The SBR Toolbox – Software for Successive Band Reduction

Christian H. Bischof
Mathematics and Computer Science Division, Argonne National Laboratory, 9700 S. Cass Ave., Argonne, IL 60439. bischof@mcs.anl.gov
and
Bruno Lang
Fachbereich Mathematik, Bergische Universität GH Wuppertal, Gaußstr. 20, D-42097 Wuppertal, Germany. lang@math.uni-wuppertal.de
and
Xiaobai Sun
Department of Computer Science, Duke University, Durham, NC 27708-0129. xiaobai@cs.duke.edu

---

We present a software toolbox for symmetric band reduction, together with a set of testing and timing drivers. The toolbox contains routines for the reduction of full symmetric matrices to banded form and the reduction of banded matrices to narrower banded or tridiagonal form, with optional accumulation of the orthogonal transformations, as well as repacking routines for storage rearrangement. The functionality and the calling sequences of the routines are described, with a detailed discussion of the "control" parameters that allow adaptation of the codes to particular machine and matrix characteristics. We also briefly describe the testing and timing drivers included in the toolbox.

Categories and Subject Descriptors: G.1.3 [**Numerical Analysis**]: Numerical Linear Algebra—*Bandwidth reduction, orthogonal transformations, eigenvalues*; G.4 [**Mathematical Software**]

Additional Key Words and Phrases: symmetric matrices, tridiagonalization, blocked Householder transformations

---

1. INTRODUCTION

Reduction to tridiagonal form is a major step in eigenvalue computations for symmetric matrices. The LAPACK library [Anderson et al. 1995] includes the blocked Householder tridiagonalization algorithm for full matrices [Golub and Van Loan 1989; Dongarra et al. 1989] (routines _SYTRD for the reduction and _ORGTR for building the transformation matrix) and a variant of Schwarz's rotation-based algorithm

This work was supported by the Advanced Research Projects Agency, under contracts DM28E04120 and P-95006. Bischof also received support from 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. Lang also received support from the Deutsche Forschungsgemeinschaft, Geschäftszeichen Fr 755/6-1 and La 734/2-1. This work was partly performed while X. Sun was a postdoctoral associate with the Mathematics and Computer Science Division at Argonne National Laboratory.

Argonne Preprint ANL/MCS-P587-0496, submitted to ACM Trans. Math. Software

Figure 1. Different paths for reducing full and banded symmetric matrices to tridiagonal form.

[Schwarz 1968; Kaufman 1984] for banded matrices (routine _SBTRD).

In [Bischof et al. 1996] the authors gave evidence that one-step reduction is not necessarily the most efficient way for tridiagonalizing either full or banded matrices and proposed a framework of successive band reductions (*SBR*) to optimize reduction algorithms with respect to complexity, data locality, and/or memory requirements. As an example, consider reducing a full matrix to banded form and then tridiagonalizing the banded matrix. The first step can be done almost entirely by using Level 3 BLAS. Therefore, data locality is significantly improved in comparison with the direct tridiagonalization, where one half of the operations is confined to matrix-vector products. The second reduction step must be done with Level 2 BLAS, but it accounts for only a small percentage of the total work. Thus, the two-step approach can be superior on machines with a distinct memory hierarchy.

The SBR toolbox is intended to complement the LAPACK routines, as illustrated in Figure 1. All toolbox routines are available in single and double precision. For the sake of brevity, we describe only the doubleprecision routines in this paper. Their singleprecision twins are identical except for a leading "`S`" instead of "`D`" in the routine's name and `REAL` instead of `DOUBLE PRECISION` scalars and arrays in the parameter list.

At the user level, the toolbox provides four (doubleprecision) computational routines:

—`DSYRDB`: reduction of a symmetric full matrix to banded form,

—`DSYGTR`: accumulation of the transformations from `DSYRDB` in an orthogonal matrix,

—`DSBRDB`: reduction of a symmetric banded matrix to narrower banded form, and

—`DSBRDT`: reduction of a symmetric banded matrix to tridiagonal form with Householder transformations [Murata and Horikoshi 1975; Lang 1993],

and two routines for changing the data layout:

—`DSY2SB`: repacking of a symmetric banded matrix from conventional storage to the LAPACK lower banded storage scheme, and

—DSB2SB: repacking of a symmetric banded matrix from the LAPACK (upper or lower) banded storage scheme to lower banded storage with prescribed leading dimension.

The storage formats employed here are described in detail by Anderson et al. Anderson, Bai, Bischof, Demmel, Dongarra, Du Croz, Greenbaum, Hammarling, McKenney, Ostrouchov, and Sorensen [1995, p. 107 ff.].

The software can be retrieved from http://www.mcs.anl.gov/projects/PRISM.

The routine DSBRDT routine offers the same functionality as the LAPACK routine DSBTRD; it is included in the SBR toolbox because it features higher data locality than does the rotation-based approach and therefore can significantly outperform the latter if the bandwidth is not too small. In our experiments, the cross-over point was at $b \approx 10$ if the transformations were not accumulated in another matrix U. With accumulation, DSBRDT was competitive or superior for all b.

A whole family of multistep reduction algorithms for both full and banded matrices can be derived by following different paths in Figure 1, as described by Bischof et al. Bischof, Lang, and Sun [1996], for example,

—two-step tridiagonalization of full matrices: call DSYRDB to reduce the matrix to banded form, DSY2SB to repack the band matrix from conventional to lower banded storage, and DSBRDT to finally tridiagonalize it, or

—multistep tridiagonalization of banded matrices: call DSBRDB to reduce the bandwidth, then (optionally) repack the band with DSB2SB; these two steps may be repeated. Finally, DSBRDT (or DSBTRD, if the bandwidth is very small) is used to tridiagonalize the banded matrix.

The optimal reduction path depends on characteristics of the machine (such as the performance of the different BLAS levels), on the available workspace, and—in some special cases—on properties of the matrix (e.g., its eigenvalue distribution). Therefore, the SBR toolbox does not contain a "black box" driver algorithm that handles all cases. Rather it provides an infrastructure for experimenting with different reduction schemes that may be tailored to particular machines and applications.

The article is organized as follows. In Section 2 we describe the functionality and the calling sequences of the main routines. in Section 3 two of the parameters are discussed in some detail; they allow the user to optimize the performance of the routines. Section 4 briefly describes the installation of the SBR toolbox and summarizes the testing and timing drivers. In Section 5, we summarize the main points of the article.

2. CALLING SEQUENCES

In this section we describe the functionality and the calling sequences of the user-level doubleprecision routines.

The *semibandwith* of a symmetric matrix is the number of its outmost nonzero sub(super)diagonal. NB denotes a block size proposed by the user, whereas n_b is the block size that is eventually used in the computations ($n_b \leq$ NB).

2.1 Reduction from Full to Banded Form

```
SUBROUTINE DSYRDB( UPLO, JOB, N, B, A, LDA, DRPTOL,
                   U, LDU, NB, TAU, WORK, LWORK, INFO )
```

DSYRDB uses blocked Householder transformations to reduce an $\texttt{N} \times \texttt{N}$ symmetric full matrix A to a symmetric banded matrix with semibandwidth $\texttt{B} \geq 1$: $A \longrightarrow Q^T A Q = A_{\mathrm{band}}$.

On entry, the matrix A must be provided in the array A (leading dimension LDA) in conventional storage with either the upper (if UPLO = 'U') or lower (if UPLO = 'L') triangle explicitly stored. The other triangle of A is not accessed during the reduction.

On exit, the main diagonal and the first B super- (if UPLO = 'U') or subdiagonals (UPLO = 'L') of A are overwritten with the upper or lower triangle of the banded matrix A_{band}, again in conventional storage. The Householder vectors that were used in the reduction are returned in the zeroed-out portion of A and an additional vector TAU.

Optionally, if JOB = 'U', all the transformations can also be applied to another matrix U (stored in an array U with leading dimension LDU), that is, $U \leftarrow UQ$. If JOB = 'N', the array U is not referenced.

Note. The use of JOB = 'U' for accumulating the transformations is recommended only if U is not the idenity matrix when entering DSYRDB. To generate U "from scratch", DSYGTR should be called *after the reduction*, because the backward accumulation used in that routine takes significantly fewer flops and does not require storage for a second full matrix.

The parameter $\texttt{NB} \leq \texttt{B}$ may be used to control the level of blocking, that is, the number of Householder transformations that are aggregated into a blocked Householder transformation; see Section 3.

We can thus summarize the meaning of the parameters as follows:

UPLO (input) CHARACTER
Reference the upper (UPLO = 'U') or lower triangle (UPLO = 'L') of A?

JOB (input) CHARACTER
Update the matrix U (JOB = 'U') or not (JOB = 'N')?

N (input) INTEGER
Order of the matrix A.

B (input) INTEGER
Semibandwidth of the reduced matrix. $\texttt{B} \geq 1$.

A (input/output) DOUBLE PRECISION array, size (LDA, N)
On entry, the matrix A in conventional storage with either the upper or lower triangle explicitly stored. On exit, the upper or lower triangle of A is overwritten with the reduced matrix (main diagonal and B super- or subdiagonals) and the Householder vectors that were used in the reduction.

LDA (input) INTEGER
Leading dimension of the array `A`.

DRPTOL (input) DOUBLE PRECISION
For the use of this parameter, see Section 3.3.

U (input/output) DOUBLE PRECISION array, size `(LDU, N)`
Matrix for accumulating the transformations. Not accessed if `JOB` = `'N'`.

LDU (input) INTEGER
Leading dimension of the array `U`.

NB (input) INTEGER
Suggested order for the blocked Householder transformations. For the use of this parameter, see Section 3.1.

TAU (output) DOUBLE PRECISION array, size `(N)`
The "scaling factors" of the Householder transformations.

WORK (workspace) DOUBLE PRECISION array, size `(LWORK)`

LWORK (input) INTEGER
Length of the workspace array. The minimum value is $\texttt{LWORK} = 3\texttt{N} - 2\texttt{B}$, but the minimum workspace prevents the use of blocked transformations and therefore degrades performance. For using n_b-blocked Householder transformations, we need $\texttt{LWORK} \geq n_b(3\texttt{N} - 2\texttt{B})$. As a rule, $\texttt{LWORK} \geq 50\texttt{N}$ should be fine.

INFO (output) INTEGER
On exit, a negative value $\texttt{INFO} = -i$ indicates that the routine stopped processing because of an error in the ith parameter, whereas a positive value $\texttt{INFO} = n_b$ indicates successful reduction with block transformations of order n_b.

2.2 Generating the Transformation Matrix

```
      SUBROUTINE DSYGTR( UPLO, N, B, A, LDA, TAU,
                         WORK, LWORK, INFO )
```

`DSYGTR` overwrites `A` with the orthogonal matrix Q that reduces A to banded form, i.e., $A \leftarrow Q$, where $Q^TAQ = A_{\text{band}}$.

The parameters `UPLO`, `N`, `B`, `A`, `LDA`, and `TAU` must be the same as in the preceding `DSYRDB` call.

Notes. (1) The banded matrix must be copied into *another* array with `DSY2SB` before calling `DSYGTR`. (2) Since the orthogonal matrix Q is not symmetric, both triangles of `A` are needed to store it. (3) `DSYGTR` does not have a parameter `NB` for controlling the block size. This routine calls a suitable LAPACK routine, which automatically sets the block size.

The meaning of the parameters is as follows:

A (input/output) DOUBLE PRECISION array, size `(LDA, N)`
On entry, `A` contains the Householder vectors that were used in the reduction, as returned by `DSYRDB`. On exit, `A` is overwritten with Q.

WORK (workspace) DOUBLE PRECISION array, size `(LWORK)`

LWORK (input) INTEGER
Length of the workspace array. The minimum value is $\texttt{LWORK} = \texttt{N} - \texttt{B}$, but workspace this small prevents the use of blocked transformations and therefore degrades performance. For using n_b-blocked Householder transformations, we need $\texttt{LWORK} \geq n_b(\texttt{N} - \texttt{B})$. As a rule, $\texttt{LWORK} \geq 20\texttt{N}$ should be fine.

INFO (output) INTEGER
On exit, a negative value $\texttt{INFO} = -i$ indicates that the routine stopped processing because of an error in the ith parameter, whereas $\texttt{INFO} = \texttt{1}$ indicates successful completion.

2.3 Reduction from Banded to Narrower Banded Form

```
      SUBROUTINE DSBRDB( JOB, N, B1, B2, A, LDA, DRPTOL,
                         U, LDU, NB, WORK, LWORK, INFO )
```

`DSBRDB` uses blocked Householder transformations to reduce a banded symmetric matrix A_1 with semibandwidth `B1` to narrower banded form: $A_1 \longrightarrow Q^T A_1 Q = A_2$, where A_2 has semibandwidth $\texttt{B2} < \texttt{B1}$.

On entry, the array `A` (leading dimension $\texttt{LDA} \geq 1 + (\texttt{B1} - \texttt{B2}) + \texttt{B1}$) must contain the matrix A_1 in the LAPACK *lower* symmetric banded storage scheme [Anderson et al. 1995, p. 109]. That is, the columns (diagonals) of A_1's lower triangle are stored in the first columns (rows) of `A`. On exit, `A` contains the matrix A_2 in lower banded storage.

Optionally ($\texttt{JOB} = \texttt{'U'}$), all the transformations can also be applied to another matrix U (stored in an array `U` with leading dimension `LDU`), that is, $U \leftarrow UQ$. If $\texttt{JOB} = \texttt{'N'}$, the array `U` is not referenced.

The parameter $\texttt{NB} \leq \texttt{B2}$ may be used to control the level of blocking, that is, the number of Householder transformations that are aggregated into a blocked Householder transformation; see Section 3.1.

Notes. (1) Calls to `DSY2SB` or `DSB2SB` may precede this routine in order to repack the matrix from (upper or lower) conventional or upper banded storage to the lower banded storage scheme. (2) `DSBRDB` *should not be used for tridiagonalizing* banded matrices, since `DSBRDT` provides optimized code for this task.

The meaning of the parameters is as follows:

JOB (input) CHARACTER
Update the matrix U ($\texttt{JOB} = \texttt{'U'}$) or not ($\texttt{JOB} = \texttt{'N'}$)?

N (input) INTEGER
Order of the banded matrix A_1.

`B1` (input) INTEGER
Semibandwidth of the matrix A_1 before the reduction.

`B2` (input) INTEGER
Semibandwidth of the reduced matrix A_2.

`A` (input/output) DOUBLE PRECISION array, size `(LDA, N)`
On entry, the main diagonal and the `B1` nonzero subdiagonals of the matrix A_1 are stored in the first `B1`$+1$ rows of `A` (LAPACK lower banded storage). On exit, the first `B2` $+ 1$ rows of `A` contain the main diagonal and the `B2` nonzero subdiagonals of the reduced matrix A_2.

`LDA` (input) INTEGER
Leading dimension of the array `A`. `LDA` $\geq 1 +$ (`B1` $-$ `B2`) $+$ `B1`; `LDA` this small requires additional data movement when n_b-blocked Householder transformations are used in the reduction. To avoid this overhead, provide `LDA` $\geq$ `NB` $+$ (`B1` $-$ `B2`) $+$ `B1`.

`DRPTOL` (input) DOUBLE PRECISION
For the use of this parameter, see Section 3.3.

`U` (input/output) DOUBLE PRECISION array, size `(LDU, N)`
Matrix for accumulating the transformations. Not accessed if `JOB` = `'N'`.

`LDU` (input) INTEGER
Leading dimension of the array `U`.

`NB` (input) INTEGER
Suggested order for the blocked Householder transformations. For the use of this parameter, see Section 3.1.

`WORK` (workspace) DOUBLE PRECISION array, size `(LWORK)`

`LWORK` (input) INTEGER
Length of the workspace array. The minimum value is `LWORK` $\geq$ `N`$+2($`B1`$-$`B2`$)+3$ if U is required, and `LWORK` $\geq 3($`B1`$-$`B2`$)+4$ otherwise. Workspace this small prevents the use of blocked transformations and therefore degrades performance. For using n_b-blocked Householder transformations, we need `LWORK` $\geq 2.5n_b^2 + ($`N` $+ 2($`B1` $-$ `B2`$) + 0.5)n_b$ if U is required and `LWORK` $\geq 3.5n_b^2 + (3($`B1` $-$ `B2`$) + 0.5)n_b$ otherwise. As a rule, `LWORK` ≥ 10`N` should be fine.

`INFO` (output) INTEGER
On exit, a negative value `INFO` $= -i$ indicates that the routine stopped processing because of an error in the ith parameter, whereas a positive value `INFO` $= n_b$ indicates successful reduction with blocked transformations of order n_b.

2.4 Reduction from Banded to Tridiagonal Form

```
      SUBROUTINE DSBRDT( JOB, N, B, A, LDA, DRPTOL,
                         D, E, U, LDU, NB, WORK, LWORK, INFO )
```

DSBRDT uses Householder transformations to reduce a banded symmetric matrix A with semibandwidth B to tridiagonal form: $A \longrightarrow Q^TAQ = T$.

On entry, the array A (leading dimension LDA $\geq$ 2B) must contain the matrix A in the LAPACK *lower* banded storage scheme, that is, the columns (diagonals) of A's lower triangle are stored in the first columns (rows) of A. On exit, D and E contain the main diagonal and the subdiagonal of T, respectively.

Optionally (JOB = 'U'), all the transformations can also be applied to another matrix U (stored in an array U with leading dimension LDU), that is, $U \leftarrow UQ$. If JOB = 'N', the array U is not referenced. U is updated with blocked Householder transformations. The parameter NB may be used to control the level of blocking, that is, the number of Householder transformations that are aggregated into a blocked Householder transformation; see Section 3.1.

Note. For very small bandwidths, and if U is not required, the LAPACK routine DSBTRD should be used since it performs better in these cases.

The meaning of the parameters is as follows:

JOB (input) CHARACTER
Update the matrix U (JOB = 'U') or not (JOB = 'N')?

N (input) INTEGER
Order of the banded matrix A_1.

B (input) INTEGER
Semibandwidth of the matrix A before the reduction.

A (input/output) DOUBLE PRECISION array, size (LDA, N)
On entry, the main diagonal and the B nonzero subdiagonals of the matrix A are stored in the first B+1 rows of A (LAPACK lower banded storage). On exit, A is destroyed.

LDA (input) INTEGER
Leading dimension of the array A. LDA $\geq$ 2B.

DRPTOL (input) DOUBLE PRECISION
For the use of this parameter, see Section 3.3.

D (output) DOUBLE PRECISION array, size (N)
Main diagonal of the tridiagonal matrix T.

E (output) DOUBLE PRECISION array, size (N−1)
Subdiagonal of the tridiagonal matrix T.

U (input/output) DOUBLE PRECISION array, size (LDU, N)
Matrix for accumulating the transformations. Not accessed if JOB = 'N'.

LDU (input) INTEGER
Leading dimension of the array U.

NB (input) INTEGER
Suggested order for the blocked Householder transformations in the update of U. For the use of this parameter, see Section 3.1.

WORK (workspace) DOUBLE PRECISION array, size (LWORK)

LWORK (input) INTEGER
Length of the workspace array. LWORK $\geq$ 2B, if U is not required. If U is required, the minimum value is LWORK = B + N, but workspace this small prevents the use of blocked transformations in the update of U and therefore degrades performance. For using n_b-blocked Householder transformations, we need LWORK $\geq 2n_b($N$+$B$+n_b-1)$. As a rule, LWORK $\geq$ 15N should be fine.

INFO (output) INTEGER
On exit, a negative value INFO $= -i$ indicates that the routine stopped processing because of an error in the ith parameter, whereas a positive value INFO $= n_b$ indicates successful reduction and update with blocked trasnformations of order n_b.

2.5 Repacking from Conventional to Banded Storage

```
SUBROUTINE DSY2SB( UPLO, N, B, AFULL, LDFULL,
                   ABAND, LDBAND, INFO )
```

DSY2SB copies a symmetric banded matrix from conventional storage (with either the upper or lower triangle explicitly stored) to LAPACK lower banded storage.

Note. The matrix may be repacked in place by setting ABAND equal to AFULL.

The meaning of the parameters is as follows:

UPLO (input) CHARACTER
A is provided in upper (UPLO = 'U') or lower (UPLO = 'L') full storage.

N (input) INTEGER
Order of the matrix A.

B (input) INTEGER
Semibandwidth of the matrix A.

AFULL (input) DOUBLE PRECISION array, size (LDFULL, N)
The main diagonal and the next B super(sub)diagonals contain the banded matrix A in full storage.

LDFULL (input) INTEGER
Leading dimension of the array AFULL.

ABAND (output) DOUBLE PRECISION array, size (LDBAND, N)
The first B + 1 rows of the array contain the matrix A in lower banded storage.

LDBAND (input) INTEGER
Leading dimension of the array ABAND.

INFO (output) INTEGER
On exit, a negative value INFO $= -i$ indicates that the routine stopped processing because of an error in the ith parameter, whereas INFO = 1 indicates successful completion.

2.6 Repacking from Banded to Banded Storage

```
      SUBROUTINE DSB2SB( UPLO, N, B, ASRC, LDSRC,
                         ADST, LDDST, INFO )
```

DSB2SB copies a symmetric banded matrix from LAPACK (upper or lower) banded storage to lower banded storage.

Note. The matrix may be repacked in place by setting ADST equal to ASRC.

The meaning of the parameters is as follows:

UPLO (input) CHARACTER
A is provided in upper (UPLO = 'U') or lower (UPLO = 'L') banded storage.

N (input) INTEGER
Order of the matrix A.

B (input) INTEGER
Semibandwidth of the matrix A.

ASRC (input) DOUBLE PRECISION array, size (LDSRC, N)
The matrix A in (upper or lower) banded storage.

LDSRC (input) INTEGER
Leading dimension of the array ASRC.

ADST (output) DOUBLE PRECISION array, size (LDDST, N)
The matrix A in lower banded storage.

LDDST (input) INTEGER
Leading dimension of the array ADST.

INFO (output) INTEGER
On exit, a negative value INFO $= -i$ indicates that the routine stopped processing because of an error in the ith parameter, whereas INFO = 1 indicates successful completion.

3. OPTIMIZING THE PERFORMANCE

In this section we discuss several ways to improve the performance of the reduction algorithms:

—selecting a suitable blocking level for the blocked Householder transformations to increase the performance of the Level 3 BLAS kernels,

—repacking the band to improve the data locality, and

—skipping some Householder transformations to reduce the flops count when possible and allowed.

These techniques minimize the execution time of each single reduction step. A suitably chosen reduction sequence can further improve the performance of the overall reduction process; see [Bischof et al. 1996].

3.1 Blocked Householder Transformations

The computational routines are designed to use blocked Householder transformations in the reduction of A (DSYRDB and DSBRDB) and in the update of the matrix U (DSYRDB, DSBRDB, and DSBRDT). Since n_b (the number of Householder transformations that are aggregated into a blocked transform) can have significant impact on the performance, the parameter NB allows the user to control the level of blocking.

Note that the routines automatically reduce the block size n_b if the workspace is not sufficient (see the parameters LWORK in Section 2). On exit, INFO returns the value n_b that was used in the reduction and/or update.

3.1.1 *Using the Default Values.* If it is not important to squeeze the highest performance out of the routines, or if the SBR toolbox was optimized upon installation, use NB = 0.

Then the auxiliary routine NBDFLT is called to obtain a default block size n_b. In a standard installation of the toolbox the default values are

$$n_b = \begin{cases} 16, & \text{for } \texttt{DSYRDB} \\ 6, & \text{for } \texttt{DSBRDB} \\ 6, & \text{for } \texttt{DSBRDT}, \text{ if } \texttt{B} \geq 6 \\ 1, & \text{for } \texttt{DSBRDT}, \text{ if } \texttt{B} < 6 \end{cases}$$

These values may be changed when installing the toolbox; see Section 4.2.

3.1.2 *Providing a Block Size in the Procedure Call.* In some cases, the default n_b values may not be appropriate. Since blocking significantly increases the flops count when the bandwidth is very small, nonblocked transformations may be faster in this case. On the other hand, for larger bandwidths it may be better to increase the block size, since the Level 3 BLAS usually deliver better performance. Figure 2 illustrates these issues.

The user may provide a "preferred" block size by calling the routine with NB > 0. In general, the routine will then use NB-blocked Householder transformations in the reduction of A and/or the update of U. It may, however, be forced to reduce the block size to a value $n_b <$ NB for two reasons:

—Algorithmic restrictions: $n_b \leq$ B in DSYRDB and $n_b \leq$ B2 in DSBRDB (see the right picture of Figure 2; in the reduction from bandwidth $b_1 = 16$ to $b_2 = 8 = d$, all NB > 8 were reduced to $n_b = 8$). The tridiagonalization routine DSBRDT cannot block the transformations in the reduction, but there is no restriction on n_b in the update of U.

—Workspace restrictions: LWORK is too small to allow NB-blocked Householder transformations; see the discussion of LWORK in Section 2.

The routine will never increase the block size except when NB = 0. In particular, NB = 1 will force nonblocked Householder transformations. The block size n_b used

Figure 2. Speedup of tridiagonalization (left picture, including the update of U) and bandwidth reduction from $b_1 = 2d$ to $b_2 = d$ (right picture, no update) with different block sizes NB as compared with using the default block size $n_b = 6$. The timings were made on a single node of the IBM SP.

Table I. Timings (in seconds) for the two-step tridiagonalization of banded matrices of order $n = 1200$ on one node of the IBM SP.

	B = 64, LDA = 128	B = 128, LDA = 256
No repacking	14.76	23.67
Repacking to LDA = 64 before DSBRDT	14.31	21.32
Repacking before DSBRDB and before DSBRDT	13.57	20.03

in the reduction/update is returned in the status indicator INFO.

3.2 Repacking the Band

The data locality in the bandwidth reduction algorithms can be improved by providing the band as tightly packed as possible (that is, LDA = 2B for _SBRDT and LDA = NB + (B1 − B2) + B1 for _SBRDB).

This repacking does not require much time, and it may significantly speed up the reduction, see Table I. Here, the matrix was provided in a slightly oversized array (LDA = 2B); the intermediate bandwidth was B2 = 32, and NB = 0 (default blocking). Note that the timings include the repacking, but not the update of U.

3.3 Skipping Transformations

The reduction routines DSYRDB, DSBRDB, and DSBRDT have a parameter DRPTOL that may be used to reduce the number of arithmetic operations by skipping selected Householder transformations.

More precisely, let $x = (x_1, x_2, \ldots, x_k)^T$ be some vector in the matrix A that must be reduced to a multiple $\tilde{x} = \xi e_1$ of the first unit vector. If $||(x_2, \ldots, x_k)^T||_2 >$ DRPTOL, we determine a Householder transformation H to zero out these elements and apply the transformation to the matrix A (and, optionally, to U). If $||(x_2, \ldots, x_k)^T||_2 \leq$ DRPTOL, we simply set $\tilde{x} = (x_1, 0, \ldots, 0)^T$, without computing and applying any Householder transformation.

The amount of savings from skipping transformations depends very much on the distribution of A's spectrum. When the eigenvalues of A are contained in a few rather narrow clusters considerable speedup can be obtained [Bischof and Sun 1995]. Such matrices arise, e.g., in the invariant subspace decomposition approach

Table II. Timings (in seconds) for tridiagonalizing ISDA-type matrices with $\omega = 1000 \cdot \varepsilon$ ($\varepsilon \approx 1.11 \cdot 10^{-16}$ is the machine precision) on one node of the IBM SP.

	$n = 400$		$n = 1200$	
	$b = 8$	$b = 64$	$b = 8$	$b = 64$
LAPACK routine `DSBTRD`				
without update	0.36	4.58	5.30	35.81
with update	5.46	9.50	134.52	183.21
`DSBRDT` with `DRPTOL` $= 0$				
without update	0.93	1.45	8.54	15.45
with update	5.34	3.93	137.81	84.95
`DSBRDT` with `DRPTOL` $= \sqrt{n} \cdot \varepsilon$				
without update	0.90	0.98	8.37	10.36
with update	5.22	2.83	134.95	58.41
`DSBRDT` with `DRPTOL` $= n \cdot \varepsilon$				
without update	0.77	0.55	7.12	8.17
with update	4.48	1.75	116.14	46.81

Table III. Residual $||U^TAU - T||$ and orthogonality $||U^TU - I||$ (in multiples of the machine precision $\varepsilon \approx 1.11 \cdot 10^{-16}$) for tridiagonalizing an ISDA-type matrix with $n = 400$, $b = 8$, and $\omega = 1000 \cdot \varepsilon$ on one node of the IBM SP.

	$\|\|U^TAU - T\|\|/\varepsilon$	$\|\|U^TU - I\|\|/\varepsilon$
LAPACK routine `DSBTRD`	344	291
`DSBRDT` with `DRPTOL` $= 0$	345	349
`DSBRDT` with `DRPTOL` $= \sqrt{n} \cdot \varepsilon$	381	413
`DSBRDT` with `DRPTOL` $= n \cdot \varepsilon$	4590	424

(ISDA) for eigensystem computations [Lederman et al. 1991]. In this method, the spectrum of a matrix is condensed into two narrow clusters by repeatedly applying a function f to the matrix.

Table II shows the timings for tridiagonalizing ISDA-type matrices; half of the eigenvalues are randomly chosen from $[0, \omega]$; the other eigenvalues are in $[1 - \omega, 1]$, where ω is the width of the clusters.

As may be observed from Table III, a large skipping threshold can significantly increase the reduction residual $||U^TAU - T||$, whereas the orthogonality $||U^TU - I||$ of the transformation matrix is not affected. We propose to choose `DRPTOL` smaller than p, where p is the uncertainty already contained in the original matrix or the perturbation introduced in the "standard" reduction (without skipping).

For matrices with a more uniform eigenvalue distribution, the potential for skipping transformations is very limited. Therefore, if no a priori information of the spectrum is available or if this feature should not be used, set `DRPTOL` $= 0$.

4. INSTALLATION

The SBR toolbox comes with a `UNIX` makefile for easy installation and a set of testing and timing drivers for validation and performance tuning.

In this section we briefly describe the "standard" installation procedure. More information (e.g., how to run only a subset of the testing and timing drivers) may be found in the `README` file distributed with the software.

4.1 Installing the SBR Toolbox

The installation consists of the following steps.

(1) Get the SBR toolbox from the software directory at `http://www.mcs.anl.gov`, and unpack it with

```
zcat sbr.tar.Z | tar xf -
```

This command puts all the SBR software into a new directory `sbr`.

(2) Edit the first few lines of the `makefile` to match your system setup (e.g., the location of the LAPACK library, if the latter is installed on your machine).

(3) Type

```
make library
```

This command will build the library (called `libSBR.a` if you did not change the name).

(4) (Optional.) Fine-tune the performance of the algorithms; see Section 4.2.

(5) (Recommended.) Run the validation tests; see Section 4.3.

(6) (Optional.) Run the timings; see Section 4.4.

(7) (Optional.) Move the SBR library to a directory searched by the linker.

4.2 Performance Tuning

The performance of the computational routines can be optimized at installation time by modifying the file `nbdflt.f`, which sets the default block sizes (see Section 3.1.1).

For the reduction of banded matrices, the optimum block size is determined mainly by the number d of diagonals to be removed, by the precision (the BLAS may feature different optimizations for single or double precision), and by the value of `JOB` (is U updated or not?). To a lesser extent, the matrix size may also influence the optimum block size. The "template" `NBDFLT` routine included in the SBR toolbox does not make use of these parameters. This routine may be tailored to a particular machine in the following way.

The command

```
make tuning
```

produces timings for a fairly comprehensive set of combinations of the above parameters. By analyzing the output file corresponding to each reduction routine, a simple function $f(n, d, \text{precision}, \texttt{JOB})$ is derived that gives an (almost) optimum block size, and `nbdflt.f` is changed accordingly. Some advice on how to construct f is given in the `README` file.

4.3 Running the Testing Drivers

The testing drivers provide the residuals $||U^TAU - \tilde{A}||$, the orthogonality errors $||U^TU - I||$, and the timings for the reduction of a symmetric (full or banded) matrix A to a (narrower) banded or tridiagonal matrix $\tilde{A}$ for matrices of order $n \leq 600$ and different semibandwidths. The command

```
make checks
```

will run the testing drivers for the following (successive) reduction algorithms (the two-step and multistep reductions are discussed by Bischof et al. Bischof, Lang, and Sun [1996]):

—one-step tridiagonalization of symmetric full matrices: `_SYTRD` and `_ORGTR`;

—two-step tridiagonalization of symmetric full matrices: `_SYRDB` (with `JOB` = `'U'` to accumulate the transformations), `_SY2SB`, and `_SBRDT`;
—two-step tridiagonalization of symmetric full matrices: `_SYRDB` (with `JOB` = `'N'`), `_SYGTR` (to accumulate the transformations), `_SY2SB`, and `_SBRDT`;
—one-step tridiagonalization of symmetric banded matrices: `_SBTRD`;
—one-step tridiagonalization of symmetric banded matrices: `_SBRDT`;
—two-step tridiagonalization of symmetric banded matrices: `_SBRDB` and `_SBRDT`;
—multistep tridiagonalization of symmetric banded matrices with the doubling-stride sequence: multiple calls to `_SBRDB` and one call to `_SBRDT`.

This command produces 14 output files. For example, `DSbrdbSbrdt.chk` contains the data for the two-step tridiagonalization of symmetric banded matrices in double precision. Note that the residuals and the orthogonality errors are not available if the matrix U is not accumulated.

4.4 Running the Timing Drivers

The timing drivers run the reduction algorithms listed in Section 4.3 on larger matrices ($n \leq 1200$) and hence take significantly longer to complete. Therefore,

```
make timings
```

should be used only if the timings provided by the testing drivers are not sufficient. For memory reasons the timing drivers do not compute the residuals and the orthogonality errors.

5. CONCLUSIONS

We presented a software toolbox for symmetric band reduction. The toolbox contains routines for the reduction of full symmetric matrices to banded form and the reduction of banded matrices to narrower banded or tridiagonal form, with optional accumulation of the orthogonal transformations, as well as repacking routines for storage rearrangement. The software is available from `http://www.mcs.anl.gov` and is intended to enable computational practitioners to experiment with the SBR approach.

References

Anderson, E., Bai, Z., Bischof, C., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Ostrouchov, S., and Sorensen, D. 1995. *LAPACK User's Guide* (2nd ed.). SIAM, Philadelphia.

Bischof, C. H., Lang, B., and Sun, X. 1996. A framework for symmetric band reduction. Preprint ANL/MCS-P586-0496, Mathematics and Computer Science Division, Argonne National Laboratory.

Bischof, C. H. and Sun, X. 1995. On tridiagonalizing and diagonalizing symmetric matrices with repeated eigenvalues. Preprint ANL/MCS-P5454-1095, Mathematics and Computer Science Division, Argonne National Laboratory.

Dongarra, J. J., Hammarling, S. J., and Sorensen, D. C. 1989. Block reduction of matrices to condensed forms for eigenvalue computations. *J. Comput. Appl. Math. 27*, 215–227.

Golub, G. H. and Van Loan, C. F. 1989. *Matrix Computations* (2nd ed.). The Johns Hopkins University Press, Baltimore.

Kaufman, L. 1984. Banded eigenvalue solvers on vector machines. *ACM Trans. Math. Soft. 10*, 1 (March), 73–86.

LANG, B. 1993. A parallel algorithm for reducing symmetric banded matrices to tridiagonal form. *SIAM J. Sci. Comput. 14*, 6 (November), 1320–1338.

LEDERMAN, S., TSAO, A., AND TURNBULL, T. 1991. A parallelizable eigensolver for real diagonalizable matrices with real eigenvalues. Technical Report TR-91-042, Supercomputing Research Center, Institure for Defense Analysis, Bowie.

MURATA, K. AND HORIKOSHI, K. 1975. A new method for the tridiagonalization of the symmetric band matrix. *Information Processing in Japan 15*, 108–112.

SCHWARZ, H. R. 1968. Tridiagonalization of a symmetric band matrix. *Numer. Math. 12*, 231–241.