On Tridiagonalizing and Diagonalizing Symmetric Matrices with Repeated Eigenvalues*

Christian H. Bischof
Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, IL 60439–4801
bischof@mcs.anl.gov

Xiaobai Sun†
Department of Computer Science
Duke University
Durham, NC 27708-0129
xiaobai@cs.duke.edu

Argonne Preprint MCS–P545–1095
PRISM Working Note #25‡

Abstract. We describe a divide-and-conquer tridiagonalization approach for matrices with repeated eigenvalues. Our algorithm hinges on the fact that, under easily constructively verifiable conditions, a symmetric matrix with bandwidth b and k distinct eigenvalues must be block diagonal with diagonal blocks of size at most bk. A slight modification of the usual orthogonal band-reduction algorithm allows us to reveal this structure, which then leads to potential parallelism in the form of independent diagonal blocks. Compared with the usual Householder reduction algorithm, the new approach exhibits improved data locality, significantly more scope for parallelism, and the potential to reduce arithmetic complexity by close to 50% for matrices that have only two numerically distinct eigenvalues. The actual improvement depends to a large extent on the number of distinct eigenvalues and a good estimate thereof. However, at worst the algorithm behaves like a successive bandreduction approach to tridiagonalization. Moreover, we provide a numerically reliable and effective algorithm for computing the eigenvalue decomposition of a symmetric matrix with two numerically distinct eigenvalues. Such matrices arise, for example, in invariant subspace decomposition approaches to the symmetric eigenvalue problem.

1 Introduction

Let A be an $n \times n$ symmetric matrix. Our goal is to compute an orthogonal-tridiagonal decomposition of A, $AQ = QT$, where Q is orthogonal and T is tridiagonal. Reduction to tridiagonal form is a standard preprocessing step in dense eigensolvers based on QR iteration, bisection, or Cuppen's method [16]. The conventional tridiagonalization procedure [16, p. 419] reduces A one column at a time through a Householder transformation at a cost of $O(4n^3/3)$ flops for the reduction of A, and an

*This work was supported by the Advanced Research Projects Agency, under contract DM28E04120 and P-95006, and the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

†The work of this author was partially performed while she was a postdoctoral associate at Argonne National Laboratory.

‡All PRISM Working Notes can be retrieved via anonymous ftp from the pub/prism directory at ftp.super.org.

additional $O(4n^3/3)$ flops if the orthogonal matrix is accumulated at the same time. This algorithm employs mainly matrix-vector multiplications and symmetric rank-one updates, which require more memory references than the matrix-matrix operations [9,8,14].

The block tridiagonalization algorithm in [5,15] combines sets of p successive symmetric rank-1 updates into one symmetric rank-p update, at the cost of $O(2pn^2)$ extra flops. As a result, this algorithm exhibits improved data locality and hence is likely to be preferable on cache-based architectures. This block algorithm has been incorporated into the LAPACK library of portable linear algebra codes for high-performance architectures [1, 2]. Parallel versions for distributed-memory machines of the standard algorithm and of the block algorithm are described in [12] and in [13], respectively. A different approach to tridiagonalization is the so-called successive band reduction (SBR) method, which completes the tridiagonal reduction through a sequence of band reductions [10,7]. This approach leads to algorithms that exhibit an even greater degree of memory locality, among other desirable features.

In this paper we show that if the number k (say) of distinct eigenvalues of a symmetric matrix A is small, considerable scope exists for further savings in tridiagonalization algorithms. As will be demonstrated, A can be cheaply reduced to a block diagonal banded form through a slightly modified SBR approach. The final tridiagonal form is then achieved by applying the algorithm recursively on the subblocks on the diagonal. Compared with the conventional approach, this approach has the following advantages.

Improved data locality: The tridiagonalization process can employ mainly matrix-matrix operations, both in the reduction of A and in the update of the transformation matrix Q (see also [10,7]).

Enhanced scope for parallelism: In the traditional algorithm, the scope for the exploitation of parallelism in the reduction of A is limited to the application of the rank-1 update (for the unblocked algorithm) or the rank-p update (for the blocked algorithm), and the scope for parallelism decreases as subproblems become smaller. In contrast, our algorithm generates independent subproblems during the reduction of A, which can be worked on independently and whose number increases as the iteration proceeds. Thus, a shift occurs from data parallelism (updates of large matrices) to functional parallelism (several independent subproblems), but at any stage there is sufficient parallelism to exploit.

Reduced complexity: Depending on the number of distinct eigenvalues, we may almost halve the number of floating-point operations. In addition, the need for data movement is reduced.

One particular situation where repeated eigenvalues arise is in the context of invariant-subspace methods for eigenvalue problems [3,19,6,4], where a matrix with only two distinct, predetermined, eigenvalues is generated either by repeated application of incomplete beta functions [19] or the matrix sign function [4]. In exact arithmetic, our tridiagonalization procedure would result in a block diagonal matrix with diagonal blocks of order no larger than 2. Hence the eigenvalue decomposition could be computed easily by independently diagonalizing the 2×2 blocks on the diagonal. In the presence of roundoff errors, the computed tridiagonal matrix may not have this desirable structure. However, we can prove that such a tridiagonal matrix can be diagonalized as reliably as with any other method by two "cleanup sweeps," where each sweep solves at most $n/2$ independent 2×2 eigenvalue problems.

The paper is organized as follows. We show in Section 2 that, under certain conditions that can be verified easiliy, a banded symmetric matrix with bandwidth b and k distinct eigenvalues is block diagonal with diagonal blocks of order at most bk. In Section 3, we present a reduction algorithm to achieve the desired banded block-diagonal structure, through a slight modification of the conventional band reduction procedure. This approach is then employed to develop a divide-and-conquer tridiagonalization algorithm. An inexpensive algorithm for decoupling invariant subspaces of matrices with eigenvalue clusters at 0 and 1 is given and verified in Section 4. Numerical experiments with a Matlab implementation are reported in Section 5. Lastly, we summarize our results.

2 The Structure of Band Matrices with Repeated Eigenvalues

A tridiagonal matrix whose off-diagonal entries are all nonzero is called *unreduced*. It is well known [18, p. 66] that an unreduced tridiagonal matrix does not have multiple eigenvalues. Consequently, if an $n\times n$ tridiagonal matrix has only $k \ll n$ distinct eigenvalues, it must be block diagonal, and the largest block cannot be larger than $k\times k$. The generalization of this fact to banded matrices underpins the algorithm we propose, yet it is not as straightforward as it might seem.

Assuming that A is an $n\times n$ symmetric matrix, we define the *ith row bandwidth* of A, denoted by band_row(i), as

$$\text{band\_row}(i) \stackrel{\text{def}}{=} \max_j\{i-j \mid j=i \text{ or } j<i \text{ and } a_{ij}\neq 0\}, \quad 1\le i\le n. \tag{1}$$

That is, band_row(i) is the distance of the first nonzero element in row i from the ith diagonal element. Further, we say that A is *nonincreasing in row bandwidth from b* if

$$a(b,1)\neq 0 \text{ and } \text{band\_row}(i)\le \text{band\_row}(i-1), \quad b+1<i\le n. \tag{2}$$

In particular, a banded matrix that is all zero below the bth subdiagonal, and all nonzero on the bth subdiagonal is nonincreasing in row bandwidth from b.

With these definitions, we can now prove the following theorem.

Theorem 1 *Let T be symmetric matrix with k distinct eigenvalues. If T is block diagonal, with each diagonal block nonincreasing in bandwidth from at most b, the size of the largest block cannot exceed kb.*

Proof. Assume T has a diagonal block D of size $p>kb$. By assumption, D is nonincreasing in bandwidth from b; that is, D has $p-b$ rows with their first nonzero elements in different columns to the left of the diagonal. Thus, for any λ, rank$(D-\lambda I)$ is not less than $p-b$.

On the other hand, since $p>kb$ and D has at most k distinct eigenvalues, D has an eigenvalue μ with multiplicity greater than b. Hence, rank$(D-\mu I)$ is less than $p-b$. The contradiction verifies the result of the theorem. ∎

The following example shows the necessity of the "nonincreasing bandwidth" restriction in Theorem 1. Let

$$Q^{\mathrm{T}} = \begin{pmatrix} \xi & \eta & \mu & -\nu & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \beta & 0 & \gamma & \delta \\ 0 & 0 & \nu & \mu & 0 & \alpha & 0 & 0 \end{pmatrix},$$

where $\nu^2+\mu^2+\alpha^2 = 1$, $\xi^2+\eta^2 = \alpha^2$, and $\beta^2+\gamma^2+\delta^2 = 1$. Then Q has orthonormal columns and $A = QQ^{\mathrm{T}}$ is symmetric with only 0 and 1 as eigenvalues. In fact,

$$A = \begin{bmatrix} \times & \times & \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & \times & 0 & 0 \\ \times & \times & 0 & \times & 0 & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & \times & 0 & \times & \times \\ 0 & 0 & \times & \times & 0 & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & \times & 0 & \times & \times \\ 0 & 0 & 0 & 0 & \times & 0 & \times & \times \end{bmatrix}. \tag{3}$$

We see that A is banded with semi-bandwidth $b = 3$, but it is *not* block diagonal with blocks of size at most $2b \times 2b = 6 \times 6$, since the "nonincreasing bandwidth condition" is violated by $a(5,2) = a(7,4) = 0$.

3 A Divide-and-Conquer Tridiagonalization Approach

The example in the preceding section showed that the standard Householder band reduction algorithm will not necessarily reveal the block-diagonal structure. For example, if we had applied the standard algorithm for reduction to bandwidth 3 to the matrix of example (3), the matrix would have remained unchanged. Fortunately, a minor modification of the standard algorithm enforces nonincreasing row-bandwidth, and hence the prerequisites of Theorem 1.

Let us consider the conventional reduction approach, where the matrix is reduced one column at a time to semibandwidth b. In each reduction, the pivot row is always b rows below the diagonal, whether the reduction of the previous column was skipped (i.e., the transformation was an identity) or not. For example, if we reduce the matrix A in (3) to semibandwidth 3, row number 4 is the pivot row for the reduction of the second column, and, since $a(4:8,2) = 0$, this reduction is skipped. We then proceed to column 3, using row 5 as pivot row, and the row-bandwidth increases. If, instead, we employ a Householder transformation acting on $a(4:8,3)$ to eliminate $a(5:8,3)$, keeping row 4 as pivot row, we obtain

$$\tilde{A} = \begin{bmatrix} \times & \times & \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & \times & 0 & 0 & 0 \\ \times & \times & 0 & \times & \times & 0 & 0 & 0 \\ 0 & 0 & \times & \times & \times & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times & \times & \times \end{bmatrix}.$$

Now, $\tilde{A}$ is decoupled into two diagonal blocks of size at most 6×6.

This example shows that nonincreasing bandwidth can be obtained easily if we do not increase the pivot row when the previous reduction was skipped. For computational purposes, we define the row bandwidth with respect to a threshold τ:

$$\text{band\_row}(i,\tau) \stackrel{\text{def}}{=} \max_j \{ i-j \mid j = i \text{ or } j < i \text{ and } \|a(i:n,j)\|_2 > \tau \}, \quad 1 \le i \le n. \tag{4}$$

That is, given a tolerance threshold τ, a column $a(i:n)$ is considered numerically zero if its 2-norm is at most τ. The Matlab function `bred` in Figure 3 shows the conventional bandreduction algorithm augmented with (1) a threshold criterion for the generation of a Householder vector and (2) a modified pivot row selection strategy, which does not change the pivot row if a transformation is skipped.

The subroutines `gen_hh`, `pre_hh`, `post_hh`, and `sym_hh` generate a Householder vector and apply it from the left, right, and symmetrically, respectively. Note that for simplicity the algorithm presented here does not exploit the symmetry of A. However, if we wish to do so, we have `sym_hh` work only with a triangular part of A and omit the `post_hh` (`pre_hh`) call when working only with the lower (upper) triangle. We also note that all the algorithms presented in this paper are available via anonymous ftp from the `pub/prism` directory at `ftp.super.org`.

If no transformations are skipped, the procedure is identical to the conventional band reduction procedure; otherwise, it may terminate earlier when the reduction reaches the last column of the first diagonal block, and the problem is decoupled. Since we drop pivot columns whose norm is $O(\tau)$, the decomposition will be accurate up to a residual of order τ.

For simplicity we omitted an optimization in Figure 3. If the reduction of the first column of A results in a bandwidth $\tilde{b}$, say, where $\tilde{b} < b$, due to the small size of entries $a(\tilde{b}+1:n,1)$, we can directly pursue a reduction of the trailing block to nonincreasing bandwidth $\tilde{b}$, in the same fashion as shown above.

If the parameter b is chosen such that $kb < n$, where k is the number of distinct eigenvalues of A, Theorem 1 predicts a decoupling of the problem, with the leading block being of size no larger than kb. In particular, if b is chosen such that $kb = n/2$, we can expect `bred` to generate two decoupled subproblems of about the same size. We can then recursively divide the problem until the transformed matrix becomes tridiagonal (i.e., $b = 1$). Figure 3 is a serial implementation of tridiagonalization based on this approach. Note that the various subproblems can be dealt with independently and simultaneously. The subroutine `blk_diag`, which is called in `tri_sbr`, is shown in Figure 3 and reduces a matrix to block diagonal form with a given bandwidth.

For example, if we reduce a 12×12 matrix A with only two eigenvalues to bandwidth 3, no diagonal block can be larger than 6×6. Thus, if $a(4,1)$, $a(5,2)$, and $a(6,3)$ are all nonzero after the reductions in the first three columns have been completed, the next three columns must already be reduced, and the (partially reduced) matrix A is of the form

$$\begin{bmatrix}
\times & \times & \times & \times & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\times & \times & \times & \times & \times & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\times & \times & \times & \times & \times & \times & 0 & 0 & 0 & 0 & 0 & 0 \\
\times & \times & \times & \times & \times & \times & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \times & \times & \times & \times & \times & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \times & \times & \times & \times & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times & \times & \times & \times \\
0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times & \times & \times & \times \\
0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times & \times & \times & \times \\
0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times & \times & \times & \times \\
0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times & \times & \times & \times \\
0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times & \times & \times & \times
\end{bmatrix}.$$

As a result, we do not need to perform the reductions that would otherwise have occurred in columns 4 through 6. The complexity of the algorithm for the case $k = 2$ is $O(0.55\,n^3)$ for the reduction of A, and $O(1.25\,n^3)$ for the update of Q, as compared with $O(4n^3/3)$ for both these operations in the usual approach. The savings for Q are minor, since updates at later stages still involve vectors of length n, whereas only diagonal subblocks are affected in A. In addition, we can work in parallel on independent problems. If the estimate k of the number of distinct eigenvalues is inaccurate, the algorithm becomes either the standard eigenvalue algorithm (for $k > n/2$) or the SBR tridiagonalization procedure suggested in [10], but in either case, it will return numerically accurate results.

4 Invariant Subspace Splitting

The computational cost and the degree of parallelism in the algorithm depend on k, the number of distinct eigenvalues. One particularly intriguing case is matrices that have only two eigenvalues, which arise in eigensolvers based on variant subspace decompositions [3, 19, 4]. We may assume without loss of generality that the eigenvalues are at 1 and 0 (any other two eigenvalues can be mapped to 0 and 1 by shifting and scaling). The following corollary is a special case of Theorem 1.

Corollary 2 *Let A be a matrix with two distinct eigenvalues, and let $A = Q^T T Q$ be a tridiagonalization of A. Then T is block diagonal with diagonal blocks of size at most 2×2.*

Corollary 2 implies that one can determine the range space, $\mathcal{R}(A)$, and the null space, $\mathcal{N}(A)$, in essence by tridiagonalizing A. Let $AQ = QT$ be the orthogonal-tridiagonal decomposition of A. For a 1×1 diagonal block $T(j, j)$,

$$Q(:, j) \in \mathcal{R}(A) \text{ if } T(j, j) = 1, \quad \text{and} \quad Q(:, j) \in \mathcal{N}(A) \text{ if } T(j, j) = 0.$$

Since the eigenvalues of A and T are the same, a 2×2 diagonal block $T(j{:}j+1, j{:}j+1)$ must have eigenvalues 0 and 1. Since the trace is the sum of the eigenvalues, and the offdiagonal entry is nonzero, we have

$$T(j{:}j+1, j{:}j+1) = \begin{pmatrix} 1-\gamma & \mu \\ \mu & \gamma \end{pmatrix},$$

where $\mu \neq 0$ and $0 < \gamma < 1$. Since

$$\begin{pmatrix} 1-\gamma & \mu \\ \mu & \gamma \end{pmatrix} = \frac{1}{\gamma} \begin{pmatrix} \mu & \gamma \\ \gamma & -\mu \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \mu & \gamma \\ \gamma & -\mu \end{pmatrix}^{\mathrm{T}},$$

we conclude

$$Q\,(:, j{:}j+1) \begin{pmatrix} \mu \\ \gamma \end{pmatrix} \in \mathcal{R}(A), \quad \text{and} \quad Q\,(:, j{:}j+1) \begin{pmatrix} \gamma \\ -\mu \end{pmatrix} \in \mathcal{N}(A).$$

One can see that the separation of the range and null subspaces of A, and in fact its eigenvalue decomposition, can be affected by diagonalizing (potentially in parallel) the 2×2 subproblems still occurring in the block tridiagonal decomposition.

In the presence of rounding errors, a computed tridiagonal matrix may, however, not exhibit the block structure we could expect from Corollary 2, because of perturbations in the eigenvalues. That is, $\lambda(T) \subset \{[-\nu, \nu] \cup [1-\nu, 1+\nu]\}$, and a repeated eigenvalue numerically manifests itself as an eigenvalue cluster.

```
function [A, block1, Q] = bred( A, b, tau, Q );

%    Given a symmetric matrix A, a bandwidth b, and a threshold tau, bred
%    computes an orthogonal-banded matrix decomposition,
%           A_input * W = W * A_output + O(tau)
%    where O(tau) denotes a matrix with a two-norm of order tau, and
%    W is an orthogonal matrix.
%    The output matrix A_output will be a 2x2 block diagonal matrix,
%    where the first diagonal block A_output(1:block1,1:block1)
%    is banded with bandwidth nonincreasing from b, and the second block
%    may be empty.

[ m, n ] = size(A); if (m~=n) error('nonsquare A'); end
piv_row  = min(b+1,n);      % current pivot row
if (piv_row == n) block1 = n; return; end;
for j = 1:n-b
          % matrix is decoupled, stop
      if (piv_row == j), break, end
          % row and column sets involved in current transformation
      rows = (piv_row : n); cols = (j+1:piv_row-1);
          % generate HH matrix to annihilate A(piv_row+1:n,j)
      [ v, beta, gamma ] = gen_hh( A( rows, j), tau );
          % update jth row and column of A
      A( rows, j) = zeros(size(rows')); A(piv_row, j) = gamma;
      A( j, rows) = zeros(size(rows));  A(j, piv_row) = gamma;
          % if the reduction is not "skipped", perform symmetric
          % update of A, update Q if required, and shift the pivot row
      if ( beta ~= 0)
        if( cols~= [] )
           A(rows, cols) =  pre_hh( beta, v, A(rows, cols) );
           A(cols, rows) = post_hh( beta, v, A(cols, rows) );
        end
        A( rows, rows ) = symm_hh( beta, v, A(rows, rows) );
        if( Q ~= [] ), Q(:, rows) = post_hh( beta, v, Q(:, rows ) ); end
      end % beta
         % increase pivot row if A(piv_row,j) is not negligible
      if (abs(A(j,piv_row)) > tau), piv_row = piv_row + 1; end
end % j-loop
if (j == n - b)
   if (piv_row == j+1), block1 = piv_row - 1; else, block1 = n; end
else
   block1 = piv_row-1;
end
return; end
```

Figure 1: Nonincreasing Row–Bandwidth–Preserving Bandreduction Algorithm

```
function [A, Q] = tri_sbr( A, k, tau, Q )
%
%  produces an orthogonal-tridiagonal decomposition of
%  a symmetric matrix A such that
%            A_old*Q = Q*A_new + O(tau)
%  where A_new is tridiagonal and Q is orthogonal.
%
%  The number k is a guess at the number of numerically distinct
%  eigenvalues of A.
%
%  Matrices are successively reduced to smaller bandwidth in an
%  attempt to exploit the divide-and-conquer nature becoming
%  apparent in the successive bandreduction algorithm when the number
%  k chosen is a good guess at the actual number of numerically distinct
%  eigenvalues.

[m, n] = size(A); if( m ~= n ) error('non-square A'); end

b = max( floor(n/(2*k)), 1 );

[A, block1, Q] = bred( A, b, tau, Q );

if (block1 == n) % If problem didn't decouple, just reduce to
                 % tridiagonal form
   [A,blkvec,Q] = blk_diag(A,1,tau,Q); return;
else
 if( b > 1 )     % first subproblem is not tridiagonal yet
   sub = 1:block1; V = eye(block1);
   [ A(sub,sub), V ] = tri_sbr( A( sub, sub), k, tau, V );
   Q(:,sub) = Q(:,sub) * V;
 end;
 if( n-block1 > 2 )  % second subproblem is nontrivial
   sub = (block1+1):n; V = eye(n-block1);
   [ A(sub, sub), V ] = tri_sbr( A(sub, sub), k, tau, V );
   Q(:,sub) = Q(:,sub) * V;
 end
end

return;
end
```

Figure 2: Divide–and–Conquer Tridiagonalization

```
function [ A, blkvec, Q ] = blk_diag( A, b, tau, Q )
%
% Given a symmetric matrix A, a desired bandwidth b, and a threshold tau,
%          [ A, bvec, Q ] = blk_diag( A, b, tau, Q )
% produces an orthogonal-block-diagonal decomposition
%           A_input * W = W * A_output + O(tau)
% where O(tau) denotes a matrix whose norm is of order tay, and
% W is an orthogonal matrix.
%
% A_output will be a block diagonal matrix with each block banded with
% nonincreasing bandwidth b. The i-th diagonal block starts
% at (blkvec(i), blkvec(i)).
%
% If Q is not the empty matrix on input, Q is postmultiplied by W,
%    i.e.,, Q_output = Q_input * W.

[m, n] = size(A); if( m ~= n ) error('non-square A'); end

j = 1; blkvec = [];
while( j < n )
   blkvec = [ blkvec j ]; rows = j:n; cols = j:n;
   [A(rows, cols), dj, Q(:,cols) ] = bred( A(rows,cols), b, tau, Q(:,cols);
   j = j + dj;
end

return; end
```

Figure 3: Reduction to Block Diagonal Form

Example 3 *The matrix*

$$T = \begin{pmatrix} 1 & e_1 & & & & & \\ e_1 & 0 & e_2 & & & & \\ & e_2 & 1 & e_3 & & & \\ & & e_3 & 0 & e_4 & & \\ & & & e_4 & \ddots & & \\ & & & & & 1 & e_{n-1} \\ & & & & & e_{n-1} & 0 \end{pmatrix},$$

where $e_j = O(\sqrt{\epsilon})$*, has eigenvalues* $\lambda(T) \subset \{[-\nu, \nu] \cup [1-\nu, 1+\nu]\}$ *with* $\nu = O(\epsilon)$*.*

Hence, it seems that, for numerically relevant computations, we now would be faced with computing the eigenvalue decomposition of a tridiagonal matrix. This is not the case, however. By exploiting the special structure of the tridiagonal matrix, we can diagonalize it in two "sweeps" that compute the eigendecomposition of all "even" or "odd" 2×2 blocks on the diagonal (simultaneously), respectively. As we show below, the fill-ins generated by these sweeps are of the same order as the perturbation in the eigenvalues and hence can be considered negligible.

Lemma 4 *Let* T *be a symmetric tridiagonal matrix with*

$$\lambda(T) \subset [-\nu, \nu] \cup [1-\nu, 1+\nu],$$

, where $\nu \stackrel{\text{def}}{=} \max_{\lambda \in \lambda(T)}\{\min(|\lambda - 1|, |\lambda|)\} \ll 1$*. Then* $\|T^2 - T\|_2 \leq \bar{\nu}$ *where*

$$\bar{\nu} \stackrel{\text{def}}{=} \nu + \nu^2. \tag{5}$$

Proof. Let Q be orthogonal and $E = \text{diag}(E_1, E_0)$ be diagonal, respectively, such that

$$T = Q \begin{pmatrix} I + E_1 & \\ & E_0 \end{pmatrix} Q^T.$$

Then, $\|E\|_2 = \nu$, and

$$T^2 = T + Q\left(\begin{pmatrix} E_1 & \\ & -E_2 \end{pmatrix} + E^2\right) Q^T.$$

Thus, $\|T^2 - T\|_2 \leq \|\,|E| + E^2\|_2 = \bar{\nu}$. ∎

The next lemma gives bounds on the elements of the Givens rotation we will choose to diagonalize a 2×2 block and minimize the size of fill-ins.

Lemma 5 *Let* $G = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}$ *be a Givens rotation that diagonalizes a* 2×2 *symmetric matrix* $\begin{pmatrix} \alpha_1 & \beta \\ \beta & \alpha_2 \end{pmatrix}$*. Assume without loss of generality that* $\beta > 0$*, and define* $\sigma \geq 0$ *by*

$$\sigma^2 \stackrel{\text{def}}{=} \left(\frac{\alpha_1 - \alpha_2}{2}\right)^2 + \beta^2. \tag{6}$$

Then, s and c can be chosen such that

$$0 \le |s| \le \frac{\beta}{\sqrt{2}\,\sigma} \quad \text{and} \quad \frac{1}{\sqrt{2}} \le c \le 1.$$

Proof. Let $c = \cos(\theta)$ and $s = \sin(\theta)$. Since we wish to eliminate the off-diagonal elements in $G\begin{pmatrix} \alpha_1 & \beta \\ \beta & \alpha_2 \end{pmatrix}G^{\mathrm{T}}$, we obtain

$$0 = (c^2 - s^2)\beta + 2cs\left(\frac{\alpha_2 - \alpha_1}{2}\right) = \beta\cos(2\theta) - \left(\frac{\alpha_2 - \alpha_1}{2}\right)\sin(2\theta).$$

If we choose

$$\cos(2\theta) = \frac{|\alpha_1 - \alpha_2|}{2\sigma}, \tag{7}$$

with σ as defined in (6), then

$$s^2 = \frac{1 - \cos(2\theta)}{2} = \frac{\beta^2}{2\sigma(\sigma + |\alpha_1 - \alpha_2|/2)},$$

and

$$c^2 = \frac{1 + cos(2\theta)}{2} = \frac{\sigma + |\alpha_1 - \alpha_2|/2}{2\sigma}$$

as claimed. ■

In the following theorem we now show that, employing these Givens rotations, we can limit the size of the fill-in entries generated when applying these rotations to a tridiagonal matrix with eigenvalue clusters around 0 and 1.

Theorem 6 *Let T and $\bar{\nu}$ be as in Lemma 4. Let $G = \mathrm{diag}(I, \begin{pmatrix} c & s \\ -s & c \end{pmatrix}, I)$ be the Givens rotation that diagonalizes one 2×2 diagonal block of T, namely,*

$$G \cdot \begin{pmatrix} \ddots & \overline{\beta} & & & \\ \overline{\beta} & \alpha_1 & \beta & & \\ & \beta & \alpha_2 & \underline{\beta} & \\ & & \underline{\beta} & \ddots & \end{pmatrix} \cdot G^T = \begin{pmatrix} \ddots & * & \gamma & \\ * & \tilde{\alpha_1} & 0 & \delta \\ \gamma & 0 & \tilde{\alpha_2} & * \\ & \delta & * & \ddots \end{pmatrix},$$

where we assume $\beta > 0$ without loss of generality. If $\beta > \sqrt{7}\,\bar{\nu}$ and c and s are chosen as suggested in Lemma 5, then

$$\gamma \le \sqrt{7}\,\bar{\nu} \quad \text{and} \quad \delta \le \sqrt{7}\,\bar{\nu}.$$

Proof. Comparing corresponding entries in T^2 and T and invoking Lemma 4, we know that there exist $\overline{\epsilon}$, $\underline{\epsilon}$, and ϵ_o, $|\overline{\epsilon}|, |\underline{\epsilon}|, |\epsilon_o| \le \bar{\nu}$, such that

$$\beta(\alpha_1 + \alpha_2) = \beta + \epsilon_o \tag{8}$$

$$\overline{\beta}^2 + \alpha_1^2 + \beta^2 = \alpha_1 + \overline{\epsilon} \tag{9}$$

$$\beta^2 + \alpha_2^2 + \underline{\beta}^2 = \alpha_2 + \underline{\epsilon} \tag{10}$$

$$\overline{\beta}\beta \le \bar{\nu}, \quad \underline{\beta}\beta \le \bar{\nu}. \tag{11}$$

Using these identities, we have

$$\beta^2 - \alpha_1\alpha_2 = \frac{\alpha_1+\alpha_2}{2}(1-(\alpha_1+\alpha_2)) + \frac{(\underline{\epsilon}+\overline{\epsilon}) - (\underline{\beta}^2+\overline{\beta}^2)}{2},$$

and hence we can express σ^2 defined as in (6) as

$$\begin{aligned}\sigma^2 &= (\frac{\alpha_1+\alpha_2}{2})^2 + (\beta^2 - \alpha_1\alpha_2) \\ &= \frac{1}{4}(1-\frac{\epsilon_o^2}{\beta^2}) + \frac{\overline{\epsilon}+\underline{\epsilon}}{2} - \frac{(\overline{\beta}\beta)^2 + (\underline{\beta}\beta)^2}{2\beta^2}\end{aligned}.$$

Thus,

$$\sigma^2 \geq \frac{1}{4} - \bar{\nu} - \frac{5}{4}(\frac{\bar{\nu}}{\beta})^2.$$

Now let $\tau \geq 1$ be chosen such that $\beta > \tau\bar{\nu}$. Then

$$\sigma^2 \geq \frac{1}{4} - \bar{\nu} - \frac{5}{4}(\frac{1}{\tau})^2. \tag{12}$$

Equations (11) together with $s \leq \dfrac{\beta}{\sqrt{2}\sigma}$ imply

$$\gamma = s\overline{\beta} \leq \frac{\bar{\nu}}{\sqrt{2}\sigma} \text{ and } \delta = s\underline{\beta} \leq \frac{\bar{\nu}}{\sqrt{2}\sigma}.$$

Using (12), we can easily show that $\tau \geq \sqrt{7}$ implies $\dfrac{1}{\sqrt{2}\sigma} \leq \tau$ and hence the result of the theorem. ∎

As a consequence of Theorem 6, we are able to compute the eigenvalue decomposition of a 2×2 diagonal block in a tridiagonal matrix T with eigenvalue clusters at 0 and 1 such that the generated fill-in is negligible compared with the eigenvalue perturbation. Thus, the diagonalization of T can be done by two sweeps of (potentially concurrent) 2×2 eigenvalue problems as shown in Figure 4. In the first sweep, we diagonalize an "odd-even" 2×2 problem if the off-diagonal entry is not too small, and set the fill-in entries to zero, or otherwise just set the off-diagonal entry to zero to zero. In the second sweep, we diagonalize the "even-odd" blocks. Since no more rotations follow, there is no need to set the fill-in entries to zero.

Theorem 6 shows that the Frobenius norm of the fill-in matrix introduced by the algorithm `rr_diag`shown in Figure 4 is bounded by $3\sqrt{n}\bar{\nu}$, which is of the same order as the perturbation in eigenvalues. The subroutine `diag2`, which is not shown here, computes the diagonalizing rotations as outlined in Lemma 5. Hence, Algorithm `rr_diag` is as numerically reliable as any other approach for diagonalizing T, albeit much cheaper because it exploits the special structure of T.

5 Numerical Experiments

In this section we report on some numerical experiments with the algorithms presented in this paper. All experiments were performed with Matlab Version 4.2a on a Sun Sparcstation iPX. For

```
function [Q, D] = rr_diag( A, Q, tau )
%
% Given a tridiagonal matrix A with eigenvalues 1 and 0, with
% lambda(A) contained in [1-tau,1+tau] or [-tau,tau]
% rr_diag computes an approximate eigendecomposition
%        D = Q' * A * Q
% where
%       || D - Q'* A * Q||_Frobenius <= sqrt(7*n)*tau*(1+tau)

[m,n] = size(A); if( m~=n ) error('non-square A'); end
drop_threshold = sqrt(7)*tau*(1+tau);

for j = 1:2:floor(n/2)*2          % diagonalize all (odd-even)
  k = j:j+1;                      % diagonal 2x2 matrices
  if (abs(A(j+1,j)) > drop_threshold)
    [G A(k,k)] = diag2( A(k,k) );
    if( j+2 <= n )
      A(j+2,k) = A(j+2,k)*G; A(k,j+2) = G'*A(k,j+2);
      A(j+2,j) = 0; A(j,j+2) = 0; % zero out negligible fill-ins
    end
    if( j-1 >= 1 )
      A(j-1,k) = A(j-1,k)*G; A(k,j-1) = G'*A(k,j-1);
      A(j-1,j+1) = 0; A(j+1,j-1) = 0;
    end
    Q(:,k) = Q(:, k)*G;
  end
end
for j = 2:2:floor((n-1)/2)*2       % diagonalize all (even-odd)
  k = j:j+1;                       % diagonal 2x2 matrices
  if (abs(A(j+1,j)) > drop_threshold)
    [G A(k,k)] = diag2( A(k,k) );
    if( j+2 <= n )
      A(j+2,k) = A(j+2,k)*G; A(k,j+2) = G'*A(k,j+2);
   % no more need to zero fill-ins
    end
    if( j-1 >= 1 )
      A(j-1,k) = A(j-1,k)*G; A(k,j-1) = G'*A(k,j-1);
    end
    Q(:,k) = Q(:, k)*G;
  end
end
D = diag(diag(A));
return; end
```

Figure 4: Diagonalization of a Tridiagonal Matrix with Eigenvalue Clusters at 0 and 1

readers wishing to experiment on their own, the Matlab files employed to generate these results can be retrieved via anonymous ftp from the `pub/prism` directory at `ftp.super.org`.

First, we apply the bandreduction algorithm `bred` of Figure 3 recursively to the trailing subblock of a 200×200 matrix with two eigenvalue clusters of size 50 each at $\lambda = \{-1, -2, 0, 1\}$. The radius of each cluster is $\epsilon\, 1.0e^3$, where ϵ is the machine precision. The drop threshold `tau` in `bred` is set to $\sqrt{7}\,\epsilon\, 1.0e^3$, and at each step the bandwidth is chosen so as to decouple the problem in the middle. The succession of matrices generated is shown in Figure 5. The title of each picture shows the current matrix size being worked on and the bandwidth to which it is to be reduced. At each step, we compute the residual

$$\delta \stackrel{\text{def}}{=} \|A_{original} * Q - Q * A_{current}\|_2.$$

We observe $\delta \approx 7.2e^{-13}$, which, given a machine precision $\epsilon = 2.2e^{-16}$, is consistent with our theory.

The same experiment, employing a matrix with 100 eigenvalues at 0 and 1 each, with the same eigenvalue perturbation and drop threshold, is shown in Figure 6. Note that it is sufficient to reduce the matrix to half the bandwidth chosen in Figure 5 to achieve decoupling. We observe $\delta \approx 2.7e^{-13}$. We also note that in both cases, the first, third, and fourth splits occur at row (and column) 100, 176, and 188, respectively. The second split occurs at row 152 for Figure 5 and at row 150 for Figure 6.

To test the behavior of our rank-revealing tridiagonalization (RRDG), we compare it with the standard eigenvalue decomposition (EIG) and the QR factorization with column pivoting (QR). Our test matrices are

1. tridiagonal matrices with eigenvalue clusters of radius $p\,\epsilon$ generated by inserting random off-diagonal perturbations of the order $\sqrt{p\,\epsilon}$ in the matrix shown in Example 3, and
2. matrices generated by symmetrically multiplying the matrices from Example 3 with orthogonal matrices generated via the QR factorization of a random matrix.

In the first case, we call `rr_diag` shown in Figure 4, in the second case, we precede the call to `rr_diag` by a call to `tri_sbr` as shown in Figure 3. The drop threshold for the divide-and-conquer tridiagonalization is set to $\sqrt{7}\,p\,\epsilon$, which is the same threshold employed in the two final diagonalization sweeps. For each of $p = 1, 10, 100$, we run 50 test cases each with matrix sizes 125, 250, and 375. RRDG and EIG both compute an eigenvalue decomposition $Q^T AQ = D$, with D diagonal. We compute $\tilde{D} \stackrel{\text{def}}{=} \text{round}(D)$, that is, round each diagonal entry to the nearest integer, and we report both the relative eigenvalue residual $\|Q^T A - \tilde{D}Q\|_F/\sqrt{n/2}$ as well as the relative orthogonality residual $\|Q^T Q - I\|_F/\sqrt{n}$. Note that $\sqrt{n/2}$ is an estimate of $\|A\|_F$. In the case of the QR factorization with pivoting, which computes $AP = QR$ for a permutation matrix P and an upper triangular matrix R, we compute the rank

$$r \stackrel{\text{def}}{=} \max_i |r_{ii}| > \sqrt{7}\,p\,\epsilon$$

and $\tilde{A} \stackrel{\text{def}}{=} Q^T * A * Q$. We then report

$$|\,\|\tilde{A}(1:r, 1:r)\|_F - \|A\|_F\,|/\sqrt{n/2},$$

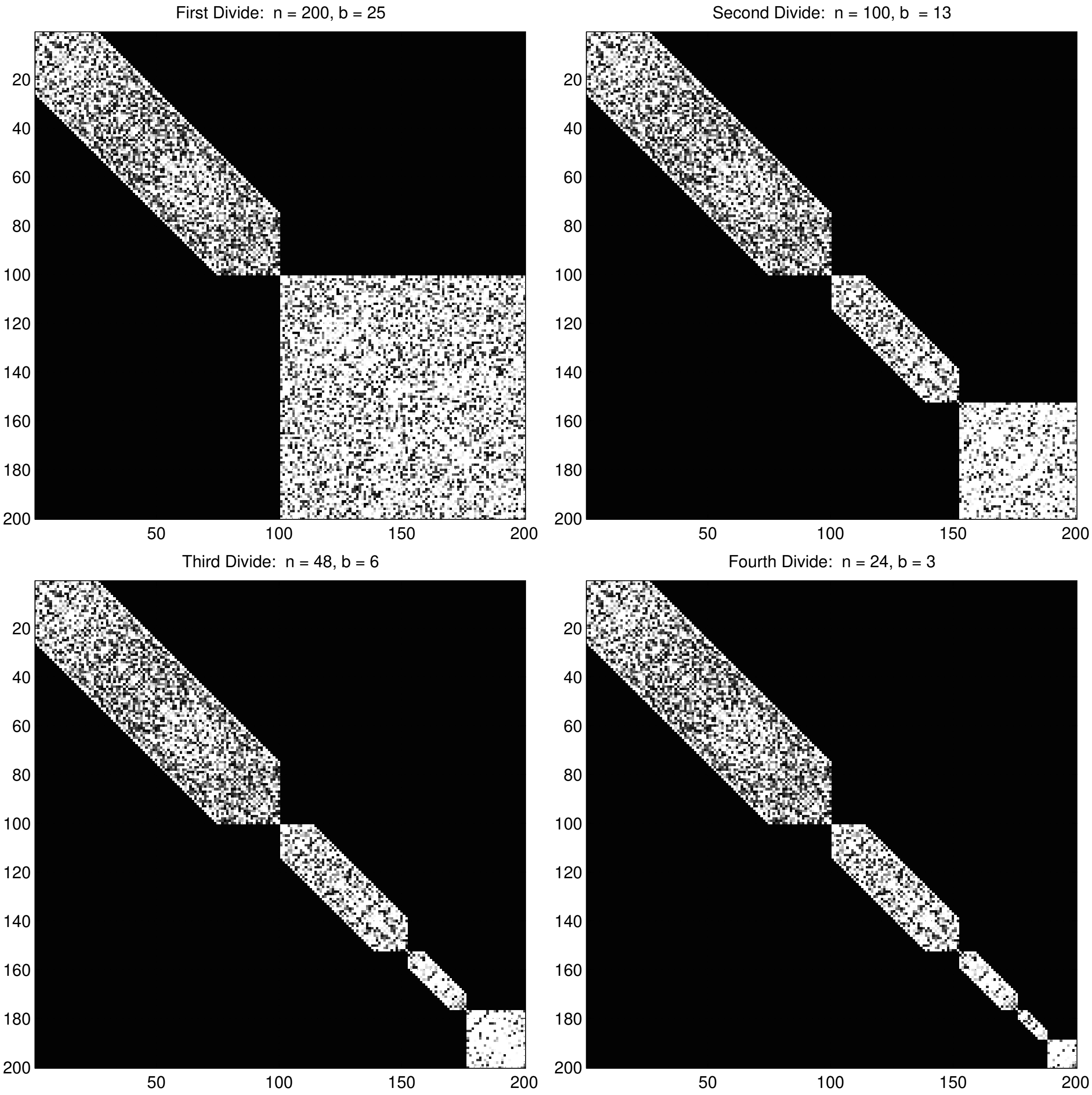

Figure 5: Bandreduction Applied to Trailing Subblock of a 200×200 Matrix with Four Distinct Eigenvalue Clusters

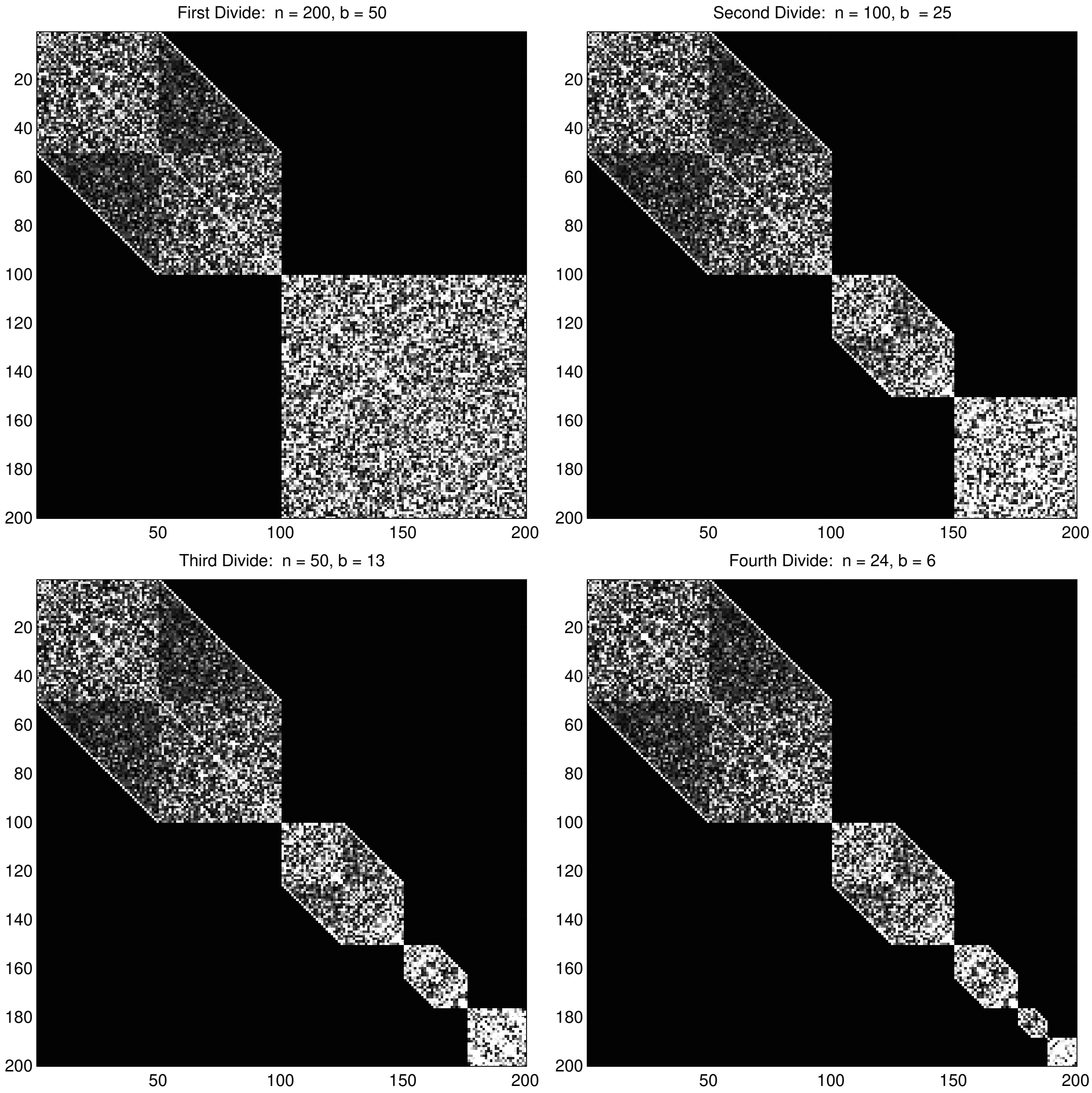

Figure 6: Bandreduction Applied to Trailing Subblock of a 200×200 Matrix with Two Distinct Eigenvalue Clusters

Table 1: Relative Residual in Subspace Splitting

Tridiagonal Matrices

n	$\mathrm{RRDG}_{\max}$	$\mathrm{EIG}_{\max}$	$\mathrm{QR}_{\max}$
$p = 1$			
125	5.3e-16	1.6e-15	1.7e-15
250	5.0e-16	1.6e-15	3.8e-15
375	4.9e-16	1.5e-15	5.6e-15
$p = 10$			
125	3.5e-15	4.2e-15	2.2e-15
250	3.3e-15	4.9e-15	5.1e-15
375	3.4e-15	4.5e-15	4.3e-15
$p = 100$			
125	3.3e-14	3.3e-14	2.7e-15
250	3.2e-14	3.2e-14	6.8e-15
375	3.2e-14	4.4e-14	6.6e-15
$p = 1000$			
125	3.3e-13	3.3e-13	2.5e-15
250	3.2e-13	3.2e-13	4.1e-15
375	3.2e-13	3.2e-13	6.2e-15

Full Matrices

n	$\mathrm{RRDG}_{\max}$	$\mathrm{EIG}_{\max}$	$\mathrm{QR}_{\max}$
$p = 1$			
125	5.3e-14	1.7e-14	1.4e-14
250	1.5e-13	3.3e-14	3.7e-14
375	2.4e-14	3.8e-14	5.5e-14
$p = 10$			
125	5.0e-15	6.0e-15	1.6e-14
250	5.5e-15	3.0e-14	4.0e-14
375	6.1e-15	4.1e-14	4.8e-14
$p = 100$			
125	4.6e-14	3.5e-14	1.4e-14
250	4.5e-14	5.2e-14	3.9e-14
375	4.2e-14	3.2e-14	4.9e-14
$p = 1000$			
125	4.6e-13	3.5e-13	1.6e-14
250	4.4e-13	3.4e-13	3.6e-14
375	4.2e-13	3.2e-13	4.2e-14

Table 2: Relative Residual in Orthogonality

Tridiagonal Matrices

n	$\mathrm{RRDG}_{\max}$	$\mathrm{EIG}_{\max}$	$\mathrm{QR}_{\max}$
$p=1$			
125	2.3e-16	1.2e-15	1.1e-15
250	2.2e-16	1.3e-15	1.3e-15
375	2.1e-16	1.2e-15	1.3e-15
$p=10$			
125	3.0e-16	2.8e-15	1.1e-15
250	2.8e-16	3.0e-15	1.4e-15
375	2.8e-16	2.8e-15	1.6e-15
$p=100$			
125	3.4e-16	1.1e-14	1.3e-15
250	3.2e-16	2.0e-14	1.4e-15
375	3.1e-16	1.9e-14	1.7e-15
$p=1000$			
125	3.2e-16	1.0e-14	1.2e-15
250	3.1e-16	2.3e-14	1.4e-15
375	3.2e-16	3.3e-14	1.6e-15

Full Matrices

n	$\mathrm{RRDG}_{\max}$	$\mathrm{EIG}_{\max}$	$\mathrm{QR}_{\max}$
$p=1$			
125	2.1e-15	1.2e-14	1.7e-15
250	3.0e-15	2.4e-14	2.4e-15
375	3.6e-15	2.7e-14	2.8e-15
$p=10$			
125	1.4e-15	1.1e-14	1.7e-15
250	1.9e-15	2.1e-14	2.3e-15
375	3.4e-15	2.9e-14	2.9e-15
$p=100$			
125	1.4e-15	1.1e-14	1.7e-15
250	1.9e-15	2.2e-14	2.4e-15
375	2.3e-15	2.6e-14	2.9e-15
$p=1000$			
125	1.4e-15	1.3e-14	1.8e-15
250	1.9e-15	2.4e-14	2.4e-15
375	2.3e-15	3.3e-14	2.9e-15

which should be small, since $Q(1:r,:)$ is a basis for the range space of A. For each case, we report the worst residual.

We see that the divide-and-conquer tridiagonalization, followed by the two cleanup sweeps over the resulting tridiagonal matrix, performs just as well as a full-fledged eigenvalue decomposition. In both cases, the residual in the subspace splitting is of $O(p\,\epsilon)$, as expected. The residual for QR factorization does not include the perturbation at the eigenvalue 1, as do the other two approaches, and therefore is smaller in all cases. In any case, the computed orthogonal matrices are orthogonal up to machine precision. The Q computed by the `eig` function in Matlab is slightly less orthogonal, since `eig` involves more transformations and as a result accumulates more rounding errors. Note that all three approaches are worse for a full matrix in the case $p = 1$ because the roundoff errors in the orthogonal reductions are of the same order of machine precision. When p is bigger, the roundoff errors are dominated by the perturbation in the eigenvalues, and hence RRDG and EIG behave about the same for tridiagonal and full matrices.

6 Conclusions

This paper introduced an algorithm for reducing a symmetric matrix with repeated eigenvalues to tridiagonal form. The algorithm progresses through a series of band reductions, each band reduction stage forcing a decoupling of the band matrix into independent subblocks. Compared with the usual Householder tridiagonalization procedure, this approach can save up to 50% of the floating-point operations. We also developed a robust and inexpensive numerical procedure for diagonalizing the resulting tridiagonal matrix in the case where the matrix has only two eigenvalue clusters around zero and one. This case arises in eigenvalue decomposition algorithms based on invariant subspace approaches. Taken together, these two algorithms allow for an efficient diagonalization of such matrices.

The algorithm can be generalized immediately to the reduction of unsymmetric matrices to Hessenberg form. The same irreducibility argument underlying Theorem 1 goes through for Hessenberg matrices. We also note that in exact arithmetic, conjugate transposed eigenvalue pairs would end up in the same block. However, since one triangle of a Hessenberg matrix is still full, the potential for computational savings is greatly reduced.

Apart from its divide-and-conquer nature and the resulting potential for parallelism, as well as its reduced operation count, our divide-and-conquer algorithm has another attractive feature. Since our algorithm (at least in the early stages) reduces matrices to banded form with a relatively wide band, it is easy to block the Householder transformations by using the WY representation [11] or the compact WY representation [20], as has, for example, been described in [17]. In this fashion, one can easily capitalize on the favorable memory transfer characteristics of block algorithms.

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. DuCroz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK User's Guide*. SIAM, Philadelphia, 1992.

[2] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. DuCroz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK User's Guide Release 2.0*. SIAM, Philadelphia, 1994.

[3] Louis Auslander and Anna Tsao. A divide-and-conquer algorithm for the eigenproblem via complementary invariant subspace decomposition. Technical Report SRC-TR-89-003, Supercomputing Research Center, Institute for Defense Analysis, Bowie, Maryland, 1989.

[4] Z. Bai and J. Demmel. Design of a parallel nonsymmetric toolbox, part i. Technical Report 92-09, Department of Mathematics, University of Kentucky, 1992.

[5] Christian Bischof, George Corliss, and Andreas Griewank. Structured second- and higher-order derivatives through univariate Taylor series. *Optimization Methods and Software*, 2:211–232, 1993.

[6] Christian Bischof, Steven Huss-Lederman, Xiaobai Sun, and Anna Tsao. The PRISM project: Infrastructure and algorithms for parallel eigensolvers. In *Proceedings of the Scalable Parallel Libraries Conference*, pages 123–131, Washington, D.C., 1994. IEEE Computer Society.

[7] Christian Bischof, Xiaobai Sun, and Bruno Lang. Parallel tridiagonalization through two-step band reduction. In *Proceedings of Scalable High Performance Computing Conference*, pages 23–27. IEEE Computer Society Press, May 1994.

[8] Christian H. Bischof. *Fundamental Linear Algebra Computations on High-Performance Computers*, volume 250 of *Informatik Fachberichte*, pages 167–182. Springer Verlag, Berlin, 1990.

[9] Christian H. Bischof and Jack J. Dongarra. A project for developing a linear algebra library for high-performance computers. In Graham Carey, editor, *Parallel and Vector Supercomputing: Methods and Algorithms*, pages 45–56. John Wiley & Sons, Somerset, N.J., 1989.

[10] Christian H. Bischof and Xiaobai Sun. A framework for band reduction and tridiagonalization of symmetric matrices. Preprint MCS-P298-0392, Mathematics and Computer Science Division, Argonne National Laboratory, 1992.

[11] Christian H. Bischof and Charles F. Van Loan. The WY representation for products of Householder matrices. *SIAM Journal on Scientific and Statistical Computing*, 8:s2–s13, 1987.

[12] H. Chang, S. Utku, M Salama, and D. Rapp. A parallel Householder tridiagonalization stratagem using scattered square decomposition. *Parallel Computing*, 6:297–311, 1988.

[13] J. Dongarra and R. van de Geijn. Reduction to condensed form for the eigenvalue problem on distributed-memory architectures. *Parallel Computing*, 18:973–982, 1992.

[14] Jack Dongarra and Sven Hammarling. *Evolution of Numerical Software for Dense Linear Algebra*, pages 297–327. Oxford University Press, Oxford, UK, 1989.

[15] Jack J. Dongarra, Sven J. Hammarling, and Danny C. Sorensen. Block reduction of matrices to condensed form for eigenvalue computations. Technical Report ANL/MCS–TM–99, Mathematics and Computer Science Division, Argonne National Laboratory, September 1987.

[16] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. 2nd ed. The Johns Hopkins University Press, Baltimore, 1989.

[17] Roger G. Grimes and Horst D. Simon. Solution of large, dense symmetric generalized eigenvalue problems using secondary storage. *ACM Transactions on Mathematical Software*, 14(3):241–256, 1988.

[18] Alston S. Householder. *The Theory of Matrices in Numerical Analysis*. Dover Publications, Inc., New York, 1964.

[19] Steven Lederman, Anna Tsao, and Thomas Turnbull. A parallelizable eigensolver for real diagonalizable matrices with real eigenvalues. Technical Report TR-91-042, Supercomputing Research Center, Institute for Defense Analysis, Bowie, Maryland, 1991.

[20] Robert Schreiber and Charles Van Loan. A storage efficient WY representation for products of Householder transformations. *SIAM Journal on Scientific and Statistical Computing*, 10(1):53–57, 1989.