Electronic Transactions on Numerical Analysis.
Volume 40, pp. 17-35, 2013.
Copyright © 2013, Kent State University.
ISSN 1068-9613.

AN EXTENSION OF THE QZ ALGORITHM BEYOND THE HESSENBERG-UPPER TRIANGULAR PENCIL[*]

RAF VANDEBRIL[†] AND DAVID S. WATKINS[‡]

Abstract. Recently, an extension of the class of matrices admitting a Francis type of multishift QR algorithm was proposed by the authors. These so-called condensed matrices admit a storage cost identical to that of the Hessenberg matrix and share all of the properties essential for the development of an effective implicit QR type method. This article continues along this trajectory by discussing the generalized eigenvalue problem. The novelty does not lie in the almost trivial extension of replacing the Hessenberg matrix in the pencil by a condensed matrix, but in the fact that both pencil matrices can be partially of condensed form. Again, the storage cost and crucial features of the Hessenberg-upper triangular pencil are retained, giving rise to an equally viable QZ-like method. The associated implicit algorithm also relies on bulge chasing and exhibits a sort of bulge hopping from one to the other matrix. This article presents the reduction to a condensed pencil form and an extension of the QZ algorithm. Relationships between these new ideas and some known algorithms are also discussed.

Key words. condensed matrices, generalized eigenvalues, QZ algorithm, QR algorithm, extended Krylov

AMS subject classifications. 65F15, 15A18

1. Introduction.

The QZ algorithm is one of the most popular methods for computing (generalized) eigenvalues of pencils (A, B). It is well known that the QZ algorithm originates from Francis's implicitly shifted QR algorithm [9, 10]. To achieve a computationally economical QR or QZ algorithm, the matrix or pencil is first transformed to a condensed form, usually a Hessenberg matrix or a Hessenberg-upper triangular pencil. In [16] a whole family of condensed matrices admitting low cost QR steps was proposed. In [18] a convergence theory was provided and it was shown that the type of condensed form affects the convergence speed.

In this article, we continue this research by studying the generalized eigenvalue problem. We will not elaborate on the almost trivial extension of considering a pencil composed of a condensed and an upper triangular matrix. Instead, we will consider pencils (A, B) in which both matrices are partially of condensed form. Both A and B are stored in factored form $A = G_A R_A$ and $B = G_B R_B$, where $G_A G_B = C_{i_1} \cdots C_{i_{n-1}}$, with $\{i_1, \ldots, i_{n-1}\}$ a permutation of $\{1, \ldots, n-1\}$ and each C_k is a *core* transformation, acting on two consecutive rows k and $k+1$. In total there are thus $n-1$ core transformations $C_1, \ldots, C_{n-1}$ distributed between A and B. The matrices R_A and R_B are upper triangular and for simplicity assumed to be nonsingular.

We will show that it is possible to achieve any condensed pencil form by a finite number of equivalence transformations. Moreover, under the mild condition of unitarity of the transforming matrices, uniqueness is guaranteed. A chasing procedure to execute an implicit QZ step on such a condensed pencil is proposed and a convergence is studied. To conclude, a discussion relating this new algorithm to existing algorithms is included. It will be shown, e.g., that the Schur-parameter pencil approach of Bunse-Gerstner and Elsner is an instance of this general framework.

[*] Received June 12, 2012. Accepted November 26, 2012. Published online on March 8, 2013. Recommended by W. B. Gragg.
The research was partially supported by the Research Council KU Leuven: OT/11/055 Spectral Properties of Perturbed Normal Matrices and their Applications and by the Fund for Scientific Research–Flanders (Belgium): G034212N Reestablishing smoothness for matrix manifold optimization via the resolution of singularities.

[†] Department of Computer Science, KU Leuven, 3001 Leuven (Heverlee), Belgium (`raf.vandebril@cs.kuleuven.be`).

[‡] Department of Mathematics, Washington State University, Pullman, WA 99164-3113, USA (`watkins@math.wsu.edu`).

The paper is organized as follows. Section 2 discusses preliminaries and the condensed pencil formats under consideration. In Section 3 an explicit equivalence to this condensed format is proposed. Section 4 presents a novel implicit GZ algorithm. In Section 5 the link with extended Krylov methods is studied, and based thereon, the uniqueness of the reduction and a proof of convergence are presented. Sections 6 and 7 present two appealing instances of the general framework.

Concerning notation, we have the following conventions. Matrices are typeset in an uppercase font: A, Q; vectors appear as bold-face, lowercase letters: $\mathbf{p}$, $\mathbf{v}$; scalars are depicted as lowercase letters, e.g., the elements of matrices and vectors: $A = [a_{ij}]_{ij}$ and $\mathbf{p} = [p_j]_j$; uppercase calligraphic letters stand for subspaces: $\mathcal{K}$, $\mathcal{E}$.

2. Condensed matrix pencils and handling core transformations.

Consider two possibly complex $n \times n$ matrices A and B, whose generalized eigenvalues we wish to compute. Assume that no zero or infinite eigenvalues exists, thus A and B are nonsingular.

2.1. A detailed factorization by core transformations.

The pencil (A, B) will be stored in compact format as a product of (possibly nonunitary) core transformations and an upper triangular matrix. A *core transformation* C_i is the embedding of a nonsingular 2×2 matrix at the intersection of the ith and $(i + 1)$st rows and columns in the identity matrix. The inverse of a core transformation is again a core transformation. Left multiplying a matrix with a core transformation C_i only alters two consecutive rows; C_i is said to *act* on rows i and $i + 1$. Unless stated otherwise, the subscript i in C_i points to the rows the core transformation acts on. A *detailed factorization* of a matrix is a GR factorization with R upper triangular and G decomposed entirely as a product of core transforms.

Consider a matrix pencil (A, B) which has detailed GR decompositions $A = G_A R_A$ and $B = G_B R_B$. The pencil is in *condensed* form if the total number of core transformations in the factorizations of G_A and G_B is $n - 1$, and the set of core transformations includes exactly one C_i acting on rows i and $i + 1$ for $i = 1, \ldots, n - 1$.

We call a core transformation C_i *nontrivial* if it is not upper triangular. Observe that if any of the core transforms is trivial, the associated generalized eigenvalue problem can be split into two smaller eigenvalue problems. A matrix pencil in condensed form without trivial core transformations will be called *irreducible*. To avoid a discussion of degenerate cases, we assume from this point on that the pencil is irreducible and the pencil matrices are nonsingular.

2.2. Examples.

The sparseness pattern of a core transformation reveals that C_i and C_j commute whenever i and j differ by more than one. So besides the fact that a core transform is assigned to either A or B, the mutual relative position of two successive core transforms plays a big role. A fine and coarse grained graphical manner to keep track of the position of each of the individual rotations is therefore presented.

The Hessenberg-upper triangular matrix pencil is in condensed format, as the Hessenberg matrix admits a GR decomposition with G factored as $G = C_1 C_2 \cdots C_{n-1}$. For unitary G and C_i's, the factorization of G coincides with the Schur parameterization [13]. The detailed graphical factorization of G is presented in Figure 2.1(a): each bracket represents a core transformation with arrows pointing to the rows affected by the transformation. For clarity, the upper triangular part will often be omitted. The factorization of G manifests a *descending* sequence of core transformations. The corresponding coarse grained graphical depiction is shown in Figure 2.1(b): the dots stand for core transformations which in turn are connected by a line to stress the ordering. Again, typically, we will omit the upper triangular matrix, and when the position of the dots is clear from the context, they might be omitted as well. For a pencil (A, B) with B upper triangular, A can be, for instance, of inverse Hessenberg form

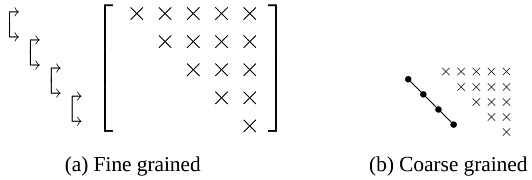

(a) Fine grained (b) Coarse grained

Fig. 2.1: Graphical depictions of a detailed factorization of a Hessenberg matrix.

shown in Figure 2.2(a) or CMV form as in Figure 2.2(b). Both A and B can be of reducible Hessenberg form with a non-conformable pattern; see, e.g., Figure 2.2(c), where the rotations tied to A are represented by black disks, the ones bound to B are circles. Linking the even core transforms to A and the odd ones to B also results in an admissible condensed pencil shown in Figure 2.2(d). Many of these generalizations appeared in other contexts. In particular, Fiedler factorizations, which provide new various decompositions of the companion pencil for retrieving roots of polynomials, fit into this framework [7, 8]. For unitary matrices, we refer to the overview article on CMV matrices [15] and [19], the paper by Kimura [14], and the generalizations in [5]. The CMV and unitary matrices in general are linked to orthogonal polynomials on the unit circle for which a rich variety of literature is available; early contributions and references can be found in [1–3]. For instance, the iterative eigenvalue algorithm proposed in [6] operates on the alternating factorization, shown in Figure 2.2(d); see Section 7.

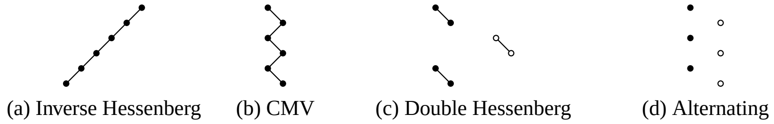

(a) Inverse Hessenberg (b) CMV (c) Double Hessenberg (d) Alternating

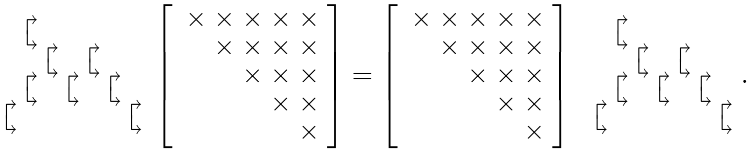
Fig. 2.2: Condensed formats for 7 × 7 matrices, omitting the upper triangular matrix.

2.3. The position vector. The *position vector* stores the mutual ordering of the core transformations. This vector $\mathbf{p}$ of length $n-2$ contains elements ℓ, r, and s, where ℓ or r indicates that the core transformation C_i is positioned to the left or right of the next core transformation C_{i+1} and s stands for a matrix swap, i.e., the next core transform C_{i+1} belongs to B (or A) for C_i belonging to A (or B). Here we tacitly assume that C_1 goes with A; exchanging variable names and mapping the eigenvalues to their reciprocals demonstrates that there is no loss of generality in this assumption. In a few cases, however, we will explicitly mention C_1 tied to B.

Reconsider the examples from Subsection 2.2. The matrix pencil associated with Figure 2.1 admits a factorization $G_A = C_1C_2\cdots C_{n-1}$, $G_B = I$ and has an associated position vector $\mathbf{p} = [\ell, \ell, \ldots, \ell]$. For Figure 2.2(a), $G_A = C_{n-1}C_{n-2}\cdots C_1$, $G_B = I$ and $\mathbf{p} = [r, r, \ldots, r]$; the core transformations are ordered in an *ascending* sequence. The CMV-shaped pencil partially depicted in Figure 2.2(b) corresponds to $\mathbf{p} = [\ell, r, \ell, r, \ell]$, $G_A = C_1C_3C_5 \cdot C_2C_4C_6$, and $G_B = I$. The double Hessenberg pencil relates to a position vector $\mathbf{p} = [\ell, s, \ell, s, \ell]$ and two factorizations $G_A = C_1C_2C_5C_6$ and $G_B = C_3C_4$. The alternate positioning of the core transformations as in Figure 2.2(d) corresponds to the vector $\mathbf{p} = [s, s, s, s, s]$.

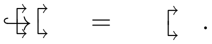
2.4. Juggling core transforms. The algorithms in this article consist entirely of systematic modifications and repositionings of core transformations. We will utilize three types of operations, which we call *passing through*, *turnover*, and *fusion*.

In the *passing through* operation, core transformations are "passed through" upper triangular matrices. Consider, e.g., the product C_iR. The resulting matrix is upper triangular, except for a bulge in position $(i+1, i)$. The bulge can be removed by applying a core transformation on the right involving columns i and $i+1$, resulting in a new factorization $\tilde{R}\tilde{C}_i$, with $\tilde{R}$ again upper triangular. We have $C_iR = \tilde{R}\tilde{C}_i$, so the core transformation has been passed from the left to the right of the upper triangular matrix. Similarly one can pass core transformations from right to left. Given a long sequence of core transformations in a particular *pattern* to the left of an upper triangular matrix, we can pass the core transformations through one by one so that the same pattern of core transformations emerges on the right-hand side, e.g.,

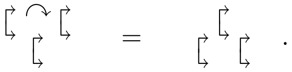


The triangular matrices are not equal, and neither are the core transformations on the left equal to those on the right. What is preserved is the *pattern* of the core transformations. The possibility of transferring core transforms from one side to the other side of an upper triangular matrix without altering the mutual ordering allows us to suppress the triangular matrix in forthcoming descriptions.

Multiplying two core transformations acting on the same two rows results in a new core transformation. This operation is called *fusion* and is depicted graphically as



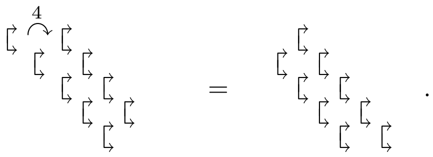
The final operation we need to describe is the *turnover* (or *shift-through*) operation, depicted by



A factorization of three core transformations, one acting on rows i and $i+1$ sandwiched between two others acting on rows $i-1$ and i, is reshuffled into a different product of three core transformations, one acting on rows $i-1$ and i sandwiched between two transforms acting on rows i and $i+1$. This operation is always possible if the core transformations are unitary [17]. In the non-unitary case, the operation is almost always possible but can fail in exceptional cases. The procedure is described in [4].

A chain of turnovers of length ℓ, is a simple succession of ℓ successive turnover operations. Graphically we depict 4 consecutive turnovers simultaneously as follows

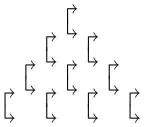
(2.1)



One can interpret this as moving the core transformation on the far left-hand side in (2.1) to the far right-hand side in (2.1).

3. Conversion to condensed pencil format. Starting from any pencil (A, B), we will present an algorithm for computing nonsingular matrices U, V such that $(U^H AV, U^H BV)$ with $G_A R_A = U^H AV$ and $G_B R_B = U^H BV$ satisfies a prespecified position vector.

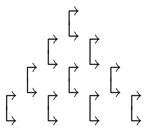
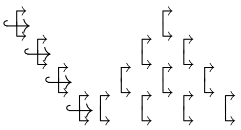
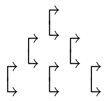
We presume both A and B of non-structured form, admitting a detailed GR factorization in *pyramid* shape [17]. This means that, for example, for both A and B of size 5×5, the G factor admits a decomposition into core transforms of the form

$G =$  .

This shape naturally emerges when computing the GR factorization by eliminating elements columnwise by rotations or elementary Gauss-transforms acting only on successive rows.

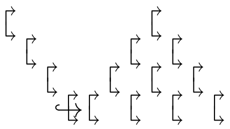
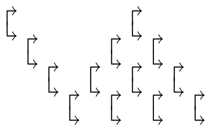
Before presenting and demonstrating the algorithm, we elaborate on two requisite multiplicative operations and their effect on the detailed pyramid shaped factorization.

3.1. Removal of core transforms and updating the pyramid. *Removing* right (left) outer core transforms from a detailed factorization in pyramid form is done by a right (left) multiplication with the inverses of the core transformations designated for removal. Graphically, this is depicted as

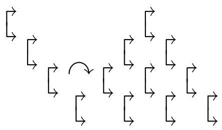
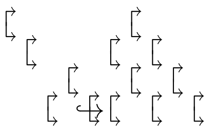
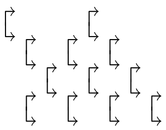
$G =$  $\Rightarrow$ $UG =$  $=$  ,

where a multiplication on the left with U annihilates four outer-left core transforms by executing four individual fusions.

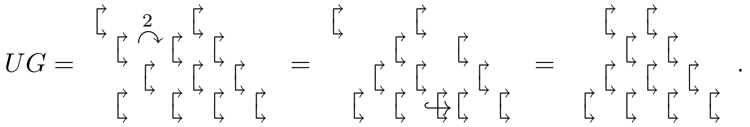
Updating a detailed factorization in pyramid shape after a left or right multiplication by core transformations differs from the *removal* operation in the sense that basically no operation will be annihilated. It is possible, however, to reduce the total number of core transformations by incorporating the multiplication factors in the pyramid pattern. Graphically, we start identically to the removal operation by a multiplication on the left (or right)

$UG =$  $=$  .

Executing the bottom fusion reduces the overall number of transforms already by one. The turnover operation depicted on the left of (3.1) extracts one of the core transforms of the outer-left descending sequence and deposits it inside the pyramid shape, ready to be fused. We have eliminated two transforms already.

(3.1) $UG =$  $=$  $=$  .

To get rid of another transformation, a chain of two turnover operations is carried out, followed by a fusion.



A sole core transformation dangling on the upper-left side of the pyramid remains. Three turnovers and a fusion suffice to incorporate this transform into the pyramid pattern. In a similar manner, one can dispose core transforms performed on the right of the pyramid shape.

3.2. The conversion algorithm.

ALGORITHM 3.1. *Let the detailed GR decompositions of A and B be given as well as a prespecified position vector $\mathbf{p}$. Set $X = A$ and $Z = B$.*
Execute steps $i = 1, \dots, n-2$.

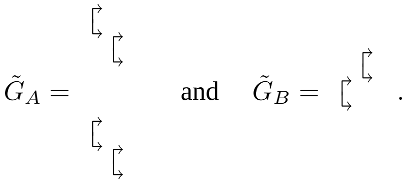
1. *Remove $n-i$ core transformations from the detailed factorization of Z. For $i = 1$, choose to remove the outer left or right ones. For $i > 1$: if $p_{i-1} = \ell$, remove from the right side; if $p_{i-1} = r$, take the left side; if $p_{i-1} = s$, take left/right, opposite to the choice made in step 3 in the previous passage of this loop.*
2. *Apply the multiplication on X; update its detailed factorization.*
3. *Remove $n-i-1$ core transformations from the detailed factorization of X. If $p_i = \ell$, remove from the left side; if $p_i = r$, take right; if $p_i = s$, choose right/left.*
4. *Apply the multiplication on Z; update its detailed factorization.*
5. *If $p_i = s$, interchange X and Z.*

The final step completes the transition to a condensed pencil.

1. *Remove one core transformation from the detailed factorization of Z. If $p_{n-2} = \ell$, remove from right side; if $p_{n-2} = r$, take left side; if $p_{n-2} = s$, take left/right, opposite to the choice made in step 3 in run $n-2$ of the loop.*
2. *Apply the multiplication on X; update its detailed factorization.*

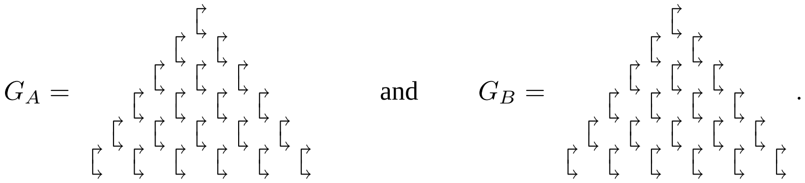
We remark that if G_1 were bound to B, then we initially would have required that $X = B$ and $Z = A$.

3.3. Example. Let $A, B \in \mathbb{C}^{7\times 7}$, and take $\mathbf{p} = [\ell, s, r, s, \ell]$. The reduction algorithm results in the following G factors belonging to the detailed GR factorizations of $U^H(A,B)V$



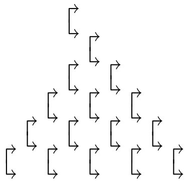
For the ease of exposition, the upper triangular matrices are suppressed.

We begin with two detailed factorizations of the matrices A and B, having G_A and G_B in pyramid shape.



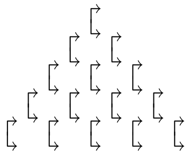
Step $i = 1$ *($p_1 = \ell$).* Remove 6 core transformations from the right—in this initial step one has the freedom to choose also left—of the matrix G_B by a right multiplication with V_1. In this description, it is assumed that the detailed representation of the counter-matrix is updated straight away. Next, remove 5 core transformations from the left of the factorization bound to A by a left multiplication with U_1^H. It is easily verified by techniques offered in Section 3.1 that the updating process does not recreate previously removed core transformations. The result is depicted as

$U_1^H G_A V_1 =$



and

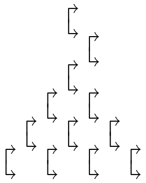
$U_1^H G_B V_1 =$



.

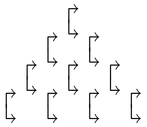
Step $i = 2$ *($p_2 = s$)*, remove 5 core transformations from the right, as $p_1 = \ell$, of the matrix associated to B by V_2. Removing them from the left is not feasible: one cannot update the factorization of the matrix A as the top rotation blocks everything. Next remove 4 transformations from the right (or left) of the matrix linked to A by the transformation V_3. We get

$U_1^H G_A V_1 V_2 V_3 =$



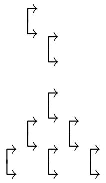
and

$U_1^H G_B V_1 V_2 V_3 =$

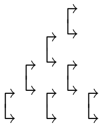


.

Step $i = 3$ *($p_3 = r$).* We switched the matrices X and Z. We dispose 4 core transformations from the matrix stemming from A by multiplying on the left (or right, depending on what was taken in step $i = 2$) with U_2^H. This is followed by annihilating 3 transforms from the right of the matrix tied to B by multiplication with V_4. Graphically, we have the following detailed factorizations for $U_2^H U_1^H (G_A, G_B) V_1 V_2 V_3 V_4$

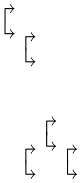


and

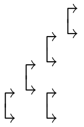


.

Step $i = 4$ *($p_4 = s$).* We get rid of 3 transforms from the matrix bound to A by a left multiplication; remove 2 transforms from the matrix related to B by operating on the right (or left). The final shape becomes visible in $U_3^H U_2^H U_1^H (G_A, G_B) V_1 V_2 V_3 V_4 V_5$



and

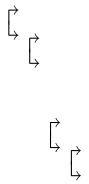


.

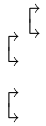
In *step* $i = 5$ *(*$p_5 = \ell$*)*, 2 transforms are removed from the left (or right, check step $i = 4$) of the matrix stemming from B. A single core transformation is removed by a left multiplication applied to A. The factorization of

$$U_5^H U_4^H U_3^H U_2^H U_1^H (G_A, G_B) V_1 V_2 V_3 V_4 V_5$$

corresponds to

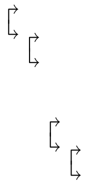
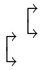


and



and displays only one more transformation designated for removal.

Final step. Clear the transformation from the bottom of the matrix stemming from B by a right multiplication. We obtain the desired condensed pencil structure

$U^H G_A V =$  and $U^H G_B V =$  .

3.4. Conversion to Hessenberg - upper triangular pencil. If Algorithm 3.1 is applied with $\mathbf{p} = [\ell, \ell, \ell, \ldots, \ell]$, the pencil is reduced to Hessenberg-triangular form. The order of reduction is different from that of the traditional algorithm that is presented in most textbooks. The latter begins by reducing B to triangular form, then reduces A to Hessenberg form while defending the upper triangular form of B.

In [20, § 6.2] two reduction algorithms are presented. The first is the traditional algorithm. The second begins by creating zeros in the first column of B below the main diagonal. Then it creates zeros in the first column of A below the subdiagonal. Then it creates zeros in the second column of B, then the second column of A, and so on. The algorithm can be summarized as follows.

ALGORITHM 3.2. *Execute steps* $i = 1, \ldots, n-2$.

1. *Determine an elimination matrix* V *such that* BV *has zeros in column* i *below the diagonal. Multiplication by* V *only affects the trailing* $n-i-1$ *columns of* B *and* A.
2. *Set* $B = BV$ *and* $A = AV$.
3. *Determine an elimination matrix* U *such that* $U^H A$ *has zeros in column* i *below the subdiagonal. Multiplication by* U^H *only affects the trailing* $n - i$ *rows.*
4. *Set* $A = U^H A$ *and* $B = U^H B$.

Finally remove by right elimination the bottom element of the $(n-1)$*st column of* B.

Algorithm 3.1 is just like this. It begins by removing core transformations from the detailed factorization of B in such a way that the new B has zeros in its first column below the main diagonal. Then it removes core transformations from A with the effect that the first column of A has zeros below the subdiagonal. The next removal of core transformations creates zeros in the second column of B, and so on. Thus, the two algorithms are essentially the same except that one operates on the matrices directly while the other operates on the detailed factorization.

3.5. Conversion to a mirrored Hessenberg pair.

If Algorithm 3.1 is applied with a position vector that contains only the symbols ℓ and s, the result is a pair in which both matrices are upper Hessenberg. In fact each of them is partly Hessenberg and partly triangular in character, as shown by the example in the final picture of Figure 3.1. If we partition each of the matrices so that its main diagonal consists of alternating Hessenberg and triangular blocks, we find that the Hessenberg blocks of B begin where the Hessenberg blocks of A end, and vice versa. Therefore we call this a *mirrored Hessenberg pair*.

In the case of reductions to a mirrored Hessenberg pair, it is possible to reformulate the conversion algorithm so that it acts directly on the matrices.

ALGORITHM 3.3. *Let (A, B) represent an $n \times n$ pencil. The following steps execute a transition to a mirrored Hessenberg pair. Set $X = A$, $Z = B$.*

```
Execute steps $i = 1, \dots, n-2$.
    1. Determine an elimination matrix $V$ such that $ZV$ has zeros in column $i$ below the
       diagonal. Multiplication by $V$ only affects the trailing $n-i-1$ columns.
    2. Set $Z = ZV$ and $X = XV$.
    3. Determine an elimination matrix $U$ such that $U^H X$ has zeros in column $i$ below the
       subdiagonal. Multiplication by $U^H$ only affects the trailing $n-i$ rows.
    4. Set $X = U^H X$ and $Z = U^H Z$.
    5. If $p_i = s$, interchange $X$ and $Z$.
Finally remove by right elimination the bottom element of the $(n-1)$st column of $Z$.
```

The intermediate structures when running this algorithm on two 5×5 matrices are depicted in Figure 3.1. The outcome corresponds to a position vector $\mathbf{p} = [s, \ell, s]$.

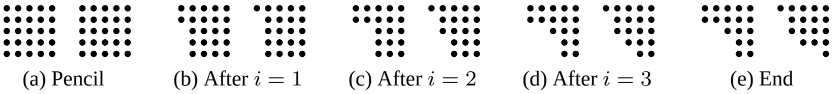

(a) Pencil (b) After $i = 1$ (c) After $i = 2$ (d) After $i = 3$ (e) End

Fig. 3.1: Sparsity pattern invoked by Algorithm 3.3.

4. Generalization of the GZ algorithm.

The algorithm presented below executes a single shifted generalized GZ step on a condensed matrix pencil. We can also do double generalized GZ steps or indeed steps of arbitrary degree, but the description becomes quite complicated. For this reason, we are restricting our discussion to single steps. The designation GZ means that we are allowing non-unitary transformation matrices. When we want to confine our attention to the unitary case, we will use QZ instead of GZ.

A core transformation is said to be *left or right free* if it can be relocated (commuted with other core transforms) in the detailed factorization to become the outer left or right core transformation. Being *free* implies thus that this transform can be canceled easily by a left or right multiplication.

A single shifted GZ step will push up the shape of the core transformations by one position. Strictly speaking, the first element of the position vector drops off, all other elements move to the left leaving an open spot at position $n-2$. One can freely choose which element to place in this final position: ℓ, r, or s. Though one usually does not reflect on this, this behavior takes place in all QR and GR like algorithms [16, 18]. For example, in the Hessenberg case one always takes ℓ to fill up the free spot to retain the Hessenberg structure throughout the successive iterates.

4.1. Implicit single shifted extension of the GZ algorithm.

Let the pencil (A,B) be condensed and irreducible. The next steps perform a generalized implicitly shifted GR step.

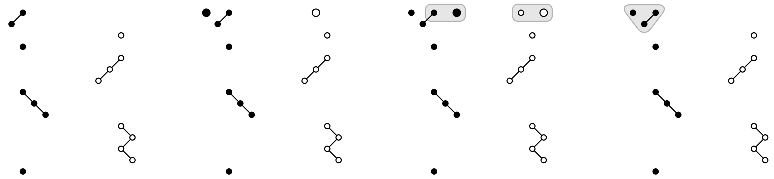
- Step 0. *Build and apply an initial perturbing core transformation and prepare the pencil for a turnover operation in the matrix possessing transform C_2.* More precisely:
  - Pick a shift ρ. Let $\mathbf{x} = (A - \rho B)\mathbf{e}_1$ if $p_1 = \ell$ or s, and $\mathbf{x} = (B^{-1} - \rho A^{-1})\mathbf{e}_1$ if $p_1 = r$, with $\mathbf{e}_j$ the jth canonical basis vector. Either way $\mathbf{x}$ has only its first two entries nonzero and depends only upon C_1 and the top entries of R_A and R_B. Let $\tilde{C}_1$ be a core transformation such that $\tilde{C}_1\mathbf{x} = \alpha\mathbf{e}_1$ for some α.
  - Left multiply A and B by the initial perturbing core transformation $\tilde{C}_1$.
  - If not blocked by C_2, fuse $\tilde{C}_1$ with the top transform C_1.
  - Remove the right free transform acting on rows 1 and 2 from the matrix *not* containing C_2 by a right multiplication.
  - If C_1 was not yet fused, fuse it with the transform along its right.
- Steps $i = 1, \dots, n-3$. *Execute the turnover and prepare by well-chosen left and right multiplications the pencil for the next turnover taking place in the matrix possessing transformation C_{i+2}.* More specifically:
  - Execute the turnover operation in the matrix containing transformation C_{i+1}. This involves two transforms acting on rows i and $i+1$, and the in-between located transform C_{i+1} operating on rows $i+1$ and $i+2$.
  - The matrix just affected has at least one free core transform acting on rows $i+1$ and $i+2$. Execute a left/right multiplication to remove it.
  - Remove the right/left (the opposite side as above) free transform acting on rows $i+1$ and $i+2$ from the matrix *not* containing C_{i+2}.
- Step $n-2$. *Select l, r, or s to position the new trailing transform.*
  - Execute the turnover in the matrix containing C_{n-1}.
  - The matrix just affected has now two free transforms, say C_ℓ and C_r, for respectively the left and right free one.
  - Choose ℓ, r, or s to append to the position vector.
  - If choice $= \ell$: remove C_r by a right multiplication; remove the newly created core transform from the other matrix by a left multiplication; fuse the new transform with C_ℓ.
  - If choice $= r$: remove C_ℓ by a left multiplication; remove the newly created core transform from the other matrix by a right multiplication; fuse the new transform with C_r.
  - If choice $= s$: remove C_ℓ by a left multiplication; remove C_r by a right multiplication; fuse both new transforms into the other matrix.

4.2. Example. We will elucidate the implicit GZ algorithm by an example. Let A and B both be of dimension 16×16 and $\mathbf{p} = [r, s, s, s, r, r, s, \ell, \ell, s, \ell, r, \ell, s]$.

Left or right multiplication introduces extra transforms in the schemes, for clarity initially rendered slightly bigger than the other bullets. Transformations bound to be removed, fused, or subjected to a turnover operation are outlined in gray.

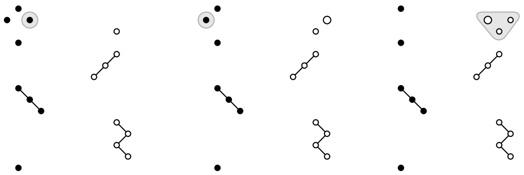
Figure 4.1 illustrates Step 0. As $p_1 = r$, the newly added core transformations from a left multiplication (Figure 4.1(b)) cannot be fused immediately. First a right multiplication is performed (Figure 4.1(c)) to dispose of one transformation and enabling a fusion resulting in Figure 4.1(d).

In the main loop of the algorithm, we first execute, for $i = 1$, the turnover operation indicated in Figure 4.1(d) leading to Figure 4.2(a). Both transformations bound to A acting on rows 2 and 3 are free. The right one is marked and removed first in Figure 4.2(b), next the left one is removed (Figure 4.2(c)). We end up with three transformations in turnover-format in Figure 4.2(b).



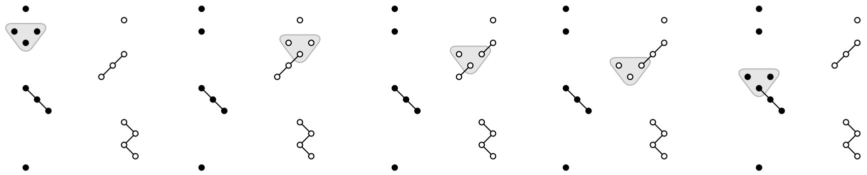
(a) Initially (b) Left multiplication (c) Right multiplication (d) Result

Fig. 4.1: Graphical depiction of Step 0 of the GZ algorithm.



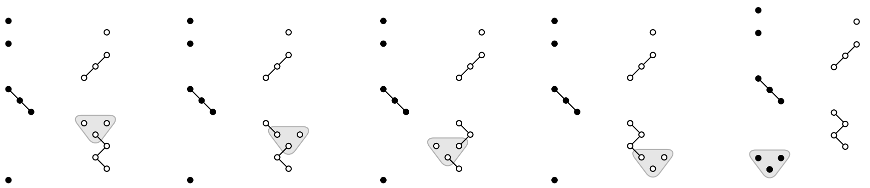
(a) Turnover (b) Right multiplication (c) Left multiplication

Fig. 4.2: Graphical depiction of Step 1 of the GZ algorithm.



(a) $i = 2$ (b) $i = 3$ (c) $i = 4$ (d) $i = 5$ (e) $i = 6$

Fig. 4.3: Graphical depiction of the outcome of Steps 2,3,4,5, and 6.

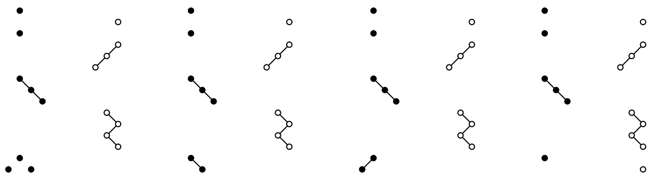


(a) $i = 9$ (b) $i = 10$ (c) $i = 11$ (d) $i = 12$ (e) $i = 13$

Fig. 4.4: Graphical depiction of the outcome of Steps 9,10,11,12, and 13.

Figure 4.3 displays the end results of steps $i = 2, 3, 4, 5, 6$, highlighting already the subsequent turnover operation. The upward movement of the pattern is plainly visible. To illustrate the algorithm in case of a zigzag shape, steps $i = 9, 10, 11, 12, 13$ are visualized in Figure 4.4. In the last step, $i = n - 2 = 14$, a choice has to be made for the updated position vector's 14th element. Figure 4.5 shows the result after the final turnover and also the three possible outcomes of the GZ step for the new value of p_{14}.

5. Associated Krylov spaces. Krylov spaces linked to the condensed matrix pencil provide us with an elegant tool for proving uniqueness and convergence of the presented algorithms. For a Hessenberg-upper triangular pencil (A, B), standard Krylov spaces generated by the matrices AB^{-1} and $B^{-1}A$ play an accommodating role. To deal with the extended



(a) Turnover (b) $p_{14} = \ell$ (c) $p_{14} = r$ (d) $p_{14} = s$

Fig. 4.5: Possible outcomes ensuing from carrying out step $i = 14$.

Krylov spaces stemming from AB^{-1} and $B^{-1}A$ linked to a condensed pencil, we will draw from [16, 18].

The nomenclature for pencils is inherited by matrices. For instance, a matrix is said to be in *condensed* format if it admits a detailed factorization

$$C_{i_1} \cdots C_{i_{n-1}} R, \tag{5.1}$$

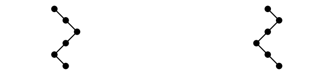
where $i_1, \ldots, i_{n-1}$ denotes a permutation of the integers $1, \ldots, n-1$. Each core transformation C_{i_j} acts on rows i_j and $i_j + 1$, and R is upper triangular. A matrix is said to be in *irreducible condensed* format if it is nonsingular and there are no upper triangular core transforms. The position vector regulating the factorization of a condensed matrix will only comprise values ℓ and r, not s as there is only one single matrix.

5.1. Position vectors associated with the product matrices. It remains to derive the position vectors of the products AB^{-1} and $B^{-1}A$.

Replacing the matrices A and B by their detailed factorization and relying on the passing through operation, we can gather the upper triangular matrices in the back and the core transforms in front without having altered the patterns.

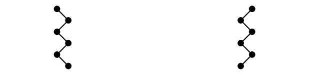
For the product AB^{-1}, the core transforms tied to A are always situated to the left of those bound to B, for $B^{-1}A$ the reverse statement holds. We continue to assume that C_1 is attached to A. To construct the position vector for AB^{-1} from the position vector related to the pencil, we have the ℓ, r entries of $\mathbf{p}$ shaping the patterns in the matrix B swapped due to the inversion of B, and we alternatingly replace the values s by ℓ and r. For the product $B^{-1}A$, the elements ℓ, r linked to B are again swapped and the s's are overwritten by r and ℓ in turn. To clearly differentiate between the various position vectors, we denote the one associated with the pencil by $\mathbf{p}$, the one regulating the shape of AB^{-1} by $\mathbf{p}_{AB}$, and the vector controlling the pattern of $B^{-1}A$ by $\mathbf{p}_{BA}$.

EXAMPLE 5.1. *Consider the pencils of Figures 2.2(c) and 2.2(d). For the double Hessenberg factorization, the position vector* $\mathbf{p} = [\ell, s, \ell, s, \ell]$ *of the pencil is converted into* $\mathbf{p}_{AB} = [\ell, \ell, r, r, \ell]$ *and* $\mathbf{p}_{BA} = [\ell, r, r, \ell, \ell]$*; see Figure 5.1. The position vector bound to Figure 2.2(d) is transformed into* $\mathbf{p}_{AB} = [\ell, r, \ell, r, \ell, r]$ *and* $\mathbf{p}_{BA} = [r, \ell, r, \ell, r]$.



(a) For AB^{-1} (b) For $B^{-1}A$

Fig. 5.1: Shapes linked to Figure 2.2(c).



(a) For AB^{-1} (b) For $B^{-1}A$

Fig. 5.2: Shapes linked to Figure 2.2(d).

EXAMPLE 5.2. *We proceed with the example from Section 4.2, where the position vector was* $\mathbf{p} = [r, s, s, s, r, r, s, \ell, \ell, s, \ell, r, \ell, s]$. *We retrieve* $\mathbf{p}_{AB} = [r, \ell, r, \ell, \ell, \ell, r, \ell, \ell, \ell, \ell, r, \ell, r]$ *and* $\mathbf{p}_{BA} = [r, r, \ell, r, \ell, \ell, \ell, \ell, \ell, r, \ell, r, \ell, \ell]$.

5.2. Extended Krylov spaces. This section is merely a summary of indispensable results from [16, 18] and as such does not contain new theoretical results.

Standard Krylov spaces are denoted by $\mathcal{K}_k(A, \mathbf{v}) = \text{span}\{\mathbf{v}, A\mathbf{v}, \dots, A^{k-1}\mathbf{v}\}$. For a position vector $\mathbf{p}$ bound to a condensed form (5.1), the extended Krylov space $\mathcal{K}_{\mathbf{p},k}(A, \mathbf{v})$ is spanned by k vectors from the bilateral sequence

$$\dots,\ A^3\mathbf{v},\ A^2\mathbf{v},\ A\mathbf{v},\ \mathbf{v},\ A^{-1}\mathbf{v},\ A^{-2}\mathbf{v}, A^{-3}\mathbf{v},\ \dots \tag{5.2}$$

where the position vector $\mathbf{p}$ determines the vectors to be taken.

LEMMA 5.3 (Lemma 3.5 in [18]). *Suppose that in the first $k-1$ components of $\mathbf{p}$ the symbol ℓ appears i times and the symbol r appears j times ($i + j = k - 1$). Then*

$$\mathcal{K}_{\mathbf{p},k}(A, \mathbf{v}) = \text{span}\{A^{-j}\mathbf{v}, \dots, A^{i}\mathbf{v}\} = A^{-j}\mathcal{K}_k(A, \mathbf{v}) = A^{i}\mathcal{K}_k(A^{-1}, \mathbf{v}).$$

EXAMPLE 5.4 (*Hessenberg matrix*). The position vector associated with a Hessenberg matrix only takes values ℓ (Figure 2.1(a)). As a consequence, we retrieve the standard Krylov subspaces: $\mathcal{K}_{\mathbf{p},k}(A, \mathbf{v}) = \text{span}\{\mathbf{v}, A\mathbf{v}, \dots, A^{k-1}\mathbf{v}\} = \mathcal{K}_k(A, \mathbf{v})$.

EXAMPLE 5.5 (*Inverse Hessenberg matrix*). For an inverse Hessenberg matrix (Figure 2.2(a)), the position vector is mirrored as well and only comprises the values r. As in the previous example, the standard Krylov spaces are obtained, but built from the inverse of the matrix A: $\mathcal{K}_{\mathbf{p},k}(A, \mathbf{v}) = \text{span}\{\mathbf{v}, A^{-1}\mathbf{v}, \dots, A^{-k+1}\mathbf{v}\} = \mathcal{K}_k(A^{-1}, \mathbf{v})$.

EXAMPLE 5.6 (*CMV-pattern*). A CMV matrix is characterized by the alternating occurrence of ℓ and r in the position vector. The Krylov spaces accompanying Figure 2.2(b) read $\mathcal{K}_{\mathbf{p},k}(A, \mathbf{v}) = \text{span}\{\mathbf{v}, A\mathbf{v}, A^{-1}\mathbf{v}, A^2\mathbf{v}, A^{-2}\mathbf{v}, \dots\}$.

Let $\mathcal{E}_k = \text{span}\{\mathbf{e}_1, \dots, \mathbf{e}_k\}$, with $1 \le k \le n$. The standard Krylov spaces satisfy the identity $\mathcal{E}_k = \mathcal{K}_k(A, \mathbf{e}_1)$; see, e.g., [11, 20]. Making use of the previously defined extended Krylov spaces, this property carries over neatly.

THEOREM 5.7 (Theorem 3.7 in [18]). *Let A be a matrix in irreducible condensed form (5.1) with associated position vector $\mathbf{p}$. Then for $k = 1, \dots, n-1$,*

$$\mathcal{E}_k = \mathcal{K}_{\mathbf{p},k}(A, \mathbf{e}_1).$$

5.3. Uniqueness of the unitary conversion to a condensed pencil. In the case of unitary core transformations, we can prove essential uniqueness of the conversion to a condensed pencil form. Let $K_{\mathbf{p},k}(A, \mathbf{v})$ denote the matrix having as columns exactly the vectors that generate the sequence of spaces $\mathcal{K}_{\mathbf{p},k}(A, \mathbf{v})$, for $k = 1, \dots, n-1$. More precisely, $K_{\mathbf{p},k}(A, \mathbf{v})$ has $\mathbf{v}$ as the first column, for the second column we take either $A\mathbf{v}$ (if $p_1 = \ell$) or $A^{-1}\mathbf{v}$ (if $p_1 = r$). Each next column is taken from the left (if $p_i = \ell$) or right (if $p_i = r$) of the bilateral sequence (5.2) from columns not yet taken.

The next theorem is self-contained and as such does not inherit the assumption that C_1 is associated to A. The proof proceeds along the lines outlined in [20, 21].

THEOREM 5.8. *Given nonsingular A and B and a position vector $\mathbf{p}$. Let U_1, U_2, V_1, and V_2 be unitary, with U_1 and U_2 sharing the first column (up to a unimodular factor) such that*

$$(A_1, B_1) = U_1^H(A, B)V_1 \quad \text{and} \quad (A_2, B_2) = U_2^H(A, B)V_2,$$

are irreducible and obey the shape imposed by $\mathbf{p}$. Then the pencils (A_1, B_1) and (A_2, B_2) are essentially identical.

Proof. By employing elementary operations on Krylov spaces, we get the following equalities (σ is unimodular):

$$\begin{aligned} U_1 K_{\mathbf{p}_{AB}}(A_1B_1^{-1}, \mathbf{e}_1) &= K_{\mathbf{p}_{AB}}(U_1A_1B_1^{-1}U_1^H, U_1\mathbf{e}_1) \\ &= K_{\mathbf{p}_{AB}}(AB^{-1}, U_1\mathbf{e}_1) \\ &= \sigma K_{\mathbf{p}_{AB}}(AB^{-1}, U_2\mathbf{e}_1) \\ &= \sigma K_{\mathbf{p}_{AB}}(U_2A_2B_2^{-1}U_2^H, U_2\mathbf{e}_1) = \sigma U_2 K_{\mathbf{p}_{AB}}(A_2B_2^{-1}, \mathbf{e}_1). \end{aligned}$$

Starting with an irreducible pencil, both products AB^{-1} and $B^{-1}A$ are irreducible as well. Theorem 5.7 employs this irreducibility constraint and reveals that both $K_{\mathbf{p}_{AB}}(A_1B_1^{-1}, \mathbf{e}_1)$ and $K_{\mathbf{p}_{AB}}(A_2B_2^{-1}, \mathbf{e}_1)$ are upper triangular. By the uniqueness of the QR factorization, we thus conclude that U_1 and U_2 are essentially interchangeable.

It remains to prove that also V_1 and V_2 are essentially indistinguishable. Matching the first columns for V_1 and V_2 is not required as this can be deduced from the restrictions posed on the leading columns of U_1 and U_2.

Both condensed pencils (A_1, B_1) and (A_2, B_2) have the same position vector and thus the first core transform is assigned to either the A or B matrices. Suppose that this top core transform C_1 is associated to B_1 and for the other pencil to B_2. In this case, it holds that the columns $A_1\mathbf{e}_1$ and $A_2\mathbf{e}_1$ are multiples of $\mathbf{e}_1$. Based on $U_1\mathbf{e}_1 = \sigma U_2\mathbf{e}_1$ and

$$V_1\mathbf{e}_1 = A^{-1}U_1A_1\mathbf{e}_1 \quad \text{and} \quad V_2\mathbf{e}_1 = A^{-1}U_2A_2\mathbf{e}_1, \tag{5.3}$$

we deduce that V_1 and V_2 share the first column up to a unimodular factor. For C_1 bound to the A matrices, the B matrices have the first columns as multiples of $\mathbf{e}_1$ and (5.3) still holds for A, A_1, and A_2 substituted by B, B_1, and B_2, respectively. Thus V_1 and V_2 are essentially the same. □

When loosening the constraint of irreducibility in Theorem 5.8, uniqueness can only be guaranteed up to a certain point, which is identical to what happens in the Hessenberg setting [11]. This theorem also proves the appealing result that the outcome of a QZ step, starting from an identical perturbation, is essentially unique.

5.4. Convergence theory. An iteration of the generalized GZ algorithm on (A, B) results in a pair

$$(\hat{A}, \hat{B}) = U^{-1}(A, B)V.$$

Thus,

$$\hat{A}\hat{B}^{-1} = U^{-1}AB^{-1}U. \tag{5.4}$$

The matrix U^{-1} is the product of all of the left transformations that were applied in the course of the iteration. By construction, the first column of U is proportional to $(AB^{-1} - \rho I)\mathbf{e}_1$ if the first component of p_{AB} equals ℓ and $(AB^{-1})^{-1}(AB^{-1} - \rho I)\mathbf{e}_1$ if this component equals r. It follows from Theorem 6.2 of [18] that (5.4) is an iteration of the generalized GR algorithm on AB^{-1}. Therefore, applying Theorem 6.3 of [18], we see that the generalized GR step effects nested subspace iterations on nested generalized Krylov subspaces $\mathcal{E}_k = \mathcal{K}_{\mathbf{p},k}(AB^{-1}, e_1)$. Therefore, good choices of shift will result in rapid convergence of the algorithm.

REMARK 5.9. In [18] we assumed unitary transformations. However, the theorems from that paper that we have cited here do not depend upon the transformations being unitary.

6. Bulge hopping for a mirrored Hessenberg pair. In Section 3 we showed that if the position vector contains only the symbols ℓ and s, the conversion to condensed form can be achieved directly by eliminations on the matrices. The resulting pencil is a mirrored Hessenberg pair. It follows that it is also possible to execute the generalized GZ algorithm directly on the matrices without recourse to detailed factorizations. The GZ iteration then becomes a classical bulge chase, except that the bulge hops back and forth between the matrices in the course of an iteration.

We illustrate the general procedure by an example. Take a pencil with position vector $\mathbf{p} = [\ell, \ell, s, \ell, s, \ell]$. Figure 6.1(a) pictures the sparsity pattern of the Hessenberg pencil. Running a single iteration of the generalized GZ algorithm will result in a bulge chase in which the Hessenberg parts will move up and the bulge will hop from the Hessenberg parts in one matrix to the Hessenberg parts in the other matrix. Figure 6.1(a) shows the initial situation. Step 0 effects a left multiplication that produces a bulge in the B matrix, outlined in gray in Figure 6.1(b). This deviation from the structure is eliminated by a column operation leading to Figure 6.1(c), having a bulge in the A part.

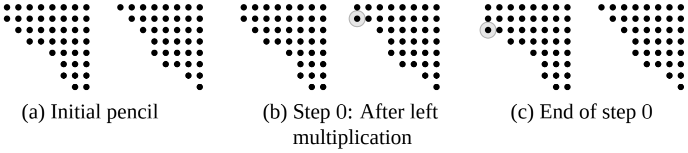

(a) Initial pencil (b) Step 0: After left multiplication (c) End of step 0

Fig. 6.1: Step 0 in a bulge hopping GZ step for mirrored Hessenberg pairs.

Up to the end of step 1 (Figure 6.2(a)), there is no difference between this and the classical GZ algorithm. Nothing changes until we come to the end of the Hessenberg part of A. Step 2 starts as before by removal of the bulge in A by a left multiplication (Figure 6.2(b)), but at this point we must do something different. Removal of the newly created subdiagonal element in Figure 6.2(b) by a column operation would create more fill-in than desired. As the pattern has to move up one position, we will keep this element and instead remove the element $(4, 3)$ from the A matrix by a right multiplication. This multiplication creates a perturbation in the Hessenberg part of the matrix bound to B (Figure 6.2(c)). The bulge has hopped from the Hessenberg part in A to the Hessenberg structure of B.

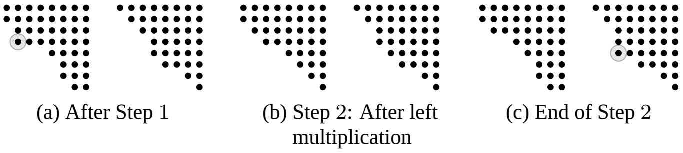

(a) After Step 1 (b) Step 2: After left multiplication (c) End of Step 2

Fig. 6.2: Step 1 and 2 in a bulge hopping GZ step.

In Figure 6.3, the results of steps $3, 4, 5$ are visualized. Again, at the end of the Hessenberg part of B, the bulge migrated back to A.

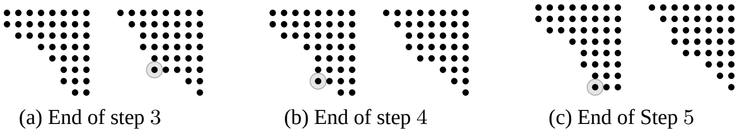
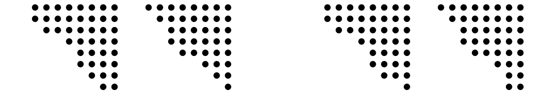

(a) End of step 3 (b) End of step 4 (c) End of Step 5

Fig. 6.3: Steps $3, 4$, and 5 in a bulge hopping GZ step.

The final step offers again flexibility to put the last subdiagonal element in the A or B matrix. The possible outcomes are shown in Figure 6.4, where Figure 6.4(a) has position vector $\mathbf{p} = [\ell, \ell, s, \ell, s, \ell, \ell]$, and Figure 6.4(b) corresponds to $\mathbf{p} = [\ell, \ell, s, \ell, s, \ell, s]$.



(a) End of step 6: choice $= \ell$ (b) End of step 6: choice $= s$

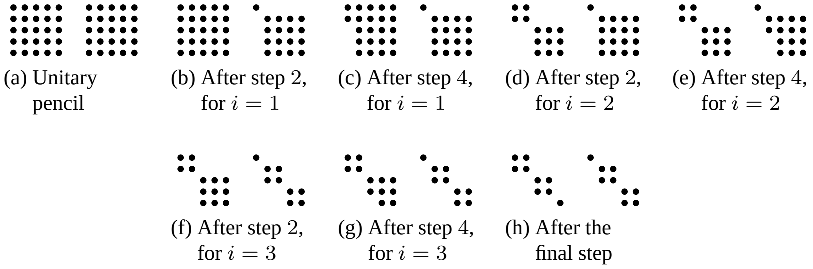
Fig. 6.4: Possible endings of the GZ step.

7. The unitary case.

Consider a pencil (A, B), where both A and B are unitary. In the QR decomposition of a unitary matrix, the R factor is the identity matrix. Therefore, the upper triangular matrices are absent from the detailed factorization of (A, B). After conversion to a condensed form (U, V), the detailed factorization consists of just $n-1$ unitary core transformations distributed between U and V. If we then apply our generalization of the QZ algorithm to this condensed form, the algorithm is greatly simplified by the absence of the upper triangular factors. The work per iteration is reduced from $O(n^2)$ to $O(n)$, so this qualifies as a “fast” algorithm.

Let us consider some special cases. If we convert (A, B) to a condensed form using the position vector $\mathbf{p} = [\ell, \ell, \ldots, \ell]$, we obtain a pencil (U, I), where $U = C_1 \cdots C_{n-1}$ is a unitary upper Hessenberg matrix factored as a product of $n-1$ unitary core transformations. This is essentially a Schur parameterization [3,12]. If we apply our generalized QZ algorithm in this setting making the choice ℓ at the end of each iteration, we have a fast unitary QR algorithm similar to the one originally proposed by Gragg [12].

If we convert to condensed form using the position vector $\mathbf{p} = [s, s, \ldots, s]$, we obtain a pencil (U, V) with $U = C_1 C_3 \cdots$ and $V = C_2 C_4 \cdots$. The matrices in this pencil, which is called a *Schur parameter pencil*, are block diagonal. Since this is a case where the position vector does not contain the symbol r, the results of Section 3.5 apply: we can use Algorithm 3.3, which carries out the reduction by elimination operations directly on the matrices A and B. When we do this in the unitary case, we find that the unitary structure forces many more zeros to appear in the matrices in addition to the ones that are caused directly by the eliminations. Taking these extra zeros into account, we arrive at an algorithm proposed by Bunse-Gerstner and Elsner [6]. Figure 7.1 gives a high level overview of the flow of the algorithm.



(a) Unitary pencil (b) After step 2, for $i = 1$ (c) After step 4, for $i = 1$ (d) After step 2, for $i = 2$ (e) After step 4, for $i = 2$

(f) After step 2, for $i = 3$ (g) After step 4, for $i = 3$ (h) After the final step

Fig. 7.1: Reduction of unitary pencil to Schur parameter form.

Bunse-Gerstner and Elsner [6] also presented fast single- and double-shift QZ algorithms for unitary pencils in this condensed form. It is interesting and puzzling that their single-shift algorithm differs from our generalized QZ algorithm presented here. Our algorithm requires that the pattern moves up one position in each iteration. This means that if U and V contain the odd and even core transformations, respectively, before an iteration, the situation will be reversed after the iteration. The flow of the algorithm is shown graphically in Figure 7.2. Here we are depicting the version of the algorithm that acts directly on the matrices U and V, not on the detailed factorizations. The iteration begins by perturbing the pencil of Figure 7.2(a) by applying a single core transformation on the left involving rows 1 and 2. Figure 7.2(b), which has three extra nonzero elements, is obtained. One element, the one highlighted in gray, is dedicated for removal by operating on the right. After executing the multiplication, the structure of both matrices changes quite a bit. Figure 7.2(c) shows one matrix having a 3×3 block on its diagonal. This block will be trimmed by removing its lower left and upper right element; see Figures 7.2(d) and 7.2(e). After having done that, the counter matrix will have a 3×3 block (Figure 7.2(e))—the elements designated for removal are highlighted—and the procedure continues. Subsequent steps are visualized in Figures 7.2(f) and 7.2(g). As a result, an upward shifted Schur parameter pencil is obtained.

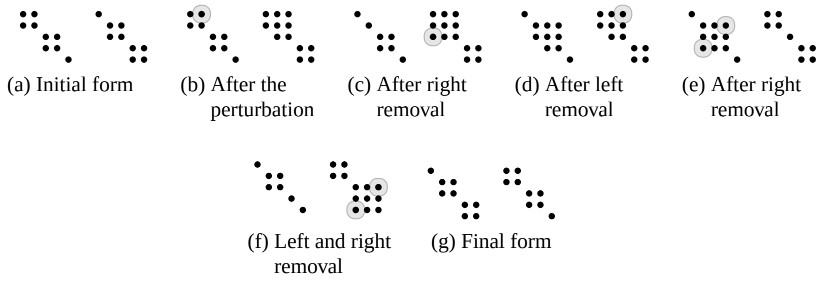

(a) Initial form
(b) After the perturbation
(c) After right removal
(d) After left removal
(e) After right removal
(f) Left and right removal
(g) Final form

Fig. 7.2: Flow of the generalized QZ algorithm on a Schur parameter pencil.

The single-shift algorithm of [6] does not behave this way. The pattern does not move up; the odd core transformations always stay in U. The flow of this algorithm is shown in Figure 7.3. Figures 7.2(b) and 7.3(b) are exactly the same, but the element marked for removal is different. Figure 7.3(c) then illustrates that another element of the matrix on the right will be removed next, again by a right multiplication. A perturbation brings in many nonzero elements and an elimination removes quite some elements, but when comparing the sparseness pattern of Figure 7.2(c) with the one of Figure 7.3(d), we note that the second one accommodates many more nonzero elements. Moreover, instead of alternating left and right annihilations, two successive right will be followed by two successive left annihilations; see Figures 7.3(e) and 7.3(f). After a few more annihilations we arrive at a new Schur parameter pencil in Figure 7.3(h).

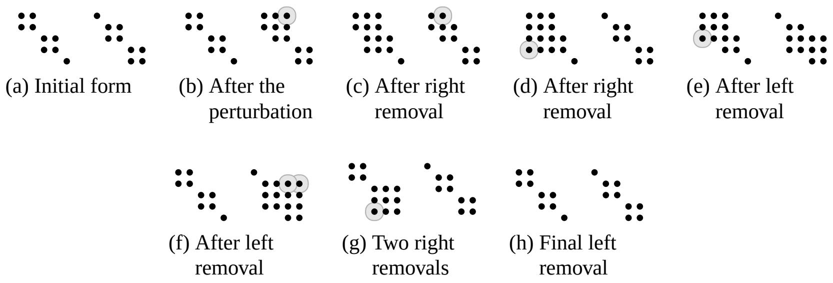

(a) Initial form
(b) After the perturbation
(c) After right removal
(d) After right removal
(e) After left removal
(f) After left removal
(g) Two right removals
(h) Final left removal

Fig. 7.3: Flow of the single shifted QZ algorithm from [6].

This algorithm requires twice as many eliminations as ours does. It also leaves the pattern unchanged. These anomalies can be explained by noticing that the single-shift algorithm of [6] is actually a double-shift algorithm in disguise, as we shall presently explain.

The double-shift algorithm presented in [6] also leaves the pattern invariant, but this is to be expected: a double step should move the pattern up two positions, which would leave the pattern invariant in this particular case. The double-shift algorithm of [6] does not require any more eliminations than the single-shift algorithm does. To gain an understanding of this, consider how the algorithms are set in motion. The single-shift iteration with shift ρ begins with a perturbing core transformation determined by the vector

$$\mathbf{x} = (U - \rho V)\mathbf{e}_1. \tag{7.1}$$

The double-shift iteration [18] with shifts ρ and θ begins with a pair of perturbing core transformations determined by

$$\mathbf{x} = (U - \rho V)(V^H - \theta U^H)\mathbf{e}_1. \tag{7.2}$$

Now notice that if we take $\theta = 0$ in Equation (7.2), we get the same vector $\mathbf{x}$ as in (7.1) because $V^H\mathbf{e}_1 = \mathbf{e}_1$. Therefore, the single-shift algorithm of [6] is really a double-shift algorithm with shifts ρ and 0.

8. Conclusions.

We have described a new, substantially enlarged, class of matrix pencils admissible as condensed forms for computing generalized eigenvalues. A direct reduction procedure to the condensed pencil form and a generalized GZ iteration to compute the eigenvalues were presented. Some well-known algorithms were shown to be special cases of this new family of algorithms.

REFERENCES

[1] V. Adamyan, I. Gohberg, A. Kochubei, G. Popov, Y. Berezansky, M. Gorbachuk, V. Gorbachuk, and H. Langer, eds., *Modern Analysis and Applications. The Mark Krein Centenary Conference*, vol. 190 of Operator Theory: Advances and Applications, Birkhäuser, Basel, 2009.

[2] N. I. Akhiezer, *The Classical Moment Problem and Some Related Questions in Analysis*, Oliver & Boyd, Edinburgh, 1965.

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

[4] J. Aurentz, R. Vandebril, and D. S. Watkins, *Fast computation of the zeros of a polynomial via factorization of the companion matrix*, SIAM J. Sci. Comput., 35 (2013), pp. A255-A269.

[5] R. Crux-Barroso and S. Delvaux, *Orthogonal Laurent polynomials on the unit circle and snake-shaped matrix factorizations*, J. Approx. Theory, 161 (2009), pp. 65–87.

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

[7] M. Fiedler, *A note on companion matrices*, Linear Algebra Appl., 372 (2003), pp. 325–331.

[8] ———, *Complementary basic matrices*, Linear Algebra Appl., 384 (2004), pp. 199–206.

[9] J. G. F. Francis, *The QR transformation a unitary analogue to the LR transformation. I*, Comput. J., 4 (1961), pp. 265–271.

[10] ———, *The QR transformation II.*, Comput. J., 4 (1962), pp. 332–345.

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

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

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

[14] H. Kimura, *Generalized Schwarz form and lattice-ladder realizations of digital filters*, IEEE Trans. Circuits and Systems, 32 (1985), pp. 1130–1139.

[15] B. Simon, *CMV matrices: Five years after*, J. Comput. Appl. Math., 208 (2007), pp. 120–154.

[16] R. VANDEBRIL, *Chasing bulges or rotations? A metamorphosis of the QR-algorithm*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 217–247.

[17] R. VANDEBRIL, M. VAN BAREL, AND N. MASTRONARDI, *Matrix Computations and Semiseparable Matrices, Vol. I: Linear Systems*, Johns Hopkins University Press, Baltimore, 2008.

[18] R. VANDEBRIL AND D. S. WATKINS, *A generalization of the multishift QR algorithm*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 759–779.

[19] D. S. WATKINS, *Some perspectives on the eigenvalue problem*, SIAM Rev., 35 (1993), pp. 430–471.

[20] ———, *The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods*, SIAM, Philadelphia, 2007.

[21] ———, *The QR algorithm revisited*, SIAM Rev., 50 (2008), pp. 133–145.