A GENERAL MATRIX EIGENVALUE ALGORITHM*

CHARLES F. VAN LOAN†

Abstract. A general QR-type process called the VZ algorithm is presented for the solution of the general matrix eigenvalue problem $ACx = \lambda BDx$. The matrices involved may be rectangular. For appropriate choices of $A$, $B$, $C$ and $D$, we have some of the more familiar types of eigenproblems, and this is reflected in the fact that the $QR$, $OZ$ and $SVD$ algorithms are all special cases of the VZ algorithm. The main emphasis is upon the algorithm's generality as well as its bearing upon the generalized singular value problem $A^T A x = \mu^2 B^T B x$.

1. Introduction and notation. The $QR$ algorithm [2] for the standard real eigenvalue problem

(I) \[ Ax = \lambda x, \quad A \in \mathbb{R}^{n \times n}, \quad 0 \neq x \in \mathbb{R}^n, \quad \lambda \in \mathbb{C}, \]

has two important derivatives in the $QZ$ algorithm [5], which solves the generalized eigenvalue problem

(II) \[ Ax = \lambda Bx, \quad A, B \in \mathbb{R}^{n \times n}, \quad 0 \neq x \in \mathbb{R}^n, \quad \lambda \in \mathbb{C}, \]

and the $SVD$ algorithm [3], which solves the singular value problem

\[ A^T A x = \mu^2 x, \quad A \in \mathbb{R}^{m \times n}, \quad m \geq n, \]
\[ 0 \neq x \in \mathbb{R}^n, \quad \mu \geq 0. \]

This paper is about another $QR$-type process called the VZ algorithm which was devised in conjunction with the generalized singular value problem

(IV) \[ A^T A x = \mu^2 B^T B x, \quad A, B \in \mathbb{R}^{m \times n}, \quad m \geq n, \]
\[ 0 \neq x \in \mathbb{R}^n, \quad \mu \geq 0. \]

(A discussion of these problems can be found in [7].) This new routine solves a problem even more general than (IV), namely,

(V) \[ ACx = \lambda BDx, \quad A, B \in \mathbb{R}^{m \times n}, \quad m \geq n, \]
\[ C, D \in \mathbb{R}^{n \times n}, \quad 0 \neq x \in \mathbb{C}^n, \quad \lambda \in \mathbb{C}. \]

The problems (I)–(IV) are clearly special cases of the problem (V), and hence $VZ$ is capable of solving all of the various eigenproblems alluded to thus far. We shall in fact show that $VZ$ is equivalent to $QR$, $OZ$ and $SVD$ when it solves the problems (I), (II) and (III), respectively. However, our purpose is not to advocate the blanket use of this general algorithm but rather to advance it as a useful device for unifying the theoretical and computational aspects of the entire $QR$ family of algorithms.

We begin in §2 by reviewing the numerical dangers of transforming the problems (II)–(V) into standard form (e.g., $ACx = \lambda BDx \Leftrightarrow Yx = \lambda x$, where

---

* Received by the editors October 4, 1974.
† Mathematics Department, University of Manchester, Manchester, England, M13 9PL. This work was supported by the Science Research Council under Grant B/RG/4071.
Y = (BD)^{-1}(AC)). This leads naturally to the decomposition theorems of § 3 which the various members of the QR family compute. Next, we examine the features which these algorithms have in common—an underlying equivalence relation, a finite step initial reduction and a Francis-type iteration. These topics are explored in § 4, § 5 and § 6. In § 7 we conclude with a numerical example and a comment upon the relevance of the VZ algorithm to the generalized singular value problem.

Our notation is as follows: if A is a matrix, then a_{ij} is its (i, j) entry, \( A^T \) is its transpose, \( A^* \) is its complex conjugate transpose, \( \| A \| \) is its 2-norm, and \( \lambda (A) \) is its set of eigenvalues (multiplicities included). For a pair of square matrices A and B, we let \( \lambda (A, B) \) denote the zeros of \( \det (A - \lambda B) \).

\( A = \nabla \) means A is upper triangular (\( a_{ij} = 0, i > j \)); \( A = \bigtriangleup \) means that \( A^T = \nabla \); \( A = \bigtriangledown \) means that A is diagonal (\( a_{ii} = 0, i \neq j \)); \( A = \bigtriangledown \) means that A is upper bidiagonal (\( a_{ii} = 0, i \neq j, j-1 \)); and \( A = \nabla \) means that A is upper Hessenberg (\( a_{ii} = 0, i > j + 1 \)). \( I_{nn} \) denotes the \( A \in \mathbb{R}^{n \times n} \) for which \( a_{ii} = \delta_{ii} \). As a special case, \( I_n = I_{nn} \).

To indicate computed quantities, we use the notation fl[exp] where exp is any algebraic expression. For a given computing machine, \( \varepsilon \) will denote the largest floating point number for which \( \text{fl}[1 + \varepsilon] \) equals 1. If Q and A are stored matrices, \( A := QA \) means (a) form QA and (b) store the result in A.

2. The numerical dangers of putting general eigenproblems into standard form. We shall see that the computed eigenvalues \( \lambda \) of the problem (V) which the VZ algorithm returns satisfy

\[
\det \left[(A + E_A)(C + E_C) - \lambda (B + E_B)(D + E_D)\right] = 0,
\]

where \( E_A, E_B, E_C \) and \( E_D \) are matrices whose norms have orders \( \varepsilon \| A \|, \varepsilon \| B \|, \varepsilon \| C \| \) and \( \varepsilon \| D \| \), respectively. Similar statements can be made about the computed results of the QR, QZ and SVD algorithms. The result (2.1) is made possible because the VZ algorithm has incorporated certain features of the SVD and QZ routines. First, there is no attempt to form the matrix products AC or BD. (This corresponds to the avoidance of \( A^T A \) in SVD.) Second, there is no attempt to work with \( (BD)^{-1}(AC) \). (This corresponds to the avoidance of \( B^{-1} A \) in QZ.)

To show how matrix products can cause undue inaccuracy, we consider the singular value problem (III) and borrow an example from [3]. Let A be given by

\[
A = \begin{bmatrix}
1 & 1 \\
\sqrt{\varepsilon} & 0 \\
0 & \sqrt{\varepsilon}
\end{bmatrix}.
\]

Now, the singular values of any matrix A are the nonnegative square roots of the eigenvalues of \( A^T A \). This immediately suggests the following method for our 3 \times 2 example:

(a) Compute \( Y = \text{fl}[A^T A] \);
(b) Calculate \( \lambda (Y) = \{ \lambda_1, \lambda_2 \} \) (say, with QR);
(c) Take \( \sqrt{\lambda_1} \) and \( \sqrt{\lambda_2} \) as our computed singular values.
Since \( f[1+\varepsilon] = 1 \), we find that

\[
Y = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},
\]

giving, at best, \( \lambda_1 = 2 \) and \( \lambda_2 = 0 \). Thus, by the above technique, we are led to believe that \( A \) has singular values \( \sqrt{2} \) and 0.

We now think of \( \sqrt{2} \) and 0 as the exact singular values of some matrix \( A + E \) where \( E \) is hopefully small. Using the Wielandt–Hoffman theorem for singular values, we shall show that this is not exactly the case. This theorem states that if \( A \) has singular values \( \{\mu_i\} \) with \( \mu_1 \geq \mu_2 \geq \ldots \geq \mu_n \) and \( A + E \) has singular values \( \{\tilde{\mu}_i\} \) with \( \tilde{\mu}_1 \geq \tilde{\mu}_2 \geq \ldots \geq \tilde{\mu}_n \), then

\[
\sum_i (\mu_i - \tilde{\mu}_i)^2 \leq \|E\|^2 = \text{trace} (E^T E) \leq n\|E\|^2.
\]

In our example, \( A \) has exact singular values \( \mu_1 = \sqrt{2+\varepsilon} \) and \( \mu_2 = \sqrt{\varepsilon} \), while \( A + E \) has singular values \( \tilde{\mu}_1 = \sqrt{2} \) and \( \tilde{\mu}_2 = 0 \). Hence from (2.2), we obtain

\[
2\|E\|^2 \geq (\sqrt{2+\varepsilon} - \sqrt{2})^2 + (\sqrt{\varepsilon} - 0)^2 > \varepsilon,
\]

which shows that \( \|E\| \) has order \( \sqrt{\varepsilon}\|A\| \). Thus \( \|E\| \) is not small relative to the machine precision \( \varepsilon \). This shows why the formation of the products \( AC \) and \( BD \) in the problem (V) can lead to displeasing and unnecessary errors.

A second numerical difficulty arises in connection with the \( ACx = \lambda BDx \) problem if we attempt to form \( (BD)^{-1} \) with the intention of solving the theoretically equivalent standard problem \( (BD)^{-1} ACx = \lambda x \). Of course, we reach an impasse with this technique if \( BD \) is singular, but practical difficulties can arise long before singularity in \( BD \) sets in. To see why, consider the \( Ax = \lambda Bx \) problem with

\[
A = \begin{bmatrix} \sqrt{\varepsilon} & \sqrt{\varepsilon} \\ 1 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 1 \\ \sqrt{\varepsilon} & -\sqrt{\varepsilon} \end{bmatrix}.
\]

A and \( B \) have identical singular values, the smaller of which is \( \sqrt{2}\varepsilon \). Now suppose we have computed \( B^{-1} \) exactly,

\[
B^{-1} = \frac{1}{2} \begin{bmatrix} 1 & \varepsilon^{-1/2} \\ 1 & -\varepsilon^{-1/2} \end{bmatrix}
\]

and then compute \( B^{-1} A \). Since \( f[\varepsilon^{-1/2} \pm \varepsilon^{1/2}] = \varepsilon^{-1/2} \), we find that

\[
f[B^{-1} A] = \frac{1}{2} \begin{bmatrix} \varepsilon^{-1/2} & -\varepsilon^{-1/2} \\ -\varepsilon^{-1/2} & \varepsilon^{-1/2} \end{bmatrix},
\]

which has eigenvalues 0 and \( \varepsilon^{-1/2} \). Hence, for some perturbation matrices \( E_A \) and \( E_B \), we have

\[
0 = \det [(A + E_A) - \lambda (B + E_B)], \quad \lambda = 0, \quad \varepsilon^{-1/2}.
\]

By arguments similar to those given above, we can show that \( \|E_A\| \) has order \( \sqrt{\varepsilon}\|A\| \).
Returning to the \( ACx = \lambda BDx \) problem, we can conclude from the preceding discussion that if we work with \((BD)^{-1}AC\), then our computed eigenvalues \( \lambda \) may not satisfy (2.1) for small perturbation matrices \( E_a, E_b, E_c \) and \( E_d \).

3. A general unitary decomposition theorem. The results of the previous section place us in a difficult position in that we are seeking to solve (V) without forming \( AC, BD \) or \((BD)^{-1}\). The way out of this dilemma lies in the Decomposition 4, which we shall shortly prove and which the \( VZ \) algorithm computes. We motivate the discussion of this general result by reminding the reader that the \( QR, QZ \) and \( SVD \) algorithms, respectively, compute (in the limit) the following unitary decompositions.

DECOMPOSITION 1 (The Schur decomposition). If \( A \in \mathbb{R}^{n \times n} \), then there exists an \( n \times n \) unitary matrix \( Q \) such that \( Q^*AQ = R \) is upper triangular. (The eigenvalues of \( A \) are given by the \( r_n \).)

DECOMPOSITION 2 (The generalized Schur decomposition [6]). If \( A, B \in \mathbb{R}^{m \times n} \), then there exist \( n \times n \) unitary matrices \( Q \) and \( Z \) such that \( QAZ = R \) and \( QBZ = S \) are both upper triangular. (The generalized eigenvalues are then given by the ratios \( r_n/s_m, s_m \neq 0 \).)

DECOMPOSITION 3 (The singular value decomposition. See [1] for example). If \( A \in \mathbb{R}^{m \times n} \), then there exist orthogonal matrices \( U \) and \( V \) of orders \( m \) and \( n \), respectively, such that \( U^*AV = D \) is a diagonal matrix with nonnegative entries. (The singular values of \( A \) are then given by the \( d_n \).)

Notice (a) that Decomposition 2 indicates how problem (II) can be solved without trying to form \( B^{-1}A \) and (b) how Decomposition 3 suggests how we can find the singular values of \( A \) without forming \( A^T \). The Decomposition 4 below has the Decompositions 2 and 3 as special cases and, not surprisingly, indicates how we can solve the \( ACx = \lambda BDx \) problem without forming \( AC, BD \) or \((BD)^{-1}\).

We shall need the following two lemmas.

**Lemma 1.** Any sequence of \( n \times n \) unitary matrices \( \{Q_k\} \) contains a converging subsequence \( \{Q_k\} \) whose limit \( Q \) is itself unitary.

**Proof.** The proof follows from the \( \mathbb{C}^* \) version of the Bolzano–Weierstrass theorem. See [8, p. 105].

**Lemma 2.** Let \( G \in \mathbb{C}^{m \times m} \) and \( H \in \mathbb{C}^{m \times n} \) (\( m \geq n \)) have the property that rank \((G) = \text{rank}(H) = n \). If the product \( GH = R \) is upper triangular, then there exists an \( m \times m \) unitary matrix \( U \) such that \( GU \) and \( U^*H \) are both upper triangular.

**Proof.** We can always find an \( m \times n \) unitary matrix \( U \) such that \( U^*H \) is upper triangular [4]. Partition \( GU \) and \( U^*H \) as follows:

\[
GU = \begin{bmatrix} G_1 & G_2 \end{bmatrix}, \quad U^*H = \begin{bmatrix} H_1 \\ 0 \end{bmatrix},
\]

where \( G_1, H_1 \in \mathbb{C}^{n \times n} \), \( H_1 = \nabla \) and \( G_2 \in \mathbb{C}^{(m-n) \times n} \). We must show that \( G_1 \) is upper triangular, but this is easy since

\[
R = GH = (GU)(U^*H) = G_1H_1
\]

implies

\[
G_1 = RH_1^{-1} = \nabla \nabla^{-1} = \nabla.
\]

\((H_1^{-1} \text{ exists because } n = \text{rank}(H) = \text{rank}(U^*H) = \text{rank}(H_1)). \quad \square\)
We now come to the main result of this section. The decomposition which follows is computed (in the limit) by the VZ algorithm.

**Decomposition 4.** Let $A$ and $B$ be in $\mathbb{R}^{m \times n}$, and let $C$ and $D$ be in $\mathbb{R}^{m \times n}$ with $m \geq n$. There exist unitary matrices $Q$ and $U$ in $\mathbb{C}^{n \times n}$ and unitary matrices $V$ and $Z$ in $\mathbb{C}^{m \times m}$ such that

$$
\tilde{A} = QAZ = \nabla, \quad \tilde{B} = QBV = \nabla, \quad \tilde{C} = Z^*CU = \nabla, \quad \tilde{D} = V^*DU = \nabla.
$$

**Proof.** First assume that $A$, $B$, $C$, and $D$ all have rank $n$ and that the square $n \times n$ matrix $BD$ is invertible. By Decomposition 1, we can find an $n \times n$ unitary matrix $Q$ such that $Q(AC)(BD)^{-1}Q^* = \nabla$. Using Lemma 2 (with $G = QAC$ and $H = (BD)^{-1}Q^*$), we can thus find an $n \times n$ unitary matrix $U$ such that

$$
(3.1) \quad Q(AC)U = \nabla,
$$
$$
(3.2) \quad U^*(BD)^{-1}Q^* = \nabla.
$$

Inverting both sides of (3.2) gives

$$
(3.3) \quad Q(BD)U = \nabla.
$$

Again we apply Lemma 2 to (3.1) and (3.3) to produce $m \times m$ unitary matrices $Z$ and $V$ such that, from (3.1), $QAZ = \nabla$ and $Z^*CU = \nabla$, and from (3.3) $QBV = \nabla$ and $V^*DU = \nabla$.

We have thus proved the decomposition in the case when our matrices obey some rather strict conditions on rank. We can nevertheless proceed without these restrictions. For general $A$, $B$, $C$, and $D$ let

$$
D = Q_B \begin{bmatrix} R_D \\ 0 \end{bmatrix}, \quad R_D = \nabla \in \mathbb{C}^{n \times n},
$$

be the Householder QR decomposition of $D$ where $R_D$ has nonnegative diagonal elements. Similarly, let

$$
BQ_D = Q_B \begin{bmatrix} R_B \\ S_B \end{bmatrix}, \quad R_B = \nabla \in \mathbb{C}^{n \times n}, \quad S_B \in \mathbb{C}^{n \times (m-n)}
$$

be the QR decomposition of $BQ_D$. If

$$
D_k = Q_B \begin{bmatrix} R_D + (1/k)I_k \\ 0 \end{bmatrix} \quad \text{and} \quad B_k = Q_B \begin{bmatrix} R_B + (1/k)I_k \\ S_B \end{bmatrix} Q_B^*,
$$

then it can easily be shown that $\lim \|B_k - B\| = \lim \|D_k - D\| = 0$ and that, for every positive integer $k$, rank $(B_k) = \text{rank } (D_k) = \text{rank } (B_kD_k) = n$. We also have full rank matrices $A_k$ and $C_k$ which converge to $A$ and $C$, respectively.

By the first part of the proof, we can find, for each $k$, unitary $Q_k$, $U_k$, $V_k$, and $Z_k$ such that $Q_kA_kZ_k$, $Q_kB_kV_k$, $Z_k^*C_kU_k$, and $V_k^*D_kU_k$ are all upper triangular. Using Lemma 2 repeatedly, we can find a subsequence $\{k_i\}$ of the positive integers such that $Q_{k_i} \rightarrow Q$, $U_{k_i} \rightarrow U$, $V_{k_i} \rightarrow V$, and $Z_{k_i} \rightarrow Z$. It is easy to verify that $QAZ$, $QBV$, $Z^*CU$, and $V^*DU$ are all upper triangular. $\Box$

**Remark 1.** Suppose that the diagonal elements of $\tilde{A}$, $\tilde{B}$, $\tilde{C}$ and $\tilde{D}$ are given by $\alpha_n$, $\beta_n$, $\gamma_i$, and $\delta_n$, respectively. We then have $\prod_i (\alpha_i \gamma_i - \lambda \beta_i \delta_i) = \det \begin{bmatrix} \tilde{A} & \tilde{C} \\ \tilde{B} & \tilde{D} \end{bmatrix}$.
\(- \lambda \tilde{B} \tilde{D} \) = \( \det \left[ (QAZ)(Z^*CU) - \lambda (QBV)(V^*DU) \right] = \det (QU) \det (AC - \lambda BD) \). Thus, the \( \lambda \) which make \( AC - \lambda BD \) singular are given by the ratios
\[
\frac{\alpha_i \gamma_i}{\beta_i \delta_i}, \quad \beta_i \delta_i \neq 0,
\]
which clearly demonstrates the relevance of this decomposition to the problem (V).

Remark 2. As an illustration of the generality of Decomposition 4, we use it to prove the singular value decomposition. Specifically, we can find unitary \( Q, U, V \) and \( Z \) such that

\[
\begin{align*}
(3.4) \quad & QA^*Z = \nabla, \\
(3.5) \quad & Z^*AU = \nabla, \\
(3.6) \quad & QI_{nm}V = \nabla, \\
(3.7) \quad & V^*I_{nm}U = \nabla.
\end{align*}
\]

Equations (3.6) and (3.7) tell us that \( QU = \nabla \), which implies that \( QU = D = \nabla \) since a unitary triangular matrix must be diagonal. Substituting \( Q = DU^* \) into (3.4), we find that
\[
DU^*A^*Z = \nabla \quad \text{or} \quad U^*A^*Z = \nabla,
\]
which implies

\[
(3.8) \quad Z^*AU = \nabla.
\]

By comparing (3.5) and (3.8), we can conclude that \( Z^*AU = \nabla \).

4. The underlying equivalence relation. Having established the importance of Decomposition 4 to the problem (V), we now turn our attention to its computation. Greater understanding is achieved if we introduce the idea of an "underlying equivalence relation". Behind every eigenvalue routine is an equivalence relation under which the sought-after quantities are preserved. The following table summarizes this for \( QR, QZ \) and \( SVD \):

<table>
<thead>
<tr>
<th>Routine</th>
<th>Underlying Equivalence Relation</th>
<th>Relevance</th>
</tr>
</thead>
<tbody>
<tr>
<td>( QR )</td>
<td>( A_1 \sim A_1, \text{ iff } A_2 = Q^*A_1Q, \quad Q^*Q = I_n )</td>
<td>( \lambda(A_1) = \lambda(A_1) )</td>
</tr>
<tr>
<td>( QZ )</td>
<td>( (A_2, B_2) \sim (A_1, B_1) \text{ iff } A_2 = QA_1Z, \quad B_2 = QB_1Z, \quad Q^*Q = Z^*Z = I_n )</td>
<td>( \lambda(A_2, B_2) = \lambda(A_1, B_1) )</td>
</tr>
<tr>
<td>( SVD )</td>
<td>( A_1 \sim A_1, \text{ iff } A_2 = U^*AV, \quad U^*U = I_m, \quad V^*V = I_n )</td>
<td>( \lambda(A_1^TV^*A_1) = \lambda(A_1^TA_1) )</td>
</tr>
</tbody>
</table>

(These equivalence relations have obvious complex analogies.)
As far as the VZ algorithm is concerned, it iterates upon 4-tuples of matrices and revolves around the following equivalence relation:

\[(A_2, B_2, C_2, D_2) \sim (A_1, B_1, C_1, D_1)\]

iff there exist orthogonal matrices \(Q, U, V\) and \(Z\) of appropriate orders such that

\[
\begin{align*}
A_2 &= QA_1Z, & B_2 &= QB_1V, \\
C_2 &= Z^TC_1U, & D_2 &= V^TD_1U.
\end{align*}
\]

(4.1)

It is easy to show that \(\lambda(A_2C_2, B_2D_2) = \lambda(A_1C_1, B_1D_1)\). Thus, the sought-after eigenvalues are preserved whenever the four matrices are updated in the fashion indicated by (4.1).

5. The finite step initial reduction. It is typical in a QR process to begin by reducing the matrix (matrices) involved to some equivalent condensed form. We review this fact in the following table:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Original Data</th>
<th>Equivalent Condensed Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>QR</td>
<td>(A_0)</td>
<td>(A_1 = Q^TA_0Q = \nabla)</td>
</tr>
<tr>
<td>QZ</td>
<td>(A_0, B_0)</td>
<td>(A_1 = QA_0Z = \nabla) (\quad B_1 = QB_0V = \nabla)</td>
</tr>
<tr>
<td>SVD</td>
<td>(A_0)</td>
<td>(A_1 = U^TA_0V = \nabla)</td>
</tr>
</tbody>
</table>

In each case the reduction to condensed form is accomplished in a finite number of operations.

The object of the initial reduction in VZ is to find orthogonal \(Q, U, V\) and \(Z\) such that

\[
\begin{align*}
A_1 &= QA_0Z = \nabla, & B_1 &= QB_0V = \nabla, \\
C_1 &= Z^TC_0U = \nabla, & D_1 &= V^TD_0U = \nabla,
\end{align*}
\]

(5.1)

where \(A_0, B_0, C_0\) and \(D_0\) are the four original matrices. For particular choices of these matrices, (5.1) reduces to the QR, QZ and SVD reductions summarized in Table 2. For example, if \(C_0 = D_0 = I_o\), then \(QA_0U = (QA_0Z)(Z^TL_U) = \nabla \nabla = \nabla\) and, similarly, \(QB_0U = (QB_0V)(V^TL_U) = \nabla \nabla = \nabla\). This is precisely the condensed form of the QZ algorithm applied to \(A_0\) and \(B_0\).

We now adopt some conventions in order to describe efficiently how VZ computes the reduction (5.1). Let \(H^{(k)}\) denote the set of real \(d \times d\) Householder matrices of the form \(L_u + \rho uu^T\) where

(a) \(u\) and \(v\) are in \(R^d\) and are linearly dependent,
(b) \(\rho = -2/\langle u, v \rangle\),
(c) only components \(k, k + 1, \ldots, k + r - 1\) of \(u\) are nonzero.

When \(r = 2\) or 3, our Householder matrices will actually be of the modified type (see [5]), but we shall largely ignore this. It serves only to remark here that
premultiplication (postmultiplication) by a matrix in $H^r(k)$ affects only rows (columns) $k, k+1, \cdots, k+r-1$.

Another convention which we must understand is that the computed orthogonal matrices are denoted by $Q, U, V$ and $Z$ and are applied to the “current” arrays $A, B, C$ and $D$ in a fashion suggested by (4.1). For example, an orthogonal matrix applied on the right of the current $B$ is called a “$V$”, and its transpose $V^T$ must be applied on the left of the current $D$. Similar comments can be made concerning the orthogonal matrices $Q, U$ and $Z$, and together these comments make up the “rules of $VZ$” which must be obeyed if the eigenvalues of the original matrices are to be preserved.

We are now equipped to show how the reduction (5.1) is carried out. The process begins with three Householder triangularizations.

(i) Find an orthogonal $Z$ so $Z^TC = \nabla$ and apply as follows:

$$C := Z^TC, \quad A := AZ.$$  

(ii) Find an orthogonal $V$ so $V^TD = \nabla$ and apply as follows:

$$D := V^TD, \quad B := BV.$$  

(iii) Find an orthogonal $Q$ so $QB = \nabla$ and apply as follows:

$$B := QB, \quad A := QA.$$  

The details of these triangularizations can be found in [4]. If, as an example, $m = 7$ and $n = 5$, then the updated matrices have the following form:

\[
\begin{array}{cccccc}
\text{x} & \text{x} & \text{x} & \text{x} & \text{x} & \text{x} \\
\text{x} & \text{x} & \text{x} & \text{x} & \text{x} & \text{x} \\
A = \text{x} & \text{x} & \text{x} & \text{x} & \text{x} & \text{x} \\
\text{x} & \text{x} & \text{x} & \text{x} & \text{x} & \text{x} \\
\text{x} & \text{x} & \text{x} & \text{x} & \text{x} & \text{x} \\
\text{x} & \text{x} & \text{x} & \text{x} & \text{x} & \text{x} \\
\text{x} & \text{x} & \text{x} & \text{x} & \text{x} & \text{x} \\
\text{x} & \text{x} & \text{x} & \text{x} & \text{x} & \text{x} \\
\text{0} & \text{x} & \text{x} & \text{x} & \text{x} & \text{x} \\
\text{0} & \text{0} & \text{x} & \text{x} & \text{x} & \text{x} \\
\text{0} & \text{0} & \text{0} & \text{x} & \text{x} & \text{x} \\
C = \text{0} & \text{0} & \text{0} & \text{x} & \text{x} & \text{x} \\
\text{0} & \text{0} & \text{0} & \text{x} & \text{x} & \text{x} \\
\text{0} & \text{0} & \text{0} & \text{x} & \text{x} & \text{x} \\
\text{0} & \text{0} & \text{0} & \text{x} & \text{x} & \text{x} \\
\text{D} = \text{0} & \text{0} & \text{0} & \text{x} & \text{x} & \text{x} \\
\text{0} & \text{0} & \text{0} & \text{x} & \text{x} & \text{x} \\
\text{0} & \text{0} & \text{0} & \text{x} & \text{x} & \text{x} \\
\text{0} & \text{0} & \text{0} & \text{x} & \text{x} & \text{x} \\
\text{0} & \text{0} & \text{0} & \text{x} & \text{x} & \text{x} \\
\end{array}
\]

(Here “x” denotes an arbitrary nonzero element.)

Now the trick is to reduce $A$ to upper Hessenberg while preserving the triangularity of the other three matrices. Our technique is patterned after the $QZ$ initial reduction, and we present a few steps of it with our $7 \times 5$ example so as to convey the general idea.
Find \( Q \in H_2^2(4) \) to zero \( a_{51} \).

\[
\begin{array}{cccccccc}
  x & x & x & x & x & x & x & x \\
  x & x & x & x & x & x & x & x \\
  x & x & x & x & x & x & x & x \\
  x & x & x & x & x & x & x & x \\
  x & x & x & x & x & x & x & x \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

(5.2) \( A := QA = x \ x \ x \ x \ x \ x, \quad B := QB = 0 \ 0 \ 0 \ 0 \ x \ x \ x \ x \).

\[
\begin{array}{cccccccc}
  x & x & x & x & x & x & x & x \\
  x & x & x & x & x & x & x & x \\
  x & x & x & x & x & x & x & x \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

(Here "+" indicates the presence of an unwanted nonzero entry.) The \( C \) and \( D \) arrays are unchanged, and observe that in the process of zeroing \( a_{51} \), a nonzero element was generated in the \((5,4)\) position of \( B \).

Find \( V \in H_2^2(4) \) to zero \( b_{54} \).

\[
\begin{array}{cccccccc}
  x & x & x & x & x & x & x & x \\
  x & x & x & x & x & x & x & x \\
  x & x & x & x & x & x & x & x \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

(5.3) \( B := BV = 0 \ 0 \ 0 \ x \ x \ x \ x, \quad D := V^T D = 0 \ 0 \ 0 \ 0 \ x \ x \ x \ x \).

\[
\begin{array}{cccccccc}
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

The \( A \) and \( C \) arrays remain the same while in the process of zeroing \( b_{54} \), a nonzero element was generated in the \((5,4)\) position of \( D \).

Find \( U \in H_2^2(4) \) to zero \( d_{54} \).

\[
\begin{array}{cccccccc}
  x & x & x & x & x & x & x & x \\
  x & x & x & x & x & x & x & x \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

(5.4) \( D := DU = 0 \ 0 \ 0 \ x \ x \ x \ x, \quad C := CU = 0 \ 0 \ 0 \ 0 \ x \ x \ x \).

\[
\begin{array}{cccccccc}
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

The result of this step is that a nonzero element was generated in the \((5,4)\) position of \( C \). The \( A \) and \( B \) arrays are left unchanged.
Find $Z \in H^2 \{4\}$ to zero $c_{44}$.
\[
\begin{array}{cccccc}
0 & x & x & x & x & x \\
0 & 0 & x & x & x & x \\
C := ZC = 0 & 0 & 0 & x \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{array}
\]
\[
A := AZ^T = x & x & x & x & x & x.
\]
With this application of $Z$, no unwanted nonzero elements were generated. This 
"zero chasing" technique, evident in (5.2)–(5.5), is repeated as $a_{41}$, $a_{42}$, $a_{52}$, $a_{52}$ 
and $a_{52}$, are zeroed in turn. The process terminates with $A$, $B$, $C$ and $D$ in the 
condensed form specified by (5.1). For general $m$ and $n$, the $VZ$ initial reduction 
goes as follows:
1. For $k = 1, 2, \ldots, n$, find $Z^T \in H^m_{m-k+1}(k)$ to zero $c_{k+1,k}, \ldots, c_{mk}$. Apply $Z$
as follows: $C := Z^TC$, $A := AZ$.
2. For $k = 1, 2, \ldots, n$, find $V^T \in H^m_{m-k+1}(k)$ to zero $d_{k+1,k}, \ldots, d_{mk}$. Apply $V$
as follows: $D := V^TD$, $B := BV$.
3. For $k = 1, 2, \ldots, n-1$, find $Q \in H^m_{n-k+1}(k)$ to zero $b_{k+1,k}, \ldots, b_{nk}$. Apply $Q$
as follows: $A := QA$, $B := QB$.
4. For $l = 1, 2, \ldots, n-2$ and $k = n-1, n-2, \ldots, l+1$, do steps
   (i)–(iv) (assume $n > 2$):
   (i) Find $Q \in H^m_{n-2}(k)$ to zero $a_{k+1,k}$. Apply: $A := QA$, $B := QB$.
   (ii) Find $V \in H^m_{n-2}(k)$ to zero $b_{k+1,k}$. Apply: $B := BV$, $D := V^TD$.
   (iii) Find $U \in H^m_{n-2}(k)$ to zero $d_{k+1,k}$. Apply: $D := DU$, $C := CU$.
   (iv) Find $Z \in H^m_{n-2}(k)$ to zero $c_{k+1,k}$. Apply: $C := ZC$, $A := AZ^T$.
Assuming $m = n$, this reduction requires about $14n^3$ multiplies, $14n^3$ adds 
and $6n^2$ square roots. This is about $2\frac{1}{3}$ times the work involved in the $QZ$ initial 
reduction and about $9$ times the work involved in the comparable reduction of a 
single matrix as in $QR$.
We conclude this section with the remark that as far as subsequent computations 
are concerned, we are left with an equivalent square problem. To see this, 
partition our reduced matrices $A$, $B$, $C$ and $D$ as follows:
\[
\begin{bmatrix}
A_1 & A_2 \\
B_1 & B_2
\end{bmatrix}, \quad \begin{bmatrix}
C_1 \\
0
\end{bmatrix}, \quad \begin{bmatrix}
D_1 \\
0
\end{bmatrix},
\]
where $A_1$, $B_1$, $C_1$ and $D_1$ are all in $R^{n \times n}$. Clearly, $A^TC = A_1C_1$ and $B^TD = B_1D_1$.

6. The $VZ$ Iteration. We now turn our attention to the iterative portion of the 
algorithm having reduced the four matrices to the condensed form of the previous 
section. Let $A$, $B$, $C$ and $D$ denote these reduced matrices. The iteration we are 
about to describe has a simple task—find orthogonal matrices $Q$, $U$, $V$ and $Z$ so that
\[
QAZ = 0, \quad OBV = 0, \quad Z^TCU = 0, \quad V^TDU = 0
\]
and
\[
QAZ \text{ is somehow "more upper triangular" than } A.
\]
The objective (6.2) is purposely imprecise, but together with (6.1), it does inform us that the general idea is to drive the subdiagonal elements in \( A \) to zero while preserving the triangularity in the other three matrices.

Before outlining how this is accomplished, we must mention how the \( \text{AC} - \lambda BD \) problem deflates. Prior to a VZ iteration, the subdiagonal of \( A \) and the diagonals of \( B, C \) and \( D \) are checked for small elements. If any of these elements are negligible, then the problem can be decoupled into two smaller subproblems. This is obvious if any of the subdiagonal elements in \( A \) are small. Less obvious is the fact that if any of the \( b_{nn} \) or \( d_{nn} \) are small, then we can still effect a decoupling. This is accomplished by some straightforward "zero chasing," the details of which we shall omit.

In checking matrix entries for smallness, we have four separate criteria: \( \epsilon_A = \epsilon \|A\|, \epsilon_B = \epsilon \|B\|, \epsilon_C = \epsilon \|C\| \) and \( \epsilon_D = \epsilon \|D\| \). An element \( x_{ij} \) of a matrix \( X \) is considered negligible if \( |x_{ij}| \leq \epsilon_X \). Thus, when looking for small elements, candidates are judged relative to the matrix from which they come.

We now describe the details of a VZ iteration, and henceforth assume that \( |a_{ii-1}| > \epsilon_A, |b_{ii}| > \epsilon_B, |c_{ii}| > \epsilon_C \) and \( |d_{ii}| > \epsilon_D \). This will enable us to form the matrix \( F = AC(BD)^{-1} \) in the theoretical discussion later. For clarity, we shall work a 6x6 example.

Let \( Q_i \) be an arbitrary orthogonal matrix in \( H_3(1) \). We shall show later how \( Q_i \) is picked so that the goal (6.2) is realized. Applying \( Q_i \) to \( A \) and \( B \) gives

\[
A := Q_i A = \begin{bmatrix}
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \\
+ & \times & \times & \times & \times & \\
0 & 0 & \times & \times & \times & \times \\
0 & 0 & 0 & \times & \times & \\
0 & 0 & 0 & 0 & \times & \\
\end{bmatrix}
\]

\[
B := Q_i B = \begin{bmatrix}
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \\
+ & \times & \times & \times & \times & \\
0 & 0 & 0 & \times & \times & \times \\
0 & 0 & 0 & 0 & \times & \times \\
0 & 0 & 0 & 0 & 0 & \times \\
\end{bmatrix}
\]

while the arrays \( C \) and \( D \) remain unchanged. With an eye to restoring the system to condensed form, we

(a) find \( V_i \in H_3(1) \) to zero \( b_{31} \) and \( b_{32} \); apply: \( B := BV_i, D := V_i^T D \);
(b) find \( V_i \in H_3(1) \) to zero \( b_{31} \); apply: \( B := BV_i, D := (V_i^T)^T D \).

The arrays \( A \) and \( C \) are left alone by these operations while \( B \) and \( D \) assume the following forms:

\[
B = \begin{bmatrix}
\times & \times & \times & \times & \times & \times \\
0 & \times & \times & \times & \times & \times \\
0 & 0 & \times & \times & \times & \times \\
0 & 0 & 0 & \times & \times & \times \\
0 & 0 & 0 & 0 & \times & \times \\
0 & 0 & 0 & 0 & 0 & \times \\
\end{bmatrix}
\]

\[
D = \begin{bmatrix}
\times & \times & \times & \times & \times & \times \\
0 & \times & \times & \times & \times & \times \\
0 & 0 & \times & \times & \times & \times \\
0 & 0 & 0 & \times & \times & \times \\
0 & 0 & 0 & 0 & \times & \times \\
0 & 0 & 0 & 0 & 0 & \times \\
\end{bmatrix}
\]
Next, we
(a) find $U_1 \in H_5^3(3)$ to zero $d_{31}$ and $d_{32}$; Apply: $D := DU_1$, $C := CU_1$;
(b) find $U'_1 \in H_5^3(3)$ to zero $d_{21}$; Apply: $D := DU'_1$, $C := CU'_1$.

The arrays $A$ and $B$ are untouched, while $C$ and $D$ take the following shapes:

\[
\begin{array}{cccccccc}
  x & x & x & x & x & & x & x & x & x & x \\
  0 & x & x & x & x & & + & x & x & x & x \\
  0 & 0 & 0 & x & x & & C = & + & + & x & x & x \\
  0 & 0 & 0 & 0 & x & & 0 & 0 & 0 & x & x & x \\
  0 & 0 & 0 & 0 & 0 & x & & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

$VD = C$.

Now we chase the unwanted nonzero elements in $C$ to the array $A$:
(a) Find $Z_1 \in H_5^2(1)$ to zero $c_{21}$ and $c_{31}$. Apply: $C := Z_1 C$, $A := \mathcal{A} Z_1^T$.
(b) Find $Z'_1 \in H_5^2(2)$ to zero $c_{32}$. Apply: $C := Z'_1 C$, $A := (\mathcal{A} Z'_1)^T$.

$C$ and $A$ now have the form

\[
\begin{array}{cccccccc}
  x & x & x & x & x & & x & x & x & x & x \\
  0 & x & x & x & x & & x & x & x & x & x \\
  0 & 0 & 0 & x & x & & + & x & x & x & x \\
  0 & 0 & 0 & 0 & x & & 0 & 0 & 0 & x & x & x \\
  0 & 0 & 0 & 0 & 0 & x & & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

$C = \mathcal{A}$.

while the arrays $B$ and $D$ are unchanged. As a final step in the cycle, we find a $Q_2 \in H_5^2(2)$ to zero $a_{31}$ and $a_{41}$. Applying $Q_2$ to $A$ and $B$ gives

\[
\begin{array}{cccccccc}
  x & x & x & x & x & & x & x & x & x & x \\
  x & x & x & x & x & & 0 & x & x & x & x \\
  0 & x & x & x & x & & 0 & + & x & x & x \\
  0 & + & x & x & x & & B = & + & + & x & x & x \\
  0 & 0 & 0 & x & x & & 0 & 0 & 0 & 0 & x & x \\
  0 & 0 & 0 & 0 & x & & 0 & 0 & 0 & 0 & 0 & x \\
\end{array}
\]

$A = Q_2 A Q_2^T$.

This fully illustrates one cycle within a $VZ$ iteration. Just as the first cycle (depicted above) involved rows and columns 1, 2 and 3, so will the next cycle involve rows and columns 2, 3 and 4. In this way, the nonzero "+" quantities are chased from $A$ to $B$ to $D$ to $C$ to $A$, etc. The process terminates with the four matrices restored to condensed form, and we summarize this for general $n$:

1. For $k = 1, 2, \cdots, n-2$ do steps (a)-(g):

   (a) if $k > 1$, find $Q_k \in H_5^2(k)$ to zero $a_{k+1,k-1}$ and $a_{k+2,k-1}$.
A GENERALIZED MATRIX EIGENVALUE ALGORITHM

If \( k = 1 \), find \( Q_1 \in H_2(k) \) in accordance with shift calculations (see below). In either case, apply: \( A := Q_k A, B := Q_k B \).

(b) Find \( V_k \in H_2(k) \) to zero \( b_{k+2,k} \) and \( b_{k+2,k+1} \). Apply: \( B := BV_k, D := V_k^T D \).

(c) Find \( V_k \in H_2(k) \) to zero \( b_{k+1,k} \). Apply: \( B := BV_k, D := (V_k^T)D \).

(d) Find \( U_k \in H_2(k) \) to zero \( d_{k+2,k} \) and \( d_{k+2,k+1} \). Apply: \( D := DU_k, C := CU_k \).

(e) Find \( U_k \in H_2(k) \) to zero \( d_{k+1,k} \). Apply: \( D := DU_k, C := CU_k \).

(f) Find \( Z_k \in H_2(k) \) to zero \( c_{k+1,k} \) and \( c_{k+2,k} \). Apply: \( C := Z_k C, A := A(Z_k)^T \).

(g) Find \( Z_k \in H_2(k+1) \) to zero \( c_{k+2,k+1} \). Apply: \( C := Z_k C, A := A(Z_k)^T \).

We are now only four plane rotations away from condensed form.

2. Do steps (h)-(k):

(h) Find \( Q_{n-1} \in H_2(n-1) \) to zero \( a_{n-1} \). Apply: \( A := Q_{n-1} A, B := Q_{n-1}^T B \).

(i) Find \( V_{n-1} \in H_2(n-1) \) to zero \( b_{n,n-1} \). Apply: \( B := BV_{n-1}, D := (V_{n-1}^T)^T \).

(j) Find \( U_{n-1} \in H_2(n-1) \) to zero \( d_{n,n-1} \). Apply: \( D := DU_{n-1}, C := CU_{n-1} \).

(k) Find \( Z_{n-1} \in H_2(n-1) \) to zero \( c_{n,n-1} \). Apply: \( C := Z_{n-1} C, A := A(Z_{n-1})^T \).

It can be verified that the \( A, B, C \) and \( D \) which emerge from this process are in condensed form (5.1). The volume of work entailed in a \( VZ \) iteration is approximately \( 29n^2 \) multiplies, \( 29n^2 \) adds and \( 2n \) square roots. This is roughly \( 2i \) times the work involved in a \( OZ \) iteration and about \( 6 \) times the work present in a \( QR \) (double implicit shift) iteration.

It remains to show how \( Q_k \) can be selected such that the goal (6.2) is realized. To this end, let \( k_1 \) and \( k_2 \) be the roots of the quadratic

\[
\det \left[ \begin{pmatrix} f_{n-1} & f_{n} \\ f_{n} & f_{n+1} \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right] = 0,
\]

where \( r = n - 1 \) and \( F = (f_n) \) is the upper Hessenberg matrix \( AC(BD)^{-1} \). Let \( v \in R^n \) be the first column of the real matrix \((F - k_1 I_n)(F - k_2 I_n)\). It can be shown that \( v \) has the form

\[
v^T = (x, y, z, 0, \ldots, 0),
\]

and that, most importantly, we need only selected portions of \((BD)^{-1}\) to compute this "zeroth column". We now choose \( Q_1 \in H_2(1) \) such that \( Q_1 v \) is a scalar multiple of \( e_1 \), the first column of \( I_n \). Define the orthogonal matrix \( Q \) by \( Q = Q_{n-1}Q_{n-2} \cdots Q_1 \). We can conclude from the construction of the \( Q_k \)'s in (a) and (h) above that

\[
e_1^T Q = e_1^T Q_1, \text{(the first rows of } Q \text{ and } Q_1 \text{ are equal)}.
\]

Moreover, if we let \( U, V \) and \( Z \) to be the accumulation of the other three orthogonal transformations, then

\[
QFQ^T = QAC(BD)^{-1}Q^T
\]

\[
= (QAZ)(Z^T CU)((OBV)(V^T DU))^{-1}
\]

\[
= \nabla \nabla [\nabla \nabla]^{-1} = \nabla.
\]
Since \( k_1 \) and \( k_2 \) are the eigenvalues of the lower \( 2 \times 2 \) portion of \( F \), we see from (6.3), (6.4) and the Francis theorem [8, p. 352] that \( Q \) is precisely the matrix one would obtain by doing a double implicit QR iteration on the single matrix \( F \). The implication of this is straightforward. Let \( A_k, B_k, C_k \) and \( D_k \) denote the matrices \( A, B, C \) and \( D \) after the \( k \)th VZ iteration. If we define \( F_k \) by \( F_k = A_k C_k (B_k D_k)^{-1} \), then the above tells us that VZ acting upon the \( A_k, B_k, C_k \) and \( D_k \) is tantamount to QR acting upon the \( F_k \). By what we know about the QR algorithm’s convergence (see [8]), we see that the matrices \( F_k \) are tending to quasi-triangular form. (A quasi-triangular matrix is an upper Hessenberg matrix with no two adjacent subdiagonal elements nonzero.) But the constant, full rank triangularity of \( C_k (B_k D_k)^{-1} \) leads us to conclude from the definition of \( F_k \) that the \( A_k \) are themselves tending to quasi-triangular form. We thus find that after enough iterations the original problem deflates into a collection of \( 1 \times 1 \) and \( 2 \times 2 \) subproblems by virtue of our setting small elements to zero as mentioned earlier in this section. We normally find that \( A \) is reduced to quasi-triangular form in about \( 1.3n \) iterations. This is consistent with the rate of convergence in the double implicit shift QR algorithm.

A word is in order about the errors incurred in the algorithm thus far. Let \( \tilde{A}, \tilde{B}, \tilde{C} \) and \( \tilde{D} \) denote the original matrices \( A, B, C \) and \( D \) after they have been transformed into quasi-triangular/triangular form. Because of the sole reliance upon orthogonal transformations, we know from Wilkinson [8] that \( \tilde{A} = \tilde{Q}(A + E_A)\tilde{Z}, \tilde{B} = \tilde{Q}(B + E_B)\tilde{V}, \tilde{C} = \tilde{Z}^T(C + E_C)\tilde{U}, \) and \( \tilde{D} = \tilde{V}^T(D + E_D)\tilde{L} \) where \( Q, U, V \) and \( Z \) are exactly orthogonal and \( E_A, E_B, E_C \) and \( E_D \) are matrices whose norms are of order \( \| e \| \| A \|, \| e \| \| B \|, \| e \| \| C \| \) and \( \| e \| \| D \| \), respectively. In solving the \( 2 \times 2 \) subproblems, great care must be exercised in order to preserve this kind of desirable perturbation result.

Finally, we remark that the generalized eigenvectors of the reduced problem can be obtained by a back substitution process. By applying the accumulated \( U \)'s we can then get the eigenvectors to the initial problem.

7. An application of the VZ algorithm to the generalized singular value problem. We tested the VZ algorithm on the generalized singular value problem \( A^T Ax = \mu^2 B^T Bx \), where

\[
A = \begin{bmatrix}
1 & 2 & 1 & 2 & -3 & -3 \\
7 & 1 & -1 & 1 & -4 & -4 \\
-4 & 4 & 2 & 5 & -4 & -3 \\
-6 & -5 & -4 & 8 & 3 & 4 \\
1 & -2 & -6 & 2 & -2 & 7 \\
-9 & 6 & 5 & -4 & -3 & 5 \\
6 & -9 & 1 & -1 & 2 & 1 \\
5 & 8 & -4 & 3 & -6 & -6 \\
-2 & 1 & 3 & -2 & 3 & -3 \\
-9 & -7 & 2 & 5 & 1 & 8
\end{bmatrix}
\]
A GENERALIZED MATRIX EIGENVALUE ALGORITHM

\[
B = \begin{bmatrix}
4 & 1 & -3 & -6 & 5 & -1 \\
3 & -4 & -6 & 1 & 4 & 2 \\
-5 & 7 & 8 & 3 & -9 & -4 \\
6 & -6 & 4 & -5 & 2 & -1 \\
1 & 3 & -9 & -7 & 5 & 7 \\
2 & -4 & 4 & 2 & -6 & 2 \\
-7 & -5 & 1 & 4 & 2 & 5 \\
-8 & 2 & -7 & 5 & 7 & 1 \\
-9 & -5 & -3 & 1 & 8 & 8 \\
6 & 4 & 5 & -6 & -1 & -8
\end{bmatrix}
\]

Notice that the transpose of the vector \((1, 1, 1, 1, 1, 1)\) is annihilated by both matrices. Thus, \(A^TB - \mu^2B^TB\) is singular for all scalars \(\mu\). To see how the VZ algorithm detected this, we tabulate the quantities \(\alpha_i, \beta_i, \gamma_i\) and \(\delta_i\) which are, respectively, the diagonal elements of the upper triangular \(QA^TZ, Z^TAU, QB^TV,\) and \(V^TB\).

<table>
<thead>
<tr>
<th>(\alpha_i)</th>
<th>(\gamma_i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>18. 4408 9224 2101 4200</td>
<td>13. 8919 7585 5565 7000</td>
</tr>
<tr>
<td>14. 2621 7822 4083 0700</td>
<td>10. 9609 5294 3934 9100</td>
</tr>
<tr>
<td>15. 4007 5100 8590 0000</td>
<td>8. 9356 6755 8123 6070</td>
</tr>
<tr>
<td>6. 9798 0352 5926 1540</td>
<td>7. 0220 2359 9356 9530</td>
</tr>
<tr>
<td>14. 0941 9284 0413 0300</td>
<td>17. 0261 3202 3272 9700</td>
</tr>
<tr>
<td>0. 0000 0000 0000 00009</td>
<td>0. 0000 0000 0000 0012</td>
</tr>
<tr>
<td>(\beta_i)</td>
<td>(\delta_i)</td>
</tr>
<tr>
<td>9. 4162 8385 9280 6950</td>
<td>7. 0935 1678 2628 8390</td>
</tr>
<tr>
<td>18. 8916 4327 8751 4100</td>
<td>14. 5188 4906 7692 8600</td>
</tr>
<tr>
<td>14. 0796 5698 4272 1200</td>
<td>8. 1691 5361 0971 4360</td>
</tr>
<tr>
<td>22. 3549 3610 3468 4500</td>
<td>22. 4901 5878 4526 1100</td>
</tr>
<tr>
<td>11. 3857 9924 2506 4700</td>
<td>13. 7543 2586 2278 0800</td>
</tr>
<tr>
<td>0. 0000 0000 0000 0007</td>
<td>0. 0000 0000 0000 0015</td>
</tr>
</tbody>
</table>

Our computed generalized singular values are then given by

\[
\mu_i = \left(\frac{\alpha_i\gamma_i}{\beta_i\delta_i}\right)^{1/2},
\]

\[
\begin{align*}
1.9584 & 0445 3145 9270 \\
0.7549 & 4640 7448 0300 \\
1.0938 & 3027 7119 8620 \\
0.3122 & 2650 1672 7363 \\
1.2378 & 7470 1654 2610 \\
1.0638 & 8743 0396 0920
\end{align*}
\]
(These results were obtained using an IBM 360/67 with REAL*8 arithmetic.) Observe that $\mu_e$ is determined as the quotient of rounding errors, but that this is not at all obvious if we examine the set of $\mu_i$ alone. Hence, we see the importance of the "QZ philosophy" which in our case advises that we return the $\alpha_i, \beta_i, \gamma_i$ and $\delta_i$ to the user so that he can compute and interpret the quotients $\alpha_i\gamma_i/\beta_i\delta_i$.

As we remarked, the matrices $A$ and $B$ above have the property that $A^TA - \mu^2B^TB$ is singular for all scalars $\mu$. Hence, we expect that regardless of $\mu$,

\[
\prod_i (\alpha_i\gamma_i - \mu_i^2\beta_i\delta_i) = \det[(OAZ)^T(Z^TAU) - \mu^2(QU)^TV^TBU)]
= \det(QU) \det[A^TA - \mu^2B^TB]
= 0,
\]

which implies $\alpha_k\gamma_k = \beta_k\delta_k = 0$ for some $k$. Modulo rounding errors, we find this to be the case of $k = 6$.

While the VZ algorithm can successfully solve the generalized singular value problem, even in pathological situations as above, it does have the drawback of being very inefficient. This is because it takes no advantage of the symmetry inherent in the generalized singular value problem. Other algorithms exist which do take advantage of symmetry but, as usual, at some expense [7]. Some of these routines break down when $B$ is rank deficient, while others experience difficulty only when $A$ and $B$ have nontrivially intersecting nullspaces. Despite this, their economy over VZ makes their use more attractive than that of the general algorithm presented in this paper.

**Acknowledgments.** This work was part of the author's Ph.D. thesis written at the University of Michigan under Prof. Cleve B. Moler. The author wishes to thank Professor Moler for his assistance.

**REFERENCES**


