

# Algorithm 841: BHES: Gaussian Reduction to a Similar Banded Hessenberg Form

GARY W. HOWELL

North Carolina State University

and

NADIA DIAA

---

BHES uses Gaussian similarity transformations to reduce a general real square matrix to similar upper Hessenberg form. Multipliers are bounded in root mean square by a user-supplied parameter. If the input matrix is not highly nonnormal and the user-supplied tolerance on multipliers is of a size greater than ten, the returned matrix usually has small upper bandwidth. In such a case, eigenvalues of the returned matrix can be determined by the bulge-chasing BR iteration or by Rayleigh quotient iteration. BHES followed by BR iteration determines a complete spectrum in about one-fifth the time required for orthogonal reduction to Hessenberg form followed by QR iterations. The FORTRAN 77 code provided for BHES runs efficiently on a cache-based architecture.

Categories and Subject Descriptors: G.1.3 [Numerical Analysis]: Numerical Linear Algebra—*Eigenvalues and eigenvectors (direct and iterative methods)*; G.4 [Mathematical Software]—*Efficiency*

General Terms: Algorithms

Additional Key Words and Phrases: Gaussian similarity transformations, matrix eigenvalues, spectra, Hessenberg form, cache-efficient, Sylvester equation

---

## 1. INTRODUCTION

The BHES algorithm reduces a general real square matrix to small-band upper Hessenberg form by Gaussian similarity transformations. This section

---

The first author thanks ORISE/ORAU for summer support in 1994 and 1995. He acknowledges the Hewlett-Packard Corporation, North Carolina State University, and the Florida Institute of Technology for support and use of facilities. This work was supported in part by a grant of computer time from the Department of Defense High Performance Computing Modernization Program at the U.S. Army Engineer Research and Development Center Major Shared Research Center, Information Technology Laboratory, Vicksburg, MS. The author also acknowledges support from NSF grant 0103642—Next Generation Software: Cache Efficient and Parallel Householder Bidiagonalization. Authors' addresses: G. W. Howell, High Performance Computing, North Carolina State University, 2620 Hillsborough Street, Box 7109, Raleigh, NC 27695-1709; email: gary\_howell@ncsu.edu; N. Diaa, 512 Farmington Woods Dr., Cary, NC 27511; email: zananoga@yahoo.com.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 1515 Broadway, New York, NY 10036 USA, fax: +1 (212) 869-0481, or [permissions@acm.org](mailto:permissions@acm.org).

© 2005 ACM 0098-3500/05/0300-0001 \$5.00

2 • G. W. Howell and N. Diaa

compares BHES direct reduction to “look-ahead” Lanczos methods. Section 2 explains why tridiagonalization of general matrices is not stable and can break down. Section 3 shows how allowing increased bandwidth can avoid poorly conditioned row and column eliminations. Section 4 describes the BHES algorithm and Section 5 describes some numerical experiments. Sections 6 and 7 describe a cache-efficient implementation and indicate practical uses of the algorithm. As is seen in Section 7, one can obtain eigenvalues by first applying BHES, then applying the BR bulge-chasing iteration [Geist et al. 1999]. If  $A$  is  $n \times n$ , BHES-BR requires about  $8/3 n^3$  flops compared to around  $16 n^3$  flops for orthogonal reduction to Hessenberg form followed by a bulge-chasing QR iteration.

Direct reduction methods may usefully be compared to Lanczos methods. As in near-tridiagonalization by “look-ahead” Lanczos methods [Bai et al. 1995; Freund et al. 1993; Parlett 1992; Taylor 1982; Ye 1994], BHES improves stability at the expense of allowing the bandwidth of the returned matrix to increase. In exact arithmetic, Lanczos methods form biorthogonal bases of two-sided Krylov subspaces, accessing the original matrix only to perform matrix-vector multiplies. Since short-term recurrences are used, only a few locally biorthogonal basis vectors need to be stored. For “look-ahead” Lanczos methods, extra steps can be taken to avoid breakdown or near-breakdown after a given matrix-vector multiplication. For QMR (Quasi Minimal Residual) [Freund et al. 1993], a “look-ahead” step is performed if a given step would otherwise lose half or more of the significant floating-point digits. In practice, the locally biorthogonal basis vectors for “look-ahead” Lanczos methods lose linear independence after a moderate number of steps [Parlett 1993].

Direct reduction methods such as BHES can also be thought of as Krylov subspace methods. BHES improves stability compared to Lanczos methods by bounding multipliers and by using all basis vectors in the recurrence. As in recent algorithms for direct tridiagonalization [Dongarra et al. 1992; Geist 1991; Geist et al. 1989], alternating elimination of rows and columns often allows the one available permutation for a row-column elimination to be chosen so that neither row nor column elimination has large multipliers.

BHES eliminates the first  $n - 2$  columns sequentially, attempting to pair with the  $k$ th column elimination the elimination of any nonzero row, rows  $1, \dots, k$ . If no paired row-column elimination satisfies the user-specified tolerance on multipliers, then only the column elimination is performed. When no row is eliminated, the maximal column value is permuted to the subdiagonal so that the multipliers on this step are bounded by one. Bandwidth increases and the number of rows eligible for elimination on subsequent steps is increased by one. Some other procedures that directly reduce to similar small-band form are found in Howell [1994]; Howell and Geist [1995]; Wachspress (personal communication).

After  $k$  steps a “look-ahead” Lanczos method returns a  $k \times k$  small-band matrix. If  $n$  is the matrix size and  $n$  steps of “look-ahead” Lanczos are performed, then in an exact computation the small-band matrix is similar to the original matrix. Eigenvalues of the  $k \times k$  small-band matrix are called Ritz values and

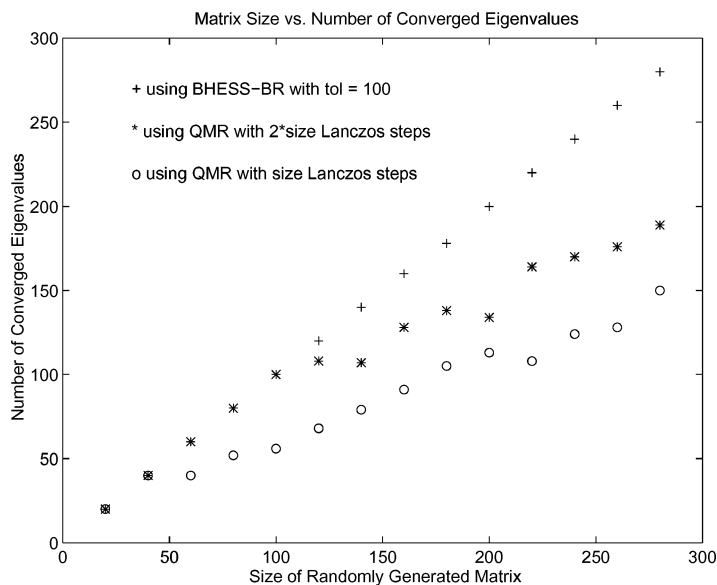


Fig. 1. Comparison of the number of converged eigenvalues for double precision QMR-QR and double precision BHESS-BR. We compute the number of converged eigenvalues as the number of LAPACK eigenvalues that have a QMR-QR Ritz value or BHESS-BR eigenvalue within a tolerance of  $2 \times 10^{-6}$ . In these experiments, Algorithm BHESS-BR with  $tol = 100$  gets all eigenvalues to within  $2 \times 10^{-6}$  of LAPACK values. Many eigenvalues are not found by QMR-QR with  $2n$  steps.

can be used as approximate eigenvalues of the original matrix. Ritz values tend to cluster around the large isolated eigenvalues of the original matrix.

Figure 1 compares the number of correct eigenvalues from BHESS reduction to the well-known “look-ahead” Lanczos method QMR [Freund et al. 1993]. The matrices in these examples had entries randomly generated from a normal distribution of mean zero and standard deviation one. Even with many Lanczos steps, some eigenvalues remain undetected. For the small-band Hessenberg matrix produced by BHESS for the same example matrices, all eigenvalues agree with those determined by LAPACK. In the experiments shown in Figure 1, the largest upper bandwidth for matrices returned by BHESS was 3 and for those returned by “look-ahead” Lanczos the largest upper bandwidth was 4.<sup>1</sup>

Breakdown in tridiagonalization is examined in the next section. BHESS does not break down, which is to say it always runs successfully to completion. Experience with the “look-ahead” QMR Lanczos method indicates that breakdowns do sometimes occur so that not as many Lanczos steps as requested actually occur. For example, in thousands of trials with more than 300 Lanczos steps applied to pentadiagonal Toeplitz matrices of size  $> 2000$ , breakdowns occurred in about 2 percent of cases.

<sup>1</sup>BR iteration can be used to find eigenvalues of the small-band Hessenberg matrices returned either by BHESS or by QMR. The BR algorithm is discussed briefly in the concluding section of this article and at more length in Geist et al. [1999]. A FORTRAN 77 version of BR iteration is available by anonymous ftp from pub/howell at cs.fit.edu, as is a postscript file of Geist et al. [1999].

4 • G. W. Howell and N. Diaa

As yet, no backward error analysis is available for “look-ahead” Lanczos procedures [Bai 1994]. For BHESS, backward error is bounded as follows [Howell et al. 1997].

**THEOREM 1.1.** *Let  $A$  be an  $n \times n$  matrix and let  $H = fl(Z^{-1}AZ)$  be the Hessenberg matrix produced by the BHESS algorithm in floating-point arithmetic. Then*

$$H = \tilde{Z}^{-1}(A_0 + E) \tilde{Z} \quad (1.1)$$

and

$$\|E\|_2 \leq 13 n \text{cond}_2(Z) |A| \sum_{k=1}^{n-2} \|\tilde{m}_{k+1}\|_2^2 u, \quad (1.2)$$

where  $u$  is the machine precision,  $A_0 = PAP^T$  is  $A$  with rows and columns permuted so that the reduction to banded Hessenberg form proceeds without permutations;  $\tilde{Z}$  is the accumulated similarity transformation expressed without permutations; the vectors  $\tilde{m}$  are the multipliers used to eliminate columns; and  $|A|$  is the maximal element of  $A$  encountered during the repeated overwriting of  $A$  that finally results in  $H$ .

In practice, (1.1) and (1.2) overestimate backward error. An experimental estimate for backward error is

$$\|E\|_\infty \leq n \sqrt{\text{cond}_\infty(Z)} |H| u, \quad (1.3)$$

where  $|H|$  is the maximal element in the returned banded Hessenberg matrix. The routine BHAPC (packaged with BHESS) estimates the  $\infty$  condition number of  $Z$  in  $O(n^2)$  computations.

## 2. BREAKDOWN IN TRIDIAGONALIZATION

Reducing a general matrix to similar small-band form requires nonorthogonal transformations.<sup>2</sup> The BHESS algorithm discussed here uses elementary similarity transformations to eliminate rows and columns. For the elementary transformation to eliminate a column, let  $\mathbf{l}_k^T = (0, \dots, 0, l_{k+1}, \dots, l_n)$  and  $L_k = I - \mathbf{l}_k \mathbf{e}_k^T$ , with  $L_k^{-1} = I + \mathbf{l}_k \mathbf{e}_k^T$ . Multiplying  $A$  on the left by  $L_k$  adds multiples of the  $k$ th row of  $A$  to successive rows: it is a rank-one update  $A \leftarrow A - \mathbf{l}_k A(k, :)$  of the  $k + 1$ st to  $n$ th rows of  $A$ . The operation  $AL_k^{-1} = A + (A\mathbf{l}_k) \mathbf{e}_k^T$  changes only the  $k$ th column of  $A$ , incrementing it by the matrix-vector product  $A\mathbf{l}_k$ . An elementary transformation for eliminating rows of a matrix can be defined by  $\mathbf{r}_k^T = (0 \dots 0, r_{k+1}, \dots, r_n)$ ,  $R_k = I - \mathbf{e}_k \mathbf{r}_k^T$ ,  $R_k^{-1} = I + \mathbf{e}_k \mathbf{r}_k^T$ . The product  $AR_k$  is a rank-one update of the last  $n - k - 1$  columns of  $A$ ,  $A \leftarrow A - A(:, k) \mathbf{r}_k^T$ , adding multiples of the  $k$ th column to succeeding columns. The multiplication  $R_k^{-1}A = A + \mathbf{e}_k (\mathbf{r}_k^T A)$  increments the  $k$ th row of  $A$  by  $\mathbf{r}_k^T A$ .

<sup>2</sup>With a little MATLAB experimentation a reader can verify that Householder similarity transformations for reduction of a general matrix to Hessenberg form return a bandwidth of at least  $n/2$ .

Another useful nonorthogonal similarity transformation is balancing—scaling by diagonal similarity transformations so that the overall size of entries above and below the diagonal is roughly equal. It is good practice to run a balancing routine before determining eigenvalues. For example, the balancing routine DGEBAL in LAPACK is a helpful preliminary step before applying DGEHRD to reduce to Hessenberg form by Householder transformations. Balancing should also be used before BHESS.

The difficulties of tridiagonalizing a general matrix can be inferred from the following lemma [Bauer 1959; Howell and Geist 1995; Rutishauser 1953].

**LEMMA 2.1.** *Consider two similar tridiagonal matrices  $S$  and  $T$  with all sub and superdiagonal entries nonzero. If  $S$  and  $T$  are related by  $S = GTG^{-1}$  where  $G = \text{diag}(1, G_{n-1})$ , then  $G$  is a diagonal matrix.*

**COROLLARY 2.1.** *For the  $n \times n$  tridiagonal matrices  $S$  and  $T$  of Lemma 2.1,*

$$s_{i+1,i}s_{i,i+1} = t_{i+1,i}t_{i,i+1}, \quad i = 1, 2, \dots, n-1.$$

In words, unless similarity transformations influence the (1,1) entry, similar tridiagonal matrices are related by diagonal similarity transformations.

Reduction algorithms typically proceed row by row and column by column with similarity transformations of the form  $\hat{G} = \text{diag}(I_k, G_{n-k})$ . Suppose that  $k+1$  steps of a tridiagonalization algorithm have reduced a matrix to the form of (2.1) below, where  $F_k$  is tridiagonal. Breakdown at the  $k$ th step of a tridiagonalization algorithm is said to occur if a sub or superdiagonal entry is exactly zero. Consider elimination of a row and column by a similarity transformation of the form  $\text{diag}(I_k, G_{n-k})$ , as illustrated below.

$$\begin{aligned} & \left[ \begin{array}{c|c} I_k & 0 \\ \hline 0 & G_{n-k} \end{array} \right] \left[ \begin{array}{c|c} F_k & 0 \\ \hline 0 & u \quad B_{n-k} \end{array} \right] \left[ \begin{array}{c|c} I_k & 0 \\ \hline 0 & G_{n-k}^{-1} \end{array} \right] \\ & = \left[ \begin{array}{c|c} & 0 \\ \hline F_k & v^T G_{n-k}^{-1} \\ \hline 0 & G_{n-k} u \quad G_{n-k} B_{n-k} G_{n-k}^{-1} \end{array} \right]. \end{aligned} \quad (2.1)$$

Here  $u$  is a vector in the  $k+1$ st to  $n$ th entries of the  $k$ th column and  $v^T$  is a row vector in the  $k+1$ st to  $n$ th entries of the  $k$ th row. The inner product  $v^T u = v^T G_{n-k}^{-1} G_{n-k} u$  is unchanged. In the case that  $G_{n-k} u = \beta e_1$ ,  $\beta \neq 0$ , and  $v^T u = 0$ , then  $v^T G_{n-k}^{-1} = (0, z^T)$ , so that the resulting matrix is of the

6 • G. W. Howell and N. Diaa

form

$$\left[ \begin{array}{c|c} F_k & 0 \\ \hline 0 & (0 \ z^T) \\ \hline 0 & \begin{pmatrix} \beta \\ 0 \end{pmatrix} \\ \hline & G_{n-k} B_{n-k} G_{n-k}^{-1} \end{array} \right]. \quad (2.2)$$

For row elimination to occur, some nonzero entry of  $z^T$  would have to be permuted with the zero, swapping some column  $j > k + 1$  with the  $k + 1$ st column.<sup>3</sup> To preserve similarity, the  $j$ th row must also be swapped with the  $k + 1$ st row. Then any permutation of the row vector  $(0, z^T)$  to get a nonzero pivot element causes  $\beta e_1$  to be permuted, undoing the column elimination. According to Lemma 2.1 a tridiagonalization algorithm can avoid a zero inner product only by a similarity transformation not of the form  $\text{diag}(1, G_{n-1})$ —by a restart of the algorithm.

In rounding arithmetic, occurrence of a zero inner product is rare. More commonly, near-breakdown results in a poorly conditioned similarity transformation. Near-breakdown at the  $k$ th step arises when the product of the  $(k + 1, k)$  and  $(k, k + 1)$  entries is small. For the near-breakdown case, change the  $k$ th row of (2.2) to have  $(\dots, \epsilon, z^T)$  with  $\epsilon$  near zero and  $v^T u = \epsilon \beta$ :

$$\left[ \begin{array}{c|c} F_k & 0 \\ \hline 0 & \begin{pmatrix} \epsilon & z^T \end{pmatrix} \\ \hline 0 & \beta \\ \hline & G_{n-k} B_{n-k} G_{n-k}^{-1} \end{array} \right]. \quad (2.3)$$

Then the  $\epsilon$  generates large multipliers if  $z^T$  were to be eliminated without pivoting. When the inner product  $v^T u$  is small, BHESST therefore does not eliminate the row.

Preservation of similarity means that swapping rows entails a corresponding column swap. Since  $v^T u = \epsilon \beta$ , choosing the maximal element of  $u$  as the subdiagonal element  $\beta$  minimizes  $\epsilon$ . The vector  $z$  consists of entries of  $v$  that were not swapped to the  $k + 1$ st column. The multipliers will be  $z/\epsilon$ , maximized when  $\epsilon$  is minimized. A more reasonable way to choose a permutation is given below.

In summary, breakdown or near-breakdown can be avoided by

- Restarting a reduction with a similarity transformation that also affects the first row and column. This was the method used in Geist [1991].
- Allowing a greater bandwidth. BHESST allows the tridiagonal matrix to have additional entries above the first superdiagonal.

<sup>3</sup>From this point, it is assumed that the similarity transformations used to eliminate rows and columns are the elementary similarity transformations defined above. For an alternate approach using transformations of the form  $I - \alpha uv^T$  see Howell and Geist [1995].

### 3. AVOIDING BREAKDOWN BY INCREASING BANDWIDTH

#### 3.1 It is Useful to Eliminate Any Nonzero Row Above the Diagonal

Reconsider (2.2). As shown above,  $z^T$  cannot be eliminated. In (2.3),  $z^T$  can be eliminated, but only if large multipliers are allowed. In either case allowing a larger bandwidth avoids large multipliers. Repartitioning the matrix of (2.3) so that the upper left block is  $k + 1 \times k + 1$  results in

$$\left[ \begin{array}{ccc|c} F_k & \begin{pmatrix} 0 \\ \epsilon \end{pmatrix} & & 0 \\ (0 & \beta) & \times & z^T \\ \hline 0 & u & & v^T \\ \hline & & & B_{n-k-1} \end{array} \right]. \quad (3.1)$$

Applying a similarity transformation of the form  $\text{diag}(I_{k+1}, G_{n-k-1})$  results in

$$\left[ \begin{array}{ccc|c} F_k & \begin{pmatrix} 0 \\ \epsilon \end{pmatrix} & & 0 \\ (0 & \beta) & \times & z^T G_{n-k-1}^{-1} \\ \hline 0 & G_{n-k-1}u & & v^T G_{n-k-1}^{-1} \\ \hline & & & G_{n-k-1} B_{n-k-1} G_{n-k-1}^{-1} \end{array} \right] \quad (3.2)$$

with  $v^T u$ ,  $\epsilon$ ,  $\times$  and  $z^T u = z^T G_{n-k-1}^{-1} G_{n-k-1} u$  unchanged. Moreover, any row up to and including the  $k$ th with zeros in the last  $n - k + 1$  entries preserves those zeros. If either  $z^T u$  or  $v^T u$  is large enough, the last  $n - k - 2$  entries of  $z$  or  $v$  can be zeroed without using large multipliers. The zeroed elements remain zero during later row and column eliminations. Below is a tolerance test (3.5) that avoids excessively large multipliers and rounding errors by making a reasonably good choice of which row to eliminate.

#### 3.2 Choice of Row to Eliminate

For the reduction to small-band Hessenberg form, a general step analogous to (2.2) is given. After the first  $k + 1$  columns of a matrix have been eliminated, the current matrix will have the partitioning

$$\left[ \begin{array}{cc|c} & & 0 \\ & & \vdots \\ & & 0 \\ H_k & & v_{i_0}^T \\ & & \vdots \\ & & v_k^T \\ \hline 0 & u & B_{n-k} \end{array} \right] = \left[ \begin{array}{c|c} H_k & 0 \\ \hline 0 & u \\ \hline & B_{n-k} \end{array} \right], \quad (3.3)$$

8 • G. W. Howell and N. Diaa

where  $H_k$  is a Hessenberg matrix (compared to the strictly tridiagonal  $F_k$  of the matrices in the last section) and  $V^T$  is a block of rows,  $v_{i_0}^T \dots v_k^T$ , where some rows of  $V^T$  may have already been zeroed. For an example of what  $H_k$  might look like, see (4.1) below. The next column (or paired column-row) elimination then takes the form

$$\begin{aligned} & \left[ \begin{array}{c|c} I_k & 0 \\ \hline 0 & G_{n-k} \end{array} \right] \left[ \begin{array}{c|c} H_k & 0 \\ \hline 0 & u \quad B_{n-k} \end{array} \right] \left[ \begin{array}{c|c} I_k & 0 \\ \hline 0 & G_{n-k}^{-1} \end{array} \right] \\ & = \left[ \begin{array}{cc|cc} & & & 0 \\ & & & V^T G_{n-k}^{-1} \\ \hline & H_k & & \\ 0 & G_{n-k} u & G_{n-k} B_{n-k} G_{n-k}^{-1} & \end{array} \right]. \end{aligned} \quad (3.4)$$

Clearly, when the  $k$ th column is eliminated, any zero row of  $V^T$  will remain a zero row of  $V^T G_{n-k}^{-1}$ . Thus, if the  $k$ th column is to be eliminated, BHES allows elimination of any row  $i \leq k$  of  $V^T$  that has not been eliminated. If no row elimination can be performed in conjunction with the  $k$ th column, then the number of nonzero rows in  $V^T$  is increased by one. As the number of nonzero rows of  $V^T$  grows, so does the likelihood that some row can be eliminated in conjunction with a given column.

It has been shown that any eliminated row of  $V^T$  will remain zero in later steps. Remaining questions are

- How to select a row to eliminate.
- Given a row to be eliminated, how to select a paired column-row permutation.

The authors choose to constrain their selection methods to at most  $O(n)$  computations and comparisons per nonzero row of  $V^T$ . When  $V^T$  in (3.4) has only a few rows nonzero, such methods will not significantly increase the total computation.

As in Section 3.1, a similarity transformation of the form  $\text{diag}(I_k, G_{n-k})$  converts  $v^{(i)T}$  to  $\hat{v}^T = v^{(i)T} G_{n-k}^{-1}$  and  $u$  to  $\hat{u} = G_{n-k} u$ , so that the similarity transformation preserves the inner product  $v^{(i)T} u = \hat{v}^T \hat{u}$ . Since the inner product is preserved, it is natural to consider the angle between the row and column as a criterion for accepting a row for elimination.

Let  $\theta$  be the angle between  $u$  and  $v^{(i)}$ . A candidate criterion for allowing elimination of row  $v^{(i)}$  could be<sup>4</sup>

$$|\sec \theta| = \frac{\|u\| \|v^{(i)}\|}{|v^{(i)T} u|} \leq \text{tol}.$$

It turns out that for long vectors,  $|\sec \theta| \leq \text{tol}$  often requires unreasonably small multipliers. Consider for example the case that  $u$  is all ones and of odd length  $m$ , and that  $v^{(i)}$  has alternating ones and minus ones. Then  $|v^{(i)T} u| = 1$

<sup>4</sup>Unless otherwise indicated, vector norms  $\|\cdot\|$  are 2-norms.

and

$$|\sec \theta| = \frac{\|u\| \|v^{(i)}\|}{|v^{(i)T} u|} = m.$$

An elementary similarity transformation that eliminates all but the first entry of  $u$  does not change its first entry  $u_1 = 1$ . Denote the first entry of the transformed  $\hat{v}^T = v^{(i)T} G_{n-k}^{-1}$  by  $\alpha$ . Then  $1 = |v^{(i)T} u| = |\alpha u_1| = |\alpha|$ : the absolute value of the first entry of  $v^{(i)T}$  is unchanged. No other entry of  $v^{(i)T}$  is changed, so row multipliers are all of absolute value one. Both row and column multipliers are all ones in absolute value, but are rejected by any  $tol < m$ .

BHESS scales  $\sec \theta$  by the vector length. A column-row pair  $u$  and  $v$  is eligible for elimination if the criterion

$$\frac{|\sec \theta|}{m} = \frac{\|u\| \|v\|}{m |v^T u|} \leq tol \quad (3.5)$$

is satisfied, where  $m$  is the length of  $u$  and  $v$ . In the example of the preceding paragraph,  $u$  and  $v$  satisfy  $tol = 1$ . Suppose  $u$  and  $v$  are to be eliminated. Then  $v^T u = \alpha\beta$ , where  $\alpha$  and  $\beta$  are the leading entries of the eliminated column and row, respectively. The column multipliers are  $u/\alpha$  and the row multipliers are  $v/\beta$ . Recall that the root mean square of a vector  $b$  of length  $m$  is  $\|b\|/\sqrt{m}$ . Thus the computed quantity

$$\frac{\|u\| \|v\|}{m |v^T u|} = \frac{\|u/\alpha\| \|v/\beta\|}{m}$$

is the product of the root mean squares of the vectors of row and column multipliers. For example, row and column multipliers each of absolute value 10 would be allowable with  $tol = 100$ . When  $tol > 30$ , the upper bandwidth of the returned Hessenberg matrix is likely to be small. If  $tol$  is chosen smaller than about 0.5, rather few rows are likely to be eliminated, and the returned Hessenberg matrix is likely to be essentially full above the diagonal.

Root mean square is somewhat unconventional as a criterion for multiplier size, more usually bounded by the max norm. The authors offer the following justifications for departing from convention:

- $tol$  in condition (3.5) is proportional to the 2-norm conditioning and local backward error of a paired row-column elimination [Howell et al. 1997].
- The overall backward error estimate of Theorem 1.1 is the only one available for reduction to similar small-band form. It is phrased in terms of multiplier two norm, not in terms of max norm.
- Condition (3.5) is easy to compute and allows choice of row to eliminate to be made prior to choice of best row-column permutation.
- As seen in Section 5, (3.5) works well in practice.

### 3.3 Choice of an Appropriate Permutation

Once (3.5) has been used to choose which, if any, row is to be eliminated, an appropriate permutation is chosen. If the  $k$ th column elimination cannot be

10 • G. W. Howell and N. Diaa

paired with a row elimination satisfying (3.5), then no row will be eliminated. In this case, it makes sense to permute the largest entry of the eliminated column to the subdiagonal, so that all multipliers for eliminating the column are bounded by one in absolute value.

Otherwise, if a row-column elimination is paired, the following procedure (adapted from Geist [1991]) chooses the permutation to minimize the maximal multiplier for a paired row-column elimination.

Suppose that data are stored by columns. In order to maximize reuse of data fetched from main memory, row eliminations should be performed first (see section below on cache efficiency). Consider the paired elimination of the last  $n-k-2$  entries of the  $k$ th column of  $A$ , denoted  $u$ , and the last  $n-k-2$  entries of some row  $i \leq k$  of  $A$ , denoted  $v^T$ . If  $PAP^T$  permutes  $v_j$  to the  $k+1$ st column and  $u_j$  to the  $k+1$ st row, then after the row-eliminating similarity transformation  $R_{k+1}^{-1}AR_{k+1}$  (as defined in the beginning of Section 2), the first entry of  $v$  will still be  $v_j$ . Only the first entry of  $u$  will be changed, so denote the new first entry of  $u$  by  $\alpha$ . Then  $\alpha = v^T u / v_j$ . The multipliers that eliminate the row are

$$\frac{v_i}{v_j}, \quad k+1 \leq i \leq n, \quad i \neq j.$$

The multipliers that eliminate the column are

$$\frac{u_i}{\alpha}, \quad k+1 \leq i \leq n, \quad i \neq j.$$

Choosing the  $j$ th column to be swapped with the  $k+1$ st column gives the maximum of the row and column multipliers to be

$$M_j = \max \left( \frac{\max_{i \neq j} |v_i|}{|v_j|}, \frac{\max_{i \neq j} |u_i|}{|v^T u|} |v_j| \right).$$

Looping through  $j = k+1, \dots, n$ , choose  $p$  such that

$$M_p = \min_{j=k+1, \dots, n} M_j \tag{3.6}$$

and permute column and row  $p$  with column and row  $k+1$ .

#### 4. ALGORITHM DESCRIPTION

The pieces to construct a practical algorithm for reduction of a general matrix to small-band Hessenberg form are now in place.

Let  $A$  be an  $n \times n$  matrix. BHES uses elementary similarity transformations to eliminate the first  $n-2$  columns below the subdiagonal. A Hessenberg matrix is returned. Columns are eliminated in the sequence  $1, \dots, n-2$ . When the  $k$ th column is eliminated, any nonzero row  $i = 1, \dots, k$  can be eliminated without being filled in on successive steps. The algorithm requires the paired elimination of the  $i$ th row and  $k$ th column to satisfy the user-specified tolerance  $tol$  in (3.5), eliminating the first row  $i \leq k$  that satisfies (3.5).

#### 4.1 Algorithm BHES

```

For  $k = 1:n - 2$ 
   $u = A(k + 1:n, k)$ 
/* Determine whether some row is eligible to be eliminated. */
  Find the first row  $v^{(i)}$ ,  $i < k$  not yet eliminated, such that
     $\|v^{(i)T}\| \|u\| / |v^{(i)T}u| \leq (n - k - 1) tol.$ 
  If there is a first row  $i \leq k$  eligible to be eliminated then
/* Eliminate row-column pair  $u$  ( $k$ th column) and  $v^{(i)}$  ( $i$ th row). */
    1. Choose the pivot  $p \geq k + 1$  by (3.6) to minimize the
       maximal multiplier;
    2. Swap the  $k + 1$ st and  $p$ th rows and columns to put the
       pivot elements in the  $k + 1$ st row and column;
    3. Eliminate the  $i$ th row to the right of the  $k + 1$ st column;
    4. Eliminate the  $k$ th column below the subdiagonal;
  Else /* only eliminate column  $k$ . */
    1. Choose the maximal element of the  $k$ th column as pivot;
    2. Swap the largest element to the  $k + 1$ st row of the column
       and swap columns to preserve similarity;
    3. Eliminate the  $k$ th column below the subdiagonal;
  End if
End for

```

Since the  $k$ th step of the outer loop eliminates the  $k$ th column below the subdiagonal, a Hessenberg matrix is returned in all cases. If  $tol = 0$ , no rows are eliminated. A full Hessenberg matrix is returned and BHES is equivalent to ELMHES from EISPACK. If  $tol$  is taken so large that a row is eliminated in conjunction with each column, then BHES is Geist's A2TRI [Geist 1991]. As with A2TRI and ELMHES, multipliers are stored in the location they were used to eliminate, so that the original matrix is overwritten by the small-band matrix and multipliers.

The banded Hessenberg matrix returned by BHES may have a gradually widening profile. A typical profile is not so smooth, but has rows of nonzero entries protruding horizontally to the right from the diagonal. The following matrix has one possible form returned in the event that a row is eliminated on every step but the first column elimination. Elimination of row 2 is paired with elimination of column 2. Elimination of row 3 is paired with elimination of column 3. Elimination of row 1 is paired with elimination of column 4. Thereafter elimination of row  $k - 1$  is paired with elimination of column  $k$ , and successive rows have two nonzero entries to the right of the diagonal.

$$\begin{bmatrix}
 x & \epsilon & x & x & x & 0 & \dots & 0 \\
 x & x & x & 0 & 0 & 0 & \dots & 0 \\
 0 & x & x & x & 0 & 0 & 0 & \dots & 0 \\
 0 & 0 & x & x & x & x & 0 & \dots & 0 \\
 \vdots & & \ddots & \vdots & & & & & \vdots \\
 0 & \dots & & 0 & x & x & x & x & 0 & 0 \\
 0 & \dots & & & 0 & x & x & x & x & 0 \\
 0 & & \dots & & & 0 & x & x & x & x \\
 0 & & & \dots & & & 0 & x & x & x \\
 0 & & & & \dots & & & 0 & x & x
 \end{bmatrix} \tag{4.1}$$

## 5. NUMERICAL EXPERIMENTS

The BHES algorithm has been tested on a variety of matrices. Backward error is in practice small when  $tol$  is small and tends to increase smoothly with an increase in  $tol$ . As indicated by the backward error analysis of Theorem 1.1, any bound on multipliers tends to reduce rounding error.

For example, define the class of  $n \times n$  matrices  $AU(n)$  with entries uniformly distributed between  $-1$  and  $1$ . For thousands of trials for which  $tol$  from (3.5) satisfies  $tol < 5$ , and with matrix sizes  $200 \leq n \leq 1500$ , all eigenvalues returned by BHES-DHSEQR (BHES followed by LAPACK QR iteration) have agreed within  $10^{-6}$  with eigenvalues returned by LAPACK DGEHRD-DHSEQR (orthogonal reduction to Hessenberg form followed by QR iteration). For the  $AU(n)$  case with  $tol = 1$  in (3.5), the bandwidth of the returned matrix is of the order  $\sqrt{n}$ .

MATLAB experiments described in Howell et al. [1997] calculate overall conditioning of transformations and growth of element size. Explicit calculation of the overall conditioning was used to calibrate several estimators for backward error, including (1.3). For each of the matrices in Higham's test collection of matrices [Higham 1991], element growth was modest and overall conditioning of similarity transformations was small.

BHES was tested with the package INSTAB [Rowan 1990; Higham 1996, p. 488], which uses a steepest descent approach to automate the discovery of cases with large backward error. INSTAB is often successful in detecting instances of instability. For instance, INSTAB easily finds cases such that LINPACK and LAPACK estimators greatly underestimate matrix condition number and examples for which divide and conquer does not work well for determining spectra of unsymmetric matrices. INSTAB ran for several weeks on a SUN SPARC workstation, failing to find instances of large backward errors for BHES [Howell et al. 1997].

Results of some experiments with moderately sized matrices are given here. As a first test,

- (1) Lower triangular matrices were generated with real two-by-two diagonal blocks corresponding to complex eigenvalues. Complex eigenvalues were generated with real and imaginary parts uniformly distributed between minus one and one; real eigenvalues were uniformly distributed between minus one and one. 90 per cent of eigenvalues were taken as complex, 10 percent as real. The two-by-two and one-by-one blocks corresponding to the known eigenvalues were taken as  $D$ , and the rest of the lower triangular entries as  $L$ . The uniformly distributed entries of  $L$  (initially drawn from the interval  $[-1, 1]$ ) were scaled so that  $\|L\|_F/\|D\|_F = 0.2$ . This ratio is approximately the degree of nonnormality.
- (2) The matrix was converted to a full matrix by a sequence of random orthogonal transformations.
- (3) The known eigenvalues were then computed by LAPACK DGEHRD-DHSEQR, by BHES followed by BR with  $tol = 35$ , and by BHES followed by the LAPACK QR iteration DHSEQR.

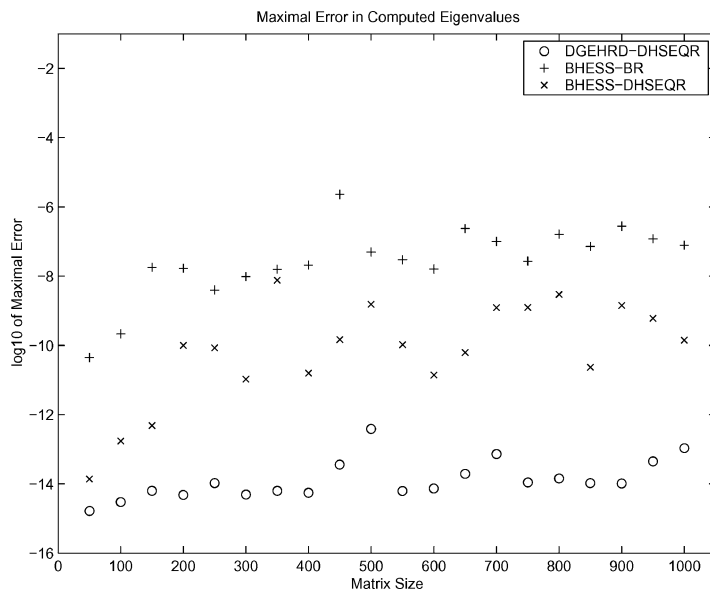


Fig. 2. Allowing moderately sized multipliers retains some accuracy for BHES-BR. Here  $tol = 35$ . The worst computed eigenvalues in these examples retain six significant digits.

Figure 2 plots maximal errors of the three resulting spectra for matrices of sizes from 50 to 1000. For the same computations, Figure 3 compares the execution times for BHES-BR and LAPACK DGEHRD-DGESQR running on a 266-MHz Pentium II. Both algorithms use Intel-supplied BLAS libraries.

A lower  $tol$  decreases the error for BHES-DHSEQR, but BR iteration is less able to constrain bandwidth, so that in some instances execution time and inaccuracy for BR increase. In test problems, large errors can be induced by setting multiple eigenvalues or by specifying clusters of eigenvalues.

For instance, five eigenvalues  $0.0, 0.0 \pm 0.0001, 0.0 \pm 0.00002$  were specified. The remainder of 200 eigenvalues were generated from a uniform distribution on the complex unit square. The differences (errors) between the near-zero eigenvalues and the computed eigenvalues from either LAPACK DGEHRD-DHSEQR or BHES-DHSEQR are of size  $10^{-3}$ . Errors for the other eigenvalues were  $O(10^{-14})$ . The low precision for near-repeated eigenvalues is consistent with the  $\epsilon^{1/k}$  perturbation in computed eigenvalues expected for a  $k$ -fold eigenvalue [Demmel 1987; Howell and Rekab 1995; Wilkinson 1965, 1984].<sup>5</sup>

The authors were able to generate poor stability behavior for BHES with small  $tol$  in two cases. One case was the two related matrices proposed by Businger to demonstrate the instability of the ELMHES reduction to Hessenberg form by elementary similarity transformations [Businger 1969]. This is BHES with  $tol = 0$ . In one of the Businger cases, element growth is

<sup>5</sup>QR iteration can be reliably applied to a small-band Hessenberg matrix. The first sweep fills in the matrix and computational savings are not realized, but comparing DGEHRD-DHSEQR, BHES-DHSEQR, and BHES-BR errors is helpful in assessing the relative stability of the algorithms.

14 • G. W. Howell and N. Diaa

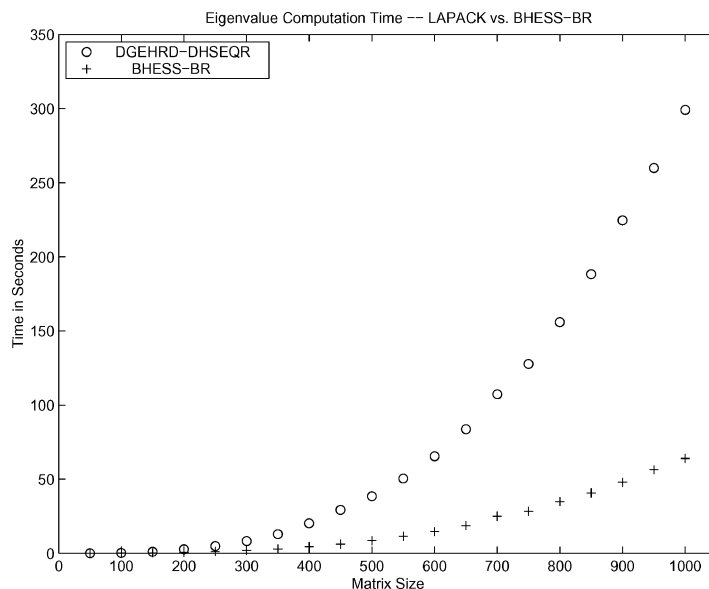


Fig. 3. For the same matrices as Figure 2, BHES-BR computes eigenvalues in about one-fifth the time required by LAPACK. BHES-BR and LAPACK are both using an optimized BLAS library.

excessive. In the other, conditioning of the overall similarity transformations exhibits exponential growth with matrix size.

The second case uses the fact that sequences of randomly generated Gaussian transformations are not well conditioned [Trefethen and Schreiber 1990]. The authors generated an  $n \times n$  matrix  $A$  by applying the BHES algorithm in reverse to a tridiagonal matrix  $T$ , using randomly generated but small multipliers. Though the entries of  $T$  are small, entries of  $A$  have magnitude proportional to  $\alpha^n$ , where  $n$  is the matrix size and  $\alpha > 1$ . Applying BHES to  $A$  to produce  $\hat{T}$  results in

$$\frac{\|\hat{T} - T\|}{\|T\|}$$

large, and

$$Z(A + E)Z^{-1} = \hat{T}$$

with  $\|E\|/\|A\|$  small and  $\|Z\| \|Z^{-1}\|$  large.

## 6. EFFICIENT IMPLEMENTATION OF BHES

For an  $n \times n$  matrix, BHES accomplishes reduction to a near-tridiagonal Hessenberg form in roughly  $8/3 n^3$  flops. Reduction to full Hessenberg form by Householder transformations (for example LAPACK DGEHRD) requires  $10/3 n^3$  flops. Reduction to full Hessenberg form by elementary similarity transformations, for example, EISPACK ELMHES, requires  $5/3 n^3$  flops. On computers with cache-based architectures, the volume of data transferred between levels of the memory hierarchy is often a better indication of speed of execution than is the number of flops. For matrices too large to fit into cache memory, but

small enough that several columns fit in cache memory, the LAPACK  $10/3 n^3$  flops algorithm DGEHRD reduces a general matrix to Hessenberg form by Householder transformations in much less time than the  $5/3 n^3$  flops EISPACK routine ELMHES.

Early versions of BHES performed a paired row-column elimination by four calls to BLAS-2 routines. Two of these were rank-one updates (DGER) performing the multiplication  $L_{k+1}A$  (adding the  $k + 1$ st row to successive rows) and  $AR_{k+1}$  (adding the  $k + 1$ st column to successive columns). Two are matrix-vector multiplies (DGEMV) performing the operations  $AL_{k+1}^{-1}$  (incrementing the  $k + 1$ st column of  $A$  by a linear combination of columns  $k + 2$  to  $n$ ) and  $R_{k+1}^{-1}A$  (incrementing the  $k + 1$ st row of  $A$  by a linear combination of rows  $k + 2$  to  $n$ ).

When the active part of the matrix is too large to fit in cache, these four BLAS-2 operations entail four reads of the active matrix from memory and two writes back to main memory. Typically these data transfers require more time than do the eight floating-point operations performed for each element of the active matrix. The speed of the resulting algorithm compares well to EISPACK, but not to LAPACK.

The current version of BHES replaces the four BLAS-2 calls by a sequence of BLAS-1 calls, performing a column by column sweep through the matrix. The same eight floating-point operations per element of the matrix take the form of three DAXPYs ( $\alpha x + y$ ) and one inner product (DDOT) per column. When several columns of a matrix fit in cache memory, there is only one read from main memory to cache and one write from cache to main memory for a paired row-column elimination. Before a paired elimination of the  $k$ th column and the  $i \leq k$  row, a pivot element is chosen and the  $k + 1$ st column stored as a temporary vector.

Specifically, when the  $j > k + 1$  column  $A(:, j)$  of  $A$  is in cache,

—Perform the rank-one update  $AR_{k+1}$ , incrementing the current column by a multiple of the  $k$ th column of  $A$ . The operation is the BLAS-1 DAXPY,

$$A(:, j) \leftarrow A(:, j) - r_j * A(:, k).$$

—The matrix-vector multiply  $R_{k+1}^{-1}A$  increments

$$A(k + 1, j) \leftarrow A(k + 1, j) + r^T A(k + 2:n, j).$$

The inner product is a BLAS-1 DDOT.

—Perform the rank-one update  $L_{k+1}A$  on  $A(:, j)$ . This is the BLAS-1 DAXPY,

$$A(k + 2:n, j) \leftarrow A(k + 2:n, j) - A(k + 1, j) * l$$

—Perform part of the matrix-vector multiply  $AL_{k+1}^{-1}$  by incrementing the temporary  $k + 1$ st column of  $A$  (stored in  $A(:, n + 1)$ ) by a multiple of  $A(:, j)$ . This is the DAXPY,

$$A(:, n + 1) \leftarrow A(:, n + 1) + l_j * A(:, j).$$

If tuned BLAS are not available, BHES reduction typically requires somewhat less time than the LAPACK DGEHRD orthogonal reduction to Hessenberg form. DGEHRD and BHES require much less time than the orthogonal ORTHES and nonorthogonal ELMHES algorithms from EISPACK.

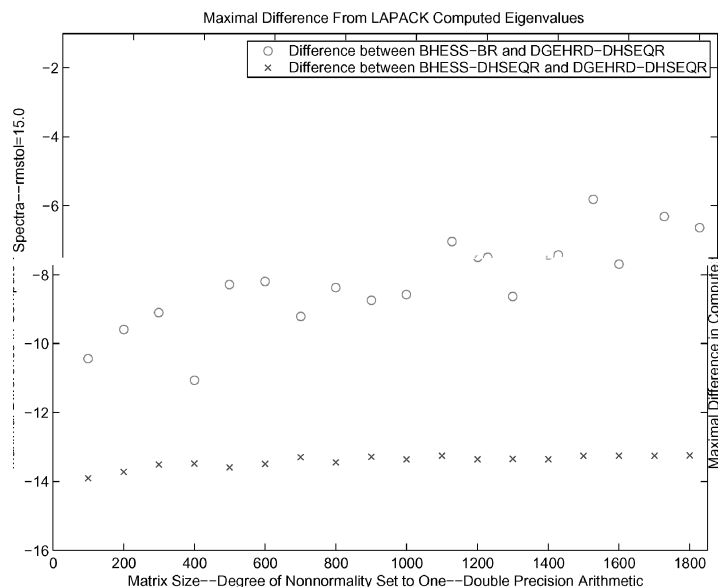


Fig. 4. The differences between the BHESS-BR and the LAPACK spectra show the error resulting from a quick determination of eigenvalues. For eigenvalues for which greater accuracy is desired, inexpensive refinement of the BR values is possible. The likely error of the refined eigenvalues is indicated by the difference between the BHESS-DHSEQR and DGEHRD-DHSEQR spectra.

If a tuned BLAS library is available, BHESS-BR computation of eigenvalues can be four or five times as fast as the LAPACK computation. This was the case in the results of Figure 3.

As another example, consider the following case. A symmetric tridiagonal matrix was randomly generated with entries from a uniform distribution between minus one and one. The lower triangular part of the matrix was filled with uniformly distributed random values in the range  $[-\sqrt{2}/(n-1), \sqrt{2}/(n-1)]$ . Thus the degree of nonnormality was about one. A random sequence of orthogonal transformations produced a dense matrix. Figures 4 and 5 show the numerical results. These runs were on an Alpha ev68 running at 1 GHz and linked to the dxml library. This CPU has a peak computational rate of 2 floating-point operations per clock cycle: 2 Gflops. For a  $1600 \times 1600$  matrix BHESS required 19.3 seconds (566 MFlops) and BR iteration 1.97 seconds for a total of 21.3 seconds to determine the spectrum. LAPACK DGEHRD Householder reduction took 20.3 seconds (672 Mflops), and LAPACK QR iteration DHSEQR required 65.7 seconds for a total of 86.0 seconds for spectral determination (DGEHRD is able to use about half BLAS-2 and half BLAS-3 operators).

BHESS reduces a general matrix  $A$  to a banded Hessenberg matrix  $H$  satisfying  $H = f(Z^{-1}AZ)$ . The matrix  $A$  passed to the subroutine BHESS is overwritten by  $H$  and the multipliers used to form  $Z$ . Some auxiliary routines are also provided. Given a complex vector  $x$ , BHAP1 returns  $x^T Z^{-1}$  and BHAP3 returns  $Zx$ . For a given real vector  $y$ , BHAP2 returns  $y^T Z$ . BHRPCK zeros out the multipliers, leaving the Hessenberg matrix  $H$  stored in an  $n \times n$  matrix.

## Algorithm 841: BHES: Reduction to a Similar-Banded Hessenberg Form • 17

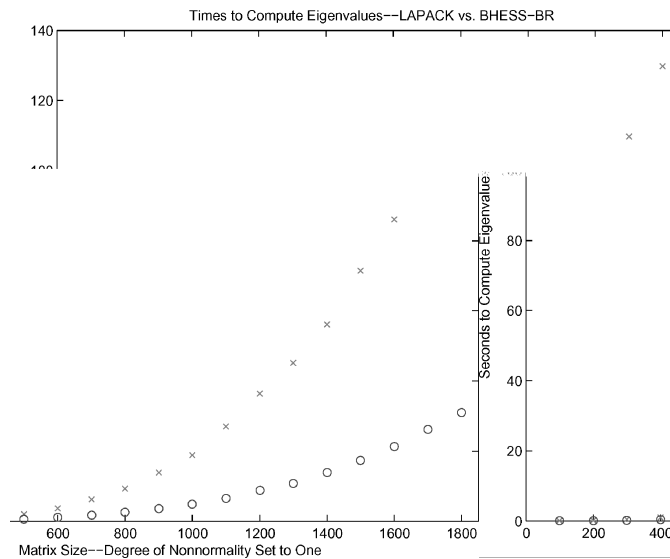


Fig. 5. For the same matrices as Figure 4, BHES-BR computes eigenvalues in about one-quarter the time required by LAPACK. BHES-BR and LAPACK are both using an optimized BLAS library.

BHUNPK gives  $H$  in diagonal storage, with the first column being the sub-diagonal, the second column the diagonal, etc. BHAPC is a LINPACK style estimator for  $\text{cond}_\infty(Z)$ .

A test package enables the user to experiment with various values of  $\text{tol}$  along with the resulting bandwidth of  $H$  as well as the computed Frobenius norm of  $\|A - Z^{-1}HZ\|$ . Some utilities are also provided for comparing results to a MATLAB/OCTAVE script. The MATLAB script returns  $Z$  explicitly, allowing convenient computations of condition number, and so on.

## 7. PRACTICAL USE OF BHES

BR iteration [Geist et al. 1999] was developed to determine eigenvalues of small-band Hessenberg matrices such as those returned by “look-ahead” Lanczos methods or BHES. Like QR or LR iteration, implicit double-shift BR iteration is a bulge-chasing algorithm. As in other implicit double-shift bulge-chasing algorithms, a bulge of length two is introduced in the first column of a Hessenberg matrix and chased down the matrix, with the double-shift enabling the use of real arithmetic to determine complex conjugate pairs of eigenvalues. When zero elements appear on the subdiagonal the matrix is deflated. Most typically, matrix deflation shortens the matrix by one or two. For a general theory of bulge-chasing algorithms for determining eigenvalues of Hessenberg matrices, see Watkins and Elsner [1991]. BR iteration uses simultaneous row and column eliminations to allow pivoting and bounded multipliers while at the same time constricting bandwidth.<sup>6</sup>

<sup>6</sup>LR iteration without pivoting preserves bandwidth, but often requires very large multipliers and is consequently less stable than BR iteration. LR with pivoting results in a rapid fill of the Hessenberg matrix.

Whereas QR iteration on a small-band Hessenberg matrix results in a full Hessenberg matrix, and thus requires  $O(n^2)$  flops per bulge chase, a BR bulge chase on a near-tridiagonal Hessenberg matrix usually requires  $O(n)$  flops. Thus BR can often find the eigenvalues of the Hessenberg matrix returned by BHES in  $O(n^2)$  flops [Geist et al. 1999]. In practice, BR iteration is effective only when BHES has been used with relatively large allowable multipliers. Typically, for BHES-BR, the authors set  $tol = 35$ . The overall cost of BHES-BR is typically  $8/3 n^3$  flops.<sup>7</sup> In comparison, the standard reduction to Hessenberg form followed by QR iteration requires  $10/3 n^3$  flops for the reduction and 10 to  $16 n^3$  flops for QR iteration on the resulting Hessenberg form, or around six times as many operations.

BHES-BR is appropriate for mildly nonnormal matrices when a number of nonextremal eigenvalues are desired. For example, matrices arising from PDE discretizations may be symmetric for homogeneous materials and mildly nonnormal for nonhomogenous material properties. Users with a cautious disposition should verify eigenvalues returned by BHES-BR by other means, for example, iterative refinement. Eigenvalues from BHES-BR can be refined by Rayleigh quotient iteration [Schrauf 1991] to the full precision of the BHES reduction.

If there are  $O(mn)$  nonzero entries in the matrix returned by BHES, Rayleigh quotient determination of a single eigenvalue requires  $O(mn)$  operations, so it is relatively inexpensive for  $m \ll n$ . Rayleigh quotient iteration returns left and right eigenvectors of the banded Hessenberg matrix  $H$ . The auxiliary routine BHAP1 forms  $v^T Z^{-1}$ , converting a left eigenvector of  $H$  to an

conv Tfaeans7tind/F1 1 Tf1.5 9.2(of)TJ-2(,)45TJ-33.686632 0 TD(H)Tj/F1773Tf1.1502 0 T66.2(a0 T66.4

mathematical and programming advice and encouragement, and M. Saunders and C. T. Fulton for careful reading and constructive suggestions.

## REFERENCES

- BAI, Z. 1994. Error analysis of the Lanczos algorithm for the nonsymmetric eigenvalue problem. *Math. Comp.* 62, 205, 209–226.
- BAI, Z., DAY, D., AND YE, Q. 1995. ABLE: An adaptive block Lanczos method for non-Hermitian eigenvalue problems. Research Report 95-04, Department of Mathematics, University of Kentucky.
- BAUER, F. L. 1959. Sequential reduction to tridiagonal form. *SIAM J.* 7, 107–113.
- BUSINGER, P. A. 1969. Reducing a matrix to Hessenberg form. *Math. Comp.* 23, 819–921.
- CHOI, C. H. AND LAUB, A. J. 1990. Efficient matrix-valued algorithms for solving stiff Riccati differential equations. *IEEE Trans. Autom. Contr.* 35, 770–776.
- DEMME, J. W. 1987. On condition numbers and the distance to the nearest ill-posed problem. *Numer. Math.* 51, 251–289.
- DONGARRA, J., GEIST, G., AND ROMINE, C. 1992. Algorithm 710: FORTRAN subroutines for computing the eigenvalues and eigenvectors of a general matrix by reduction to tridiagonal form. *ACM Trans. Math. Softw.* 18, 392–400.
- ELLNER, N. S. AND WACHSPRESS, E. L. 1991. Alternating direction implicit iteration for systems with complex spectra. *SIAM J. Num. Anal.* 28, 859–870.
- FREUND, R. W., GUTKNECHT, M. H., AND NACHTIGAL, N. M. 1993. An implementation of the “look-ahead” Lanczos algorithm for non-Hermitian matrices. *SIAM J. Sci. Computing* 14, 1, 137–158.
- GEIST, G. A. 1991. Reduction of a general matrix to tridiagonal form. *SIAM J. Matrix Anal. Appl.* 12, 2, 362–373.
- GEIST, G. A., HOWELL, G. W., AND WATKINS, D. S. 1999. The BR eigenvalue algorithm. *SIAM J. Matrix Anal. Appl.* 20, 4, 1083–1099.
- GEIST, G. A., LU, A., AND WACHSPRESS, E. L. 1989. Stabilized Gaussian reduction of an arbitrary matrix to tridiagonal form. Tech. Rep. ORNL/TM-11089, Mathematical Sciences Section, Oak Ridge National Laboratory.
- GOLUB, G. H., NASH, S., AND VAN LOAN, C. 1979. A Hessenberg-Schur method for the problem  $AX+XB=C$ . *IEEE Trans. Automa. Contr.* AC-24, 909–913.
- HIGHAM, N. J. 1991. Algorithm 694: A collection of test matrices in MATLAB. *ACM Trans. Math. Soft.* 17, 3 (Sept.), 289–305.
- HIGHAM, N. J. 1996. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia.
- HOWELL, G. W. 1994. Efficient computation of eigenvalues of randomly generated matrices. *J. Applied Math. and Comp.* 6, 9–24.
- HOWELL, G. W. AND GEIST, G. A. 1995. Reduction to a similar near-tridiagonal form. In *Proceedings of the ISCA 8th International Conference on Parallel and Distributed Computing Systems* (Raleigh, NC, 1995), pp. 426–432. ISCA.
- HOWELL, G. W., GEIST, G. A., AND ROWAN, T. 1997. Stability of reduction to a similar near-tridiagonal Hessenberg form. Tech. Rep. ORNL/TM-11097, Mathematical Sciences Section, Oak Ridge National Laboratory.
- HOWELL, G. W. AND REKAB, K. 1995. Expected conditioning for eigenvalues of randomly generated matrices. *Neural, Parallel & Scientific Computations* 3, 2, 263–270.
- PARLETT, B. N. 1992. Reduction to tridiagonal form and minimal realizations. *SIAM J. Matrix Anal. Appl.* 13, 2, 567–593.
- PARLETT, B. N. December, 1993. Do we understand the symmetric Lanczos algorithm yet? Corneliu Lanczos International Centenary Conference, North Carolina State University.
- ROWAN, T. H. 1990. *Functional Stability Analysis of Numerical Algorithms*. Ph.D. thesis, Department of Computer Sciences, University of Texas at Austin, Austin, TX.
- RUTISHAUSER, H. 1953. Beiträge zur Kenntnis des Biorthogonalisierungs-Algorithmus von Lanczos. *Zeitschrift für angewandte Mathematik und Physik* 4, 35–56.
- SCHRAUF, G. 1991. Algorithm 696: An inverse Rayleigh iteration for complex band matrices. *ACM Trans. Math. Softw.* 17, 3, 335–340.

20 • G. W. Howell and N. Diaa

- TAYLOR, D. R. 1982. *Analysis of the look ahead Lanczos algorithm*. Ph.D. thesis, University of California, Berkeley, Berkeley, CA.
- TREFETHEN, L. N. AND SCHREIBER, R. S. 1990. Average-case stability of Gaussian elimination. *SIAM J. Matrix Anal. Appl.* 11, 335–360.
- WACHSPRESS, E. L. 1988. Iterative solution of the Lyapunov matrix equation. *Appl. Math. Lett.* 1, 87–90.
- WATKINS, D. S. AND ELSNER, L. 1991. Convergence of algorithms of decomposition type for the eigenvalue problem. *Lin. Alg. and Applic.* 143, 19–47.
- WILKINSON, J. H. 1965. *The Algebraic Eigenvalue Problem*. Clarendon Press, Oxford, England.
- WILKINSON, J. H. 1984. On neighbouring matrices with quadratic elementary divisors. *Numer. Math.* 44, 1–21.
- YE, Q. 1994. A breakdown-free variation of the nonsymmetric Lanczos algorithms. *Math. Comp.* 62, 179–207.

Received December 1999; revised August 2000, July 2001, and December 2003; accepted October 2004