A TRUNCATED RQ-ITERATION FOR LARGE SCALE EIGENVALUE CALCULATIONS[*]

D. C. SORENSEN[†] AND C. YANG[‡]

Abstract. We introduce a new Krylov subspace iteration for large scale eigenvalue problems that is able to accelerate the convergence through an inexact (iterative) solution to a shift-invert equation. The method also takes full advantage of exact solutions when when they can be obtained with sparse direct method. We call this new iteration the Truncated RQ iteration (TRQ). It is based upon a recursion that develops in the leading k columns of the implicitly shifted RQ iteration for dense matrices. Inverse-iteration-like convergence to a partial Schur decomposition occurs in the leading k columns of the updated basis vectors and Hessenberg matrices. The TRQ iteration is competitive with the Rational Krylov Method of Ruhe when the shift-invert equations can be solved directly and with the Jacobi-Davidson Method of Sleijpen and Van der Vorst when these equations are solved inexactly with a preconditioned iterative method. The TRQ iteration is related to both of these but is derived directly from the RQ iteration and thus inherits the convergence properties of that method. Existing RQ deflation strategies may be employed directly in the TRQ iteration.

Key words. Krylov methods, Arnoldi method, Lanczos method, eigenvalues, deflation, preconditioning, restarting

AMS subject classifications. Primary 65F15, Secondary 65G05

1. Introduction.

Recently, there have been a number of research developments in the numerical solution of large scale eigenvalue problems [21], [11], [17], [6], [19], [16], [13], [1], [5]. The state of the art has advanced considerably and general purpose numerical software is emerging for the nonsymmetric problem [8], [4], [12], [3], [18], [10]. The development of this new general purpose software for the nonsymmetric problem is a welcomed advance. However, the methods in these packages are not able to effectively utilize a preconditioned iterative solver to implement a shift and invert spectral transformation to accelerate convergence. They all require highly accurate solutions to the shift-invert equations and the cost of producing such accuracy with an iterative method is generally prohibitive. In this paper, we introduce a new iteration for large scale problems that is in the same spirit as the Implicitly Restarted Arnoldi Method used in ARPACK [21], [12]. However, this new method is very amenable to acceleration of convergence with inexact (iterative) solutions to the shift-invert equations. Moreover, the algorithm introduced here can take full advantage of exact solutions when they can be obtained with a sparse direct method.

We call this new iteration the Truncated RQ iteration (TRQ). It is based upon a recursion that develops in the leading k columns of the implicitly shifted RQ iteration for dense matrices. This iteration is analogous to the well known QR iteration, but it implicitly factors the shifted Hessenberg matrix into an RQ factorization (triangular times orthogonal) and then multiplies the factors in reverse order rather than using a QR factorization for this iteration. The main advantage in the large scale setting is that inverse-iteration-like convergence occurs in the leading column of the updated basis matrix. Thus, eigenvalues rapidly converge in the leading principal submatrix of

[*]This work was supported in part by NSF cooperative agreement CCR-9120008, and by ARPA contract number DAAL03-91-C-0047 (administered by the U.S. Army Research Office).

[†]Department of Computational and Applied Mathematics, Rice University, Houston, TX 77005-1892, (`sorensen@caam.rice.edu`).

[‡]Department of Computational and Applied Mathematics, Rice University, Houston, TX 77005-1892, (`chao@caam.rice.edu`).

the iterated Hessenberg matrix. A partial Schur form rapidly emerges in the leading portion of the factorization. The leading principal submatrix of the iterated Hessenberg matrix becomes upper triangular with the desired eigenvalues appearing as diagonal elements.

A k-step TRQ iteration is derived by developing a set of equations that define the $k+1$-st column of the updated set of basis vectors and the updated projected Hessenberg matrix that would occur if a full RQ iteration were carried out. The resulting equations have a great deal in common with the update equation that defines the Rational Krylov Method of Ruhe [16], and also with the projected correction equation that defines the Jacobi-Davidson Method of Sleijpen and Van der Vorst. [19]. The TRQ iteration is comparable to and quite competitive with the Rational Krylov Method when it is possible to factor and solve the shift-invert equations directly. With restarting, it is possible to define an inexact TRQ iteration that compares very favorably with the Jacobi-Davidson Method. The TRQ iterations developed here are derived directly from the RQ iteration and may take advantage of all that is known about deflation strategies in the dense case. Moreover, the convergence behavior follows directly from the convergence properties of the RQ iteration.

In Section 2, we derive the TRQ equations that will define the TRQ iteration and investigate the existence and uniqueness of the solution to these equations. We also introduce the formal specification of the TRQ iteration. In Section 3 we turn to some implementation issues that arise when a sparse direct solution to the shift-invert equation is possible. We show that the Arnoldi relation existing in the leading k columns may be used to greatly reduce the amount of computation required to solve the TRQ equations. In Section 3 we also discuss the selection of shifts to be used in the TRQ iteration when factorizations are only allowed intermittently. Also, deflation schemes are introduced. In Section 4 we give several numerical examples to illustrate the convergence behavior of the TRQ iteration. We demonstrate that the convergence is cubic on symmetric problems and quadratic on nonsymmetric problems when a factorization is done at each step. We also show that the more practical alternative of factoring intermittently is quite competitive with the Rational Krylov Method employing the same type of shift strategy. A comparison is made with Implicitly Restarted Arnoldi (IRA) in the case that only one factorization is allowed, and we observe that IRA is more efficient than TRQ in this case.

In Section 5, we develop the inexact TRQ iteration with restarting. Restarting is required to maintain an Arnoldi factorization and hence a Krylov relationship amongst the columns of the k-step factorization. As convergence takes place, standard deflation techniques are employed to lock converged Schur vectors and orthogonalization against these converged vectors takes place naturally through the Arnoldi process. In some sense, this process is closely related to inverse iteration with Wielandt deflation [23, pp. 596], [17, pp. 117]. We illustrate an apparent numerical advantage of placing the inverse iteration within the context of the TRQ iteration and show some explicit comparisons with deflated inverse iteration indicating clear superiority of the TRQ scheme. Of course, the purpose of introducing possibly inexact solutions to the shift-invert equations is to provide for the use of preconditioned iterative solution techniques on these equations. We show numerical experiments indicating very favorable comparison with the Jacobi-Davidson Method using the same iterative method for solving the update equations in both schemes. Moreover, we give some preliminary evidence that a shifted form of a standard preconditioner for the original matrix is a satisfactory preconditioner for the update equations. Constructing a modified preconditioner for the projected update equations (as required in Jacobi-Davidson) does not seem to be necessary with inexact TRQ.

Throughout this paper, capital and lower case Latin letters denote matrices and vectors respectively, while lower case Greek letters denote scalars. The j-th canonical basis vector is denoted by e_j. The Euclidean norm is used exclusively and is denoted by $\|\cdot\|$. The transpose of a matrix A is denoted by A^T and conjugate transpose by A^H. Upper Hessenberg matrices will appear frequently and are usually denoted by the letter H. The subdiagonal elements of such Hessenberg matrices play a special role in our algorithms. The j-th subdiagonal element (i.e. the $(j+1, j)$-st element) of an upper Hessenberg matrix H will be denoted by β_j. The conjugate of a complex number α is denoted by $\bar{\alpha}$.

2. Truncating the RQ Iteration . The implicitly shifted QR iteration is generally the method of choice for the computation of all the eigenvalues and eigenvectors of a square matrix A. Practical implementation of the algorithm begins with a complete reduction of A to upper Hessenberg form:

$$AV = VH$$

with $V^H V = I$ and H upper Hessenberg. The QR iteration is then applied to H to produce a sequence of orthogonal similarity transformations

$$H^{(j+1)} \leftarrow (Q^{(j)})^H H^{(j)} Q^{(j)}, \quad V^{(j+1)} \leftarrow V^{(j)} Q^{(j)}$$

with $H^{(1)} \equiv H$, $V^{(1)} \equiv V$ and $Q^{(j)}$ implicitly constructed and applied through a "bulge chase" process that is mathematically equivalent to obtaining $Q^{(j)}$ through the QR-factorization $Q^{(j)} R^{(j)} = H^{(j)} - \mu_j I$, $j = 1, 2, \cdots$, where $\{\mu_j\}$ is a set of shifts selected as the algorithm proceeds. We use $v_i^{(j)}$ to denote the i-th column of $V^{(j)}$, and $\rho_{ii}^{(j)}$ to denote the (i, i)-th entry of $R^{(j)}$. It is straightforward to show that $H^{(j)}$ remains upper Hessenberg throughout and that

$$v_1^{(j+1)} \rho_{11}^{(j)} = (A - \mu_j I) v_1^{(j)} \quad \text{and} \quad (A - \mu_j I)^H v_n^{(j+1)} = v_n^{(j)} \bar{\rho}_{nn}^{(j)}.$$

Hence, the last column is an inverse iteration sequence and the first column is a power method or polynomial iteration. The Implicitly Restarted Arnoldi Method provides a means to truncate this QR iteration and take advantage of the shifted-power-method-like convergence properties of the leading k columns of the iterated basis $V^{(j)}$ without computing the full QR factorizations. The relations between $v_i^{(j+1)}$ and $v_i^{(j)}$ for $i = 1, 2, \cdots, k$ on successive iterations are preserved in this truncated IRA iteration as if the full QR iteration had been carried out. Appropriate shift selection will force desired eigenvalues and corresponding eigenvectors to emerge in the leading portion of the factorization as the iteration proceeds.

Some advantages of the IRA approach are: (i) the number of basis vectors stored is pre-determined and fixed so that orthogonality of the Arnoldi basis vectors may be enforced numerically, and (ii) the iteration proceeds without having to compute a matrix factorization. In many situations this iteration is successful but it can be slow to converge or fail when the desired portion of the spectrum does not have a favorable distribution with respect to the entire spectrum of A. It would be very desirable to devise a scheme that could take advantage of the inverse iteration properties of the QR iteration instead of the power iteration properties.

```
Algorithm 1: Implicitly Shifted $RQ$-iteration

Input: $(A, V, H)$ with $AV = VH$, $V^H V = I$, and $H$ is upper
       Hessenberg.
Output: $(V, H)$ such that $AV = VH, V^H V = I$ and $H$ is upper triangular.

1. for $j = 1, 2, 3, \ldots$ until convergence,
   1.1. Select a shift $\mu \leftarrow \mu_j$;
   1.2. Factor $H - \mu I = RQ$;
   1.3. $H \leftarrow QHQ^H$ ; $V \leftarrow VQ^H$;
2. end;
```

FIG. 2.1. *Implicitly Shifted RQ-iteration.*

An alternative to the Implicitly Shifted QR iteration is the Implicitly Shifted RQ iteration. Again, the iteration begins with a reduction to Hessenberg form and then the iteration demonstrated in Figure 2.1 is applied.

It is easily shown that

$$(A - \mu_j I) v_1^{(j+1)} = v_1^{(j)} \rho_{11}^{(j)}.$$

Thus, the sequence $v_1^{(j)}$ in the first column is an inverse iteration sequence and one would expect very rapid convergence of leading columns of $V^{(j)}$ to Schur vectors of A.

In the large scale setting it is generally impossible to carry out the full iteration involving $n \times n$ orthogonal similarity transformations. It would be desirable to truncate this update procedure after k steps to maintain and update only the leading portion of the factorizations occurring in this sequence. This truncation is obtained from a set of defining equations that emerge during the partial completion of an RQ step. To derive these relations, partition $V = (V_k, \hat{V})$ where V_k denotes the leading k columns of V and let

$$H = \begin{pmatrix} H_k & M \\ \beta_k e_1 e_k^T & \hat{H} \end{pmatrix}$$

be partitioned conformably so that

$$A(V_k, \hat{V}) = (V_k, \hat{V}) \begin{pmatrix} H_k & M \\ \beta_k e_1 e_k^T & \hat{H} \end{pmatrix}. \tag{2.1}$$

Now, for a given shift μ, partially factor $H - \mu I$ to obtain

$$H - \mu I = \begin{pmatrix} H_k - \mu I_k & \hat{M} \\ \beta_k e_1 e_k^T & \hat{R} \end{pmatrix} \begin{pmatrix} I_k & 0 \\ 0 & \hat{Q} \end{pmatrix}$$

where $\hat{H} - \mu I = \hat{R}\hat{Q}$. Then

$$(A - \mu I)(V_k, \hat{V}\hat{Q}^H) = (V_k, \hat{V}) \begin{pmatrix} H_k - \mu I_k & \hat{M} \\ \beta_k e_1 e_k^T & \hat{R} \end{pmatrix} \tag{2.2}$$

If Givens transformations were being used, for example, then to complete the RQ factorization in (2.2), one would continue applying Givens rotations from the right using each rotation to annihilate a subdiagonal element. However, at this point of the factorization, there is a set of equations that uniquely determines the first column v_+ of the matrix $\hat{V}\hat{Q}^H$. If these equations can be formulated and solved, then the leading portion of this iteration may be obtained using just the leading $k+1$ columns $(V_k, \hat{V}e_1)$ and the leading k columns of the Hessenberg matrix H. The remaining $n-k-1$ columns of V and of H need never be formed or factored. To formulate the defining relations, equate the leading $k+1$ columns on both sides of equation (2.2) to obtain

$$(A-\mu I)(V_k, v_+) = (V_k, v)\begin{pmatrix} H_k - \mu I_k & h \\ \beta_k e_k^T & \alpha \end{pmatrix}$$

where $v = \hat{V}e_1$, $v_+ = \hat{V}\hat{Q}^H e_1$, $h = \hat{M}e_1$, and $\alpha = e_1^T \hat{R}e_1$. From this relationship, it follows that v_+ must satisfy

$$(A-\mu I)v_+ = V_k h + v\alpha, \tag{2.3}$$

with $V_k^H v_+ = 0$ and $\|v_+\| = 1$ since the columns of (V_k, v_+) must be orthonormal.

These conditions may be expressed succinctly through the *TRQ equations*

$$\begin{pmatrix} A-\mu I & V_k \\ V_k^H & 0 \end{pmatrix}\begin{pmatrix} v_+ \\ -h \end{pmatrix} = \begin{pmatrix} v\alpha \\ 0 \end{pmatrix}, \quad \|v_+\| = 1. \tag{2.4}$$

In addition to these *TRQ* equations, we note that the first k columns on both sides of (2.2) are in a k-step Arnoldi relationship

$$(A-\mu I)V_k = V_k(H_k - \mu I_k) + f_k e_k^T \tag{2.5}$$

with $f_k = v\beta_k$.

The algorithm we shall develop depends upon the determination of v_+, h, and α directly from equation (2.4) rather than from the *RQ* factorization procedure. The fact that the RQ factorization exists assures that a solution to (2.4) exists even when the bordered matrix in (2.4) is singular.

The following lemmas characterize how singularity can occur in these equations. Moreover, we prove that the solution to (2.4) is unique even when the bordered matrix is singular. In the next section we show that the singular case in (2.4) is benign and easily dealt with numerically.

Lemma 2.1. *Assume $A-\mu I$ is nonsingular (i.e., that μ is not an eigenvalue of A) and that equations (2.4) and (2.5) hold as a result of the partial RQ-factorization described by (2.2). Then the bordered matrix*

$$B \equiv \begin{pmatrix} A-\mu I & V_k \\ V_k^H & 0 \end{pmatrix} \tag{2.6}$$

is nonsingular if and only if $V_k^H(A-\mu I)^{-1}V_k$ is nonsingular. Moreover, if $V_k^H(A-\mu I)^{-1}V_k$ is singular and z is any nonzero vector such that $V_k^H(A-\mu I)^{-1}V_k z = 0$, then $w = -(A-\mu I)^{-1}V_k z$ is nonzero, and $v_+ = \frac{w}{\|w\|}$, $h = -\frac{z}{\|w\|}$, and $\alpha = 0$ satisfy the TRQ equations.

Proof. Since the RQ-factorization $\hat{R}\hat{Q} = \hat{H} - \mu I$ always exists, it follows that (2.4) must hold in any case. The assumption that $A - \mu I$ is nonsingular provides the block factorization

$$B = \begin{pmatrix} I & 0 \\ V_k^H(A-\mu I)^{-1} & I \end{pmatrix} \begin{pmatrix} A-\mu I & V_k \\ 0 & -V_k^H(A-\mu I)^{-1}V_k \end{pmatrix}. \tag{2.7}$$

Clearly, B is nonsingular if and only if $V_k^H(A-\mu I)^{-1}V_k$ is nonsingular.

To establish the second part of the lemma, we show that the equation

$$\begin{pmatrix} A-\mu I & V_k \\ 0 & V_k^H(A-\mu I)^{-1}V_k \end{pmatrix} \begin{pmatrix} w \\ z \end{pmatrix} = \begin{pmatrix} v \\ V_k^H(A-\mu I)^{-1}v \end{pmatrix} \alpha. \tag{2.8}$$

has a nonzero solution $(w^H, z^H)^H$ with $\alpha = 0$ if and only if and only if $V_k^H(A-\mu I)^{-1}V_k$ is singular.

To prove this, suppose first that $\alpha = 0$ and $(w^H, z^H)^H$ is a nonzero solution to (2.8). Then $V_k^H(A-\mu I)^{-1}V_k$ must be singular because $A - \mu I$ is assumed to be nonsingular. On the other hand, if we assume $V_k^H(A-\mu I)^{-1}V_k$ is singular and z is a nonzero vector such that $V_k^H(A-\mu I)^{-1}V_k z = 0$, then putting $w = -(A-\mu I)^{-1}V_k z$ will provide a nonzero solution to (2.8) with $\alpha = 0$. Moreover, w must be nonzero since z is nonzero and $(A-\mu I)^{-1}V_k$ has linearly independent columns. Therefore, $v_+ = \frac{w}{\|w\|}$, $h = -\frac{z}{\|w\|}$, and $\alpha = 0$ will satisfy the TRQ equations. □

Lemma 2.1 indicates that the solution to (2.4) will be unique if and only if $V_k^H(A-\mu I)^{-1}V_k$ is either nonsingular or has a one dimensional null space. The following lemma establishes this fact and hence the uniqueness of the solution to the TRQ equations (2.4).

LEMMA 2.2. *Assume $A - \mu I$ is nonsingular and that equations (2.4) and (2.5) hold. If $G \equiv V_k^H(A-\mu I)^{-1}V_k$ is singular, then the null space of G^H is span$\{e_k\}$.*

Proof. Let $y = V_k^H(A-\mu I)^{-1}f_k$ and define $H_\mu \equiv H_k - \mu I_k$. Then

$$\begin{aligned} GH_\mu &= V_k^H(A-\mu I)^{-1}V_k H_\mu \\ &= V_k^H(A-\mu I)^{-1}[(A-\mu I)V_k - f_k e_k^T] \\ &= I_k - y e_k^T \end{aligned} \tag{2.9}$$

If G is singular, and x is any nonzero vector such that $0 = x^H G$, then (2.9) implies

$$0 = x^H G H_\mu = x^H - (x^H y)e_k^T.$$

Since $x \neq 0$ this equation implies $x^H y \neq 0$ which in turn implies that $x/(x^H y) = e_k$. Hence $e_k^T G = 0$ and the null space of G^H is *span*$\{e_k\}$. This concludes the proof. □

Finally, the following lemma indicates that exact singularity of B rarely occurs.

LEMMA 2.3. *Assume $A - \mu I$ is nonsingular and that equations (2.4) and (2.5) hold. Then $\alpha = 0$ in (2.4) and $V_k^H(A-\mu I)^{-1}V_k$ is singular if and only if the shift μ is an eigenvalue of $\hat{H}$ in equation (2.1).*

Proof. It is sufficient to show $V_k^H(A-\mu I)^{-1}V_k$ is singular if and only if the shift μ is an eigenvalue of $\hat{H}$ in equation (2.1). To this end, note that $V_k^H(A-\mu I)^{-1}V_k$ is singular if and only if $V_k^H(A-\mu I)^{-1}V_k z = 0$ for some $z \neq 0$. Since $(V_k, \hat{V})$ is unitary, any such z must satisfy

$$V_k z = (A-\mu I)\hat{V}g = (A-\mu I)(V_k, \hat{V})\begin{pmatrix} 0 \\ g \end{pmatrix}$$

for some nonzero vector g (i.e. $(A-\mu I)^{-1}V_kz$ must be in the range of $\hat{V}$). This implies

$$\begin{aligned} V_kz &= (V_k,\hat{V})\begin{pmatrix} H_k-\mu I_k & M \\ \beta_k e_1 e_k^T & \hat{H}-\mu I_{n-k}\end{pmatrix}\begin{pmatrix}0\\ g\end{pmatrix} \\ &= V_kMg+\hat{V}(\hat{H}-\mu I_{n-k})g. \end{aligned}$$

Since $(V_k,\hat{V})$ is unitary, it follows that

$$(\hat{H}-\mu I_{n-k})g=0 \qquad (2.10)$$

and since g is nonzero this implies the singularity of $(\hat{H}-\mu I_{n-k})$.

Now, suppose that there is a nonzero g that satisfies (2.10). Observe that $Mg\neq 0$ since this would imply $A-\mu I$ is singular. Hence, the argument just given may be reversed to produce a nonzero z such that $V_k^H(A-\mu I)^{-1}V_kz=0$ and lemma is proved. □

The TRQ equations may be used to develop a truncated k-step version of the Implicitly Shifted RQ iteration. If a k-step Arnoldi factorization (2.5) has been obtained then a k-step TRQ iteration may be implemented as shown in Algorithm 2 (Figure 2.2.)

```
Algorithm 2: (TRQ) Truncated RQ-iteration

Input: $(A, V_k, H_k, f_k)$ with $AV_k = V_kH_k + f_ke_k^T$, $V_k^HV_k = I$, $H_k$ upper
       Hessenberg.
Output: $(V_k, H_k)$ such that $AV_k = V_kH_k$, $V_k^HV_k = I$ and $H_k$ is upper triangular.

1. Put $\beta_k = \|f_k\|$ and put $v = f_k/\beta_k$;
2. for $j = 1, 2, 3, \ldots$ until convergence,
   2.1. Select a shift $\mu \leftarrow \mu_j$;
   2.2. Solve $\begin{pmatrix} A-\mu I & V_k \\ V_k^H & 0 \end{pmatrix}\begin{pmatrix} v_+ \\ -h \end{pmatrix} = \begin{pmatrix} v\alpha \\ 0 \end{pmatrix}$ with $\|v_+\| = 1$;
   2.3. RQ Factor $\begin{pmatrix} H_k-\mu I_k & h \\ \beta_k e_k^T & \alpha \end{pmatrix} = \begin{pmatrix} R_k & r \\ 0 & \rho \end{pmatrix}\begin{pmatrix} Q_k & q \\ \sigma e_k^T & \gamma \end{pmatrix}$;
   2.4. $V_k \leftarrow V_kQ_k^H + v_+q^H$;
   2.5. $\beta_k \leftarrow \sigma e_k^T R_k e_k$; $v \leftarrow v_k\bar{\sigma} + v_+\bar{\gamma}$;
   2.6. $H_k \leftarrow Q_kR_k + \mu I_k$;
3. end;
```

FIG. 2.2. *The Truncated RQ-iteration.*

The key idea here is to determine the $k+1$-st column v_+ of the updated matrix V and the $k+1$-st column of H that would have been produced in the RQ iteration by solving the linear system (2.4). Then, the iteration is completed through the normal RQ iteration. As eigenvalues converge, the standard deflation rules of the RQ iteration may be applied. Orthogonality of the basis vectors is explicitly maintained through accurate solution of the defining equation. Moreover, even if the accuracy of this solution is relaxed, orthogonality may be enforced explicitly through the orthogonalization scheme developed in [7]. We shall refer to this as the DGKS procedure.

Potentially, the linear solve indicated at Step 2.2 of Algorithm 2 could be provided by a straightforward block elimination scheme. However, considerable refinements to this scheme are possible due to the existing k-step Arnoldi relationship (2.5). This will be discussed in the next section.

3. Implementation Issues.

In this section, we address some practicalities associated with efficient implementation of the TRQ iteration.

3.1. Solving the TRQ Equations.

The Truncated RQ iteration described in the previous section will only be effective in the large scale setting if there is an efficient means for solving the TRQ equations. Recall that A, H_k, V_k and $f_k = v\beta_k$ are in a k-step Arnoldi relation (3.1) so that

$$(A - \mu I)V_k = V_k(H_k - \mu I_k) + f_k e_k^T. \tag{3.1}$$

Rescaling the right hand side of the system (2.4) leads to

$$\begin{pmatrix} A-\mu I & V_k \\ V_k^H & 0 \end{pmatrix} \begin{pmatrix} w \\ z \end{pmatrix} = \begin{pmatrix} f_k \\ 0 \end{pmatrix}. \tag{3.2}$$

If we put $d = (A-\mu I)^{-1} f_k$ and $y = V_k^H d$, then block Gaussian elimination leads to solving the equations

(a) $V_k^H (A-\mu I)^{-1} V_k z = y$,

(b) $(A - \mu I)w = f_k - V_k z$.

If $A-\mu I$ is nonsingular, these two equations together with equation (3.1) may be used to derive a solution to equation (3.2) with just a single linear solve. It is not necessary to solve a blocked system of k equations as the straightforward application of block Gaussian elimination described in the previous section would indicate. Moreover, this efficient solution scheme does not depend on determining the singularity of the TRQ equations (2.4) in any way. The underlying theory is developed with the following lemma.

Lemma 3.1. *Assume $A - \mu I$ is nonsingular and define $G \equiv V_k^H (A-\mu I)^{-1} V_k$ and $H_\mu \equiv (H_k - \mu I_k)$. There is a vector s such that either*

$$(I_k - H_\mu G)s \neq 0 \quad \text{or} \quad e_k^T G s \neq 0. \tag{3.3}$$

For any such s, put

$$w \equiv (I - V_k V_k^H)(A-\mu I)^{-1} V_k s.$$

Then $w \neq 0$ and a solution v_+, h, α to (2.4) is given by

$$v_+ = w/\|w\|, \quad h = (I_k - H_\mu G)s/\|w\|, \quad \alpha = -\beta_k e_k^T G s/\|w\|.$$

Proof. If $e_k^T G s = 0$ for all vectors s, then the matrix $H_\mu G$ is singular and there must be a nonzero vector s such that $(I_k - H_\mu G)s \neq 0$. Therefore, there is a k-dimensional vector s that satisfies either $\theta \equiv e_k^T G s \neq 0$ or $(I_k - H_\mu G)s \neq 0$.

For any such s, put $w \equiv (I - V_k V_k^H)(A-\mu I)^{-1} V_k s$. Observe that

$$\begin{aligned} (A-\mu I)w &= (A-\mu I)(I - V_k V_k^H)(A-\mu I)^{-1} V_k s \\ &= V_k s - (A-\mu I)V_k G s \\ &= V_k s - [V_k H_\mu + f_k e_k^T] G s \\ &= V_k (I_k - H_\mu G)s - f_k \theta. \end{aligned} \tag{3.4}$$

The conditions on s assure that the right hand side of (3.4) is nonzero. It follows that $w \neq 0$ and that

$$(A - \mu I)v_+ = V_k h + v\alpha$$

where $v_+ = w/\|w\|$, $h = (I_k - H_\mu G)s/\|w\|$, and $\alpha = -\beta_k \theta/\|w\|$. $\square$

Remark 1 Our original motivation for developing Lemma 3.1 was to handle the case when μ is an eigenvalue of H_k. A particular choice of s for this case is to put $s = q$ where $q^H H_\mu = 0$, and $q^H q = 1$. Then

$$q^H(I_k - H_\mu G)s = q^H s = q^H q = 1.$$

The conditions of Lemma 3.1 are clearly satisfied with this choice of s. However, we do not use this choice in practice.

Remark 2 The most general form of selecting a right hand side for constructing w is to take

$$w \equiv (I - V_k V_k^H)(A - \mu I)^{-1}(V_k t + f_k \eta)$$

where $s \equiv t - H_\mu e_k \eta$ is chosen to satisfy the conditions of Lemma 3.1. To see this, observe that

$$\begin{aligned} V_k t + f_k \eta &= V_k s + [V_k H_\mu + f_k e_k^T] e_k \eta \\ &= V_k s + (A - \mu I) V_k e_k \eta. \end{aligned}$$

Hence,

$$(I - V_k V_k^H)(A - \mu I)^{-1}(V_k t + f_k \eta) = (I - V_k V_k^H)(A - \mu I)^{-1} V_k s.$$

Thus, there is no mathematical reason to include the term $f_k \eta$, but the additional freedom may eventually have some numerical consequences that are not apparent at the moment. Note that when the shift μ is an eigenvalue of H_k then the combination of $t = 0$,$\eta = 1$ is prohibited because the corresponding vector s does not satisfy either of the conditions 3.3 required for constructing the solution in Lemma 3.1. The parameters t and η here are obviously related to the corresponding parameters appearing in the RKS method. It is interesting to note that the choice $t = 0$,$\eta = 1$ is also prohibited in RKS when μ is an eigenvalue of H_k.

Remark 3 An alternative to forming h as described in Lemma 3.1 is to form w as described above and normalize to get $v_+ = w/\|w\|$. Then, construct h and α using the DGKS procedure to orthogonalize the vector $(A - \mu I)v_+$ against V_k and f_k respectively. Thus

$$h \leftarrow V_k^H (A - \mu I) v_+ = V_k^H A v_+, \quad \alpha \leftarrow f_k^H (A - \mu I) v_+ / \|f_k\|.$$

Lemma 3.1 justifies Algorithm 3 to solve the TRQ equations. Once again, we remark that the DGKS procedure may be used at Steps 2,3,4 of Algorithm 3 to assure that both $V_k^H v_+ = 0$ and $(A - \mu I)v_+ = V_k h + v\alpha$ to working accuracy. For relatively small

```
Algorithm 3: Direct Solution of the TRQ Equations

Input: $(A, V_k, H_k, f_k, \mu)$ with $AV_k = V_k H_k + f_k e_k^T$, $V_k^H V_k = I$ and $V_k^H f_k = 0$.
Output: $(v_+, h, \alpha)$ such that $(A - \mu I)v_+ = V_k h + f_k \alpha$, $V_k^H v_+ = 0$ and $\|v_+\| = 1$.

1. Choose $t$ and $\eta$ and solve $(A - \mu I)w = V_k t + f_k \eta$;
2. $y \leftarrow V_k^H w$;
3. $w \leftarrow w - V_k y$;
4. $v_+ \leftarrow \frac{w}{\|w\|}$; $\alpha \leftarrow f_k^H (A - \mu I)v_+ / \|f_k\|$; $h \leftarrow V_k^H A v_+$;
```

FIG. 3.1. *Direct Solution of the TRQ Equations.*

values of k, the main computational effort is the solution of the equation $(A-\mu I)w = V_k t + f_k \eta$. As mentioned in Remark 2, there may be advantageous choices of t and η to overcome inaccuracies due to ill-conditioning when μ is very nearly an eigenvalue of A. We used $t = e_k$ and $\eta = 0$ in all of the experiments reported in Section 4. This choice seemed to perform consistently well as compared to many of the obvious choices such as taking t to be an eigenvector of H_k. Finally, it is clear that incremental re-scaling may be introduced as in inverse iteration to avoid overflow and that the scalar θ appearing in the proof of Lemma 3.1 need not be computed explicitly.

The formulation just developed is appropriate when a sparse direct factorization of $A - \mu I$ is feasible. When this is not the case we must resort to an iterative scheme. For an iterative scheme, there may be an advantage to solving the projected equation

$$(I - V_k V_k^H)(A - \mu I)(I - V_k V_k^H)\hat{w} = f_k$$

and putting

$$v_+ \leftarrow \frac{w}{\|w\|};$$

where $w = (I - V_k V_k^H)\hat{w}$. This is mathematically equivalent to solving the TRQ equations. The advantage here is that the matrix

$$(I - V_k V_k^H)(A - \mu I)(I - V_k V_k^H)$$

is most likely to be much better conditioned than $A - \mu I$ when μ is near an eigenvalue of A. A projected equation of this form plays a key role in the Jacobi-Davidson Method recently developed in [19], [20], [9]. It also provides a means for allowing inaccurate solutions and preconditioning as we shall discuss later in Section 5.

3.2. Selection of Shifts. Another important issue to be addressed in the TRQ iteration is the selection of shifts. Various options are available. They lead to different convergence behavior. We discuss only a few simple options below. The tradeoffs and comparison to other algorithms will also be discussed in Section 4.

The simplest strategy is to use a fixed shift μ throughout the TRQ iteration. This shift is referred to as the *target* shift in the following discussion. In this case, a single matrix factorization of $A - \mu I$ may be used repeatedly to get inverse power method type of convergence. However, if the ratio

$$\sigma = \frac{|\lambda_j - \mu|}{|\lambda_{j+1} - \mu|} \tag{3.5}$$

is close to 1, the approximation to λ_j converges extremely slow. In Section 5, we compare this approach with the shifted and inverted IRA. It is observed that the shifted and inverted IRA is often more efficient in obtaining a few eigenvalues near a prescribed shift.

At the other extreme, we could adjust the shift at each iteration to enhance the rate of convergence. Eigenvalues of H_k are natural candidates for the shift. They provide the best approximations to eigenvalues of A from the subspace spanned by the columns of V_k, and are referred to as the Ritz values. Before each TRQ update we compute the Ritz values, and choose the one closest to the target shift as the next shift. A converged Ritz value should not be selected as a shift.

This choice of shift usually leads to quadratic or cubic convergence rate. However, this rapid convergence is obtained at the cost of factoring a matrix at each iteration. It is observed from our experiments that Ritz values tend to jump around during the early stage of the TRQ iteration. Thus, the target shift is used during the first few iterations until Ritz values start to settle down.

A compromise between the first and the second choice is to use a fixed shift until an eigenvalue has converged. Another possibility is to use each shift for (at most) a fixed number of iterations. In either case, the best Ritz value that has not yet converged may be selected as the next shift. Rapid convergence is generally obtained with this strategy. The cost for matrix factorization is reduced in comparison with the second approach. It will be shown in Section 5 that this scheme is very competitive with the Rational Krylov method of Ruhe [15], [14], [16].

Finally, the leading k-columns of the Implicitly Shifted RQ iteration may be obtained by selecting the same set of shifts as the full dense algorithm if desired. For example, if the elements of the matrix H are denoted by γ_{ij}, we could use γ_{11} as the shift. This corresponds to the Rayleigh quotient shift in the RQ algorithm. Another alternative is the Wilkinson shift. This is defined to be the eigenvalue of the leading 2×2 matrix

$$\begin{pmatrix} \gamma_{11} & \gamma_{12} \\ \gamma 21 & \gamma_{22} \end{pmatrix}$$

that is the nearest to γ_{11}. These strategies may be used when no target shift is given in advance, or when the TRQ iteration is used in conjunction with a deflation scheme to compute the full spectrum of A.

Once the shift is chosen, an RQ update as described in Steps 2.3 through 2.6 of Algorithm 2 is taken. Clearly, it can be done explicitly, but there may be some advantage to an implicit application. An implicit shift application is straightforward since

$$\begin{pmatrix} H_k - \mu I_k & h \\ \beta_k e_k^T & \alpha \end{pmatrix} = \begin{pmatrix} H_k & h \\ \beta_k e_k^T & \tilde{\alpha} \end{pmatrix} - \mu \begin{pmatrix} I_k & 0 \\ 0 & 1 \end{pmatrix},$$

where $\tilde{\alpha} = \alpha + \mu$. Thus the standard bulge-chase implementation of an RQ sweep corresponding to the shift μ may be applied to the matrix

$$\begin{pmatrix} H_k & h \\ \beta_k e_k^T & \tilde{\alpha} \end{pmatrix}.$$

Finally, when the matrix A is real nonsymmetric, we would like to perform the TRQ iteration in real arithmetic. However, there seems to be no simple analog to the double shifting strategy used in the QR algorithm. Applying double shifts implicitly in the TRQ iteration is possible. However, the corresponding TRQ equation involves $\hat{A} = (A - \bar{\mu}I)(A - \mu I)$, and more work is required to solve this equation. It is still questionable whether a truncated double implicit shifting strategy should be used in practice. Therefore, shall not present the details here. A double shift algorithm that involves solving $\hat{A}w = v$ may be found in [22].

3.3. Deflation. As discussed earlier, in each TRQ iteration the TRQ equation (2.4) is solved so that a truncated Hessenberg reduction of the form

$$A(V_k, v_+) = (V_k, v) \begin{pmatrix} H_k & h \\ \beta_k e_k^T & \alpha \end{pmatrix} \tag{3.6}$$

is maintained. As the TRQ iteration proceeds, the leading subdiagonal elements of H_k become small. Usually, they will become small in order (from top down) but occasionally this convergence happens further down the subdiagonal. When the magnitude of a subdiagonal element β_j falls below some numerical threshold, it is set to zero and the matrix H_k is split to give

$$H_k = \begin{pmatrix} H_j & M \\ 0 & \hat{H}_{k-j} \end{pmatrix}.$$

The first j columns of V_k form a basis for an invariant subspace of A, and j eigenvalues of A may be extracted from H_j. The deflation technique used in the QR algorithm can be applied here to obtain subsequent eigenvalues. We rewrite (3.6) as

$$(A - \mu I)(V_j, \hat{V}_{k-j}, v^+) = (V_j, \hat{V}_{k-j}, v) \begin{pmatrix} H_j - \mu I_j & M & h_1 \\ 0 & \hat{H}_{k-j} - \mu I_{k-j} & h_2 \\ 0 & \beta_k e_{k-j}^T & \alpha \end{pmatrix}, \tag{3.7}$$

where

$$V_k = (V_j, \hat{V}_{k-j}) \quad \text{and} \quad h = \begin{pmatrix} h_1 \\ h_2 \end{pmatrix},$$

have been partitioned conformably with V_j representing the leading j columns of V_k and h_1 representing the first j components of h.

An upper triangular matrix $\hat{R}$ and an orthogonal matrix $\hat{Q}$ of the form

$$\hat{R} = \begin{pmatrix} R_2 & r \\ 0 & \rho \end{pmatrix}, \quad \hat{Q} = \begin{pmatrix} Q_2 & q \\ \sigma e_{k-j}^T & \gamma \end{pmatrix}$$

are constructed such that

$$\begin{pmatrix} \hat{H}_{k-j} - \mu I_{k-j} & h_2 \\ \beta_k e_{k-j}^T & \alpha \end{pmatrix} = \hat{R}\hat{Q}.$$

Multiplying (3.7) from the right by $\tilde{Q}^H = \begin{pmatrix} I_j & \\ & \hat{Q}^H \end{pmatrix}$ yields

$$(A - \mu I)(V_j, \hat{V}_{k-j}^+, \hat{v}_+) = (V_j, \hat{V}_{k-j}, v) \begin{pmatrix} H_j - \mu I_j & \hat{M} & \hat{h}_1 \\ 0 & R_2 & r \\ 0 & 0 & \rho \end{pmatrix},$$

where $\hat{V}_{k-j}^{+} = \hat{V}_{k-j}Q_2^H + v_+q^H$, $\hat{v}_+ = \bar{\sigma}\hat{V}_{k-j}e_{k-j} + \bar{\gamma}v_+$, $\hat{M} = MQ_2^H + h_1q^H$, and $\hat{h}_1 = \bar{\sigma}Me_{k-j} + \bar{\gamma}h_1$. Note that the V_j and H_j are not modified during the deflation.

The next cycle of TRQ iteration starts with the selection of a new shift. The role of $\hat{H}_{k-j}$,$\hat{V}_j$ and $\hat{v}_+$ are replaced by $\hat{H}_{k-j}^{+} = Q_2R_2 + \mu I_{k-j}$, $\hat{V}_j^{+}$ and $\hat{v}_+$ respectively. If the subdiagonal elements of H_k converge to zero in order (from top to bottom,) a partial Schur form

$$AV_j = V_jR_j,$$

is obtained. Of course, when a subdiagonal β_j approaches zero out of order, then the splitting described in equation (split) above will still yield a partial Schur form since the Schur form of $H_jQ_j = Q_jR_j$ can be used to make an explicit transformation.

4. Numerical Examples. In this section, we evaluate the cost and performance of the Truncated RQ (TRQ) iteration. We first show an example indicating that the convergence rate of TRQ is exactly the same as that of the RQ iteration when the TRQ equations (2.4) are solved exactly. Comparisons will be made with the shifted and inverted IRA, the Rational Krylov Subspace (RKS) method and the recently proposed Jacobi-Davidson QR (JDQR) method [9]. We show that if the the shift is fixed, TRQ does not provide much advantage over the shifted and inverted IRA. However, if the shifts are allowed to change during the iteration, TRQ often performs better than IRA in terms of number of iterations, and is competitive with the RKS and the JDQR algorithms. Numerical examples will be presented to demonstrate the performance of the algorithm. All numerical experiments are performed using MATLAB 4.2 on a SUN-SPARC 2.

4.1. Convergence Rate of TRQ. The rate of convergence of TRQ follows from that of the full RQ iteration. For certain choices of shifts, it is cubic for symmetric eigenvalue problems and quadratic for nonsymmetric problems. In fact, if the Arnoldi iteration with the starting vector v_0 is used to produce the Hessenberg reduction required by Algorithm 1 as an input, the first k eigenvalues appearing on the diagonal of the output triangular matrix will be exactly the same as the those computed by TRQ with the same starting vector.

In the following, we present an example that verifies the fast convergence of TRQ. We choose to work with a standard 5-point discrete Laplacian defined on $[0,1] \times [0,1]$ with zero Dirichlet boundary conditions. For simplicity, the 100 by 100 symmetric matrix is scaled by h^2, where $h = 1/101$ is the mesh size of the discretization. We are interested in 4 eigenvalues with the smallest magnitude. The size of the Arnoldi factorization used in the TRQ iteration is set to be 5 ($k = 5$.) In each TRQ iteration, eigenvalues of the 5×5 tridiagonal matrix H_5 defined in Step 2.6 of Algorithm 2 are computed. The one nearest to zero that has not yet converged is chosen as the next shift μ. Table 4.1 lists the subdiagonal element β_j $(j = 1, 2, 3, 4)$ of H_5 at each iteration. Once $|\beta_j|/(|H_{j,j}| + |H_{j+1,j+1}|)$ drops below a prescribed tolerance of 10^{-15}, we set β_j to zero. Clearly, the first eigenvalue converges cubically, and the second one shows cubic convergence rate after the first one has converged. At the end of the 12-th iteration, all four eigenvalues

$$\lambda_1 = 0.16203$$
$$\lambda_2 = 0.39851$$
$$\lambda_3 = 0.39851$$

iteration	μ	β_1	β_2	β_3	β_4
1	0.18638	2.31×10^{-2}	2.18×10^{0}	1.91×10^{0}	1.58×10^{0}
2	0.16204	2.33×10^{-7}	6.23×10^{-1}	1.84×10^{0}	2.18×10^{0}
3	0.16203	1.11×10^{-21}	2.10×10^{-1}	1.36×10^{0}	1.84×10^{0}
4	0.44417	0	7.92×10^{-2}	1.27×10^{-1}	1.55×10^{0}
5	0.39857	0	1.36×10^{-5}	3.83×10^{-2}	7.24×10^{-1}
6	0.39851	0	4.08×10^{-17}	1.36×10^{-1}	9.47×10^{-2}
7	0.40410	0	0	1.34×10^{-2}	3.14×10^{-2}
8	0.39851	0	0	3.84×10^{-8}	4.24×10^{-2}
9	0.39851	0	0	8.58×10^{-21}	5.71×10^{-2}
10	0.63614	0	0	0	2.15×10^{-3}
11	0.63499	0	0	0	1.52×10^{-10}
12	0.63499	0	0	0	1.88×10^{-28}

TABLE 4.1
Convergence history of the 4 computed eigenvalues of a 2-D Laplacian.

$$\lambda_4 = 0.63499$$

are found. The convergence criterion here was a tolerance of 10^{-15} in the test for declaring a subdiagonal element to be zero. The computed direct residuals for all converged eigenpairs were on the order of 10^{-15}. The multiplicity of the eigenvalue 0.39851 is detected.

4.2. Comparison with IRA. It is mentioned in Section 3.2 that a simple way of selecting a shift in Step 2.1 of Algorithm 2 is to use a fixed shift throughout the TRQ iteration. Besides its simplicity, this strategy may also reduce the computational cost when factoring $A - \mu I$ is expensive. However, as one may expect, the convergence rate of each desired eigenvalue is typically linear in this case. When the ratio σ defined in (3.5) is close to 1, slow convergence is usually observed. In the following, we compare this variant of the TRQ algorithm with the shifted and inverted IRA since both algorithms factor the matrix $A - \mu I$ only once. It is shown in Table 4.2 that TRQ requires slightly less work and storage per iteration. However, our numerical experiments often show that the shifted and inverted IRA converges faster than TRQ with the same shift. An example is presented below to demonstrate this phenomenon. The problem involves the 2-dimensional Laplacian used in the previous section. Four smallest eigenvalues are sought. We placed the target shift at zero, and ran TRQ with $k = 5$ (TRQ(5)). The results are compared with IRA with $k = 4$, $p = 1$ (IRA(1)) and IRA with $k = 4$, $p = 4$ (IRA(4).) The value of p indicates the number of shifts used in the IRA iteration [21]. Since the ratio $\rho = |\lambda_1|/|\lambda_2|$ is close to 1, we expect TRQ to converge slowly. In Table 4.3, we list the converged eigenvalues and the number of linear systems solved before each eigenvalue has converged. One way to accelerate the TRQ iteration is to increase the size of the Arnoldi factorization. The motivation is to take advantage of large gaps that may exist in the unwanted portion of the spectrum. However, the gain is usually not significant unless such gaps are large enough. In Table 4.4, we compare the total number of linear solves used in finding the four desired eigenvalues of the 2-dimensional Laplacian with different k values. We observe that as k increases, the number of linear solves required in TRQ does not always decrease. Clearly, one does not want to use a k that is too large for this will increase the computational cost.

	TRQ	IRA
initialization cost	MATVEC (k times): *variable* GEMV: $O(nk^2)$ Factorization: *variable*	SOLVE ($k+p$ times): *variable* GEMV: $O(n(k+p)^2)$ Factorization: *variable*
cost per iteration	SOLVE: *variable* Shift selection $O(k^3)$ RQ update: $O(nk^2+k^2)$	SOLVE (p times): *variable* GEMV: $O(n(k+p)^2)$ Shift selection:$O((k+p)^3)$ QR update: $O((k+p)^3)$
storage	$O(n(k+1)+(k+1)^2)$	$O(n(k+p+1)+(k+p+1)^2)$

TABLE 4.2

Comparison of computational work and storage between TRQ and IRA. We assume that k eigenvalues closest to the shift σ are of interest. An Arnoldi factorization of length k is maintained in TRQ, and $p(\geq 1)$ shifts are applied in each IRA iteration (i.e., an Arnoldi factorization of length $k+p$ is maintained.) We use MATVEC to denote the matrix vector multiplication used in TRQ, and use SOLVE to indicate the cost of solving a linear system in both TRQ and IRA. The operation GEMV refers to dense matrix vector multiplications needed in carrying out Arnoldi factorization. The RQ or QR update refers to the bulge chase process used in both algorithms.

eigenvalue	TRQ(5)	IRA(1)	IRA(4)
0.16203	36	11	6
0.39851	79	16	7
0.39851	161	26	8
0.63499	186	40	11

TABLE 4.3

Comparison of IRA and TRQ on a 2-D Laplacian.

4.3. Comparison with RKS. The convergence rate of TRQ may be improved if shifts are chosen to be the best eigenvalue approximations from the subspace spanned by columns of V_k. However, this scheme requires factoring a matrix $A-\mu_j I$ at each iteration. To reduce the overall cost of TRQ, the third shift selection strategy discussed in Section 3.2 may be used, i.e., a shift is used repeatedly until either a Ritz value has converged or a fixed number of iterations has occurred. Then a new shift is selected. This strategy is also employed in the Rational Krylov Method (RKS) introduced by Ruhe [15], [14], [16]. In this section, we show by numerical example that TRQ is competitive with RKS.

The basic recursion involved in RKS [15] may be characterized by the equation

$$AV_{k+1}\hat{H}_k = V_{k+1}\hat{G}_k,$$

where V_{k+1} is n by $k+1$, $\hat{H}_k$ and $\hat{G}_k$ are $k+1$ by k, and $V_{k+1}^H V_{k+1} = I_{k+1}$. We denote the j-th column of V_{k+1}, $\hat{H}_k$ and $\hat{G}_{k+1}$ by v_j, h_j and g_j respectively. They are produced by a sequence of Arnoldi-like steps shown in Figure 4.1.

The choice of t_j is arbitrary, but $t_j = e_j$ is recommended. The subspace spanned by the columns of V_k do not form a Krylov subspace, and approximate eigenvalues may be obtained by solving the generalized eigenvalue problem

$$G_k s = \mu H_k s, \tag{4.1}$$

where G_k and H_k are the submatrices consisting of the first k rows of $\hat{G}_k$ and $\hat{H}_k$ respectively. The convergence of each Ritz value can be monitored by the estimate

k	no. of linear solves
5	186
10	132
15	132

TABLE 4.4
Comparison of TRQ(k) with different values of k.

```
(RKS) Rational Krylov Subspace Iteration

Input: $(A, v_1)$ such that $||v_1|| = 1$.
Output: $(V_{k+1}, \hat{H}_k, \hat{G}_k)$ such that $AV_{k+1}\hat{H}_k = V_{k+1}\hat{G}_k$, $V_{k+1}^H V_{k+1} = I$,
        and $H_{k+1}$ is upper Hessenberg.

1. Choose $t_1 = e_1$;
2. $V_1 \leftarrow (v_1)$; $\hat{H}_0 = (\,)$; $\hat{G}_0 = (\,)$;
3. for $j = 1, 2, 3, ..., k$.
   3.1. Choose a shift $\mu_j$;
   3.2. $w_{j+1} \leftarrow (A - \mu_j I)^{-1}(V_j t_j)$;
   3.3. $h_j \leftarrow V_j^H w_j$; $\hat{H}_j \leftarrow (\hat{H}_{j-1},\ h_j)$;
   3.4. $g_j \leftarrow h_j \mu_j + t_j$; $\hat{G}_j \leftarrow (\hat{G}_{j-1},\ g_j)$;
   3.5. $w_{j+1} \leftarrow w_{j+1} - V_j h_j$; $\beta_j = ||w_{j+1}||$;
   3.6. $\hat{H}_j \leftarrow \begin{pmatrix} \hat{H}_j \\ \beta_j e_j^T \end{pmatrix}$; $\hat{G}_j \leftarrow \begin{pmatrix} \hat{G}_j \\ \mu_j \beta_j e_j^T \end{pmatrix}$;
   3.7. $v_{j+1} \leftarrow w_{j+1}/\beta_j$; $V_{j+1} \leftarrow (V_j,\ v_{j+1})$;
   3.8. Choose a vector $t_{j+1}$;
4. end
```

FIG. 4.1. *Rational Krylov Subspace Iteration.*

derived in [15]. Deflation must be done properly [16] to avoid missing multiple eigenvalues. The cost of RKS per iteration is listed in Table 4.5.

It is mentioned in [16] that a large basis is needed when the eigenvalue problem is ill-conditioned. Thus reorthogonalization becomes expensive. Purging and restarting have been proposed in [16]. However, these schemes are still experimental and not well understood. In contrast, the size of V_k is fixed during the TRQ iteration, and the update is done by an orthogonal transformation. The convergence can be monitored by checking the magnitude of subdiagonal elements of H_k. Deflation is built into the TRQ iteration, and eigenvalues with multiplicity greater than one cause no difficulty. At convergence, a partial Schur form is constructed automatically without further reordering.

In the following, we compare TRQ and RKS on a 340×340 Tolosa matrix [2]. The Tolosa matrix is a model problem that has the important features of matrices that arise in the stability analysis of an airplane in flight. The full spectrum of this matrix is plotted in Figure 4.2. Eigenvalues with largest imaginary parts are of interest. We use the RKS code developed by Ruhe [16] for comparison. The same random starting vector is used in both RKS and TRQ. In the RKS code, Ritz values are computed

Operation	Cost
Factorization(intermittently)	*variable*
SOLVE	*variable*
GEMV	$O(nk^2)$
Ritz approximation	$O(k^3)$
Purging & restart	$O(nkk_d + k^4)$
storage	$O(n(k+1) + 2(k+1)^2)$

TABLE 4.5
The cost of the RKS iteration. The value of k is usually much larger than the number of desired eigenvalues k_d. Again, SOLVE refers to solving a linear system in Step 3.2 of the algorithm. The operation GEMV refers to dense matrix vector multiplications needed in carrying out the RKS factorization. Ritz approximation refers to solving the generalized eigenvalue problem $H_k s = \mu G_k s$.

from (4.1) at each iteration. A Ritz value is flagged as converged when the Ritz estimate falls below $tol = 10^{-10}$. The initial shift is placed at $\mu = -150 + 410i$. The same shift is used for at most m iterations. A new shift is selected after the current shift has been used for m iterations, or after convergence of a Ritz value. The same shift selection strategy is used in TRQ for comparison. In Table 4.6, we list the first five computed eigenvalues, and the number of iterations taken before each eigenvalue has converged. We choose $m = 5$ and $m = 10$ in RKS. The size of the Arnoldi factorization used in TRQ is set to be 6 ($k = 6$.) We tried $m = 1$ (optimal shift selection), $m = 5$ and $m = 10$ in TRQ. At the bottom of the table, we accumulated the total number of factorizations used in each run. For $m = 5$, the convergence history of RKS and TRQ are plotted in Figure 4.3 and Figure 4.4 respectively. In these figures, we plot the residual norm of each approximate eigenvalue against the number of flops (floating point operations). The vertical dotted line marks the end of each iteration, the dash-dot line marks the end of a matrix factorization. It is

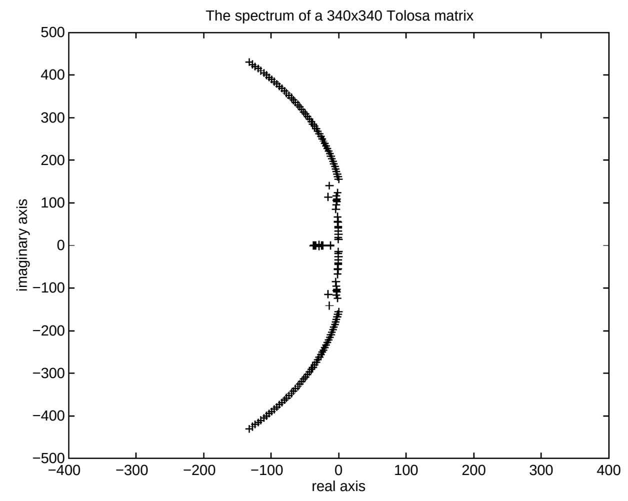

FIG. 4.2. The spectrum of a 340×340 Tolosa matrix.

observed from Table 4.6 that it takes more than 10 iterations for both RKS and TRQ to locate the first eigenvalue. Once the first one emerges, both algorithms converge at a rate of two iterations per eigenvalue. Notice that horizontal axes in Figure 4.3 and

eigenvalue	RKS	RKS	TRQ(6)	TRQ(6)	TRQ(6)	IRA(10)
	$m = 5$	$m = 10$	$m = 1$	$m = 5$	$m = 10$	
$-132.3 + 430.1i$	12	14	11	14	15	14
$-127.9 + 425.2i$	15	16	13	16	17	51
$-123.5 + 420.2i$	17	18	15	17	18	64
$-119.3 + 415.2i$	19	20	17	19	20	125
$-115.1 + 410.2i$	31	22	19	21	22	201
factorizations	9	6	19	7	6	1

TABLE 4.6
Comparison of IRA and TRQ on a Tolosa matrix.

4.4 are labeled with different scales. For this problem, RKS builds a larger subspace than TRQ in order to capture all desired eigenpairs. Thus more orthogonalizations are performed in RKS. This explains the larger number flops required by RKS. Residual norms of all Ritz pairs are plotted in Figure 4.3. Only five of them have converged to the desired tolerance of 10^{-10}. Clearly, TRQ is competitive with RKS in terms of both the number of factorizations and the number of iterations, and both algorithms compare favorably with IRA with $p = 10$ (IRA(10).)

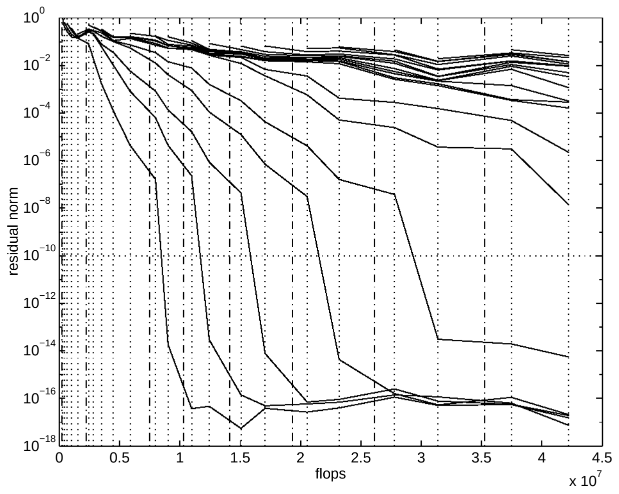

FIG. 4.3. The convergence history of RKS.

4.4. Comparison with JDQR. If factoring $A - \mu I$ is inexpensive, we may consider using an optimal shift described in Section 3.2 in each TRQ iteration. In this case, the performance of TRQ is comparable with that of the Jacobi-Davidson method.

Given an initial approximation v_0 of a desired eigenvector, the Jacobi-Davidson method [19] finds, at each step, a correction vector z_k that is orthogonal to the previous approximate eigenvector u_k. A new subspace is created by adjoining this vector to the previous subspace and taking the span. The next approximate eigenpairs are drawn from projection onto the new subspace. The correction vector z_k is obtained from the

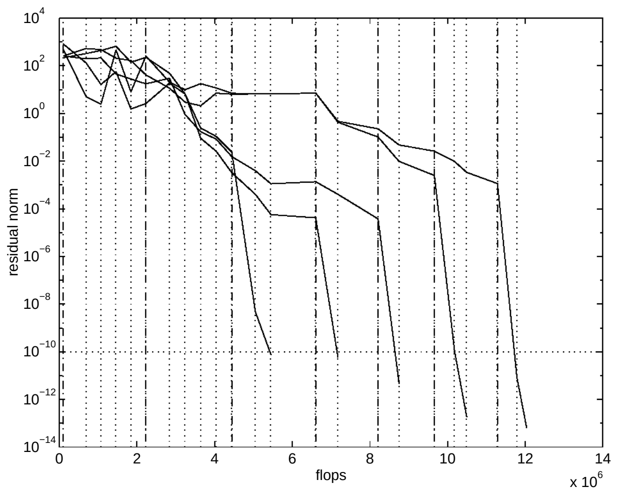

FIG. 4.4. The convergence history of TRQ.

equation

$$(I - u_k u_k^H)(A - \theta_k I)(I - u_k u_k^H)z_k = -r_k \quad \text{and} \quad z_k \perp u_k, \tag{4.2}$$

where $r_k = Au_k - \theta_k u_k$, and θ_k is the current approximation to the eigenvalue of interest. It can be shown [19] that if (4.2) is solved exactly, the Jacobi-Davidson method becomes the accelerated inverse iteration, i.e., it builds an orthonormal basis of the subspace

$$\mathcal{S}(A, v_0, \{\theta_j\}) = \text{span}\{v_0, v_1, v_2, ...v_k\},$$

where $v_j = (A - \theta_j I)^{-1} v_{j-1}$. Ritz approximations are extracted from this subspace. It is shown in [16] that this method is equivalent to RKS with an optimal shift selected in each iteration. The subspace $\mathcal{S}(A, v_0, \{\mu_j\})$ is not a Krylov subspace. The Hessenberg relationship (3.1) is not preserved in the Jacobi-Davidson iteration. To obtain several eigenvalues and eigenvectors, some standard deflation schemes [17] are needed. To avoid building a large dimensional subspace $\mathcal{S}$, restarting is also necessary. The implementation of the Jacobi Davidson QR (JDQR) algorithm is explained in detail in [9]. We compare the performance of TRQ and JDQR on a standard eigenvalue problem arising from the stability analysis of the Brusselator wave model (BWM) [2]. Eigenvalues with largest real parts are of interest. They help to determine the existence of stable periodic solutions to the Brusselator wave equation as some parameter varies. The size of the matrix we choose is 200×200. The 32 rightmost eigenvalues are plotted in Figure 4.5. We place the target shift at $\sigma = 1.0$, and use TRQ and JDQR to find 4 eigenvalues closest to σ. In Table 4.7, we list the first four computed eigenvalues and number of factorizations used to obtain each one of them. In the runs using TRQ, we tried $k = 5$ and $k = 8$. In JDQR, the maximum dimension of subspace from which approximate eigenpairs are drawn is 8. Restart begins at the 6th column ($j_{min} = 5$.) It is denoted by JDQR(5,8) in Table 4.7.

It is observed from Table 4.7 that TRQ takes fewer iterations to find all four eigenvalues of interest. However, as pointed out in [9], the correction equation may

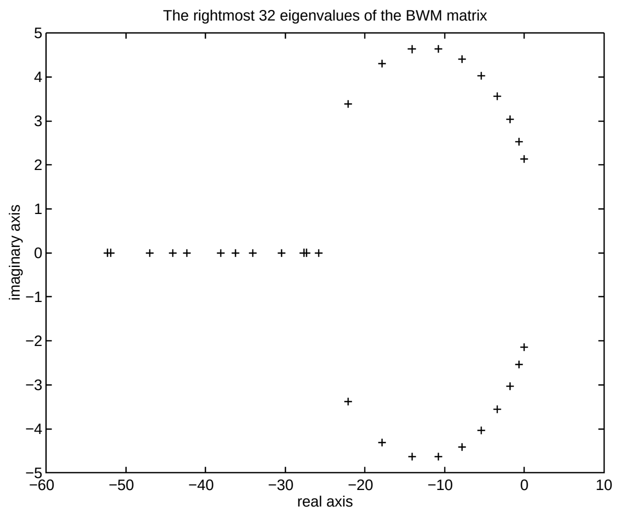

FIG. 4.5. The 32 rightmost eigenvalues of a 200×200 BWM matrix.

eigenvalue	TRQ(5)	TRQ(8)	JDQR(5,8)
$1.820 \times 10^{-5} + 2.140i$	8	6	14
$1.820 \times 10^{-5} - 2.140i$	11	8	17
$-0.6747 + 2.529i$	14	12	19
$-0.6747 - 2.529i$	16	14	21

TABLE 4.7
Comparison of TRQ and JDQR on the BWM problem.

be solved by one step of GMRES iteration in the first j_{min} steps of JDQR iterations. This is equivalent to building the initial Jacobi Davidson search space [9] by running a j_{min}-step Arnoldi iteration. For the BWM problem, this technique reduces the total number of exact solves in JDQR(5,8) to 16.

5. Inexact TRQ and Restarting. Rapid convergence of the TRQ algorithm is observed in Section 4 when the TRQ equation

$$(I - V_k V_k^H)(A - \mu I)(I - V_k V_k^H)v_+ = v\alpha, \quad \text{with} \quad V_k^H v_+ = 0, \quad \|v_+\| = 1 \tag{5.1}$$

is solved exactly in each iteration. In this section, we explore the possibility of relaxing the solution accuracy of (5.1) while maintaining the rapid convergence of TRQ iteration. This is extremely important for many applications in which the factorization of $A - \mu I$ is too costly, and an approximate solution of $(A - \mu I)x = b$ can be provided by an iterative solver.

Recall that one of the important characteristics of the TRQ algorithm is the inverse iteration relation between the first column of V_k^+ and the first column of V_k, i.e.,

$$(A - \mu I)v_1^+ = v_1.$$

If an optimal shift is chosen at each iteration, the convergence of v_1 to an eigenvector of A is often quadratic or cubic. We will show in the following that if the projected equation is solved approximately, an inexact inverse iteration is maintained between v_1^+ and v_1. Superlinear convergence can still be achieved if optimal shifts are used.

Suppose $\tilde{v}_+$ is an approximate solution to (5.1). Since $I - V_k V_k^H$ is a projection, we may replace $\tilde{v}_+$ with $(I - V_k V_k^H)\tilde{v}_+$ in (5.1). Thus, we explicitly orthogonalize the approximate solution $\tilde{v}_+$ against all columns of V_k through

$$\tilde{v}_+ \leftarrow (I - V_k V_k^H)\tilde{v}_+,$$

and normalize it so that $\|\tilde{v}_+\| = 1$. The unknowns h and α present in (2.4) are then computed directly as if $\tilde{v}_+$ were an exact solution to (5.1), i.e.,

$$\tilde{h} \leftarrow V_k^H (A - \mu I)\tilde{v}_+ = V_k^H A \tilde{v}_+, \quad \tilde{\alpha} \leftarrow v^H (A - \mu I)\tilde{v}_+.$$

These lead to the equation

$$(A - \mu I)(V_k, \tilde{v}_+) = (V_k, v) \begin{pmatrix} H_k - \mu I_k & \tilde{h} \\ \beta_k e_k^T & \tilde{\alpha} \end{pmatrix} + z e_{k+1}^T, \tag{5.2}$$

where $z e_{k+1}^T$ is an error term with

$$z \equiv (A - \mu I)\tilde{v}_+ - (V_k, v) \begin{pmatrix} \tilde{h} \\ \tilde{\alpha} \end{pmatrix}.$$

By construction, z satisfies

$$V_k^H z = 0, \quad v^H z = 0.$$

We may now compute an upper triangular $\hat{R} = \begin{pmatrix} R_k & r \\ 0 & \rho \end{pmatrix}$ and an orthogonal $\hat{Q} = \begin{pmatrix} Q_k & q \\ \sigma e_k^T & \gamma \end{pmatrix}$ such that

$$\begin{pmatrix} H_k - \mu I_k & \tilde{h} \\ \beta_k e_k^T & \tilde{\alpha} \end{pmatrix} = \hat{R}\hat{Q},$$

and multiply (5.2) from the right by $\hat{Q}^H$ to get

$$(A - \mu I)(V_k Q_k^H + \tilde{v}_+ q^H, v_k \bar{\sigma} + \tilde{v}_+ \bar{\gamma}) = (V_k, v) \begin{pmatrix} R_k & r \\ 0 & \rho \end{pmatrix} + z(q^H, \bar{\gamma}).$$

The first column of $V_k^+ = V_k Q_k^H + \tilde{v}_+ q^H$ is related to the first column of V_k through the equation

$$(A - \mu I)v_1^+ = \rho_{11} v_1 + z\delta, \tag{5.3}$$

where $v_1^+ = V_k^+ e_1$ and δ is the first element of the vector q. Since the orthogonal matrix $\hat{Q}$ is constructed from accumulation of a sequence of Givens rotations used in the RQ factorization, δ is a product of $(k-1)$ sines. Its magnitude is bounded by 1 and it is likely to be quite small due to the accumulated product of sines. Thus the error term present in the inexact inverse iteration (5.3) is at worst of the same magnitude as the error introduced in solving (5.1) and is very likely to be much smaller. In fact if the first subdiagonal element β_1 is small (indicating the (1,1) element of H_k is

```
Algorithm 4: (RTRQ) Truncated RQ-iteration with restart

Input: $(A, V_k, H_k, f_k)$ with $AV_k = V_k H_k + f_k e_k^T, V_k^H V_k = I$, $H_k$ upper
       Hessenberg.
Output: $(V_k, H_k)$ such that $AV_k = VH_k, V_k^H V_k = I$ and $H_k$ is upper triangular.

1. Put $\beta_k = \|f_k\|$ and put $v = f_k/\beta_k$;
2. for $j = 1, 2, 3, \ldots$ until convergence,
   2.1. Select a shift $\mu \leftarrow \mu_j$;
   2.2. Solve $(I - V_k V_k^H)(A - \mu I)(I - V_k V_k^H)w = v$ approximately;
   2.3. $w \leftarrow (I - V_k V_k^H)w$, $v_+ \leftarrow w/\|w\|$;
   2.4. $h \leftarrow V_k^H A v_+$,  $\alpha \leftarrow v^H (A - \mu I)v_+$ ;
   2.5. RQ Factor $\begin{pmatrix} H_k - \mu I_k & h \\ \beta_k e_k^T & \alpha \end{pmatrix} = \begin{pmatrix} R_k & r \\ 0 & \rho \end{pmatrix} \begin{pmatrix} Q_k & q \\ \sigma e_k^T & \gamma \end{pmatrix}$;
   2.6. $v_1 \leftarrow V_k Q_k^H e_1 + v_+ q^H e_1$;
   2.7. Restart: $(H_k, V_k, v, \beta_k) \leftarrow$ Arnoldi$(A, v_1)$;
3. end;
```

FIG. 5.1. *Restarted TRQ iteration.*

eigenvalue	RTRQ(5)	JDQR(5,8)
5.5024	3	15
5.5024	6	23
1.5940	11	25
1.5940	17	35

TABLE 5.1
Comparison of RTRQ and JDQR on the CK656 problem.

nearly an eigenvalue of A) then $|\delta|$ is very likely to be smaller than $|\beta_1|$ which may be verified by considering the effect of the final Givens rotation to occur in the RQ step. Therefore, the error committed by accepting the inexact solution to the linear system (5.1) is damped by the RQ step to obtain a more accurate inverse-iteration relation between the vectors v_1^+ and v_1 than might be expected.

We would like to continue the TRQ update as described in Steps 2.4-2.6 of Algorithm 2. However, because of the error incurred in (5.2), the updated orthonormal basis $V_k^+ = V_k Q^H + \tilde{v}_+ q^H$ no longer spans a Krylov subspace. However, the first column of V_k^+ is approximately what we would have obtained if the TRQ equation is solved exactly. Thus one may recover a truncated Hessenberg reduction by running a k-step Arnoldi process with v_1^+ as the starting vector. We refer to this step as a *restart*. The restarted TRQ (RTRQ) iteration is summarized in Algorithm 4.

If a Krylov subspace type of method (such as conjugate gradient or GMRES) is used to solve the TRQ equation in step 2.2 of the above algorithm, it maybe of advantage to work with the operator $B \equiv (I - V_k V_k^H)(A - \mu I)(I - V_k V_k^H)$ directly since B may be better conditioned in the subspace $V_k^\perp$. Of course, the matrix B need not be formed explicitly, only the matrix vector multiplication Bv is required.

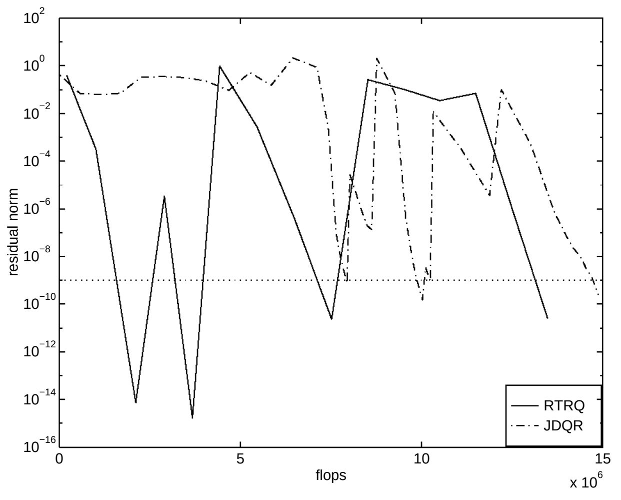

FIG. 5.2. Convergence history of RTRQ and JDQR for the CK656 matrix

5.1. Comparison with JDQR. In the following, we present a numerical example of using the inexact TRQ iteration with restart (RTRQ) to compute the eigenvalues of the CK656 matrix described in [2]. Eigenvalues of this matrix all have multiplicity two. We look for 4 eigenvalues near the target shift $\sigma = 5.0$, and set $k = 5$ in RTRQ (RTRQ(5).) The computational result is compared with JDQR with $j_{min} = 5$, $j_{max} = 8$ (JDQR(5,8).) The same random starting vector is used in both tests. The TRQ equation and the projected correction equation in JDQR are solve by GMRES with no preconditioning or restart. The maximum GMRES steps allowed in each linear solve is set to be 10. The GMRES residual tolerance is set to be 10^{-6}. The optimal shift selection strategy is used in both tests, i.e., the Ritz value that is the nearest to the target shift but has not converged is used as the next shift. No tracking [9] is used in JDQR. In Table 5.1, we list the four eigenvalues of interest and the number of iterations taken by RTRQ and JDQR before each eigenvalue has converged. We observe that for this example, RTRQ takes fewer iterations than JDQR to capture eigenvalues of interest. In particular, RTRQ is able to capture the first eigenvalue much quicker than JDQR. However, RTRQ costs more per iteration than JDQR because the projection in the TRQ equation always involves k vectors, and k matrix vector multiplications must be performed in each iteration to reconstruct an Arnoldi factorization. Thus, the overall performance should be compared in terms of total number of matrix vector multiplications or flops used in both methods. This is illustrated in Figure 5.2. We plot the residual of each approximate eigenpair against the number of flops. The residuals of the approximate eigenpairs are monitored one at a time. When the residual curve corresponding to the approximation to the eigenpair (λ_j, z_j) drops below 10^{-9}, we start to monitor and record the residual for the next approximate eigenpair (λ_{j+1}, z_{j+1}). We should point out that the comparison made here is still preliminary. Several techniques are available to improve the performance of JDQR [9], and many of these may be used in RTRQ as well.

5.2. The Effect of Preconditioning. Solving the TRQ equation is the most expensive part of the TRQ iteration. When an iterative method is used, a good preconditioner may accelerate the convergence and reduce the overall cost. The improved

eigenvalue	diagonal	ILU(0)	tridiagonal
$1.820\times 10^{-5}+2.140i$	80	21	7
$1.820\times 10^{-5}-2.140i$	>100	38	12
$-0.6747+2.529i$	>100	52	19
$-0.6747-2.529i$	>100	66	23

TABLE 5.2
Comparison of RTRQ with and without preconditioner

accuracy in the solution to the TRQ equation often brings about a reduction in the total number of TRQ iterations.

One may precondition the projected system

$$(I-V_kV_k^T)(A-\mu I)(I-V_kV_k^T)w=v,$$

directly to obtain an approximate solution to the TRQ equation. However, it may not be easy to find a good preconditioner M for the projected matrix $(I-V_kV_k^T)(A-\mu I)(I-V_kV_k^T)$. Instead, one usually has a preconditioner for the matrix A. As pointed out in [9], this preconditioner may need to be projected into $V_k^\perp$ in order to accelerate the convergence of the Jacobi Davidson iteration. The projected shifted preconditioner is sometimes not a good preconditioner for the projected shifted matrix A. This extra projection does not seem to be necessary in the TRQ iteration since the TRQ equations may be solved using the scheme discussed in Section 3. This scheme solves a linear system $(A-\mu I)w=v$. Thus a preconditioner of A may be easily applied. In the following we present an example that demonstrates the effect

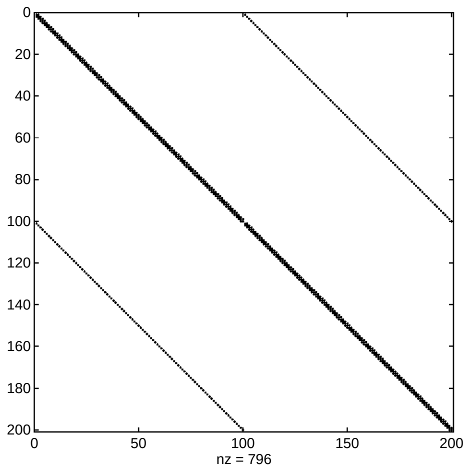

FIG. 5.3. The structure of a 200×200 BWM matrix

of preconditioning on the restarted TRQ iteration. Four eigenvalues of the the BWM matrix used in Section 4 are computed, and the size of the Arnoldi factorization in the TRQ iteration is set to be 5 ($k=5$.) The target shift is placed at 1.0. The TRQ equation is solved using a preconditioned GMRES with no restart. The maximum number of GMRES iterations allowed in each solve is set to be 10. The GMRES residual tolerance is set to be 10^{-6}. The structure of the BWM matrix is shown in Figure 5.3. We used the diagonal part, the tridiagonal part and the incomplete LU factors

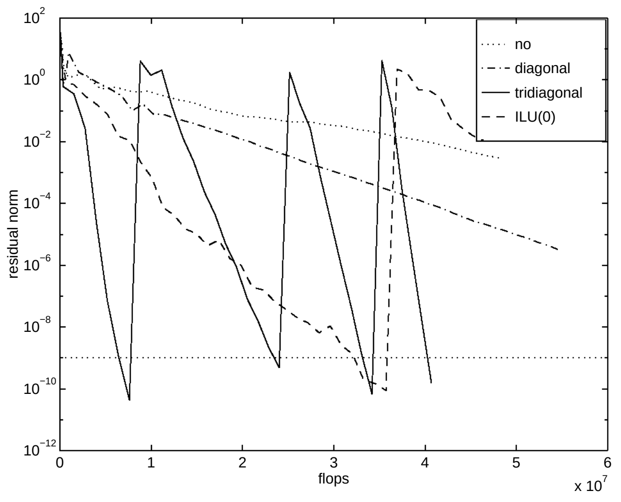

FIG. 5.4. Convergence history of Preconditioned TRQ for the BWM matrix

(ILU(0)) of the matrix A as the preconditioner. The number of iterations used to obtain the four eigenvalues near 1.0 are listed in Table 5.2. Without a preconditioner no eigenvalue is found in 100 iterations. The convergence history of RTRQ with various preconditioners is shown in Figure 5.4. The residual norm of each approximate eigenpair is plotted against the number flops subsequentially. The solid curve corresponds to RTRQ with tridiagonal preconditioning. The dashed curve corresponds to RTRQ with ILU(0) preconditioning. The dash-dot curve corresponds to RTRQ with diagonal preconditioning. The dotted curve is associated with RTRQ with no preconditioning. When the residual curve drops below the dotted line indicating the acceptable residual tolerance 10^{-9}, we start to monitor and record the residual of the next approximate eigenpair. It is observed that a good preconditioner improves the convergence of RTRQ dramatically.

5.3. Comparison with Accelerated Inverse Iteration with Wielandt Deflation. The inexact TRQ iteration with restart does not completely mimic the exact TRQ. In particular, the truncated Hessenberg reduction is enforced through an Arnoldi iteration rather than an implicit RQ update. The method behaves more like a single vector iteration with deflation than an RQ iteration in which the rapid convergence of one eigenvalue is often accompanied with the convergence of other eigenvalues at a slower pace.

In this section, we compare restarted TRQ with the accelerated inverse iteration combined with a deflation scheme that is very close to the Wielandt deflation (INVWD) [17, pp. 117] for computing a few eigenvalues of A. We show that the exact TRQ performs better than the exact INVWD and the inexact TRQ appears to be more reliable than the inexact INVWD.

The inverse iteration can be viewed as a shifted and inverted power iteration. It requires solving

$$(A - \mu I)w = v,$$

where v is the previous approximation to an eigenvector and w is the current approximation. The acceleration is achieved by choosing, at each iteration, a shift μ

```
(INVWD) A Schur-Wielandt Deflated Inverse Iteration

Input: $(A,\ \mu,\ v,\ U)$ such that $(\mu, v)$ is the current approximation to
       the desired eigenpair, and columns of $U$ contain the converged
       Schur vectors.
Output: A new approximate eigenpair $(\mu_+,\ v_+)$ that may be used in the
        next cycle of an inverse iteration.

1. Solve $(A - \mu I)w = v$;
2. $v \leftarrow (I - UU^H)w$; $v \leftarrow v/\|v\|$;
3. $f \leftarrow Av$; $\alpha = v^H w$;
4. $H_1 = (\alpha)$; $V = (v)$; $f \leftarrow f - v\alpha$;
5. $f \leftarrow (I - UU^H)f$;
6. for $j = 1, 2, ..., k$
   6.1. $\beta_j = \|f\|$; $v_{j+1} \leftarrow f/\beta_{j+1}$;
   6.2. $V_{j+1} = (V_j,\ v_{j+1})$; $H_j \leftarrow \begin{pmatrix} H_j \\ \beta_j e_j^T \end{pmatrix}$;
   6.3. $z \leftarrow Av_{j+1}$; $z \leftarrow (I - UU^H)z$;
   6.4. $h \leftarrow V_j^H z$; $H_{j+1} = (H_j,\ h)$;
   6.5. $f \leftarrow z - V_{j+1}h$;
7. end;
8. Compute an desired Ritz pair $(\mu_+, v_+)$ from $H_k$ and $V_k$ to be used in the
   next cycle of an inverse iteration.
```

FIG. 5.5. *Schur-Wielandt Deflated Inverse Iteration*

that is the best approximation to the desired eigenvalue. Once an eigenpair (λ, u) has been found, the next pair may be obtained by applying shifted power iteration to the deflated operator $A_1 = (A - \mu I)^{-1} - uq^H$, where $q = (A - \mu I)^{-H}u$. This deflation scheme is an variant of the *explicit* Wielandt deflation [23, pp. 596], [17, pp. 117]. The deflated operator $A_1 = (I - uu^H)(A - \mu I)^{-1}$ does not preserve right eigenvectors of A in general, unless A is normal. However, it does preserve Schur vectors of A. Thus, to generalize this deflation scheme for a converged invariant subspace, one should replace u with a matrix of Schur vectors U that spans the converged invariant subspace and satisfies $U^H U = I$. This is a more stable variant of a technique referred to as the Schur-Wielandt deflation in [17, pp. 122]. It leads to the algorithm INVWD (Figure 5.5) which we adopt here for comparison to RTRQ.

In the following, we first present an example that demonstrates the advantage of using TRQ over using inverse iteration with Schur-Wielandt-like deflation. Then we compare the performance of the inexact TRQ with restart to the inverse iteration in which the linear system is solved approximately.

In the first example, we choose A to be the 2-dimensional discrete Laplacian used before. Six eigenvalues of the smallest magnitude are computed. The size the Arnoldi factorization maintained in the TRQ iteration is 7 ($k = 7$.) The same size is chosen for the deflated Arnoldi iteration used in INVWD to help determine the shift. The same random starting vector is used in both TRQ and INVWD. In INVWD, a Ritz pair (μ_j, z_j) is considered to be converged if the direct residual norm $\|r_j\| =$

eigenvalue	TRQ	INVWD
0.16203	3	3
0.39851	6	7
0.39851	7	13
0.63499	10	17
0.77129	12	20
0.77129	13	24

TABLE 5.3
Comparison of TRQ and INVWD on a 2-D Laplacian.

$||Az_j - \mu_j z_j||$ falls below $tol = 10^{-12}$. In TRQ, the convergence criterion is a tolerance of machine epsilon in the test for declaring a subdiagonal element to zero. Table

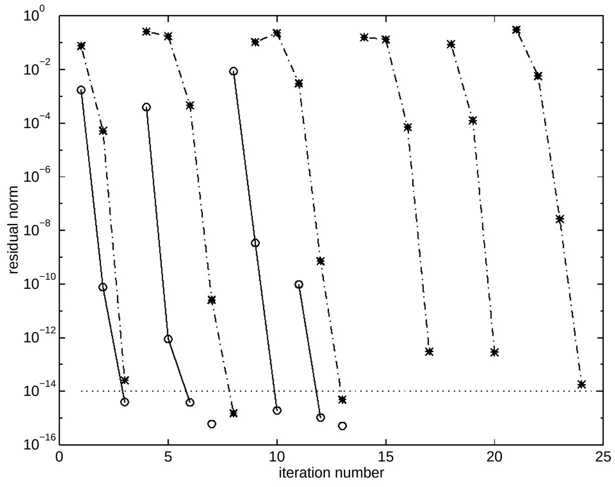

FIG. 5.6. Traces of the residual in TRQ and INVWD

5.3 shows the number of iterations taken before each eigenvalue has converged. In Figure 5.6, the convergence history of the residual for each computed eigenpair is shown. The height of each circle and star corresponds to the residual of the eigenpair computed by TRQ and INVWD respectively. The TRQ residuals corresponding to the approximations to the same eigenpair are connected by a solid line. The INVWD residuals are connected by a dash dot line. The circles below the dotted line correspond to the residuals of converged eigenpairs computed by TRQ. It is easily observed that the global convergence of TRQ is better than INVWD. In INVWD, every residual curve starts from the top ($||r|| \approx 10^{-1}$,) whereas in TRQ, the convergence of the second and fifth eigenpairs are followed by the immediate convergence of the third and the sixth pairs. The residual for the fifth eigenpair starts from roughly 10^{-10}, and drops below 10^{-14} in one iteration. We should also mention that the convergence of INVWD is sensitive to the starting vector and the size of the subspace used to obtained the shift. Eigenvalues may not necessarily converge in order. For example, large eigenvalues may appear early when we look for the ones with the smallest magnitude.

In the next example, we compare the performance of the inexact TRQ with that of the inexact INVWD. We consider computing eigenvalues of the DW1024 matrix that arises from dielectric waveguide problems in integrated circuit applications [2]. Four eigenvalues near 1.0 are of interest. In both methods, linear systems are solved by GMRES with no restart. The maximum number of GMRES iterations allowed is set to be 10. The GMRES residual tolerance is set to be 10^{-8}. The size of the Arnoldi factorization maintained in the inexact TRQ iteration is set to be 5 ($k = 5$.) The same size is set for the deflated Arnoldi iteration used in INVWD to determine the shift. The traces of the residual for each computed eigenpair are shown in Figure 5.7. Residual norms are plotted against the number flops. The solid curve corresponds to the residual norm of the inexact TRQ. The dotted curve corresponds to the residual of the inexact INVWD. We observe that the inexact INVWD converges much slower than the inexact TRQ.

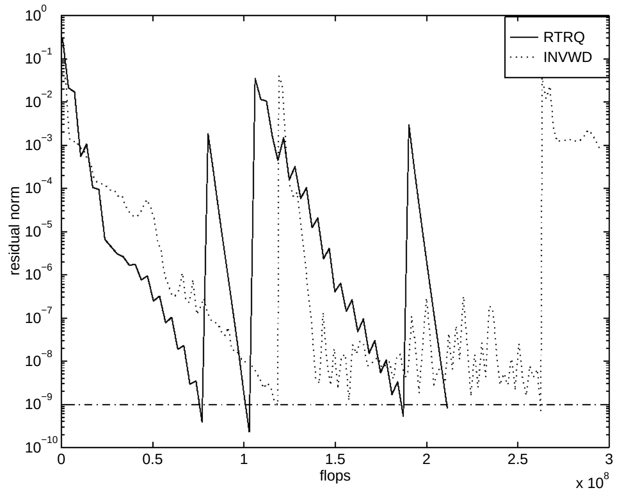

FIG. 5.7. Traces of the residual in inexact TRQ and INVWD

6. Conclusions.

This development of the Truncated RQ iteration has led to a promising way to take advantage of situations where shift-invert equations can be solved directly and also when they can only be solved inexactly through iterative means. We have demonstrated with several numerical experiments that this scheme provides a promising and competitive alternative to Rational Krylov Methods and the Jacobi Davidson Method in the two respective cases. The scheme is relatively simple and very efficient in terms of required numerical computation compared to these and other related methods. Finally, the convergence properties and deflation schemes are easily understood through the close connection with the RQ iteration for dense matrices.

Future research will focus upon analyzing the filtering properties obtained from embedding the shift-invert equations in the TRQ iteration. Equation (5.3) indicates a damping of the error introduced by inexact solution when the RQ iteration is carried out. The numerical properties and implications of this phenomenon are not yet understood.

We chose the GMRES method to solve the TRQ equation iteratively in the inexact TRQ method because of its simplicity and reliability. Certainly, other iterative solvers such as QMR, BICGSTAB could have been used. It would be interesting to compare the performance of these iterative solvers in the TRQ context. More research is required with respect to preconditioners and how they should be utilized within the TRQ equations. Exhaustive computational experimentation and comparisons are needed to determine whether the TRQ equations should be solved in bordered form, projected form, or by utilizing Lemma 3.1. These are issues both for direct and iterative solutions of the TRQ equations. The extension of these ideas to the generalized eigenvalue problem will also be important. Eventually, we expect to produce numerical software based upon this scheme to complement the IRA schemes already available in ARPACK.

Acknowledgement. We would like to thank R. B. Lehoucq and G. L. G. Sleijpen for reading the manuscript in detail and providing us with numerous corrections and suggestions. In particular, suggestions from Drs. Sleijpen and Lehoucq improved Sections 3.1 and 5.3 respectively. We would also like to thank the anonymous referees for careful reading and helpful comments.

REFERENCES

[1] J. Baglama, D. Calvetti, and L. Reichel. Iterative methods for the computation of a few eigenvalues of a large symmetric matrix. *BIT*, 36(3):400–421, 1996.

[2] Z. Bai, R. Barrett, D. Day, J. Demmel, and J. Dongarra. Test matrix collection (non-hermitian eigenvalue problems). Research report, Department of Mathematics, University of Kentucky, 1995.

[3] Z. Bai and G. W. Stewart. SRRIT — A FORTRAN subroutine to calculate the dominant invariant subspace of a nonsymmetric matrix. Technical Report 2908, Department of Computer Science, University of Maryland, 1992. To appear in *ACM Transactions on Mathematical Software*.

[4] T. Braconnier. The Arnoldi–Tchebycheff algorithm for solving large nonsymmetric eigenproblems. Technical Report TR/PA/93/25, CERFACS, Toulouse, France, 1993.

[5] D. Calvetti, L. Reichel, and D. C. Sorensen. An implicitly restarted Lanczos method for large symmetric eigenvalue problems. *ETNA*, 2:1–21, March 1994.

[6] F. Chatelin. *Eigenvalues of Matrices.* Wiley, 1993.

[7] J. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart. Reorthogonalization and stable algorithms for updating the Gram–Schmidt QR factorization. *Mathematics of Computation*, 30:772–795, 1976.

[8] I. S. Duff and J. A. Scott. Computing selected eigenvalues of sparse unsymmetric matrices using subspace iteration. *ACM Transactions on Mathematical Software*, 19(2):137–159, June 1993.

[9] D. R. Fokkema, G. L. G. Sleijpen, and H.A. Van der Vorst. A Jacobi-Davidson style QR and QZ algorithm for partial reduction of matrix pencils. Technical Report 941, University Utrecht, Department of Mathematics, 1996. To appear in *SIAM Journal on Scientific Computing*.

[10] R. W. Freund and N. M. Nachtigal. QMRPACK: A package of QMR algorithms. *ACM Transactions on Mathematical Software*, 22(1):46–77, March 1996.

[11] R. B. Lehoucq and D. C. Sorensen. Deflation techniques for an implicitly restarted Arnoldi iteration. *SIAM Journal on Matrix Analysis and Applications*, 17(4), 1996.

[12] R. B. Lehoucq, D. C. Sorensen, P. Vu, and C. Yang. ARPACK: *An implementation of the Implicitly Re-started Arnoldi Iteration that computes some of the eigenvalues and eigenvectors of a large sparse matrix*, 1995. Available from ftp.caam.rice.edu under the directory pub/software/ARPACK.

[13] R. B. Morgan. On restarting the Arnoldi method for large nonsymmetric eigenvalue problems. *Mathematics of Computation*, 65(215), 1995.

[14] A. Ruhe. The rational Krylov algorithm for nonsymmetric eigenvalue problems, III: Complex shifts for real matrices. *BIT*, 34:165–176, 1994.

[15] A. Ruhe. Rational Krylov algorithms for nonsymmetric eigenvalue problems, II: Matrix pairs. *Linear Algebra and Its Applications*, 197,198:283–295, 1994.

[16] A. Ruhe. Rational Krylov, a practical algorithm for large sparse nonsymmetric matrix pencils. Technical Report UCB/CSD-95-871 (revised), Computer Science Division(EECS), University of California Berkeley, CA 94720, 1995.

[17] Y. Saad. *Numerical Methods for Large Eigenvalue Problems.* Halsted Press, 1992.

[18] J. A. Scott. An Arnoldi code for computing selected eigenvalues of sparse real unsymmetric matrices. *ACM Transactions on Mathematical Software*, 21:432–475, 1995.

[19] G. L. G. Sleijpen and H.A. Van der Vorst. A Jacobi-Davidson iteration method for linear eigenvalue problems. *SIAM Journal on Matrix Analysis and Applications*, 17(2):401–425, April 1996.

[20] G.L.G. Sleijpen, J.G.L. Booten, D.R. Fokkema, and H.A. Van der Vorst. Jacobi-Davidson type methods for generalized eigenproblems and polynomial eigenproblems. *BIT*, 36(3):595–633, 1996.

[21] D. C. Sorensen. Implicit application of polynomial filters in a k-step Arnoldi method. *SIAM Journal on Matrix Analysis and Applications*, 13(1):357–385, January 1992.

[22] D.C. Sorensen and C. Yang. A truncated RQ-iteration for large scale eigenvalue calculations. Technical Report TR96-06, Department of Computational & Applied Mathematics, Rice Univeristy, Houston, TX 77005, 1996.

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