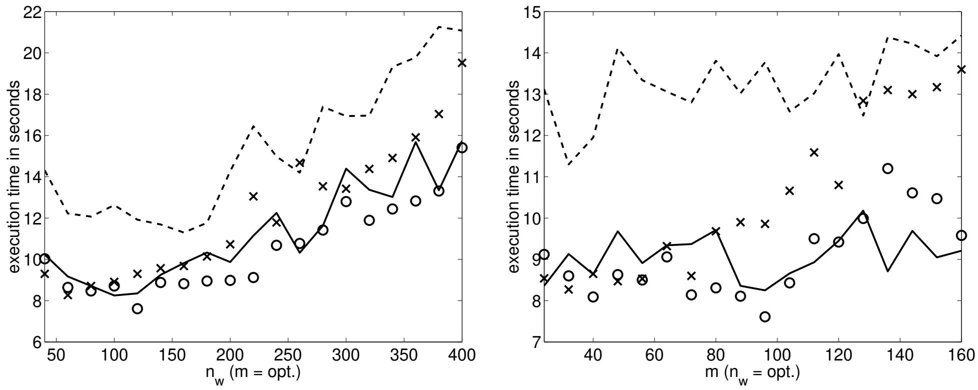
A.2. BCSST09. The 1083×1083 matrix pair BCSST09 consists of the matrices BCSSTK09 and BCSSTM09 from the Matrix Market collection and represents the structural engineering analysis of a clamped square plate. The QZ algorithm detected no infinite eigenvalues.

BCSST09	opt. time	opt. n_w	opt. m	opt. n_s
`DHGEQZ`	36.9 sec			
`KDHGEQZ`	23.7 sec			
`DHGEQZ+AGG`	18.9 sec	280	160	
`KDHGEQZ+AGG`	13.6 sec	180	72	
`MULTIQZ+AGG`	13.3 sec	120	24	2



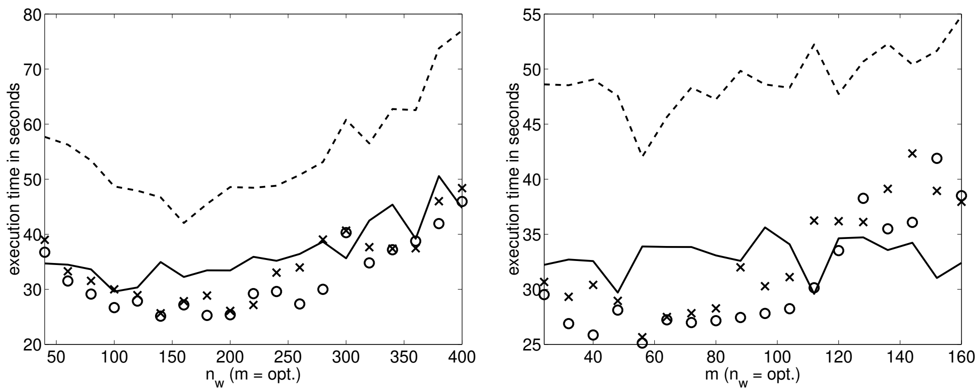
A.3. BCSST10. The 1086×1086 matrix pair BCSST10 consists of the matrices BCSSTK10 and BCSSTM10 from the Matrix Market collection and represents the structural engineering analysis of the buckling of a hot washer. The QZ algorithm detected no infinite eigenvalues.

BCSST10	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	23.3 sec			
KDHGEQZ	18.4 sec			
DHGEQZ+AGG	11.3 sec	160	32	
KDHGEQZ+AGG	8.3 sec	100	96	
MULTIQZ+AGG	7.6 sec	120	96	4



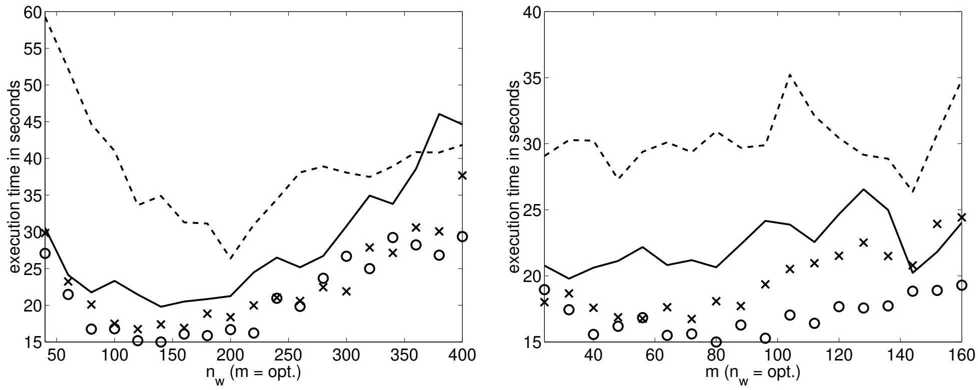
A.4. BCSST11. The 1473×1473 matrix pair BCSST11 consists of the matrices BCSSTK11 and BCSSTM11 from the Matrix Market collection and represents the structural engineering analysis of an ore car with lumped mass. The QZ algorithm detected no infinite eigenvalues.

BCSST11	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	83.5 sec			
KDHGEQZ	54.2 sec			
DHGEQZ+AGG	42.0 sec	160	56	
KDHGEQZ+AGG	29.6 sec	100	112	
MULTIQZ+AGG	25.1 sec	140	56	4



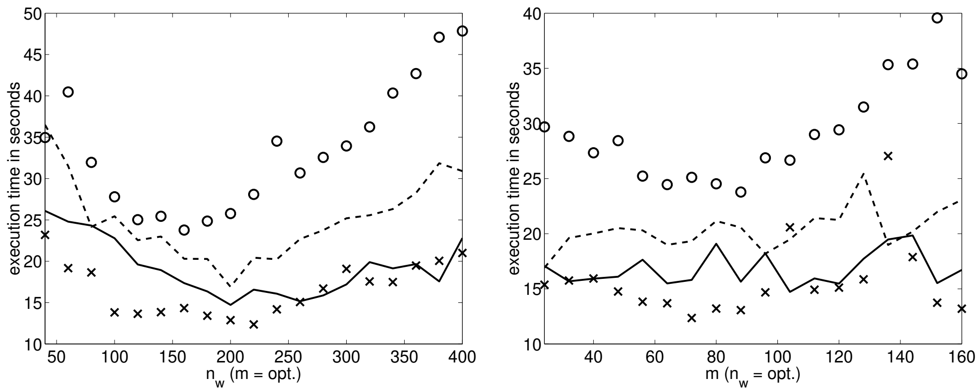
A.5. BCSST12. The 1473×1473 matrix pair BCSST12 consists of the matrices BCSSTK12 and BCSSTM12 from the Matrix Market collection and represents the structural engineering analysis of an ore car with consistent mass. The QZ algorithm detected no infinite eigenvalues.

BCSST12	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	81.8 sec			
KDHGEQZ	62.0 sec			
DHGEQZ+AGG	26.4 sec	200	144	
KDHGEQZ+AGG	19.8 sec	140	32	
MULTIQZ+AGG	15.0 sec	140	80	4



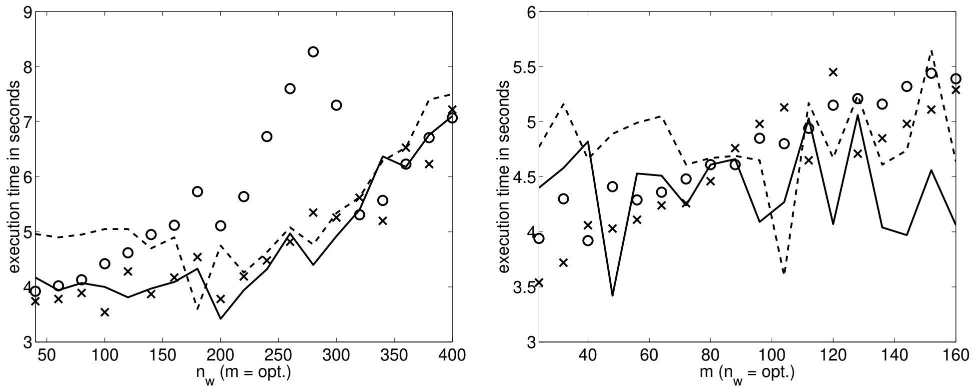
A.6. BCSST13. The 2003×2003 matrix pair BCSST13 consists of the matrices BCSSTK13 and BCSSTM13 from the Matrix Market collection, which correspond to a fluid flow problem. The QZ algorithm detected 762 infinite eigenvalues using DHGEQZ.

BCSST13	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	147.2 sec			
KDHGEQZ	134.3 sec			
DHGEQZ+AGG	16.9 sec	200	24	
KDHGEQZ+AGG	14.7 sec	200	104	
MULTIQZ+AGG	12.4 sec	220	72	2



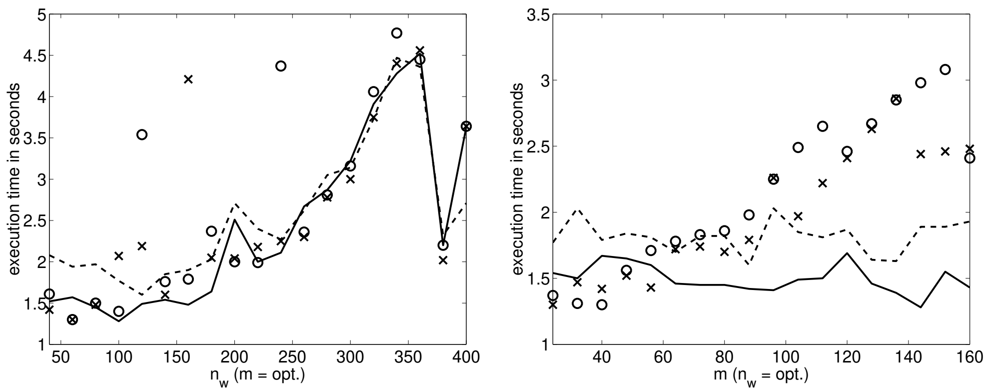
A.7. BCSST19. The 817×817 matrix pair BCSST19 consists of the matrices BCSSTK19 and BCSSTM19 from the Matrix Market collection and represents the structural engineering analysis of parts of a suspension bridge. The QZ algorithm detected no infinite eigenvalues.

BCSST19	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	4.52 sec			
KDHGEQZ	3.59 sec			
DHGEQZ+AGG	3.60 sec	180	104	
KDHGEQZ+AGG	3.42 sec	200	48	
MULTIQZ+AGG	3.54 sec	100	24	2



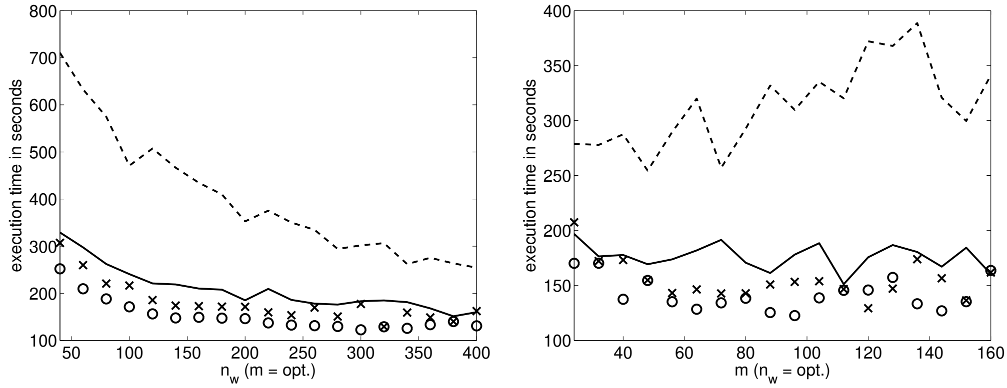
A.8. BCSST20. The 485×485 matrix pair BCSST20 consists of the matrices BCSSTK20 and BCSSTM20 from the Matrix Market collection and represents the structural engineering analysis of a frame within a suspension bridge. The QZ algorithm detected no infinite eigenvalues.

BCSST20	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	1.89 sec			
KDHGEQZ	1.73 sec			
DHGEQZ+AGG	1.60 sec	120	88	
KDHGEQZ+AGG	1.28 sec	100	144	
MULTIQZ+AGG	1.30 sec	60	24	2



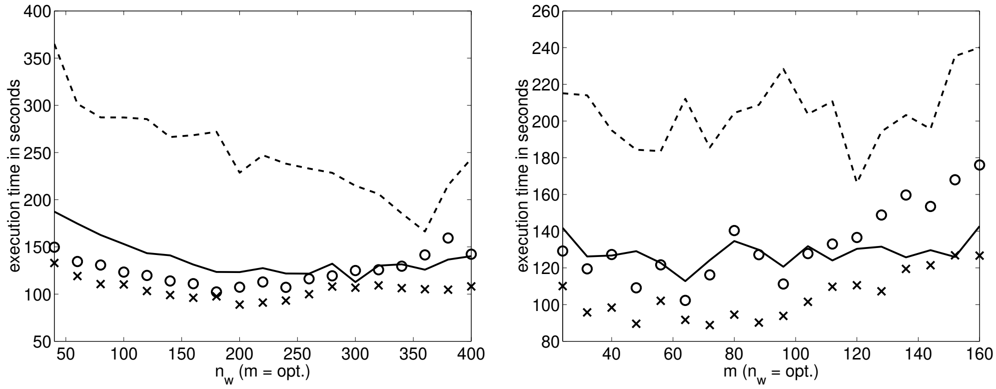
A.9. BCSST21. The 3600×3600 matrix pair BCSST21 consists of the matrices BCSSTK21 and BCSSTM21 from the Matrix Market collection and represents the structural engineering analysis of a clamped square plate. The QZ algorithm detected no infinite eigenvalues.

BCSST21	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	946 sec			
KDHGEQZ	445 sec			
DHGEQZ+AGG	254 sec	400	48	
KDHGEQZ+AGG	151 sec	380	112	
MULTIQZ+AGG	123 sec	300	96	4



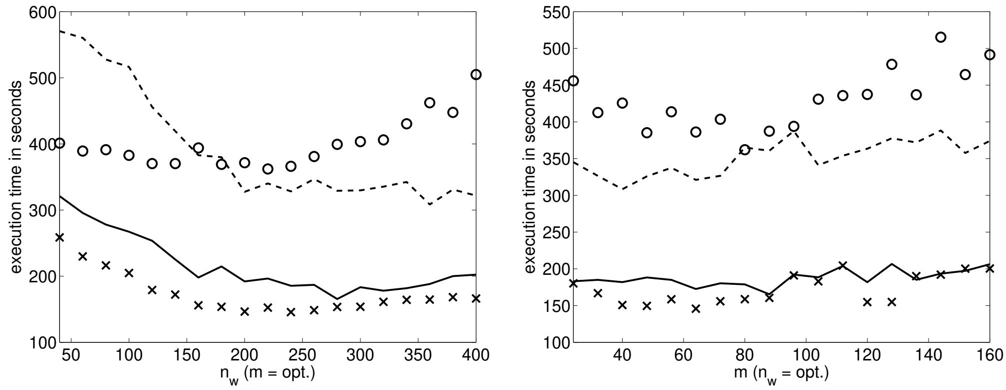
A.10. BCSST23. The 3134×3134 matrix pair BCSST23 consists of the matrices BCSSTK23 and BCSSTM23 from the Matrix Market collection and represents the structural engineering analysis of parts of a 3D globally triangularized building. The QZ algorithm detected no infinite eigenvalues.

BCSST23	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	350 sec			
KDHGEQZ	282 sec			
DHGEQZ+AGG	166 sec	360	120	
KDHGEQZ+AGG	113 sec	300	64	
MULTIQZ+AGG	89 sec	200	72	2



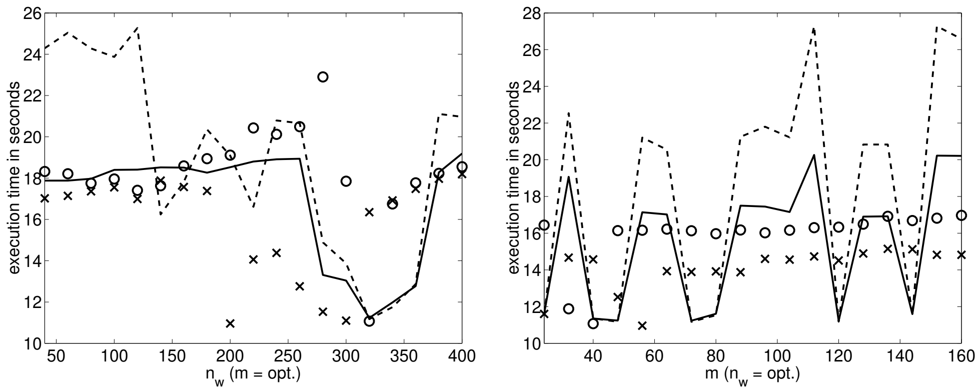
A.11. BCSST24. The 3562×3562 matrix pair BCSST24 consists of the matrices BCSSTK24 and BCSSTM24 from the Matrix Market collection and represents the structural engineering analysis of parts of the Calgary olympic saddledome arena. The QZ algorithm detected no infinite eigenvalues.

BCSST24	opt. time	opt. n_w	opt. m	opt. n_s
`DHGEQZ`	770 sec			
`KDHGEQZ`	393 sec			
`DHGEQZ+AGG`	308 sec	360	40	
`KDHGEQZ+AGG`	165 sec	280	88	
`MULTIQZ+AGG`	146 sec	240	64	2



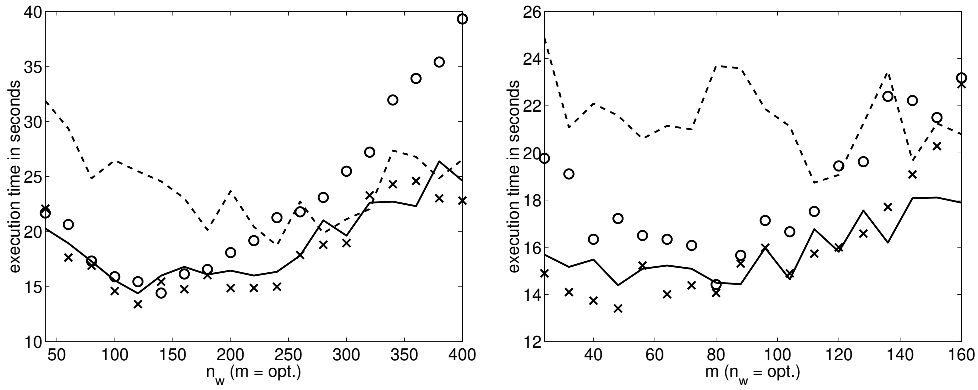
A.12. BCSST26. The 1922×1922 matrix pair BCSST26 consists of the matrices BCSSTK26 and BCSSTM26 from the Matrix Market collection and represents the seismic analysis of a nuclear power station. The QZ algorithm detected no infinite eigenvalues.

BCSST26	opt. time	opt. n_w	opt. m	opt. n_s
`DHGEQZ`	24.54 sec			
`KDHGEQZ`	24.14 sec			
`DHGEQZ+AGG`	11.17 sec	320	48	
`KDHGEQZ+AGG`	11.23 sec	320	120	
`MULTIQZ+AGG`	10.96 sec	200	56	2



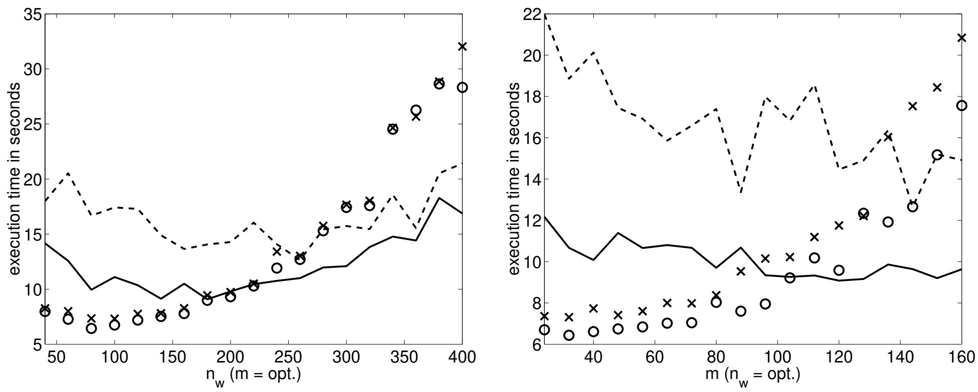
A.13. BCSST27. The 1224 × 1224 matrix pair BCSST27 consists of the matrices BCSSTK27 and BCSSTM27 from the Matrix Market collection and represents a buckling analysis of the engine inlet from a Boeing jetliner. The QZ algorithm detected no infinite eigenvalues.

BCSST27	opt. time	opt. n_w	opt. m	opt. n_s
`DHGEQZ`	38.0 sec			
`KDHGEQZ`	29.9 sec			
`DHGEQZ+AGG`	18.7 sec	240	112	
`KDHGEQZ+AGG`	14.4 sec	120	48	
`MULTIQZ+AGG`	13.4 sec	120	48	2



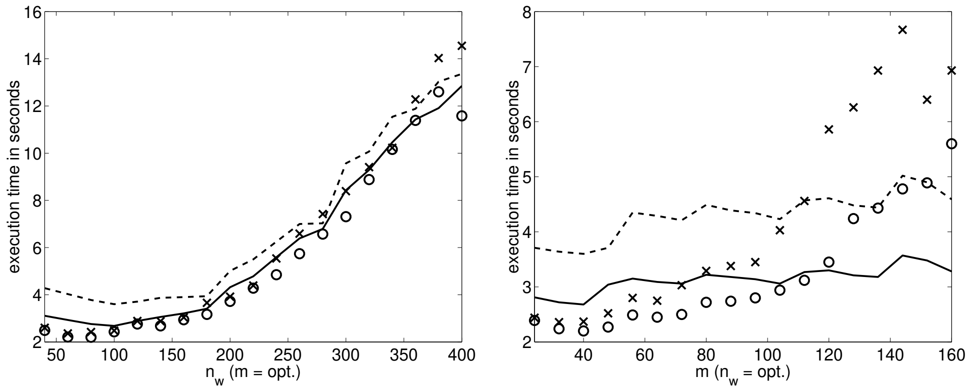
A.14. BFW782. The 782 × 782 matrix pair BFW782 consists of the matrices BFW782A and BFW782B from the Matrix Market collection and arises in the finite element analysis of Maxwell's equation for finding the propagating modes and magnetic field profiles of a rectangular waveguide filled with dielectric and PEC structures. The QZ algorithm detected no infinite eigenvalues.

BFW782	opt. time	opt. n_w	opt. m	opt. n_s
`DHGEQZ`	22.82 sec			
`KDHGEQZ`	16.63 sec			
`DHGEQZ+AGG`	12.65 sec	260	144	
`KDHGEQZ+AGG`	9.08 sec	180	120	
`MULTIQZ+AGG`	6.43 sec	80	32	4



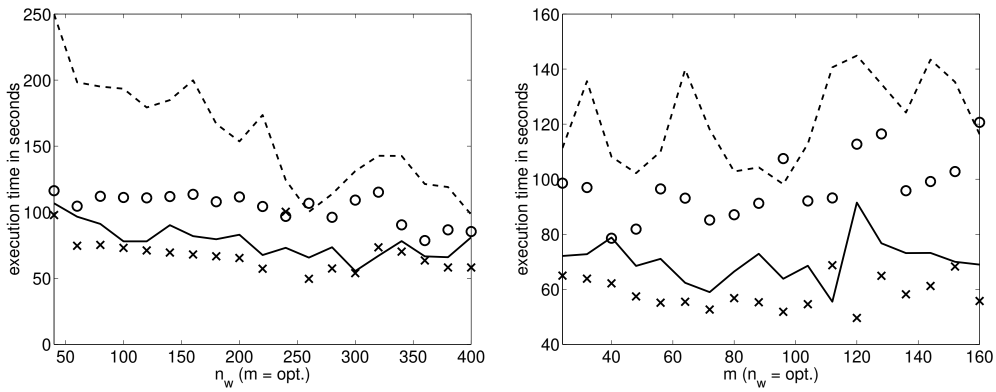
A.15. DWA512. The 512×512 matrix pair DWA512 consists of the matrices DWA512A and DWA512B from the Matrix Market collection and represents a finite difference discretization of the governing Helmholtz equation for a dielectric channel waveguide. The QZ algorithm detected no infinite eigenvalues.

DWA512	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	5.01 sec			
KDHGEQZ	4.25 sec			
DHGEQZ+AGG	3.60 sec	100	40	
KDHGEQZ+AGG	2.68 sec	100	40	
MULTIQZ+AGG	2.20 sec	80	40	4



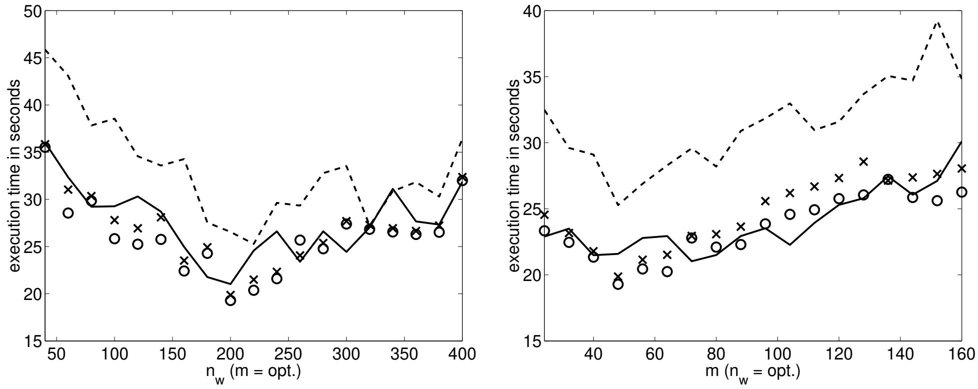
A.16. MHD3200. The 3200×3200 matrix pair MHD3200 consists of the matrices MHD3200A and MHD3200B from the Matrix Market collection and arises in the modal analysis of dissipative magnetohydrodynamics. The QZ algorithm detected no infinite eigenvalues.

MHD3200	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	327 sec			
KDHGEQZ	165 sec			
DHGEQZ+AGG	98 sec	400	96	
KDHGEQZ+AGG	55 sec	300	112	
MULTIQZ+AGG	50 sec	260	120	2



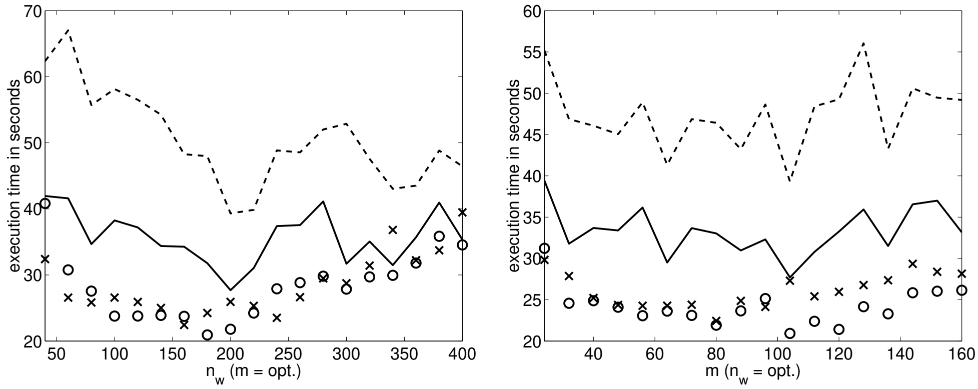
A.17. BEAM. The 1992×1992 matrix pair BEAM is from the Oberwolfach benchmark collection and corresponds to the finite element discretization of a one-dimensional beam with four degrees of freedom per node. The QZ algorithm detected 1 infinite eigenvalue.

BEAM	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	63.3 sec			
KDHGEQZ	47.7 sec			
DHGEQZ+AGG	25.3 sec	220	48	
KDHGEQZ+AGG	21.0 sec	200	72	
MULTIQZ+AGG	19.3 sec	200	48	4



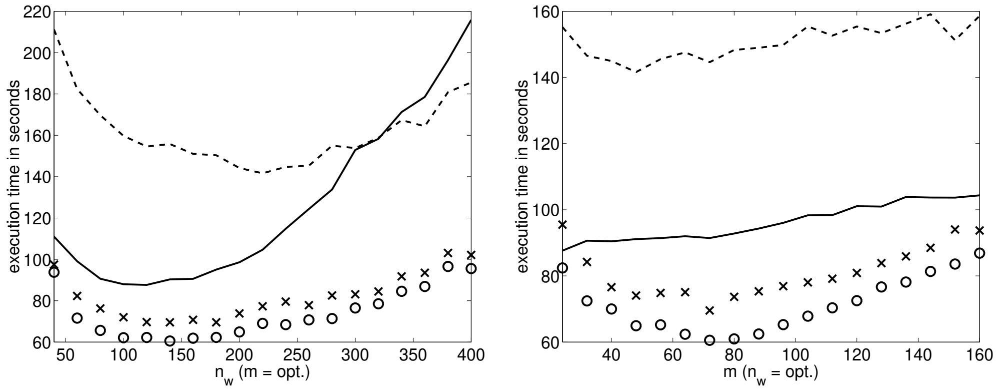
A.18. FILTER. The 1668×1668 matrix pair FILTER is from the Oberwolfach benchmark collection and corresponds to a simplified thermal model of a two-dimensional, tunable optical filter. The QZ algorithm detected no infinite eigenvalues.

FILTER	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	84.9 sec			
KDHGEQZ	76.6 sec			
DHGEQZ+AGG	39.3 sec	200	104	
KDHGEQZ+AGG	27.7 sec	200	104	
MULTIQZ+AGG	20.9 sec	180	104	4



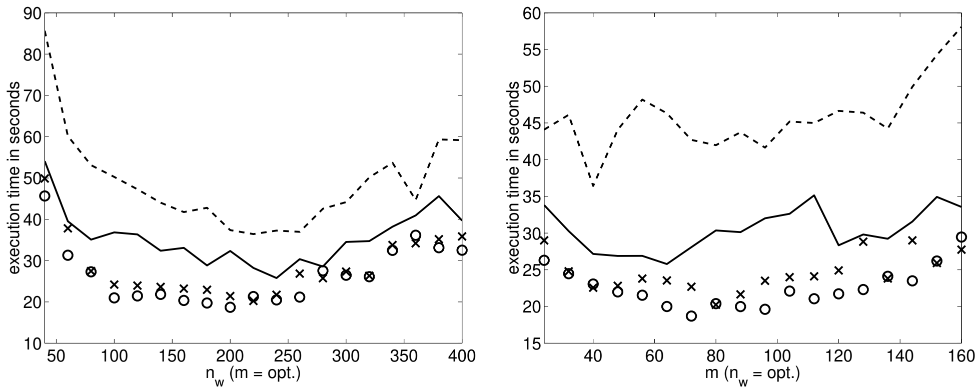
A.19. HEAT. The 1900 × 1900 matrix pair HEAT is from the Oberwolfach benchmark collection and represents the semi-discretized model of heat transfer along a one-dimensional beam. The QZ algorithm detected no infinite eigenvalues.

HEAT	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	338 sec			
KDHGEQZ	199 sec			
DHGEQZ+AGG	141 sec	220	48	
KDHGEQZ+AGG	88 sec	120	24	
MULTIQZ+AGG	61 sec	140	72	4



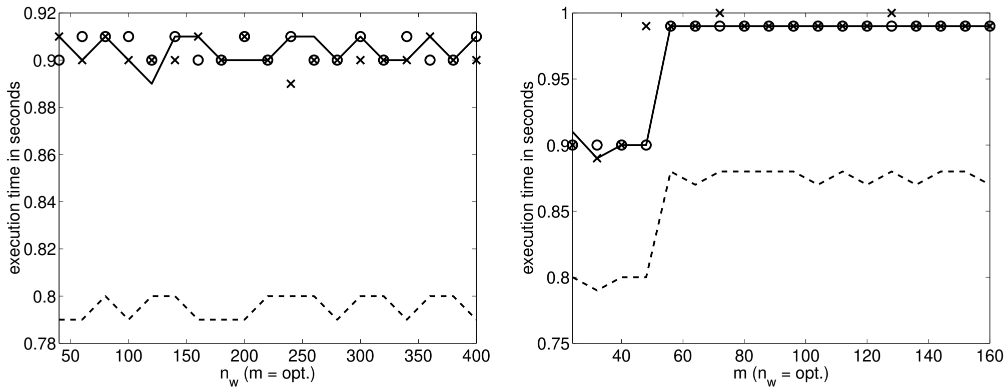
A.20. STEEL. The 1357 × 1357 matrix pair STEEL is from the Oberwolfach benchmark collection and represents the semi-discretized model of heat transfer in a steel profile. The QZ algorithm detected no infinite eigenvalues.

STEEL	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	94.6 sec			
KDHGEQZ	82.9 sec			
DHGEQZ+AGG	36.4 sec	220	40	
KDHGEQZ+AGG	25.8 sec	240	64	
MULTIQZ+AGG	18.7 sec	200	72	4



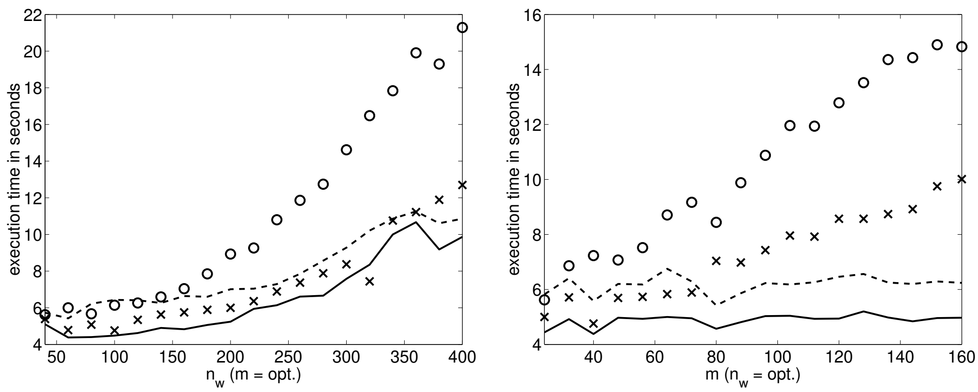
A.21. MNA1. The 578×578 matrix pair MNA1 is from the SLICOT benchmark collection and represents the modified nodal analysis of an electric circuit. The QZ algorithm detected 284 infinite eigenvalues.

MNA1	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	3.12 sec			
KDHGEQZ	2.82 sec			
DHGEQZ+AGG	0.79 sec	40	32	
KDHGEQZ+AGG	0.89 sec	120	32	
MULTIQZ+AGG	0.89 sec	240	32	2



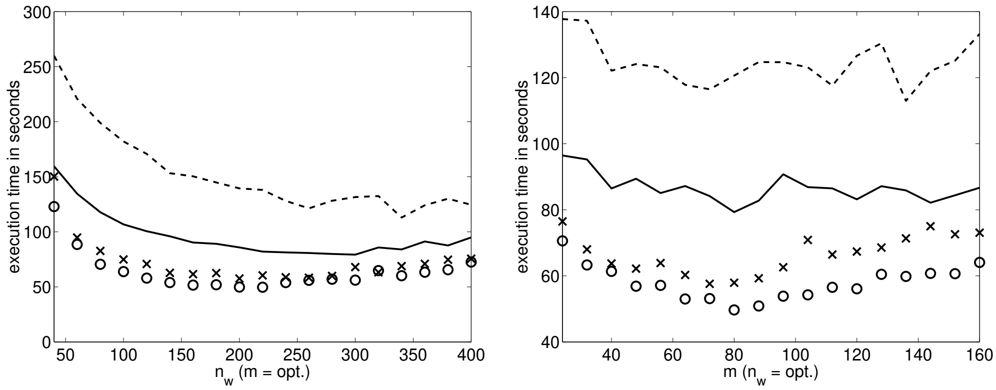
A.22. MNA4. The 980×980 matrix pair MNA4 is from the SLICOT benchmark collection and represents the modified nodal analysis of an electric circuit. The QZ algorithm detected 261 infinite eigenvalues.

MNA4	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	15.8 sec			
KDHGEQZ	13.4 sec			
DHGEQZ+AGG	5.4 sec	60	80	
KDHGEQZ+AGG	4.4 sec	60	40	
MULTIQZ+AGG	4.8 sec	100	40	2



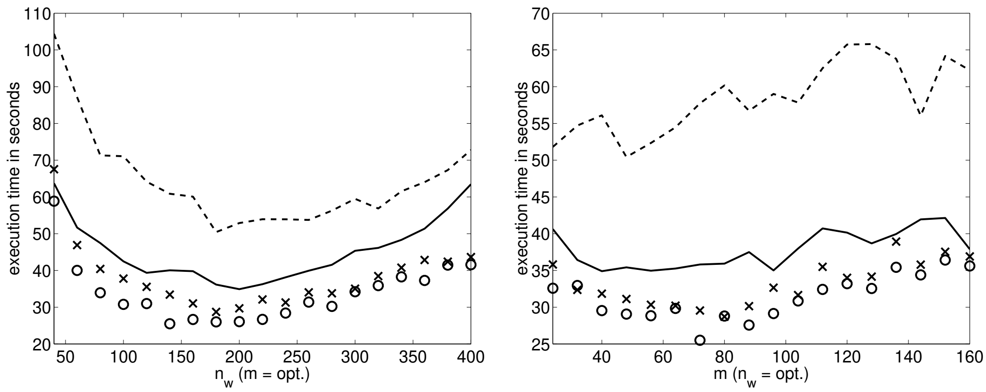
A.23. CIRC90. The 2166 × 2166 matrix pair CIRC90 corresponds to a discretized Laplace equation with Dirichlet boundary condition on a circular cone with an opening angle of 90 degrees, see [36] for more details. The QZ algorithm detected no infinite eigenvalues.

CIRC90	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	340 sec			
KDHGEQZ	210 sec			
DHGEQZ+AGG	113 sec	340	136	
KDHGEQZ+AGG	79 sec	300	80	
MULTIQZ+AGG	50 sec	220	80	4



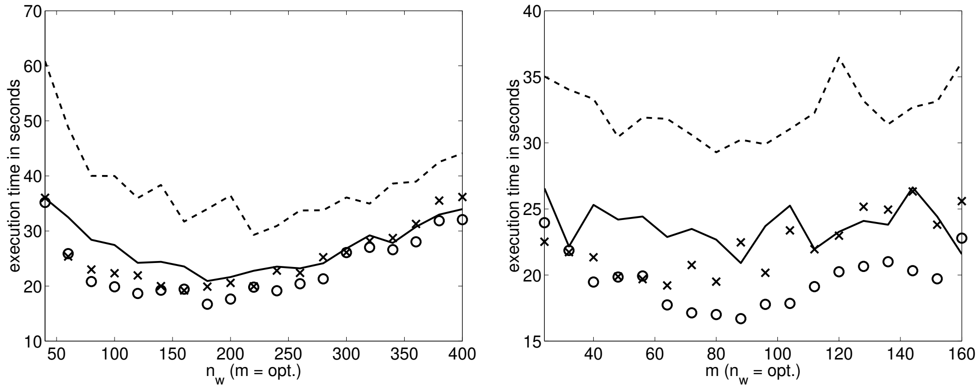
A.24. CIRC120. The 1626 × 1626 matrix pair CIRC120 corresponds to the Laplace equation with Dirichlet boundary condition on a circular cone with an opening angle of 120 degrees. The QZ algorithm detected no infinite eigenvalues.

CIRC120	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	141 sec			
KDHGEQZ	102 sec			
DHGEQZ+AGG	50 sec	180	48	
KDHGEQZ+AGG	35 sec	200	40	
MULTIQZ+AGG	25 sec	140	72	4



A.25. FICH669. The 1338×1338 matrix pair FICH669 corresponds to the Laplace equation with Dirichlet boundary condition on the Fichera domain. The QZ algorithm detected no infinite eigenvalues.

FICH669	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	74.4 sec			
KDHGEQZ	50.5 sec			
DHGEQZ+AGG	29.3 sec	220	80	
KDHGEQZ+AGG	20.1 sec	180	88	
MULTIQZ+AGG	16.7 sec	180	88	4



A.26. FICH1227. The 2454×2454 matrix pair FICH1227 corresponds to the same setting as FICH669 but with a finer discretization. The QZ algorithm detected no infinite eigenvalues.

FICH1227	opt. time	opt. n_w	opt. m	opt. n_s
DHGEQZ	469 sec			
KDHGEQZ	268 sec			
DHGEQZ+AGG	144 sec	320	96	
KDHGEQZ+AGG	102 sec	320	96	
MULTIQZ+AGG	65 sec	200	80	4