Part 2

Specifications of Routines

Notes

1. The specifications that follow give the calling sequence, purpose, and descriptions of the arguments of each LAPACK driver and computational routine (but not of auxiliary routines).

2. Specifications of pairs of real and complex routines have been merged (for example SBD-SQR/CBDSQR). In a few cases, specifications of three routines have been merged, one for real symmetric, one for complex symmetric, and one for complex Hermitian matrices (for example SSYTRF/CSYTRF/CHETRF). A few routines for real matrices have no complex equivalent (for example SSTEBZ).

3. Specifications are given only for *single precision* routines. To adapt them for the double precision version of the software, simply interpret REAL as DOUBLE PRECISION, COMPLEX as COMPLEX*16 (or DOUBLE COMPLEX), and the initial letters S- and C- of LAPACK routine names as D- and Z-.

4. Specifications are arranged in alphabetical order of the real routine name.

5. The text of the specifications has been derived from the leading comments in the source-text of the routines. It makes only a limited use of mathematical typesetting facilities. To eliminate redundancy, A^H has been used throughout the specifications. Thus, the reader should note that A^H is equivalent to A^T in the real case.

6. If there is a discrepancy between the specifications listed in this section and the actual source code, the source code should be regarded as the most up-to-date.

SBDSDC

```
      SUBROUTINE SBDSDC( UPLO, COMPQ, N, D, E, U, LDU, VT, LDVT, Q, IQ,
     $                   WORK, IWORK, INFO )
      CHARACTER          COMPQ, UPLO
      INTEGER            INFO, LDU, LDVT, N
      INTEGER            IQ( * ), IWORK( * )
      REAL               D( * ), E( * ), Q( * ), U( LDU, * ),
     $                   VT( LDVT, * ), WORK( * )
```

Purpose

SBDSDC computes the singular value decomposition (SVD) of a real n-by-n (upper or lower) bidiagonal matrix B: B = U*S*VT, using a divide and conquer method, where S is a diagonal matrix with non-negative diagonal elements (the singular values of B), and U and VT are orthogonal matrices of left and right singular vectors, respectively. SBDSDC can be used to compute all singular values, and optionally, singular vectors or singular vectors in compact form.

This code makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

The code currently call SLASDQ if singular values only are desired. However, it can be slightly modified to compute singular values using the divide and conquer method.

Arguments

UPLO (input) CHARACTER*1
- = 'U': B is upper bidiagonal;
- = 'L': B is lower bidiagonal.

COMPQ (input) CHARACTER*1
- = 'N': Compute singular values only;
- = 'P': Compute singular values and compute singular vectors in compact form;
- = 'I': Compute singular values and singular vectors.

N (input) INTEGER
The order of the matrix B. $N \geq 0$.

D (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the bidiagonal matrix B.
On exit, if INFO=0, the singular values of B.

E (input/output) REAL array, dimension (N)
On entry, the elements of E contain the off-diagonal elements of the bidiagonal matrix whose SVD is desired.
On exit, E has been destroyed.

U (output) REAL array, dimension (LDU, N)
If COMPQ = 'I', then:
On exit, if INFO = 0, U contains the left singular vectors of the bidiagonal matrix.
For other values of COMPQ, U is not referenced.

LDU (input) INTEGER
The leading dimension of the array U. $LDU \geq 1$.
If singular vectors are desired, then $LDU \geq \max(1,N)$.

VT (output) REAL array, dimension (LDVT, N)
If COMPQ = 'I', then:
On exit, if INFO = 0, VT' contains the right singular vectors of the bidiagonal matrix.
For other values of COMPQ, VT is not referenced.

LDVT (input) INTEGER
The leading dimension of the array VT. $LDVT \geq 1$. If singular vectors are desired, then $LDVT \geq \max(1,N)$.

Q (output) REAL array, dimension (LDQ)
If COMPQ = 'P', then: On exit, if INFO = 0, Q and IQ contain the left and right singular vectors in a compact form, requiring O(N log N) space instead of $2*N^2$. In particular, Q contains all the REAL data in $LDQ \geq N*(11 + 2*SMLSIZ + 8*INT(LOG_2(N/(SMLSIZ+1))))$ words of memory, where SMLSIZ is returned by ILAENV and is equal to the maximum size of the subproblems at the bottom of the computation tree (usually about 25). For other values of COMPQ, Q is not referenced.

IQ (output) INTEGER array, dimension (LDIQ)
If COMPQ = 'P', then: On exit, if INFO = 0, Q and IQ contain the left and right singular vectors in a compact form, requiring O(N log N) space instead of $2*N^2$. In particular, IQ contains all INTEGER data in $LDIQ \geq N*(3 + 3*INT(LOG_2(N/(SMLSIZ+1))))$ words of memory, where SMLSIZ is returned by ILAENV and is equal to the maximum size of the subproblems at the bottom of the computation tree (usually about 25). For other values of COMPQ, IQ is not referenced.

WORK (workspace) REAL array, dimension (LWORK)
If COMPQ = 'N' then $LWORK \geq (4*N)$.
If COMPQ = 'P' then $LWORK \geq (6*N)$.
If COMPQ = 'I' then $LWORK \geq (3*N^2 + 2*N)$.

IWORK (workspace) INTEGER array, dimension (7*N)

INFO (output) INTEGER
- = 0: successful exit
- < 0: If INFO = −i, the i^{th} argument had an illegal value.
- > 0: the algorithm failed to compute a singular value. The update process of divide and conquer failed.

SBDSQR/CBDSQR

```
      SUBROUTINE SBDSQR( UPLO, N, NCVT, NRU, NCC, D, E, VT, LDVT, U,
     $                   LDU, C, LDC, WORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDC, LDU, LDVT, N, NCC, NCVT, NRU
      REAL               C( LDC, * ), D( * ), E( * ), U( LDU, * ),
     $                   VT( LDVT, * ), WORK( * )

      SUBROUTINE CBDSQR( UPLO, N, NCVT, NRU, NCC, D, E, VT, LDVT, U,
     $                   LDU, C, LDC, WORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDC, LDU, LDVT, N, NCC, NCVT, NRU
      REAL               D( * ), E( * ), WORK( * )
      COMPLEX            C( LDC, * ), U( LDU, * ), VT( LDVT, * )
```

Purpose

SBDSQR/CBDSQR computes the singular value decomposition (SVD) of a real n-by-n (upper or lower) bidiagonal matrix B: $B = Q*S*P^T$, where S is a diagonal matrix with non-negative diagonal elements (the singular values of B), and Q and P are orthogonal matrices.

The routine computes S, and optionally computes U*Q, P^T*VT, or Q^T*C, for given real/complex input matrices U, VT, and C.

Arguments

UPLO (input) CHARACTER*1
= 'U': B is upper bidiagonal;
= 'L': B is lower bidiagonal.

N (input) INTEGER
The order of the matrix B. $N \geq 0$.

NCVT (input) INTEGER
The number of columns of the matrix VT. $NCVT \geq 0$.

NRU (input) INTEGER
The number of rows of the matrix U. $NRU \geq 0$.

NCC (input) INTEGER
The number of columns of the matrix C. $NCC \geq 0$.

D (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the bidiagonal matrix B.
On exit, if INFO=0, the singular values of B in decreasing order.

E (input/output) REAL array, dimension (N)
On entry, the elements of E contain the off-diagonal elements of the bidiagonal matrix whose SVD is desired.
On normal exit (INFO = 0), E is destroyed. If the algorithm does not converge (INFO > 0), D and E will contain the diagonal and superdiagonal elements of a bidiagonal matrix orthogonally equivalent to the one given as input. E(n) is used for workspace.

VT (input/output) REAL/COMPLEX array, dimension (LDVT, NCVT)
On entry, an n-by-ncvt matrix VT.
On exit, VT is overwritten by P^T*VT.
VT is not referenced if NCVT = 0.

LDVT (input) INTEGER
The leading dimension of the array VT.
$LDVT \geq \max(1,N)$ if $NCVT > 0$; $LDVT \geq 1$ if $NCVT = 0$.

U (input/output) REAL/COMPLEX array, dimension (LDU, N)
On entry, an nru-by-n matrix U.
On exit, U is overwritten by U*Q.
U is not referenced if NRU = 0.

LDU (input) INTEGER
The leading dimension of the array U. $LDU \geq \max(1,NRU)$.

C (input/output) REAL/COMPLEX array, dimension (LDC, NCC)
On entry, an n-by-ncc matrix C.
On exit, C is overwritten by Q^T*C.
C is not referenced if NCC = 0.

LDC (input) INTEGER
The leading dimension of the array C.
$LDC \geq \max(1,N)$ if $NCC > 0$; $LDC \geq 1$ if $NCC = 0$.

WORK (workspace) REAL array, dimension (4*N)

INFO (output) INTEGER
= 0: successful exit
< 0: If INFO = −i, the i^{th} argument had an illegal value.
> 0: the algorithm did not converge; D and E contain the elements of a bidiagonal matrix which is orthogonally similar to the input matrix B; if INFO = i, i elements of E have not converged to zero.

SDISNA

```
      SUBROUTINE SDISNA( JOB, M, N, D, SEP, INFO )
      CHARACTER          JOB
      INTEGER            INFO, M, N
      REAL               D( * ), SEP( * )
```

Purpose

SDISNA computes the reciprocal condition numbers for the eigenvectors of a real symmetric or complex Hermitian matrix or for the left or right singular vectors of a general m-by-n matrix. The reciprocal condition number is the 'gap' between the corresponding eigenvalue or singular value and the nearest other one.

The bound on the error, measured by angle in radians, in the i^{th} computed vector is given by

SLAMCH('E')*(ANORM / SEP(i))

where ANORM = || A $||_2$ = max(abs(D(j))). SEP(i) is not allowed to be smaller than SLAMCH('E')*ANORM in order to limit the size of the error bound.

SDISNA may also be used to compute error bounds for eigenvectors of the generalized symmetric definite eigenproblem.

Arguments

JOB (input) CHARACTER*1
Specifies for which problem the reciprocal condition numbers should be computed:
= 'E': the eigenvectors of a symmetric/Hermitian matrix;
= 'L': the left singular vectors of a general matrix;
= 'R': the right singular vectors of a general matrix.

M (input) INTEGER
The number of rows of the matrix. M $\geq$ 0.

N (input) INTEGER
If JOB = 'L' or 'R', the number of columns of the matrix, in which case N $\geq$ 0. Ignored if JOB = 'E'.

D (input) REAL array, dimension (M) if JOB = 'E'
dimension (min(M,N)) if JOB = 'L' or 'R'
The eigenvalues (if JOB = 'E') or singular values (if JOB = 'L' or 'R') of the matrix, in either increasing or decreasing order. If singular values, they must be non-negative.

SEP (output) REAL array, dimension (M) if JOB = 'E'
dimension (min(M,N)) if JOB = 'L' or 'R'
The reciprocal condition numbers of the vectors.

INFO (output) INTEGER
= 0: successful exit.
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGBBRD/CGBBRD

```
      SUBROUTINE SGBBRD( VECT, M, N, NCC, KL, KU, AB, LDAB, D, E, Q,
     $                   LDQ, PT, LDPT, C, LDC, WORK, INFO )
      CHARACTER          VECT
      INTEGER            INFO, KL, KU, LDAB, LDC, LDPT, LDQ, M, N, NCC
      REAL               AB( LDAB, * ), C( LDC, * ), D( * ), E( * ),
     $                   PT( LDPT, * ), Q( LDQ, * ), WORK( * )
```

```
      SUBROUTINE CGBBRD( VECT, M, N, NCC, KL, KU, AB, LDAB, D, E, Q,
     $                   LDQ, PT, LDPT, C, LDC, WORK, RWORK, INFO )
      CHARACTER          VECT
      INTEGER            INFO, KL, KU, LDAB, LDC, LDPT, LDQ, M, N, NCC
      REAL               D( * ), E( * ), RWORK( * )
      COMPLEX            AB( LDAB, * ), C( LDC, * ), PT( LDPT, * ),
     $                   Q( LDQ, * ), WORK( * )
```

Purpose

SGBBRD/CGBBRD reduces a real/complex general m-by-n band matrix A to real upper bidiagonal form B by an orthogonal/unitary transformation: $Q^H*A*P = B$.

The routine computes B, and optionally forms Q or P^H, or computes Q^H*C for a given matrix C.

Arguments

VECT (input) CHARACTER*1
Specifies whether or not the matrices Q and P^H are to be formed.
= 'N': do not form Q or P^H;
= 'Q': form Q only;
= 'P': form P^H only;
= 'B': form both.

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0.

NCC (input) INTEGER
The number of columns of the matrix C. NCC $\geq$ 0.

KL (input) INTEGER
The number of subdiagonals of the matrix A. KL $\geq$ 0.

KU (input) INTEGER
The number of superdiagonals of the matrix A. KU $\geq$ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB,N)
On entry, the m-by-n band matrix A, stored in rows 1 to kl+ku+1. The j^{th} column of A is stored in the j^{th} column of the array AB as follows: AB(ku+1+i−j,j) = A(i,j) for max(1,j−ku)$\leq$i$\leq$min(m,j+kl).
On exit, A is overwritten by values generated during the reduction.

LDAB (input) INTEGER
The leading dimension of the array A. LDAB $\geq$ KL+KU+1.

D (output) REAL array, dimension (min(M,N))
The diagonal elements of the bidiagonal matrix B.

E (output) REAL array, dimension (min(M,N)−1)
The superdiagonal elements of the bidiagonal matrix B.

Q (output) REAL/COMPLEX array, dimension (LDQ,M)
If VECT = 'Q' or 'B', the m-by-m orthogonal/unitary matrix Q.
If VECT = 'N' or 'P', the array Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q.
LDQ $\geq$ max(1,M) if VECT = 'Q' or 'B'; LDQ $\geq$ 1 otherwise.

PT (output) REAL/COMPLEX array, dimension (LDPT,N)
If VECT = 'P' or 'B', the n-by-n orthogonal/unitary matrix P^H.
If VECT = 'N' or 'Q', the array PT is not referenced.

LDPT (input) INTEGER
The leading dimension of the array PT.
LDPT $\geq$ max(1,N) if VECT = 'P' or 'B'; LDPT $\geq$ 1 otherwise.

C (input/output) REAL/COMPLEX array, dimension (LDC,NCC)
On entry, an m-by-ncc matrix C.
On exit, C is overwritten by Q^H*C.
C is not referenced if NCC = 0.

LDC (input) INTEGER
The leading dimension of the array C.
LDC $\geq$ max(1,M) if NCC > 0; LDC $\geq$ 1 if NCC = 0.

WORK *SGBBRD* (workspace) REAL array, dimension (2*max(M,N))
CGBBRD (workspace) COMPLEX array, dimension (max(M,N))

RWORK *CGBBRD only* (workspace) REAL array, dimension (max(M,N))

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGBCON/CGBCON

```
      SUBROUTINE SGBCON( NORM, N, KL, KU, AB, LDAB, IPIV, ANORM, RCOND,
     $                   WORK, IWORK, INFO )
      CHARACTER          NORM
      INTEGER            INFO, KL, KU, LDAB, N
      REAL               ANORM, RCOND
      INTEGER            IPIV( * ), IWORK( * )
      REAL               AB( LDAB, * ), WORK( * )

      SUBROUTINE CGBCON( NORM, N, KL, KU, AB, LDAB, IPIV, ANORM, RCOND,
     $                   WORK, RWORK, INFO )
      CHARACTER          NORM
      INTEGER            INFO, KL, KU, LDAB, N
      REAL               ANORM, RCOND
      INTEGER            IPIV( * )
      REAL               RWORK( * )
      COMPLEX            AB( LDAB, * ), WORK( * )
```

Purpose

SGBCON/CGBCON estimates the reciprocal of the condition number of a real/complex general band matrix A, in either the 1-norm or the infinity-norm, using the LU factorization computed by SGBTRF/CGBTRF.

An estimate is obtained for $\|A^{-1}\|$, and the reciprocal of the condition number is computed as RCOND = $1/(\|A\| * \|A^{-1}\|)$.

Arguments

NORM (input) CHARACTER*1
Specifies whether the 1-norm condition number or the infinity-norm condition number is required:
= '1' or 'O': 1-norm;
= 'I': Infinity-norm.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

KL (input) INTEGER
The number of subdiagonals within the band of A. KL $\geq$ 0.

KU (input) INTEGER
The number of superdiagonals within the band of A. KU $\geq$ 0.

AB (input) REAL/COMPLEX array, dimension (LDAB,N)
Details of the LU factorization of the band matrix A, as computed by SGBTRF/CGBTRF. U is stored as an upper triangular band matrix with kl+ku superdiagonals in rows 1 to kl+ku+1, and the multipliers used during the factorization are stored in rows kl+ku+2 to 2*kl+ku+1.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ 2*kl+ku+1.

IPIV (input) INTEGER array, dimension (N)
The pivot indices; for 1 $\leq$ i $\leq$ N, row i of the matrix was interchanged with row IPIV(i).

ANORM (input) REAL
If NORM = '1' or 'O', the 1-norm of the original matrix A.
If NORM = 'I', the infinity-norm of the original matrix A.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as RCOND = $1/(\|A\| * \|A^{-1}\|)$.

WORK *SGBCON* (workspace) REAL array, dimension (3*N)
CGBCON (workspace) COMPLEX array, dimension (2*N)

IWORK *SGBCON only* (workspace) INTEGER array, dimension (N)

RWORK *CGBCON only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGBEQU/CGBEQU

```
      SUBROUTINE SGBEQU( M, N, KL, KU, AB, LDAB, R, C, ROWCND, COLCND,
     $                   AMAX, INFO )
      INTEGER            INFO, KL, KU, LDAB, M, N
      REAL               AMAX, COLCND, ROWCND
      REAL               AB( LDAB, * ), C( * ), R( * )

      SUBROUTINE CGBEQU( M, N, KL, KU, AB, LDAB, R, C, ROWCND, COLCND,
     $                   AMAX, INFO )
      INTEGER            INFO, KL, KU, LDAB, M, N
      REAL               AMAX, COLCND, ROWCND
      REAL               C( * ), R( * )
      COMPLEX            AB( LDAB, * )
```

Purpose

SGBEQU/CGBEQU computes row and column scalings intended to equilibrate an m-by-n band matrix A and reduce its condition number. R returns the row scale factors and C the column scale factors, chosen to try to make the largest element in each row and column of the matrix B with elements $B(i,j) = R(i)*A(i,j)*C(j)$ have absolute value 1.

R(i) and C(j) are restricted to be between SMLNUM = smallest safe number and BIGNUM = largest safe number. Use of these scaling factors is not guaranteed to reduce the condition number of A but works well in practice.

Arguments

M (input) INTEGER
The number of rows of the matrix A. $M \geq 0$.

N (input) INTEGER
The number of columns of the matrix A. $N \geq 0$.

KL (input) INTEGER
The number of subdiagonals within the band of A. $KL \geq 0$.

KU (input) INTEGER
The number of superdiagonals within the band of A. $KU \geq 0$.

AB (input) REAL/COMPLEX array, dimension (LDAB,N)
The band matrix A, stored in rows 1 to kl+ku+1. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
$AB(ku+1+i-j,j) = A(i,j)$ for $\max(1,j-ku) \leq i \leq \min(m,j+kl)$.

LDAB (input) INTEGER
The leading dimension of the array AB. $LDAB \geq KL+KU+1$.

R (output) REAL array, dimension (M)
If INFO = 0, or INFO > M, R contains the row scale factors for A.

C (output) REAL array, dimension (N)
If INFO = 0, C contains the column scale factors for A.

ROWCND (output) REAL
If INFO = 0 or INFO > M, ROWCND contains the ratio of the smallest R(i) to the largest R(i). If ROWCND $\geq$ 0.1 and AMAX is neither too large nor too small, it is not worth scaling by R.

COLCND (output) REAL
If INFO = 0, COLCND contains the ratio of the smallest C(i) to the largest C(i). If COLCND $\geq$ 0.1, it is not worth scaling by C.

AMAX (output) REAL
Absolute value of largest matrix element. If AMAX is very close to overflow or very close to underflow, the matrix should be scaled.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = $-i$, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
$\leq$ M: the i^{th} row of A is exactly zero
> M: the $(i\text{-}M)^{th}$ column of A is exactly zero

SGBRFS/CGBRFS

```
      SUBROUTINE SGBRFS( TRANS, N, KL, KU, NRHS, AB, LDAB, AFB, LDAFB,
     $                   IPIV, B, LDB, X, LDX, FERR, BERR, WORK, IWORK,
     $                   INFO )
      CHARACTER          TRANS
      INTEGER            INFO, KL, KU, LDAB, LDAFB, LDB, LDX, N, NRHS
      INTEGER            IPIV( * ), IWORK( * )
      REAL               AB( LDAB, * ), AFB( LDAFB, * ), B( LDB, * ),
     $                   BERR( * ), FERR( * ), WORK( * ), X( LDX, * )

      SUBROUTINE CGBRFS( TRANS, N, KL, KU, NRHS, AB, LDAB, AFB, LDAFB,
     $                   IPIV, B, LDB, X, LDX, FERR, BERR, WORK, RWORK,
     $                   INFO )
      CHARACTER          TRANS
      INTEGER            INFO, KL, KU, LDAB, LDAFB, LDB, LDX, N, NRHS
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            AB( LDAB, * ), AFB( LDAFB, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SGBRFS/CGBRFS improves the computed solution to a system of linear equations when the coefficient matrix is banded, and provides error bounds and backward error estimates for the solution.

Arguments

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)
= 'C': A^H*X = B (Conjugate transpose)

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

KL (input) INTEGER
The number of subdiagonals within the band of A. KL $\geq$ 0.

KU (input) INTEGER
The number of superdiagonals within the band of A. KU $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

AB (input) REAL/COMPLEX array, dimension (LDAB,N)
The original band matrix A, stored in rows 1 to kl+ku+1.
The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
AB(ku+1+i−j,j) = A(i,j) for max(1,j−ku)$\leq$i$\leq$min(n,j+kl).

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KL+KU+1.

AFB (input) REAL/COMPLEX array, dimension (LDAFB,N)
Details of the LU factorization of the band matrix A, as computed by SGBTRF/CGBTRF. U is stored as an upper triangular band matrix with kl+ku superdiagonals in rows 1 to kl+ku+1, and the multipliers used during the factorization are stored in rows kl+ku+2 to 2*kl+ku+1.

LDAFB (input) INTEGER
The leading dimension of the array AFB. LDAFB $\geq$ 2*KL*KU+1.

IPIV (input) INTEGER array, dimension (N)
The pivot indices from SGBTRF/CGBTRF; for 1$\leq$i$\leq$N, row i of the matrix was interchanged with row IPIV(i).

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (input/output) REAL/COMPLEX array, dimension (LDX,NRHS)
On entry, the solution matrix X, as computed by SGBTRS/CGBTRS.
On exit, the improved solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SGBRFS* (workspace) REAL array, dimension (3*N)
CGBRFS (workspace) COMPLEX array, dimension (2*N)

IWORK *SGBRFS only* (workspace) INTEGER array, dimension (N)

RWORK *CGBRFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGBSV/CGBSV

```
SUBROUTINE SGBSV( N, KL, KU, NRHS, AB, LDAB, IPIV, B, LDB, INFO )
INTEGER            INFO, KL, KU, LDAB, LDB, N, NRHS
INTEGER            IPIV( * )
REAL               AB( LDAB, * ), B( LDB, * )

SUBROUTINE CGBSV( N, KL, KU, NRHS, AB, LDAB, IPIV, B, LDB, INFO )
INTEGER            INFO, KL, KU, LDAB, LDB, N, NRHS
INTEGER            IPIV( * )
COMPLEX            AB( LDAB, * ), B( LDB, * )
```

Purpose

SGBSV/CGBSV computes the solution to a real/complex system of linear equations A*X = B, where A is a band matrix of order n with kl subdiagonals and ku superdiagonals, and X and B are n-by-nrhs matrices.

The LU decomposition with partial pivoting and row interchanges is used to factor A as A = L*U, where L is a product of permutation and unit lower triangular matrices with kl subdiagonals, and U is upper triangular with kl+ku superdiagonals. The factored form of A is then used to solve the system of equations A*X = B.

Arguments

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N $\geq$ 0.

KL (input) INTEGER
The number of subdiagonals within the band of A. KL $\geq$ 0.

KU (input) INTEGER
The number of superdiagonals within the band of A. KU $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB,N)
On entry, the matrix A in band storage, in rows kl+1 to 2*kl+ku+1; rows 1 to kl of the array need not be set. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
AB(kl+ku+1+i−j,j) = A(i,j) for max(1,j−ku)$\leq$i$\leq$min(n,j+kl)
On exit, details of the factorization: U is stored as an upper triangular band matrix with kl+ku superdiagonals in rows 1 to kl+ku+1, and the multipliers used during the factorization are stored in rows kl+ku+2 to 2*kl+ku+1.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ 2*KL+KU+1.

IPIV (output) INTEGER array, dimension (N)
The pivot indices that define the permutation matrix P; row i of the matrix was interchanged with row IPIV(i).

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if INFO = 0, the n-by-nrhs solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, U(i,i) is exactly zero. The factorization has been completed, but the factor U is exactly singular, and the solution has not been computed.

SGBSVX/CGBSVX

```
      SUBROUTINE SGBSVX( FACT, TRANS, N, KL, KU, NRHS, AB, LDAB, AFB,
     $                   LDAFB, IPIV, EQUED, R, C, B, LDB, X, LDX,
     $                   RCOND, FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          EQUED, FACT, TRANS
      INTEGER            INFO, KL, KU, LDAB, LDAFB, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * ), IWORK( * )
      REAL               AB( LDAB, * ), AFB( LDAFB, * ), B( LDB, * ),
     $                   BERR( * ), C( * ), FERR( * ), R( * ),
     $                   WORK( * ), X( LDX, * )

      SUBROUTINE CGBSVX( FACT, TRANS, N, KL, KU, NRHS, AB, LDAB, AFB,
     $                   LDAFB, IPIV, EQUED, R, C, B, LDB, X, LDX,
     $                   RCOND, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          EQUED, FACT, TRANS
      INTEGER            INFO, KL, KU, LDAB, LDAFB, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * )
      REAL               BERR( * ), C( * ), FERR( * ), R( * ),
     $                   RWORK( * )
      COMPLEX            AB( LDAB, * ), AFB( LDAFB, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SGBSVX/CGBSVX uses the LU factorization to compute the solution to a real/complex system of linear equations A*X = B, A^T*X = B, or A^H*X = B, where A is a band matrix of order n with kl subdiagonals and ku superdiagonals, and X and B are n-by-nrhs matrices.

Error bounds on the solution and a condition estimate are also provided.

Description

The following steps are performed:

1. If FACT = 'E', real scaling factors are computed to equilibrate the system:

   TRANS = 'N': diag(R)*A*diag(C)*diag(C)$^{-1}$*X = diag(R)*B
   TRANS = 'T': (diag(R)*A*diag(C))^T*diag(R)$^{-1}$*X = diag(C)*B
   TRANS = 'C': (diag(R)*A*diag(C))^H*diag(R)$^{-1}$*X = diag(C)*B

   Whether or not the system will be equilibrated depends on the scaling of the matrix A, but if equilibration is used, A is overwritten by diag(R)*A*diag(C) and B by diag(R)*B (if TRANS='N') or diag(C)*B (if TRANS = 'T' or 'C').

2. If FACT = 'N' or 'E', the LU decomposition is used to factor the matrix A (after equilibration if FACT = 'E') as A = L*U, where L is a product of permutation and unit lower triangular matrices with kl subdiagonals, and U is upper triangular with kl+ku superdiagonals.

3. If some U(i,i)=0, so that U is exactly singular, then the routine returns with INFO = i. Otherwise, the factored form of A is used to estimate the condition number of the matrix A. If the reciprocal of the condition number is less than machine precision, INFO = N+1 is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.
4. The system of equations is solved for X using the factored form of A.
5. Iterative refinement is applied to improve the computed solution matrix and calculate error bounds and backward error estimates for it.
6. If equilibration was used, the matrix X is premultiplied by diag(C) (if TRANS = 'N') or diag(R) (if TRANS = 'T' or 'C') so that it solves the original system before equilibration.

Arguments

FACT (input) CHARACTER*1
Specifies whether or not the factored form of the matrix A is supplied on entry, and if not, whether the matrix A should be equilibrated before it is factored.
= 'F': On entry, AFB and IPIV contain the factored form of A. If EQUED ≠ 'N', the matrix A has been equilibrated with scaling factors given by R and C.
AB, AFB, and IPIV are not modified.
= 'N': The matrix A will be copied to AFB and factored.
= 'E': The matrix A will be equilibrated if necessary, then copied to AFB and factored.

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)
= 'C': A^H*X = B (Conjugate Transpose)

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N ≥ 0.

KL (input) INTEGER
The number of subdiagonals within the band of A. KL ≥ 0.

KU (input) INTEGER
The number of superdiagonals within the band of A. KU ≥ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS ≥ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB,N)
On entry, the matrix A in band storage, in rows 1 to kl+ku+1. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
AB(ku+1+i−j,j) = A(i,j) for max(1,j−ku)≤i≤min(n,j+kl)
If FACT = 'F' and EQUED is not 'N', then A must have been equilibrated by the scaling factors in R and/or C. AB is not modified if FACT = 'F' or 'N', or if FACT = 'E' and EQUED = 'N' on exit.
On exit, if EQUED ≠ 'N', A is scaled as follows:
EQUED = 'R': A := diag(R)*A;
EQUED = 'C': A := A*diag(C);
EQUED = 'B': A := diag(R)*A*diag(C).

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB ≥ KL+KU+1.

AFB (input or output) REAL/COMPLEX array, dimension (LDAFB,N)
If FACT = 'F', then AFB is an input argument and on entry contains details of the LU factorization of the band matrix A, as computed by SGBTRF/CGBTRF. U is stored as an upper triangular band matrix with kl+ku superdiagonals in rows 1 to kl+ku+1, and the multipliers used during the factorization are stored in rows kl+ku+2 to 2*kl+ku+1.
If EQUED ≠ 'N', then AFB is the factored form of the equilibrated matrix A.
If FACT = 'N', then AFB is an output argument and on exit returns details of the LU factorization of A.
If FACT = 'E', then AFB is an output argument and on exit returns details of the LU factorization of the equilibrated matrix A (see the description of AB for the form of the equilibrated matrix).

LDAFB (input) INTEGER
The leading dimension of the array AFB. LDAFB ≥ 2*KL+KU+1.

IPIV (input or output) INTEGER array, dimension (N)
If FACT = 'F', then IPIV is an input argument and on entry contains the pivot indices from the factorization A = L*U as computed by SGBTRF/CGBTRF; row i of the matrix was interchanged with row IPIV(i).
If FACT = 'N', then IPIV is an output argument and on exit contains the pivot indices from the factorization A = L*U of the original matrix A.
If FACT = 'E', then IPIV is an output argument and on exit contains the pivot indices from the factorization A = L*U of the equilibrated matrix A.

EQUED (input or output) CHARACTER*1
Specifies the form of equilibration that was done:
= 'N': No equilibration (always true if FACT = 'N').
= 'R': Row equilibration, i.e., A has been premultiplied by diag(R).
= 'C': Column equilibration, i.e., A has been postmultiplied by diag(C).
= 'B': Both row and column equilibration, i.e., A has been replaced by diag(R)*A*diag(C).
EQUED is an input argument if FACT = 'F'; otherwise, it is an output argument.

R (input or output) REAL array, dimension (N)
The row scale factors for A. If EQUED = 'R' or 'B', A is multiplied on the left by diag(R); if EQUED = 'N' or 'C', R is not accessed. R is an input argument if FACT = 'F'; otherwise, R is an output argument.

If FACT = 'F' and EQUED = 'R' or 'B', each element of R must be positive.

C (input or output) REAL array, dimension (N)
The column scale factors for A. If EQUED = 'C' or 'B', A is multiplied on the right by diag(C); if EQUED = 'N' or 'R', C is not accessed. C is an input argument if FACT = 'F'; otherwise, C is an output argument. If FACT = 'F' and EQUED = 'C' or 'B', each element of C must be positive.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit,
if EQUED = 'N', B is not modified;
if TRANS = 'N' and EQUED = 'R' or 'B', B is overwritten by diag(R)*B;
if TRANS = 'T' or 'C' and EQUED = 'C' or 'B', B is overwritten by diag(C)*B.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,N).

X (output) REAL/COMPLEX array, dimension (LDX,NRHS)
If INFO = 0 or INFO = N+1, the N-by-NRHS solution matrix X to the original system of equations. Note that A and B are modified on exit if EQUED .ne. 'N', and the solution to the equilibrated system is $\mathrm{diag}(C)^{-1}*X$ if TRANS = 'N' and EQUED = 'C' or 'B', or $\mathrm{diag}(R)^{-1}*X$ if TRANS = 'T' or 'C' and EQUED = 'R' or 'B'.

LDX (input) INTEGER
The leading dimension of the array X. LDX ≥ max(1,N).

RCOND (output) REAL
The estimate of the reciprocal condition number of the matrix A after equilibration (if done). If RCOND is less than the machine precision (in particular, if RCOND = 0), the matrix is singular to working precision. This condition is indicated by a return code of INFO > 0.

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SGBSVX* (workspace/output) REAL array, dimension (3*N)
CGBSVX (workspace) COMPLEX array, dimension (2*N)
SGBSVX only
On exit, WORK(1) contains the reciprocal pivot growth factor $\|A\|/\|U\|$. The "max absolute element" norm is used. If WORK(1) is much less than 1, then the stability of the LU factorization of the (equilibrated) matrix A could be poor. This also means that the solution X, condition estimator RCOND, and forward error bound FERR could be unreliable. If factorization fails with 0<INFO≤N, then WORK(1) contains the reciprocal pivot growth factor for the leading INFO columns of A.

IWORK *SGBSVX only* (workspace) INTEGER array, dimension (N)

RWORK *CGBSVX only* (workspace/output) REAL array, dimension (N)
On exit, RWORK(1) contains the reciprocal pivot growth factor $\|A\|/\|U\|$. The "max absolute element" norm is used. If RWORK(1) is much less than 1, then the stability of the LU factorization of the (equilibrated) matrix A could be poor. This also means that the solution X, condition estimator RCOND, and forward error bound FERR could be unreliable. If factorization fails with 0<INFO≤N, then RWORK(1) contains the reciprocal pivot growth factor for the leading INFO columns of A.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
≤ N: U(i,i) is exactly zero. The factorization has been completed, but the factor U is exactly singular, so the solution and error bounds could not be computed. RCOND = 0 is returned.
= N+1: U is nonsingular, but RCOND is less than machine precision, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of RCOND would suggest.

SGBTRF/CGBTRF

```
SUBROUTINE SGBTRF( M, N, KL, KU, AB, LDAB, IPIV, INFO )
INTEGER            INFO, KL, KU, LDAB, M, N
INTEGER            IPIV( * )
REAL               AB( LDAB, * )

SUBROUTINE CGBTRF( M, N, KL, KU, AB, LDAB, IPIV, INFO )
INTEGER            INFO, KL, KU, LDAB, M, N
INTEGER            IPIV( * )
COMPLEX            AB( LDAB, * )
```

Purpose

SGBTRF/CGBTRF computes an LU factorization of a real/complex m-by-n band matrix A using partial pivoting with row interchanges.

Arguments

M — (input) INTEGER
The number of rows of the matrix A. M ≥ 0.

N — (input) INTEGER
The number of columns of the matrix A. N ≥ 0.

KL — (input) INTEGER
The number of subdiagonals within the band of A. KL ≥ 0.

KU — (input) INTEGER
The number of superdiagonals within the band of A. KU ≥ 0.

AB — (input/output) REAL/COMPLEX array, dimension (LDAB,N)
On entry, the matrix A in band storage, in rows kl+1 to 2*kl+ku+1; rows 1 to kl of the array need not be set. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
AB(kl+ku+1+i−j,j) = A(i,j) for max(1,j−ku)≤i≤min(m,j+kl)
On exit, details of the factorization: U is stored as an upper triangular band matrix with kl+ku superdiagonals in rows 1 to kl+ku+1, and the multipliers used during the factorization are stored in rows kl+ku+2 to 2*kl+ku+1.

LDAB — (input) INTEGER
The leading dimension of the array AB. LDAB ≥ 2*KL+KU+1.

IPIV — (output) INTEGER array, dimension (min(M,N))
The pivot indices; for 1 ≤ i ≤ min(M,N), row i of the matrix was interchanged with row IPIV(i).

INFO — (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, U(i,i) is exactly zero. The factorization has been completed, but the factor U is exactly singular, and division by zero will occur if it is used to solve a system of equations.

SGBTRS/CGBTRS

```
      SUBROUTINE SGBTRS( TRANS, N, KL, KU, NRHS, AB, LDAB, IPIV, B, LDB,
     $                   INFO )
      CHARACTER          TRANS
      INTEGER            INFO, KL, KU, LDAB, LDB, N, NRHS
      INTEGER            IPIV( * )
      REAL               AB( LDAB, * ), B( LDB, * )
```

```
      SUBROUTINE CGBTRS( TRANS, N, KL, KU, NRHS, AB, LDAB, IPIV, B, LDB,
     $                   INFO )
      CHARACTER          TRANS
      INTEGER            INFO, KL, KU, LDAB, LDB, N, NRHS
      INTEGER            IPIV( * )
      COMPLEX            AB( LDAB, * ), B( LDB, * )
```

Purpose

SGBTRS/CGBTRS solves a system of linear equations A*X = B, A^T*X = B, or A^H*X = B, with a general band matrix A using the LU factorization computed by SGBTRF/CGBTRF.

Arguments

TRANS — (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)
= 'C': A^H*X = B (Conjugate Transpose)

N — (input) INTEGER
The order of the matrix A. N ≥ 0.

KL — (input) INTEGER
The number of subdiagonals within the band of A. KL ≥ 0.

KU — (input) INTEGER
The number of superdiagonals within the band of A. KU ≥ 0.

NRHS — (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS ≥ 0.

AB — (input) REAL/COMPLEX array, dimension (LDAB,N)
Details of the LU factorization of the band matrix A, as computed by SGBTRF/CGBTRF. U is stored as an upper triangular band matrix with kl+ku superdiagonals in rows 1 to kl+ku+1, and the multipliers used during the factorization are stored in rows kl+ku+2 to 2*kl+ku+1.

LDAB — (input) INTEGER
The leading dimension of the array AB. LDAB ≥ 2*KL+KU+1.

IPIV — (input) INTEGER array, dimension (N)
The pivot indices; for 1 ≤ i ≤ N, row i of the matrix was interchanged with row IPIV(i).

B — (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, the solution matrix X.

LDB — (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,N).

INFO — (output) INTEGER
= 0: successful exit

< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGEBAK/CGEBAK

```
      SUBROUTINE SGEBAK( JOB, SIDE, N, ILO, IHI, SCALE, M, V, LDV,
     $                   INFO )
      CHARACTER          JOB, SIDE
      INTEGER            IHI, ILO, INFO, LDV, M, N
      REAL               V( LDV, * ), SCALE( * )

      SUBROUTINE CGEBAK( JOB, SIDE, N, ILO, IHI, SCALE, M, V, LDV,
     $                   INFO )
      CHARACTER          JOB, SIDE
      INTEGER            IHI, ILO, INFO, LDV, M, N
      REAL               SCALE( * )
      COMPLEX            V( LDV, * )
```

Purpose

SGEBAK/CGEBAK forms the right or left eigenvectors of a real/complex general matrix by backward transformation on the computed eigenvectors of the balanced matrix output by SGEBAL/CGEBAL.

Arguments

JOB (input) CHARACTER*1
Specifies the type of backward transformation required:
= 'N': do nothing, return immediately;
= 'P': do backward transformation for permutation only;
= 'S': do backward transformation for scaling only;
= 'B': do backward transformations for both permutation and scaling.
JOB must be the same as the argument JOB supplied to SGEBAL/CGEBAL.

SIDE (input) CHARACTER*1
= 'R': V contains right eigenvectors;
= 'L': V contains left eigenvectors.

N (input) INTEGER
The number of rows of the matrix V. N ≥ 0.

ILO, IHI (input) INTEGER
The integers ILO and IHI determined by SGEBAL/CGEBAL.
$1 \le ILO \le IHI \le N$, if $N > 0$; ILO = 1 and IHI = 0, if N = 0.

SCALE (input) REAL array, dimension (N)
Details of the permutation and scaling factors, as returned by SGEBAL/CGEBAL.

M (input) INTEGER
The number of columns of the matrix V. M ≥ 0.

V (input/output) REAL/COMPLEX array, dimension (LDV,M)
On entry, the matrix of right or left eigenvectors to be transformed, as returned by SHSEIN/CHSEIN or STREVC/CTREVC.
On exit, V is overwritten by the transformed eigenvectors.

LDV (input) INTEGER
The leading dimension of the array V. LDV ≥ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGEBAL/CGEBAL

```
      SUBROUTINE SGEBAL( JOB, N, A, LDA, ILO, IHI, SCALE, INFO )
      CHARACTER          JOB
      INTEGER            IHI, ILO, INFO, LDA, N
      REAL               A( LDA, * ), SCALE( * )

      SUBROUTINE CGEBAL( JOB, N, A, LDA, ILO, IHI, SCALE, INFO )
      CHARACTER          JOB
      INTEGER            IHI, ILO, INFO, LDA, N
      REAL               SCALE( * )
      COMPLEX            A( LDA, * )
```

Purpose

SGEBAL/CGEBAL balances a general real/complex matrix A. This involves, first, permuting A by a similarity transformation to isolate eigenvalues in the first 1 to ilo−1 and last ihi+1 to n elements on the diagonal; and second, applying a diagonal similarity transformation to rows and columns ilo to ihi to make the rows and columns as close in norm as possible. Both steps are optional.

Balancing may reduce the 1-norm of the matrix, and improve the accuracy of the computed eigenvalues and/or eigenvectors.

Arguments

JOB (input) CHARACTER*1
Specifies the operations to be performed on A:
= 'N': none: simply set ILO = 1, IHI = N, SCALE(I) = 1.0 for i = 1,...,N;
= 'P': permute only;
= 'S': scale only;
= 'B': both permute and scale.

N (input) INTEGER
The order of the matrix A. N ≥ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the input matrix A.

On exit, A is overwritten by the balanced matrix.
If JOB = 'N', A is not referenced.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

ILO, IHI (output) INTEGER
ILO and IHI are set to integers such that on exit A(i,j) = 0 if i > j and j = 1,...,ilo−1 or i = ihi+1,...,n.
If JOB = 'N' or 'S', ILO = 1 and IHI = N.

SCALE (output) REAL array, dimension (N)
Details of the permutations and scaling factors applied to A. If P(j) is the index of the row and column interchanged with row and column j, and D(j) is the scaling factor applied to row and column j, then
SCALE(j) = P(j) for j = 1,...,ilo−1
= D(j) for j = ilo,...,ihi
= P(j) for j = ihi+1,...,n.
The order in which the interchanges are made is n to ihi+1, then 1 to ilo−1.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGEBRD/CGEBRD

```
      SUBROUTINE SGEBRD( M, N, A, LDA, D, E, TAUQ, TAUP, WORK, LWORK,
     $                   INFO )
      INTEGER            INFO, LDA, LWORK, M, N
      REAL               A( LDA, * ), D( * ), E( * ), TAUP( * ),
     $                   TAUQ( * ), WORK( LWORK )

      SUBROUTINE CGEBRD( M, N, A, LDA, D, E, TAUQ, TAUP, WORK, LWORK,
     $                   INFO )
      INTEGER            INFO, LDA, LWORK, M, N
      REAL               D( * ), E( * )
      COMPLEX            A( LDA, * ), TAUP( * ), TAUQ( * ),
     $                   WORK( LWORK )
```

Purpose

SGEBRD/CGEBRD reduces a general real/complex m-by-n matrix A to upper or lower bidiagonal form B by an orthogonal/unitary transformation: $Q^H*A*P = B$.

If m $\geq$ n, B is upper bidiagonal; if m < n, B is lower bidiagonal.

Arguments

M (input) INTEGER
The number of rows in the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns in the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n general matrix to be reduced.
On exit, if m $\geq$ n, the diagonal and the first superdiagonal are overwritten with the upper bidiagonal matrix B; the elements below the diagonal, with the array TAUQ, represent the orthogonal/unitary matrix Q as a product of elementary reflectors, and the elements above the first superdiagonal, with the array TAUP, represent the orthogonal/unitary matrix P as a product of elementary reflectors; if m < n, the diagonal and the first subdiagonal are overwritten with the lower bidiagonal matrix B; the elements below the first subdiagonal, with the array TAUQ, represent the orthogonal/unitary matrix Q as a product of elementary reflectors, and the elements above the diagonal, with the array TAUP, represent the orthogonal/unitary matrix P as a product of elementary reflectors.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

D (output) REAL array, dimension (min(M,N))
The diagonal elements of the bidiagonal matrix B: D(i) = A(i,i).

E (output) REAL array, dimension (min(M,N)−1)
The off-diagonal elements of the bidiagonal matrix B:
if m $\geq$ n, E(i) = A(i,i+1) for i = 1,2,...,n−1;
if m < n, E(i) = A(i+1,i) for i = 1,2,...,m−1.

TAUQ (output) REAL/COMPLEX array dimension (min(M,N))
The scalar factors of the elementary reflectors which represent the orthogonal/unitary matrix Q.

TAUP (output) REAL/COMPLEX array, dimension (min(M,N))
The scalar factors of the elementary reflectors which represent the orthogonal/unitary matrix P.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The length of the array WORK. LWORK $\geq$ max(1,M,N). For optimum performance LWORK $\geq$ (M+N)*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGECON/CGECON

```
      SUBROUTINE SGECON( NORM, N, A, LDA, ANORM, RCOND, WORK, IWORK,
     $                   INFO )
      CHARACTER          NORM
      INTEGER            INFO, LDA, N
      REAL               ANORM, RCOND
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), WORK( * )

      SUBROUTINE CGECON( NORM, N, A, LDA, ANORM, RCOND, WORK, RWORK,
     $                   INFO )
      CHARACTER          NORM
      INTEGER            INFO, LDA, N
      REAL               ANORM, RCOND
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), WORK( * )
```

Purpose

SGECON/CGECON estimates the reciprocal of the condition number of a general real/complex matrix A, in either the 1-norm or the infinity-norm, using the LU factorization computed by SGETRF/CGETRF.

An estimate is obtained for $||A^{-1}||$, and the reciprocal of the condition number is computed as RCOND $= 1/(||A|| * ||A^{-1}||)$.

Arguments

NORM (input) CHARACTER*1
Specifies whether the 1−norm condition number or the infinity-norm condition number is required:
= '1' or 'O': 1-norm;
= 'I': Infinity-norm.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The factors L and U from the factorization A = P*L*U as computed by SGETRF/CGETRF.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

ANORM (input) REAL
If NORM = '1' or 'O', the 1-norm of the original matrix A.
If NORM = 'I', the infinity-norm of the original matrix A.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as RCOND $= 1/(||A|| * ||A^{-1}||)$.

WORK *SGECON* (workspace) REAL array, dimension (4*N)
CGECON (workspace) COMPLEX array, dimension (2*N)

IWORK *SGECON only* (workspace) INTEGER array, dimension (N)

RWORK *CGECON only* (workspace) REAL array, dimension (2*N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGEEQU/CGEEQU

```
      SUBROUTINE SGEEQU( M, N, A, LDA, R, C, ROWCND, COLCND, AMAX,
     $                   INFO )
      INTEGER            INFO, LDA, M, N
      REAL               AMAX, COLCND, ROWCND
      REAL               A( LDA, * ), C( * ), R( * )

      SUBROUTINE CGEEQU( M, N, A, LDA, R, C, ROWCND, COLCND, AMAX,
     $                   INFO )
      INTEGER            INFO, LDA, M, N
      REAL               AMAX, COLCND, ROWCND
      REAL               C( * ), R( * )
      COMPLEX            A( LDA, * )
```

Purpose

SGEEQU/CGEEQU computes row and column scalings intended to equilibrate an m-by-n matrix A and reduce its condition number. R returns the row scale factors and C the column scale factors, chosen to try to make the largest element in each row and column of the matrix B with elements B(i,j) = R(i)*A(i,j)*C(j) have absolute value 1.

R(i) and C(j) are restricted to be between SMLNUM = smallest safe number and BIGNUM = largest safe number. Use of these scaling factors is not guaranteed to reduce the condition number of A but works well in practice.

Arguments

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The m-by-n matrix whose equilibration factors are to be computed.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

R (output) REAL array, dimension (M)
If INFO = 0 or INFO > M, R contains the row scale factors for A.

C (output) REAL array, dimension (N)
If INFO = 0, C contains the column scale factors for A.

ROWCND (output) REAL
If INFO = 0 or INFO > M, ROWCND contains the ratio of the smallest R(i) to the largest R(i). If ROWCND $\geq$ 0.1 and AMAX is neither too large nor too small, it is not worth scaling by R.

COLCND (output) REAL
If INFO = 0, COLCND contains the ratio of the smallest C(i) to the largest C(i). If COLCND $\geq$ 0.1, it is not worth scaling by C.

AMAX (output) REAL
Absolute value of largest matrix element. If AMAX is very close to overflow or very close to underflow, the matrix should be scaled.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = $-$i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
$\leq$ M: the i^{th} row of A is exactly zero
> M: the $(i\text{-}M)^{th}$ column of A is exactly zero

SGEES/CGEES

```
      SUBROUTINE SGEES( JOBVS, SORT, SELECT, N, A, LDA, SDIM, WR, WI,
     $                  VS, LDVS, WORK, LWORK, BWORK, INFO )
      CHARACTER          JOBVS, SORT
      INTEGER            INFO, LDA, LDVS, LWORK, N, SDIM
      LOGICAL            BWORK( * )
      REAL               A( LDA, * ), VS( LDVS, * ), WI( * ), WORK( * ),
     $                   WR( * )
      LOGICAL            SELECT
      EXTERNAL           SELECT

      SUBROUTINE CGEES( JOBVS, SORT, SELECT, N, A, LDA, SDIM, W, VS,
     $                  LDVS, WORK, LWORK, RWORK, BWORK, INFO )
      CHARACTER          JOBVS, SORT
      INTEGER            INFO, LDA, LDVS, LWORK, N, SDIM
      LOGICAL            BWORK( * )
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), VS( LDVS, * ), W( * ), WORK( * )
      LOGICAL            SELECT
      EXTERNAL           SELECT
```

Purpose

SGEES/CGEES computes for an n-by-n real/complex nonsymmetric matrix A, the eigenvalues, the real-Schur/Schur form T, and, optionally, the matrix of Schur vectors Z. This gives the Schur factorization $A = Z*T*Z^H$.

Optionally, it also orders the eigenvalues on the diagonal of the real-Schur/Schur form so that selected eigenvalues are at the top left. The leading columns of Z then form an orthonormal basis for the invariant subspace corresponding to the selected eigenvalues.

A real matrix is in real-Schur form if it is upper quasi-triangular with 1-by-1 and 2-by-2 diagonal blocks. 2-by-2 diagonal blocks will be standardized in the form $\begin{pmatrix} a & b \\ c & a \end{pmatrix}$ where b*c < 0. The eigenvalues of such a block are $a \pm \sqrt{bc}$.

A complex matrix is in Schur form if it is upper triangular.

Arguments

JOBVS (input) CHARACTER*1
= 'N': Schur vectors are not computed;
= 'V': Schur vectors are computed.

SORT (input) CHARACTER*1
Specifies whether or not to order the eigenvalues on the diagonal of the Schur form.
= 'N': Eigenvalues are not ordered;
= 'S': Eigenvalues are ordered (see SELECT).

SELECT *SGEES* (input) LOGICAL FUNCTION of two REAL arguments
CGEES (input) LOGICAL FUNCTION of one COMPLEX argument
SELECT must be declared EXTERNAL in the calling subroutine.
If SORT = 'S', SELECT is used to select eigenvalues to sort to the top left of the Schur form.
If SORT = 'N', SELECT is not referenced.
SGEES
An eigenvalue WR(j) + i*WI(j) is selected if SELECT(WR(j),WI(j)) is true. A complex eigenvalue is selected if either SELECT(WR(j),WI(j)) or SELECT(WR(j),−WI(j)) is true: i.e., if either one of a complex conjugate pair of eigenvalues is selected, then both are. Note that a selected complex eigenvalue may no longer satisfy SELECT(WR(j),WI(j)) = .TRUE. after ordering, since ordering may change the value of complex eigenvalues (especially if the eigenvalue is ill-conditioned); in this case INFO may be set to N+2 (see INFO below).
CGEES
An eigenvalue W(j) is selected if SELECT(W(j)) is true.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the n-by-n matrix A.

On exit, A has been overwritten by its real-Schur/Schur form T.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

SDIM (output) INTEGER
If SORT = 'N', SDIM = 0.
If SORT = 'S', SDIM = number of eigenvalues (after sorting) for which SELECT is true.
SGEES only
(Complex conjugate pairs for which SELECT is true for either eigenvalue count as 2.)

WR, WI *SGEES only* (output) REAL/COMPLEX array, dimension (N)
WR and WI contain the real and imaginary parts, respectively, of the computed eigenvalues, in the same order that they appear on the diagonal of the output Schur form T. Complex conjugate pairs of eigenvalues appear consecutively with the eigenvalue having the positive imaginary part first.

W *CGEES only* (output) COMPLEX array, dimension (N)
W contains the computed eigenvalues, in the same order that they appear on the diagonal of the output Schur form T.

VS (output) REAL/COMPLEX array, dimension (LDVS,N)
If JOBVS = 'V', VS contains the orthogonal/unitary matrix Z of Schur vectors.
If JOBVS = 'N', VS is not referenced.

LDVS (input) INTEGER
The leading dimension of the array VS. LDVS $\geq$ 1; if JOBVS = 'V', LDVS $\geq$ N.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) contains the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
LWORK $\geq$ max(1,3*N) (*SGEES*)
LWORK $\geq$ max(1,2*N) (*CGEES*).
For good performance, LWORK must generally be larger.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CGEES only* (workspace) REAL array, dimension (N)

BWORK (workspace) LOGICAL array, dimension (N)
Not referenced if SORT = 'N'.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
$\leq$ N: the QR algorithm failed to compute all the eigenvalues; elements 1:ilo−1 and i+1:n of WR and WI (*SGEES*) or W (*CGEES*) contain those eigenvalues which have converged; if JOBVS = 'V', VS contains the matrix which reduces A to its partially converged Schur form.
= N+1: the eigenvalues could not be reordered because some eigenvalues were too close to separate (the problem is very ill-conditioned);
= N+2: after reordering, roundoff changed values of some complex eigenvalues so that leading eigenvalues in the Schur form no longer satisfy SELECT=.TRUE. This could also be caused by underflow due to scaling.

SGEESX/CGEESX

```
      SUBROUTINE SGEESX( JOBVS, SORT, SELECT, SENSE, N, A, LDA, SDIM,
     $                   WR, WI, VS, LDVS, RCONDE, RCONDV, WORK, LWORK,
     $                   IWORK, LIWORK, BWORK, INFO )
      CHARACTER          JOBVS, SENSE, SORT
      INTEGER            INFO, LDA, LDVS, LIWORK, LWORK, N, SDIM
      REAL               RCONDE, RCONDV
      LOGICAL            BWORK( * )
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), VS( LDVS, * ), WI( * ), WORK( * ),
     $                   WR( * )
      LOGICAL            SELECT
      EXTERNAL           SELECT

      SUBROUTINE CGEESX( JOBVS, SORT, SELECT, SENSE, N, A, LDA, SDIM, W,
     $                   VS, LDVS, RCONDE, RCONDV, WORK, LWORK, RWORK,
     $                   BWORK, INFO )
      CHARACTER          JOBVS, SENSE, SORT
      INTEGER            INFO, LDA, LDVS, LWORK, N, SDIM
      REAL               RCONDE, RCONDV
      LOGICAL            BWORK( * )
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), VS( LDVS, * ), W( * ), WORK( * )
      LOGICAL            SELECT
      EXTERNAL           SELECT
```

Purpose

SGEESX/CGEESX computes for an n-by-n real/complex nonsymmetric matrix A, the eigenvalues, the real-Schur/Schur form T, and, optionally, the matrix of Schur vectors Z. This gives the Schur factorization A = $Z*T*Z^H$.

Optionally, it also orders the eigenvalues on the diagonal of the real-Schur/Schur form so that selected eigenvalues are at the top left; computes a reciprocal condition number for the average of the selected eigenvalues (RCONDE); and computes a reciprocal condition number for the right invariant subspace corresponding to the selected eigenvalues (RCONDV). The leading columns of Z form an orthonormal basis for this invariant subspace.

For further explanation of the reciprocal condition numbers RCONDE and RCONDV, see Section 4.10 (where these quantities are called s and sep respectively).

A real matrix is in real-Schur form if it is upper quasi-triangular with 1-by-1 and 2-by-2 diagonal blocks. 2-by-2 diagonal blocks will be standardized in the form $\begin{pmatrix} a & b \\ c & a \end{pmatrix}$ where b*c < 0. The eigenvalues of such a block are $a \pm \sqrt{bc}$.

A complex matrix is in Schur form if it is upper triangular.

Arguments

JOBVS (input) CHARACTER*1
= 'N': Schur vectors are not computed;
= 'V': Schur vectors are computed.

SORT (input) CHARACTER*1
Specifies whether or not to order the eigenvalues on the diagonal of the Schur form.
= 'N': Eigenvalues are not ordered;
= 'S': Eigenvalues are ordered (see SELECT).

SELECT *SGEESX* (input) LOGICAL FUNCTION of two REAL arguments
CGEESX (input) LOGICAL FUNCTION of one COMPLEX argument
SELECT must be declared EXTERNAL in the calling subroutine.
If SORT = 'S', SELECT is used to select eigenvalues to sort to the top left of the Schur form.
If SORT = 'N', SELECT is not referenced.
SGEESX
An eigenvalue WR(j) + i*WI(j) is selected if SELECT(WR(j),WI(j)) is true. A complex eigenvalue is selected if either SELECT(WR(j),WI(j)) or SELECT(WR(j),−WI(j)) is true: i.e., if either one of a complex conjugate pair of eigenvalues is selected, then both are. Note that a selected complex eigenvalue may no longer satisfy SELECT(WR(j),WI(j)) = .TRUE. after ordering, since ordering may change the value of complex eigenvalues (especially if the eigenvalue is ill-conditioned); in this case INFO may be set to N+2 (see INFO below).
CGEESX
An eigenvalue W(j) is selected if SELECT(W(j)) is true.

SENSE (input) CHARACTER*1
Determines which reciprocal condition numbers are computed.
= 'N': None are computed;
= 'E': Computed for average of selected eigenvalues only;
= 'V': Computed for selected right invariant subspace only;
= 'B': Computed for both.
If SENSE = 'E', 'V' or 'B', SORT must equal 'S'.

N (input) INTEGER
The order of the matrix A. N ≥ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the n-by-n matrix A.
On exit, A is overwritten by its real-Schur/Schur form T.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,N).

SDIM (output) INTEGER
If SORT = 'N', SDIM = 0.
If SORT = 'S', SDIM = number of eigenvalues (after sorting) for which SELECT is true.
SGEESX only
(Complex conjugate pairs for which SELECT is true for either eigenvalue count as 2.)

WR, WI *SGEESX only* (output) REAL array, dimension (N)
WR and WI contain the real and imaginary parts, respectively, of the computed eigenvalues, in the same order that they appear on the diagonal of the output Schur form T. Complex conjugate pairs of eigenvalues appear consecutively with the eigenvalue having the positive imaginary part first.

W *CGEESX only* (output) COMPLEX array, dimension (N)
W contains the computed eigenvalues, in the same order that they appear on the diagonal of the output Schur form T.

VS (output) REAL/COMPLEX array, dimension (LDVS,N)
If JOBVS = 'V', VS contains the orthogonal/unitary matrix Z of Schur vectors.
If JOBVS = 'N', VS is not referenced.

LDVS (input) INTEGER
The leading dimension of the array VS. LDVS ≥ 1; if JOBVS = 'V', LDVS ≥ N.

RCONDE (output) REAL
If SENSE = 'E' or 'B', RCONDE contains the reciprocal condition number for the average of the selected eigenvalues.
Not referenced if SENSE = 'N' or 'V'.

RCONDV (output) REAL
If SENSE = 'V' or 'B', RCONDV contains the reciprocal condition number for the selected right invariant subspace.
Not referenced if SENSE = 'N' or 'E'.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.

LWORK ≥ max(1,3*N) (*SGEESX*)
LWORK ≥ max(1,2*N) (*CGEESX*).
Also, if SENSE = 'E' or 'V' or 'B',
LWORK ≥ N+2*SDIM*(N−SDIM) (*SGEESX*)
LWORK ≥ 2*SDIM*(N−SDIM) (*CGEESX*),
where SDIM is the number of selected eigenvalues computed by this routine. Note that 2*SDIM*(N−SDIM) ≤ N*N/2.
For good performance, LWORK must generally be larger.

IWORK — *SGEESX only* (workspace) INTEGER array, dimension (LIWORK)
Not referenced if SENSE = 'N' or 'E'.

LIWORK — *SGEESX only* (input) INTEGER
The dimension of the array IWORK. LIWORK ≥ 1; if SENSE = 'V' or 'B', LIWORK ≥ SDIM*(N−SDIM).

RWORK — *CGEESX only* (workspace) REAL array, dimension (N)

BWORK — (workspace) LOGICAL array, dimension (N)
Not referenced if SORT = 'N'.

INFO — (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
≤ N: the QR algorithm failed to compute all the eigenvalues; elements 1:ilo−1 and i+1:n of WR and WI (*SGEESX*) or W (*CGEESX*) contain those eigenvalues which have converged; if JOBVS = 'V', VS contains the transformation which reduces A to its partially converged Schur form.
= N+1: the eigenvalues could not be reordered because some eigenvalues were too close to separate (the problem is very ill-conditioned);
= N+2: after reordering, roundoff changed values of some complex eigenvalues so that leading eigenvalues in the Schur form no longer satisfy SELECT=.TRUE. This could also be caused by underflow due to scaling.

SGEEV/CGEEV

```
      SUBROUTINE SGEEV( JOBVL, JOBVR, N, A, LDA, WR, WI, VL, LDVL, VR,
     $                  LDVR, WORK, LWORK, INFO )
      CHARACTER          JOBVL, JOBVR
      INTEGER            INFO, LDA, LDVL, LDVR, LWORK, N
      REAL               A( LDA, * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   WI( * ), WORK( * ), WR( * )
```

```
      SUBROUTINE CGEEV( JOBVL, JOBVR, N, A, LDA, W, VL, LDVL, VR, LDVR,
     $                  WORK, LWORK, RWORK, INFO )
      CHARACTER          JOBVL, JOBVR
      INTEGER            INFO, LDA, LDVL, LDVR, LWORK, N
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   W( * ), WORK( * )
```

Purpose

SGEEV/CGEEV computes for an n-by-n real/complex nonsymmetric matrix A, the eigenvalues and, optionally, the left and/or right eigenvectors.

The right eigenvector v(j) of A satisfies

$$A * v(j) \quad = \quad \lambda(j) * v(j)$$

where λ(j) is its eigenvalue.
The left eigenvector u(j) of A satisfies

$$u(j)^H * A \quad = \quad \lambda(j) * u(j)^H$$

where $u(j)^H$ denotes the conjugate transpose of u(j).

The computed eigenvectors are normalized to have Euclidean norm equal to 1 and largest component real.

Arguments

JOBVL — (input) CHARACTER*1
= 'N': left eigenvectors of A are not computed;
= 'V': left eigenvectors of A are computed.

JOBVR — (input) CHARACTER*1
= 'N': right eigenvectors of A are not computed;
= 'V': right eigenvectors of A are computed.

N — (input) INTEGER
The order of the matrix A. N ≥ 0.

A — (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the n-by-n matrix A.
On exit, A has been overwritten.

LDA — (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,N).

WR, WI — *SGEEV only* (output) REAL array, dimension (N)
WR and WI contain the real and imaginary parts, respectively, of the computed eigenvalues. Complex conjugate pairs of eigenvalues appear consecutively with the eigenvalue having the positive imaginary part first.

W — *CGEEV only* (output) COMPLEX array, dimension (N)
W contains the computed eigenvalues.

VL (output) REAL/COMPLEX array, dimension (LDVL,N)
If JOBVL = 'V', the left eigenvectors u(j) are stored one after another in the columns of VL, in the same order as their eigenvalues.
If JOBVL = 'N', VL is not referenced.
SGEEV
If the j^{th} eigenvalue is real, then u(j) = VL(:,j), the j^{th} column of VL.
If the j^{th} and $(j+1)^{th}$ eigenvalues form a complex conjugate pair, then u(j) = VL(:,j) + i*VL(:,j+1) and u(j+1) = VL(:,j) − i*VL(:,j+1).
CGEEV
u(j) = VL(:,j), the j^{th} column of VL.

LDVL (input) INTEGER
The leading dimension of the array VL. LDVL $\geq$ 1; if JOBVL = 'V', LDVL $\geq$ N.

VR (output) REAL/COMPLEX array, dimension (LDVR,N)
If JOBVR = 'V', the right eigenvectors v(j) are stored one after another in the columns of VR, in the same order as their eigenvalues.
If JOBVR = 'N', VR is not referenced.
SGEEV
If the j^{th} eigenvalue is real, then v(j) = VR(:,j), the j^{th} column of VR.
If the j^{th} and $(j+1)^{th}$ eigenvalues form a complex conjugate pair, then v(j) = VR(:,j) + i*VR(:,j+1) and v(j+1) = VR(:,j) − i*VR(:,j+1).
CGEEV
v(j) = VR(:,j), the j^{th} column of VR.

LDVR (input) INTEGER
The leading dimension of the matrix VR. LDVR $\geq$ 1; if JOBVR = 'V', LDVR $\geq$ N.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
LWORK $\geq$ max(1,3*N), and if JOBVL = 'V' or JOBVR = 'V', LWORK $\geq$ max(1,4*N) (*SGEEV*)
LWORK $\geq$ max(1,2*N) (*CGEEV*).
For good performance, LWORK must generally be larger.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CGEEV only* (workspace) REAL array, dimension (2*N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the QR algorithm failed to compute all the eigenvalues, and no eigenvectors have been computed; elements i+1:n of WR and WI (*SGEEV*) or W (*CGEEV*) contain eigenvalues which have converged.

SGEEVX/CGEEVX

```
      SUBROUTINE SGEEVX( BALANC, JOBVL, JOBVR, SENSE, N, A, LDA, WR, WI,
     $                   VL, LDVL, VR, LDVR, ILO, IHI, SCALE, ABNRM,
     $                   RCONDE, RCONDV, WORK, LWORK, IWORK, INFO )
      CHARACTER          BALANC, JOBVL, JOBVR, SENSE
      INTEGER            IHI, ILO, INFO, LDA, LDVL, LDVR, LWORK, N
      REAL               ABNRM
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), RCONDE( * ), RCONDV( * ),
     $                   SCALE( * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   WI( * ), WORK( * ), WR( * )

      SUBROUTINE CGEEVX( BALANC, JOBVL, JOBVR, SENSE, N, A, LDA, W, VL,
     $                   LDVL, VR, LDVR, ILO, IHI, SCALE, ABNRM, RCONDE,
     $                   RCONDV, WORK, LWORK, RWORK, INFO )
      CHARACTER          BALANC, JOBVL, JOBVR, SENSE
      INTEGER            IHI, ILO, INFO, LDA, LDVL, LDVR, LWORK, N
      REAL               ABNRM
      REAL               RCONDE( * ), RCONDV( * ), RWORK( * ),
     $                   SCALE( * )
      COMPLEX            A( LDA, * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   W( * ), WORK( * )
```

Purpose

SGEEVX/CGEEVX computes for an n-by-n real/complex nonsymmetric matrix A, the eigenvalues and, optionally, the left and/or right eigenvectors.

Optionally also, it computes a balancing transformation to improve the conditioning of the eigenvalues and eigenvectors (ILO, IHI, SCALE, and ABNRM), reciprocal condition numbers for the eigenvalues (RCONDE), and reciprocal condition numbers for the right eigenvectors (RCONDV).

The right eigenvector v(j) of A satisfies

$$A * v(j) \quad = \quad \lambda(j) * v(j)$$

where λ(j) is its eigenvalue.
The left eigenvector u(j) of A satisfies

$$u(j)^H * A \quad = \quad \lambda(j) * u(j)^H$$

where $u(j)^H$ denotes the conjugate transpose of u(j).

The computed eigenvectors are normalized to have Euclidean norm equal to 1 and largest component real.

Balancing a matrix means permuting the rows and columns to make it more nearly upper triangular, and applying a diagonal similarity transformation $D*A*D^{-1}$, where D is a diagonal matrix, to make its rows and columns closer in norm and the condition numbers of its eigenvalues and eigenvectors smaller. The computed reciprocal condition numbers correspond to the balanced matrix.

Permuting rows and columns will not change the condition numbers (in exact arithmetic) but diagonal scaling will. For further explanation of balancing, see section 4.8.1.2.

For further explanation of the reciprocal condition numbers RCONDE and RCONDV, see Section 4.8.1.3 (where these quantities are called s and sep respectively).

Arguments

BALANC (input) CHARACTER*1
Indicates how the input matrix should be diagonally scaled and/or permuted to improve the conditioning of its eigenvalues.
= 'N': Do not diagonally scale or permute;
= 'P': Perform permutations to make the matrix more nearly upper triangular. Do not diagonally scale;
= 'S': Diagonally scale the matrix, i.e/ie. replace A by $D*A*D^{-1}$, where D is a diagonal matrix chosen to make the rows and columns of A more equal in norm. Do not permute;
= 'B': Both diagonally scale and permute A.
Computed reciprocal condition numbers will be for the matrix after balancing and/or permuting. Permuting does not change condition numbers (in exact arithmetic), but balancing does.

JOBVL (input) CHARACTER*1
= 'N': left eigenvectors of A are not computed;
= 'V': left eigenvectors of A are computed.
If SENSE = 'E' or 'B', JOBVL must = 'V'.

JOBVR (input) CHARACTER*1
= 'N': right eigenvectors of A are not computed;
= 'V': right eigenvectors of A are computed.
If SENSE = 'E' or 'B', JOBVR must = 'V'.

SENSE (input) CHARACTER*1
Determines which reciprocal condition numbers are computed.
= 'N': None are computed;
= 'E': Computed for eigenvalues only;
= 'V': Computed for right eigenvectors only;
= 'B': Computed for eigenvalues and right eigenvectors.
If SENSE = 'E' or 'B', both left and right eigenvectors must also be computed (JOBVL = 'V' and JOBVR = 'V').

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the n-by-n matrix A.
On exit, A has been overwritten. If JOBVL = 'V' or JOBVR = 'V', A contains the real-Schur/Schur form of the balanced version of the input matrix A.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

WR, WI *SGEEVX only* (output) REAL array, dimension (N)
WR and WI contain the real and imaginary parts, respectively, of the computed eigenvalues. Complex conjugate pairs of eigenvalues appear consecutively with the eigenvalue having the positive imaginary part first.

W *CGEEVX only* (output) COMPLEX array, dimension (N)
W contains the computed eigenvalues.

VL (output) REAL/COMPLEX array, dimension (LDVL,N)
If JOBVL = 'V', the left eigenvectors u(j) are stored one after another in the columns of VL, in the same order as their eigenvalues.
If JOBVL = 'N', VL is not referenced.
SGEEV
If the j^{th} eigenvalue is real, then u(j) = VL(:,j), the j^{th} column of VL.
If the j^{th} and $(j+1)^{th}$ eigenvalues form a complex conjugate pair, then u(j) = VL(:,j) + i*VL(:,j+1) and u(j+1) = VL(:,j) − i*VL(:,j+1).
CGEEV
u(j) = VL(:,j), the j^{th} column of VL.

LDVL (input) INTEGER
The leading dimension of the array VL. LDVL $\geq$ 1; if JOBVL = 'V', LDVL $\geq$ N.

VR (output) REAL/COMPLEX array, dimension (LDVR,N)
If JOBVR = 'V', the right eigenvectors v(j) are stored one after another in the columns of VR, in the same order as their eigenvalues.
If JOBVR = 'N', VR is not referenced.
SGEEV
If the j^{th} eigenvalue is real, then v(j) = VR(:,j), the j^{th} column of VR.
If the j^{th} and $(j+1)^{th}$ eigenvalues form a complex conjugate pair, then v(j) = VR(:,j) + i*VR(:,j+1) and v(j+1) = VR(:,j) − i*VR(:,j+1).
CGEEV
v(j) = VR(:,j), the j^{th} column of VR.

LDVR (input) INTEGER
The leading dimension of the matrix VR. LDVR $\geq$ 1; if JOBVR = 'V', LDVR $\geq$ N.

ILO, IHI (output) INTEGER
ILO and IHI are integer values determined when A was balanced. The

balanced A(i,j) = 0 if i > j and j = 1,...,ilo−1 or i = ihi+1,...,n.
If BALANC = 'N' or 'S', ILO = 1 and IHI = N.

SCALE (output) REAL array, dimension (N)
Details of the permutations and scaling factors applied when balancing A. If P(j) is the index of the row and column interchanged with row and column j, and D(j) is the scaling factor applied to row and column j, then

SCALE(j) = P(j)	for j = 1,...,ilo−1
= D(j)	for j = ilo,...,ihi
= P(j)	for j = ihi+1,...,n.

The order in which the interchanges are made is n to ihi+1, then 1 to ilo−1.

ABNRM (output) REAL
The one-norm of the balanced matrix (the maximum of the sum of absolute values of elements of any column).

RCONDE (output) REAL array, dimension (N)
RCONDE(j) is the reciprocal condition number of the j^{th} eigenvalue.

RCONDV (output) REAL array, dimension (N)
RCONDV(j) is the reciprocal condition number of the j^{th} right eigenvector.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
SGEEVX
If SENSE = 'N' or 'E', LWORK ≥ max(1,2*N), and
if JOBVL = 'V' or JOBVR = 'V', LWORK ≥ 3*N.
If SENSE = 'V' or 'B', LWORK ≥ N*(N+6).
For good performance, LWORK must generally be larger.
CGEEVX
If SENSE = 'N' or 'E', LWORK ≥ max(1,2*N), and
if SENSE = 'V' or 'B', LWORK ≥ N*N+2*N.
For good performance, LWORK must generally be larger.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CGEEVX only* (workspace) REAL array, dimension (2*N)

IWORK *SGEEVX only* (workspace) INTEGER array, dimension (2*N−2)
If SENSE = 'N' or 'E', not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the QR algorithm failed to compute all the eigenvalues, and no eigenvectors or condition numbers have been computed; elements 1:ilo−1 and i+1:n of WR and WI (*SGEEVX*) or W (*CGEEVX*) contain eigenvalues which have converged.

SGEHRD/CGEHRD

```
SUBROUTINE SGEHRD( N, ILO, IHI, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            IHI, ILO, INFO, LDA, LWORK, N
REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

SUBROUTINE CGEHRD( N, ILO, IHI, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            IHI, ILO, INFO, LDA, LWORK, N
COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SGEHRD/CGEHRD reduces a real/complex general matrix A to upper Hessenberg form H by an orthogonal/unitary similarity transformation: $Q^H*A*Q = H$.

Arguments

N (input) INTEGER
The order of the matrix A. N ≥ 0.

ILO, IHI (input) INTEGER
It is assumed that A is already upper triangular in rows and columns 1:ilo−1 and ihi+1:n. ILO and IHI are normally set by a previous call to SGEBAL/CGEBAL; otherwise they should be set to 1 and N respectively.
1 ≤ ILO ≤ IHI ≤ N, if N > 0; ILO = 1 and IHI = 0, if N = 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the n-by-n general matrix to be reduced.
On exit, the upper triangle and the first subdiagonal of A are overwritten with the upper Hessenberg matrix H, and the elements below the first subdiagonal, with the array TAU, represent the orthogonal/unitary matrix Q as a product of elementary reflectors.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,N).

TAU (output) REAL/COMPLEX array, dimension (N−1)
The scalar factors of the elementary reflectors. Elements 1:ilo−1 and ihi:n−1 of TAU are set to zero.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The length of the array WORK. LWORK ≥ max(1,N). For optimum

performance LWORK $\geq$ N*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGELQF/CGELQF

```
      SUBROUTINE SGELQF( M, N, A, LDA, TAU, WORK, LWORK, INFO )
      INTEGER            INFO, LDA, LWORK, M, N
      REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

      SUBROUTINE CGELQF( M, N, A, LDA, TAU, WORK, LWORK, INFO )
      INTEGER            INFO, LDA, LWORK, M, N
      COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SGELQF/CGELQF computes an LQ factorization of a real/complex m-by-n matrix A: A = L*Q.

Arguments

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit, the elements on and below the diagonal of the array contain the m-by-min(m,n) lower trapezoidal matrix L (L is lower triangular if m $\leq$ n); the elements above the diagonal, with the array TAU, represent the orthogonal/unitary matrix Q as a product of elementary reflectors.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

TAU (output) REAL/COMPLEX array, dimension (min(M,N))
The scalar factors of the elementary reflectors.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ max(1,M). For optimum performance LWORK $\geq$ M*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGELS/CGELS

```
      SUBROUTINE SGELS( TRANS, M, N, NRHS, A, LDA, B, LDB, WORK, LWORK,
     $                  INFO )
      CHARACTER          TRANS
      INTEGER            INFO, LDA, LDB, LWORK, M, N, NRHS
      REAL               A( LDA, * ), B( LDB, * ), WORK( LWORK )

      SUBROUTINE CGELS( TRANS, M, N, NRHS, A, LDA, B, LDB, WORK, LWORK,
     $                  INFO )
      CHARACTER          TRANS
      INTEGER            INFO, LDA, LDB, LWORK, M, N, NRHS
      COMPLEX            A( LDA, * ), B( LDB, * ), WORK( LWORK )
```

Purpose

SGELS/CGELS solves overdetermined or underdetermined real/complex linear systems involving an m-by-n matrix A, or its transpose/conjugate-transpose, using a QR or LQ factorization of A. It is assumed that A has full rank.

The following options are provided:

1. If TRANS = 'N' and m $\geq$ n: find the least squares solution of an overdetermined system, i.e., solve the least squares problem

$$\text{minimize } \| b - A*x \|_2.$$

2. If TRANS = 'N' and m < n: find the minimum norm solution of an underdetermined system A*x = B.

3. If TRANS = 'T'/'C' and m $\geq$ n: find the minimum norm solution of an undetermined system A^H*X = B.

4. If TRANS = 'T'/'C' and m < n: find the least squares solution of an overdetermined system, i.e., solve the least squares problem

$$\text{minimize } \| b - A^H*x \|_2.$$

Several right hand side vectors b and solution vectors x can be handled in a single call; they are stored as the columns of the m-by-nrhs right hand side matrix B and the n-by-nrhs solution matrix X.

Arguments

TRANS (input) CHARACTER
= 'N': the linear system involves A;
= 'T': the linear system involves A^T (*SGELS*);
= 'C': the linear system involves A^H (*CGELS*).

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit,
if m $\geq$ n, A is overwritten by details of its QR factorization as returned by SGEQRF/CGEQRF;
if m < n, A is overwritten by details of its LQ factorization as returned by SGELQF/CGELQF.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the matrix B of right hand side vectors, stored columnwise; B is m-by-nrhs if TRANS = 'N', or n-by-nrhs if TRANS = 'T' or 'C'.
On exit, B is overwritten by the solution vectors, stored columnwise:
if TRANS = 'N' and m $\geq$ n, rows 1 to n of B contain the least squares solution vectors; the residual sum of squares for the solution in each column is given by the sum of squares of elements n+1 to m in that column;
if TRANS = 'N' and m < n, rows 1 to n of B contain the minimum norm solution vectors;
if TRANS = 'T' or 'C' and m $\geq$ n, rows 1 to m of B contain the minimum norm solution vectors;
if TRANS = 'T' or 'C' and m < n, rows 1 to m of B contain the least squares solution vectors; the residual sum of squares for the solution in each column is given by the sum of squares of elements m+1 to n in that column.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,M,N).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ min(M,N) + max(1,M,N,NRHS). For optimal performance, LWORK $\geq$ min(M,N) + max(1,M,N,NRHS)*NB where NB is the optimum block size.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGELSD/CGELSD

```
      SUBROUTINE SGELSD( M, N, NRHS, A, LDA, B, LDB, S, RCOND, RANK,
     $                   WORK, LWORK, IWORK, INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, NRHS, RANK
      REAL               RCOND
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), B( LDB, * ), S( * ), WORK( * )

      SUBROUTINE CGELSD( M, N, NRHS, A, LDA, B, LDB, S, RCOND, RANK,
     $                   WORK, LWORK, RWORK, IWORK, INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, NRHS, RANK
      REAL               RCOND
      INTEGER            IWORK( * )
      REAL               RWORK( * ), S( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), WORK( * )
```

Purpose

SGELSD/CGELSD computes the minimum norm solution to a real/complex linear least squares problem:

minimize $\| b - A*x \|_2$.

using the singular value decomposition (SVD) of A. A is an m-by-n matrix which may be rank-deficient.

Several right hand side vectors b and solution vectors x can be handled in a single call; they are stored as the columns of the m-by-nrhs right hand side matrix B and the n-by-nrhs solution matrix X.

The problem is solved in three steps:

1. Reduce the coefficient matrix A to bidiagonal form with Householder transformations, reducing the original problem into a "bidiagonal least squares

problem" (BLS).

2. Solve the BLS using a divide and conquer approach.

3. Apply back all the Householder tranformations to solve the original least squares problem.

The effective rank of A is determined by treating as zero those singular values which are less than RCOND times the largest singular value.

The divide and conquer algorithm makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

Arguments

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit, the first min(m,n) rows of A are overwritten with its right singular vectors, stored rowwise.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the m-by-nrhs right hand side matrix B.
On exit, B is overwritten by the n-by-nrhs solution matrix X.
If m $\geq$ n and RANK = n, the residual sum-of-squares for the solution in the i^{th} column is given by the sum of squares of elements n+1:m in that column.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,M,N).

S (output) REAL array, dimension (min(M,N))
The singular values of A in decreasing order.
The condition number of A in the 2-norm = $S(1)/S(min(m,n))$.

RCOND (input) REAL
RCOND is used to determine the effective rank of A. Singular values $S(i) \leq RCOND * S(1)$ are treated as zero. If RCOND < 0, machine precision is used instead.

RANK (output) INTEGER
The effective rank of A, i.e., the number of singular values which are greater than $RCOND * S(1)$.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ 1. If M $\geq$ N, LWORK $\geq 11*N+2*N*SMLSIZ+8*N*NLVL+N*NRHS$. If M < N, LWORK $\geq 11*M+2*M*SMLSIZ+8*M*NLVL+M*NRHS$. SMLSIZ is returned by ILAENV and is equal to the maximum size of the subproblems at the bottom of the computation tree (usually about 25), and $NLVL = INT(LOG_2(MIN(M,N)/(SMLSIZ+1)))+1$ For good performance, LWORK should generally be larger.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

IWORK (workspace) INTEGER array, dimension (LIWORK)
LIWORK $\geq 3*MINMN*NLVL+11*MINMN$, where MINMN = MIN(M,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: the algorithm for computing the SVD failed to converge; if INFO = i, i off-diagonal elements of an intermediate bidiagonal form did not converge to zero.

SGELSS/CGELSS

```
      SUBROUTINE SGELSS( M, N, NRHS, A, LDA, B, LDB, S, RCOND, RANK,
     $                   WORK, LWORK, INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, NRHS, RANK
      REAL               RCOND
      REAL               A( LDA, * ), B( LDB, * ), S( * ), WORK( * )

      SUBROUTINE CGELSS( M, N, NRHS, A, LDA, B, LDB, S, RCOND, RANK,
     $                   WORK, LWORK, RWORK, INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, NRHS, RANK
      REAL               RCOND
      REAL               RWORK( * ), S( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), WORK( * )
```

Purpose

SGELSS/CGELSS computes the minimum norm solution to a real/complex linear least squares problem:

minimize $\| b - A*x \|_2$.

using the singular value decomposition (SVD) of A. A is an m-by-n matrix which may be rank-deficient.

Several right hand side vectors b and solution vectors x can be handled in a single call; they are stored as the columns of the m-by-nrhs right hand side matrix B and the n-by-nrhs solution matrix X.

The effective rank of A is determined by treating as zero those singular values which are less than RCOND times the largest singular value.

Arguments

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit, the first min(m,n) rows of A are overwritten with its right singular vectors, stored rowwise.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the m-by-nrhs right hand side matrix B.
On exit, B is overwritten by the n-by-nrhs solution matrix X.
If m $\geq$ n and RANK = n, the residual sum-of-squares for the solution in the i^{th} column is given by the sum of squares of elements n+1:m in that column.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,M,N).

S (output) REAL array, dimension (min(M,N))
The singular values of A in decreasing order. The condition number of A in the 2-norm is $\kappa_2(A)$ = S(1)/S(min(m,n)).

RCOND (input) REAL
RCOND is used to determine the effective rank of A.
Singular values S(i) $\leq$ RCOND*S(1) are treated as zero.
If RCOND < 0, machine precision is used instead.

RANK (output) INTEGER
The effective rank of A, i.e., the number of singular values which are greater than RCOND*S(1).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ 1, and also:
SGELSS
LWORK $\geq$ 3*min(M,N)+max(2*min(M,N), max(M,N), NRHS)
CGELSS
LWORK $\geq$ 2*min(M,N)+max(M,N,NRHS)
For good performance, LWORK should generally be larger.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CGELSS only* (workspace) REAL array,
dimension (max(1,5*min(M,N)))

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: the algorithm for computing the SVD failed to converge; if INFO = i, i off-diagonal elements of an intermediate bidiagonal form did not converge to zero.

SGELSY/CGELSY

```
      SUBROUTINE SGELSY( M, N, NRHS, A, LDA, B, LDB, JPVT, RCOND, RANK,
     $                   WORK, LWORK, INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, NRHS, RANK
      REAL               RCOND
      INTEGER            JPVT( * )
      REAL               A( LDA, * ), B( LDB, * ), WORK( LWORK )

      SUBROUTINE CGELSY( M, N, NRHS, A, LDA, B, LDB, JPVT, RCOND, RANK,
     $                   WORK, LWORK, RWORK, INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, NRHS, RANK
      REAL               RCOND
      INTEGER            JPVT( * )
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), WORK( * )
```

Purpose

SGELSY/CGELSY computes the minimum-norm solution to a real/complex linear least squares problem:

minimize $\| b - A*x \|_2$

using a complete orthogonal factorization of A. A is an m-by-n matrix which may be rank-deficient.

Several right hand side vectors b and solution vectors x can be handled in a single call; they are stored as the columns of the m-by-nrhs right hand side matrix B and the n-by-nrhs solution matrix X.

The routine first computes a QR factorization with column pivoting:

$$A * P \;=\; Q * \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}$$

with R_{11} defined as the largest leading submatrix whose estimated condition number is less than 1/RCOND. The order of R_{11}, RANK, is the effective rank of A.

Then, R_{22} is considered to be negligible, and R_{12} is annihilated by orthogonal/unitary transformations from the right, arriving at the complete orthogonal factorization:

$$A * P \;=\; Q * \begin{pmatrix} T_{11} & 0 \\ 0 & 0 \end{pmatrix} * Z$$

The minimum norm solution is then

$$x \;=\; P * Z^H * \begin{pmatrix} T_{11}^{-1} * Q_1^H * b \\ 0 \end{pmatrix}$$

where Q_1 consists of the first RANK columns of Q.

This routine is basically identical to the original xGELSX except three differences:

- The call to the subroutine xGEQPF has been substituted by the the call to the subroutine xGEQP3. This subroutine is a Blas-3 version of the QR factorization with column pivoting.
- Matrix B (the right hand side) is updated with Blas-3.
- The permutation of matrix B (the right hand side) is faster and more simple.

Arguments

M — (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N — (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0.

NRHS — (input) INTEGER
The number of right hand sides, i.e., the number of columns of matrices B and X. NRHS $\geq$ 0.

A — (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit, A has been overwritten by details of its complete orthogonal factorization.

LDA — (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

B — (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the m-by-nrhs right hand side matrix B.
On exit, the n-by-nrhs solution matrix X.

LDB — (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,M,N).

JPVT — (input/output) INTEGER array, dimension (N)
On entry, if JPVT(i) $\neq$ 0, the i^{th} column of A is permuted to the front of AP, otherwise column i is a free column.
On exit, if JPVT(i) = k, then the i^{th} column of AP was the k^{th} column of A.

RCOND — (input) REAL
RCOND is used to determine the effective rank of A, which is defined as the order of the largest leading triangular submatrix R_{11} in the QR factorization with pivoting of A, whose estimated condition number < 1/RCOND.

RANK — (output) INTEGER
The effective rank of A, i.e., the order of the submatrix R_{11}. This is the same as the order of the submatrix T_{11} in the complete orthogonal factorization of A.

WORK — (workspace) REAL/COMPLEX array, dimension (LWORK)

LWORK — INTEGER
The dimension of the array WORK.
SGELSY
The unblocked strategy requires that:
LWORK $\geq$ MAX(MN+3*N+1, 2*MN+NRHS),
where MN = min(M,N). The block algorithm requires that:
LWORK $\geq$ MAX(MN+2*N+NB*(N+1), 2*MN+NB*NRHS), where NB is an upper bound on the blocksize returned by ILAENV for the routines SGEQP3, STZRZF, STZRQF, SORMQR, and SORMRZ.
CGELSY
The unblocked strategy requires that:
LWORK $\geq$ MN + MAX(2*MN, N+1, MN+NRHS)
where MN = min(M,N). The block algorithm requires that:

LWORK $\geq$ MN + MAX(2*MN, NB*(N+1), MN+MN*NB, MN+NB*NRHS)
where NB is an upper bound on the blocksize returned by ILAENV for the routines CGEQP3, CTZRZF, CTZRQF, CUNMQR, and CUNMRZ.

RWORK *CGELSX only* (workspace) REAL array, dimension (2*N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGEQLF/CGEQLF

```
SUBROUTINE SGEQLF( M, N, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, LDA, LWORK, M, N
REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

SUBROUTINE CGEQLF( M, N, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, LDA, LWORK, M, N
COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SGEQLF/CGEQLF computes a QL factorization of a real/complex m-by-n matrix A: A = Q*L.

Arguments

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit,
if m $\geq$ n, the lower triangle of the subarray A(m-n+1:m,1:n) contains the n-by-n lower triangular matrix L;
if m $\leq$ n, the elements on and below the $(n\text{-}m)^{th}$ superdiagonal contain the m-by-n lower trapezoidal matrix L;
the remaining elements, with the array TAU, represent the orthogonal/unitary matrix Q as a product of elementary reflectors.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

TAU (output) REAL/COMPLEX array, dimension (min(M,N))
The scalar factors of the elementary reflectors.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ max(1,N). For optimum performance LWORK $\geq$ N*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGEQP3/CGEQP3

```
SUBROUTINE SGEQP3( M, N, A, LDA, JPVT, TAU, WORK, LWORK, INFO )
INTEGER            INFO, LDA, LWORK, M, N
INTEGER            JPVT( * )
REAL               A( LDA, * ), TAU( * ), WORK( * )

SUBROUTINE CGEQP3( M, N, A, LDA, JPVT, TAU, WORK, LWORK, RWORK,
$                  INFO )
INTEGER            INFO, LDA, LWORK, M, N
INTEGER            JPVT( * )
REAL               RWORK( * )
COMPLEX            A( LDA, * ), TAU( * ), WORK( * )
```

Purpose

SGEQP3/CGEQP3 computes a QR factorization with column pivoting of a real/complex m-by-n matrix A: A*P = Q*R using Level 3 BLAS.

Arguments

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit, the upper triangle of the array contains the min(m,n)-by-n upper trapezoidal matrix R; the elements below the diagonal, together with the array TAU, represent the orthogonal/unitary matrix Q as a product of min(m,n) elementary reflectors.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

JPVT (input/output) INTEGER array, dimension (N)
On entry, if $JPVT(i) \neq 0$, the i^{th} column of A is permuted to the front of A*P (a leading column); if JPVT(i) = 0, the i^{th} column of A is a free column.
On exit, if JPVT(i) = k, then the i^{th} column of A*P was the k^{th} column of A.

TAU (output) REAL/COMPLEX array, dimension (min(M,N))
The scalar factors of the elementary reflectors.

WORK *SGEQP3* (workspace) REAL array, dimension (3*N+1)
CGEQP3 (workspace) COMPLEX array, dimension (N+1)

RWORK *CGEQP3 only* (workspace) REAL array, dimension (2*N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGEQRF/CGEQRF

```
      SUBROUTINE SGEQRF( M, N, A, LDA, TAU, WORK, LWORK, INFO )
      INTEGER            INFO, LDA, LWORK, M, N
      REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

      SUBROUTINE CGEQRF( M, N, A, LDA, TAU, WORK, LWORK, INFO )
      INTEGER            INFO, LDA, LWORK, M, N
      COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SGEQRF/CGEQRF computes a QR factorization of a real/complex m-by-n matrix A: A = Q*R.

Arguments

M (input) INTEGER
The number of rows of the matrix A. $M \geq 0$.

N (input) INTEGER
The number of columns of the matrix A. $N \geq 0$.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit, the elements on and above the diagonal of the array contain the min(m,n)-by-n upper trapezoidal matrix R (R is upper triangular if $m \geq n$); the elements below the diagonal, with the array TAU, represent the orthogonal/unitary matrix Q as a product of min(m,n) elementary reflectors.

LDA (input) INTEGER
The leading dimension of the array A. $LDA \geq max(1,M)$.

TAU (output) REAL/COMPLEX array, dimension (min(M,N))
The scalar factors of the elementary reflectors.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. $LWORK \geq max(1,N)$. For optimum performance $LWORK \geq N*NB$, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGERFS/CGERFS

```
      SUBROUTINE SGERFS( TRANS, N, NRHS, A, LDA, AF, LDAF, IPIV, B, LDB,
     $                   X, LDX, FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          TRANS
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      INTEGER            IPIV( * ), IWORK( * )
      REAL               A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   BERR( * ), FERR( * ), WORK( * ), X( LDX, * )

      SUBROUTINE CGERFS( TRANS, N, NRHS, A, LDA, AF, LDAF, IPIV, B, LDB,
     $                   X, LDX, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          TRANS
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SGERFS/CGERFS improves the computed solution to a system of linear equations and provides error bounds and backward error estimates for the solution.

Arguments

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)

= 'C': $A^H*X = B$ (Conjugate transpose)

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The original n-by-n matrix A.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

AF (input) REAL/COMPLEX array, dimension (LDAF,N)
The factors L and U from the factorization A = P*L*U as computed by SGETRF/CGETRF.

LDAF (input) INTEGER
The leading dimension of the array AF. LDAF $\geq$ max(1,N).

IPIV (input) INTEGER array, dimension (N)
The pivot indices from SGETRF/CGETRF; for $1 \leq i \leq N$, row i of the matrix was interchanged with row IPIV(i).

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (input/output) REAL/COMPLEX array, dimension (LDX,NRHS)
On entry, the solution matrix X, as computed by SGETRS/CGETRS.
On exit, the improved solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) – XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SGERFS* (workspace) REAL array, dimension (3*N)
CGERFS (workspace) COMPLEX array, dimension (2*N)

IWORK *SGERFS only* (workspace) INTEGER array, dimension (N)

RWORK *CGERFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGERQF/CGERQF

```
      SUBROUTINE SGERQF( M, N, A, LDA, TAU, WORK, LWORK, INFO )
      INTEGER            INFO, LDA, LWORK, M, N
      REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

      SUBROUTINE CGERQF( M, N, A, LDA, TAU, WORK, LWORK, INFO )
      INTEGER            INFO, LDA, LWORK, M, N
      COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SGERQF/CGERQF computes an RQ factorization of a real/complex m-by-n matrix A: A = R*Q.

Arguments

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit,
if m $\leq$ n, the upper triangle of the subarray A(1:m,n-m+1:n) contains the m-by-m upper triangular matrix R;
if m $\geq$ n, the elements on and above the $(m-n)^{th}$ subdiagonal contain the m-by-n upper trapezoidal matrix R;
the remaining elements, with the array TAU, represent the orthogonal/unitary matrix Q as a product of min(m,n) elementary reflectors.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

TAU (output) REAL/COMPLEX array, dimension (min(M,N))
The scalar factors of the elementary reflectors.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ max(1,M). For optimum performance LWORK $\geq$ M*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGESDD/CGESDD

```
      SUBROUTINE SGESDD( JOBZ, M, N, A, LDA, S, U, LDU, VT, LDVT, WORK,
     $                   LWORK, IWORK, INFO )
      CHARACTER          JOBZ
      INTEGER            INFO, LDA, LDU, LDVT, LWORK, M, N
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), S( * ), U( LDU, * ),
     $                   VT( LDVT, * ), WORK( * )

      SUBROUTINE CGESDD( JOBZ, M, N, A, LDA, S, U, LDU, VT, LDVT, WORK,
     $                   LWORK, RWORK, IWORK, INFO )
      CHARACTER          JOBZ
      INTEGER            INFO, LDA, LDU, LDVT, LWORK, M, N
      INTEGER            IWORK( * )
      REAL               RWORK( * ), S( * )
      COMPLEX            A( LDA, * ), U( LDU, * ), VT( LDVT, * ),
     $                   WORK( * )
```

Purpose

SGESDD/CGESDD computes the singular value decomposition (SVD) of a real/complex m-by-n matrix A, optionally computing the left and right singular vectors. If singular vectors are desired, it uses a divide-and-conquer algorithm.

The SVD is written

$$A = U*\Sigma*V^H$$

where Σ is an m-by-n matrix which is zero except for its min(m,n) diagonal elements, U is an m-by-m orthogonal/unitary matrix, and V is an n-by-n orthogonal/unitary matrix. The diagonal elements of Σ are the singular values of A; they are real and non-negative, and are returned in descending order. The first min(m,n) columns of U and V are the left and right singular vectors of A.

Note that the routine returns V^H, not V.

The divide and conquer algorithm makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

Arguments

JOBZ (input) CHARACTER*1
Specifies options for computing all or part of the matrix U:
= 'A': all M columns of U and all N rows of V^T are returned in the arrays U and VT;
= 'S': the first min(M,N) columns of U and the first min(M,N) rows of V^T are returned in the arrays U and VT;
= 'O': If $M \geq N$, the first N columns of U are overwritten on the array A and all rows of V^T are returned in the array VT; otherwise, all columns of U are returned in the array U and the first M rows of V^T are overwritten in the array VT;
= 'N': no columns of U or rows of V^T are computed.

M (input) INTEGER
The number of rows of the matrix A. $M \geq 0$.

N (input) INTEGER
The number of columns of the matrix A. $N \geq 0$.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit,
if JOBZ = 'O', A is overwritten with the first N columns of U (the left singular vectors, stored columnwise) if $M \geq N$;
A is overwritten with the first M rows of V^T (the right singular vectors, stored rowwise) otherwise.
if JOBZ $\neq$ 'O', the contents of A are destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

S (output) REAL array, dimension (min(M,N))
The singular values of A, sorted so that $S(i) \geq S(i+1)$.

U (output) REAL/COMPLEX array, dimension (LDU,UCOL)
UCOL = M if JOBZ = 'A' or JOBZ = 'O' and $M < N$;
UCOL = min(M,N) if JOBZ = 'S'.
If JOBZ = 'A' or JOBZ = 'O' and $M < N$, U contains the M-by-M orthogonal/unitary matrix U;
if JOBZ = 'S', U contains the first min(M,N) columns of U (the left singular vectors, stored columnwise);
if JOBZ = 'O' and $M \geq N$, or JOBZ = 'N', U is not referenced.

LDU (input) INTEGER
The leading dimension of the array U. LDU $\geq$ 1; if JOBZ = 'S' or 'A' or JOBZ = 'O' and $M < N$, LDU $\geq$ M.

VT (output) REAL/COMPLEX array, dimension (LDVT,N)
If JOBZ = 'A' or JOBZ = 'O' and $M \geq N$, VT contains the N-by-N

unitary matrix V^T;
if JOBZ = 'S', VT contains the first min(M,N) rows of V^T (the right singular vectors, stored rowwise);
if JOBZ = 'O' and M < N, or JOBZ = 'N', VT is not referenced.

LDVT (input) INTEGER
The leading dimension of the array VT. LDVT $\geq$ 1; if JOBZ = 'A' or JOBZ = 'O' and M $\geq$ N, LDVT $\geq$ N; if JOBZ = 'S', LDVT $\geq$ min(M,N).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ 1.
SGESDD
If JOBZ = 'N', LWORK $\geq$ 5*min(M,N).
If JOBZ = 'O',
LWORK $\geq$ 5*min(M,N)*min(M,N) + max(M,N) + 9*min(M,N).
If JOBZ = 'S' or 'A'
LWORK $\geq$ 4*min(M,N)*min(M,N) + max(M,N) + 9*min(M,N).
CGESDD
If JOBZ = 'N', LWORK $\geq$ 2*min(M,N)+max(M,N).
If JOBZ = 'O',
LWORK $\geq$ 2*min(M,N)*min(M,N)+2*min(M,N)+max(M,N).
If JOBZ = 'S' or 'A',
LWORK $\geq$ min(M,N)*min(M,N)+2*min(M,N)+max(M,N).
For good performance, LWORK should generally be larger.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CGESDD only* (workspace) REAL array, dimension (LRWORK)
If JOBZ = 'N', LRWORK $\geq$ 5*min(M,N).
Otherwise, LRWORK $\geq$ 5*min(M,N)*min(M,N) + 7*min(M,N).

IWORK (workspace) INTEGER array, dimension (8*min(M,N))

INFO (output) INTEGER
= 0: successful exit.
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: SBDSDC did not converge, updating process failed.

SGESV/CGESV

```
SUBROUTINE SGESV( N, NRHS, A, LDA, IPIV, B, LDB, INFO )
INTEGER            INFO, LDA, LDB, N, NRHS
INTEGER            IPIV( * )
REAL               A( LDA, * ), B( LDB, * )

SUBROUTINE CGESV( N, NRHS, A, LDA, IPIV, B, LDB, INFO )
INTEGER            INFO, LDA, LDB, N, NRHS
INTEGER            IPIV( * )
COMPLEX            A( LDA, * ), B( LDB, * )
```

Purpose

SGESV/CGESV computes the solution to a real/complex system of linear equations A*X = B, where A is an n-by-n matrix and X and B are n-by-nrhs matrices.

The LU decomposition with partial pivoting and row interchanges is used to factor A as A = P*L*U, where P is a permutation matrix, L is unit lower triangular, and U is upper triangular. The factored form of A is then used to solve the system of equations A*X = B.

Arguments

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the n-by-n coefficient matrix A.
On exit, the factors L and U from the factorization A = P*L*U; the unit diagonal elements of L are not stored.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

IPIV (output) INTEGER array, dimension (N)
The pivot indices that define the permutation matrix P; row i of the matrix was interchanged with row IPIV(i).

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs matrix of right hand side matrix B.
On exit, if INFO = 0, the n-by-nrhs solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

> 0: if INFO = i, U(i,i) is exactly zero. The factorization has been completed, but the factor U is exactly singular, so the solution could not be computed.

SGESVD/CGESVD

```
      SUBROUTINE SGESVD( JOBU, JOBVT, M, N, A, LDA, S, U, LDU, VT, LDVT,
     $                   WORK, LWORK, INFO )
      CHARACTER          JOBU, JOBVT
      INTEGER            INFO, LDA, LDU, LDVT, LWORK, M, N
      REAL               A( LDA, * ), S( * ), U( LDU, * ),
     $                   VT( LDVT, * ), WORK( * )

      SUBROUTINE CGESVD( JOBU, JOBVT, M, N, A, LDA, S, U, LDU, VT, LDVT,
     $                   WORK, LWORK, RWORK, INFO )
      CHARACTER          JOBU, JOBVT
      INTEGER            INFO, LDA, LDU, LDVT, LWORK, M, N
      REAL               RWORK( * ), S( * )
      COMPLEX            A( LDA, * ), U( LDU, * ), VT( LDVT, * ),
     $                   WORK( * )
```

Purpose

SGESVD/CGESVD computes the singular value decomposition (SVD) of a real/complex m-by-n matrix A, optionally computing the left and/or right singular vectors. The SVD is written

$$A = U * \Sigma * V^H$$

where Σ is an m-by-n matrix which is zero except for its min(m,n) diagonal elements, U is an m-by-m orthogonal/unitary matrix, and V is an n-by-n orthogonal/unitary matrix. The diagonal elements of Σ are the singular values of A; they are real and non-negative, and are returned in descending order. The first min(m,n) columns of U and V are the left and right singular vectors of A.

Note that the routine returns V^H, not V.

Arguments

JOBU (input) CHARACTER*1
Specifies options for computing all or part of the matrix U:
= 'A': all M columns of U are returned in array U;
= 'S': the first min(m,n) columns of U (the left singular vectors) are returned in the array U;
= 'O': the first min(m,n) columns of U (the left singular vectors) are overwritten on the array A;
= 'N': no columns of U (no left singular vectors) are computed.

JOBVT (input) CHARACTER*1
Specifies options for computing all or part of the matrix V^H:
= 'A': all N rows of V^H are returned in the array VT;
= 'S': the first min(m,n) rows of V^H (the right singular vectors) are returned in the array VT;
= 'O': the first min(m,n) rows of V^H (the right singular vectors) are overwritten on the array A;
= 'N': no rows of V^H (no right singular vectors) are computed.
JOBVT and JOBU cannot both be 'O'.

M (input) INTEGER
The number of rows of the matrix A. $M \geq 0$.

N (input) INTEGER
The number of columns of the matrix A. $N \geq 0$.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit,
if JOBU = 'O', A is overwritten with the first min(m,n) columns of U (the left singular vectors, stored columnwise);
if JOBVT = 'O', A is overwritten with the first min(m,n) rows of V^H (the right singular vectors, stored rowwise);
if JOBU $\neq$ 'O' and JOBVT $\neq$ 'O', the contents of A are destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

S (output) REAL array, dimension (min(M,N))
The singular values of A, sorted so that $S(i) \geq S(i+1)$.

U (output) REAL/COMPLEX array, dimension (LDU,M) if JOBU = 'A' or (LDU,min(M,N)) if JOBU = 'S'.
If JOBU = 'A', U contains the m-by-m orthogonal/unitary matrix U;
if JOBU = 'S', U contains the first min(m,n) columns of U (the left singular vectors, stored columnwise);
if JOBU = 'N' or 'O', U is not referenced.

LDU (input) INTEGER
The leading dimension of the array U. LDU $\geq$ 1; if JOBU = 'S' or 'A', LDU $\geq$ M.

VT (output) REAL/COMPLEX array, dimension (LDVT,N)
If JOBVT = 'A', VT contains the n-by-n orthogonal/unitary matrix V^H;
if JOBVT = 'S', VT contains the first min(m,n) rows of V^H (the right singular vectors, stored rowwise);
if JOBVT = 'N' or 'O', VT is not referenced.

LDVT (input) INTEGER
The leading dimension of the array VT. LDVT $\geq$ 1; if JOBVT = 'A', LDVT $\geq$ N; if JOBVT = 'S', LDVT $\geq$ min(M,N).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK;

SGESVD
if INFO > 0, WORK(2:min(M,N)) contains the unconverged superdiagonal elements of an upper bidiagonal matrix B whose diagonal is in S (not necessarily sorted). B satisfies A = U*B*VT, so it has the same singular values as A, and singular vectors related by U and VT.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK ≥ 1.
LWORK ≥ max(3*min(M,N)+max(M,N),5*min(M,N)). (*SGESVD*)
LWORK ≥ 2*min(M,N)+max(M,N). (*CGESVD*)
For good performance, LWORK should generally be larger.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CGESVD only* (workspace) REAL array, dimension (5*min(M,N))
On exit, if INFO > 0, RWORK(1:min(M,N)−1) contains the unconverged superdiagonal elements of an upper bidiagonal matrix B whose diagonal is in S (not necessarily sorted). B satisfies A = U*B*VT, so it has the same singular values as A, and singular vectors related by U and VT.

INFO (output) INTEGER
= 0: successful exit.
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if SBDSQR/CBDSQR did not converge, INFO specifies how many superdiagonals of an intermediate bidiagonal form B did not converge to zero.

SGESVX/CGESVX

```
      SUBROUTINE SGESVX( FACT, TRANS, N, NRHS, A, LDA, AF, LDAF, IPIV,
     $                   EQUED, R, C, B, LDB, X, LDX, RCOND, FERR, BERR,
     $                   WORK, IWORK, INFO )
      CHARACTER          EQUED, FACT, TRANS
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * ), IWORK( * )
      REAL               A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   BERR( * ), C( * ), FERR( * ), R( * ),
     $                   WORK( * ), X( LDX, * )
```

```
      SUBROUTINE CGESVX( FACT, TRANS, N, NRHS, A, LDA, AF, LDAF, IPIV,
     $                   EQUED, R, C, B, LDB, X, LDX, RCOND, FERR, BERR,
     $                   WORK, RWORK, INFO )
      CHARACTER          EQUED, FACT, TRANS
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * )
      REAL               BERR( * ), C( * ), FERR( * ), R( * ),
     $                   RWORK( * )
      COMPLEX            A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SGESVX/CGESVX uses the LU factorization to compute the solution to a real/complex system of linear equations A*X = B, where A is an n-by-n matrix and X and B are n-by-nrhs matrices.

Error bounds on the solution and a condition estimate are also provided.

Description

The following steps are performed:

1. If FACT = 'E', real scaling factors are computed to equilibrate the system:

   TRANS = 'N': diag(R)*A*diag(C)*diag(C)^{-1}*X = diag(R)*B
   TRANS = 'T': $(\text{diag(R)*A*diag(C)})^T$*$\text{diag(R)}^{-1}$*X = diag(C)*B
   TRANS = 'C': $(\text{diag(R)*A*diag(C)})^H$*$\text{diag(R)}^{-1}$*X = diag(C)*B

   Whether or not the system will be equilibrated depends on the scaling of the matrix A, but if equilibration is used, A is overwritten by diag(R)*A*diag(C) and B by diag(R)*B (if TRANS='N') or diag(C)*B (if TRANS = 'T' or 'C').

2. If FACT = 'N' or 'E', the LU decomposition is used to factor the matrix A (after equilibration if FACT = 'E') as A = P*L*U, where P is a permutation matrix, L is a unit lower triangular matrix, and U is upper triangular.

3. If some U(i,i)=0, so that U is exactly singular, then the routine returns with INFO = i. Otherwise, the factored form of A is used to estimate the condition number of the matrix A. If the reciprocal of the condition number is less than machine precision, INFO = N+1 is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.

4. The system of equations is solved for X using the factored form of A.

5. Iterative refinement is applied to improve the computed solution matrix and calculate error bounds and backward error estimates for it.

6. If equilibration was used, the matrix X is premultiplied by diag(C) (if TRANS = 'N') or diag(R) (if TRANS = 'T' or 'C') so that it solves the original system before equilibration.

Arguments

FACT (input) CHARACTER*1
Specifies whether or not the factored form of the matrix A is supplied on entry, and if not, whether the matrix A should be equilibrated before it is factored.
= 'F': On entry, AF and IPIV contain the factored form of A. If EQUED $\neq$ 'N', the matrix A has been equilibrated with scaling factors given by R and C. A, AF, and IPIV are not modified.
= 'N': The matrix A will be copied to AF and factored.
= 'E': The matrix A will be equilibrated if necessary, then copied to AF and factored.

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)
= 'C': A^H*X = B (Conjugate transpose)

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the n-by-n matrix A. If FACT = 'F' and EQUED $\neq$ 'N', then A must have been equilibrated by the scaling factors in R and/or C. A is not modified if FACT = 'F' or 'N', or if FACT = 'E' and EQUED = 'N' on exit.
On exit, if EQUED $\neq$ 'N', A is scaled as follows:
EQUED = 'R': A := diag(R)*A;
EQUED = 'C': A := A*diag(C);
EQUED = 'B': A := diag(R)*A*diag(C).

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

AF (input or output) REAL/COMPLEX array, dimension (LDAF,N)
If FACT = 'F', then AF is an input argument and on entry contains the factors L and U from the factorization A = P*L*U as computed by SGETRF/CGETRF. If EQUED $\neq$ 'N', then AF is the factored form of the equilibrated matrix A.
If FACT = 'N', then AF is an output argument and on exit returns the factors L and U from the factorization A = P*L*U of the original matrix A.
If FACT = 'E', then AF is an output argument and on exit returns the factors L and U from the factorization A = P*L*U of the equilibrated matrix A (see the description of A for the form of the equilibrated matrix).

LDAF (input) INTEGER
The leading dimension of the array AF. LDAF $\geq$ max(1,N).

IPIV (input or output) INTEGER array, dimension (N)
If FACT = 'F', then IPIV is an input argument and on entry contains the pivot indices from the factorization A = P*L*U as computed by SGETRF/CGETRF; row i of the matrix was interchanged with row IPIV(i).
If FACT = 'N', then IPIV is an output argument and on exit contains the pivot indices from the factorization A = P*L*U of the original matrix A.
If FACT = 'E', then IPIV is an output argument and on exit contains the pivot indices from the factorization A = P*L*U of the equilibrated matrix A.

EQUED (input or output) CHARACTER*1
Specifies the form of equilibration that was done.
= 'N': No equilibration (always true if FACT = 'N').
= 'R': Row equilibration, i.e., A has been premultiplied by diag(R).
= 'C': Column equilibration, i.e., A has been postmultiplied by diag(C).
= 'B': Both row and column equilibration, i.e., A has been replaced by diag(R)*A*diag(C).
EQUED is an input argument if FACT = 'F'; otherwise, it is an output argument.

R (input or output) REAL array, dimension (N)
The row scale factors for A. If EQUED = 'R' or 'B', A is multiplied on the left by diag(R); if EQUED = 'N' or 'C', R is not accessed. R is an input argument if FACT = 'F'; otherwise, R is an output argument. If FACT = 'F' and EQUED = 'R' or 'B', each element of R must be positive.

C (input or output) REAL array, dimension (N)
The column scale factors for A. If EQUED = 'C' or 'B', A is multiplied on the right by diag(C); if EQUED = 'N' or 'R', C is not accessed. C is an input argument if FACT = 'F'; otherwise, C is an output argument. If FACT = 'F' and EQUED = 'C' or 'B', each element of C must be positive.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit,
if EQUED = 'N', B is not modified;
if TRANS = 'N' and EQUED = 'R' or 'B', B is overwritten by diag(R)*B;
if TRANS = 'T' or 'C' or EQUED = 'C' or 'B', B is overwritten by diag(C)*B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (output) REAL/COMPLEX array, dimension (LDX,NRHS)
If INFO = 0 or INFO = N+1, the N-by-NRHS solution matrix X to the original system of equations. Note that A and B are modified on exit if EQUED .ne. 'N', and the solution to the equilibrated system is $\mathrm{diag}(C)^{-1}*X$ if TRANS = 'N' and EQUED = 'C' or 'B', or $\mathrm{diag}(R)^{-1}*X$ if TRANS = 'T' or 'C' and EQUED = 'R' or 'B'.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

RCOND (output) REAL
The estimate of the reciprocal condition number of the matrix A after equilibration (if done). If RCOND is less than the machine precision (in particular, if RCOND = 0), the matrix is singular to working precision. This condition is indicated by a return code of INFO > 0.

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SGESVX* (workspace/output) REAL array, dimension (4*N)
CGESVX (workspace) COMPLEX array, dimension (2*N)
SGESVX only
On exit, WORK(1) contains the reciprocal pivot growth factor $||A||/||U||$. The "max absolute element" norm is used. If WORK(1) is much less than 1, then the stability of the LU factorization of the (equilibrated) matrix A could be poor. This also means that the solution X, condition estimator RCOND, and forward error bound FERR could be unreliable. If factorization fails with 0<INFO$\leq$N, then WORK(1) contains the reciprocal pivot growth factor for the leading INFO columns of A.

IWORK *SGESVX only* (workspace) INTEGER array, dimension (N)

RWORK *CGESVX only* (workspace/output) REAL array, dimension (2*N)
On exit, RWORK(1) contains the reciprocal pivot growth factor $||A||/||U||$. The "max absolute element" norm is used. If RWORK(1) is much less than 1, then the stability of the LU factorization of the (equilibrated) matrix A could be poor. This also means that the solution X, condition estimator RCOND, and forward error bound FERR could be unreliable. If factorization fails with 0<INFO$\leq$N, then RWORK(1) contains the reciprocal pivot growth factor for the leading INFO columns of A.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
$\leq$ N: U(i,i) is exactly zero. The factorization has been completed, but the factor U is exactly singular, so the solution and error bounds could not be computed. RCOND = 0 is returned.
= N+1: U is nonsingular, but RCOND is less than machine precision, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of RCOND would suggest.

SGETRF/CGETRF

```
SUBROUTINE SGETRF( M, N, A, LDA, IPIV, INFO )
INTEGER            INFO, LDA, M, N
INTEGER            IPIV( * )
REAL               A( LDA, * )

SUBROUTINE CGETRF( M, N, A, LDA, IPIV, INFO )
INTEGER            INFO, LDA, M, N
INTEGER            IPIV( * )
COMPLEX            A( LDA, * )
```

Purpose

SGETRF/CGETRF computes an LU factorization of a general m-by-n matrix A using partial pivoting with row interchanges.

The factorization has the form A = P*L*U where P is a permutation matrix, L is lower triangular with unit diagonal elements (lower trapezoidal if m > n), and U is upper triangular (upper trapezoidal if m < n).

Arguments

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix to be factored.
On exit, the factors L and U from the factorization A = P*L*U; the unit diagonal elements of L are not stored.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

IPIV (output) INTEGER array, dimension (min(M,N))
The pivot indices; for $1 \leq i \leq$ min(M,N), row i of the matrix was interchanged with row IPIV(i).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, U(i,i) is exactly zero. The factorization has been completed, but the factor U is exactly singular, and division by zero will occur if it is used to solve a system of equations.

SGETRI/CGETRI

```
SUBROUTINE SGETRI( N, A, LDA, IPIV, WORK, LWORK, INFO )
INTEGER            INFO, LDA, LWORK, N
INTEGER            IPIV( * )
REAL               A( LDA, * ), WORK( LWORK )

SUBROUTINE CGETRI( N, A, LDA, IPIV, WORK, LWORK, INFO )
INTEGER            INFO, LDA, LWORK, N
INTEGER            IPIV( * )
COMPLEX            A( LDA, * ), WORK( LWORK )
```

Purpose

SGETRI/CGETRI computes the inverse of a matrix using the LU factorization computed by SGETRF/CGETRF.

This method inverts U and then computes A^{-1} by solving the system $A^{-1} * L = U^{-1}$ for A^{-1}.

Arguments

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the factors L and U from the factorization A = P*L*U as computed by SGETRF/CGETRF.
On exit, if INFO = 0, the inverse of the original matrix A.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

IPIV (input) INTEGER array, dimension (N)
The pivot indices from SGETRF/CGETRF; for $1 \leq i \leq N$, row i of the matrix was interchanged with row IPIV(i).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, then WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ max(1,N). For optimal performance LWORK $\geq$ N*NB, where NB is the optimal blocksize returned by ILAENV.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, U(i,i) is exactly zero; the matrix is singular and its inverse could not be computed.

SGETRS/CGETRS

```
SUBROUTINE SGETRS( TRANS, N, NRHS, A, LDA, IPIV, B, LDB, INFO )
CHARACTER          TRANS
INTEGER            INFO, LDA, LDB, N, NRHS
INTEGER            IPIV( * )
REAL               A( LDA, * ), B( LDB, * )

SUBROUTINE CGETRS( TRANS, N, NRHS, A, LDA, IPIV, B, LDB, INFO )
CHARACTER          TRANS
INTEGER            INFO, LDA, LDB, N, NRHS
INTEGER            IPIV( * )
COMPLEX            A( LDA, * ), B( LDB, * )
```

Purpose

SGETRS/CGETRS solves a system of linear equations A*X = B, A^T*X = B, or A^H*X = B, with a general n-by-n matrix A using the LU factorization computed by SGETRF/CGETRF.

Arguments

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)
= 'C': A^H*X = B (Conjugate transpose)

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS ≥ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The factors L and U from the factorization A = P*L*U as computed by SGETRF/CGETRF.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

IPIV (input) INTEGER array, dimension (N)
The pivot indices from SGETRF/CGETRF; for $1 \leq i \leq N$, row i of the matrix was interchanged with row IPIV(i).

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGGBAK/CGGBAK

```
      SUBROUTINE SGGBAK( JOB, SIDE, N, ILO, IHI, LSCALE, RSCALE, M, V,
     $                   LDV, INFO )
      CHARACTER          JOB, SIDE
      INTEGER            IHI, ILO, INFO, LDV, M, N
      REAL               LSCALE( * ), RSCALE( * ), V( LDV, * )

      SUBROUTINE CGGBAK( JOB, SIDE, N, ILO, IHI, LSCALE, RSCALE, M, V,
     $                   LDV, INFO )
      CHARACTER          JOB, SIDE
      INTEGER            IHI, ILO, INFO, LDV, M, N
      REAL               LSCALE( * ), RSCALE( * )
      COMPLEX            V( LDV, * )
```

Purpose

SGGBAK/CGGBAK forms the right or left eigenvectors of a real/complex generalized eigenvalue problem A*x = λ*B*x, by backward transformation on the computed eigenvectors of the balanced pair of matrices output by SGGBAL/CGGBAL.

Arguments

JOB (input) CHARACTER*1
Specifies the type of backward transformation required:
= 'N': do nothing, return immediately;
= 'P': do backward transformation for permutation only;
= 'S': do backward transformation for scaling only;
= 'B': do backward transformations for both permutation and scaling.
JOB must be the same as the argument JOB supplied to SGGBAL/CGGBAL.

SIDE (input) CHARACTER*1
= 'R': V contains right eigenvectors;
= 'L': V contains left eigenvectors.

N (input) INTEGER
The number of rows of the matrix V. N ≥ 0.

ILO, IHI (input) INTEGER
The integers ILO and IHI determined by SGGBAL/CGGBAL.
$1 \leq$ ILO $\leq$ IHI $\leq$ N, if N > 0; ILO = 1 and IHI = 0, if N = 0.

LSCALE (input) REAL array, dimension (N)
Details of the permutations and/or scaling factors applied to the left side of A and B, as returned by SGGBAL/CGGBAL.

RSCALE (input) REAL array, dimension (N)
Details of the permutations and/or scaling factors applied to the right side of A and B, as returned by SGGBAL/CGGBAL.

M (input) INTEGER
The number of columns of the matrix V. M ≥ 0.

V (input/output) REAL/COMPLEX array, dimension (LDV,M)
On entry, the matrix of right or left eigenvectors to be transformed, as returned by STGEVC/CTGEVC.
On exit, V is overwritten by the transformed eigenvectors.

LDV (input) INTEGER
The leading dimension of the matrix V. LDV $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGGBAL/CGGBAL

```
      SUBROUTINE SGGBAL( JOB, N, A, LDA, B, LDB, ILO, IHI, LSCALE,
     $                   RSCALE, WORK, INFO )
      CHARACTER          JOB
      INTEGER            IHI, ILO, INFO, LDA, LDB, N
      REAL               A( LDA, * ), B( LDB, * ), LSCALE( * ),
     $                   RSCALE( * ), WORK( * )
```

```
      SUBROUTINE CGGBAL( JOB, N, A, LDA, B, LDB, ILO, IHI, LSCALE,
     $                   RSCALE, WORK, INFO )
      CHARACTER          JOB
      INTEGER            IHI, ILO, INFO, LDA, LDB, N
      REAL               LSCALE( * ), RSCALE( * ), WORK( * )
      COMPLEX            A( LDA, * ), B( LDB, * )
```

Purpose

SGGBAL/CGGBAL balances a pair of general real/complex matrices (A,B). This involves, first, permuting A and B by similarity transformations to isolate eigenvalues in the first 1 to ilo−1 and last ihi+1 to n elements on the diagonal; and second, applying a diagonal similarity transformation to rows and columns ilo to ihi to make the rows and columns as close in norm as possible. Both steps are optional.

Balancing may reduce the 1-norm of the matrices, and improve the accuracy of the computed eigenvalues and/or eigenvectors in the generalized eigenvalue problem $A*x = \lambda*B*x$.

Arguments

JOB (input) CHARACTER*1
Specifies the operations to be performed on A and B:
= 'N': none: simply set ILO = 1, IHI = N, LSCALE(i) = 1.0 and RSCALE(i) = 1.0 for i=1,...,n;
= 'P': permute only;
= 'S': scale only;
= 'B': both permute and scale.

N (input) INTEGER
The order of the matrices A and B. N ≥ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the input matrix A.
On exit, A is overwritten by the balanced matrix.
If JOB = 'N', A is not referenced.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB,N)
On entry, the input matrix B.
On exit, B is overwritten by the balanced matrix.
If JOB = 'N', B is not referenced.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,N).

ILO, IHI (output) INTEGER
ILO and IHI are set to integers such that on exit A(i,j) = 0 and B(i,j) = 0 if i > j and j = 1,...,ilo−1 or i = ihi+1,...,n.
If JOB = 'N' or 'S', ILO = 1 and IHI = N.

LSCALE (output) REAL array, dimension (N)
Details of the permutations and scaling factors applied to the left side of A and B. If P(j) is the index of the row interchanged with row j, and D(j) is the scaling factor applied to row j, then
LSCALE(j) = P(j) for j = 1,...,ilo−1
= D(j) for j = ilo,...,ihi
= P(j) for j = ihi+1,...,n.
The order in which the interchanges are made is n to ihi+1, then 1 to ilo−1.

RSCALE (output) REAL array, dimension (N)
Details of the permutations and scaling factors applied to the right side of A and B. If P(j) is the index of the column interchanged with column j, and D(j) is the scaling factor applied to column j, then
RSCALE(j) = P(j) for j = 1,...,ilo−1
= D(j) for j = ilo,...,ihi
= P(j) for j = ihi+1,...,n.
The order in which the interchanges are made is n to ihi+1, then 1 to ilo−1.

WORK (workspace) REAL array, dimension (6*N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGGES/CGGES

```
      SUBROUTINE SGGES( JOBVSL, JOBVSR, SORT, SELCTG, N, A, LDA, B, LDB,
     $                  SDIM, ALPHAR, ALPHAI, BETA, VSL, LDVSL, VSR,
     $                  LDVSR, WORK, LWORK, BWORK, INFO )
      CHARACTER          JOBVSL, JOBVSR, SORT
      INTEGER            INFO, LDA, LDB, LDVSL, LDVSR, LWORK, N, SDIM
      LOGICAL            BWORK( * )
      REAL               A( LDA, * ), ALPHAI( * ), ALPHAR( * ),
     $                   B( LDB, * ), BETA( * ), VSL( LDVSL, * ),
     $                   VSR( LDVSR, * ), WORK( * )
      LOGICAL            SELCTG
      EXTERNAL           SELCTG
```

```
      SUBROUTINE CGGES( JOBVSL, JOBVSR, SORT, SELCTG, N, A, LDA, B, LDB,
     $                  SDIM, ALPHA, BETA, VSL, LDVSL, VSR, LDVSR, WORK,
     $                  LWORK, RWORK, BWORK, INFO )
      CHARACTER          JOBVSL, JOBVSR, SORT
      INTEGER            INFO, LDA, LDB, LDVSL, LDVSR, LWORK, N, SDIM
      LOGICAL            BWORK( * )
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), ALPHA( * ), B( LDB, * ),
     $                   BETA( * ), VSL( LDVSL, * ), VSR( LDVSR, * ),
     $                   WORK( * )
      LOGICAL            SELCTG
      EXTERNAL           SELCTG
```

Purpose

SGGES/CGGES computes for a pair of N-by-N real/complex nonsymmetric matrices (A,B), the generalized eigenvalues, the generalized real/complex Schur form (S,T), optionally, the left and/or right matrices of Schur vectors (VSL and VSR). This gives the generalized Schur factorization

$(A,B) = ((VSL)*S*(VSR)^H, (VSL)*T*(VSR)^H)$

Optionally, it also orders the eigenvalues so that a selected cluster of eigenvalues appears in the leading diagonal blocks of the upper quasi-triangular matrix S and the upper triangular matrix T. The leading columns of VSL and VSR then form a(n) orthonormal/unitary basis for the corresponding left and right eigenspaces (deflating subspaces).

(If only the generalized eigenvalues are needed, use the driver SGGEV/CGGEV instead, which is faster.)

A generalized eigenvalue for a pair of matrices (A,B) is a scalar w or a ratio alpha/beta = w, such that A - w*B is singular. It is usually represented as the pair (alpha,beta), as there is a reasonable interpretation for beta=0 or both being zero.

A pair of matrices (S,T) is in generalized real Schur form if T is upper triangular with non-negative diagonal and S is block upper triangular with 1-by-1 and 2-by-2 blocks. 1-by-1 blocks correspond to real generalized eigenvalues, while 2-by-2 blocks of S will be "standardized" by making the corresponding elements of T have the form: $\begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}$ and the pair of corresponding 2-by-2 blocks in S and T will have a complex conjugate pair of generalized eigenvalues.

A pair of matrices (S,T) is in generalized complex Schur form if S and T are upper triangular and, in addition, the diagonal elements of T are non-negative real numbers.

Arguments

JOBVSL (input) CHARACTER*1
= 'N': do not compute the left Schur vectors;
= 'V': compute the left Schur vectors.

JOBVSR (input) CHARACTER*1
= 'N': do not compute the right Schur vectors;
= 'V': compute the right Schur vectors.

SORT (input) CHARACTER*1
Specifies whether or not to order the eigenvalues on the diagonal of the generalized Schur form.
= 'N': Eigenvalues are not ordered;
= 'S': Eigenvalues are ordered (see SELCTG).

SELCTG *SGGES* (input) LOGICAL FUNCTION of three REAL arguments
CGGES (input) LOGICAL FUNCTION of two COMPLEX arguments
SELCTG must be declared EXTERNAL in the calling subroutine.
If SORT = 'N', SELCTG is not referenced.
If SORT = 'S', SELCTG is used to select eigenvalues to sort to the top left of the Schur form.
SGGES
An eigenvalue (ALPHAR(j)+ALPHAI(j))/BETA(j) is selected if SELCTG(ALPHAR(j),ALPHAI(j),BETA(j)) is true; i.e. if either one of a complex conjugate pair of eigenvalues is selected, then both complex eigenvalues are selected.
Note that in the ill-conditioned case, a selected complex eigenvalue may no longer satisfy SELCTG(ALPHAR(j),ALPHAI(j), BETA(j)) = .TRUE. after ordering. INFO is to be set to N+2 in this case.
CGGES
An eigenvalue ALPHA(j)/BETA(j) is selected if SELCTG(ALPHA(j),BETA(j)) is true.
Note that a selected complex eigenvalue may no longer satisfy SELCTG(ALPHA(j),BETA(j)) = .TRUE. after ordering, since ordering may change the value of complex eigenvalues (especially if the eigenvalue is ill-conditioned), in this case INFO is set to N+2 (See INFO below).

N (input) INTEGER
The order of the matrices A, B, VSL, and VSR. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the first of the pair of matrices.
On exit, A has been overwritten by its generalized Schur form S.

LDA (input) INTEGER
The leading dimension of A. LDA $\geq$ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB, N)
On entry, the second of the pair of matrices.
On exit, B has been overwritten by its generalized Schur form T.

LDB (input) INTEGER
The leading dimension of B. LDB $\geq$ max(1,N).

SDIM (output) INTEGER
If SORT = 'N', SDIM = 0. If SORT = 'S', SDIM = number of eigenvalues (after sorting) for which SELCTG is true. (Complex conjugate pairs for which SELCTG is true for either eigenvalue count as 2.)

ALPHAR *SGGES only* (output) REAL array, dimension (N)

ALPHAI *SGGES only* (output) REAL array, dimension (N)

ALPHA *CGGES only* (output) COMPLEX array, dimension (N)

BETA (output) REAL/COMPLEX array, dimension (N)
Note: the quotients ALPHAR(j)/BETA(j) and ALPHAI(j)/BETA(j) (*SGGES*) or ALPHA(j)/BETA(j) (*CGGES*) may easily over- or underflow, and BETA(j) may even be zero. Thus, the user should avoid naively computing the ratio alpha/beta. However, ALPHAR and ALPHAI (*SGGES*) or ALPHA (*CGGES*) will be always less than and usually comparable with norm(A) in magnitude, and BETA always less than and usually comparable with norm(B).
SGGES only
On exit, (ALPHAR(j) + ALPHAI(j)*i)/BETA(j), j=1,...,n, will be the generalized eigenvalues. ALPHAR(j) + ALPHAI(j)*i, j=1,...,n and BETA(j),j=1,...,n are the diagonals of the complex Schur form (A,B) that would result if the 2-by-2 diagonal blocks of the real Schur form of (A,B) were further reduced to triangular form using 2-by-2 complex unitary transformations. If ALPHAI(j) is zero, then the j^{th} eigenvalue is real; if positive, then the j^{th} and $(j+1)^{st}$ eigenvalues are a complex conjugate pair, with ALPHAI(j+1) negative.
CGGES only
On exit, ALPHA(j)/BETA(j), j=1,...,n, will be the generalized eigenvalues. ALPHA(j), j=1,...,n and BETA(j), j=1,...,n are the diagonals of the complex Schur form (A,B) output by CGGES. The BETA(j) will be non-negative real.

VSL (output) REAL/COMPLEX array, dimension (LDVSL,N)
If JOBVSL = 'V', VSL will contain the left Schur vectors.
Not referenced if JOBVSL = 'N'.

LDVSL (input) INTEGER
The leading dimension of the matrix VSL.
LDVSL $\geq$ 1, and if JOBVSL = 'V', LDVSL $\geq$ N.

VSR (output) REAL/COMPLEX array, dimension (LDVSR,N)
If JOBVSR = 'V', VSR will contain the right Schur vectors.
Not referenced if JOBVSR = 'N'.

LDVSR (input) INTEGER
The leading dimension of the matrix VSR.
LDVSR $\geq$ 1, and if JOBVSR = 'V', LDVSR $\geq$ N.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
LWORK $\geq$ max(1,8*N+16). (*SGGES*)
LWORK $\geq$ max(1,2*N). (*CGGES*)
For good performance, LWORK must generally be larger.
If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CGGES only* (workspace) REAL array, dimension (8*N)

BWORK (workspace) LOGICAL array, dimension (N)
Not referenced if SORT = 'N'.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
=1,...,N: The QZ iteration failed. (A,B) are not in Schur form, but ALPHAR(j), ALPHAI(j) (*SGGES*) or ALPHA(j) (*CGGES*), and BETA(j) should be correct for j=info+1,...,n.
>N: errors that usually indicate LAPACK problems:
=N+1: other than QZ iteration failed in SHGEQZ/CHGEQZ
=N+2: after reordering, roundoff changed values of some complex eigenvalues so that leading eigenvalues in the Generalized Schur form no longer satisfy SELCTG=.TRUE. This could also be caused due to scaling.
=N+3: reordering failed in STGSEN/CTGSEN.

SGGESX/CGGESX

```
      SUBROUTINE SGGESX( JOBVSL, JOBVSR, SORT, SELCTG, SENSE, N, A, LDA,
     $                   B, LDB, SDIM, ALPHAR, ALPHAI, BETA, VSL, LDVSL,
     $                   VSR, LDVSR, RCONDE, RCONDV, WORK, LWORK, IWORK,
     $                   LIWORK, BWORK, INFO )
      CHARACTER          JOBVSL, JOBVSR, SENSE, SORT
      INTEGER            INFO, LDA, LDB, LDVSL, LDVSR, LIWORK, LWORK, N,
     $                   SDIM
      LOGICAL            BWORK( * )
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), ALPHAI( * ), ALPHAR( * ),
     $                   B( LDB, * ), BETA( * ), RCONDE( 2 ),
     $                   RCONDV( 2 ), VSL( LDVSL, * ), VSR( LDVSR, * ),
     $                   WORK( * )
      LOGICAL            SELCTG
      EXTERNAL           SELCTG
```

```
      SUBROUTINE CGGESX( JOBVSL, JOBVSR, SORT, SELCTG, SENSE, N, A, LDA,
     $                   B, LDB, SDIM, ALPHA, BETA, VSL, LDVSL, VSR,
     $                   LDVSR, RCONDE, RCONDV, WORK, LWORK, RWORK,
     $                   IWORK, LIWORK, BWORK, INFO )
      CHARACTER          JOBVSL, JOBVSR, SENSE, SORT
      INTEGER            INFO, LDA, LDB, LDVSL, LDVSR, LIWORK, LWORK, N,
     $                   SDIM
      LOGICAL            BWORK( * )
      INTEGER            IWORK( * )
      REAL               RCONDE( 2 ), RCONDV( 2 ), RWORK( * )
      COMPLEX            A( LDA, * ), ALPHA( * ), B( LDB, * ),
     $                   BETA( * ), VSL( LDVSL, * ), VSR( LDVSR, * ),
     $                   WORK( * )
      LOGICAL            SELCTG
      EXTERNAL           SELCTG
```

Purpose

SGGESX/CGGESX computes for a pair of N-by-N real/complex nonsymmetric matrices (A,B), the generalized eigenvalues, the real/complex Schur form (S,T), and, optionally, the left and/or right matrices of Schur vectors (VSL and VSR). This gives the generalized Schur factorization

$$(A,B) = ((VSL)*S*(VSR)^H, (VSL)*T*(VSR)^H)$$

Optionally, it also orders the eigenvalues so that a selected cluster of eigenvalues appears in the leading diagonal blocks of the upper triangular matrix S and the upper triangular matrix T; computes a reciprocal condition number for the average of the selected eigenvalues (RCONDE); and computes a reciprocal condition number for the right and left deflating subspaces corresponding to the selected eigenvalues (RCONDV). The leading columns of VSL and VSR then form a(n) orthonormal/unitary basis for the corresponding left and right eigenspaces (deflating subspaces).

A generalized eigenvalue for a pair of matrices (A,B) is a scalar w or a ratio alpha/beta = w, such that A - w*B is singular. It is usually represented as the pair (alpha,beta), as there is a reasonable interpretation for beta=0 or for both being zero.

A pair of matrices (S,T) is in generalized real/complex Schur form if T is upper triangular with non-negative diagonal and S is upper triangular.

Arguments

JOBVSL (input) CHARACTER*1
- = 'N': do not compute the left Schur vectors;
- = 'V': compute the left Schur vectors.

JOBVSR (input) CHARACTER*1
- = 'N': do not compute the right Schur vectors;
- = 'V': compute the right Schur vectors.

SORT (input) CHARACTER*1
Specifies whether or not to order the eigenvalues on the diagonal of the generalized Schur form.
- = 'N': Eigenvalues are not ordered;
- = 'S': Eigenvalues are ordered (see SELCTG).

SELCTG *SGGESX* (input) LOGICAL FUNCTION of three REAL arguments
CGGESX (input) LOGICAL FUNCTION of two COMPLEX arguments
SELCTG must be declared EXTERNAL in the calling subroutine.
If SORT = 'N', SELCTG is not referenced.
If SORT = 'S', SELCTG is used to select eigenvalues to sort to the top left of the Schur form.
SGGESX
An eigenvalue (ALPHAR(j)+ALPHAI(j))/BETA(j) is selected if SELCTG(ALPHAR(j),ALPHAI(j),BETA(j)) is true; i.e. if either one of a complex conjugate pair of eigenvalues is selected, then both complex eigenvalues are selected.
Note that a selected complex eigenvalue may no longer satisfy SELCTG(ALPHAR(j),ALPHAI(j),BETA(j)) = .TRUE. after ordering, since ordering may change the value of complex eigenvalues (especially if the eigenvalue is ill-conditioned), in this case INFO is set to N+3.
CGGESX
An eigenvalue ALPHA(j)/BETA(j) is selected if SELCTG(ALPHA(j),BETA(j)) is true.
Note that a selected complex eigenvalue may no longer satisfy SELCTG(ALPHA(j),BETA(j)) = .TRUE. after ordering, since ordering may change the value of complex eigenvalues (especially if the eigenvalue is ill-conditioned), in this case INFO is set to N+3 (See INFO below).

SENSE (input) CHARACTER*1
Determines which reciprocal condition numbers are computed.
- = 'N': None are computed;
- = 'E': Computed for average of selected eigenvalues only;
- = 'V': Computed for selected deflating subspaces only;
- = 'B': Computed for both.

If SENSE = 'E', 'V', or 'B', SORT must equal 'S'.

N (input) INTEGER
The order of the matrices A, B, VSL, and VSR. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the first of the pair of matrices.
On exit, A has been overwritten by its generalized Schur form S.

LDA (input) INTEGER
The leading dimension of A. LDA $\geq$ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB, N)
On entry, the second of the pair of matrices.
On exit, B has been overwritten by its generalized Schur form T.

LDB (input) INTEGER
The leading dimension of B. LDB $\geq$ max(1,N).

SDIM (output) INTEGER
If SORT = 'N', SDIM = 0. If SORT = 'S', SDIM = number of eigenvalues (after sorting) for which SELCTG is true. (Complex conjugate pairs for which SELCTG is true for either eigenvalue count as 2.)

ALPHAR *SGGESX only* (output) REAL array, dimension (N)

ALPHAI *SGGESX only* (output) REAL array, dimension (N)

ALPHA *CGGESX only* (output) COMPLEX array, dimension (N)

BETA (output) REAL/COMPLEX array, dimension (N)
Note: the quotients ALPHAR(j)/BETA(j) and ALPHAI(j)/BETA(j) (*SGGESX*) or ALPHA(j)/BETA(j) (*CGGESX*) may easily over- or underflow, and BETA(j) may even be zero. Thus, the user should avoid naively computing the ratio alpha/beta. However, ALPHAR and ALPHAI (*SGGESX*) or ALPHA (*CGGESX*) will be always less than and usually comparable with norm(A) in magnitude, and BETA always less than and usually comparable with norm(B).
SGGESX only
On exit, (ALPHAR(j) + ALPHAI(j)*i)/BETA(j), j=1,...,n, will be the generalized eigenvalues. ALPHAR(j) + ALPHAI(j)*i, j=1,...,n and BETA(j),j=1,...,n are the diagonals of the complex Schur form (A,B) that would result if the 2-by-2 diagonal blocks of the real Schur form of (A,B) were further reduced to triangular form using 2-by-2 complex unitary transformations. If ALPHAI(j) is zero, then the j^{th} eigenvalue is real; if positive, then the j^{th} and $(j+1)^{st}$ eigenvalues are a complex conjugate pair, with ALPHAI(j+1) negative.
CGGESX only
On exit, ALPHA(j)/BETA(j), j=1,...,n, will be the generalized eigenvalues. ALPHA(j), j=1,...,n and BETA(j), j=1,...,n are the diagonals of the complex Schur form (A,B) output by CGGESX. The BETA(j) will be non-negative real.

VSL (output) REAL/COMPLEX array, dimension (LDVSL,N)
If JOBVSL = 'V', VSL will contain the left Schur vectors.
Not referenced if JOBVSL = 'N'.

LDVSL (input) INTEGER
The leading dimension of the matrix VSL. LDVSL ≥ 1, and if JOBVSL = 'V', LDVSL ≥ N.

VSR (output) REAL/COMPLEX array, dimension (LDVSR,N)
If JOBVSR = 'V', VSR will contain the right Schur vectors.
Not referenced if JOBVSR = 'N'.

LDVSR (input) INTEGER
The leading dimension of the matrix VSR. LDVSR ≥ 1, and if JOBVSR = 'V', LDVSR ≥ N.

RCONDE (output) REAL array, dimension (2)
If SENSE = 'E' or 'B', RCONDE(1) and RCONDE(2) contain the reciprocal condition numbers for the average of the selected eigenvalues.
Not referenced if SENSE = 'N' or 'V'.

RCONDV (output) REAL array, dimension (2)
If SENSE = 'V' or 'B', RCONDV(1) and RCONDV(2) contain the reciprocal condition numbers for the selected deflating subspaces.
Not referenced if SENSE = 'N' or 'E'.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
SGGESX
LWORK ≥ max(1,8*(N+1)+16). If SENSE = 'E', 'V', or 'B', LWORK ≥ MAX(8*(N+1)+16, 2*SDIM*(N-SDIM)).
CGGESX
LWORK ≥ max(1,2*N). If SENSE = 'E', 'V', or 'B', LWORK ≥ MAX(2*N, 2*SDIM*(N-SDIM)).
For good performance, LWORK must generally be larger.

RWORK *CGGESX only* (workspace) REAL array, dimension (8*N)

IWORK (workspace) INTEGER array, dimension (LIWORK)
Not referenced if SENSE = 'N'.

LIWORK (input) INTEGER
The dimension of the array WORK. LIWORK ≥ N+6. (*SGGESX*)
The dimension of the array WORK. LIWORK ≥ N+2. (*CGGESX*)

BWORK (workspace) LOGICAL array, dimension (N)
Not referenced if SORT = 'N'.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
=1,...,N: The QZ iteration failed. (A,B) are not in Schur form, but ALPHAR(j), ALPHAI(j) (*SGGESX*) or ALPHA(j) (*CGGESX*), and BETA(j) should be correct for j=info+1,...,n.
>N: errors that usually indicate LAPACK problems:
=N+1: other than QZ iteration failed in SHGEQZ/CHGEQZ
=N+2: after reordering, roundoff changed values of some complex eigenvalues so that leading eigenvalues in the Generalized Schur form no longer satisfy SELCTG=.TRUE. This could also be caused due to scaling.
=N+3: reordering failed in STGSEN/CTGSEN.

SGGEV/CGGEV

```
      SUBROUTINE SGGEV( JOBVL, JOBVR, N, A, LDA, B, LDB, ALPHAR, ALPHAI,
     $                  BETA, VL, LDVL, VR, LDVR, WORK, LWORK, INFO )
      CHARACTER          JOBVL, JOBVR
      INTEGER            INFO, LDA, LDB, LDVL, LDVR, LWORK, N
      REAL               A( LDA, * ), ALPHAI( * ), ALPHAR( * ),
     $                   B( LDB, * ), BETA( * ), VL( LDVL, * ),
     $                   VR( LDVR, * ), WORK( * )

      SUBROUTINE CGGEV( JOBVL, JOBVR, N, A, LDA, B, LDB, ALPHA, BETA,
     $                  VL, LDVL, VR, LDVR, WORK, LWORK, RWORK, INFO )
      CHARACTER          JOBVL, JOBVR
      INTEGER            INFO, LDA, LDB, LDVL, LDVR, LWORK, N
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), ALPHA( * ), B( LDB, * ),
     $                   BETA( * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   WORK( * )
```

Purpose

SGGEV/CGGEV computes for a pair of N-by-N real/complex nonsymmetric matrices (A,B), the generalized eigenvalues, and optionally, the left and/or right generalized eigenvectors.

A generalized eigenvalue for a pair of matrices (A,B) is a scalar λ or a ratio alpha/beta = λ, such that A - λ*B is singular. It is usually represented as the pair (alpha,beta), as there is a reasonable interpretation for beta=0, and even for both being zero.

The right generalized eigenvector $v(j)$ corresponding to the generalized eigenvalue $\lambda(j)$ of (A,B) satisfies

$$A * v(j) = \lambda(j) * B * v(j).$$

The left generalized eigenvector $u(j)$ corresponding to the generalized eigenvalues $\lambda(j)$ of (A,B) satisfies

$$u(j)^H * A = \lambda(j) * u(j)^H * B$$

where $u(j)^H$ denotes the conjugate-transpose of $u(j)$.

Arguments

JOBVL (input) CHARACTER*1
= 'N': do not compute the left generalized eigenvectors;
= 'V': compute the left generalized eigenvectors.

JOBVR (input) CHARACTER*1
= 'N': do not compute the right generalized eigenvectors;
= 'V': compute the right generalized eigenvectors.

N (input) INTEGER
The order of the matrices A, B, VL, and VR. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the matrix A in the pair (A,B).
On exit, A has been overwritten.

LDA (input) INTEGER
The leading dimension of A. LDA $\geq$ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB, N)
On entry, the matrix B in the pair (A,B).
On exit, B has been overwritten.

LDB (input) INTEGER
The leading dimension of B. LDB $\geq$ max(1,N).

ALPHAR *SGGEV only* (output) REAL array, dimension (N)

ALPHAI *SGGEV only* (output) REAL array, dimension (N)

ALPHA *CGGEV only* (output) COMPLEX array, dimension (N)

BETA (output) REAL/COMPLEX array, dimension (N)
Note: the quotients ALPHAR(j)/BETA(j) and ALPHAI(j)/BETA(j) (*SGGEV*) or ALPHA(j)/BETA(j) (*CGGEV*) may easily over- or underflow, and BETA(j) may even be zero. Thus, the user should avoid naively computing the ratio alpha/beta. However, ALPHAR and ALPHAI (*SGGEV*) or ALPHA (*CGGEV*) will be always less than and usually comparable with norm(A) in magnitude, and BETA always less than and usually comparable with norm(B).
SGGES only
On exit, (ALPHAR(j) + ALPHAI(j)*i)/BETA(j), j=1,...,n, will be the generalized eigenvalues. If ALPHAI(j) is zero, then the j^{th} eigenvalue is real; if positive, then the j^{th} and $(j+1)^{st}$ eigenvalues are a complex conjugate pair, with ALPHAI(j+1) negative.
CGGES only
On exit, ALPHA(j)/BETA(j), j=1,...,n, will be the generalized eigenvalues.

VL (output) REAL/COMPLEX array, dimension (LDVL,N)
SGGEV
If JOBVL = 'V', the left eigenvectors u(j) are stored one after another in the columns of VL, in the same order as their eigenvalues. If the j^{th} eigenvalue is real, then u(j) = VL(:,j), the j^{th} column of VL. If the j^{th} and $(j+1)^{th}$ eigenvalues form a complex conjugate pair, then u(j) = VL(:,j)+i*VL(:,j+1) and u(j+1) = VL(:,j)−i*VL(:,j+1). Each eigenvector will be scaled so the largest component have abs(real part)+abs(imag. part)=1. Not referenced if JOBVL = 'N'.
CGGEV
If JOBVL = 'V', the left generalized eigenvectors u(j) are stored one after another in the columns of VL, in the same order as their eigenvalues. Each eigenvector will be scaled so the largest component will have abs(real part) + abs(imag. part) = 1. Not referenced if JOBVL = 'N'.

LDVL (input) INTEGER
The leading dimension of the matrix VL.
LDVL $\geq$ 1, and if JOBVL = 'V', LDVL $\geq$ N.

VR (output) REAL/COMPLEX array, dimension (LDVR,N)
Not referenced if JOBVR = 'N'.
SGGEV
If JOBVR = 'V', the right eigenvectors v(j) are stored one after another in the columns of VR, in the same order as their eigenvalues. If the j^{th} eigenvalue is real, then v(j) = VR(:,j), the j-th column of VR. If the j^{th} and $(j+1)^{th}$ eigenvalues form a complex conjugate pair, then v(j) = VR(:,j)+i*VR(:,j+1) and v(j+1) = VR(:,j)−i*VR(:,j+1). Each eigenvector will be scaled so the largest component have abs(real part)+abs(imag. part)=1. *CGGEV*
If JOBVR = 'V', the right generalized eigenvectors v(j) are stored one after another in the columns of VR, in the same order as their eigenvalues. Each eigenvector will be scaled so the largest component will have abs(real part) + abs(imag. part) = 1.

LDVR (input) INTEGER
The leading dimension of the matrix VR.
LDVR $\geq$ 1, and if JOBVR = 'V', LDVR $\geq$ N.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
LWORK $\geq$ max(1,8*N+16). (*SGGEV*)
LWORK $\geq$ max(1,2*N). (*CGGEV*)
For good performance, LWORK must generally be larger.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CGGEV only* (workspace) REAL array, dimension (8*N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
=1,...,N: The QZ iteration failed. No eigenvectors have been calculated, but ALPHAR(j), ALPHAI(j) (*SGGEV*) or ALPHA(j) (*CGGEV*), and BETA(j) should be correct for j=info+1,...,n.
>N: errors that usually indicate LAPACK problems:
=N+1: other than QZ iteration failed in SHGEQZ/CHGEQZ.
=N+2: error return from STGEVC/CTGEVC.

SGGEVX/CGGEVX

```
      SUBROUTINE SGGEVX( BALANC, JOBVL, JOBVR, SENSE, N, A, LDA, B, LDB,
     $                   ALPHAR, ALPHAI, BETA, VL, LDVL, VR, LDVR, ILO,
     $                   IHI, LSCALE, RSCALE, ABNRM, BBNRM, RCONDE,
     $                   RCONDV, WORK, LWORK, IWORK, BWORK, INFO )
      CHARACTER          BALANC, JOBVL, JOBVR, SENSE
      INTEGER            IHI, ILO, INFO, LDA, LDB, LDVL, LDVR, LWORK, N
      REAL               ABNRM, BBNRM
      LOGICAL            BWORK( * )
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), ALPHAI( * ), ALPHAR( * ),
     $                   B( LDB, * ), BETA( * ), LSCALE( * ),
     $                   RCONDE( * ), RCONDV( * ), RSCALE( * ),
     $                   VL( LDVL, * ), VR( LDVR, * ), WORK( * )

      SUBROUTINE CGGEVX( BALANC, JOBVL, JOBVR, SENSE, N, A, LDA, B, LDB,
     $                   ALPHA, BETA, VL, LDVL, VR, LDVR, ILO, IHI,
     $                   LSCALE, RSCALE, ABNRM, BBNRM, RCONDE, RCONDV,
     $                   WORK, LWORK, RWORK, IWORK, BWORK, INFO )
      CHARACTER          BALANC, JOBVL, JOBVR, SENSE
      INTEGER            IHI, ILO, INFO, LDA, LDB, LDVL, LDVR, LWORK, N
      REAL               ABNRM, BBNRM
      LOGICAL            BWORK( * )
      INTEGER            IWORK( * )
      REAL               LSCALE( * ), RCONDE( * ), RCONDV( * ),
     $                   RSCALE( * ), RWORK( * )
      COMPLEX            A( LDA, * ), ALPHA( * ), B( LDB, * ),
     $                   BETA( * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   WORK( * )
```

Purpose

SGGEVX computes for a pair of N-by-N real nonsymmetric matrices (A,B) the generalized eigenvalues, and optionally, the left and/or right generalized eigenvectors.

Optionally also, it computes a balancing transformation to improve the conditioning of the eigenvalues and eigenvectors (ILO, IHI, LSCALE, RSCALE, ABNRM, and BBNRM), reciprocal condition numbers for the eigenvalues (RCONDE), and reciprocal condition numbers for the right eigenvectors (RCONDV).

A generalized eigenvalue for a pair of matrices (A,B) is a scalar λ or a ratio alpha/beta = λ, such that A $-\lambda$*B is singular. It is usually represented as the pair (alpha,beta), as there is a reasonable interpretation for beta=0, and even for both being zero.

The right eigenvector $v(j)$ corresponding to the eigenvalue $\lambda(j)$ of (A,B) satisfies

$$A * v(j) \quad = \quad \lambda(j) * B * v(j).$$

The left eigenvector $u(j)$ corresponding to the eigenvalue $\lambda(j)$ of (A,B) satisfies

$$u(j)^H * A \quad = \quad \lambda(j) * u(j)^H * B$$

where $u(j)^H$ denotes the conjugate-transpose of $u(j)$.

Arguments

BALANC (input) CHARACTER*1
Specifies the balance option to be performed.
= 'N': do not diagonally scale or permute;
= 'P': permute only;
= 'S': scale only;
= 'B': both permute and scale.
Computed reciprocal condition numbers will be for the matrices after permuting and/or balancing. Permuting does not change condition numbers (in exact arithmetic), but balancing does.

JOBVL (input) CHARACTER*1
= 'N': do not compute the left generalized eigenvectors;
= 'V': compute the left generalized eigenvectors.

JOBVR (input) CHARACTER*1
= 'N': do not compute the right generalized eigenvectors;
= 'V': compute the right generalized eigenvectors.

SENSE (input) CHARACTER*1
Determines which reciprocal condition numbers are computed.
= 'N': none are computed;
= 'E': computed for eigenvalues only;
= 'V': computed for eigenvectors only;
= 'B': computed for eigenvalues and eigenvectors.

N (input) INTEGER
The order of the matrices A, B, VL, and VR. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the matrix A in the pair (A,B).
On exit, A has been overwritten. If JOBVL='V' or JOBVR='V' or both, then A contains the first part of the real Schur form of the "balanced" versions of the input A and B.

LDA (input) INTEGER
The leading dimension of A. LDA $\geq$ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB, N)
On entry, the matrix B in the pair (A,B).
On exit, B has been overwritten. If JOBVL='V' or JOBVR='V' or both, then B contains the second part of the real Schur form of the "balanced" versions of the input A and B.

LDB (input) INTEGER
The leading dimension of B. LDB $\geq$ max(1,N).

ALPHAR *SGGEVX only* (output) REAL array, dimension (N)

ALPHAI *SGGEVX only* (output) REAL array, dimension (N)

ALPHA *CGGEVX only* (output) COMPLEX array, dimension (N)

BETA (output) REAL/COMPLEX array, dimension (N)
Note: the quotients ALPHAR(j)/BETA(j) and ALPHAI(j)/BETA(j) (*SGGEVX*) or ALPHA(j)/BETA(j) (*CGGEVX*) may easily over- or underflow, and BETA(j) may even be zero. Thus, the user should avoid naively computing the ratio alpha/beta. However, ALPHAR and ALPHAI (*SGGEVX*) or ALPHA (*CGGEVX*) will be always less than and usually comparable with norm(A) in magnitude, and BETA always less than and usually comparable with norm(B).
SGGEVX only
On exit, (ALPHAR(j) + ALPHAI(j)*i)/BETA(j), j=1,...,n, will be the generalized eigenvalues. If ALPHAI(j) is zero, then the j^{th} eigenvalue is real; if positive, then the j^{th} and $(j+1)^{st}$ eigenvalues are a complex conjugate pair, with ALPHAI(j+1) negative.
CGGEVX only
On exit, ALPHA(j)/BETA(j), j=1,...,n, will be the generalized eigenvalues.

VL (output) REAL/COMPLEX array, dimension (LDVL,N)
Not referenced if JOBVL = 'N'.
SGGEVX
If JOBVL = 'V', the left eigenvectors u(j) are stored one after another in the columns of VL, in the same order as their eigenvalues. If the j^{th} eigenvalue is real, then u(j) = VL(:,j), the j^{th} column of VL. If the j^{th} and $(j+1)^{st}$ eigenvalues form a complex conjugate pair, then u(j) = VL(:,j)+i*VL(:,j+1) and u(j+1) = VL(:,j)−i*VL(:,j+1). Each eigenvector will be scaled so the largest component have abs(real part) + abs(imag. part) = 1.
CGGEVX
If JOBVL = 'V', the left generalized eigenvectors u(j) are stored one after another in the columns of VL, in the same order as their eigenvalues. Each eigenvector will be scaled so the largest component will have abs(real part) + abs(imag. part) = 1.

LDVL (input) INTEGER
The leading dimension of the matrix VL.
LDVL $\geq$ 1, and if JOBVL = 'V', LDVL $\geq$ N.

VR (output) REAL/COMPLEX array, dimension (LDVR,N)
Not referenced if JOBVR = 'N'.
SGGEVX
If JOBVR = 'V', the right eigenvectors v(j) are stored one after another in the columns of VR, in the same order as their eigenvalues. If the j^{th} eigenvalue is real, then v(j) = VR(:,j), the j^{th} column of VR. If the j^{th} and $(j+1)^{st}$ eigenvalues form a complex conjugate pair, then v(j) = VR(:,j)+i*VR(:,j+1) and v(j+1) = VR(:,j)−i*VR(:,j+1). Each eigenvector will be scaled so the largest component have abs(real part)

+ abs(imag. part) = 1.
CGGEVX
If JOBVL = 'V', the right generalized eigenvectors v(j) are stored one after another in the columns of VR, in the same order as their eigenvalues. Each eigenvector will be scaled so the largest component will have abs(real part) + abs(imag. part) = 1.

LDVR (input) INTEGER
The leading dimension of the matrix VR.
LDVR ≥ 1, and if JOBVR = 'V', LDVR ≥ N.

ILO, IHI (output) INTEGER
ILO and IHI are integer values such that on exit A(i,j) = 0 and B(i,j) = 0 if i > j and
j = 1,...,ilo−1 or i = ihi+1,...,n.
If BALANC = 'N' or 'S', ilo = 1 and ihi = N.

LSCALE (output) REAL array, dimension (N)
Details of the permutations and scaling factors applied to the left side of A and B. If PL(j) is the index of the row interchanged with row j, and DL(j) is the scaling factor applied to row j, then
LSCALE(j) = PL(j) for j = 1,...,ilo−1
= DL(j) for j = ilo,...,ihi
= PL(j) for j = ihi+1,...,n.
The order in which the interchanges are made is n to ihi+1, then 1 to ilo−1.

RSCALE (output) REAL array, dimension (N)
Details of the permutations and scaling factors applied to the right side of A and B. If PR(j) is the index of the column interchanged with column j, and DR(j) is the scaling factor applied to column j, then
RSCALE(j) = PR(j) for j = 1,...,ilo−1
= DR(j) for j = ilo,...,ihi
= PR(j) for j = ihi+1,...,n.
The order in which the interchanges are made is n to ihi+1, then 1 to ilo−1.

ABNRM (output) REAL
The one-norm of the balanced matrix A.

BBNRM (output) REAL
The one-norm of the balanced matrix B.

RCONDE (output) REAL array, dimension (N)
If SENSE = 'V', RCONDE is not referenced.
SGGEVX
If SENSE = 'E' or 'B', the reciprocal condition numbers of the selected eigenvalues, stored in consecutive elements of the array. For a complex conjugate pair of eigenvalues two consecutive elements of RCONDE are set to the same value. Thus RCONDE(j), RCONDV(j), and the j^{th} columns of VL and VR all correspond to the same eigenpair (but not in general the j^{th} eigenpair, unless all eigenpairs are selected). If SENSE = 'V', RCONDE is not referenced.
CGGEVX
If SENSE = 'E' or 'B', the reciprocal condition numbers of the selected eigenvalues, stored in consecutive elements of the array.

RCONDV (output) REAL array, dimension (N)
If SENSE = 'E', RCONDV is not referenced.
SGGEVX
If SENSE = 'V' or 'B', the estimated reciprocal condition numbers of the selected eigenvectors, stored in consecutive elements of the array. For a complex eigenvector two consecutive elements of RCONDV are set to the same value. If the eigenvalues cannot be reordered to compute RCONDV(j), RCONDV(j) is set to 0; this can only occur when the true value would be very small anyway.
CGGEVX
If SENSE = 'V' or 'B', the estimated reciprocal condition numbers of the selected eigenvectors, stored in consecutive elements of the array. If the eigenvalues cannot be reordered to compute RCONDV(j), RCONDV(j) is set to 0; this can only occur when the true value would be very small anyway.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
SGGEVX
The dimension of the array WORK. LWORK ≥ max(1,6*N).
If SENSE = 'E', LWORK ≥ 12*N.
If SENSE = 'V' or 'B', LWORK ≥ 2*N*N+12*N+16.
CGGEVX
The dimension of the array WORK. LWORK ≥ max(1,2*N).
If SENSE = 'N' or 'E', LWORK ≥ 2*N.
If SENSE = 'V' or 'B', LWORK ≥ 2*N*N+2*N.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CGGEVX only* (workspace) REAL array, dimension (6*N)

IWORK *SGGEVX* (workspace) INTEGER array, dimension (N+6)
CGGEVX (workspace) INTEGER array, dimension (N+2)
If SENSE = 'E', IWORK is not referenced.

BWORK (workspace) LOGICAL array, dimension (N)
If SENSE = 'N', BWORK is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
=1,...,N: The QZ iteration failed. No eigenvectors have been calculated, but ALPHAR(j), ALPHAI(j) (*SGGEVX*) or ALPHA(j)

($CGGEVX$), and BETA(j) should be correct for j=info+1,...,n.
- \>N: errors that usually indicate LAPACK problems:
  - =N+1: other than QZ iteration failed in SHGEQZ/CHGEQZ
  - =N+2: error return from STGEVC/CTGEVC.

SGGGLM/CGGGLM

```
      SUBROUTINE SGGGLM( N, M, P, A, LDA, B, LDB, D, X, Y, WORK, LWORK,
     $                   INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, P
      REAL               A( LDA, * ), B( LDB, * ), D( * ), WORK( * ),
     $                   X( * ), Y( * )

      SUBROUTINE CGGGLM( N, M, P, A, LDA, B, LDB, D, X, Y, WORK, LWORK,
     $                   INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, P
      COMPLEX            A( LDA, * ), B( LDB, * ), D( * ), WORK( * ),
     $                   X( * ), Y( * )
```

Purpose

SGGGLM/CGGGLM solves a general Gauss-Markov linear model (GLM) problem:

$$\min_x \| y \|_2 \text{ subject to } d = A*x + B*y$$

where A is an n-by-m matrix, B is an n-by-p matrix, and d is a given n-vector. It is assumed that $m \le n \le m+p$ and

$$\mathrm{rank}(A) = m, \quad \text{and} \quad \mathrm{rank}(\ A\ B\ ) = n.$$

Under these assumptions, the constrained equation is always consistent, and there is a unique solution x and a minimal 2-norm solution y, which is obtained using a generalized QR factorization of A and B.

In particular, if matrix B is square nonsingular, then the problem GLM is equivalent to the following weighted linear least squares problem

$$\min_x \| B^{-1}*(d - A*x) \|_2.$$

Arguments

N (input) INTEGER
The number of rows of the matrices A and B. $N \ge 0$.

M (input) INTEGER
The number of columns of the matrix A. $0 \le M \le N$.

P (input) INTEGER
The number of columns of the matrix B. $P \ge N-M$.

A (input/output) REAL/COMPLEX array, dimension (LDA,M)
On entry, the n-by-m matrix A.
On exit, A is destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\ge$ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB,P)
On entry, the n-by-p matrix B.
On exit, B is destroyed.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\ge$ max(1,N).

D (input/output) REAL/COMPLEX array, dimension (N)
On entry, D is the left hand side of the GLM equation.
On exit, D is destroyed.

X (output) REAL/COMPLEX array, dimension (M)

Y (output) REAL/COMPLEX array, dimension (P)
On exit, X and Y are the solutions of the GLM problem.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\ge$ max(1,N+M+P).
For optimum performance LWORK $\ge$ M+min(N,P)+max(N,P)*NB, where NB is an upper bound for the optimal blocksizes for SGEQRF/CGEQRF, SGERQF/CGERQF, SORMQR/CUNMQR, and SORMRQ/CUNMRQ.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
- = 0: successful exit
- < 0: if INFO = −i, the i^{th} argument had an illegal value.

SGGHRD/CGGHRD

```
      SUBROUTINE SGGHRD( COMPQ, COMPZ, N, ILO, IHI, A, LDA, B, LDB, Q,
     $                   LDQ, Z, LDZ, INFO )
      CHARACTER          COMPQ, COMPZ
      INTEGER            IHI, ILO, INFO, LDA, LDB, LDQ, LDZ, N
      REAL               A( LDA, * ), B( LDB, * ), Q( LDQ, * ),
     $                   Z( LDZ, * )
```

```
      SUBROUTINE CGGHRD( COMPQ, COMPZ, N, ILO, IHI, A, LDA, B, LDB, Q,
     $                   LDQ, Z, LDZ, INFO )
      CHARACTER          COMPQ, COMPZ
      INTEGER            IHI, ILO, INFO, LDA, LDB, LDQ, LDZ, N
      COMPLEX            A( LDA, * ), B( LDB, * ), Q( LDQ, * ),
     $                   Z( LDZ, * )
```

Purpose

SGGHRD/CGGHRD reduces a pair of real/complex matrices (A,B) to generalized upper Hessenberg form using orthogonal/unitary transformations, where A is a general matrix and B is upper triangular: $Q^H*A*Z = H$ and $Q^H*B*Z = T$, where H is upper Hessenberg, T is upper triangular, and Q and Z are orthogonal/unitary.

The orthogonal/unitary matrices Q and Z are determined as products of Givens rotations. They may either be formed explicitly, or they may be postmultiplied into input matrices Q_1 and Z_1, so that

$$Q_1*A*Z_1^H = (Q_1Q)*H*(Z_1Z)^H$$

$$Q_1*B*Z_1^H = (Q_1Q)*T*(Z_1Z)^H$$

Arguments

COMPQ (input) CHARACTER*1
- = 'N': do not compute Q;
- = 'I': Q is initialized to the unit matrix, and the orthogonal/unitary matrix Q is returned;
- = 'V': Q must contain an orthogonal/unitary matrix Q_1 on entry, and the product Q_1*Q is returned.

COMPZ (input) CHARACTER*1
- = 'N': do not compute Z;
- = 'I': Z is initialized to the unit matrix, and the orthogonal/unitary matrix Z is returned;
- = 'V': Z must contain an orthogonal/unitary matrix Z_1 on entry, and the product Z_1*Z is returned.

N (input) INTEGER
The order of the matrices A and B. N $\geq$ 0.

ILO, IHI (input) INTEGER
It is assumed that A is already upper triangular in rows and columns 1:ilo−1 and ihi+1:n. ILO and IHI are normally set by a previous call to SGGBAL/CGGBAL; otherwise they should be set to 1 and N respectively.
$1 \leq$ ILO $\leq$ IHI $\leq$ N, if N > 0; ILO = 1 and IHI = 0, if N = 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the n-by-n general matrix to be reduced.
On exit, the upper triangle and the first subdiagonal of A are overwritten with the upper Hessenberg matrix H, and the rest is set to zero.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB,N)
On entry, the n-by-n upper triangular matrix B.
On exit, the upper triangular matrix $T = Q^H*B*Z$. The elements below the diagonal are set to zero.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

Q (input/output) REAL/COMPLEX array, dimension (LDQ,N)
If COMPQ = 'N': Q is not referenced.
If COMPQ = 'I': on entry, Q need not be set, and on exit it contains the orthogonal/unitary matrix Q, where Q^H is the product of the Givens transformations which are applied to A and B on the left.
If COMPQ = 'V': on entry, Q must contain an orthogonal/unitary matrix Q_1, and on exit this is overwritten by Q_1*Q.

LDQ (input) INTEGER
The leading dimension of the array Q.
LDQ $\geq$ max(1,N) if COMPQ = 'V' or 'I'; LDQ $\geq$ 1 otherwise.

Z (input/output) REAL/COMPLEX array, dimension (LDZ,N)
If COMPZ = 'N': Z is not referenced.
If COMPZ = 'I': on entry, Z need not be set, and on exit it contains the orthogonal/unitary matrix Z, which is the product of the Givens transformations which are applied to A and B on the right.
If COMPZ = 'V': on entry, Z must contain an orthogonal/unitary matrix Z_1, and on exit this is overwritten by Z_1*Z.

LDZ (input) INTEGER
The leading dimension of the array Z.
LDZ $\geq$ max(1,N) if COMPZ = 'V' or 'I'; LDZ $\geq$ 1 otherwise.

INFO (output) INTEGER
- = 0: successful exit
- < 0: if INFO = −i, the i^{th} argument had an illegal value.

SGGLSE/CGGLSE

```
      SUBROUTINE SGGLSE( M, N, P, A, LDA, B, LDB, C, D, X, WORK, LWORK,
     $                   INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, P
      REAL               A( LDA, * ), B( LDB, * ), C( * ), D( * ),
     $                   WORK( * ), X( * )

      SUBROUTINE CGGLSE( M, N, P, A, LDA, B, LDB, C, D, X, WORK, LWORK,
     $                   INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, P
      COMPLEX            A( LDA, * ), B( LDB, * ), C( * ), D( * ),
     $                   WORK( * ), X( * )
```

Purpose

SGGLSE/CGGLSE solves the linear equality-constrained least squares (LSE) problem:

min || c − A∗x ||₂ subject to B∗x = d

where A is an m-by-n matrix, B is a p-by-n matrix, c is a given m-vector, and d is a given p-vector. It is assumed that p ≤ n ≤ m+p, and

$$\text{rank}(B) = p, \quad \text{and} \quad \text{rank} \begin{pmatrix} A \\ B \end{pmatrix} = n.$$

These conditions ensure that the LSE problem has a unique solution, which is obtained using a GRQ factorization of the matrices B and A.

Arguments

M (input) INTEGER
The number of rows of the matrix A. M ≥ 0.

N (input) INTEGER
The number of columns of the matrices A and B. N ≥ 0.

P (input) INTEGER
The number of rows of the matrix B. 0 ≤ P ≤ N ≤ M+P.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit, A is destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,M).

B (input/output) REAL/COMPLEX array, dimension (LDB,N)
On entry, the p-by-n matrix B.
On exit, B is destroyed.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,P).

C (input/output) REAL/COMPLEX array, dimension (M)
On entry, C contains the right hand side vector for the least squares part of the LSE problem.
On exit, the residual sum of squares for the solution is given by the sum of squares of elements n−p+1 to m of vector C.

D (input/output) REAL/COMPLEX array, dimension (P)
On entry, D contains the right hand side vector for the constrained equation.
On exit, D is destroyed.

X (output) REAL/COMPLEX array, dimension (N)
On exit, X is the solution of the LSE problem.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK ≥ max(1,M+N+P).
For optimum performance LWORK ≥ P+min(M,N)+max(M,N)∗NB, where NB is an upper bound for the optimal blocksizes for SGEQRF/CGEQRF, SGERQF/CGERQF, SORMQR/CUNMQR, and SORMRQ/CUNMRQ.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGGQRF/CGGQRF

```
      SUBROUTINE SGGQRF( N, M, P, A, LDA, TAUA, B, LDB, TAUB, WORK,
     $                   LWORK, INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, P
      REAL               A( LDA, * ), B( LDB, * ), TAUA( * ), TAUB( * ),
     $                   WORK( * )

      SUBROUTINE CGGQRF( N, M, P, A, LDA, TAUA, B, LDB, TAUB, WORK,
     $                   LWORK, INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, P
      COMPLEX            A( LDA, * ), B( LDB, * ), TAUA( * ), TAUB( * ),
     $                   WORK( * )
```

Purpose

SGGQRF/CGGQRF computes a generalized QR factorization of an n-by-m matrix A and an n-by-p matrix B:

A = Q∗R, B = Q∗T∗Z,

where Q is an n-by-n orthogonal/unitary matrix, Z is a p-by-p orthogonal/unitary matrix, and R and T assume one of the forms:

$$\text{if} \quad n \geq m, \qquad R = \begin{array}{c} \\ m \\ n-m \end{array} \begin{array}{c} m \\ \begin{pmatrix} R_{11} \\ 0 \end{pmatrix} \end{array}$$

or

$$\text{if} \quad n < m, \qquad R = \begin{array}{c} \\ n \end{array} \begin{array}{cc} n & m-n \\ \begin{pmatrix} R_{11} & R_{12} \end{pmatrix} \end{array}$$

where R_{11} is upper triangular, and

$$\text{if} \quad n \le p, \qquad T = \begin{array}{c c} & \begin{array}{cc} p-n & n \end{array} \\ n & \begin{pmatrix} 0 & T_{12} \end{pmatrix} \end{array}$$

or

$$\text{if} \quad n > p, \qquad T = \begin{array}{c c} & p \\ \begin{array}{c} n-p \\ p \end{array} & \begin{pmatrix} T_{11} \\ T_{21} \end{pmatrix} \end{array}$$

where T_{12} or T_{21} is a p-by-p upper triangular matrix.

In particular, if B is square and nonsingular, the GQR factorization of A and B implicitly gives the QR factorization of $B^{-1}*A$:

$B^{-1}*A = Z^H*(T^{-1}*R)$.

Arguments

N (input) INTEGER
The number of rows of the matrices A and B. N $\geq$ 0.

M (input) INTEGER
The number of columns of the matrix A. M $\geq$ 0.

P (input) INTEGER
The number of columns of the matrix B. P $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,M)
On entry, the n-by-m matrix A.
On exit, the elements on and above the diagonal of the array contain the min(n,m)-by-m upper trapezoidal matrix R (R is upper triangular if N $\geq$ M); the elements below the diagonal, with the array TAUA, represent the orthogonal/unitary matrix Q as a product of min(N,M) elementary reflectors.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

TAUA (output) REAL/COMPLEX array, dimension (min(N,M))
The scalar factors of the elementary reflectors which represent the orthogonal/unitary matrix Q.

B (input/output) REAL/COMPLEX array, dimension (LDB,P)
On entry, the n-by-p matrix B.
On exit, if N $\leq$ P, the upper triangle of the subarray B(1:n,p−n+1:p) contains the n-by-n upper triangular matrix T;
if N > P, the elements on and above the $(n-p)^{th}$ subdiagonal contain the n-by-p upper trapezoidal matrix T; the remaining elements, with the array TAUB, represent the orthogonal/unitary matrix Z as a product of elementary reflectors.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

TAUB (output) REAL/COMPLEX array, dimension (min(N,P))
The scalar factors of the elementary reflectors which represent the orthogonal/unitary matrix Z.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension for the array WORK. LWORK $\geq$ max(1,N,M,P). For optimum performance LWORK $\geq$ max(1,N,M,P)*max(NB1,NB2,NB3), where NB1 is the optimal blocksize for the QR factorization of an n-by-m matrix, NB2 is the optimal blocksize for the RQ factorization of an n-by-p matrix, and NB3 is the optimal blocksize for a call of SORMQR/CUNMQR.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGGRQF/CGGRQF

```
      SUBROUTINE SGGRQF( M, P, N, A, LDA, TAUA, B, LDB, TAUB, WORK,
     $                   LWORK, INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, P
      REAL               A( LDA, * ), B( LDB, * ), TAUA( * ), TAUB( * ),
     $                   WORK( * )

      SUBROUTINE CGGRQF( M, P, N, A, LDA, TAUA, B, LDB, TAUB, WORK,
     $                   LWORK, INFO )
      INTEGER            INFO, LDA, LDB, LWORK, M, N, P
      COMPLEX            A( LDA, * ), B( LDB, * ), TAUA( * ), TAUB( * ),
     $                   WORK( * )
```

Purpose

SGGRQF/CGGRQF computes a generalized RQ factorization of an m-by-n matrix A and a p-by-n matrix B:

A = R*Q, B = Z*T*Q,

where Q is an n-by-n orthogonal/unitary matrix, and Z is a p-by-p orthogonal/unitary matrix, and R and T assume one of the forms:

$$\text{if} \quad m \le n, \qquad R = \begin{array}{c c} & \begin{array}{cc} n-m & m \end{array} \\ m & \begin{pmatrix} 0 & R_{12} \end{pmatrix} \end{array}$$

or

$$\text{if} \quad m > n, \qquad R = \begin{array}{cc} & n \\ m-n & \\ n & \end{array}\!\!\begin{pmatrix} R_{11} \\ R_{21} \end{pmatrix}$$

where R_{12} or R_{21} is upper triangular, and

$$\text{if} \quad p \ge n, \qquad T = \begin{array}{cc} & n \\ n & \\ p-n & \end{array}\!\!\begin{pmatrix} T_{11} \\ 0 \end{pmatrix}$$

or

$$\text{if} \quad p < n, \qquad T = \begin{array}{ccc} & p & n-p \\ p & & \end{array}\!\!\begin{pmatrix} T_{11} & T_{12} \end{pmatrix}$$

where T_{11} is upper triangular.

In particular, if B is square and nonsingular, the GRQ factorization of A and B implicitly gives the RQ factorization of A$*$B^{-1}:

A$*$B^{-1} = (R$*$T^{-1})$*$Z^H

Arguments

M	(input) INTEGER The number of rows of the matrix A. M $\ge$ 0.
P	(input) INTEGER The number of rows of the matrix B. P $\ge$ 0.
N	(input) INTEGER The number of columns of the matrices A and B. N $\ge$ 0.
A	(input/output) REAL/COMPLEX array, dimension (LDA,N) On entry, the m-by-n matrix A. On exit, if M $\le$ N, the upper triangle of the subarray A(1:m,n−m+1:n) contains the m-by-m upper triangular matrix R; if M > N, the elements on and above the (m−n)th subdiagonal contain the m-by-n upper trapezoidal matrix R; the remaining elements, with the array TAUA, represent the orthogonal/unitary matrix Q as a product of elementary reflectors.
LDA	(input) INTEGER The leading dimension of the array A. LDA $\ge$ max(1,M).
TAUA	(output) REAL/COMPLEX array, dimension (min(M,N)) The scalar factors of the elementary reflectors which represent the orthogonal/unitary matrix Q.
B	(input/output) REAL/COMPLEX array, dimension (LDB,N) On entry, the p-by-n matrix B. On exit, the elements on and above the diagonal of the array contain the min(p,n)-by-n upper trapezoidal matrix T (T is upper triangular if P $\ge$ N); the elements below the diagonal, with the array TAUB, represent the orthogonal/unitary matrix Z as a product of elementary reflectors.
LDB	(input) INTEGER The leading dimension of the array B. LDB $\ge$ max(1,P).
TAUB	(output) REAL/COMPLEX array, dimension (min(P,N)) The scalar factors of the elementary reflectors which represent the orthogonal/unitary matrix Z.
WORK	(workspace/output) REAL/COMPLEX array, dimension (LWORK) On exit, if INFO = 0, WORK(1) returns the optimal LWORK.
LWORK	(input) INTEGER The dimension of the array WORK. LWORK $\ge$ max(1,N,M,P). For optimum performance LWORK $\ge$ max(1,N,M,P)$*$max(NB1,NB2,NB3), where NB1 is the optimal blocksize for the RQ factorization of an m-by-n matrix, NB2 is the optimal blocksize for the QR factorization of the p-by-n matrix, and NB3 is the optimal blocksize for a call of SORMRQ/CUNMRQ. If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.
INFO	(output) INTEGER = 0: successful exit < 0: if INFO = −i, the ith argument had an illegal value.

SGGSVD/CGGSVD

```
      SUBROUTINE SGGSVD( JOBU, JOBV, JOBQ, M, N, P, K, L, A, LDA, B,
     $                   LDB, ALPHA, BETA, U, LDU, V, LDV, Q, LDQ,
     $                   WORK, IWORK, INFO )
      CHARACTER          JOBQ, JOBU, JOBV
      INTEGER            INFO, K, L, LDA, LDB, LDQ, LDU, LDV, M, N, P
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), ALPHA( * ), B( LDB, * ),
     $                   BETA( * ), Q( LDQ, * ), U( LDU, * ),
     $                   V( LDV, * ), WORK( * )

      SUBROUTINE CGGSVD( JOBU, JOBV, JOBQ, M, N, P, K, L, A, LDA, B,
     $                   LDB, ALPHA, BETA, U, LDU, V, LDV, Q, LDQ,
     $                   WORK, RWORK, IWORK, INFO )
      CHARACTER          JOBQ, JOBU, JOBV
      INTEGER            INFO, K, L, LDA, LDB, LDQ, LDU, LDV, M, N, P
      INTEGER            IWORK( * )
      REAL               ALPHA( * ), BETA( * ), RWORK( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), Q( LDQ, * ),
     $                   U( LDU, * ), V( LDV, * ), WORK( * )
```

Purpose

SGGSVD/CGGSVD computes the generalized singular value decomposition (GSVD) of an m-by-n real/complex matrix A and p-by-n real/complex matrix B:

$$U^H*A*Q = D_1*\begin{pmatrix} 0 & R \end{pmatrix}, \quad V^H*B*Q = D_2*\begin{pmatrix} 0 & R \end{pmatrix}$$

where U, V and Q are orthogonal/unitary matrices. Let k+l = the effective numerical rank of the matrix $(A^H,B^H)^H$, then R is a (k+l)-by-(k+l) nonsingular upper triangular matrix, D_1 and D_2 are m-by-(k+l) and p-by-(k+l) "diagonal" matrices and of the following structures, respectively:

If m−k−l ≥ 0,

$$D_1 = \begin{array}{r} \\ k \\ l \\ m-k-l \end{array} \begin{array}{c} \begin{array}{cc} k & l \end{array} \\ \begin{pmatrix} I & 0 \\ 0 & C \\ 0 & 0 \end{pmatrix} \end{array}$$

$$D_2 = \begin{array}{r} \\ l \\ p-l \end{array} \begin{array}{c} \begin{array}{cc} k & l \end{array} \\ \begin{pmatrix} 0 & S \\ 0 & 0 \end{pmatrix} \end{array}$$

$$\begin{pmatrix} 0 & R \end{pmatrix} = \begin{array}{r} \\ k \\ l \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & R_{11} & R_{12} \\ 0 & 0 & R_{22} \end{pmatrix} \end{array}$$

where

C = diag(ALPHA(k+1), ... , ALPHA(k+l)),
S = diag(BETA(k+1), ... , BETA(k+l)),
$C^2 + S^2 = I$;

R is stored in A(1:k+l, n−k−l+1:n) on exit.

If m−k−l < 0,

$$D_1 = \begin{array}{r} \\ k \\ m-k \end{array} \begin{array}{c} \begin{array}{ccc} k & m-k & k+l-m \end{array} \\ \begin{pmatrix} I & 0 & 0 \\ 0 & C & 0 \end{pmatrix} \end{array}$$

$$D_2 = \begin{array}{r} \\ m-k \\ k+l-m \\ p-l \end{array} \begin{array}{c} \begin{array}{ccc} k & m-k & k+l-m \end{array} \\ \begin{pmatrix} 0 & S & 0 \\ 0 & 0 & I \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

$$\begin{pmatrix} 0 & R \end{pmatrix} = \begin{array}{r} \\ k \\ m-k \\ k+l-m \end{array} \begin{array}{c} \begin{array}{cccc} n-k-l & k & m-k & k+l-m \end{array} \\ \begin{pmatrix} 0 & R_{11} & R_{12} & R_{13} \\ 0 & 0 & R_{22} & R_{23} \\ 0 & 0 & 0 & R_{33} \end{pmatrix} \end{array}$$

where

C = diag(ALPHA(k+1), ... , ALPHA(m)),
S = diag(BETA(k+1), ... , BETA(m)),
$C^2 + S^2 = I$;

$\begin{pmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \end{pmatrix}$ is stored in A(1:m, n−k−l+1:n), and R_{33} is stored in B(m−k+1:l, n+m−k−l+1:n) on exit.

The routine computes C, S, R, and optionally the orthogonal/unitary transformation matrices U, V and Q.

In particular, if B is an n-by-n nonsingular matrix, then the GSVD of A and B implicitly gives the SVD of $A*B^{-1}$:

$$A*B^{-1} = U*(D_1*D_2{}^{-1})*V^H.$$

If $(A^H,B^H)^H$ has orthonormal columns, then the GSVD of A and B is also equal to the CS decomposition of A and B. Furthermore, the GSVD can be used to derive the solution of the eigenvalue problem:

$$A^H * A * x = \lambda * B^H * B * x.$$

In some literature, the GSVD of A and B is presented in the form

$$U^H * A * X = \begin{pmatrix} 0 & D_1 \end{pmatrix},$$
$$V^H * B * X = \begin{pmatrix} 0 & D_2 \end{pmatrix}$$

where U and V are orthogonal/unitary and X is nonsingular, D_1 and D_2 are "diagonal". The former GSVD form can be converted to the latter form by taking the nonsingular matrix X as

$$X = Q * \begin{pmatrix} I & 0 \\ 0 & R^{-1} \end{pmatrix}.$$

Arguments

JOBU (input) CHARACTER*1
= 'U': Orthogonal/Unitary matrix U is computed;
= 'N': U is not computed.

JOBV (input) CHARACTER*1
= 'V': Orthogonal/Unitary matrix V is computed;
= 'N': V is not computed.

JOBQ (input) CHARACTER*1
= 'Q': Orthogonal/Unitary matrix Q is computed;
= 'N': Q is not computed.

M (input) INTEGER
The number of rows of the matrix A. M ≥ 0.

N (input) INTEGER
The number of columns of the matrices A and B. $N \geq 0$.

P (input) INTEGER
The number of rows of the matrix B. $P \geq 0$.

K (output) INTEGER

L (output) INTEGER
On exit, K and L specify the dimension of the subblocks.
$K + L$ = effective numerical rank of $(A^H, B^H)^H$.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit, A contains the triangular matrix R, or part of R.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

B (input/output) REAL/COMPLEX array, dimension (LDB,N)
On entry, the p-by-n matrix B.
On exit, B contains part of the triangular matrix R if $M-K-L < 0$.

LDB (input) INTEGER
The leading dimension of the array B. LDA/LDB $\geq$ max(1,P).

ALPHA (output) REAL array, dimension (N)

BETA (output) REAL array, dimension (N)
On exit, ALPHA and BETA contain the generalized singular value pairs of A and B;
ALPHA(1:K) = 1,
BETA(1:K) = 0,
and if $M-K-L \geq 0$,
ALPHA(K+1:K+L) = C,
BETA(K+1:K+L) = S,
or if $M-K-L < 0$,
ALPHA(K+1:M) = C, ALPHA(M+1:K+L) = 0
BETA(K+1:M) = S, BETA(M+1:K+L) = 1
and
ALPHA(K+L+1:N) = 0
BETA(K+L+1:N) = 0

U (output) REAL/COMPLEX array, dimension (LDU,M)
If JOBU = 'U', U contains the m-by-m orthogonal/unitary matrix U.
If JOBU = 'N', U is not referenced.

LDU (input) INTEGER
The leading dimension of the array U.
LDU $\geq$ max(1,M) if JOBU = 'U'; LDU $\geq$ 1 otherwise.

V (output) REAL/COMPLEX array, dimension (LDV,P)
If JOBV = 'V', V contains the p-by-p orthogonal/unitary matrix V.
If JOBV = 'N', V is not referenced.

LDV (input) INTEGER
The leading dimension of the array V.
LDV $\geq$ max(1,P) if JOBV = 'V'; LDV $\geq$ 1 otherwise.

Q (output) REAL/COMPLEX array, dimension (LDQ,N)
If JOBQ = 'Q', Q contains the n-by-n orthogonal/unitary matrix Q.
If JOBQ = 'N', Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q. LDQ $\geq$ max(1,N) if JOBQ = 'Q';
LDQ $\geq$ 1 otherwise.

WORK (workspace) REAL/COMPLEX array,
dimension (max(3*N,M,P)+N)

IWORK (workspace/output) INTEGER array, dimension (N)
On exit, IWORK stores the sorting information. More precisely, the following loop will sort ALPHA

```
   for I = K+1, min(M,K+L)
      swap ALPHA(I) and ALPHA(IWORK(I))
   endfor
```

such that ALPHA(1) $\geq$ ALPHA(2) $\geq \ldots \geq$ ALPHA(N).

RWORK *CGGSVD only* (workspace) REAL array, dimension (2*N)

INFO (output)INTEGER
= 0: successful exit
< 0: if INFO = $-i$, the i^{th} argument had an illegal value.
> 0: if INFO = 1, the Jacobi-type procedure failed to converge. For further details, see subroutine STGSJA/CTGSJA.

SGGSVP/CGGSVP

```
      SUBROUTINE SGGSVP( JOBU, JOBV, JOBQ, M, P, N, A, LDA, B, LDB,
     $                   TOLA, TOLB, K, L, U, LDU, V, LDV, Q, LDQ,
     $                   IWORK, TAU, WORK, INFO )
      CHARACTER          JOBQ, JOBU, JOBV
      INTEGER            INFO, K, L, LDA, LDB, LDQ, LDU, LDV, M, N, P
      REAL               TOLA, TOLB
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), B( LDB, * ), Q( LDQ, * ),
     $                   TAU( * ), U( LDU, * ), V( LDV, * ), WORK( * )
```

```
      SUBROUTINE CGGSVP( JOBU, JOBV, JOBQ, M, P, N, A, LDA, B, LDB,
     $                   TOLA, TOLB, K, L, U, LDU, V, LDV, Q, LDQ,
     $                   IWORK, RWORK, TAU, WORK, INFO )
      CHARACTER          JOBQ, JOBU, JOBV
      INTEGER            INFO, K, L, LDA, LDB, LDQ, LDU, LDV, M, N, P
      REAL               TOLA, TOLB
      INTEGER            IWORK( * )
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), Q( LDQ, * ),
     $                   TAU( * ), U( LDU, * ), V( LDV, * ), WORK( * )
```

Purpose

SGGSVP/CGGSVP computes orthogonal/unitary matrices U, V and Q such that

$$U^H * A * Q = \begin{array}{c} \\ k \\ l \\ m-k-l \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & A_{12} & A_{13} \\ 0 & 0 & A_{23} \\ 0 & 0 & 0 \end{pmatrix} \end{array} \text{ if } m-k-l \geq 0;$$

$$= \begin{array}{c} \\ k \\ m-k \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & A_{12} & A_{13} \\ 0 & 0 & A_{23} \end{pmatrix} \end{array} \text{ if } m-k-l < 0;$$

$$V^H * B * Q = \begin{array}{c} \\ l \\ p-l \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & 0 & B_{13} \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

where the k-by-k matrix A_{12} and l-by-l matrix B_{13} are nonsingular upper triangular; A_{23} is l-by-l upper triangular if m−k−l ≥ 0, otherwise A_{23} is (m−k)-by-l upper trapezoidal. k+l = the effective numerical rank of the (m+p)-by-n matrix $(A^H, B^H)^H$.

This decomposition is the preprocessing step for computing the Generalized Singular Value Decomposition (GSVD), see subroutine SGGSVD/CGGSVD.

Arguments

JOBU (input) CHARACTER*1
= 'U': Orthogonal/Unitary matrix U is computed;
= 'N': U is not computed.

JOBV (input) CHARACTER*1
= 'V': Orthogonal/Unitary matrix V is computed;
= 'N': V is not computed.

JOBQ (input) CHARACTER*1
= 'Q': Orthogonal/Unitary matrix Q is computed;
= 'N': Q is not computed.

M (input) INTEGER
The number of rows of the matrix A. M ≥ 0.

P (input) INTEGER
The number of rows of the matrix B. P ≥ 0.

N (input) INTEGER
The number of columns of the matrices A and B. N ≥ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit, A contains the triangular (or trapezoidal) matrix described in the Purpose section.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,M).

B (input/output) REAL/COMPLEX array, dimension (LDB,N)
On entry, the p-by-n matrix B.
On exit, B contains the triangular matrix described in the Purpose section.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,P).

TOLA (input) REAL

TOLB (input) REAL
TOLA and TOLB are the thresholds to determine the effective numerical rank of matrix B and a subblock of A. Generally, they are set to
TOLA = max(M,N)*$\|A\|$*MACHEPS,
TOLB = max(P,N)*$\|B\|$*MACHEPS.
The size of TOLA and TOLB may affect the size of backward errors of the decomposition.

K, L (output) INTEGER
On exit, K and L specify the dimension of the subblocks.
K+L = effective numerical rank of $(A^H, B^H)^H$.

U (output) REAL/COMPLEX array, dimension (LDU,M)
If JOBU = 'U', U contains the orthogonal/unitary matrix U.
If JOBU = 'N', U is not referenced.

LDU (input) INTEGER
The leading dimension of the array U.
LDU ≥ max(1,M) if JOBU = 'U'; LDU ≥ 1 otherwise.

V (output) REAL/COMPLEX array, dimension (LDV,M)
If JOBV = 'V', V contains the orthogonal/unitary matrix V.
If JOBV = 'N', V is not referenced.

LDV (input) INTEGER
The leading dimension of the array V.
LDV ≥ max(1,P) if JOBV = 'V'; LDV ≥ 1 otherwise.

Q (output) REAL/COMPLEX array, dimension (LDQ,N)
If JOBQ = 'Q', Q contains the orthogonal/unitary matrix Q.
If JOBQ = 'N', Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q. LDQ $\geq$ max(1,N). LDQ $\geq$ max(1,N) if JOBQ = 'Q'; LDQ $\geq$ 1 otherwise.

IWORK (workspace) INTEGER array, dimension (N)

RWORK *CGGSVP only* (workspace) REAL array, dimension (2*N)

TAU (workspace) REAL/COMPLEX array, dimension (N)

WORK (workspace) REAL/COMPLEX array, dimension (max(3*N,M,P))

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGTCON/CGTCON

```
      SUBROUTINE SGTCON( NORM, N, DL, D, DU, DU2, IPIV, ANORM, RCOND,
     $                   WORK, IWORK, INFO )
      CHARACTER          NORM
      INTEGER            INFO, N
      REAL               ANORM, RCOND
      INTEGER            IPIV( * ), IWORK( * )
      REAL               D( * ), DL( * ), DU( * ), DU2( * ), WORK( * )

      SUBROUTINE CGTCON( NORM, N, DL, D, DU, DU2, IPIV, ANORM, RCOND,
     $                   WORK, INFO )
      CHARACTER          NORM
      INTEGER            INFO, N
      REAL               ANORM, RCOND
      INTEGER            IPIV( * )
      COMPLEX            D( * ), DL( * ), DU( * ), DU2( * ), WORK( * )
```

Purpose

SGTCON/CGTCON estimates the reciprocal of the condition number of a real/complex tridiagonal matrix A using the LU factorization as computed by SGTTRF/CGTTRF.

An estimate is obtained for $||A^{-1}||$, and the reciprocal of the condition number is computed as RCOND = $1/(||A|| * ||A^{-1}||)$.

Arguments

NORM (input) CHARACTER*1
Specifies whether the 1-norm condition number or the infinity-norm condition number is required:
= '1' or 'O': 1-norm;
= 'I': Infinity-norm.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

DL (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) multipliers that define the matrix L from the LU factorization of A as computed by SGTTRF/CGTTRF.

D (input) REAL/COMPLEX array, dimension (N)
The n diagonal elements of the upper triangular matrix U from the LU factorization of A.

DU (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) elements of the first superdiagonal of U.

DU2 (input) REAL/COMPLEX array, dimension (N−2)
The (n−2) elements of the second superdiagonal of U.

IPIV (input) INTEGER array, dimension (N)
The pivot indices; for $1 \leq i \leq n$, row i of the matrix was interchanged with row IPIV(i). IPIV(i) will always be either i or i+1; IPIV(i) = i indicates a row interchange was not required.

ANORM (input) REAL
If NORM = '1' or 'O', the 1-norm of the original matrix A.
If NORM = 'I', the infinity-norm of the original matrix A.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as RCOND = $1/(||A|| * ||A^{-1}||)$.

WORK (workspace) REAL/COMPLEX array, dimension (2*N)

IWORK *SGTCON only* (workspace) INTEGER array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGTRFS/CGTRFS

```
      SUBROUTINE SGTRFS( TRANS, N, NRHS, DL, D, DU, DLF, DF, DUF, DU2,
     $                   IPIV, B, LDB, X, LDX, FERR, BERR, WORK, IWORK,
     $                   INFO )
      CHARACTER          TRANS
      INTEGER            INFO, LDB, LDX, N, NRHS
      INTEGER            IPIV( * ), IWORK( * )
      REAL               B( LDB, * ), BERR( * ), D( * ), DF( * ),
     $                   DL( * ), DLF( * ), DU( * ), DU2( * ), DUF( * ),
     $                   FERR( * ), WORK( * ), X( LDX, * )
```

```
      SUBROUTINE CGTRFS( TRANS, N, NRHS, DL, D, DU, DLF, DF, DUF, DU2,
     $                   IPIV, B, LDB, X, LDX, FERR, BERR, WORK, RWORK,
     $                   INFO )
      CHARACTER          TRANS
      INTEGER            INFO, LDB, LDX, N, NRHS
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            B( LDB, * ), D( * ), DF( * ), DL( * ),
     $                   DLF( * ), DU( * ), DU2( * ), DUF( * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SGTRFS/CGTRFS improves the computed solution to a system of linear equations when the coefficient matrix is tridiagonal, and provides error bounds and backward error estimates for the solution.

Arguments

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)
= 'C': A^H*X = B (Conjugate transpose)

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

DL (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) subdiagonal elements of A.

D (input) REAL/COMPLEX array, dimension (N)
The diagonal elements of A.

DU (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) superdiagonal elements of A.

DLF (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) multipliers that define the matrix L from the LU factorization of A as computed by SGTTRF/CGTTRF.

DF (input) REAL/COMPLEX array, dimension (N)
The n diagonal elements of the upper triangular matrix U from the LU factorization of A.

DUF (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) elements of the first superdiagonal of U.

DU2 (input) REAL/COMPLEX array, dimension (N−2)
The (n−2) elements of the second superdiagonal of U.

IPIV (input) INTEGER array, dimension (N)
The pivot indices; for 1 $\leq$ i $\leq$ n, row i of the matrix was interchanged with row IPIV(i). IPIV(i) will always be either i or i+1; IPIV(i) = i indicates a row interchange was not required.

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (input/output) REAL/COMPLEX array, dimension (LDX,NRHS)
On entry, the solution matrix X, as computed by SGTTRS/CGTTRS. On exit, the improved solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SGTRFS* (workspace) REAL array, dimension (3*N)
CGTRFS (workspace) COMPLEX array, dimension (2*N)

IWORK *SGTRFS only* (workspace) INTEGER array, dimension (N)

RWORK *CGTRFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SGTSV/CGTSV

```
      SUBROUTINE SGTSV( N, NRHS, DL, D, DU, B, LDB, INFO )
      INTEGER            INFO, LDB, N, NRHS
      REAL               B( LDB, * ), D( * ), DL( * ), DU( * )

      SUBROUTINE CGTSV( N, NRHS, DL, D, DU, B, LDB, INFO )
      INTEGER            INFO, LDB, N, NRHS
      COMPLEX            B( LDB, * ), D( * ), DL( * ), DU( * )
```

Purpose

SGTSV/CGTSV solves the equation A*X = B, where A is an n-by-n tridiagonal matrix, by Gaussian elimination with partial pivoting.

Note that the equation A^T*X = B may be solved by interchanging the order of the arguments DU and DL.

Arguments

N (input) INTEGER
The order of the matrix A. N ≥ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS ≥ 0.

DL (input/output) REAL/COMPLEX array, dimension (N−1)
On entry, DL must contain the (n−1) subdiagonal elements of A.
On exit, DL is overwritten by the (n−2) elements of the second superdiagonal of the upper triangular matrix U from the LU factorization of A, in DL(1), ..., DL(n−2).

D (input/output) REAL/COMPLEX array, dimension (N)
On entry, D must contain the diagonal elements of A.
On exit, D is overwritten by the n diagonal elements of U.

DU (input/output) REAL/COMPLEX array, dimension (N−1)
On entry, DU must contain the (n−1) superdiagonal elements of A.
On exit, DU is overwritten by the (n−1) elements of the first superdiagonal of U.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if INFO = 0, the n-by-nrhs solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, U(i,i) is exactly zero, and the solution has not been computed. The factorization has not been completed unless i = N.

SGTSVX/CGTSVX

```
      SUBROUTINE SGTSVX( FACT, TRANS, N, NRHS, DL, D, DU, DLF, DF, DUF,
     $                   DU2, IPIV, B, LDB, X, LDX, RCOND, FERR, BERR,
     $                   WORK, IWORK, INFO )
      CHARACTER          FACT, TRANS
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * ), IWORK( * )
      REAL               B( LDB, * ), BERR( * ), D( * ), DF( * ),
     $                   DL( * ), DLF( * ), DU( * ), DU2( * ), DUF( * ),
     $                   FERR( * ), WORK( * ), X( LDX, * )

      SUBROUTINE CGTSVX( FACT, TRANS, N, NRHS, DL, D, DU, DLF, DF, DUF,
     $                   DU2, IPIV, B, LDB, X, LDX, RCOND, FERR, BERR,
     $                   WORK, RWORK, INFO )
      CHARACTER          FACT, TRANS
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            B( LDB, * ), D( * ), DF( * ), DL( * ),
     $                   DLF( * ), DU( * ), DU2( * ), DUF( * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SGTSVX/CGTSVX uses the LU factorization to compute the solution to a real/complex system of linear equations A*X = B, A^T*X = B, or A^H*X = B, where A is a tridiagonal matrix of order n and X and B are n-by-nrhs matrices.

Error bounds on the solution and a condition estimate are also provided.

Description

The following steps are performed:

1. If FACT = 'N', the LU decomposition is used to factor the matrix A as A = L*U, where L is a product of permutation and unit lower bidiagonal matrices and U is upper triangular with nonzeros in only the main diagonal and first two superdiagonals.
2. If some U(i,i)=0, so that U is exactly singular, then the routine returns with INFO = i. Otherwise, the factored form of A is used to estimate the condition number of the matrix A. If the reciprocal of the condition number is less than machine precision, INFO = N+1 is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.
3. The system of equations is solved for X using the factored form of A.
4. Iterative refinement is applied to improve the computed solution matrix and calculate error bounds and backward error estimates for it.

Arguments

FACT (input) CHARACTER*1
Specifies whether or not the factored form of A has been supplied on entry.
= 'F': DLF, DF, DUF, DU2, and IPIV contain the factored form of A; DL, D, DU, DLF, DF, DUF, DU2 and IPIV will not be modified.
= 'N': The matrix will be copied to DLF, DF, and DUF and factored.

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)
= 'C': A^H*X = B (Conjugate transpose)

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

DL (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) subdiagonal elements of A.

D (input) REAL/COMPLEX array, dimension (N)
The n diagonal elements of A.

DU (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) superdiagonal elements of A.

DLF (input or output) REAL/COMPLEX array, dimension (N−1)
If FACT = 'F', then DLF is an input argument and on entry contains the (n−1) multipliers that define the matrix L from the LU factorization of A as computed by SGTTRF/CGTTRF.
If FACT = 'N', then DLF is an output argument and on exit contains the (n−1) multipliers that define the matrix L from the LU factorization of A.

DF (input or output) REAL/COMPLEX array, dimension (N)
If FACT = 'F', then DF is an input argument and on entry contains the n diagonal elements of the upper triangular matrix U from the LU factorization of A.
If FACT = 'N', then DF is an output argument and on exit contains the n diagonal elements of the upper triangular matrix U from the LU factorization of A.

DUF (input or output) REAL/COMPLEX array, dimension (N−1)
If FACT = 'F', then DUF is an input argument and on entry contains the (n−1) elements of the first superdiagonal of U.
If FACT = 'N', then DUF is an output argument and on exit contains the (n−1) elements of the first superdiagonal of U.

DU2 (input or output) REAL/COMPLEX array, dimension (N−2)
If FACT = 'F', then DU2 is an input argument and on entry contains the (n−2) elements of the second superdiagonal of U.
If FACT = 'N', then DU2 is an output argument and on exit contains the (n−2) elements of the second superdiagonal of U.

IPIV (input or output) INTEGER array, dimension (N)
If FACT = 'F', then IPIV is an input argument and on entry contains the pivot indices from the LU factorization of A as computed by SGTTRF/CGTTRF.
If FACT = 'N', then IPIV is an output argument and on exit contains the pivot indices from the LU factorization of A; row i of the matrix was interchanged with row IPIV(i). IPIV(i) will always be either i or i+1; IPIV(i) = i indicates a row interchange was not required.

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The n-by-nrhs right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (output) REAL/COMPLEX array, dimension (LDX,NRHS)
If INFO = 0 or INFO = N+1, the N-by-NRHS solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

RCOND (output) REAL
The estimate of the reciprocal condition number of the matrix A. If RCOND is less than the machine precision (in particular, if RCOND = 0), the matrix is singular to working precision. This condition is indicated by a return code of INFO > 0.

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SGTSVX* (workspace) REAL array, dimension (3*N)
CGTSVX (workspace) COMPLEX array, dimension (2*N)

IWORK *SGTSVX only* (workspace) INTEGER array, dimension (N)

RWORK *CGTSVX only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER

= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
  ≤ N: U(i,i) is exactly zero. The factorization has not been completed unless i = N, but the factor U is exactly singular, so the solution and error bounds could not be computed. RCOND = 0 is returned.
  = N+1: U is nonsingular, but RCOND is less than machine precision, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of RCOND would suggest.

SGTTRF/CGTTRF

```
      SUBROUTINE SGTTRF( N, DL, D, DU, DU2, IPIV, INFO )
      INTEGER            INFO, N
      INTEGER            IPIV( * )
      REAL               D( * ), DL( * ), DU( * ), DU2( * )

      SUBROUTINE CGTTRF( N, DL, D, DU, DU2, IPIV, INFO )
      INTEGER            INFO, N
      INTEGER            IPIV( * )
      COMPLEX            D( * ), DL( * ), DU( * ), DU2( * )
```

Purpose

SGTTRF/CGTTRF computes an LU factorization of a real/complex tridiagonal matrix A using elimination with partial pivoting and row interchanges.

The factorization has the form A = L*U where L is a product of permutation and unit lower bidiagonal matrices and U is upper triangular with nonzeros in only the main diagonal and first two superdiagonals.

Arguments

N (input) INTEGER
The order of the matrix A. N ≥ 0.

DL (input/output) REAL/COMPLEX array, dimension (N−1)
On entry, DL must contain the (n−1) subdiagonal elements of A.
On exit, DL is overwritten by the (n−1) multipliers that define the matrix L from the LU factorization of A.

D (input/output) REAL/COMPLEX array, dimension (N)
On entry, D must contain the diagonal elements of A.
On exit, D is overwritten by the n diagonal elements of the upper triangular matrix U from the LU factorization of A.

DU (input/output) REAL/COMPLEX array, dimension (N−1)
On entry, DU must contain the (n−1) superdiagonal elements of A.
On exit, DU is overwritten by the (n−1) elements of the first superdiagonal of U.

DU2 (output) REAL/COMPLEX array, dimension (N−2)
On exit, DU2 is overwritten by the (n−2) elements of the second superdiagonal of U.

IPIV (output) INTEGER array, dimension (N)
The pivot indices; for 1 ≤ i ≤ n, row i of the matrix was interchanged with row IPIV(i). IPIV(i) will always be either i or i+1; IPIV(i) = i indicates a row interchange was not required.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, U(i,i) is exactly zero. The factorization has been completed, but the factor U is exactly singular, and division by zero will occur if it is used to solve a system of equations.

SGTTRS/CGTTRS

```
      SUBROUTINE SGTTRS( TRANS, N, NRHS, DL, D, DU, DU2, IPIV, B, LDB,
     $                   INFO )
      CHARACTER          TRANS
      INTEGER            INFO, LDB, N, NRHS
      INTEGER            IPIV( * )
      REAL               B( LDB, * ), D( * ), DL( * ), DU( * ), DU2( * )

      SUBROUTINE CGTTRS( TRANS, N, NRHS, DL, D, DU, DU2, IPIV, B, LDB,
     $                   INFO )
      CHARACTER          TRANS
      INTEGER            INFO, LDB, N, NRHS
      INTEGER            IPIV( * )
      COMPLEX            B( LDB, * ), D( * ), DL( * ), DU( * ), DU2( * )
```

Purpose

SGTTRS/CGTTRS solves one of the systems of equations A*X = B, A^T*X = B, or A^H*X = B, with a tridiagonal matrix A using the LU factorization computed by SGTTRF/CGTTRF.

Arguments

TRANS (input) CHARACTER
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)

= 'C': $A^H * X = B$ (Conjugate transpose)

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. $NRHS \geq 0$.

DL (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) multipliers that define the matrix L from the LU factorization of A.

D (input) REAL/COMPLEX array, dimension (N)
The n diagonal elements of the upper triangular matrix U from the LU factorization of A.

DU (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) elements of the first superdiagonal of U.

DU2 (input) REAL/COMPLEX array, dimension (N−2)
The (n−2) elements of the second superdiagonal of U.

IPIV (input) INTEGER array, dimension (N)
The pivot indices; for $1 \leq i \leq n$, row i of the matrix was interchanged with row IPIV(i). IPIV(i) will always be either i or i+1; IPIV(i) = i indicates a row interchange was not required.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, B is overwritten by the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. $LDB \geq \max(1,N)$.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SHGEQZ/CHGEQZ

```
      SUBROUTINE SHGEQZ( JOB, COMPQ, COMPZ, N, ILO, IHI, A, LDA, B, LDB,
     $                   ALPHAR, ALPHAI, BETA, Q, LDQ, Z, LDZ, WORK,
     $                   LWORK, INFO )
      CHARACTER          COMPQ, COMPZ, JOB
      INTEGER            IHI, ILO, INFO, LDA, LDB, LDQ, LDZ, LWORK, N
      REAL               A( LDA, * ), ALPHAI( * ), ALPHAR( * ),
     $                   B( LDB, * ), BETA( * ), Q( LDQ, * ), WORK( * ),
     $                   Z( LDZ, * )
```

```
      SUBROUTINE CHGEQZ( JOB, COMPQ, COMPZ, N, ILO, IHI, A, LDA, B, LDB,
     $                   ALPHA, BETA, Q, LDQ, Z, LDZ, WORK, LWORK,
     $                   RWORK, INFO )
      CHARACTER          COMPQ, COMPZ, JOB
      INTEGER            IHI, ILO, INFO, LDA, LDB, LDQ, LDZ, LWORK, N
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), ALPHA( * ), B( LDB, * ),
     $                   BETA( * ), Q( LDQ, * ), WORK( * ), Z( LDZ, * )
```

Purpose

SHGEQZ/CHGEQZ implements a single-/double-shift version (*SHGEQZ*) or single-shift version (*CHGEQZ*) of the QZ method for finding the generalized eigenvalues w(j) = (ALPHAR(j) + i*ALPHAI(j))/BETAR(j) (*SHGEQZ*) or w(i) = ALPHA(i)/BETA(i) (*CHGEQZ*) of the equation

$$\det(A - w(i)*B) = 0$$

SHGEQZ only
In addition, the pair A,B may be reduced to generalized Schur form: B is upper triangular, and A is block upper triangular, where the diagonal blocks are either 1-by-1 or 2-by-2, the 2-by-2 blocks having complex generalized eigenvalues (see the description of the argument JOB.)

If JOB='S', then the pair (A,B) is simultaneously reduced to Schur form by applying one orthogonal tranformation (usually called Q) on the left and another (usually called Z) on the right. The 2-by-2 upper-triangular diagonal blocks of B corresponding to 2-by-2 blocks of A will be reduced to positive diagonal matrices. (I.e., if A(j+1,j) is non-zero, then B(j+1,j) = B(j,j+1) = 0 and B(j,j) and B(j+1,j+1) will be positive.)

If JOB='E', then at each iteration, the same transformations are computed, but they are only applied to those parts of A and B which are needed to compute ALPHAR, ALPHAI, and BETAR.

CHGEQZ only
If JOB='S', then the pair (A,B) is simultaneously reduced to Schur form (i.e., A and B are both upper triangular) by applying one unitary tranformation (usually called Q) on the left and another (usually called Z) on the right. The diagonal elements of A are then ALPHA(1),...,ALPHA(n), and of B are BETA(1),...,BETA(n).

SHGEQZ and CHGEQZ
If JOB='S' and COMPQ and COMPZ are 'V' or 'I', then the orthogonal/unitary transformations used to reduce (A,B) are accumulated into the arrays Q and Z such that:

$$Q(in)*A(in)*Z(in)^H = Q(out)*A(out)*Z(out)^H$$

$$Q(in)*B(in)*Z(in)^H = Q(out)*B(out)*Z(out)^H$$

Arguments

JOB (input) CHARACTER*1
- = 'E': compute only ALPHAR, ALPHAI (*SHGEQZ*) or ALPHA (*CHGEQZ*) and BETA. A and B will not necessarily be put into generalized Schur form.
- = 'S': put A and B into generalized Schur form, as well as computing ALPHAR, ALPHAI (*SHGEQZ*) or ALPHA (*CHGEQZ*), and BETA.

COMPQ (input) CHARACTER*1
- = 'N': do not modify Q.
- = 'V': multiply the array Q on the right by the transpose/conjugate-transpose of the orthogonal/unitary tranformation that is aplied to the left side of A and B to reduce them to Schur form.
- = 'I': like COMPQ='V', except that Q will be initialized to the identity first.

COMPZ (input) CHARACTER*1
- = 'N': do not modify Z.
- = 'V': multiply the array Z on the right by the orthogonal/unitary tranformation that is applied to the right side of A and B to reduce them to Schur form.
- = 'I': like COMPZ='V', except that Z will be initialized to the identity first.

N (input) INTEGER
The order of the matrices A, B, Q, and Z. $N \geq 0$.

ILO, IHI (input) INTEGER
It is assumed that A is already upper triangular in rows and columns 1:ilo−1 and ihi+1:n.
$1 \leq ILO \leq IHI \leq N$, if $N > 0$; ILO = 1 and IHI = 0, if N = 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the n-by-n upper Hessenberg matrix A. Elements below the subdiagonal must be zero.
If JOB='S', then on exit A and B will have been simultaneously reduced to generalized Schur form.
If JOB='E', then on exit A will have been destroyed.
SHGEQZ
The diagonal blocks will be correct, but the off-diagonal portion will be meaningless.

LDA (input) INTEGER
The leading dimension of the array A. $LDA \geq \max(1,N)$.

B (input/output) REAL/COMPLEX array, dimension (LDB,N)
On entry, the n-by-n upper triangular matrix B. Elements below the diagonal must be zero.
SHGEQZ
2-by-2 blocks in B corresponding to 2-by-2 blocks in A will be reduced to positive diagonal form. (I.e., if A(j+1,j) is non-zero, then B(j+1,j) = B(j,j+1) = 0 and B(j,j) and B(j+1,j+1) will be positive.)
SHGEQZ and CHGEQZ
If JOB='S', then on exit A and B will have been simultaneously reduced to generalized Schur form.
If JOB='E', then on exit B will have been destroyed.
SHGEQZ
Elements corresponding to diagonal blocks of A will be correct, but the off-diagonal portion will be meaningless.

LDB (input) INTEGER
The leading dimension of the array B. $LDB \geq \max(1,N)$.

ALPHAR *SHGEQZ* (output) REAL array, dimension (N)
ALPHAR(1:n) will be set to real parts of the diagonal elements of A that would result from reducing A and B to Schur form and then further reducing them both to triangular form using unitary transformations s.t. the diagonal of B was non-negative real. Thus, if A(j,j) is in a 1-by-1 block (i.e., A(j+1,j)=A(j,j+1)=0), then ALPHAR(j)=A(j,j).
Note that the (real or complex) values
(ALPHAR(j) + i*ALPHAI(j))/BETA(j), j=1,...,n, are the generalized eigenvalues of the matrix pencil A − w*B.

ALPHAI *SHGEQZ* (output) REAL array, dimension (N)
ALPHAI(1:n) will be set to imaginary parts of the diagonal elements of A that would result from reducing A and B to Schur form and then further reducing them both to triangular form using unitary transformations s.t. the diagonal of B was non-negative real. Thus, if A(j,j) is in a 1-by-1 block (i.e., A(j+1,j)=A(j,j+1)=0), then ALPHAR(j)=0.
Note that the (real or complex) values
(ALPHAR(j) + i*ALPHAI(j))/BETA(j), j=1,...,n, are the generalized eigenvalues of the matrix pencil A − w*B.

ALPHA *CHGEQZ* (output) COMPLEX array, dimension (N)
The diagonal elements of A when the pair (A,B) has been reduced to Schur form. ALPHA(i)/BETA(i) i=1,...,n are the generalized eigenvalues.

BETA (output) REAL/COMPLEX array, dimension (N)
SHGEQZ
BETA(1:n) will be set to the (real) diagonal elements of B that would result from reducing A and B to Schur form and then further reducing them both to triangular form using unitary transformations s.t. the diagonal of B was non-negative real. Thus, if A(j,j) is in a 1-by-1 block (i.e., A(j+1,j)=A(j,j+1)=0), then BETA(j)=B(j,j). Note that the (real or complex) values (ALPHAR(j) + i*ALPHAI(j))/BETA(j), j=1,...,n, are the generalized eigenvalues of the matrix pencil A − w*B. (Note that BETA(1:n) will always be non-negative, and no BETAI is necessary.)
CHGEQZ
The diagonal elements of B when the pair (A,B) has been reduced to

Schur form. ALPHA(i)/BETA(i) i=1,...,n are the generalized eigenvalues. A and B are normalized so that BETA(1),...,BETA(n) are nonnegative real numbers.

Q (input/output) REAL/COMPLEX array, dimension (LDQ,N)
If COMPQ='N', then Q will not be referenced.
If COMPQ='V' or 'I', then the transpose/conjugate-transpose of the orthogonal/unitary transformations which are applied to A and B on the left will be applied to the array Q on the right.

LDQ (input) INTEGER
The leading dimension of the array Q. LDQ $\geq$ 1.
If COMPQ='V' or 'I', then LDQ $\geq$ N.

Z (input/output) REAL/COMPLEX array, dimension (LDZ,N)
If COMPZ='N', then Z will not be referenced.
If COMPZ='V' or 'I', then the orthogonal/unitary transformations which are applied to A and B on the right will be applied to the array Z on the right.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1.
If COMPZ='V' or 'I', then LDZ $\geq$ N.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO $\geq$ 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ max(1,N).

If LWORK = -1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CHGEQZ only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = $-$i, the i^{th} argument had an illegal value.
= 1,...,N: the QZ iteration did not converge. (A,B) is not in Schur form, but ALPHAR(i), ALPHAI(i) (*SHGEQZ*) or ALPHA(i) (*CHGEQZ*), and BETA(i), i=info+1,...,n should be correct.
= N+1,...,2*N: the shift calculation failed. (A,B) is not in Schur form, but ALPHAR(i), ALPHAI(i) (*SHGEQZ*), or ALPHA(i) (*CHGEQZ*) and BETA(i), i=info−n+1,...,N should be correct.
> 2*N: various "impossible" errors.

SHSEIN/CHSEIN

```
      SUBROUTINE SHSEIN( SIDE, EIGSRC, INITV, SELECT, N, H, LDH, WR, WI,
     $                   VL, LDVL, VR, LDVR, MM, M, WORK, IFAILL,
     $                   IFAILR, INFO )
      CHARACTER          EIGSRC, INITV, SIDE
      INTEGER            INFO, LDH, LDVL, LDVR, M, MM, N
      LOGICAL            SELECT( * )
      INTEGER            IFAILL( * ), IFAILR( * )
      REAL               H( LDH, * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   WI( * ), WORK( * ), WR( * )

      SUBROUTINE CHSEIN( SIDE, EIGSRC, INITV, SELECT, N, H, LDH, W, VL,
     $                   LDVL, VR, LDVR, MM, M, WORK, RWORK, IFAILL,
     $                   IFAILR, INFO )
      CHARACTER          EIGSRC, INITV, SIDE
      INTEGER            INFO, LDH, LDVL, LDVR, M, MM, N
      LOGICAL            SELECT( * )
      INTEGER            IFAILL( * ), IFAILR( * )
      REAL               RWORK( * )
      COMPLEX            H( LDH, * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   W( * ), WORK( * )
```

Purpose

SHSEIN/CHSEIN uses inverse iteration to find specified right and/or left eigenvectors of a real/complex upper Hessenberg matrix H.

The right eigenvector x and the left eigenvector y of the matrix H corresponding to an eigenvalue w are defined by:

$$H*x = w*x, \; y^H*H = w*y^H.$$

Arguments

SIDE (input) CHARACTER*1
= 'R': compute right eigenvectors only;
= 'L': compute left eigenvectors only;
= 'B': compute both right and left eigenvectors.

EIGSRC (input) CHARACTER*1
Specifies the source of eigenvalues supplied in (WR,WI)/W:
= 'Q': the eigenvalues were found using SHSEQR/CHSEQR; thus, if H has zero subdiagonal elements, and so is block-triangular, then the j^{th} eigenvalue can be assumed to be an eigenvalue of the block containing the j^{th} row/column. This property allows SHSEIN/CHSEIN to perform inverse iteration on just one diagonal block.
= 'N': no assumptions are made on the correspondence between eigenvalues and diagonal blocks. In this case, SHSEIN/CHSEIN must always perform inverse iteration using the whole matrix H.

INITV (input) CHARACTER*1
= 'N': no initial vectors are supplied;
= 'U': user-supplied initial vectors are stored in the arrays VL and/or VR.

SELECT *SHSEIN* (input/output) LOGICAL array, dimension (N)
CHSEIN (input) LOGICAL array, dimension (N)
SHSEIN
Specifies the eigenvectors to be computed. To select the real eigenvector corresponding to a real eigenvalue WR(j), SELECT(j) must be set to .TRUE.. To select the complex eigenvector corresponding to a complex eigenvalue (WR(j),WI(j)), with complex conjugate (WR(j+1),WI(j+1)), either SELECT(j) or SELECT(j+1) or both must be set to .TRUE.; then on exit SELECT(j) is .TRUE. and SELECT(j+1) is .FALSE..
CHSEIN
Specifies the eigenvectors to be computed. To select the eigenvector corresponding to the eigenvalue W(j), SELECT(j) must be set to .TRUE..

N (input) INTEGER
The order of the matrix H. N $\geq$ 0.

H (input) REAL/COMPLEX array, dimension (LDH,N)
The upper Hessenberg matrix H.

LDH (input) INTEGER
The leading dimension of the array H. LDH $\geq$ max(1,N).

WR *SHSEIN only* (input/output) REAL array, dimension (N)
WI *SHSEIN only* (input) REAL array, dimension (N)
On entry, the real and imaginary parts of the eigenvalues of H; a complex conjugate pair of eigenvalues must be stored in consecutive elements of WR and WI.
On exit, WR may have been altered since close eigenvalues are perturbed slightly in searching for independent eigenvectors.

W *CHSEIN only* (input/output) COMPLEX array, dimension (N)
On entry, the eigenvalues of H.
On exit, the real parts of W may have been altered since close eigenvalues are perturbed slightly in searching for independent eigenvectors.

VL (input/output) REAL/COMPLEX array, dimension (LDVL,MM)
On entry, if INITV = 'U' and SIDE = 'L' or 'B', VL must contain starting vectors for the inverse iteration for the left eigenvectors; the starting vector for each eigenvector must be in the same column(s) in which the eigenvector will be stored.
On exit, if SIDE = 'L' or 'B', the left eigenvectors specified by SELECT will be stored one after another in the columns of VL, in the same order as their eigenvalues. If SIDE = 'R', VL is not referenced.
SHSEIN only
A complex eigenvector corresponding to a complex eigenvalue is stored in two consecutive columns, the first holding the real part and the second the imaginary part.

LDVL (input) INTEGER
The leading dimension of the array VL.
LDVL $\geq$ max(1,N) if SIDE = 'L' or 'B'; LDVL $\geq$ 1 otherwise.

VR (input/output) REAL/COMPLEX array, dimension (LDVR,MM)
On entry, if INITV = 'U' and SIDE = 'R' or 'B', VR must contain starting vectors for the inverse iteration for the right eigenvectors; the starting vector for each eigenvector must be in the same column(s) in which the eigenvector will be stored.
On exit, if SIDE = 'R' or 'B', the right eigenvectors specified by SELECT will be stored one after another in the columns of VR, in the same order as their eigenvalues. If SIDE = 'L', VR is not referenced.
SHSEIN
A complex eigenvector corresponding to a complex eigenvalue is stored in two consecutive columns, the first holding the real part and the second the imaginary part.

LDVR (input) INTEGER
The leading dimension of the array VR.
LDVR $\geq$ max(1,N) if SIDE = 'R' or 'B'; LDVR $\geq$ 1 otherwise.

MM (input) INTEGER
The number of columns in the arrays VL and/or VR. MM $\geq$ M.

M (output) INTEGER
The number of columns in the arrays VL and/or VR actually used to store the eigenvectors.
SHSEIN
Each selected real eigenvector occupies one column and each selected complex eigenvector occupies two columns.
CHSEIN
Each selected eigenvector occupies one column.

WORK *SHSEIN* (workspace) REAL array, dimension ((N+2)*N)
CHSEIN (workspace) COMPLEX array, dimension (N*N)

RWORK *CHSEIN only* (workspace) REAL array, dimension (N)

IFAILL (output) INTEGER array, dimension (MM)
If SIDE = 'L' or 'B', IFAILL(i) = j > 0 if the left eigenvector in the i^{th} column of VL (corresponding to the eigenvalue w(j)) failed to converge; IFAILL(i) = 0 if the eigenvector converged satisfactorily.
If SIDE = 'R', IFAILL is not referenced.
SHSEIN only
If the i^{th} and $(i+1)^{st}$ columns of VL hold a complex eigenvector, then IFAILL(i) and IFAILL(i+1) are set to the same value.

IFAILR (output) INTEGER array, dimension (MM)
If SIDE = 'R' or 'B', IFAILR(i) = j > 0 if the right eigenvector in the i^{th} column of VR (corresponding to the j^{th} eigenvalue) failed to converge; IFAILR(i) = 0 if the eigenvector converged satisfactorily.
If SIDE = 'L', IFAILR is not referenced.
SHSEIN only

If the i^{th} and $(i+1)^{st}$ columns of VR hold a complex eigenvector, then IFAILR(i) and IFAILR(i+1) are set to the same value.

INFO (output) INTEGER
- = 0: successful exit
- < 0: if INFO = −i, the i^{th} argument had an illegal value.
- > 0: if INFO = i, i is the number of eigenvectors which failed to converge; see IFAILL and IFAILR for further details.

SHSEQR/CHSEQR

```
      SUBROUTINE SHSEQR( JOB, COMPZ, N, ILO, IHI, H, LDH, WR, WI, Z,
     $                   LDZ, WORK, LWORK, INFO )
      CHARACTER          COMPZ, JOB
      INTEGER            IHI, ILO, INFO, LDH, LDZ, LWORK, N
      REAL               H( LDH, * ), WI( * ), WORK( * ), WR( * ),
     $                   Z( LDZ, * )

      SUBROUTINE CHSEQR( JOB, COMPZ, N, ILO, IHI, H, LDH, W, Z, LDZ,
     $                   WORK, LWORK, INFO )
      CHARACTER          COMPZ, JOB
      INTEGER            IHI, ILO, INFO, LDH, LDZ, LWORK, N
      COMPLEX            H( LDH, * ), W( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SHSEQR/CHSEQR computes the eigenvalues of a real/complex upper Hessenberg matrix H and, optionally, the matrices T and Z from the Schur decomposition $H = Z*T*Z^H$, where T is an upper quasi-triangular/triangular matrix (the Schur form), and Z is the orthogonal/unitary matrix of Schur vectors.

Optionally Z may be postmultiplied into an input orthogonal/unitary matrix Q, so that this routine can give the Schur factorization of a matrix A which has been reduced to the Hessenberg form H by the orthogonal/unitary matrix Q:

$$A = Q*H*Q^H = (Q*Z)*T*(Q*Z)^H.$$

Arguments

JOB (input) CHARACTER*1
- = 'E': compute eigenvalues only;
- = 'S': compute eigenvalues and the Schur form T.

COMPZ (input) CHARACTER*1
- = 'N': no Schur vectors are computed;
- = 'I': Z is initialized to the unit matrix and the matrix Z of Schur vectors of H is returned;
- = 'V': Z must contain an orthogonal/unitary matrix Q on entry, and the product Q*Z is returned.

N (input) INTEGER
The order of the matrix H. N ≥ 0.

ILO, IHI (input) INTEGER
It is assumed that H is already upper triangular in rows and columns 1:ilo−1 and ihi+1:n. ILO and IHI are normally set by a previous call to SGEBAL,/CGEBAL, and then passed to SGEHRD/CGEHRD when the matrix output by SGEBAL/CGEBAL is reduced to Hessenberg form. Otherwise ILO and IHI should be set to 1 and N respectively. 1 ≤ ILO ≤ IHI ≤ N, if N > 0; ILO = 1 and IHI = 0, if N = 0.

H (input/output) REAL/COMPLEX array, dimension (LDH,N)
On entry, the upper Hessenberg matrix H.
On exit, if JOB = 'S', H contains the upper quasi-triangular/triangular matrix T from the Schur decomposition (the Schur form).
If JOB = 'E', the contents of H are unspecified on exit.
SHSEQR only
2-by-2 diagonal blocks (corresponding to complex conjugate pairs of eigenvalues) are returned in standard form, with H(i,i) = H(i+1,i+1) and H(i+1,i)*H(i,i+1) < 0.

LDH (input) INTEGER
The leading dimension of the array H. LDH ≥ max(1,N).

WR, WI *SHSEQR only* (output) REAL array, dimension (N)
The real and imaginary parts, respectively, of the computed eigenvalues. If two eigenvalues are computed as a complex conjugate pair, they are stored in consecutive elements of WR and WI, say the i^{th} and $(i+1)^{st}$, with WI(i) > 0 and WI(i+1) < 0. If JOB = 'S', the eigenvalues are stored in the same order as on the diagonal of the Schur form returned in H, with WR(i) = H(i,i) and, if H(i:i+1,i:i+1) is a 2-by-2 diagonal block, WI(i) = $\sqrt{(H(i+1,i)*H(i,i+1))}$ and WI(i+1) = −WI(i).

W *CHSEQR only* (output) COMPLEX array, dimension (N)
The computed eigenvalues. If JOB = 'S', the eigenvalues are stored in the same order as on the diagonal of the Schur form returned in H, with W(i) = H(i,i).

Z (input/output) REAL/COMPLEX array, dimension (LDZ,N)
If COMPZ = 'N': Z is not referenced.
If COMPZ = 'I': on entry, Z need not be set, and on exit, Z contains the orthogonal/unitary matrix Z of the Schur vectors of H.
If COMPZ = 'V': on entry, Z must contain an n-by-n matrix Q, which is assumed to be equal to the unit matrix except for the submatrix Z(ilo:ihi,ilo:ihi); on exit Z contains Q*Z. Normally Q is the orthogonal/unitary matrix generated by SORGHR/CUNGHR after the call to SGEHRD/CGEHRD which formed the Hessenberg matrix H.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ ≥ max(1,N) if COMPZ = 'I' or 'V'; LDZ ≥ 1 otherwise.

WORK (workspace) REAL/COMPLEX array, dimension (N)

LWORK (input) INTEGER
This argument is currently redundant.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, SHSEQR/CHSEQR failed to compute all the eigenvalues in a total of 30*(IHI−ILO+1) iterations; elements 1:ilo−1 and i+1:n of WR and WI (*SHSEQR*) or of W (*CHSEQR*) contain those eigenvalues which have been successfully computed.

SOPGTR/CUPGTR

```
SUBROUTINE SOPGTR( UPLO, N, AP, TAU, Q, LDQ, WORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDQ, N
REAL               AP( * ), Q( LDQ, * ), TAU( * ), WORK( * )

SUBROUTINE CUPGTR( UPLO, N, AP, TAU, Q, LDQ, WORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDQ, N
COMPLEX            AP( * ), Q( LDQ, * ), TAU( * ), WORK( * )
```

Purpose

SOPGTR/CUPGTR generates a real/complex orthogonal/unitary matrix Q which is defined as the product of n−1 elementary reflectors H_i of order n, as returned by SSPTRD/CHPTRD using packed storage:

if UPLO = 'U', $Q = H_{n-1} \cdots H_2 H_1$,

if UPLO = 'L', $Q = H_1 H_2 \cdots H_{n-1}$.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangular packed storage used in previous call to SSPTRD/CHPTRD;
= 'L': Lower triangular packed storage used in previous call to SSPTRD/CHPTRD.

N (input) INTEGER
The order of the matrix Q. N ≥ 0.

AP (input) REAL/COMPLEX array, dimension (N*(N+1)/2)
The vectors which define the elementary reflectors, as returned by SSPTRD/CHPTRD.

TAU (input) REAL/COMPLEX array, dimension (N−1)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SSPTRD/CHPTRD.

Q (output) REAL/COMPLEX array, dimension (LDQ,N)
The n-by-n orthogonal/unitary matrix Q.

LDQ (input) INTEGER
The leading dimension of the array Q. LDQ ≥ max(1,N).

WORK (workspace) REAL/COMPLEX array, dimension (N−1)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SOPMTR/CUPMTR

```
      SUBROUTINE SOPMTR( SIDE, UPLO, TRANS, M, N, AP, TAU, C, LDC, WORK,
     $                   INFO )
      CHARACTER          SIDE, TRANS, UPLO
      INTEGER            INFO, LDC, M, N
      REAL               AP( * ), C( LDC, * ), TAU( * ), WORK( * )

      SUBROUTINE CUPMTR( SIDE, UPLO, TRANS, M, N, AP, TAU, C, LDC, WORK,
     $                   INFO )
      CHARACTER          SIDE, TRANS, UPLO
      INTEGER            INFO, LDC, M, N
      COMPLEX            AP( * ), C( LDC, * ), TAU( * ), WORK( * )
```

Purpose

SOPMTR/CUPMTR overwrites the general real/complex m-by-n matrix C with

	SIDE = 'L'	SIDE = 'R'	
TRANS = 'N':	Q*C	C*Q	
TRANS = 'T':	Q^T*C	C*Q^T	(*SOPMTR only*)
TRANS = 'C':	Q^H*C	C*Q^H	(*CUPMTR only*)

where Q is a real/complex orthogonal/unitary matrix of order nq, with nq = m if SIDE = 'L' and nq = n if SIDE = 'R'. Q is defined as the product of nq−1 elementary reflectors, as returned by SSPTRD/CHPTRD using packed storage:

if UPLO = 'U', $Q = H_{nq-1} \cdots H_2 H_1$;

if UPLO = 'L', $Q = H_1 H_2 \cdots H_{nq-1}$.

Arguments

SIDE (input) CHARACTER*1
= 'L': apply Q or Q^H from the Left;
= 'R': apply Q or Q^H from the Right.

UPLO (input) CHARACTER*1

= 'U': Upper triangular packed storage used in previous call to SSPTRD/CHPTRD;
= 'L': Lower triangular packed storage used in previous call to SSPTRD/CHPTRD.

TRANS (input) CHARACTER*1
= 'N': No transpose, apply Q;
= 'T': Transpose, apply Q^T (*SOPMTR only*)
= 'C': Conjugate transpose, apply Q^H (*CUPMTR only*)

M (input) INTEGER
The number of rows of the matrix C. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix C. N $\geq$ 0.

AP (input) REAL/COMPLEX array, dimension
(M*(M+1)/2) if SIDE = 'L', or (N*(N+1)/2) if SIDE = 'R'
The vectors which define the elementary reflectors, as returned by SSPTRD/CHPTRD. AP is modified by the routine but restored on exit.

TAU (input) REAL/COMPLEX array, dimension
(M−1) if SIDE = 'L' or (N−1) if SIDE = 'R'
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SSPTRD/CHPTRD.

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, the m-by-n matrix C.
On exit, C is overwritten by Q*C or Q^H*C or C*Q^H or C*Q.

LDC (input) INTEGER
The leading dimension of the array C. LDC $\geq$ max(1,M).

WORK (workspace) REAL/COMPLEX array, dimension
(N) if SIDE = 'L', or (M) if SIDE = 'R'

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORGBR/CUNGBR

```
      SUBROUTINE SORGBR( VECT, M, N, K, A, LDA, TAU, WORK, LWORK, INFO )
      CHARACTER          VECT
      INTEGER            INFO, K, LDA, LWORK, M, N
      REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

      SUBROUTINE CUNGBR( VECT, M, N, K, A, LDA, TAU, WORK, LWORK, INFO )
      CHARACTER          VECT
      INTEGER            INFO, K, LDA, LWORK, M, N
      COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SORGBR/CUNGBR generates one of the real/complex orthogonal/unitary matrices Q or P^H determined by SGEBRD/CGEBRD when reducing a real/complex matrix A to bidiagonal form: A = Q*B*P^H. Q and P^H are defined as products of elementary reflectors H_i or G_i respectively.

If VECT = 'Q', A is assumed to have been an m-by-k matrix, and Q is of order M:
if M $\geq$ K, $Q = H_1 H_2 \cdots H_k$ and SORGBR/CUNGBR returns the first N columns of Q, where M $\geq$ N $\geq$ K;
if M < K, $Q = H_1 H_2 \cdots H_{m-1}$ and SORGBR/CUNGBR returns Q as an m-by-m matrix.

If VECT = 'P', A is assumed to have been a k-by-n matrix, and P^H is of order n:
if K < N, $P^H = G_k \cdots G_2 G_1$ and SORGBR/CUNGBR returns the first M rows of P^H, where N $\geq$ M $\geq$ K;
if K $\geq$ N, $P^H = G_{n-1} \cdots G_2 G_1$ and SORGBR/CUNGBR returns P^H as an n-by-n matrix.

Arguments

VECT (input) CHARACTER*1
Specifies whether the matrix Q or the matrix P^H is required, as defined in the transformation applied by SGEBRD/CGEBRD:
= 'Q': generate Q;
= 'P': generate P^H.

M (input) INTEGER
The number of rows of the matrix Q or P^H to be returned.
M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix Q or P^H to be returned.
N $\geq$ 0.
If VECT = 'Q', M $\geq$ N $\geq$ min(M,K);
if VECT = 'P', N $\geq$ M $\geq$ min(N,K).

K (input) INTEGER
If VECT = 'Q', the number of columns in the original m-by-k matrix reduced by SGEBRD/CGEBRD.
If VECT = 'P', the number of rows in the original k-by-n matrix reduced by SGEBRD/CGEBRD.
K $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the vectors which define the elementary reflectors, as returned by SGEBRD/CGEBRD.
On exit, the m-by-n matrix Q or P^H.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

TAU (input) REAL/COMPLEX array, dimension
(min(M,K)) if VECT = 'Q', or (min(N,K)) if VECT = 'P'

TAU(i) must contain the scalar factor of the elementary reflector H_i or G_i, which determines Q or P^H, as returned by SGEBRD/CGEBRD in its array argument TAUQ or TAUP.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK ≥ max(1,min(M,N)). For optimum performance LWORK ≥ min(M,N)*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORGHR/CUNGHR

```
SUBROUTINE SORGHR( N, ILO, IHI, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            IHI, ILO, INFO, LDA, LWORK, N
REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

SUBROUTINE CUNGHR( N, ILO, IHI, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            IHI, ILO, INFO, LDA, LWORK, N
COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SORGHR/CUNGHR generates a real/complex orthogonal/unitary matrix Q which is defined as the product of ihi−ilo elementary reflectors H_i of order n, as returned by SGEHRD/CGEHRD:

$Q = H_{ilo}H_{ilo+1}\cdots H_{ihi-1}$.

Arguments

N (input) INTEGER
The order of the matrix Q. N ≥ 0.

ILO, IHI (input) INTEGER
ILO and IHI must have the same values as in the previous call of SGEHRD/CGEHRD. Q is equal to the unit matrix except in the submatrix Q(ilo+1:ihi,ilo+1:ihi).
1 ≤ ILO ≤ IHI ≤ N, if N > 0; ILO = 1 and IHI = 0, if N = 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the vectors which define the elementary reflectors, as returned by SGEHRD/CGEHRD.
On exit, the n-by-n orthogonal/unitary matrix Q.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,N).

TAU (input) REAL/COMPLEX array, dimension (N−1)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SGEHRD/CGEHRD.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK ≥ IHI−ILO. For optimum performance LWORK ≥ (IHI−ILO)*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORGLQ/CUNGLQ

```
SUBROUTINE SORGLQ( M, N, K, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, K, LDA, LWORK, M, N
REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

SUBROUTINE CUNGLQ( M, N, K, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, K, LDA, LWORK, M, N
COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SORGLQ/CUNGLQ generates an m-by-n real/complex matrix Q with orthonormal rows, which is defined as the first m rows of a product of k elementary reflectors of order n

$Q = H_k\cdots H_2H_1$ $(SORGLQ)$

$Q = {H_k}^H\cdots {H_2}^H {H_1}^H$ $(CUNGLQ)$

as returned by SGELQF/CGELQF.

Arguments

M (input) INTEGER
The number of rows of the matrix Q. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix Q. N $\geq$ M.

K (input) INTEGER
The number of elementary reflectors whose product defines the matrix Q. M $\geq$ K $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the i^{th} row must contain the vector which defines the elementary reflector H_i, for i = 1,2,...,k, as returned by SGELQF/CGELQF in the first k rows of its array argument A.
On exit, the m-by-n matrix Q.

LDA (input) INTEGER
The first dimension of the array A. LDA $\geq$ max(1,M).

TAU (input) REAL/COMPLEX array, dimension (K)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SGELQF/CGELQF.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ max(1,M). For optimum performance LWORK $\geq$ M*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument has an illegal value

SORGQL/CUNGQL

```
SUBROUTINE SORGQL( M, N, K, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, K, LDA, LWORK, M, N
REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

SUBROUTINE CUNGQL( M, N, K, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, K, LDA, LWORK, M, N
COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SORGQL/CUNGQL generates an m-by-n real/complex matrix Q with orthonormal columns, which is defined as the last n columns of a product of k elementary reflectors H_i of order m

$Q = H_k \cdots H_2 H_1$

as returned by SGEQLF/CGEQLF.

Arguments

M (input) INTEGER
The number of rows of the matrix Q. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix Q. M $\geq$ N $\geq$ 0.

K (input) INTEGER
The number of elementary reflectors whose product defines the matrix Q. N $\geq$ K $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the $(n-k+i)^{th}$ column must contain the vector which defines the elementary reflector H_i, for i = 1,2,...,k, as returned by SGEQLF/CGEQLF in the last k columns of its array argument A.
On exit, the m-by-n matrix Q.

LDA (input) INTEGER
The first dimension of the array A. LDA $\geq$ max(1,M).

TAU (input) REAL/COMPLEX array, dimension (K)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SGEQLF/CGEQLF.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ max(1,N). For optimum performance LWORK $\geq$ N*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument has an illegal value

SORGQR/CUNGQR

```
SUBROUTINE SORGQR( M, N, K, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, K, LDA, LWORK, M, N
REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

SUBROUTINE CUNGQR( M, N, K, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, K, LDA, LWORK, M, N
COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SORGQR/CUNGQR generates an m-by-n real/complex matrix Q with orthonormal columns, which is defined as the first n columns of a product of k elementary reflectors H_i of order m

$Q = H_1 H_2 \cdots H_k$

as returned by SGEQRF/CGEQRF.

Arguments

M (input) INTEGER
The number of rows of the matrix Q. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix Q. M $\geq$ N $\geq$ 0.

K (input) INTEGER
The number of elementary reflectors whose product defines the matrix Q. N $\geq$ K $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the i^{th} column must contain the vector which defines the elementary reflector H_i, for i = 1,2,...,k, as returned by SGEQRF/CGEQRF in the first k columns of its array argument A.
On exit, the m-by-n matrix Q.

LDA (input) INTEGER
The first dimension of the array A. LDA $\geq$ max(1,M).

TAU (input) REAL/COMPLEX array, dimension (K)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SGEQRF/CGEQRF.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ max(1,N). For optimum performance LWORK $\geq$ N*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument has an illegal value

SORGRQ/CUNGRQ

```
SUBROUTINE SORGRQ( M, N, K, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, K, LDA, LWORK, M, N
REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

SUBROUTINE CUNGRQ( M, N, K, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, K, LDA, LWORK, M, N
COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SORGRQ/CUNGRQ generates an m-by-n real/complex matrix Q with orthonormal rows, which is defined as the last m rows of a product of k elementary reflectors H_i of order n

$Q = H_1 H_2 \cdots H_k$ $(SORGRQ)$

$Q = {H_1}^H {H_2}^H \cdots {H_k}^H$ $(CUNGRQ)$

as returned by SGERQF/CGERQF.

Arguments

M (input) INTEGER
The number of rows of the matrix Q. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix Q. N $\geq$ M.

K (input) INTEGER
The number of elementary reflectors whose product defines the matrix Q. M $\geq$ K $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the $(m-k+i)^{th}$ row must contain the vector which defines the elementary reflector H_i, for i = 1,2,...,k, as returned by SGERQF/CGERQF in the last k rows of its array argument A.
On exit, the m-by-n matrix Q.

LDA (input) INTEGER
The first dimension of the array A. LDA $\geq$ max(1,M).

TAU (input) REAL/COMPLEX array, dimension (K)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SGERQF/CGERQF.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK ≥ max(1,M). For optimum performance LWORK ≥ M*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument has an illegal value

SORGTR/CUNGTR

```
      SUBROUTINE SORGTR( UPLO, N, A, LDA, TAU, WORK, LWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LWORK, N
      REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

      SUBROUTINE CUNGTR( UPLO, N, A, LDA, TAU, WORK, LWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LWORK, N
      COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

SORGTR/CUNGTR generates a real/complex orthogonal/unitary matrix Q which is defined as the product of n−1 elementary reflectors H_i of order n, as returned by SSYTRD/CHETRD:

if UPLO = 'U', $Q = H_{n-1} \cdots H_2 H_1$,

if UPLO = 'L', $Q = H_1 H_2 \cdots H_{n-1}$.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A contains elementary reflectors from SSYTRD/CHETRD;
= 'L': Lower triangle of A contains elementary reflectors from SSYTRD/CHETRD.

N (input) INTEGER
The order of the matrix Q. N ≥ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the vectors which define the elementary reflectors, as returned by SSYTRD/CHETRD.
On exit, the n-by-n orthogonal/unitary matrix Q.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,N).

TAU (input) REAL/COMPLEX array, dimension (N−1)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SSYTRD/CHETRD.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK ≥ max(1,N−1). For optimum performance LWORK ≥ (N−1)*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORMBR/CUNMBR

```
      SUBROUTINE SORMBR( VECT, SIDE, TRANS, M, N, K, A, LDA, TAU, C,
     $                   LDC, WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS, VECT
      INTEGER            INFO, K, LDA, LDC, LWORK, M, N
      REAL               A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )

      SUBROUTINE CUNMBR( VECT, SIDE, TRANS, M, N, K, A, LDA, TAU, C,
     $                   LDC, WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS, VECT
      INTEGER            INFO, K, LDA, LDC, LWORK, M, N
      COMPLEX            A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )
```

Purpose

If VECT = 'Q', SORMBR/CUNMBR overwrites the general real/complex m-by-n matrix C with

	SIDE = 'L'	SIDE = 'R'	
TRANS = 'N':	Q*C	C*Q	
TRANS = 'T':	Q^T*C	C*Q^T	(*SORMBR only*)
TRANS = 'C':	Q^H*C	C*Q^H	(*CUNMBR only*)

If VECT = 'P', SORMBR/CUNMBR overwrites the general real/complex m-by-n matrix C with

	SIDE = 'L'	SIDE = 'R'	
TRANS = 'N':	P*C	C*P	
TRANS = 'T':	P^T*C	C*P^T	(*SORMBR only*)
TRANS = 'C':	P^H*C	C*P^H	(*CUNMBR only*)

Here Q and P^H are the orthogonal/unitary matrices determined by SGEBRD/CGEBRD when reducing a real/complex matrix A to bidiagonal form: A = Q*B*P^H. Q and P^H are defined as products of elementary reflectors H_i and G_i, respectively.

Let nq = m if SIDE = 'L' and nq = n if SIDE = 'R'. Thus nq is the order of the orthogonal/unitary matrix Q or P^H that is applied.

If VECT = 'Q', A is assumed to have been an nq-by-k matrix:

if nq ≥ k, $Q = H_1 H_2 \cdots H_k$;

if nq < k, $Q = H_1 H_2 \cdots H_{nq-1}$.

If VECT = 'P', A is assumed to have been a k-by-nq matrix:

if k < nq, $P = G_1 G_2 \cdots G_k$;

if k ≥ nq, $P = G_1 G_2 \cdots G_{nq-1}$.

Arguments

VECT (input) CHARACTER*1
= 'Q': apply Q or Q^H;
= 'P': apply P or P^H.

SIDE (input) CHARACTER*1
= 'L': apply Q, Q^H, P or P^H from the Left;
= 'R': apply Q, Q^H, P or P^H from the Right.

TRANS (input) CHARACTER*1
= 'N': No transpose, apply Q or P;
= 'T': Transpose, apply Q^T or P^T (*SORMBR only*);
= 'C': Conjugate transpose, apply Q^H or P^H (*CUNMBR only*).

M (input) INTEGER
The number of rows of the matrix C. M ≥ 0.

N (input) INTEGER
The number of columns of the matrix C. N ≥ 0.

K (input) INTEGER
If VECT = 'Q', the number of columns in the original matrix reduced by SGEBRD/CGEBRD.
If VECT = 'P', the number of rows in the original matrix reduced by SGEBRD/CGEBRD.
K ≥ 0.

A (input) REAL/COMPLEX array, dimension
(LDA,min(nq,K)) if VECT = 'Q', or (LDA,nq) if VECT = 'P'
The vectors which define the elementary reflectors H_i and G_i, whose products determine the matrices Q and P, as returned by SGEBRD/CGEBRD.

LDA (input) INTEGER
The leading dimension of the array A.
If VECT = 'Q', LDA ≥ max(1,nq);
if VECT = 'P', LDA ≥ max(1,min(nq,K)).

TAU (input) REAL/COMPLEX array, dimension (min(nq,K))
TAU(i) must contain the scalar factor of the elementary reflector H_i or G_i which determines Q or P, as returned by SGEBRD/CGEBRD in the array argument TAUQ or TAUP.

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, the m-by-n matrix C.
On exit, C is overwritten by Q*C or Q^H*C or C*Q^H or C*Q or P*C or P^H*C or C*P or C*P^H.

LDC (input) INTEGER
The leading dimension of the array C. LDC ≥ max(1,M).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. If SIDE = 'L', LWORK ≥ max(1,N); if SIDE = 'R', LWORK ≥ max(1,M). For optimum performance LWORK ≥ N*NB if SIDE = 'L', and LWORK ≥ M*NB if SIDE = 'R', where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORMHR/CUNMHR

```
      SUBROUTINE SORMHR( SIDE, TRANS, M, N, ILO, IHI, A, LDA, TAU, C,
     $                   LDC, WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            IHI, ILO, INFO, LDA, LDC, LWORK, M, N
      REAL               A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )

      SUBROUTINE CUNMHR( SIDE, TRANS, M, N, ILO, IHI, A, LDA, TAU, C,
     $                   LDC, WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            IHI, ILO, INFO, LDA, LDC, LWORK, M, N
      COMPLEX            A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )
```

Purpose

SORMHR/CUNMHR overwrites the general real/complex m-by-n matrix C with

	SIDE = 'L'	SIDE = 'R'	
TRANS = 'N':	Q*C	C*Q	
TRANS = 'T':	Q^T*C	C*Q^T	(*SORMHR only*)
TRANS = 'C':	Q^H*C	C*Q^H	(*CUNMHR only*)

where Q is a real/complex orthogonal/unitary matrix of order nq, with nq = m if SIDE = 'L' and nq = n if SIDE = 'R'. Q is defined as the product of ihi−ilo elementary reflectors, as returned by SGEHRD/CGEHRD:

$Q = H_{ilo} H_{ilo+1} \cdots H_{ihi-1}$.

Arguments

SIDE (input) CHARACTER*1
= 'L': apply Q or Q^H from the Left;
= 'R': apply Q or Q^H from the Right.

TRANS (input) CHARACTER*1
= 'N': No transpose, apply Q
= 'T': Transpose, apply Q^T (*SORMHR only*)
= 'C': Conjugate transpose, apply Q^H (*CUNMHR only*)

M (input) INTEGER
The number of rows of the matrix C. M ≥ 0.

N (input) INTEGER
The number of columns of the matrix C. N ≥ 0.

ILO, IHI (input) INTEGER
ILO and IHI must have the same values as in the previous call of SGEHRD/CGEHRD. Q is equal to the unit matrix except in the submatrix Q(ilo+1:ihi,ilo+1:ihi).
If SIDE = 'L', then 1 ≤ ILO ≤ IHI ≤ M, if M > 0, and ILO = 1 and IHI = 0, if M = 0;
if SIDE = 'R', then 1 ≤ ILO ≤ IHI ≤ N, if N > 0, and ILO = 1 and IHI = 0, if N = 0.

A (input) REAL/COMPLEX array, dimension
(LDA,M) if SIDE = 'L', or (LDA,N) if SIDE = 'R'
The vectors which define the elementary reflectors, as returned by SGEHRD/CGEHRD.

LDA (input) INTEGER
The leading dimension of the array A.
LDA ≥ max(1,M) if SIDE = 'L';
LDA ≥ max(1,N) if SIDE = 'R'.

TAU (input) REAL/COMPLEX array, dimension
(M−1) if SIDE = 'L', or (N−1) if SIDE = 'R'
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SGEHRD/CGEHRD.

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, the m-by-n matrix C.
On exit, C is overwritten by Q*C or Q^H*C or C*Q^H or C*Q.

LDC (input) INTEGER
The leading dimension of the array C. LDC ≥ max(1,M).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. If SIDE = 'L', LWORK ≥ max(1,N); if SIDE = 'R', LWORK ≥ max(1,M). For optimum performance LWORK ≥ N*NB if SIDE = 'L', and LWORK ≥ M*NB if SIDE = 'R', where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORMLQ/CUNMLQ

```
      SUBROUTINE SORMLQ( SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            INFO, K, LDA, LDC, LWORK, M, N
      REAL               A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )

      SUBROUTINE CUNMLQ( SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            INFO, K, LDA, LDC, LWORK, M, N
      COMPLEX            A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )
```

Purpose

SORMLQ/CUNMLQ overwrites the general real/complex m-by-n matrix C with

	SIDE = 'L'	SIDE = 'R'	
TRANS = 'N':	Q*C	C*Q	
TRANS = 'T':	Q^T*C	C*Q^T	(*SORMLQ only*)
TRANS = 'C':	Q^H*C	C*Q^H	(*CUNMLQ only*)

where Q is a real/complex orthogonal/unitary matrix defined as the product of k elementary reflectors H_i

$Q = H_k \cdots H_2 H_1$ (*SORMLQ*)

$Q = H_k{}^H \cdots H_2{}^H H_1{}^H$ (*CUNMLQ*)

as returned by SGELQF/CGELQF. Q is of order m if SIDE = 'L' and of order n if SIDE = 'R'.

Arguments

SIDE (input) CHARACTER*1
= 'L': apply Q or Q^H from the Left;
= 'R': apply Q or Q^H from the Right.

TRANS (input) CHARACTER*1
= 'N': No transpose, apply Q
= 'T': Transpose, apply Q^T (*SORMLQ only*)
= 'C': Conjugate transpose, apply Q^H (*CUNMLQ only*)

M (input) INTEGER
The number of rows of the matrix C. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix C. N $\geq$ 0.

K (input) INTEGER
The number of elementary reflectors whose product defines the matrix Q.
If SIDE = 'L', M $\geq$ K $\geq$ 0;
if SIDE = 'R', N $\geq$ K $\geq$ 0.

A (input) REAL/COMPLEX array, dimension
(LDA,M) if SIDE = 'L', or (LDA,N) if SIDE = 'R'
The i^{th} row must contain the vector which defines the elementary reflector H_i, for i = 1,2,...,k, as returned by SGELQF/CGELQF in the first k rows of its array argument A. A is modified by the routine but restored on exit.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,K).

TAU (input) REAL/COMPLEX array, dimension (K)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SGELQF/CGELQF.

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, the m-by-n matrix C.
On exit, C is overwritten by Q*C or Q^H*C or C*Q^H or C*Q.

LDC (input) INTEGER
The leading dimension of the array C. LDC $\geq$ max(1,M).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. If SIDE = 'L', LWORK $\geq$ max(1,N); if SIDE = 'R', LWORK $\geq$ max(1,M). For optimum performance LWORK $\geq$ N*NB if SIDE = 'L', and LWORK $\geq$ M*NB if SIDE = 'R', where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORMQL/CUNMQL

```
      SUBROUTINE SORMQL( SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            INFO, K, LDA, LDC, LWORK, M, N
      REAL               A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )
```

```
      SUBROUTINE CUNMQL( SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            INFO, K, LDA, LDC, LWORK, M, N
      COMPLEX            A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )
```

Purpose

SORMQL/CUNMQL overwrites the general real/complex m-by-n matrix C with

	SIDE = 'L'	SIDE = 'R'	
TRANS = 'N':	Q*C	C*Q	
TRANS = 'T':	Q^T*C	C*Q^T	(*SORMQL only*)
TRANS = 'C':	Q^H*C	C*Q^H	(*CUNMQL only*)

where Q is a real/complex orthogonal/unitary matrix defined as the product of k elementary reflectors H_i

$Q = H_k \cdots H_2 H_1$

as returned by SGEQLF/CGEQLF. Q is of order m if SIDE = 'L' and of order n if SIDE = 'R'.

Arguments

SIDE (input) CHARACTER*1
= 'L': apply Q or Q^H from the Left;
= 'R': apply Q or Q^H from the Right.

TRANS (input) CHARACTER*1
= 'N': No transpose, apply Q
= 'T': Transpose, apply Q^T (*SORMQL only*)
= 'C': Conjugate transpose, apply Q^H (*CUNMQL only*)

M (input) INTEGER
The number of rows of the matrix C. M ≥ 0.

N (input) INTEGER
The number of columns of the matrix C. N ≥ 0.

K (input) INTEGER
The number of elementary reflectors whose product defines the matrix Q.
If SIDE = 'L', M ≥ K ≥ 0;
if SIDE = 'R', N ≥ K ≥ 0.

A (input) REAL/COMPLEX array, dimension (LDA,K)
The i^{th} column must contain the vector which defines the elementary reflector H_i, for i = 1,2,...,k, as returned by SGEQLF/CGEQLF in the last k columns of its array argument A. A is modified by the routine but restored on exit.

LDA (input) INTEGER
The leading dimension of the array A.
If SIDE = 'L', LDA ≥ max(1,M);
if SIDE = 'R', LDA ≥ max(1,N).

TAU (input) REAL/COMPLEX array, dimension (K)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SGEQLF/CGEQLF.

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, the m-by-n matrix C.
On exit, C is overwritten by Q*C or Q^H*C or C*Q^H or C*Q.

LDC (input) INTEGER
The leading dimension of the array C. LDC ≥ max(1,M).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. If SIDE = 'L', LWORK ≥ max(1,N); if SIDE = 'R', LWORK ≥ max(1,M). For optimum performance LWORK ≥ N*NB if SIDE = 'L', and LWORK ≥ M*NB if SIDE = 'R', where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORMQR/CUNMQR

```
      SUBROUTINE SORMQR( SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            INFO, K, LDA, LDC, LWORK, M, N
      REAL               A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )

      SUBROUTINE CUNMQR( SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            INFO, K, LDA, LDC, LWORK, M, N
      COMPLEX            A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )
```

Purpose

SORMQR/CUNMQR overwrites the general real/complex m-by-n matrix C with

	SIDE = 'L'	SIDE = 'R'	
TRANS = 'N':	Q*C	C*Q	
TRANS = 'T':	Q^T*C	C*Q^T	(*SORMQR only*)
TRANS = 'C':	Q^H*C	C*Q^H	(*CUNMQR only*)

where Q is a real/complex orthogonal/unitary matrix defined as the product of k elementary reflectors H_i

$Q = H_1 H_2 \cdots H_k$

as returned by SGEQRF/CGEQRF. Q is of order m if SIDE = 'L' and of order n if SIDE = 'R'.

Arguments

SIDE (input) CHARACTER*1
= 'L': apply Q or Q^H from the Left;
= 'R': apply Q or Q^H from the Right.

TRANS (input) CHARACTER*1
= 'N': No transpose, apply Q
= 'T': Transpose, apply Q^T (*SORMQR only*)
= 'C': Conjugate transpose, apply Q^H (*CUNMQR only*)

M (input) INTEGER
The number of rows of the matrix C. M ≥ 0.

N (input) INTEGER
The number of columns of the matrix C. N ≥ 0.

K (input) INTEGER
The number of elementary reflectors whose product defines the matrix Q.
If SIDE = 'L', M ≥ K ≥ 0;
if SIDE = 'R', N ≥ K ≥ 0.

A (input) REAL/COMPLEX array, dimension (LDA,K)
The i^{th} column must contain the vector which defines the elementary reflector H_i, for i = 1,2,...,k, as returned by SGEQRF/CGEQRF in the first k columns of its array argument A. A is modified by the routine but restored on exit.

LDA (input) INTEGER
The leading dimension of the array A.
If SIDE = 'L', LDA ≥ max(1,M);
if SIDE = 'R', LDA ≥ max(1,N).

TAU (input) REAL/COMPLEX array, dimension (K)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SGEQRF/CGEQRF.

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, the m-by-n matrix C.
On exit, C is overwritten by Q*C or Q^H*C or C*Q^H or C*Q.

LDC (input) INTEGER
The leading dimension of the array C. LDC ≥ max(1,M).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. If SIDE = 'L', LWORK ≥ max(1,N); if SIDE = 'R', LWORK ≥ max(1,M). For optimum performance LWORK ≥ N*NB if SIDE = 'L', and LWORK ≥ M*NB if SIDE = 'R', where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORMRQ/CUNMRQ

```
      SUBROUTINE SORMRQ( SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            INFO, K, LDA, LDC, LWORK, M, N
      REAL               A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )

      SUBROUTINE CUNMRQ( SIDE, TRANS, M, N, K, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            INFO, K, LDA, LDC, LWORK, M, N
      COMPLEX            A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )
```

Purpose

SORMRQ/CUNMRQ overwrites the general real/complex m-by-n matrix C with

	SIDE = 'L'	SIDE = 'R'	
TRANS = 'N':	Q*C	C*Q	
TRANS = 'T':	Q^T*C	C*Q^T	(*SORMRQ only*)
TRANS = 'C':	Q^H*C	C*Q^H	(*CUNMRQ only*)

where Q is a real/complex orthogonal/unitary matrix defined as the product of k elementary reflectors H_i

$Q = H_1 H_2 \cdots H_k$ (*SORMRQ*)

$Q = H_1{}^H H_2{}^H \cdots H_k{}^H$ (*CUNMRQ*)

as returned by SGERQF/CGERQF. Q is of order m if SIDE = 'L' and of order n if SIDE = 'R'.

Arguments

SIDE (input) CHARACTER*1
= 'L': apply Q or Q^H from the Left;
= 'R': apply Q or Q^H from the Right.

TRANS (input) CHARACTER*1
= 'N': No transpose, apply Q
= 'T': Transpose, apply Q^T (*SORMRQ only*)
= 'C': Conjugate transpose, apply Q^H (*CUNMRQ only*)

M (input) INTEGER
The number of rows of the matrix C. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix C. N $\geq$ 0.

K (input) INTEGER
The number of elementary reflectors whose product defines the matrix Q.
If SIDE = 'L', M $\geq$ K $\geq$ 0;
if SIDE = 'R', N $\geq$ K $\geq$ 0.

A (input) REAL/COMPLEX array, dimension
(LDA,M) if SIDE = 'L', or (LDA,N) if SIDE = 'R'
The i^{th} row must contain the vector which defines the elementary reflector H_i, for i = 1,2,...,k, as returned by SGERQF/CGERQF in the last k rows of its array argument A. A is modified by the routine but restored on exit.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,K).

TAU (input) REAL/COMPLEX array, dimension (K)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SGERQF/CGERQF.

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, the m-by-n matrix C.
On exit, C is overwritten by Q*C or Q^H*C or C*Q^H or C*Q.

LDC (input) INTEGER
The leading dimension of the array C. LDC $\geq$ max(1,M).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. If SIDE = 'L', LWORK $\geq$ max(1,N); if SIDE = 'R', LWORK $\geq$ max(1,M). For optimum performance LWORK $\geq$ N*NB if SIDE = 'L', and LWORK $\geq$ M*NB if SIDE = 'R', where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORMRZ/CUNMRZ

```
      SUBROUTINE SORMRZ( SIDE, TRANS, M, N, K, L, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            INFO, K, L, LDA, LDC, LWORK, M, N
      REAL               A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )

      SUBROUTINE CUNMRZ( SIDE, TRANS, M, N, K, L, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS
      INTEGER            INFO, K, L, LDA, LDC, LWORK, M, N
      COMPLEX            A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )
```

Purpose

SORMRZ/CUNMRZ overwrites the general real/complex m-by-n matrix C with

	SIDE = 'L'	SIDE = 'R'	
TRANS = 'N':	Q*C	C*Q	
TRANS = 'T':	Q^T*C	C*Q^T	(*SORMRZ only*)
TRANS = 'C':	Q^H*C	C*Q^H	(*CUNMRZ only*)

where Q is a real/complex orthogonal/unitary matrix defined as the product of k elementary reflectors H_i

$Q = H_1 H_2 \cdots H_k$ (*SORMRZ*)

$Q = H_1{}^H H_2{}^H \cdots H_k{}^H$ (*CUNMRZ*)

as returned by STZRZF/CTZRZF. Q is of order m if SIDE = 'L' and of order n if SIDE = 'R'.

Arguments

SIDE (input) CHARACTER*1
= 'L': apply Q or Q^H from the Left;

= 'R': apply Q or Q^H from the Right.

TRANS (input) CHARACTER*1
= 'N': No transpose, apply Q
= 'T': Transpose, apply Q^T (*SORMRZ only*)
= 'C': Conjugate transpose, apply Q^H (*CUNMRZ only*)

M (input) INTEGER
The number of rows of the matrix C. M $\geq$ 0.

N (input) INTEGER
The number of columns of the matrix C. N $\geq$ 0.

K (input) INTEGER
The number of elementary reflectors whose product defines the matrix Q.
If SIDE = 'L', M $\geq$ K $\geq$ 0;
if SIDE = 'R', N $\geq$ K $\geq$ 0.

L (input) INTEGER
The number of columns of the matrix A containing the meaningful part of the Householder reflectors.
If SIDE = 'L', M $\geq$ L $\geq$ 0, if SIDE = 'R', N $\geq$ L $\geq$ 0.

A (input) REAL/COMPLEX array, dimension
(LDA,M) if SIDE = 'L', or (LDA,N) if SIDE = 'R'
The i^{th} row must contain the vector which defines the elementary reflector H_i, for i = 1,2,...,k, as returned by STZRZF/CTZRZF in the last k rows of its array argument A. A is modified by the routine but restored on exit.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,K).

TAU (input) REAL/COMPLEX array, dimension (K)
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by STZRZF/CTZRZF.

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, the m-by-n matrix C.
On exit, C is overwritten by Q*C or Q^H*C or C*Q^H or C*Q.

LDC (input) INTEGER
The leading dimension of the array C. LDC $\geq$ max(1,M).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. If SIDE = 'L', LWORK $\geq$ max(1,N); if SIDE = 'R', LWORK $\geq$ max(1,M). For optimum performance LWORK $\geq$ N*NB if SIDE = 'L', and LWORK $\geq$ M*NB if SIDE = 'R', where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SORMTR/CUNMTR

```
      SUBROUTINE SORMTR( SIDE, UPLO, TRANS, M, N, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS, UPLO
      INTEGER            INFO, LDA, LDC, LWORK, M, N
      REAL               A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )

      SUBROUTINE CUNMTR( SIDE, UPLO, TRANS, M, N, A, LDA, TAU, C, LDC,
     $                   WORK, LWORK, INFO )
      CHARACTER          SIDE, TRANS, UPLO
      INTEGER            INFO, LDA, LDC, LWORK, M, N
      COMPLEX            A( LDA, * ), C( LDC, * ), TAU( * ),
     $                   WORK( LWORK )
```

Purpose

SORMTR/CUNMTR overwrites the general real/complex m-by-n matrix C with

	SIDE = 'L'	SIDE = 'R'	
TRANS = 'N':	Q*C	C*Q	
TRANS = 'T':	Q^T*C	C*Q^T	(*SORMTR only*)
TRANS = 'C':	Q^H*C	C*Q^H	(*CUNMTR only*)

where Q is a real/complex orthogonal/unitary matrix of order nq, with nq = m if SIDE = 'L' and nq = n if SIDE = 'R'. Q is defined as the product of nq−1 elementary reflectors, as returned by SSYTRD/CHETRD:

if UPLO = 'U', $Q = H_{nq-1} \cdots H_2 H_1$;

if UPLO = 'L', $Q = H_1 H_2 \cdots H_{nq-1}$.

Arguments

SIDE (input) CHARACTER*1
= 'L': apply Q or Q^H from the Left;
= 'R': apply Q or Q^H from the Right.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A contains elementary reflectors from SSYTRD/CHETRD;

= 'L': Lower triangle of A contains elementary reflectors from SSYTRD/CHETRD.

TRANS (input) CHARACTER*1
= 'N': No transpose, apply Q
= 'T': Transpose, apply Q^T (*SORMTR only*)
= 'C': Conjugate transpose, apply Q^H (*CUNMTR only*)

M (input) INTEGER
The number of rows of the matrix C. M ≥ 0.

N (input) INTEGER
The number of columns of the matrix C. N ≥ 0.

A (input) REAL/COMPLEX array, dimension
(LDA,M) if SIDE = 'L', or (LDA,N) if SIDE = 'R'
The vectors which define the elementary reflectors, as returned by SSYTRD/CHETRD.

LDA (input) INTEGER
The leading dimension of the array A.
LDA ≥ max(1,M) if SIDE = 'L'; LDA ≥ max(1,N) if SIDE = 'R'.

TAU (input) REAL/COMPLEX array, dimension
(M−1) if SIDE = 'L', or (N−1) if SIDE = 'R'
TAU(i) must contain the scalar factor of the elementary reflector H_i, as returned by SSYTRD/CHETRD.

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, the m-by-n matrix C.
On exit, C is overwritten by Q*C or Q^H*C or C*Q^H or C*Q.

LDC (input) INTEGER
The leading dimension of the array C. LDC ≥ max(1,M).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. If SIDE = 'L', LWORK ≥ max(1,N); if SIDE = 'R', LWORK ≥ max(1,M). For optimum performance LWORK ≥ N*NB if SIDE = 'L', and LWORK ≥ M*NB if SIDE = 'R', where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPBCON/CPBCON

```
      SUBROUTINE SPBCON( UPLO, N, KD, AB, LDAB, ANORM, RCOND, WORK,
     $                   IWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, KD, LDAB, N
      REAL               ANORM, RCOND
      INTEGER            IWORK( * )
      REAL               AB( LDAB, * ), WORK( * )

      SUBROUTINE CPBCON( UPLO, N, KD, AB, LDAB, ANORM, RCOND, WORK,
     $                   RWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, KD, LDAB, N
      REAL               ANORM, RCOND
      REAL               RWORK( * )
      COMPLEX            AB( LDAB, * ), WORK( * )
```

Purpose

SPBCON/CPBCON estimates the reciprocal of the condition number (in the 1-norm) of a real/complex symmetric/Hermitian positive definite band matrix using the Cholesky factorization A = U^H*U or A = L*L^H computed by SPBTRF/CPBTRF.

An estimate is obtained for $||A^{-1}||$, and the reciprocal of the condition number is computed as RCOND = $1/(||A|| * ||A^{-1}||)$.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangular factor stored in AB;
= 'L': Lower triangular factor stored in AB.

N (input) INTEGER
The order of the matrix A. N ≥ 0.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KD ≥ 0.

AB (input) REAL/COMPLEX array, dimension (LDAB,N)
The triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H of the band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of U or L is stored in the j^{th} column of the array AB as follows:
if UPLO ='U', AB(kd+1+i−j,j) = U(i,j) for max(1,j−kd)≤i≤j;
if UPLO ='L', AB(1+i−j,j) = L(i,j) for j≤i≤min(n,j+kd).

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB ≥ KD+1.

ANORM (input) REAL
The 1-norm (or infinity-norm) of the symmetric/Hermitian band matrix

A.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as RCOND = $1/(\|A\| * \|A^{-1}\|)$.

WORK *SPBCON* (workspace) REAL array, dimension (3*N)
CPBCON (workspace) COMPLEX array, dimension (2*N)

IWORK *SPBCON only* (workspace) INTEGER array, dimension (N)

RWORK *CPBCON only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the ith argument had an illegal value.

SPBEQU/CPBEQU

```
      SUBROUTINE SPBEQU( UPLO, N, KD, AB, LDAB, S, SCOND, AMAX, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, KD, LDAB, N
      REAL               AMAX, SCOND
      REAL               AB( LDAB, * ), S( * )

      SUBROUTINE CPBEQU( UPLO, N, KD, AB, LDAB, S, SCOND, AMAX, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, KD, LDAB, N
      REAL               AMAX, SCOND
      REAL               S( * )
      COMPLEX            AB( LDAB, * )
```

Purpose

SPBEQU/CPBEQU computes row and column scalings intended to equilibrate a symmetric/Hermitian positive definite band matrix A and reduce its condition number (with respect to the two-norm). S contains the scale factors, S(i) = $1/\sqrt{(A(i,i))}$, chosen so that the scaled matrix B with elements B(i,j) = S(i)*A(i,j)*S(j) has ones on the diagonal. This choice of S puts the condition number of B within a factor n of the smallest possible condition number over all possible diagonal scalings.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangular of A is stored;
= 'L': Lower triangular of A is stored.

N (input) INTEGER
The order of the matrix A. N ≥ 0.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KD ≥ 0.

AB (input) REAL/COMPLEX array, dimension (LDAB,N)
The upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first kd+1 rows of the array. The jth column of A is stored in the jth column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for max(1,j−kd)≤i≤j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j≤i≤min(n,j+kd).

LDAB (input) INTEGER
The leading dimension of the array A. LDAB ≥ KD+1.

S (output) REAL array, dimension (N)
If INFO = 0, S contains the scale factors for A.

SCOND (output) REAL
If INFO = 0, S contains the ratio of the smallest S(i) to the largest S(i). If SCOND ≥ 0.1 and AMAX is neither too large nor too small, it is not worth scaling by S.

AMAX (output) REAL
Absolute value of largest matrix element. If AMAX is very close to overflow or very close to underflow, the matrix should be scaled.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the ith argument had an illegal value.
> 0: if INFO = i, the ith diagonal element is nonpositive.

SPBRFS/CPBRFS

```
      SUBROUTINE SPBRFS( UPLO, N, KD, NRHS, AB, LDAB, AFB, LDAFB, B,
     $                   LDB, X, LDX, FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, KD, LDAB, LDAFB, LDB, LDX, N, NRHS
      INTEGER            IWORK( * )
      REAL               AB( LDAB, * ), AFB( LDAFB, * ), B( LDB, * ),
     $                   BERR( * ), FERR( * ), WORK( * ), X( LDX, * )

      SUBROUTINE CPBRFS( UPLO, N, KD, NRHS, AB, LDAB, AFB, LDAFB, B,
     $                   LDB, X, LDX, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, KD, LDAB, LDAFB, LDB, LDX, N, NRHS
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            AB( LDAB, * ), AFB( LDAFB, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SPBRFS/CPBRFS improves the computed solution to a system of linear equations when the coefficient matrix is symmetric/Hermitian positive definite and banded, and provides error bounds and backward error estimates for the solution.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N ≥ 0.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KD ≥ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS ≥ 0.

AB (input) REAL/COMPLEX array, dimension (LDAB,N)
The upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for max(1,j−kd)≤i≤j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j≤i≤min(n,j+kd).

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB ≥ KD+1.

AFB (input) REAL/COMPLEX array, dimension (LDAFB,N)
The triangular factor U or L from the Cholesky factorization $A = U^H*U$ or $A = L*L^H$ of the band matrix A as computed by SPBTRF/CPBTRF, in the same storage format as A (see AB).

LDAFB (input) INTEGER
The leading dimension of the array AFB. LDAFB ≥ KD+1.

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,N).

X (input/output) REAL/COMPLEX array, dimension (LDX,NRHS)
On entry, the solution matrix X, as computed by SPBTRS/CPBTRS.
On exit, the improved solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX ≥ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SPBRFS* (workspace) REAL array, dimension (3*N)
CPBRFS (workspace) COMPLEX array, dimension (2*N)

IWORK *SPBRFS only* (workspace) INTEGER array, dimension (N)

RWORK *CPBRFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPBSTF/CPBSTF

```
SUBROUTINE SPBSTF( UPLO, N, KD, AB, LDAB, INFO )
CHARACTER          UPLO
INTEGER            INFO, KD, LDAB, N
REAL               AB( LDAB, * )

SUBROUTINE CPBSTF( UPLO, N, KD, AB, LDAB, INFO )
CHARACTER          UPLO
INTEGER            INFO, KD, LDAB, N
COMPLEX            AB( LDAB, * )
```

Purpose

SPBSTF/CPBSTF computes a split Cholesky factorization of a real/complex symmetric/Hermitian positive definite band matrix A.

This routine is designed to be used in conjunction with SSBGST/CHBGST.

The factorization has the form $A = S^H*S$ where S is a band matrix of the same bandwidth as A and the following structure:

$$S = \begin{pmatrix} U & \\ M & L \end{pmatrix}$$

where U is upper triangular of order m = (n+kd)/2, and L is lower triangular of order n−m.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KD $\geq$ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB,N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for max(1,j−kd)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+kd).
On exit, if INFO = 0, the factor S from the split Cholesky factorization A = S^H*S.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KD+1.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = -i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the factorization could not be completed, because the updated element a(i,i) was negative; the matrix A is not positive definite.

SPBSV/CPBSV

```
SUBROUTINE SPBSV( UPLO, N, KD, NRHS, AB, LDAB, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, KD, LDAB, LDB, N, NRHS
REAL               AB( LDAB, * ), B( LDB, * )

SUBROUTINE CPBSV( UPLO, N, KD, NRHS, AB, LDAB, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, KD, LDAB, LDB, N, NRHS
COMPLEX            AB( LDAB, * ), B( LDB, * )
```

Purpose

SPBSV/CPBSV computes the solution to a real/complex system of linear equations A*X = B, where A is an n-by-n symmetric/Hermitian positive definite band matrix and X and B are n-by-nrhs matrices.

The Cholesky decomposition is used to factor A as A = U^H*U, if UPLO = 'U', or A = L*L^H, if UPLO = 'L', where U is an upper triangular band matrix, and L is a lower triangular band matrix, with the same number of superdiagonals or subdiagonals as A. The factored form of A is then used to solve the system of equations A*X = B.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N $\geq$ 0.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KD $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB,N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for max(1,j−kd)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+kd).
On exit, if INFO = 0, the triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H of the band matrix A, in the same storage format as A.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KD+1.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if INFO = 0, the n-by-nrhs solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the leading minor of order i of A is not positive definite, so the factorization could not be completed, and the solution has not been computed.

SPBSVX/CPBSVX

```
      SUBROUTINE SPBSVX( FACT, UPLO, N, KD, NRHS, AB, LDAB, AFB, LDAFB,
     $                   EQUED, S, B, LDB, X, LDX, RCOND, FERR, BERR,
     $                   WORK, IWORK, INFO )
      CHARACTER          EQUED, FACT, UPLO
      INTEGER            INFO, KD, LDAB, LDAFB, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IWORK( * )
      REAL               AB( LDAB, * ), AFB( LDAFB, * ), B( LDB, * ),
     $                   BERR( * ), FERR( * ), S( * ), WORK( * ),
     $                   X( LDX, * )

      SUBROUTINE CPBSVX( FACT, UPLO, N, KD, NRHS, AB, LDAB, AFB, LDAFB,
     $                   EQUED, S, B, LDB, X, LDX, RCOND, FERR, BERR,
     $                   WORK, RWORK, INFO )
      CHARACTER          EQUED, FACT, UPLO
      INTEGER            INFO, KD, LDAB, LDAFB, LDB, LDX, N, NRHS
      REAL               RCOND
      REAL               BERR( * ), FERR( * ), RWORK( * ), S( * )
      COMPLEX            AB( LDAB, * ), AFB( LDAFB, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SPBSVX/CPBSVX uses the Cholesky factorization $A = U^H*U$ or $A = L*L^H$ to compute the solution to a real/complex system of linear equations $A*X = B$, where A is an n-by-n symmetric/Hermitian positive definite band matrix and X and B are n-by-nrhs matrices.

Error bounds on the solution and a condition estimate are also provided.

Description

The following steps are performed:

1. If FACT = 'E', real scaling factors are computed to equilibrate the system:

   $\mathrm{diag}(S)*A*\mathrm{diag}(S)*(\mathrm{diag}(S))^{-1}*X = \mathrm{diag}(S)*B$

   Whether or not the system will be equilibrated depends on the scaling of the matrix A, but if equilibration is used, A is overwritten by diag(S)*A*diag(S) and B by diag(S)*B.

2. If FACT = 'N' or 'E', the Cholesky decomposition is used to factor the matrix A (after equilibration if FACT = 'E') as
   $A = U^H*U$, if UPLO = 'U', or
   $A = L*L^H$, if UPLO = 'L',
   where U is an upper triangular band matrix, and L is a lower triangular band matrix.

3. If the leading i-by-i principal minor is not positive definite, then the routine returns with INFO = i. Otherwise, the factored form of A is used to estimate the condition number of the matrix A. If the reciprocal of the condition number is less than machine precision, INFO = N+1 is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.

4. The system of equations is solved for X using the factored form of A.

5. Iterative refinement is applied to improve the computed solution matrix and calculate error bounds and backward error estimates for it.

6. If equilibration was used, the matrix X is premultiplied by diag(S) so that it solves the original system before equilibration.

Arguments

FACT (input) CHARACTER*1
Specifies whether or not the factored form of the matrix A is supplied on entry, and if not, whether the matrix A should be equilibrated before it is factored.
= 'F': On entry, AFB contains the factored form of A. If EQUED = 'Y', the matrix A has been equilibrated with scaling factors given by S. AB and AFB will not be modified.
= 'N': The matrix A will be copied to AFB and factored.
= 'E': The matrix A will be equilibrated if necessary, then copied to AFB and factored.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. $N \geq 0$.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. $KD \geq 0$.

NRHS (input) INTEGER
The number of right-hand sides, i.e., the number of columns of the matrices B and X. $NRHS \geq 0$.

AB (input/output) REAL/COMPLEX array, dimension (LDAB,N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first kd+1 rows of the array, except if FACT = 'F' and EQUED = 'Y', then A must contain the equilibrated matrix diag(S)*A*diag(S). The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', $AB(kd+1+i-j,j) = A(i,j)$ for $\max(1,j-kd) \leq i \leq j$;
if UPLO = 'L', $AB(1+i-j,j) = A(i,j)$ for $j \leq i \leq \min(n,j+kd)$.
On exit, if FACT = 'E' and EQUED = 'Y', A is overwritten by diag(S)*A*diag(S).

LDAB (input) INTEGER
The leading dimension of the array A. $LDAB \geq KD+1$.

AFB (input or output) REAL/COMPLEX array, dimension (LDAFB,N)
If FACT = 'F', then AFB is an input argument and on entry contains the triangular factor U or L from the Cholesky factorization A = U^H*U or A = $L*L^H$ of the band matrix A, in the same storage format as A (see AB). If EQUED = 'Y', then AFB is the factored form of the equilibrated matrix A.
If FACT = 'N', then AFB is an output argument and on exit returns the triangular factor U or L from the Cholesky factorization A = U^H*U or A = $L*L^H$.
If FACT = 'E', then AFB is an output argument and on exit returns the triangular factor U or L from the Cholesky factorization A = U^H*U or A = $L*L^H$ of the equilibrated matrix A (see the description of A for the form of the equilibrated matrix).

LDAFB (input) INTEGER
The leading dimension of the array AFB. LDAFB ≥ KD+1.

EQUED (input or output) CHARACTER*1
Specifies the form of equilibration that was done.
= 'N': No equilibration (always true if FACT = 'N').
= 'Y': Equilibration was done, i.e., A has been replaced by diag(S)*A*diag(S).
EQUED is an input argument if FACT = 'F'; otherwise, it is an output argument.

S (input or output) REAL array, dimension (N)
The scale factors for A; not accessed if EQUED = 'N'. S is an input argument if FACT = 'F'; otherwise, S is an output argument. If FACT = 'F' and EQUED = 'Y', each element of S must be positive.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if EQUED = 'N', B is not modified; if EQUED = 'Y', B is overwritten by diag(S)*B.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,N).

X (output) REAL/COMPLEX array, dimension (LDX,NRHS)
If INFO = 0 or INFO = N+1, the N-by-NRHS solution matrix X to the original system of equations. Note that if EQUED = 'Y', A and B are modified on exit, and the solution to the equilibrated system is $(\mathrm{diag}(S))^{-1}*X$.

LDX (input) INTEGER
The leading dimension of the array X. LDX ≥ max(1,N).

RCOND (output) REAL
The estimate of the reciprocal condition number of the matrix A after equilibration (if done). If RCOND is less than the machine precision (in particular, if RCOND = 0), the matrix is singular to working precision. This condition is indicated by a return code of INFO > 0.

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SPBSVX* (workspace) REAL array, dimension (3*N)
CPBSVX (workspace) COMPLEX array, dimension (2*N)

IWORK *SPBSVX only* (workspace) INTEGER array, dimension (N)

RWORK *CPBSVX only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
≤ N: the leading minor of order i of A is not positive definite, so the factorization could not be completed, and the solution has not been computed. RCOND = 0 is returned.
= N+1: U is nonsingular, but RCOND is less than machine precision, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of RCOND would suggest.

SPBTRF/CPBTRF

```
SUBROUTINE SPBTRF( UPLO, N, KD, AB, LDAB, INFO )
CHARACTER          UPLO
INTEGER            INFO, KD, LDAB, N
REAL               AB( LDAB, * )

SUBROUTINE CPBTRF( UPLO, N, KD, AB, LDAB, INFO )
CHARACTER          UPLO
INTEGER            INFO, KD, LDAB, N
COMPLEX            AB( LDAB, * )
```

Purpose

SPBTRF/CPBTRF computes the Cholesky factorization of a real/complex symmetric/Hermitian positive definite band matrix A.

The factorization has the form $A = U^H*U$, if UPLO = 'U', or $A = L*L^H$, if UPLO = 'L', where U is an upper triangular matrix and L is lower triangular.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. $KD \geq 0$.

AB (input/output) REAL/COMPLEX array, dimension (LDAB,N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for $\max(1,j-kd) \leq i \leq j$;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for $j \leq i \leq \min(n,j+kd)$.
On exit, if INFO = 0, the triangular factor U or L from the Cholesky factorization $A = U^H*U$ or $A = L*L^H$ of the band matrix A, in the same storage format as A.

LDAB (input) INTEGER
The leading dimension of the array AB. $LDAB \geq KD+1$.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the leading minor of order i is not positive definite, and the factorization could not be completed.

SPBTRS/CPBTRS

```
      SUBROUTINE SPBTRS( UPLO, N, KD, NRHS, AB, LDAB, B, LDB, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, KD, LDAB, LDB, N, NRHS
      REAL               AB( LDAB, * ), B( LDB, * )

      SUBROUTINE CPBTRS( UPLO, N, KD, NRHS, AB, LDAB, B, LDB, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, KD, LDAB, LDB, N, NRHS
      COMPLEX            AB( LDAB, * ), B( LDB, * )
```

Purpose

SPBTRS/CPBTRS solves a system of linear equations $A*X = B$ with a symmetric/Hermitian positive definite band matrix A using the Cholesky factorization $A = U^H*U$ or $A = L*L^H$ computed by SPBTRF/CPBTRF.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangular factor stored in AB;
= 'L': Lower triangular factor stored in AB.

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. $KD \geq 0$.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. $NRHS \geq 0$.

AB (input) REAL/COMPLEX array, dimension (LDAB,N)
The triangular factor U or L from the Cholesky factorization $A = U^H*U$ or $A = L*L^H$ of the band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of U or L is stored in the j^{th} column of the array AB as follows:
if UPLO ='U', AB(kd+1+i−j,j) = U(i,j) for $\max(1,j-kd) \leq i \leq j$;
if UPLO ='L', AB(1+i−j,j) = L(i,j) for $j \leq i \leq \min(n,j+kd)$.

LDAB (input) INTEGER
The leading dimension of the array AB. $LDAB \geq KD+1$.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. $LDB \geq \max(1,N)$.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPOCON/CPOCON

```
      SUBROUTINE SPOCON( UPLO, N, A, LDA, ANORM, RCOND, WORK, IWORK,
     $                   INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, N
      REAL               ANORM, RCOND
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), WORK( * )
```

```
      SUBROUTINE CPOCON( UPLO, N, A, LDA, ANORM, RCOND, WORK, RWORK,
     $                   INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, N
      REAL               ANORM, RCOND
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), WORK( * )
```

Purpose

SPOCON/CPOCON estimates the reciprocal of the condition number (in the 1-norm) of a real/complex symmetric/Hermitian positive definite matrix using the Cholesky factorization $A = U^H*U$ or $A = L*L^H$ computed by SPOTRF/CPOTRF.

An estimate is obtained for $||A^{-1}||$, and the reciprocal of the condition number is computed as $RCOND = 1/(||A|| * ||A^{-1}||)$.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The triangular factor U or L from the Cholesky factorization $A = U^H*U$ or $A = L*L^H$, as computed by SPOTRF/CPOTRF.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

ANORM (input) REAL
The 1-norm (or infinity-norm) of the symmetric/Hermitian matrix A.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as $RCOND = 1/(||A|| * ||A^{-1}||)$.

WORK *SPOCON* (workspace) REAL array, dimension (3*N)
CPOCON (workspace) COMPLEX array, dimension (2*N)

IWORK *SPOCON only* (workspace) INTEGER array, dimension (N)

RWORK *CPOCON only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPOEQU/CPOEQU

```
      SUBROUTINE SPOEQU( N, A, LDA, S, SCOND, AMAX, INFO )
      INTEGER            INFO, LDA, N
      REAL               AMAX, SCOND
      REAL               A( LDA, * ), S( * )

      SUBROUTINE CPOEQU( N, A, LDA, S, SCOND, AMAX, INFO )
      INTEGER            INFO, LDA, N
      REAL               AMAX, SCOND
      REAL               S( * )
      COMPLEX            A( LDA, * )
```

Purpose

SPOEQU/CPOEQU computes row and column scalings intended to equilibrate a symmetric/Hermitian positive definite matrix A and reduce its condition number (with respect to the two-norm). S contains the scale factors, $S(i) = 1/\sqrt{(A(i,i))}$, chosen so that the scaled matrix B with elements $B(i,j) = S(i)*A(i,j)*S(j)$ has ones on the diagonal. This choice of S puts the condition number of B within a factor n of the smallest possible condition number over all possible diagonal scalings.

Arguments

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The n-by-n symmetric/Hermitian positive definite matrix whose scaling factors are to be computed. Only the diagonal elements of A are referenced.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

S (output) REAL array, dimension (N)
If INFO = 0, S contains the scale factors for A.

SCOND (output) REAL
If INFO = 0, S contains the ratio of the smallest S(i) to the largest S(i). If SCOND $\geq$ 0.1 and AMAX is neither too large nor too small, it is not worth scaling by S.

AMAX (output) REAL
Absolute value of largest matrix element. If AMAX is very close to overflow or very close to underflow, the matrix should be scaled.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the i^{th} diagonal element is nonpositive.

SPORFS/CPORFS

```
      SUBROUTINE SPORFS( UPLO, N, NRHS, A, LDA, AF, LDAF, B, LDB, X,
     $                   LDX, FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   BERR( * ), FERR( * ), WORK( * ), X( LDX, * )

      SUBROUTINE CPORFS( UPLO, N, NRHS, A, LDA, AF, LDAF, B, LDB, X,
     $                   LDX, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SPORFS/CPORFS improves the computed solution to a system of linear equations when the coefficient matrix is symmetric/Hermitian positive definite, and provides error bounds and backward error estimates for the solution.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A, and the strictly upper triangular part of A is not referenced.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

AF (input) REAL/COMPLEX array, dimension (LDAF,N)
The triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H, as computed by SPOTRF/CPOTRF.

LDAF (input) INTEGER
The leading dimension of the array AF. LDAF $\geq$ max(1,N).

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (input/output) REAL/COMPLEX array, dimension (LDX,NRHS)
On entry, the solution matrix X, as computed by SPOTRS/CPOTRS.
On exit, the improved solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SPORFS* (workspace) REAL array, dimension (3*N)
CPORFS (workspace) COMPLEX array, dimension (2*N)

IWORK *SPORFS only* (workspace) INTEGER array, dimension (N)

RWORK *CPORFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPOSV/CPOSV

```
      SUBROUTINE SPOSV( UPLO, N, NRHS, A, LDA, B, LDB, INFO )
      CHARACTER         UPLO
      INTEGER           INFO, LDA, LDB, N, NRHS
      REAL              A( LDA, * ), B( LDB, * )

      SUBROUTINE CPOSV( UPLO, N, NRHS, A, LDA, B, LDB, INFO )
      CHARACTER         UPLO
      INTEGER           INFO, LDA, LDB, N, NRHS
      COMPLEX           A( LDA, * ), B( LDB, * )
```

Purpose

SPOSV/CPOSV computes the solution to a real/complex system of linear equations $A*X = B$, where A is an n-by-n symmetric/Hermitian positive definite matrix and X and B are n-by-nrhs matrices.

The Cholesky decomposition is used to factor A as $A = U^H*U$, if UPLO = 'U', or $A = L*L^H$, if UPLO = 'L', where U is an upper triangular matrix and L is a lower triangular matrix. The factored form of A is then used to solve the system of equations $A*X = B$.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. $N \geq 0$.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. $NRHS \geq 0$.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A, and the strictly upper triangular part of A is not referenced.
On exit, if INFO = 0, the factor U or L from the Cholesky factorization $A = U^H*U$ or $A = L*L^H$.

LDA (input) INTEGER
The leading dimension of the array A. $LDA \geq \max(1,N)$.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if INFO = 0, the n-by-nrhs solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. $LDB \geq \max(1,N)$.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = $-i$, the i^{th} argument had an illegal value.
> 0: if INFO = i, the leading minor of order i of A is not positive definite, so the factorization could not be completed, and the solution has not been computed.

SPOSVX/CPOSVX

```
      SUBROUTINE SPOSVX( FACT, UPLO, N, NRHS, A, LDA, AF, LDAF, EQUED,
     $                   S, B, LDB, X, LDX, RCOND, FERR, BERR, WORK,
     $                   IWORK, INFO )
      CHARACTER          EQUED, FACT, UPLO
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   BERR( * ), FERR( * ), S( * ), WORK( * ),
     $                   X( LDX, * )

      SUBROUTINE CPOSVX( FACT, UPLO, N, NRHS, A, LDA, AF, LDAF, EQUED,
     $                   S, B, LDB, X, LDX, RCOND, FERR, BERR, WORK,
     $                   RWORK, INFO )
      CHARACTER          EQUED, FACT, UPLO
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      REAL               RCOND
      REAL               BERR( * ), FERR( * ), RWORK( * ), S( * )
      COMPLEX            A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SPOSVX/CPOSVX uses the Cholesky factorization $A = U^H*U$ or $A = L*L^H$ to compute the solution to a real/complex system of linear equations $A*X = B$, where A is an n-by-n symmetric/Hermitian positive definite matrix and X and B are n-by-nrhs matrices.

Error bounds on the solution and a condition estimate are also provided.

Description

The following steps are performed:

1. If FACT = 'E', real scaling factors are computed to equilibrate the system:

   $\mathrm{diag}(S)*A*\mathrm{diag}(S)*(\mathrm{diag}(S))^{-1}*X = \mathrm{diag}(S)*B$

   Whether or not the system will be equilibrated depends on the scaling of the matrix A, but if equilibration is used, A is overwritten by diag(S)*A*diag(S) and B by diag(S)*B.

2. If FACT = 'N' or 'E', the Cholesky decomposition is used to factor the matrix A (after equilibration if FACT = 'E') as
   $A = U^H*U$, if UPLO = 'U', or
   $A = L*L^H$, if UPLO = 'L',
   where U is an upper triangular matrix and L is a lower triangular matrix.

3. If the leading i-by-i principal minor is not positive definite, then the routine returns with INFO = i. Otherwise, the factored form of A is used to estimate

the condition number of the matrix A. If the reciprocal of the condition number is less than machine precision, INFO = N+1 is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.

4. The system of equations is solved for X using the factored form of A.
5. Iterative refinement is applied to improve the computed solution matrix and calculate error bounds and backward error estimates for it.
6. If equilibration was used, the matrix X is premultiplied by diag(S) so that it solves the original system before equilibration.

Arguments

FACT (input) CHARACTER*1
Specifies whether or not the factored form of the matrix A is supplied on entry, and if not, whether the matrix A should be equilibrated before it is factored.
= 'F': On entry, AF contains the factored form of A. If EQUED = 'Y', the matrix A has been equilibrated with scaling factors given by S. A and AF will not be modified.
= 'N': The matrix A will be copied to AF and factored.
= 'E': The matrix A will be equilibrated if necessary, then copied to AF and factored.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the symmetric/Hermitian matrix A, except if FACT = 'F' and EQUED = 'Y', then A must contain the equilibrated matrix diag(S)*A*diag(S). If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A, and the strictly upper triangular part of A is not referenced. A is not modified if FACT = 'F' or 'N', or if FACT = 'E' and EQUED = 'N' on exit.
On exit, if FACT = 'E' and EQUED = 'Y', A is overwritten by diag(S)*A*diag(S).

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

AF (input or output) REAL/COMPLEX array, dimension (LDAF,N)
If FACT = 'F', then AF is an input argument and on entry contains the triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H, in the same storage format as A. If EQUED $\neq$ 'N', then AF is the factored form of the equilibrated matrix diag(S)*A*diag(S).
If FACT = 'N', then AF is an output argument and on exit returns the triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H of the original matrix A.
If FACT = 'E', then AF is an output argument and on exit returns the triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H of the equilibrated matrix A (see the description of A for the form of the equilibrated matrix).

LDAF (input) INTEGER
The leading dimension of the array AF. LDAF $\geq$ max(1,N).

EQUED (input or output) CHARACTER*1
Specifies the form of equilibration that was done.
= 'N': No equilibration (always true if FACT = 'N').
= 'Y': Equilibration was done, i.e., A has been replaced by diag(S)*A*diag(S).
EQUED is an input argument if FACT = 'F'; otherwise, it is an output argument.

S (input or output) REAL array, dimension (N)
The scale factors for A; not accessed if EQUED = 'N'. S is an input argument if FACT = 'F'; otherwise, S is an output argument. If FACT = 'F' and EQUED = 'Y', each element of S must be positive.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if EQUED = 'N', B is not modified; if EQUED = 'Y', B is overwritten by diag(S)*B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (output) REAL/COMPLEX array, dimension (LDX,NRHS)
If INFO = 0 or INFO = N+1, the N-by-NRHS solution matrix X to the original system of equations. Note that if EQUED = 'Y', A and B are modified on exit, and the solution to the equilibrated system is $(\text{diag}(S))^{-1}$*X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

RCOND (output) REAL
The estimate of the reciprocal condition number of the matrix A after equilibration (if done). If RCOND is less than the machine precision (in particular, if RCOND = 0), the matrix is singular to working precision. This condition is indicated by a return code of INFO > 0.

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the

magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SPOSVX* (workspace) REAL array, dimension (3*N)
CPOSVX (workspace) COMPLEX array, dimension (2*N)

IWORK *SPOSVX only* (workspace) INTEGER array, dimension (N)

RWORK *CPOSVX only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
≤ N: the leading minor of order i of A is not positive definite, so the factorization could not be completed, and the solution and error bounds could not be computed. RCOND = 0 is returned.
= N+1: U is nonsingular, but RCOND is less than machine precision, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of RCOND would suggest.

SPOTRF/CPOTRF

```
SUBROUTINE SPOTRF( UPLO, N, A, LDA, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, N
REAL               A( LDA, * )

SUBROUTINE CPOTRF( UPLO, N, A, LDA, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, N
COMPLEX            A( LDA, * )
```

Purpose

SPOTRF/CPOTRF computes the Cholesky factorization of a real/complex symmetric/Hermitian positive definite matrix A.

The factorization has the form $A = U^H*U$, if UPLO = 'U', or $A = L*L^H$, if UPLO = 'L', where U is an upper triangular matrix and L is lower triangular.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N ≥ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A, and the strictly upper triangular part of A is not referenced.
On exit, if INFO = 0, the factor U or L from the Cholesky factorization $A = U^H*U$ or $A = L*L^H$.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the leading minor of order i is not positive definite, and the factorization could not be completed.

SPOTRI/CPOTRI

```
SUBROUTINE SPOTRI( UPLO, N, A, LDA, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, N
REAL               A( LDA, * )

SUBROUTINE CPOTRI( UPLO, N, A, LDA, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, N
COMPLEX            A( LDA, * )
```

Purpose

SPOTRI/CPOTRI computes the inverse of a real/complex symmetric/Hermitian positive definite matrix A using the Cholesky factorization $A = U^H*U$ or $A = L*L^H$ computed by SPOTRF/CPOTRF.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;

= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the triangular factor U or L from the Cholesky factorization A = U^H*U or A = $L*L^H$, as computed by SPOTRF/CPOTRF.
On exit, the upper or lower triangle of the (symmetric)/(Hermitian) inverse of A, overwriting the input factor U or L.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the (i,i) element of the factor U or L is zero, and the inverse could not be computed.

SPOTRS/CPOTRS

```
SUBROUTINE SPOTRS( UPLO, N, NRHS, A, LDA, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, LDB, N, NRHS
REAL               A( LDA, * ), B( LDB, * )

SUBROUTINE CPOTRS( UPLO, N, NRHS, A, LDA, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, LDB, N, NRHS
COMPLEX            A( LDA, * ), B( LDB, * )
```

Purpose

SPOTRS/CPOTRS solves a system of linear equations A*X = B with a symmetric/Hermitian positive definite matrix A using the Cholesky factorization A = U^H*U or A = $L*L^H$ computed by SPOTRF/CPOTRF.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The triangular factor U or L from the Cholesky factorization A = U^H*U or A = $L*L^H$, as computed by SPOTRF/CPOTRF.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPPCON/CPPCON

```
SUBROUTINE SPPCON( UPLO, N, AP, ANORM, RCOND, WORK, IWORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
REAL               ANORM, RCOND
INTEGER            IWORK( * )
REAL               AP( * ), WORK( * )

SUBROUTINE CPPCON( UPLO, N, AP, ANORM, RCOND, WORK, RWORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
REAL               ANORM, RCOND
REAL               RWORK( * )
COMPLEX            AP( * ), WORK( * )
```

Purpose

SPPCON/CPPCON estimates the reciprocal of the condition number (in the 1-norm) of a real/complex symmetric/Hermitian positive definite packed matrix using the Cholesky factorization A = U^H*U or A = $L*L^H$ computed by SPPTRF/CPPTRF.

An estimate is obtained for $||A^{-1}||$, and the reciprocal of the condition number is computed as RCOND = $1/(||A|| * ||A^{-1}||)$.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

AP (input) REAL/COMPLEX array, dimension (N*(N+1)/2)
The triangular factor U or L from the Cholesky factorization $A = U^H*U$ or $A = L*L^H$, packed columnwise in a linear array. The j^{th} column of U or L is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = U(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = L(i,j) for $j \leq i \leq n$.

ANORM (input) REAL
The 1-norm (or infinity-norm) of the symmetric/Hermitian matrix A.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as $RCOND = 1/(\|A\| * \|A^{-1}\|)$.

WORK *SPPCON* (workspace) REAL array, dimension (3*N)
CPPCON (workspace) COMPLEX array, dimension (2*N)

IWORK *SPPCON only* (workspace) INTEGER array, dimension (N)

RWORK *CPPCON only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPPEQU/CPPEQU

```
      SUBROUTINE SPPEQU( UPLO, N, AP, S, SCOND, AMAX, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, N
      REAL               AMAX, SCOND
      REAL               AP( * ), S( * )

      SUBROUTINE CPPEQU( UPLO, N, AP, S, SCOND, AMAX, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, N
      REAL               AMAX, SCOND
      REAL               S( * )
      COMPLEX            AP( * )
```

Purpose

SPPEQU/CPPEQU computes row and column scalings intended to equilibrate a symmetric/Hermitian positive definite matrix A in packed storage and reduce its condition number (with respect to the two-norm). S contains the scale factors, $S(i) = 1/\sqrt{(A(i,i))}$, chosen so that the scaled matrix B with elements $B(i,j) = S(i)*A(i,j)*S(j)$ has ones on the diagonal. This choice of S puts the condition number of B within a factor n of the smallest possible condition number over all possible diagonal scalings.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

AP (input) REAL/COMPLEX array, dimension (N*(N+1)/2)
The upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.

S (output) REAL array, dimension (N)
If INFO = 0, S contains the scale factors for A.

SCOND (output) REAL
If INFO = 0, S contains the ratio of the smallest S(i) to the largest S(i). If SCOND $\geq$ 0.1 and AMAX is neither too large nor too small, it is not worth scaling by S.

AMAX (output) REAL
Absolute value of largest matrix element. If AMAX is very close to overflow or very close to underflow, the matrix should be scaled.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the i^{th} diagonal element is nonpositive.

SPPRFS/CPPRFS

```
      SUBROUTINE SPPRFS( UPLO, N, NRHS, AP, AFP, B, LDB, X, LDX, FERR,
     $                   BERR, WORK, IWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      INTEGER            IWORK( * )
      REAL               AFP( * ), AP( * ), B( LDB, * ), BERR( * ),
     $                   FERR( * ), WORK( * ), X( LDX, * )

      SUBROUTINE CPPRFS( UPLO, N, NRHS, AP, AFP, B, LDB, X, LDX, FERR,
     $                   BERR, WORK, RWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            AFP( * ), AP( * ), B( LDB, * ), WORK( * ),
     $                   X( LDX, * )
```

Purpose

SPPRFS/CPPRFS improves the computed solution to a system of linear equations when the coefficient matrix is symmetric/Hermitian positive definite and packed, and provides error bounds and backward error estimates for the solution.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

AP (input) REAL/COMPLEX array, dimension (N*(N+1)/2)
The upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.

AFP (input) REAL/COMPLEX array, dimension (N*(N+1)/2)
The triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H, as computed by SPPTRF/CPPTRF, packed columnwise in a linear array in the same format as A (see AP).

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (input/output) REAL/COMPLEX array, dimension (LDX,NRHS)
On entry, the solution matrix X, as computed by SPPTRS/CPPTRS. On exit, the improved solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SPPRFS* (workspace) REAL array, dimension (3*N)
CPPRFS (workspace) COMPLEX array, dimension (2*N)

IWORK *SPPRFS only* (workspace) INTEGER array, dimension (N)

RWORK *CPPRFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPPSV/CPPSV

```
      SUBROUTINE SPPSV( UPLO, N, NRHS, AP, B, LDB, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDB, N, NRHS
      REAL               AP( * ), B( LDB, * )

      SUBROUTINE CPPSV( UPLO, N, NRHS, AP, B, LDB, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDB, N, NRHS
      COMPLEX            AP( * ), B( LDB, * )
```

Purpose

SPPSV/CPPSV computes the solution to a real/complex system of linear equations A*X = B, where A is an n-by-n symmetric/Hermitian positive definite matrix stored in packed format and X and B are n-by-nrhs matrices.

The Cholesky decomposition is used to factor A as A = U^H*U, if UPLO = 'U', or A = L*L^H, if UPLO = 'L', where U is an upper triangular matrix and L is a lower triangular matrix. The factored form of A is then used to solve the system of equations A*X = B.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \le i \le j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \le i \le n$.
On exit, if INFO = 0, the factor U or L from the Cholesky factorization $A = U^H*U$ or $A = L*L^H$, in the same storage format as A.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if INFO = 0, the n-by-nrhs solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the leading minor of order i of A is not positive definite, so the factorization could not be completed, and the solution has not been computed.

SPPSVX/CPPSVX

```
      SUBROUTINE SPPSVX( FACT, UPLO, N, NRHS, AP, AFP, EQUED, S, B, LDB,
     $                   X, LDX, RCOND, FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          EQUED, FACT, UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IWORK( * )
      REAL               AFP( * ), AP( * ), B( LDB, * ), BERR( * ),
     $                   FERR( * ), S( * ), WORK( * ), X( LDX, * )

      SUBROUTINE CPPSVX( FACT, UPLO, N, NRHS, AP, AFP, EQUED, S, B, LDB,
     $                   X, LDX, RCOND, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          EQUED, FACT, UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               RCOND
      REAL               BERR( * ), FERR( * ), RWORK( * ), S( * )
      COMPLEX            AFP( * ), AP( * ), B( LDB, * ), WORK( * ),
     $                   X( LDX, * )
```

Purpose

SPPSVX/CPPSVX uses the Cholesky factorization $A = U^H*U$ or $A = L*L^H$ to compute the solution to a real/complex system of linear equations A*X = B, where A is an n-by-n symmetric/Hermitian positive definite matrix stored in packed format and X and B are n-by-nrhs matrices.

Error bounds on the solution and a condition estimate are also provided.

Description

The following steps are performed:

1. If FACT = 'E', real scaling factors are computed to equilibrate the system:

   $\text{diag}(S)*A*\text{diag}(S)*(\text{diag}(S))^{-1}*X = \text{diag}(S)*B$

   Whether or not the system will be equilibrated depends on the scaling of the matrix A, but if equilibration is used, A is overwritten by diag(S)*A*diag(S) and B by diag(S)*B.

2. If FACT = 'N' or 'E', the Cholesky decomposition is used to factor the matrix A (after equilibration if FACT = 'E') as
   $A = U^H*U$, if UPLO = 'U', or
   $A = L*L^H$, if UPLO = 'L',
   where U is an upper triangular matrix, and L is a lower triangular matrix.

3. If the leading i-by-i principal minor is not positive definite, then the routine returns with INFO = i. Otherwise, the factored form of A is used to estimate the condition number of the matrix A. If the reciprocal of the condition number is less than machine precision, INFO = N+1 is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.

4. The system of equations is solved for X using the factored form of A.

5. Iterative refinement is applied to improve the computed solution matrix and calculate error bounds and backward error estimates for it.

6. If equilibration was used, the matrix X is premultiplied by diag(S) so that it solves the original system before equilibration.

Arguments

FACT (input) CHARACTER*1
Specifies whether or not the factored form of the matrix A is supplied on entry, and if not, whether the matrix A should be equilibrated before it is factored.
= 'F': On entry, AFP contains the factored form of A. If EQUED = 'Y', the matrix A has been equilibrated with scaling factors given by S. AP and AFP will not be modified.
= 'N': The matrix A will be copied to AFP and factored.
= 'E': The matrix A will be equilibrated if necessary, then copied to AFP and factored.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array, except if FACT = 'F' and EQUED = 'Y', then A must contain the equilibrated matrix diag(S)*A*diag(S). The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.
A is not modified if FACT = 'F' or 'N', or if FACT = 'E' and EQUED = 'N' on exit.
On exit, if FACT = 'E' and EQUED = 'Y', A is overwritten by diag(S)*A*diag(S).

AFP (input or output) REAL/COMPLEX array, dimension (N*(N+1)/2)
If FACT = 'F', then AFP is an input argument and on entry contains the triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H, in the same storage format as A. If EQUED $\neq$ 'N', then AFP is the factored form of the equilibrated matrix A.
If FACT = 'N', then AFP is an output argument and on exit returns the triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H of the original matrix A.
If FACT = 'E', then AFP is an output argument and on exit returns the triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H of the equilibrated matrix A (see the description of AP for the form of the equilibrated matrix).

EQUED (input or output) CHARACTER*1
Specifies the form of equilibration that was done.
= 'N': No equilibration (always true if FACT = 'N').
= 'Y': Equilibration was done, i.e., A has been replaced by diag(S)*A*diag(S).
EQUED is an input argument if FACT = 'F'; otherwise, it is an output argument.

S (input or output) REAL array, dimension (N)
The scale factors for A; not accessed if EQUED = 'N'. S is an input argument if FACT = 'F'; otherwise, S is an output argument. If FACT = 'F' and EQUED = 'Y', each element of S must be positive.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if EQUED = 'N', B is not modified; if EQUED = 'Y', B is overwritten by diag(S)*B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (output) REAL/COMPLEX array, dimension (LDX,NRHS)
If INFO = 0 or INFO = N+1, the N-by-NRHS solution matrix X to the original system of equations. Note that if EQUED = 'Y', A and B are modified on exit, and the solution to the equilibrated system is $(\text{diag(S)})^{-1}$*X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

RCOND (output) REAL
The estimate of the reciprocal condition number of the matrix A after equilibration (if done). If RCOND is less than the machine precision (in particular, if RCOND = 0), the matrix is singular to working precision. This condition is indicated by a return code of INFO > 0.

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SPPSVX* (workspace) REAL array, dimension (3*N)
CPPSVX (workspace) COMPLEX array, dimension (2*N)

IWORK *SPPSVX only* (workspace) INTEGER array, dimension (N)

RWORK *CPPSVX only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
$\leq$ N: the leading minor of order i of A is not positive definite, so the factorization could not be completed, and the solution and error bounds could not be computed. RCOND = 0 is returned.
= N+1: U is nonsingular, but RCOND is less than machine precision, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of RCOND would suggest.

SPPTRF/CPPTRF

```
SUBROUTINE SPPTRF( UPLO, N, AP, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
REAL               AP( * )

SUBROUTINE CPPTRF( UPLO, N, AP, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
COMPLEX            AP( * )
```

Purpose

SPPTRF/CPPTRF computes the Cholesky factorization of a real/complex symmetric/Hermitian positive definite matrix A stored in packed format.

The factorization has the form $A = U^H*U$, if UPLO = 'U', or $A = L*L^H$, if UPLO = 'L', where U is an upper triangular matrix and L is lower triangular.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for 1$\leq$i$\leq$j;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for j$\leq$i$\leq$n.
On exit, if INFO = 0, the triangular factor U or L from the Cholesky factorization $A = U^H*U$ or $A = L*L^H$, in the same storage format as A.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the leading minor of order i is not positive definite, and the factorization could not be completed.

SPPTRI/CPPTRI

```
SUBROUTINE SPPTRI( UPLO, N, AP, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
REAL               AP( * )

SUBROUTINE CPPTRI( UPLO, N, AP, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
COMPLEX            AP( * )
```

Purpose

SPPTRI/CPPTRI computes the inverse of a real/complex symmetric/Hermitian positive definite matrix A using the Cholesky factorization $A = U^H*U$ or $A = L*L^H$ computed by SPPTRF/CPPTRF.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangular factor is stored in AP;
= 'L': Lower triangular factor is stored in AP.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the triangular factor U or L from the Cholesky factorization $A = U^H*U$ or $A = L*L^H$, packed columnwise as a linear array. The j^{th} column of U or L is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = U(i,j) for 1$\leq$i$\leq$j;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = L(i,j) for j$\leq$i$\leq$n.
On exit, the upper or lower triangle of the (symmetric)/(Hermitian) inverse of A, overwriting the input factor U or L.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the (i,i) element of the factor U or L is zero, and the inverse could not be computed.

SPPTRS/CPPTRS

```
SUBROUTINE SPPTRS( UPLO, N, NRHS, AP, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDB, N, NRHS
REAL               AP( * ), B( LDB, * )
```

```
SUBROUTINE CPPTRS( UPLO, N, NRHS, AP, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDB, N, NRHS
COMPLEX            AP( * ), B( LDB, * )
```

Purpose

SPPTRS/CPPTRS solves a system of linear equations A*X = B with a symmetric/Hermitian positive definite matrix A in packed storage using the Cholesky factorization A = U^H*U or A = L*L^H computed by SPPTRF/CPPTRF.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

AP (input) REAL/COMPLEX array, dimension (N*(N+1)/2)
The triangular factor U or L from the Cholesky factorization A = U^H*U or A = L*L^H, packed columnwise in a linear array. The j^{th} column of U or L is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = U(i,j) for 1$\leq$i$\leq$j;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = L(i,j) for j$\leq$i$\leq$n.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPTCON/CPTCON

```
SUBROUTINE SPTCON( N, D, E, ANORM, RCOND, WORK, INFO )
INTEGER            INFO, N
REAL               ANORM, RCOND
REAL               D( * ), E( * ), WORK( * )
```

```
SUBROUTINE CPTCON( N, D, E, ANORM, RCOND, WORK, INFO )
INTEGER            INFO, N
REAL               ANORM, RCOND
REAL               D( * ), WORK( * )
COMPLEX            E( * )
```

Purpose

SPTCON/CPTCON computes the reciprocal of the condition number (in the 1-norm) of a real/complex symmetric/Hermitian positive definite tridiagonal matrix using the factorization A = L*D*L^H or A = U^H*D*U computed by SPTTRF/CPTTRF.

$||A^{-1}||$ is computed by a direct method, and the reciprocal of the condition number is computed as RCOND = $1/(||A|| * ||A^{-1}||)$.

Arguments

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

D (input) REAL array, dimension (N)
The n diagonal elements of the diagonal matrix D from the factorization of A, as computed by SPTTRF/CPTTRF.

E (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) off-diagonal elements of the unit bidiagonal factor U or L from the factorization of A, as computed by SPTTRF/CPTTRF.

ANORM (input) REAL
The 1-norm of the original matrix A.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as RCOND = $1/(||A|| * ||A^{-1}||)$.

WORK (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPTEQR/CPTEQR

```
SUBROUTINE SPTEQR( COMPZ, N, D, E, Z, LDZ, WORK, INFO )
CHARACTER          COMPZ
INTEGER            INFO, LDZ, N
REAL               D( * ), E( * ), WORK( * ), Z( LDZ, * )
```

```
      SUBROUTINE CPTEQR( COMPZ, N, D, E, Z, LDZ, WORK, INFO )
      CHARACTER          COMPZ
      INTEGER            INFO, LDZ, N
      REAL               D( * ), E( * ), WORK( * )
      COMPLEX            Z( LDZ, * )
```

Purpose

SPTEQR/CPTEQR computes all eigenvalues and, optionally, eigenvectors of a symmetric positive definite tridiagonal matrix by first factoring the matrix using SPTTRF, and then calling SBDSQR/CBDSQR to compute the singular values of the bidiagonal factor.

This routine computes the eigenvalues of the positive definite tridiagonal matrix to high relative accuracy. This means that if the eigenvalues range over many orders of magnitude in size, then the small eigenvalues and corresponding eigenvectors will be computed more accurately than, for example, with the standard QR method.

The eigenvectors of a full or band symmetric/Hermitian positive definite matrix can be found if SSYTRD/CHETRD, SSPTRD/CHPTRD, or SSBTRD/CHBTRD has been used to reduce this matrix to tridiagonal form. (The reduction to tridiagonal form, however, may preclude the possibility of obtaining high relative accuracy in the small eigenvalues of the original matrix, these eigenvalues range over many orders of magnitude.)

Arguments

COMPZ (input) CHARACTER*1
= 'N': Compute eigenvalues only.
= 'V': Compute eigenvectors of original symmetric/Hermitian matrix also. Array Z contains the orthogonal/unitary matrix used to reduce the original matrix to tridiagonal form.
= 'I': Compute eigenvectors of tridiagonal matrix also.

N (input) INTEGER
The order of the matrix. $N \geq 0$.

D (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the tridiagonal matrix.
On normal exit, D contains the eigenvalues, in descending order.

E (input/output) REAL array, dimension (N−1)
On entry, the (n−1) subdiagonal elements of the tridiagonal matrix.
On exit, E has been destroyed.

Z (input/output) REAL/COMPLEX array, dimension (LDZ, N)
On entry, if COMPZ = 'V', the orthogonal/unitary matrix used in the reduction to tridiagonal form.
On exit, if COMPZ = 'V', the orthonormal eigenvectors of the original symmetric/Hermitian matrix;
if COMPZ = 'I', the orthonormal eigenvectors of the tridiagonal matrix.
If INFO > 0 on exit, Z contains the eigenvectors associated with only the stored eigenvalues.
If COMPZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if COMPZ = 'V' or 'I', LDZ $\geq$ max(1,N).

WORK (workspace) REAL array, dimension (4*N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is:
$\leq$ N: the Cholesky factorization of the matrix could not be performed because the i^{th} principal minor was not positive definite.
> N: the SVD algorithm failed to converge; if INFO = N+i, i off-diagonal elements of the bidiagonal factor did not converge to zero.

SPTRFS/CPTRFS

```
      SUBROUTINE SPTRFS( N, NRHS, D, E, DF, EF, B, LDB, X, LDX, FERR,
     $                   BERR, WORK, INFO )
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               B( LDB, * ), BERR( * ), D( * ), DF( * ),
     $                   E( * ), EF( * ), FERR( * ), WORK( * ),
     $                   X( LDX, * )

      SUBROUTINE CPTRFS( UPLO, N, NRHS, D, E, DF, EF, B, LDB, X, LDX,
     $                   FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               BERR( * ), D( * ), DF( * ), FERR( * ),
     $                   RWORK( * )
      COMPLEX            B( LDB, * ), E( * ), EF( * ), WORK( * ),
     $                   X( LDX, * )
```

Purpose

SPTRFS/CPTRFS improves the computed solution to a system of linear equations when the coefficient matrix is symmetric/Hermitian positive definite and tridiagonal, and provides error bounds and backward error estimates for the solution.

Arguments

UPLO *CPTRFS only* (input) CHARACTER*1
Specifies whether the superdiagonal or the subdiagonal of the tridiagonal matrix A is stored and the form of the factorization:
= 'U': E is the superdiagonal of A, and $A = U^H*D*U$;
= 'L': E is the subdiagonal of A, and $A = L*D*L^H$.
(The two forms are equivalent if A is real.)

N — (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS — (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

D — (input) REAL array, dimension (N)
The n diagonal elements of the tridiagonal matrix A.

E — (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) off-diagonal elements of the tridiagonal matrix A (see UPLO).

DF — (input) REAL array, dimension (N)
The n diagonal elements of the diagonal matrix D from the factorization computed by SPTTRF/CPTTRF.

EF — (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) off-diagonal elements of the unit bidiagonal factor U or L from the factorization computed by SPTTRF/CPTTRF (see UPLO).

B — (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB — (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X — (input/output) REAL/COMPLEX array, dimension (LDX,NRHS)
On entry, the solution matrix X, as computed by SPTTRS/CPTTRS.
On exit, the improved solution matrix X.

LDX — (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR — (output) REAL array, dimension (NRHS)
The forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j).

BERR — (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK — *SPTRFS* (workspace) REAL array, dimension (2*N)
CPTRFS (workspace) COMPLEX array, dimension (N)

RWORK — *CPTRFS only* (workspace) REAL array, dimension (N)

INFO — (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SPTSV/CPTSV

```
SUBROUTINE SPTSV( N, NRHS, D, E, B, LDB, INFO )
INTEGER            INFO, LDB, N, NRHS
REAL               B( LDB, * ), D( * ), E( * )

SUBROUTINE CPTSV( N, NRHS, D, E, B, LDB, INFO )
INTEGER            INFO, LDB, N, NRHS
REAL               D( * )
COMPLEX            B( LDB, * ), E( * )
```

Purpose

SPTSV/CPTSV computes the solution to a real/complex system of linear equations A*X = B, where A is an n-by-n symmetric/Hermitian positive definite tridiagonal matrix, and X and B are n-by-nrhs matrices.

A is factored as A = $L*D*L^H$, and the factored form of A is then used to solve the system of equations.

Arguments

N — (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS — (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

D — (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the tridiagonal matrix A. On exit, the n diagonal elements of the diagonal matrix D from the factorization A = $L*D*L^H$.

E — (input/output) REAL/COMPLEX array, dimension (N−1)
On entry, the (n−1) subdiagonal elements of the tridiagonal matrix A. On exit, the (n−1) subdiagonal elements of the unit bidiagonal factor L from the $L*D*L^H$ factorization of A.

B — (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if INFO = 0, the n-by-nrhs solution matrix X.

LDB — (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO — (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the leading minor of order i is not positive definite, and the solution has not been computed. The factorization has not been completed unless i = N.

SPTSVX/CPTSVX

```
      SUBROUTINE SPTSVX( FACT, N, NRHS, D, E, DF, EF, B, LDB, X, LDX,
     $                   RCOND, FERR, BERR, WORK, INFO )
      CHARACTER          FACT
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               RCOND
      REAL               B( LDB, * ), BERR( * ), D( * ), DF( * ),
     $                   E( * ), EF( * ), FERR( * ), WORK( * ),
     $                   X( LDX, * )

      SUBROUTINE CPTSVX( FACT, N, NRHS, D, E, DF, EF, B, LDB, X, LDX,
     $                   RCOND, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          FACT
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               RCOND
      REAL               BERR( * ), D( * ), DF( * ), FERR( * ),
     $                   RWORK( * )
      COMPLEX            B( LDB, * ), E( * ), EF( * ), WORK( * ),
     $                   X( LDX, * )
```

Purpose

SPTSVX/CPTSVX uses the factorization $A = L*D*L^H$ to compute the solution to a real/complex system of linear equations $A*X = B$, where A is an n-by-n symmetric/Hermitian positive definite tridiagonal matrix and X and B are n-by-nrhs matrices.

Error bounds on the solution and a condition estimate are also provided.

Description

The following steps are performed:

1. If FACT = 'N', the matrix A is factored as $A = L*D*L^H$, where L is a unit lower bidiagonal matrix and D is diagonal. The factorization can also be regarded as having the form $A = U^H*D*U$.
2. If the leading i-by-i principal minor is not positive definite, then the routine returns with INFO = i. Otherwise, the factored form of A is used to estimate the condition number of the matrix A. If the reciprocal of the condition number is less than machine precision, INFO = N+1 is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.
3. The system of equations is solved for X using the factored form of A.
4. Iterative refinement is applied to improve the computed solution matrix and calculate error bounds and backward error estimates for it.

Arguments

FACT (input) CHARACTER*1
Specifies whether or not the factored form of the matrix A is supplied on entry.
= 'F': On entry, DF and EF contain the factored form of A. D, E, DF, and EF will not be modified.
= 'N': The matrix A will be copied to DF and EF and factored.

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. $NRHS \geq 0$.

D (input) REAL array, dimension (N)
The n diagonal elements of the tridiagonal matrix A.

E (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) subdiagonal elements of the tridiagonal matrix A.

DF (input or output) REAL array, dimension (N)
If FACT = 'F', then DF is an input argument and on entry contains the n diagonal elements of the diagonal matrix D from the $L*D*L^H$ factorization of A.
If FACT = 'N', then DF is an output argument and on exit contains the n diagonal elements of the diagonal matrix D from the $L*D*L^H$ factorization of A.

EF (input or output) REAL/COMPLEX array, dimension (N−1)
If FACT = 'F', then EF is an input argument and on entry contains the (n−1) subdiagonal elements of the unit bidiagonal factor L from the $L*D*L^H$ factorization of A.
If FACT = 'N', then EF is an output argument and on exit contains the (n−1) subdiagonal elements of the unit bidiagonal factor L from the $L*D*L^H$ factorization of A.

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The n-by-nrhs right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. $LDB \geq \max(1,N)$.

X (output) REAL/COMPLEX array, dimension (LDX,NRHS)
If INFO = 0 of INFO = N+1, the N-by-NRHS solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. $LDX \geq \max(1,N)$.

RCOND (output) REAL
The reciprocal condition number of the matrix A. If RCOND is less than the machine precision (in particular, if RCOND = 0), the matrix is singular to working precision.
This condition is indicated by a return code of INFO > 0.

FERR (output) REAL array, dimension (NRHS)
The forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding

to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j).

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SPTSVX* (workspace) REAL array, dimension (2*N)
CPTSVX (workspace) COMPLEX array, dimension (N)

RWORK *CPTSVX only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
≤ N: the leading minor of order i of A is not positive definite, so the factorization could not be completed unless i = N, and the solution and error bounds could not be computed. RCOND = 0 is returned.
= N+1: U is nonsingular, but RCOND is less than machine precision, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of RCOND would suggest.

SPTTRF/CPTTRF

```
      SUBROUTINE SPTTRF( N, D, E, INFO )
      INTEGER            INFO, N
      REAL               D( * ), E( * )

      SUBROUTINE CPTTRF( N, D, E, INFO )
      INTEGER            INFO, N
      REAL               D( * )
      COMPLEX            E( * )
```

Purpose

SPTTRF/CPTTRF computes the factorization of a real/complex symmetric/Hermitian positive definite tridiagonal matrix A.

If the subdiagonal elements of A are supplied in the array E, the factorization has the form A = L*D*L^H, where D is diagonal and L is unit lower bidiagonal; if the superdiagonal elements of A are supplied, it has the form A = U^H*D*U, where U is unit upper bidiagonal. (The two forms are equivalent if A is real.)

Arguments

N (input) INTEGER
The order of the matrix A. N ≥ 0.

D (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the tridiagonal matrix A. On exit, the n diagonal elements of the diagonal matrix D from the L*D*L^H factorization of A.

E (input/output) REAL/COMPLEX array, dimension (N−1)
On entry, the (n−1) off-diagonal elements of the tridiagonal matrix A. On exit, the (n−1) off-diagonal elements of the unit bidiagonal factor L or U from the factorization of A.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the leading minor of order i is not positive definite; if i < N, the factorization could not be completed, while if i = N, the factorization was completed, but D(N) = 0.

SPTTRS/CPTTRS

```
      SUBROUTINE SPTTRS( N, NRHS, D, E, B, LDB, INFO )
      INTEGER            INFO, LDB, N, NRHS
      REAL               B( LDB, * ), D( * ), E( * )

      SUBROUTINE CPTTRS( UPLO, N, NRHS, D, E, B, LDB, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDB, N, NRHS
      REAL               D( * )
      COMPLEX            B( LDB, * ), E( * )
```

Purpose

SPTTRS/CPTTRS solves a system of linear equations A*X = B with a symmetric/Hermitian positive definite tridiagonal matrix A using the factorization A = L*D*L^H or A = U^H*D*U computed by SPTTRF/CPTTRF.

Arguments

UPLO *CPTTRS only* (input) CHARACTER*1
Specifies whether the superdiagonal or the subdiagonal of the tridiagonal matrix A is stored and the form of the factorization:
= 'U': E is the superdiagonal of A, and A = U^H*D*U;
= 'L': E is the subdiagonal of A, and A = L*D*L^H.
(The two forms are equivalent if A is real.)

N (input) INTEGER
The order of the tridiagonal matrix A. N ≥ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

D (input) REAL array, dimension (N)
The n diagonal elements of the diagonal matrix D from the factorization computed by SPTTRF/CPTTRF.

E (input) REAL/COMPLEX array, dimension (N−1)
The (n−1) off-diagonal elements of the unit bidiagonal factor U or L from the factorization computed by SPTTRF/CPTTRF (see UPLO).

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SSBEV/CHBEV

```
      SUBROUTINE SSBEV( JOBZ, UPLO, N, KD, AB, LDAB, W, Z, LDZ, WORK,
     $                  INFO )
      CHARACTER         JOBZ, UPLO
      INTEGER           INFO, KD, LDAB, LDZ, N
      REAL              AB( LDAB, * ), W( * ), WORK( * ), Z( LDZ, * )

      SUBROUTINE CHBEV( JOBZ, UPLO, N, KD, AB, LDAB, W, Z, LDZ, WORK,
     $                  RWORK, INFO )
      CHARACTER         JOBZ, UPLO
      INTEGER           INFO, KD, LDAB, LDZ, N
      REAL              RWORK( * ), W( * )
      COMPLEX           AB( LDAB, * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSBEV/CHBEV computes all the eigenvalues and, optionally, eigenvectors of a real/complex symmetric/Hermitian band matrix A.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KD $\geq$ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB, N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for max(1,j−kd)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+kd).
On exit, AB is overwritten by values generated during the reduction to tridiagonal form. If UPLO = 'U', the first superdiagonal and the diagonal of the tridiagonal matrix T are returned in rows kd and kd+1 of AB, and if UPLO = 'L', the diagonal and first subdiagonal of T are returned in the first two rows of AB.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KD + 1.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, N)
If JOBZ = 'V', then if INFO = 0, Z contains the orthonormal eigenvectors of the matrix A, with the i^{th} column of Z holding the eigenvector associated with W(i).
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

WORK *SSBEV* (workspace) REAL array, dimension (max(1,3*N−2))
CHBEV (workspace) COMPLEX array, dimension (N)

RWORK *CHBEV only* (workspace) REAL array, dimension (max(1,3*N−2))

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: the algorithm failed to converge; if INFO = i, i off-diagonal elements of an intermediate tridiagonal form did not converge to zero.

SSBEVD/CHBEVD

```
      SUBROUTINE SSBEVD( JOBZ, UPLO, N, KD, AB, LDAB, W, Z, LDZ, WORK,
     $                   LWORK, IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, KD, LDAB, LDZ, LIWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               AB( LDAB, * ), W( * ), WORK( * ), Z( LDZ, * )

      SUBROUTINE CHBEVD( JOBZ, UPLO, N, KD, AB, LDAB, W, Z, LDZ, WORK,
     $                   LWORK, RWORK, LRWORK, IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, KD, LDAB, LDZ, LIWORK, LRWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            AB( LDAB, * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSBEVD/CHBEVD computes all the eigenvalues and, optionally, eigenvectors of a real/complex symmetric/Hermitian band matrix A. If eigenvectors are desired, it uses a divide and conquer algorithm.

The divide and conquer algorithm makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KD $\geq$ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB, N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for max(1,j−kd)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+kd).
On exit, AB is overwritten by values generated during the reduction to tridiagonal form. If UPLO = 'U', the first superdiagonal and the diagonal of the tridiagonal matrix T are returned in rows kd and kd+1 of AB, and if UPLO = 'L', the diagonal and first subdiagonal of T are returned in the first two rows of AB.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KD + 1.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, N)
If JOBZ = 'V', then if INFO = 0, Z contains the orthonormal eigenvectors of the matrix A, with the i^{th} column of Z holding the eigenvector associated with W(i).
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. If N $\leq$ 1, LWORK $\geq$ 1.
SSBEVD
If JOBZ = 'N' and N > 2, LWORK $\geq$ 2*N.
If JOBZ = 'V' and N > 2, LWORK $\geq$ (1+5*N+2*N^2).
CHBEVD
If JOBZ = 'N' and N > 1, LWORK $\geq$ N.
If JOBZ = 'V' and N > 1, LWORK $\geq$ 2*N^2.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CHBEVD only* (workspace/output) REAL array, dimension (LRWORK)
On exit, if INFO = 0, RWORK(1) returns the optimal LRWORK.

LRWORK *CHBEVD only* (input) INTEGER
The dimension of array RWORK.
If N $\leq$ 1, LRWORK $\geq$ 1.
If JOBZ = 'N' and N > 1, LRWORK $\geq$ N.
If JOBZ = 'V' and N > 1, LRWORK $\geq$ (1+5*N+2*N^2).

If LRWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the RWORK array, returns this value as the first entry of the RWORK array, and no error message related to LRWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array LIWORK.
SSBEVD
If JOBZ = 'N' or N $\leq$ 1, LIWORK $\geq$ 1.
If JOBZ = 'V' and N > 2, LIWORK $\geq$ 3+5*N.
CHBEVD
If JOBZ = 'N' or N $\leq$ 1, LIWORK $\geq$ 1.
If JOBZ = 'V' and N > 1, LIWORK $\geq$ 3+5*N.

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the algorithm failed to converge; i off-diagonal elements of an intermediate tridiagonal form did not converge to zero.

SSBEVX/CHBEVX

```
      SUBROUTINE SSBEVX( JOBZ, RANGE, UPLO, N, KD, AB, LDAB, Q, LDQ, VL,
     $                   VU, IL, IU, ABSTOL, M, W, Z, LDZ, WORK, IWORK,
     $                   IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, IU, KD, LDAB, LDQ, LDZ, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               AB( LDAB, * ), Q( LDQ, * ), W( * ), WORK( * ),
     $                   Z( LDZ, * )

      SUBROUTINE CHBEVX( JOBZ, RANGE, UPLO, N, KD, AB, LDAB, Q, LDQ, VL,
     $                   VU, IL, IU, ABSTOL, M, W, Z, LDZ, WORK, RWORK,
     $                   IWORK, IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, IU, KD, LDAB, LDQ, LDZ, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            AB( LDAB, * ), Q( LDQ, * ), WORK( * ),
     $                   Z( LDZ, * )
```

Purpose

SSBEVX/CHBEVX computes selected eigenvalues and, optionally, eigenvectors of a real/complex symmetric/Hermitian band matrix A. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

RANGE (input) CHARACTER*1
= 'A': all eigenvalues will be found;
= 'V': all eigenvalues in the half-open interval (VL,VU] will be found;
= 'I': the IL^{th} through IU^{th} eigenvalues will be found.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KD $\geq$ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB, N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for max(1,j−kd)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+kd).
On exit, AB is overwritten by values generated during the reduction to tridiagonal form. If UPLO = 'U', the first superdiagonal and the diagonal of the tridiagonal matrix T are returned in rows kd and kd+1 of AB, and if UPLO = 'L', the diagonal and first subdiagonal of T are returned in the first two rows of AB.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KD + 1.

Q (output) REAL/COMPLEX array, dimension (LDQ, N)
If JOBZ = 'V', the n-by-n orthogonal/unitary matrix used in the reduction to tridiagonal form.
If JOBZ = 'N', the array Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q. If JOBZ = 'V', then LDQ $\geq$ max(1,N).

VL, VU (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be

searched for eigenvalues. VL < VU.
Not referenced if RANGE = 'A' or 'I'.

IL, IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned.
$1 \le IL \le IU \le N$, if $N > 0$; IL = 1 and IU = 0 if N = 0.
Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute error tolerance for the eigenvalues.
An approximate eigenvalue is accepted as converged when it is determined to lie in an interval [a,b] of width less than or equal to ABSTOL + EPS*max(|a|,|b|), where EPS is the machine precision. If ABSTOL ≤ 0, then EPS*$||T||_1$ will be used in its place, where T is the tridiagonal matrix obtained by reducing AB to tridiagonal form.
Eigenvalues will be computed most accurately when ABSTOL is set to twice the underflow threshold 2*SLAMCH('S'), not zero. If this routine returns with INFO>0, indicating that some eigenvectors did not converge, try setting ABSTOL to 2*SLAMCH('S').

M (output) INTEGER
The total number of eigenvalues found. $0 \le M \le N$.
If RANGE = 'A', M = N, and if RANGE = 'I', M = IU−IL+1.

W (output) REAL array, dimension (N)
The first M elements contain the selected eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, max(1,M))
If JOBZ = 'V', then if INFO = 0, the first M columns of Z contain the orthonormal eigenvectors of the matrix A corresponding to the selected eigenvalues, with the i^{th} column of Z holding the eigenvector asociated with W(i). If an eigenvector fails to converge, then that column of Z contains the latest approximation to the eigenvector, and the index of the eigenvector is returned in IFAIL.
If JOBZ = 'N', then Z is not referenced.
Note: the user must ensure that at least max(1,M) columns are supplied in the array Z; if RANGE = 'V', the exact value of M is not known in advance and an upper bound must be used.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\ge$ 1, and if JOBZ = 'V', LDZ $\ge$ max(1,N).

WORK *SSBEVX* (workspace) REAL array, dimension (7*N)
CHBEVX (workspace) COMPLEX array, dimension (N)

RWORK *CHBEVX only* (workspace) REAL array, dimension (7*N)

IWORK (workspace) INTEGER array, dimension (5*N)

IFAIL (output) INTEGER array, dimension (N)
If JOBZ = 'V', then if INFO = 0, the first M elements of IFAIL are zero; if INFO > 0, then IFAIL contains the indices of the eigenvectors that failed to converge.
If JOBZ = 'N', then IFAIL is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, then i eigenvectors failed to converge. Their indices are stored in array IFAIL.

SSBGST/CHBGST

```
      SUBROUTINE SSBGST( VECT, UPLO, N, KA, KB, AB, LDAB, BB, LDBB, X,
     $                   LDX, WORK, INFO )
      CHARACTER          UPLO, VECT
      INTEGER            INFO, KA, KB, LDAB, LDBB, LDX, N
      REAL               AB( LDAB, * ), BB( LDBB, * ), WORK( * ),
     $                   X( LDX, * )

      SUBROUTINE CHBGST( VECT, UPLO, N, KA, KB, AB, LDAB, BB, LDBB, X,
     $                   LDX, WORK, RWORK, INFO )
      CHARACTER          UPLO, VECT
      INTEGER            INFO, KA, KB, LDAB, LDBB, LDX, N
      REAL               RWORK( * )
      COMPLEX            AB( LDAB, * ), BB( LDBB, * ), WORK( * ),
     $                   X( LDX, * )
```

Purpose

SSBGST/CHBGST reduces a real/complex symmetric-definite/Hermitian-definite banded generalized eigenproblem $A*x = \lambda*B*x$ to standard form $C*y = \lambda*y$, such that C has the same bandwidth as A.

B must have been previously factorized as S^H*S by SPBSTF/CPBSTF, using a split Cholesky factorization. A is overwritten by $C = X^H*A*X$, where $X = S^{-1}*Q$ and Q is an/a orthogonal/unitary matrix chosen to preserve the bandwidth of A.

Arguments

VECT (input) CHARACTER*1
= 'N': do not form the transformation matrix X;
= 'V': form X.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrices A and B. $N \ge 0$.

KA (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KA $\geq$ 0.

KB (input) INTEGER
The number of superdiagonals of the matrix B if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KA $\geq$ KB $\geq$ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB,N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first ka+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(ka+1+i−j,j) = A(i,j) for max(1,j−ka)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+ka).
On exit, the transformed matrix X^H*A*X, stored in the same format as A.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KA+1.

BB (input) REAL/COMPLEX array, dimension (LDBB,N)
The banded factor S from the split Cholesky factorization of B, as returned by SPBSTF/CPBSTF, stored in the first kb+1 rows of the array.

LDBB (input) INTEGER
The leading dimension of the array BB. LDBB $\geq$ KB+1.

X (output) REAL/COMPLEX array, dimension (LDX,N)
If VECT = 'V', the n-by-n matrix X.
If VECT = 'N', the array X is not referenced.

LDX (input) INTEGER
The leading dimension of the array X.
LDX $\geq$ max(1,N) if VECT = 'V'; LDX $\geq$ 1 otherwise.

WORK *SSBGST* (workspace) REAL array, dimension (2*N)
CHBGST (workspace) COMPLEX array, dimension (N)

RWORK *CHBGST only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SSBGV/CHBGV

```
      SUBROUTINE SSBGV( JOBZ, UPLO, N, KA, KB, AB, LDAB, BB, LDBB, W, Z,
     $                  LDZ, WORK, INFO )
      CHARACTER         JOBZ, UPLO
      INTEGER           INFO, KA, KB, LDAB, LDBB, LDZ, N
      REAL              AB( LDAB, * ), BB( LDBB, * ), W( * ),
     $                  WORK( * ), Z( LDZ, * )
```

```
      SUBROUTINE CHBGV( JOBZ, UPLO, N, KA, KB, AB, LDAB, BB, LDBB, W, Z,
     $                  LDZ, WORK, RWORK, INFO )
      CHARACTER         JOBZ, UPLO
      INTEGER           INFO, KA, KB, LDAB, LDBB, LDZ, N
      REAL              RWORK( * ), W( * )
      COMPLEX           AB( LDAB, * ), BB( LDBB, * ), WORK( * ),
     $                  Z( LDZ, * )
```

Purpose

SSBGV/CHBGV computes all the eigenvalues, and optionally, the eigenvectors of a real/complex generalized symmetric-definite/Hermitian-definite banded eigenproblem, of the form $A*x = \lambda*B*x$. Here A and B are assumed to be symmetric/Hermitian and banded, and B is also positive definite.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangles of A and B are stored;
= 'L': Lower triangles of A and B are stored.

N (input) INTEGER
The order of the matrices A and B. N $\geq$ 0.

KA (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KA $\geq$ 0.

KB (input) INTEGER
The number of superdiagonals of the matrix B if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KB $\geq$ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB, N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first ka+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(ka+1+i−j,j) = A(i,j) for max(1,j−ka)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+ka).
On exit, the contents of AB are destroyed.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KA+1.

BB (input/output) REAL/COMPLEX array, dimension (LDBB, N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix B, stored in the first kb+1 rows of the array. The j^{th} column of B is stored in the j^{th} column of the array BB as follows:
if UPLO = 'U', BB(kb+1+i−j,j) = B(i,j) for max(1,j−kb)$\leq$i$\leq$j;
if UPLO = 'L', BB(1+i−j,j) = B(i,j) for j$\leq$i$\leq$min(n,j+kb).
On exit, the factor S from the split Cholesky factorization $B = S^H*S$, as returned by SPBSTF/CPBSTF.

LDBB (input) INTEGER
The leading dimension of the array BB. LDBB ≥ KB+1.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, N)
If JOBZ = 'V', then if INFO = 0, Z contains the matrix Z of eigenvectors, with the i^{th} column of Z holding the eigenvector associated with W(i). The eigenvectors are normalized so that $Z^H*B*Z = I$.
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ ≥ 1, and if JOBZ = 'V', LDZ ≥ N.

WORK *SSBGV* (workspace) REAL array, dimension (3*N)
CHBGV (workspace) COMPLEX array, dimension (N)

RWORK *CHBGV only* (workspace) REAL array, dimension (3*N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is:
≤ N: the algorithm failed to converge: i off-diagonal elements of an intermediate tridiagonal form did not converge to zero;
> N: if INFO = N + i, for 1 ≤ i ≤ N, then SPBSTF/CPBSTF returned INFO = i: B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

SSBGVD/CHBGVD

```
      SUBROUTINE SSBGVD( JOBZ, UPLO, N, KA, KB, AB, LDAB, BB, LDBB, W,
     $                   Z, LDZ, WORK, LWORK, IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, KA, KB, LDAB, LDBB, LDZ, LIWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               AB( LDAB, * ), BB( LDBB, * ), W( * ),
     $                   WORK( * ), Z( LDZ, * )
```

```
      SUBROUTINE CHBGVD( JOBZ, UPLO, N, KA, KB, AB, LDAB, BB, LDBB, W,
     $                   Z, LDZ, WORK, LWORK, RWORK, LRWORK, IWORK,
     $                   LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, KA, KB, LDAB, LDBB, LDZ, LIWORK, LRWORK,
     $                   LWORK, N
      INTEGER            IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            AB( LDAB, * ), BB( LDBB, * ), WORK( * ),
     $                   Z( LDZ, * )
```

Purpose

SSBGVD/CHBGVD computes all the eigenvalues, and optionally, the eigenvectors of a real/complex generalized symmetric-definite/Hermitian-definite banded eigenproblem, of the form A*x = λ*B*x. Here A and B are assumed to be symmetric/Hermitian and banded, and B is also positive definite. If eigenvectors are desired, it uses a divide and conquer algorithm.

The divide and conquer algorithm makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangles of A and B are stored;
= 'L': Lower triangles of A and B are stored.

N (input) INTEGER
The order of the matrices A and B. N ≥ 0.

KA (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KA ≥ 0.

KB (input) INTEGER
The number of superdiagonals of the matrix B if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KB ≥ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB, N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first ka+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(ka+1+i−j,j) = A(i,j) for max(1,j−ka)≤i≤j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j≤i≤min(n,j+ka).
On exit, the contents of AB are destroyed.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KA+1.

BB (input/output) REAL/COMPLEX array, dimension (LDBB, N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix B, stored in the first kb+1 rows of the array. The j^{th} column of B is stored in the j^{th} column of the array BB as follows:
if UPLO = 'U', BB(kb+1+i−j,j) = B(i,j) for max(1,j−kb)$\leq$i$\leq$j;
if UPLO = 'L', BB(1+i−j,j) = B(i,j) for j$\leq$i$\leq$min(n,j+kb).
On exit, the factor S from the split Cholesky factorization B = S^H*S, as returned by SPBSTF/CPBSTF.

LDBB (input) INTEGER
The leading dimension of the array BB. LDBB $\geq$ KB+1.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, N)
If JOBZ = 'V', then if INFO = 0, Z contains the matrix Z of eigenvectors, with the i^{th} column of Z holding the eigenvector associated with W(i). The eigenvectors are normalized so that Z^H*B*Z = I.
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ N.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
If N $\leq$ 1, LWORK $\geq$ 1.
SSBGVD
If JOBZ = 'N' and N > 1, LWORK $\geq$ 3*N.
If JOBZ = 'V' and N > 1, LWORK $\geq$ 1 + 5*N + 2*N^2.
CHBGVD
If JOBZ = 'N' and N > 1, LWORK $\geq$ N. If JOBZ = 'V' and N > 1, LWORK $\geq$ 2*N^2.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CHBGVD only* (workspace) REAL array, dimension (LRWORK)

LRWORK *CHBGVD only* (input) INTEGER
The dimension of the array RWORK.
If N $\leq$ 1, LRWORK $\geq$ 1.
If JOBZ = 'N' and N > 1, LRWORK $\geq$ N.
If JOBZ = 'V' and N > 1, LRWORK $\geq$ 1 + 5*N + 2*N^2.

If LRWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the RWORK array, returns this value as the first entry of the RWORK array, and no error message related to LRWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array IWORK.
If JOBZ = 'N' or N $\leq$ 1, LIWORK $\geq$ 1.
If JOBZ = 'V' and N > 1, LIWORK $\geq$ 3 + 5*N.

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is:
  $\leq$ N: the algorithm failed to converge: i off-diagonal elements of an intermediate tridiagonal form did not converge to zero;
  > N: if INFO = N + i, for 1 $\leq$ i $\leq$ N, then SPBSTF/CPBSTF returned INFO = i: B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

SSBGVX/CHBGVX

```
      SUBROUTINE SSBGVX( JOBZ, RANGE, UPLO, N, KA, KB, AB, LDAB, BB,
     $                   LDBB, Q, LDQ, VL, VU, IL, IU, ABSTOL, M, W, Z,
     $                   LDZ, WORK, IWORK, IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, IU, KA, KB, LDAB, LDBB, LDQ, LDZ, M,
     $                   N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               AB( LDAB, * ), BB( LDBB, * ), Q( LDQ, * ),
     $                   W( * ), WORK( * ), Z( LDZ, * )
```

```
      SUBROUTINE CHBGVX( JOBZ, RANGE, UPLO, N, KA, KB, AB, LDAB, BB,
     $                   LDBB, Q, LDQ, VL, VU, IL, IU, ABSTOL, M, W, Z,
     $                   LDZ, WORK, RWORK, IWORK, IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, IU, KA, KB, LDAB, LDBB, LDQ, LDZ, M,
     $                   N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            AB( LDAB, * ), BB( LDBB, * ), Q( LDQ, * ),
     $                   WORK( * ), Z( LDZ, * )
```

Purpose

SSBGVX/CHBGVX computes selected eigenvalues, and optionally, eigenvectors of a real generalized symmetric-definite/Hermitian-definite banded eigenproblem, of the form $A * x = \lambda * B * x$. Here A and B are assumed to be symmetric/Hermitian and banded, and B is also positive definite. Eigenvalues and eigenvectors can be selected by specifying either all eigenvalues, a range of values or a range of indices for the desired eigenvalues.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

RANGE (input) CHARACTER*1
= 'A': all eigenvalues will be found.
= 'V': all eigenvalues in the half-open interval (VL,VU] will be found.
= 'I': the ILth through IUth eigenvalues will be found.

UPLO (input) CHARACTER*1
= 'U': Upper triangles of A and B are stored;
= 'L': Lower triangles of A and B are stored.

N (input) INTEGER
The order of the matrices A and B. N $\geq$ 0.

KA (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KA $\geq$ 0.

KB (input) INTEGER
The number of superdiagonals of the matrix B if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KB $\geq$ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB, N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first ka+1 rows of the array. The jth column of A is stored in the jth column of the array AB as follows:
if UPLO = 'U', AB(ka+1+i−j,j) = A(i,j) for max(1,j−ka)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+ka).
On exit, the contents of AB are destroyed.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KA+1.

BB (input/output) REAL/COMPLEX array, dimension (LDBB, N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix B, stored in the first kb+1 rows of the array. The jth column of B is stored in the jth column of the array BB as follows:
if UPLO = 'U', BB(kb+1+i−j,j) = B(i,j) for max(1,j−kb)$\leq$i$\leq$j;
if UPLO = 'L', BB(1+i−j,j) = B(i,j) for j$\leq$i$\leq$min(n,j+kb).
On exit, the factor S from the split Cholesky factorization B = S^H*S, as returned by SPBSTF/CPBSTF.

LDBB (input) INTEGER
The leading dimension of the array BB. LDBB $\geq$ KB+1.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Q (output) REAL/COMPLEX array, dimension (LDQ, N)
If JOBZ = 'V', the n-by-n matrix used in the reduction of $A*x = \lambda*B*x$ to standard form, i.e. $C * x = \lambda * x$, and consequently C to tridiagonal form. If JOBZ = 'N', the array Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q. If JOBZ = 'N', LDQ $\geq$ 1. If JOBZ = 'V', LDQ $\geq$ max(1,N).

VL,VU (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be searched for eigenvalues. VL < VU. Not referenced if RANGE = 'A' or 'I'.

IL,IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned. 1 $\leq$ IL $\leq$ IU $\leq$ N, if N > 0; IL = 1 and IU = 0 if N = 0. Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute error tolerance for the eigenvalues.
An approximate eigenvalue is accepted as converged when it is determined to lie in an interval [a,b] of width less than or equal to ABSTOL + EPS*max(|a|,|b|), where EPS is the machine precision. If ABSTOL $\leq$ 0, then EPS*$\|T\|_1$ will be used in its place, where T is the tridiagonal matrix obtained by reducing AB to tridiagonal form.
Eigenvalues will be computed most accurately when ABSTOL is set to twice the underflow threshold 2*SLAMCH('S'), not zero. If this routine returns with INFO>0, indicating that some eigenvectors did not converge, try setting ABSTOL to 2*SLAMCH('S').

M (output) INTEGER
The total number of eigenvalues found. 0 $\leq$ M $\leq$ N. If RANGE = 'A', M = N, and if RANGE = 'I', M = IU−IL+1.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, N)
If JOBZ = 'V', then if INFO = 0, Z contains the matrix Z of eigenvectors, with the i^{th} column of Z holding the eigenvector associated with W(i). The eigenvectors are normalized so that $Z^H*B*Z = I$.
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ ≥ 1, and if JOBZ = 'V', LDZ ≥ max(1,N).

WORK *SSBGVX* (workspace) REAL array, dimension (7*N)
CHBGVX (workspace) COMPLEX array, dimension (N)

RWORK *CHBGVX only* (workspace) REAL array, dimension (7*N)

IWORK (workspace/output) INTEGER array, dimension (5*N)

IFAIL (input) INTEGER array, dimension (M)
If JOBZ = 'V', then if INFO = 0, the first M elements of IFAIL are zero. If INFO > 0, then IFAIL contains the indices of the eigenvalues that failed to converge. If JOBZ = 'N', then IFAIL is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is:
- ≤ N: the algorithm failed to converge: i off-diagonal elements of an intermediate tridiagonal form did not converge to zero;
- > N: if INFO = N + i, for 1 ≤ i ≤ N, then SPBSTF/CPBSTF returned INFO = i: B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

SSBTRD/CHBTRD

```
      SUBROUTINE SSBTRD( VECT, UPLO, N, KD, AB, LDAB, D, E, Q, LDQ,
     $                   WORK, INFO )
      CHARACTER          UPLO, VECT
      INTEGER            INFO, KD, LDAB, LDQ, N
      REAL               AB( LDAB, * ), D( * ), E( * ), Q( LDQ, * ),
     $                   WORK( * )

      SUBROUTINE CHBTRD( VECT, UPLO, N, KD, AB, LDAB, D, E, Q, LDQ,
     $                   WORK, INFO )
      CHARACTER          UPLO, VECT
      INTEGER            INFO, KD, LDAB, LDQ, N
      REAL               D( * ), E( * )
      COMPLEX            AB( LDAB, * ), Q( LDQ, * ), WORK( * )
```

Purpose

SSBTRD/CHBTRD reduces a real/complex symmetric/Hermitian band matrix A to real symmetric tridiagonal form T by an orthogonal/unitary similarity transformation: $Q^H*A*Q = T$.

Arguments

VECT (input) CHARACTER*1
= 'N': do not form Q;
= 'V': form Q;
= 'U': update a matrix X, by forming X*Q.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N ≥ 0.

KD (input) INTEGER
The number of superdiagonals of the matrix A if UPLO = 'U', or the number of subdiagonals if UPLO = 'L'. KD ≥ 0.

AB (input/output) REAL/COMPLEX array, dimension (LDAB,N)
On entry, the upper or lower triangle of the symmetric/Hermitian band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for max(1,j−kd)≤i≤j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j≤i≤min(n,j+kd).
On exit, the diagonal elements of AB are overwritten by the diagonal elements of the tridiagonal matrix T; if KD > 0, the elements on the first superdiagonal (if UPLO = 'U') or the first subdiagonal (if UPLO = 'L') are overwritten by the off-diagonal elements of T; the rest of AB is overwritten by values generated during the reduction.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB ≥ KD+1.

D (output) REAL array, dimension (N)
The diagonal elements of the tridiagonal matrix T.

E (output) REAL array, dimension (N−1)
The off-diagonal elements of the tridiagonal matrix T:
E(i) = T(i,i+1) if UPLO = 'U'; E(i) = T(i+1,i) if UPLO = 'L'.

Q (output) REAL/COMPLEX array, dimension (LDQ,N)
On entry:
if VECT = 'U', then Q must contain an N-by-N matrix X;
if VECT = 'N' or 'V', then Q need not be set.
On exit:
if VECT = 'V', Q contains the N-by-N orthogonal matrix Q;
if VECT = 'U', Q contains the product X*Q;
if VECT = 'N', the array Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q.
LDQ $\geq$ 1, and LDQ $\geq$ N if VECT = 'V' or 'U'.

WORK (workspace) REAL/COMPLEX array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the ith argument had an illegal value.

SSPCON/CSPCON/CHPCON

```
      SUBROUTINE SSPCON( UPLO, N, AP, IPIV, ANORM, RCOND, WORK, IWORK,
     $                   INFO )
      CHARACTER          UPLO
      INTEGER            INFO, N
      REAL               ANORM, RCOND
      INTEGER            IPIV( * ), IWORK( * )
      REAL               AP( * ), WORK( * )

      SUBROUTINE CSPCON( UPLO, N, AP, IPIV, ANORM, RCOND, WORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, N
      REAL               ANORM, RCOND
      INTEGER            IPIV( * )
      COMPLEX            AP( * ), WORK( * )

      SUBROUTINE CHPCON( UPLO, N, AP, IPIV, ANORM, RCOND, WORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, N
      REAL               ANORM, RCOND
      INTEGER            IPIV( * )
      COMPLEX            AP( * ), WORK( * )
```

Purpose

SSPCON/CSPCON estimates the reciprocal of the condition number (in the 1-norm) of a real/complex symmetric packed matrix A using the factorization A = U*D*U^T or A = L*D*L^T computed by SSPTRF/CSPTRF.

CHPCON estimates the reciprocal of the condition number of a complex Hermitian packed matrix A using the factorization A = U*D*U^H or A = L*D*L^H computed by CHPTRF.

An estimate is obtained for $\|A^{-1}\|$, and the reciprocal of the condition number is computed as RCOND = $1/(\|A\| * \|A^{-1}\|)$.

Arguments

UPLO (input) CHARACTER*1
Specifies whether the details of the factorization are stored as an upper or lower triangular matrix.
= 'U': Upper triangular, form is A = U*D*U^T (SSPCON/CSPCON) or A = U*D*U^H (CHPCON);
= 'L': Lower triangular, form is A = L*D*L^T (SSPCON/CSPCON) or A = L*D*L^H (CHPCON).

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

AP (input) REAL/COMPLEX/COMPLEX array, dimension (N*(N+1)/2)
The block diagonal matrix D and the multipliers used to obtain the factor U or L as computed by SSPTRF/CSPTRF/CHPTRF, stored as a packed triangular matrix.

IPIV (input) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D as determined by SSPTRF/CSPTRF/CHPTRF.

ANORM (input) REAL
The 1-norm of the original matrix A.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as RCOND = $1/(\|A\| * \|A^{-1}\|)$.

WORK (workspace) REAL/COMPLEX/COMPLEX array, dimension (2*N)

IWORK *SSPCON only* (workspace) INTEGER array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the ith argument had an illegal value.

SSPEV/CHPEV

```
      SUBROUTINE SSPEV( JOBZ, UPLO, N, AP, W, Z, LDZ, WORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, LDZ, N
      REAL               AP( * ), W( * ), WORK( * ), Z( LDZ, * )

      SUBROUTINE CHPEV( JOBZ, UPLO, N, AP, W, Z, LDZ, WORK, RWORK,
     $                  INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, LDZ, N
      REAL               RWORK( * ), W( * )
      COMPLEX            AP( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSPEV/CHPEV computes all the eigenvalues and, optionally, eigenvectors of a real/complex symmetric/Hermitian matrix A in packed storage.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for 1$\leq$i$\leq$j;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for j$\leq$i$\leq$n.
On exit, AP is overwritten by values generated during the reduction to tridiagonal form. If UPLO = 'U', the diagonal and first superdiagonal of the tridiagonal matrix T overwrite the corresponding elements of A, and if UPLO = 'L', the diagonal and first subdiagonal of T overwrite the corresponding elements of A.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, N)
If JOBZ = 'V', then if INFO = 0, Z contains the orthonormal eigenvectors of the matrix A, with the i^{th} column of Z holding the eigenvector associated with W(i).
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

WORK *SSPEV* (workspace) REAL array, dimension (3*N)
CHPEV (workspace) COMPLEX array, dimension (max(1, 2*N−1))

RWORK *CHPEV only* (workspace) REAL array, dimension (max(1,3*N−2))

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value
> 0: the algorithm failed to converge; if INFO = i, i off-diagonal elements of an intermediate tridiagonal form did not converge to zero.

SSPEVD/CHPEVD

```
      SUBROUTINE SSPEVD( JOBZ, UPLO, N, AP, W, Z, LDZ, WORK, LWORK,
     $                   IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, LDZ, LIWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               AP( * ), W( * ), WORK( * ), Z( LDZ, * )

      SUBROUTINE CHPEVD( JOBZ, UPLO, N, AP, W, Z, LDZ, WORK, LWORK,
     $                   RWORK, LRWORK, IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, LDZ, LIWORK, LRWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            AP( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSPEVD/CHPEVD computes all the eigenvalues and, optionally, eigenvectors of a real/complex symmetric/Hermitian matrix A in packed storage. If eigenvectors are desired, it uses a divide and conquer algorithm.

The divide and conquer algorithm makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for 1$\leq$i$\leq$j;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for j$\leq$i$\leq$n.
On exit, AP is overwritten by values generated during the reduction to tridiagonal form. If UPLO = 'U', the diagonal and first superdiagonal

of the tridiagonal matrix T overwrite the corresponding elements of A, and if UPLO = 'L', the diagonal and first subdiagonal of T overwrite the corresponding elements of A.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, N)
If JOBZ = 'V', then if INFO = 0, Z contains the orthonormal eigenvectors of the matrix A, with the i^{th} column of Z holding the eigenvector associated with W(i).
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
If N $\leq$ 1, LWORK $\geq$ 1.
SSPEVD
If JOBZ = 'N' and N > 1, LWORK $\geq$ 2*N.
If JOBZ = 'V' and N > 1, LWORK $\geq$ (1 + 6*N + 2*N^2).
CHPEVD
If JOBZ = 'N' and N > 1, LWORK $\geq$ N.
If JOBZ = 'V' and N > 1, LWORK $\geq$ 2*N.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CHPEVD only* (workspace/output) REAL array, dimension (LRWORK)
On exit, if INFO = 0, RWORK(1) returns the optimal LRWORK.

LRWORK *CHPEVD only* (input) INTEGER
The dimension of the array RWORK.
If N $\leq$ 1, LRWORK $\geq$ 1.
If JOBZ = 'N' and N > 1, LRWORK $\geq$ N.
If JOBZ = 'V' and N > 1, LRWORK $\geq$ (1 + 5*N + 2*N^2).

If LRWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the RWORK array, returns this value as the first entry of the RWORK array, and no error message related to LRWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array IWORK.
If JOBZ = 'N' or N $\leq$ 1, LIWORK $\geq$ 1.
If JOBZ = 'V' and N > 1, LIWORK $\geq$ 3 + 5*N.

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the algorithm failed to converge; i off-diagonal elements of an intermediate tridiagonal form did not converge to zero.

SSPEVX/CHPEVX

```
      SUBROUTINE SSPEVX( JOBZ, RANGE, UPLO, N, AP, VL, VU, IL, IU,
     $                   ABSTOL, M, W, Z, LDZ, WORK, IWORK, IFAIL,
     $                   INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, IU, LDZ, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               AP( * ), W( * ), WORK( * ), Z( LDZ, * )

      SUBROUTINE CHPEVX( JOBZ, RANGE, UPLO, N, AP, VL, VU, IL, IU,
     $                   ABSTOL, M, W, Z, LDZ, WORK, RWORK, IWORK,
     $                   IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, IU, LDZ, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            AP( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSPEVX/CHPEVX computes selected eigenvalues and, optionally, eigenvectors of a real/complex symmetric/Hermitian matrix A in packed storage. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

Arguments

JOBZ (input) CHARACTER*1

= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

RANGE (input) CHARACTER*1
= 'A': all eigenvalues will be found;
= 'V': all eigenvalues in the half-open interval (VL,VU] will be found;
= 'I': the IL^{th} through IU^{th} eigenvalues will be found.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.
On exit, AP is overwritten by values generated during the reduction to tridiagonal form. If UPLO = 'U', the diagonal and first superdiagonal of the tridiagonal matrix T overwrite the corresponding elements of A, and if UPLO = 'L', the diagonal and first subdiagonal of T overwrite the corresponding elements of A.

VL, VU (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be searched for eigenvalues. VL < VU.
Not referenced if RANGE = 'A' or 'I'.

IL, IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned.
$1 \leq IL \leq IU \leq N$, if $N > 0$; IL = 1 and IU = 0 if N = 0.
Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute error tolerance for the eigenvalues. An approximate eigenvalue is accepted as converged when it is determined to lie in an interval [a,b] of width less than or equal to ABSTOL + EPS*max(|a|,|b|), where EPS is the machine precision. If ABSTOL is less than or equal to zero, then $EPS*\|T\|_1$ will be used in its place, where T is the tridiagonal matrix obtained by reducing A to tridiagonal form.
Eigenvalues will be computed most accurately when ABSTOL is set to twice the underflow threshold 2*SLAMCH('S'), not zero. If this routine returns with INFO>0, indicating that some eigenvectors did not converge, try setting ABSTOL to 2*SLAMCH('S').

M (output) INTEGER
The total number of eigenvalues found. $0 \leq M \leq N$.
If RANGE = 'A', M = N, and if RANGE = 'I', M = IU−IL+1.

W (output) REAL array, dimension (N)
If INFO = 0, the selected eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, max(1,M))
If JOBZ = 'V', then if INFO = 0, the first M columns of Z contain the orthonormal eigenvectors of the matrix A corresponding to the selected eigenvalues, with the i^{th} column of Z holding the eigenvector asociated with W(i). If an eigenvector fails to converge, then that column of Z contains the latest approximation to the eigenvector, and the index of the eigenvector is returned in IFAIL.
If JOBZ = 'N', then Z is not referenced.
Note: the user must ensure that at least max(1,M) columns are supplied in the array Z; if RANGE = 'V', the exact value of M is not known in advance and an upper bound must be used.

LDZ (input) INTEGER
The leading dimension of the array Z. $LDZ \geq 1$, and if JOBZ = 'V', $LDZ \geq \max(1,N)$.

WORK *SSPEVX* (workspace) REAL array, dimension (8*N)
CHPEVX (workspace) COMPLEX array, dimension (2*N)

RWORK *CHPEVX only* (workspace) REAL array, dimension (7*N)

IWORK (workspace) INTEGER array, dimension (5*N)

IFAIL (output) INTEGER array, dimension (N)
If JOBZ = 'V', then if INFO = 0, the first M elements of IFAIL are zero; if INFO > 0, then IFAIL contains the indices of the eigenvectors that failed to converge.
If JOBZ = 'N', then IFAIL is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, then i eigenvectors failed to converge. Their indices are stored in array IFAIL.

SSPGST/CHPGST

```
SUBROUTINE SSPGST( ITYPE, UPLO, N, AP, BP, INFO )
CHARACTER          UPLO
INTEGER            INFO, ITYPE, N
REAL               AP( * ), BP( * )

SUBROUTINE CHPGST( ITYPE, UPLO, N, AP, BP, INFO )
CHARACTER          UPLO
INTEGER            INFO, ITYPE, N
COMPLEX            AP( * ), BP( * )
```

Purpose

SSPGST/CHPGST reduces a real/complex symmetric-definite/Hermitian-definite generalized eigenproblem to standard form, using packed storage.

If ITYPE = 1, the problem is $A*x = \lambda*B*x$, and A is overwritten by $(U^H)^{-1}*A*U^{-1}$ or $L^{-1}*A*(L^H)^{-1}$

If ITYPE = 2 or 3, the problem is $A*B*x = \lambda*x$ or $B*A*x = \lambda*x$, and A is overwritten by $U*A*U^H$ or L^H*A*L.

B must have been previously factorized as U^H*U or $L*L^H$ by SPPTRF/CPPTRF.

Arguments

ITYPE (input) INTEGER
= 1: compute $(U^H)^{-1}*A*U^{-1}$ or $L^{-1}*A*(L^H)^{-1}$;
= 2 or 3: compute $U*A*U^H$ or L^H*A*L.

UPLO (input) CHARACTER
= 'U': Upper triangle of A is stored and B is factored as U^H*U;
= 'L': Lower triangle of A is stored and B is factored as $L*L^H$.

N (input) INTEGER
The order of the matrices A and B. $N \geq 0$.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.
On exit, if INFO = 0, the transformed matrix, stored in the same format as A.

BP (input) REAL/COMPLEX array, dimension (N*(N+1)/2)
The triangular factor from the Cholesky factorization of B, stored in the same format as A, as returned by SPPTRF/CPPTRF.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SSPGV/CHPGV

```
      SUBROUTINE SSPGV( ITYPE, JOBZ, UPLO, N, AP, BP, W, Z, LDZ, WORK,
     $                  INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, ITYPE, LDZ, N
      REAL               AP( * ), BP( * ), W( * ), WORK( * ),
     $                   Z( LDZ, * )
```

```
      SUBROUTINE CHPGV( ITYPE, JOBZ, UPLO, N, AP, BP, W, Z, LDZ, WORK,
     $                  RWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, ITYPE, LDZ, N
      REAL               RWORK( * ), W( * )
      COMPLEX            AP( * ), BP( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSPGV/CHPGV computes all the eigenvalues and, optionally, the eigenvectors of a real/complex generalized symmetric-definite/Hermitian-definite eigenproblem, of the form

$A*x=\lambda*B*x$, $A*B*x=\lambda*x$, or $B*A*x=\lambda*x$.

Here A and B are assumed to be symmetric/Hermitian, stored in packed format, and B is also positive definite.

Arguments

ITYPE (input) INTEGER
Specifies the problem type to be solved:
= 1: $A*x = \lambda*B*x$
= 2: $A*B*x = \lambda*x$
= 3: $B*A*x = \lambda*x$

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangles of A and B are stored;
= 'L': Lower triangles of A and B are stored.

N (input) INTEGER
The order of the matrices A and B. $N \geq 0$.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.
On exit, the contents of AP are destroyed.

BP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix B, packed columnwise in a linear array. The j^{th} column of B is stored in the array BP as follows:
if UPLO = 'U', BP(i + (j−1)*j/2) = B(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', BP(i + (j−1)*(2*n−j)/2) = B(i,j) for $j \leq i \leq n$.
On exit, the triangular factor U or L from the Cholesky factorization $B = U^H*U$ or $B = L*L^H$, in the same storage format as B.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, N)
If JOBZ = 'V', then if INFO = 0, Z contains the matrix Z of eigenvectors. The eigenvectors are normalized as follows:
if ITYPE = 1 or 2, $Z^H*B*Z = I$;
if ITYPE = 3, $Z^H*B^{-1}*Z = I$.
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

WORK *SSPGV* (workspace) REAL array, dimension (3*N)
CHPGV (workspace) COMPLEX array, dimension (max(1, 2*N−1))

RWORK *CHPGV only* (workspace) REAL array, dimension (max(1,3*N−2))

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: SPPTRF/CPPTRF or SSPEV/CHPEV returned an error code:
$\leq$ N: SSPEV/CHPEV failed to converge; if INFO = i, i off-diagonal elements of an intermediate tridiagonal form did not converge to zero;
> N: if INFO = N + i, for 1 $\leq$ i $\leq$ N, then the leading minor of order i of B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

SSPGVD/CHPGVD

```
      SUBROUTINE SSPGVD( ITYPE, JOBZ, UPLO, N, AP, BP, W, Z, LDZ, WORK,
     $                   LWORK, IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, ITYPE, LDZ, LIWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               AP( * ), BP( * ), W( * ), WORK( * ),
     $                   Z( LDZ, * )

      SUBROUTINE CHPGVD( ITYPE, JOBZ, UPLO, N, AP, BP, W, Z, LDZ, WORK,
     $                   LWORK, RWORK, LRWORK, IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, ITYPE, LDZ, LIWORK, LRWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            AP( * ), BP( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSPGVD/CHPGVD computes all the eigenvalues, and optionally, the eigenvectors of a real generalized symmetric-definite/Hermitian-definite eigenproblem, of the form $A*x = \lambda*B*x$, $A*Bx = \lambda*x$, or $B*A*x = \lambda*x$. Here A and B are assumed to be symmetric/Hermitian, stored in packed format, and B is also positive definite. If eigenvectors are desired, it uses a divide and conquer algorithm.

The divide and conquer algorithm makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

Arguments

ITYPE (input) INTEGER
Specifies the problem type to be solved:
= 1: $A*x = \lambda*B*x$
= 2: $A*B*x = \lambda*x$
= 3: $B*A*x = \lambda*x$

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangles of A and B are stored;
= 'L': Lower triangles of A and B are stored.

N (input) INTEGER
The order of the matrices A and B. N $\geq$ 0.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for 1≤i≤j;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for j≤i≤n.
On exit, the contents of AP are destroyed.

BP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of B is stored in the array BP as follows:
if UPLO = 'U', BP(i + (j−1)*j/2) = B(i,j) for 1≤i≤j;
if UPLO = 'L', BP(i + (j−1)*(2*n−j)/2) = B(i,j) for j≤i≤n.

On exit, the triangular factor U or L from the Cholesky factorization $B = U^H*U$ or $B = L*L^H$, in the same storage format as B.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, N)
If JOBZ = 'V', then if INFO = 0, Z contains the matrix Z of eigenvectors. The eigenvectors are normalized as follows:
if ITYPE = 1 or 2, $Z^H * B * Z = I$;
if ITYPE = 3, $Z^H * B^{-1} * Z = I$.
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ ≥ 1, and if JOBZ = 'V', LDZ ≥ max(1,N).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. If N ≤ 1, LWORK ≥ 1.
SSPGVD
If JOBZ = 'N' and N > 1, LWORK ≥ 2*N.
If JOBZ = 'V' and N > 1, LWORK ≥ 1 + 6*N + 2*N^2.
CHPGVD
If JOBZ = 'N' and N > 1, LWORK ≥ N.
If JOBZ = 'V' and N > 1, LWORK ≥ 2*N.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CHPGVD only* (workspace) REAL array, dimension (LRWORK)
On exit, if INFO = 0, RWORK(1) returns the optimal LRWORK.

LRWORK *CHPGVD only* (input) INTEGER
The dimension of array RWORK.
If N ≤ 1, LRWORK ≥ 1.
If JOBZ = 'N' and N > 1, LRWORK ≥ N. If JOBZ = 'V' and N > 1, LRWORK ≥ 1 + 5*N + 2*N^2.

If LRWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the RWORK array, returns this value as the first entry of the RWORK array, and no error message related to LRWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array IWORK.
If JOBZ = 'N' or N ≤ 1, LIWORK ≥ 1. If JOBZ = 'V' and N > 1, LIWORK ≥ 3 + 5*N.

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the ith argument had an illegal value.
> 0: SPPTRF/CPPTRF or SSPEVD/CHPEVD returned an error code:
≤ N: if INFO = i, SSPEVD/CHPEVD failed to converge; i off-diagonal elements of an intermediate tridiagonal form did not converge to zero;
> N: if INFO = N + i, for 1 ≤ i ≤ N, then the leading minor of order i of B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

SSPGVX/CHPGVX

```
      SUBROUTINE SSPGVX( ITYPE, JOBZ, RANGE, UPLO, N, AP, BP, VL, VU,
     $                   IL, IU, ABSTOL, M, W, Z, LDZ, WORK, IWORK,
     $                   IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, ITYPE, IU, LDZ, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               AP( * ), BP( * ), W( * ), WORK( * ),
     $                   Z( LDZ, * )

      SUBROUTINE CHPGVX( ITYPE, JOBZ, RANGE, UPLO, N, AP, BP, VL, VU,
     $                   IL, IU, ABSTOL, M, W, Z, LDZ, WORK, RWORK,
     $                   IWORK, IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, ITYPE, IU, LDZ, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            AP( * ), BP( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSPGVX/CHPGVX computes selected eigenvalues and, optionally, eigenvectors of a complex generalized symmetric-definite/Hermitian-definite eigenproblem, of the form $A*x = \lambda*B*x$, $A*Bx = \lambda*x$, or $B*A*x = \lambda*x$. Here A and B are assumed to be symmetric/Hermitian, stored in packed format, and B is also positive definite. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

Arguments

ITYPE (input) INTEGER
Specifies the problem type to be solved:
= 1: A*x = λ*B*x
= 2: A*B*x = λ*x
= 3: B*A*x = λ*x

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

RANGE (input) CHARACTER*1
= 'A': all eigenvalues will be found.
= 'V': all eigenvalues in the half-open interval (VL,VU] will be found.
= 'I': the ILth through IUth eigenvalues will be found.

UPLO (input) CHARACTER*1
= 'U': Upper triangles of A and B are stored;
= 'L': Lower triangles of A and B are stored.

N (input) INTEGER
The order of the matrices A and B. N $\geq$ 0.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The jth column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for 1$\leq$i$\leq$j;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for j$\leq$i$\leq$n.
On exit, the contents of AP are destroyed.

BP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The jth column of B is stored in the array BP as follows:
if UPLO = 'U', BP(i + (j−1)*j/2) = B(i,j) for 1$\leq$i$\leq$j;
if UPLO = 'L', BP(i + (j−1)*(2*n−j)/2) = B(i,j) for j$\leq$i$\leq$n.

On exit, the triangular factor U or L from the Cholesky factorization $B = U^H * U$ or $B = L * L^H$, in the same storage format as B.

VL,VU (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be searched for eigenvalues. VL < VU. Not referenced if RANGE = 'A' or 'I'.

IL,IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned. 1 $\leq$ IL $\leq$ IU $\leq$ N, if N > 0; IL = 1 and IU = 0 if N = 0. Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute error tolerance for the eigenvalues. An approximate eigenvalue is accepted as converged when it is determined to lie in an interval [a,b] of width less than or equal to ABSTOL + EPS*max(|a|,|b|), where EPS is the machine precision. If ABSTOL is less than or equal to zero, then EPS*$||T||_1$ will be used in its place, where T is the tridiagonal matrix obtained by reducing A to tridiagonal form.
Eigenvalues will be computed most accurately when ABSTOL is set to twice the underflow threshold 2*SLAMCH('S'), not zero. If this routine returns with INFO>0, indicating that some eigenvectors did not converge, try setting ABSTOL to 2*SLAMCH('S').

M (output) INTEGER
The total number of eigenvalues found. 0 $\leq$ M $\leq$ N.
If RANGE = 'A', M = N, and if RANGE = 'I', M = IU−IL+1.

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, N)
If JOBZ = 'N', then Z is not referenced.
If JOBZ = 'V', then if INFO = 0, the first M columns of Z contain the orthonormal eigenvectors of the matrix A corresponding to the selected eigenvalues, with the ith column of Z holding the eigenvector associated with W(i). The eigenvectors are normalized as follows:
if ITYPE = 1 or 2, $Z^H * B * Z = I$;
if ITYPE = 3, $Z^H * B^{-1} * Z = I$.

If an eigenvector fails to converge, then that column of Z contains the latest approximation to the eigenvector, and the index of the eigenvector is returned in IFAIL. Note: the user must ensure that at least max(1,M) columns are supplied in the array Z; if RANGE = 'V', the exact value of M is not known in advance and an upper bound must be used.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

WORK *SSPGVX* (workspace) REAL array, dimension (8*N)
CHPGVX (workspace) COMPLEX array, dimension (2*N)

RWORK *CHPGVX only* (workspace) REAL array, dimension (7*N)

IWORK (workspace) INTEGER array, dimension (5*N)

IFAIL (output) INTEGER array, dimension (N)
If JOBZ = 'V', then if INFO = 0, the first M elements of IFAIL are zero. If INFO > 0, then IFAIL contains the indices of the eigenvectors that failed to converge. If JOBZ = 'N', then IFAIL is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = -i, the i-th argument had an illegal value.
> 0: SPPTRF/CPPTRF or SSPEVX/CHPEVX returned an error code:

$\leq$ N: if INFO = i, SSPEVX/CHPEVX failed to converge; i eigenvectors failed to converge. Their indices are stored in array IFAIL.

> N: if INFO = N + i, for $1 \leq i \leq n$, then the leading minor of order i of B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

SSPRFS/CSPRFS/CHPRFS

```
      SUBROUTINE SSPRFS( UPLO, N, NRHS, AP, AFP, IPIV, B, LDB, X, LDX,
     $                   FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      INTEGER            IPIV( * ), IWORK( * )
      REAL               AFP( * ), AP( * ), B( LDB, * ), BERR( * ),
     $                   FERR( * ), WORK( * ), X( LDX, * )

      SUBROUTINE CSPRFS( UPLO, N, NRHS, AP, AFP, IPIV, B, LDB, X, LDX,
     $                   FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            AFP( * ), AP( * ), B( LDB, * ), WORK( * ),
     $                   X( LDX, * )

      SUBROUTINE CHPRFS( UPLO, N, NRHS, AP, AFP, IPIV, B, LDB, X, LDX,
     $                   FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            AFP( * ), AP( * ), B( LDB, * ), WORK( * ),
     $                   X( LDX, * )
```

Purpose

SSPRFS/CSPRFS/CHPRFS improves the computed solution to a system of linear equations when the coefficient matrix is real/complex/complex symmetric/-symmetric/Hermitian indefinite and packed, and provides error bounds and backward error estimates for the solution.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

AP (input) REAL/COMPLEX/COMPLEX array, dimension (N*(N+1)/2)
The upper or lower triangle of the symmetric/symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.

AFP (input) REAL/COMPLEX/COMPLEX array, dimension (N*(N+1)/2)
The factored form of the matrix A. AFP contains the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization A = U*D*U^T or A = L*D*L^T as computed by SSPTRF/CSPTRF or the factorization A = U*D*U^H or A = L*D*L^H as computed by CHPTRF, stored as a packed triangular matrix.

IPIV (input) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D as determined by SSPTRF/CSPTRF/CHPTRF.

B (input) REAL/COMPLEX/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (input/output)
REAL/COMPLEX/COMPLEX array, dimension (LDX,NRHS)
On entry, the solution matrix X, as computed by SSPTRS/CSPTRS/CHPTRS.
On exit, the improved solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SSPRFS* (workspace) REAL array, dimension (3*N)
CSPRFS/CHPRFS (workspace) COMPLEX array, dimension (2*N)

IWORK *SSPRFS only* (workspace) INTEGER array, dimension (N)

RWORK *CSPRFS/CHPRFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SSPSV/CSPSV/CHPSV

```
SUBROUTINE SSPSV( UPLO, N, NRHS, AP, IPIV, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDB, N, NRHS
INTEGER            IPIV( * )
REAL               AP( * ), B( LDB, * )

SUBROUTINE CSPSV( UPLO, N, NRHS, AP, IPIV, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDB, N, NRHS
INTEGER            IPIV( * )
COMPLEX            AP( * ), B( LDB, * )

SUBROUTINE CHPSV( UPLO, N, NRHS, AP, IPIV, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDB, N, NRHS
INTEGER            IPIV( * )
COMPLEX            AP( * ), B( LDB, * )
```

Purpose

SSPSV/CSPSV/CHPSV computes the solution to a real/complex/complex system of linear equations A*X = B, where A is an n-by-n symmetric (SSPSV/CSPSV) or Hermitian (CHPSV) matrix stored in packed format and X and B are n-by-nrhs matrices.

The diagonal pivoting method is used to factor A as

$A = U*D*U^T$ or $A = L*D*L^T$ (SSPSV/CSPSV) or

$A = U*D*U^H$ or $A = L*D*L^H$ (CHPSV),

where U (or L) is a product of permutation and unit upper (lower) triangular matrices, and D is symmetric (SSPSV/CSPSV) or Hermitian (CHPSV) and block diagonal with 1-by-1 and 2-by-2 diagonal blocks. The factored form of A is then used to solve the system of equations A*X = B.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N ≥ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS ≥ 0.

AP (input/output)
REAL/COMPLEX/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for 1≤i≤j;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for j≤i≤n.
On exit, the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $A = U*D*U^T$ or $A = L*D*L^T$ as computed by SSPTRF/CSPTRF or the factorization $A = U*D*U^H$ or $A = L*D*L^H$ as computed by CHPTRF, stored as a packed triangular matrix in the same storage format as A.

IPIV (output) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D, as determined by SSPTRF/CSPTRF/CHPTRF. If IPIV(k) > 0, then rows and columns k and IPIV(k) were interchanged, and D(k,k) is a 1-by-1 diagonal block. If UPLO = 'U' and IPIV(k) = IPIV(k−1) < 0, then rows and columns k−1 and −IPIV(k) were interchanged and D(k−1:k,k−1:k) is a 2-by-2 diagonal block. If UPLO = 'L' and IPIV(k) = IPIV(k+1) < 0, then rows and columns k+1 and −IPIV(k) were interchanged and D(k:k+1,k:k+1) is a 2-by-2 diagonal block.

B (input/output)
REAL/COMPLEX/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if INFO = 0, the n-by-nrhs solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, D(i,i) is exactly zero. The factorization has been completed, but the block diagonal matrix D is exactly singular, so the solution could not be computed.

SSPSVX/CSPSVX/CHPSVX

```
      SUBROUTINE SSPSVX( FACT, UPLO, N, NRHS, AP, AFP, IPIV, B, LDB, X,
     $                   LDX, RCOND, FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          FACT, UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * ), IWORK( * )
      REAL               AFP( * ), AP( * ), B( LDB, * ), BERR( * ),
     $                   FERR( * ), WORK( * ), X( LDX, * )

      SUBROUTINE CSPSVX( FACT, UPLO, N, NRHS, AP, AFP, IPIV, B, LDB, X,
     $                   LDX, RCOND, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          FACT, UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            AFP( * ), AP( * ), B( LDB, * ), WORK( * ),
     $                   X( LDX, * )

      SUBROUTINE CHPSVX( FACT, UPLO, N, NRHS, AP, AFP, IPIV, B, LDB, X,
     $                   LDX, RCOND, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          FACT, UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            AFP( * ), AP( * ), B( LDB, * ), WORK( * ),
     $                   X( LDX, * )
```

Purpose

SSPSVX/CSPSVX/CHPSVX uses the diagonal pivoting factorization to compute the solution to a real/complex/complex system of linear equations A∗X = B, where A is an n-by-n symmetric (SSPSVX/CSPSVX) or Hermitian (CHPSVX) matrix stored in packed format and X and B are n-by-nrhs matrices.

Error bounds on the solution and a condition estimate are also computed.

Description

The following steps are performed:

1. If FACT = 'N', the diagonal pivoting method is used to factor A as

   $A = U*D*U^T$ or $A = L*D*L^T$ (SSPSVX/CSPSVX) or

   $A = U*D*U^H$ or $A = L*D*L^H$ (CHPSVX),

   where U (or L) is a product of permutation and unit upper (lower) triangular matrices and D is symmetric and block diagonal with 1-by-1 and 2-by-2 diagonal blocks.

2. If some D(i,i)=0, so that D is exactly singular, then the routine returns with INFO = i. Otherwise, the factored form of A is used to estimate the condition number of the matrix A. If the reciprocal of the condition number is less than machine precision, INFO = N+1 is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.

3. The system of equations is solved for X using the factored form of A.

4. Iterative refinement is applied to improve the computed solution matrix and calculate error bounds and backward error estimates for it.

Arguments

FACT (input) CHARACTER∗1
Specifies whether or not the factored form of A has been supplied on entry.
= 'F': On entry, AFP and IPIV contain the factored form of A. AP, AFP and IPIV will not be modified.
= 'N': The matrix A will be copied to AFP and factored.

UPLO (input) CHARACTER∗1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. $N \geq 0$.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. $NRHS \geq 0$.

AP (input) REAL/COMPLEX/COMPLEX array, dimension (N∗(N+1)/2)
The upper or lower triangle of the symmetric/symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)∗j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)∗(2∗n−j)/2) = A(i,j) for $j \leq i \leq n$.

AFP (input or output) REAL/COMPLEX/COMPLEX array, dimension (N∗(N+1)/2)
If FACT = 'F', then AFP is an input argument and on entry contains the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $A = U*D*U^T$ or $A = L*D*L^T$ as computed by SSPTRF/CSPTRF or the factorization $A = U*D*U^H$ or $A = L*D*L^H$ as computed by CHPTRF, stored as a packed triangular matrix in the same storage format as A.
If FACT = 'N', then AFP is an output argument and on exit returns the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization.

IPIV (input or output) INTEGER array, dimension (N)
If FACT = 'F', then IPIV is an input argument and on entry contains details of the interchanges and the block structure of D, as determined by SSPTRF/CSPTRF/CHPTRF.

If IPIV(k) > 0, then rows and columns k and IPIV(k) were interchanged and D(k,k) is a 1-by-1 diagonal block.
If UPLO = 'U' and IPIV(k) = IPIV(k−1) < 0, then rows and columns k−1 and −IPIV(k) were interchanged and D(k−1:k,k−1:k) is a 2-by-2 diagonal block. If UPLO = 'L' and IPIV(k) = IPIV(k+1) < 0, then rows and columns k+1 and −IPIV(k) were interchanged and D(k:k+1,k:k+1) is a 2-by-2 diagonal block.
If FACT = 'N', then IPIV is an output argument and on exit contains details of the interchanges and the block structure of D, as determined by SSPTRF/CSPTRF/CHPTRF.

B (input) REAL/COMPLEX/COMPLEX array, dimension (LDB,NRHS)
The n-by-nrhs right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,N).

X (output)
REAL/COMPLEX/COMPLEX array, dimension (LDX,NRHS)
If INFO = 0 or INFO = N+1, the N-by-NRHS solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX ≥ max(1,N).

RCOND (output) REAL
The estimate of the reciprocal condition number of the matrix A. If RCOND is less than the machine precision (in particular, if RCOND = 0), the matrix is singular to working precision. This condition is indicated by a return code of INFO > 0.

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SSPSVX* (workspace) REAL array, dimension (3∗N)
CSPSVX/CHPSVX (workspace) COMPLEX array, dimension (2∗N)

IWORK *SSPSVX only* (workspace) INTEGER array, dimension (N)

RWORK *CSPSVX/CHPSVX only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0 and ≤ N: if INFO = i, D(i,i) is exactly zero. The factorization has been completed, but the block diagonal matrix D is exactly singular, so the solution and error bounds could not be computed. RCOND = 0 is returned.
= N+1: D is nonsingular, but RCOND is less than machine precision, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of RCOND would suggest.

SSPTRD/CHPTRD

```
SUBROUTINE SSPTRD( UPLO, N, AP, D, E, TAU, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
REAL               AP( * ), D( * ), E( * ), TAU( * )

SUBROUTINE CHPTRD( UPLO, N, AP, D, E, TAU, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
REAL               D( * ), E( * )
COMPLEX            AP( * ), TAU( * )
```

Purpose

SSPTRD/CHPTRD reduces a real/complex symmetric/Hermitian matrix A stored in packed form to real symmetric tridiagonal form T by an orthogonal/unitary similarity transformation: $Q^H*A*Q = T$.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N ≥ 0.

AP (input/output) REAL/COMPLEX array, dimension (N∗(N+1)/2)
On entry, the upper or lower triangle of the symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)∗j/2) = A(i,j) for 1≤i≤j;
if UPLO = 'L', AP(i + (j−1)∗(2∗n−j)/2) = A(i,j) for j≤i≤n.
On exit, if UPLO = 'U', the diagonal and first superdiagonal of A are overwritten by the corresponding elements of the tridiagonal matrix T, and the elements above the first superdiagonal, with the array TAU, represent the orthogonal/unitary matrix Q as a product of elementary reflectors; if UPLO = 'L', the diagonal and first subdiagonal of A are

overwritten by the corresponding elements of the tridiagonal matrix T, and the elements below the first subdiagonal, with the array TAU, represent the orthogonal/unitary matrix Q as a product of elementary reflectors.

D (output) REAL array, dimension (N)
The diagonal elements of the tridiagonal matrix T: D(i) = A(i,i).

E (output) REAL array, dimension (N−1)
The off-diagonal elements of the tridiagonal matrix T:
E(i) = A(i,i+1) if UPLO = 'U'; E(i) = A(i+1,i) if UPLO = 'L'.

TAU (output) REAL/COMPLEX array, dimension (N−1)
The scalar factors of the elementary reflectors.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SSPTRF/CSPTRF/CHPTRF

```
SUBROUTINE SSPTRF( UPLO, N, AP, IPIV, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
INTEGER            IPIV( * )
REAL               AP( * )

SUBROUTINE CSPTRF( UPLO, N, AP, IPIV, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
INTEGER            IPIV( * )
COMPLEX            AP( * )

SUBROUTINE CHPTRF( UPLO, N, AP, IPIV, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
INTEGER            IPIV( * )
COMPLEX            AP( * )
```

Purpose

SSPTRF/CSPTRF/CHPTRF computes the factorization of a real/complex/complex symmetric/symmetric/Hermitian matrix A stored in packed format, using the diagonal pivoting method. The form of the factorization is

$$A = U*D*U^T \text{ or } A = L*D*L^T \text{ (SSPTRF/CSPTRF), or}$$

$$A = U*D*U^H \text{ or } A = L*D*L^H \text{ (CHPTRF),}$$

where U (or L) is a product of permutation and unit upper (lower) triangular matrices, and D is symmetric (SSPTRF/CSPTRF) or Hermitian (CHPTRF) and block diagonal with 1-by-1 and 2-by-2 diagonal blocks.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

AP (input/output)
REAL/COMPLEX/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangle of the symmetric/symmetric/Hermitian matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.
On exit, the block diagonal matrix D and the multipliers used to obtain the factor U or L, stored as a packed triangular matrix overwriting A.

IPIV (output) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D. If IPIV(k) > 0, then rows and columns k and IPIV(k) were interchanged and D(k,k) is a 1-by-1 diagonal block. If UPLO = 'U' and IPIV(k) = IPIV(k−1) < 0, then rows and columns k−1 and −IPIV(k) were interchanged and D(k−1:k,k−1:k) is a 2-by-2 diagonal block. If UPLO = 'L' and IPIV(k) = IPIV(k+1) < 0, then rows and columns k+1 and −IPIV(k) were interchanged and D(k:k+1,k:k+1) is a 2-by-2 diagonal block.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, D(i,i) is exactly zero. The factorization has been completed, but the block diagonal matrix D is exactly singular, and division by zero will occur if it is used to solve a system of equations.

SSPTRI/CSPTRI/CHPTRI

```
SUBROUTINE SSPTRI( UPLO, N, AP, IPIV, WORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
INTEGER            IPIV( * )
REAL               AP( * ), WORK( * )
```

```
SUBROUTINE CSPTRI( UPLO, N, AP, IPIV, WORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
INTEGER            IPIV( * )
COMPLEX            AP( * ), WORK( * )

SUBROUTINE CHPTRI( UPLO, N, AP, IPIV, WORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, N
INTEGER            IPIV( * )
COMPLEX            AP( * ), WORK( * )
```

Purpose

SSPTRI/CSPTRI/CHPTRI computes the inverse of a real/complex/complex symmetric/symmetric/Hermitian indefinite matrix A in packed storage using the factorization $A = U*D*U^T$ or $A = L*D*L^T$ computed by SSPTRF/CSPTRF or the factorization $A = U*D*U^H$ or $A = L*D*L^H$ computed by CHPTRF.

Arguments

UPLO (input) CHARACTER*1
Specifies whether the details of the factorization are stored as an upper or lower triangular matrix.
= 'U': Upper triangular, form is $A = U*D*U^T$ (SSPTRI/CSPTRI) or $A = U*D*U^H$ (CHPTRI);
= 'L': Lower triangular, form is $A = L*D*L^T$ (SSPTRI/CSPTRI) or $A = L*D*L^H$ (CHPTRI).

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

AP (input/output)
REAL/COMPLEX/COMPLEX array, dimension (N*(N+1)/2)
On entry, the block diagonal matrix D and the multipliers used to obtain the factor U or L as computed by SSPTRF/CSPTRF/CHPTRF, stored as a packed triangular matrix.
On exit, if INFO = 0, the (symmetric/symmetric/Hermitian) inverse of the original matrix, stored as a packed triangular matrix. The j^{th} column of A^{-1} is stored in the array AP as follows:
if UPLO = 'U', $AP(i + (j-1)*j/2) = A^{-1}(i,j)$ for $1 \leq i \leq j$;
if UPLO = 'L', $AP(i + (j-1)*(2*n-j)/2) = A^{-1}(i,j)$ for $j \leq i \leq n$.

IPIV (input) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D as determined by SSPTRF/CSPTRF/CHPTRF.

WORK (workspace) REAL/COMPLEX/COMPLEX array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = $-i$, the i^{th} argument had an illegal value.
> 0: if INFO = i, D(i,i) = 0; the matrix is singular and its inverse could not be computed.

SSPTRS/CSPTRS/CHPTRS

```
SUBROUTINE SSPTRS( UPLO, N, NRHS, AP, IPIV, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDB, N, NRHS
INTEGER            IPIV( * )
REAL               AP( * ), B( LDB, * )

SUBROUTINE CSPTRS( UPLO, N, NRHS, AP, IPIV, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDB, N, NRHS
INTEGER            IPIV( * )
COMPLEX            AP( * ), B( LDB, * )

SUBROUTINE CHPTRS( UPLO, N, NRHS, AP, IPIV, B, LDB, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDB, N, NRHS
INTEGER            IPIV( * )
COMPLEX            AP( * ), B( LDB, * )
```

Purpose

SSPTRS/CSPTRS/CHPTRS solves a system of linear equations $A*X = B$ with a real/complex/complex symmetric/symmetric/Hermitian matrix A stored in packed format using the factorization $A = U*D*U^T$ or $A = L*D*L^T$ computed by SSPTRF/CSPTRF or the factorization $A = U*D*U^H$ or $A = L*D*L^H$ computed by CHPTRF.

Arguments

UPLO (input) CHARACTER*1
Specifies whether the details of the factorization are stored as an upper or lower triangular matrix.
= 'U': Upper triangular, form is $A = U*D*U^T$ (SSPTRS/CSPTRS) or $A = U*D*U^H$ (CHPTRS);
= 'L': Lower triangular, form is $A = L*D*L^T$ (SSPTRS/CSPTRS) or $A = L*D*L^H$ (CHPTRS).

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. $NRHS \geq 0$.

AP (input) REAL/COMPLEX/COMPLEX array, dimension (N*(N+1)/2)
The block diagonal matrix D and the multipliers used to obtain the factor U or L as computed by SSPTRF/CSPTRF/CHPTRF, stored as a packed triangular matrix.

IPIV (input) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D as determined by SSPTRF/CSPTRF/CHPTRF.

B (input/output)
REAL/COMPLEX/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = $-$i, the i^{th} argument had an illegal value.

SSTEBZ

```
      SUBROUTINE SSTEBZ( RANGE, ORDER, N, VL, VU, IL, IU, ABSTOL, D, E,
     $                   M, NSPLIT, W, IBLOCK, ISPLIT, WORK, IWORK,
     $                   INFO )
      CHARACTER          ORDER, RANGE
      INTEGER            IL, INFO, IU, M, N, NSPLIT
      REAL               ABSTOL, VL, VU
      INTEGER            IBLOCK( * ), ISPLIT( * ), IWORK( * )
      REAL               D( * ), E( * ), W( * ), WORK( * )
```

Purpose

SSTEBZ computes the eigenvalues of a symmetric tridiagonal matrix T. The user may ask for all eigenvalues, all eigenvalues in the half-open interval (VL, VU], or the IL^{th} through IU^{th} eigenvalues.

To avoid overflow, the matrix must be scaled so that its largest element is no greater than overflow$^{1/2}$* underflow$^{1/4}$ in absolute value, and for greater accuracy, it should not be much smaller than that.

Arguments

RANGE (input) CHARACTER*1
= 'A': ("All") all eigenvalues will be found.
= 'V': ("Value") all eigenvalues in the half-open interval (VL, VU] will be found.
= 'I': ("Index") the IL^{th} through IU^{th} eigenvalues will be found.

ORDER (input) CHARACTER*1
= 'B': ("By Block") the eigenvalues will be grouped by split-off block (see IBLOCK, ISPLIT) and ordered from smallest to largest within the block.
= 'E': ("Entire matrix") the eigenvalues for the entire matrix will be ordered from smallest to largest.

N (input) INTEGER
The order of the tridiagonal matrix T. N $\geq$ 0.

VL, VU (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be searched for eigenvalues. Eigenvalues less than or equal to VL, or greater than VU, will not be returned. VL < VU.
Not referenced if RANGE = 'A' or 'I'.

IL, IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned.
$1 \leq IL \leq IU \leq N$, if $N > 0$; IL = 1 and IU = 0 if N = 0.
Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute tolerance for the eigenvalues. An eigenvalue (or cluster) is considered to be located if it has been determined to lie in an interval whose width is ABSTOL or less. If ABSTOL is less than or equal to zero, then EPS*$\|T\|_1$ will be used in its place, where EPS is the machine precision.
Eigenvalues will be computed most accurately when ABSTOL is set to twice the underflow threshold 2*SLAMCH('S'), not zero.

D (input) REAL array, dimension (N)
The n diagonal elements of the tridiagonal matrix T.

E (input) REAL array, dimension (N$-$1)
The (n$-$1) off-diagonal elements of the tridiagonal matrix T.

M (output) INTEGER
The actual number of eigenvalues found ($0 \leq M \leq N$). (See also the description of INFO = 2,3.)

NSPLIT (output) INTEGER
The number of diagonal blocks in the matrix T. ($1 \leq NSPLIT \leq N$).

W (output) REAL array, dimension (N)
On exit, the first M elements of W will contain the eigenvalues. (SSTEBZ may use the remaining N$-$M elements as workspace.)

IBLOCK (output) INTEGER array, dimension (N)
At each row/column j where E(j) is zero or small, the matrix T is considered to split into a block diagonal matrix. On exit, if INFO = 0, IBLOCK(i) specifies to which block (from 1 to NSPLIT) the eigenvalue W(i) belongs. If INFO > 0, IBLOCK(i) is set to a negative block number if the i^{th} eigenvalue did not converge. (SSTEBZ may use the remaining N$-$M elements as workspace.)

ISPLIT (output) INTEGER array, dimension (N)
The splitting points, at which T breaks up into submatrices. The

first submatrix consists of rows/columns 1 to ISPLIT(1), the second of rows/columns ISPLIT(1)+1 through ISPLIT(2), etc., and the NSPLITth consists of rows/columns ISPLIT(NSPLIT−1)+1 through ISPLIT(NSPLIT)=N. (Only the first NSPLIT elements will actually be used, but since the user cannot know a priori what value NSPLIT will have, N words must be reserved for ISPLIT.)

WORK (workspace) REAL array, dimension (4*N)

IWORK (workspace) INTEGER array, dimension (3*N)

INFO (output) INTEGER
- = 0: successful exit
- < 0: if INFO = −i, the ith argument had an illegal value.
- > 0: some or all of the eigenvalues failed to converge or were not computed:
  - = 1 or 3: Bisection failed to converge for some eigenvalues; these eigenvalues are flagged by a negative block number. The effect is that the eigenvalues may not be as accurate as the absolute and relative tolerances. This is generally caused by unexpectedly inaccurate arithmetic.
  - = 2 or 3: RANGE='I' only: Not all of the eigenvalues IL:IU were found.
    Effect: M < IU+1−IL.
    Cause: non-monotonic arithmetic, causing the Sturm sequence to be non-monotonic.
    Cure: recalculate, using RANGE='A', and pick out eigenvalues IL:IU. In some cases, increasing the internal parameter FUDGE may make things work.
  - = 4: RANGE='I', and the Gershgorin interval initially used was too small. No eigenvalues were computed.
    Probable cause: your machine has sloppy floating-point arithmetic.
    Cure: Increase the internal parameter FUDGE, recompile, and try again.

SSTEDC/CSTEDC

```
      SUBROUTINE SSTEDC( COMPZ, N, D, E, Z, LDZ, WORK, LWORK, IWORK,
     $                   LIWORK, INFO )
      CHARACTER          COMPZ
      INTEGER            INFO, LDZ, LIWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               D( * ), E( * ), WORK( * ), Z( LDZ, * )
```

```
      SUBROUTINE CSTEDC( COMPZ, N, D, E, Z, LDZ, WORK, LWORK, RWORK,
     $                   LRWORK, IWORK, LIWORK, INFO )
      CHARACTER          COMPZ
      INTEGER            INFO, LDZ, LIWORK, LRWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               D( * ), E( * ), RWORK( * )
      COMPLEX            WORK( * ), Z( LDZ, * )
```

Purpose

SSTEDC/CSTEDC computes all eigenvalues and, optionally, eigenvectors of a symmetric tridiagonal matrix using the divide and conquer method.

The eigenvectors of a full or band real/complex symmetric/Hermitian matrix can also be found if SSYTRD/CHETRD or SSPTRD/CHPTRD or SSBTRD/CHBTRD has been used to reduce this matrix to tridiagonal form.

This code makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

Arguments

COMPZ (input) CHARACTER*1
- = 'N': Compute eigenvalues only.
- = 'I': Compute eigenvectors of tridiagonal matrix also.
- = 'V': Compute eigenvectors of original symmetric/Hermitian matrix also. On entry, Z contains the orthogonal/unitary matrix used to reduce the original matrix to tridiagonal form.

N (input) INTEGER
The dimension of the symmetric tridiagonal matrix. N ≥ 0.

D (input/output) REAL array, dimension (N)
On entry, the diagonal elements of the tridiagonal matrix.
On exit, if INFO = 0, the eigenvalues in ascending order.

E (input/output) REAL array, dimension (N−1)
On entry, the subdiagonal elements of the tridiagonal matrix.
On exit, E has been destroyed.

Z (input/output) REAL/COMPLEX array, dimension (LDZ,N)
On entry, if COMPZ = 'V', then Z contains the orthogonal/unitary matrix used in the reduction to tridiagonal form.
On exit, if INFO = 0, then if COMPZ = 'V', Z contains the orthonormal eigenvectors of the original symmetric/Hermitian matrix, and if COMPZ = 'I', Z contains the orthonormal eigenvectors of the symmetric tridiagonal matrix.
If COMPZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ ≥ 1.

If eigenvectors are desired, then LDZ $\geq$ max(1,N).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
SSTEDC
If COMPZ = 'N' or N $\leq$ 1 then LWORK $\geq$ 1.
If COMPZ = 'V' and N > 1 then LWORK $\geq$ (1 + 3*N + 2*N*lg N + 3*N^2), where lg N = smallest integer k such that $2^k \geq$N. If COMPZ = 'I' and N > 1 then LWORK $\geq$ (1 + 3*N + 2*N*lg N + 2*N^2).
CSTEDC
If COMPZ = 'N' or 'I', or N $\leq$ 1, LWORK $\geq$ 1.
If COMPZ = 'V' and N > 1, LWORK $\geq$ N*N.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CSTEDC only* (workspace/output) REAL array, dimension (LRWORK) On exit, if INFO = 0, RWORK(1) returns the optimal LRWORK.

LRWORK *CSTEDC only* (input) INTEGER
The dimension of the array RWORK.
If COMPZ = 'N' or N $\leq$ 1, LRWORK $\geq$ 1.
If COMPZ = 'V' and N > 1, LRWORK $\geq$ (1 + 3*N + 2*N*lg N + 3*N^2), where lg N = smallest integer k such that $2^k \geq$N.
If COMPZ = 'I' and N > 1, LRWORK $\geq$ (1 + 3*N + 2*N*lg N + 3*N^2).

If LRWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the RWORK array, returns this value as the first entry of the RWORK array, and no error message related to LRWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array IWORK.
If COMPZ = 'N' or N $\leq$ 1 then LIWORK $\geq$ 1.
If COMPZ = 'V' and N > 1 then LIWORK $\geq$ (6 + 6*N + 5*N*lg N).
If COMPZ = 'I' and N > 1 then LIWORK $\geq$ 3 + 5*N.

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the ith argument had an illegal value.
> 0: The algorithm failed to compute an eigenvalue while working on the submatrix lying in rows and columns INFO/(N+1) through mod(INFO,N+1)

SSTEGR/CSTEGR

```
      SUBROUTINE SSTEGR( JOBZ, RANGE, N, D, E, VL, VU, IL, IU, ABSTOL,
     $                   M, W, Z, LDZ, ISUPPZ, WORK, LWORK, IWORK,
     $                   LIWORK, INFO )
      CHARACTER          JOBZ, RANGE
      INTEGER            IL, INFO, IU, LDZ, LWORK, LIWORK, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            ISUPPZ( * ), IWORK( * )
      REAL               D( * ), E( * ), W( * ), WORK( * ), Z( LDZ, * )

      SUBROUTINE CSTEGR( JOBZ, RANGE, N, D, E, VL, VU, IL, IU, ABSTOL,
     $                   M, W, Z, LDZ, ISUPPZ, WORK, LWORK, IWORK,
     $                   LIWORK, INFO )
      CHARACTER          JOBZ, RANGE
      INTEGER            IL, INFO, IU, LDZ, LWORK, LIWORK, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            ISUPPZ( * ), IWORK( * )
      REAL               D( * ), E( * ), W( * ), WORK( * )
      COMPLEX            Z( LDZ, * )
```

Purpose

SSTEGR/CSTEGR computes selected eigenvalues and, optionally, eigenvectors of a real symmetric tridiagonal matrix T. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues. The eigenvalues are computed by the dqds algorithm, while orthogonal eigenvectors are computed from various "good" LDL^T representations (also known as Relatively Robust Representations). Gram-Schmidt orthogonalization is avoided as far as possible. More specifically, the various steps of the algorithm are as follows. For the ith unreduced block of T,

(a) Compute $T - \sigma_i = L_i D_i L_i^T$, such that $L_i D_i L_i^T$ is a relatively robust representation,

(b) Compute the eigenvalues, λ_j, of $L_i D_i L_i^T$ to high relative accuracy by the dqds algorithm,

(c) If there is a cluster of close eigenvalues, "choose" σ_i close to the cluster, and go to step (a),

(d) Given the approximate eigenvalue λ_j of $L_i D_i L_i^T$, compute the corresponding eigenvector by forming a rank-revealing twisted factorization.

The desired accuracy of the output can be specified by the input parameter ABSTOL.

Note 1: Currently SSTEGR/CSTEGR is only set up to find ALL the n eigenvalues and eigenvectors of T in $O(n^2)$ time.

Note 2 : Currently the routine SSTEIN/CSTEIN is called when an appropriate σ_i cannot be chosen in step (c) above. SSTEIN/CSTEIN invokes modified Gram-Schmidt when eigenvalues are close.

Note 3 : SSTEGR/CSTEGR works only on machines which follow ieee-754 floating-point standard in their handling of infinities and NaNs. Normal execution of SSTEGR/CSTEGR may create NaNs and infinities and hence may abort due to a floating point exception in environments which do not conform to the ieee standard.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

RANGE (input) CHARACTER*1
= 'A': all eigenvalues will be found;
= 'V': all eigenvalues in the half-open interval (VL,VU] will be found;
= 'I': the ILth through IUth eigenvalues will be found.
Only RANGE = 'A' is currently supported.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

D (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the tridiagonal matrix T.
On exit, D is overwritten.

E (input/output) REAL array, dimension (N)
On entry, the (n−1) subdiagonal elements of the tridiagonal matrix T in elements 1 to N−1 of E; E(N) need not be set.
On exit, E is overwritten.

VL,VR (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be searched for eigenvalues. VL < VU.
Not referenced if RANGE = 'A' or 'I'.

IL,IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned. 1 $\leq$ IL $\leq$ IU $\leq$ N, if N > 0; IL = 1 and IU = 0 if N = 0.
Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute error tolerance for the eigenvalues/eigenvectors.
IF JOBZ = 'V', the eigenvalues and eigenvectors output have residual norms bounded by ABSTOL, and the dot products between different eigenvectors are bounded by ABSTOL. If ABSTOL < N*EPS*$\|T\|_1$, then N*EPS*$\|T\|_1$ will be used in its place, where EPS is the machine precision. The eigenvalues are computed to an accuracy of EPS*$\|T\|_1$ irrespective of ABSTOL. If high relative accuracy is important, set ABSTOL to xLAMCH('S').

M (output) INTEGER
The total number of eigenvalues found. 0 $\leq$ M $\leq$ N.
If RANGE = 'A', M = N, and if RANGE = 'I', M = IU−IL+1.

W (output) REAL array, dimension (N)
The first M elements contain the selected eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, max(1,M))
If JOBZ = 'V', then if INFO = 0, the first M columns of Z contain the orthonormal eigenvectors of the matrix T corresponding to the selected eigenvalues, with the ith column of Z holding the eigenvector associated with W(i).
If JOBZ = 'N', then Z is not referenced.
Note: the user must ensure that at least max(1,M) columns are supplied in the array Z; if RANGE = 'V', the exact value of M is not known in advance and an upper bound must be used.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

ISUPPZ (output) INTEGER array, dimension (2*max(1,M))
The support of the eigenvectors in Z, i.e., the indices indicating the nonzero elements in Z. The ith eigenvector is nonzero only in elements ISUPPZ(2*i−1) through ISUPPZ(2*i).

WORK (workspace/output) REAL array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal (and minimal) LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ max(1,18*N).

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array IWORK. LIWORK $\geq$ max(1, 10*N).

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = 1, internal error in SLARRE
if INFO = 2, internal error in SLARRV/CLARRV

SSTEIN/CSTEIN

```
      SUBROUTINE SSTEIN( N, D, E, M, W, IBLOCK, ISPLIT, Z, LDZ, WORK,
     $                   IWORK, IFAIL, INFO )
      INTEGER            INFO, LDZ, M, N
      INTEGER            IBLOCK( * ), IFAIL( * ), ISPLIT( * ),
     $                   IWORK( * )
      REAL               D( * ), E( * ), W( * ), WORK( * ), Z( LDZ, * )

      SUBROUTINE CSTEIN( N, D, E, M, W, IBLOCK, ISPLIT, Z, LDZ, WORK,
     $                   IWORK, IFAIL, INFO )
      INTEGER            INFO, LDZ, M, N
      INTEGER            IBLOCK( * ), IFAIL( * ), ISPLIT( * ),
     $                   IWORK( * )
      REAL               D( * ), E( * ), W( * ), WORK( * )
      COMPLEX            Z( LDZ, * )
```

Purpose

SSTEIN/CSTEIN computes the eigenvectors of a real symmetric tridiagonal matrix T corresponding to specified eigenvalues, using inverse iteration.

The maximum number of iterations allowed for each eigenvector is specified by an internal parameter MAXITS (currently set to 5).

CSTEIN only
Although the eigenvectors are real, they are stored in a complex array, which may be passed to CUNMTR or CUPMTR for back transformation to the eigenvectors of a complex Hermitian matrix which was reduced to tridiagonal form.

Arguments

N (input) INTEGER
The order of the matrix. N $\geq$ 0.

D (input) REAL array, dimension (N)
The n diagonal elements of the tridiagonal matrix T.

E (input) REAL array, dimension (N)
The (n−1) subdiagonal elements of the tridiagonal matrix T, stored in elements 1 to n−1; E(n) need not be set.

M (input) INTEGER
The number of eigenvectors to be found. $0 \leq M \leq N$.

W (input) REAL array, dimension (N)
The first M elements of W contain the eigenvalues for which eigenvectors are to be computed. The eigenvalues should be grouped by split-off block and ordered from smallest to largest within the block. (The output array W from SSTEBZ with ORDER = 'B' is expected here.)

IBLOCK (input) INTEGER array, dimension (N)
The submatrix indices associated with the corresponding eigenvalues in W; IBLOCK(i)=1 if eigenvalue W(i) belongs to the first submatrix from the top, =2 if W(i) belongs to the second submatrix, etc. (The output array IBLOCK from SSTEBZ is expected here.)

ISPLIT (input) INTEGER array, dimension (N)
The splitting points, at which T breaks up into submatrices. The first submatrix consists of rows/columns 1 to ISPLIT(1), the second of rows/columns ISPLIT(1)+1 through ISPLIT(2), etc. (The output array ISPLIT from SSTEBZ is expected here.)

Z (output) REAL/COMPLEX array, dimension (LDZ, M)
The computed eigenvectors. The eigenvector associated with the eigenvalue W(i) is stored in the i^{th} column of Z. Any vector which fails to converge is set to its current iterate after MAXITS iterations.
CSTEIN only
The imaginary parts of the eigenvectors are set to zero.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ max(1,N).

WORK (workspace) REAL array, dimension (5*N)

IWORK (workspace) INTEGER array, dimension (N)

IFAIL (output) INTEGER array, dimension (M)
On normal exit, all elements of IFAIL are zero. If one or more eigenvectors fail to converge after MAXITS iterations, then their indices are stored in array IFAIL.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

> 0: if INFO = i, then i eigenvectors failed to converge in MAXITS iterations. Their indices are stored in array IFAIL.

SSTEQR/CSTEQR

```
SUBROUTINE SSTEQR( COMPZ, N, D, E, Z, LDZ, WORK, INFO )
CHARACTER          COMPZ
INTEGER            INFO, LDZ, N
REAL               D( * ), E( * ), WORK( * ), Z( LDZ, * )

SUBROUTINE CSTEQR( COMPZ, N, D, E, Z, LDZ, WORK, INFO )
CHARACTER          COMPZ
INTEGER            INFO, LDZ, N
REAL               D( * ), E( * ), WORK( * )
COMPLEX            Z( LDZ, * )
```

Purpose

SSTEQR/CSTEQR computes all eigenvalues and, optionally, eigenvectors of a symmetric tridiagonal matrix using the implicit QL or QR method. The eigenvectors of a full or band symmetric or Hermitian matrix can also be found if SSYTRD/CHETRD, SSPTRD/CHPTRD, or SSBTRD/CHBTRD has been used to reduce this matrix to tridiagonal form.

Arguments

COMPZ (input) CHARACTER*1
= 'N': Compute eigenvalues only.
= 'V': Compute eigenvalues and eigenvectors of the original symmetric/Hermitian matrix. On entry, Z must contain the orthogonal/unitary matrix used to reduce the original matrix to tridiagonal form.
= 'I': Compute eigenvalues and eigenvectors of the tridiagonal matrix. Z is initialized to the identity matrix.

N (input) INTEGER
The order of the matrix. N $\geq$ 0.

D (input/output) REAL array, dimension (N)
On entry, the diagonal elements of the tridiagonal matrix.
On exit, if INFO = 0, the eigenvalues in ascending order.

E (input/output) REAL array, dimension (N−1)
On entry, the (n−1) subdiagonal elements of the tridiagonal matrix.
On exit, E has been destroyed.

Z (input/output) REAL/COMPLEX array, dimension (LDZ, N)
On entry, if COMPZ = 'V', then Z contains the orthogonal/unitary matrix used in the reduction to tridiagonal form.
On exit, if INFO = 0, then if COMPZ = 'V', Z contains the orthonormal eigenvectors of the original symmetric/Hermitian matrix, and if COMPZ = 'I', Z contains the orthonormal eigenvectors of the symmetric tridiagonal matrix.
If COMPZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if eigenvectors are desired, then LDZ $\geq$ max(1,N).

WORK (workspace) REAL array, dimension (max(1,2*N−2))
If COMPZ = 'N', then WORK is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: the algorithm has failed to find all the eigenvalues in a total of 30*N iterations; if INFO = i, then i elements of E have not converged to zero; on exit, D and E contain the elements of a symmetric tridiagonal matrix which is orthogonally/unitarily similar to the original matrix.

SSTERF

```
SUBROUTINE SSTERF( N, D, E, INFO )
INTEGER            INFO, N
REAL               D( * ), E( * )
```

Purpose

SSTERF computes all eigenvalues of a symmetric tridiagonal matrix using the Pal-Walker-Kahan variant of the QL or QR algorithm.

Arguments

N (input) INTEGER
The order of the matrix. N $\geq$ 0.

D (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the tridiagonal matrix.
On exit, if INFO = 0, D contains the eigenvalues in ascending order.

E (input/output) REAL array, dimension (N−1)
On entry, the (n−1) subdiagonal elements of the tridiagonal matrix.
On exit, E has been destroyed.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

> 0: the algorithm has failed to find all the eigenvalues in a total of 30*N iterations; if INFO = i, then i elements of E have not converged to zero.

SSTEV

```
      SUBROUTINE SSTEV( JOBZ, N, D, E, Z, LDZ, WORK, INFO )
      CHARACTER          JOBZ
      INTEGER            INFO, LDZ, N
      REAL               D( * ), E( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSTEV computes all eigenvalues and, optionally, eigenvectors of a real symmetric tridiagonal matrix A.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

D (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the tridiagonal matrix A.
On exit, if INFO = 0, the eigenvalues in ascending order.

E (input/output) REAL array, dimension (N)
On entry, the (n−1) subdiagonal elements of the tridiagonal matrix A, stored in elements 1 to n−1 of E; E(n) need not be set, but is used by the routine.
On exit, the contents of E are destroyed.

Z (output) REAL array, dimension (LDZ,N)
If JOBZ = 'V', then if INFO = 0, Z contains the orthonormal eigenvectors of the matrix A, with the i^{th} column of Z holding the eigenvector associated with the eigenvalue returned in D(i).
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

WORK (workspace) REAL array, dimension (max(1,2*N−2))
If JOBZ = 'N', WORK is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the algorithm failed to converge; i elements of E did not converge to zero.

SSTEVD

```
      SUBROUTINE SSTEVD( JOBZ, N, D, E, Z, LDZ, WORK, LWORK, IWORK,
     $                   LIWORK, INFO )
      CHARACTER          JOBZ
      INTEGER            INFO, LDZ, LIWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               D( * ), E( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSTEVD computes all eigenvalues and, optionally, eigenvectors of a real symmetric tridiagonal matrix. If eigenvectors are desired, it uses a divide and conquer algorithm.

The divide and conquer algorithm makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

N (input) INTEGER
The order of the matrix. N $\geq$ 0.

D (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the tridiagonal matrix A.
On exit, if INFO = 0, the eigenvalues in ascending order.

E (input/output) REAL array, dimension (N)
On entry, the (n−1) subdiagonal elements of the tridiagonal matrix A, stored in elements 1 to n−1 of E; E(n) need not be set, but is used by the routine.
On exit, the contents of E are destroyed.

Z (output) REAL array, dimension (LDZ, N)
If JOBZ = 'V', then if INFO = 0, Z contains the orthonormal eigenvectors of the matrix A, with the i^{th} column of Z holding the eigenvector associated with D(i).
If JOBZ = 'N', then Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ ≥ 1, and if JOBZ = 'V', LDZ ≥ max(1,N).

WORK (workspace/output) REAL array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
If JOBZ = 'N' or N ≤ 1 then LWORK ≥ 1.
If JOBZ = 'V' and N > 1 then LWORK ≥ (1 + 3*N + 2*N*lg N + 2*N^2), where lg N = smallest integer k such that $2^k \geq$ N).

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array IWORK.
If JOBZ = 'N' or N ≤ 1 then LIWORK ≥ 1. If JOBZ = 'V' and N > 1 then LIWORK ≥ 3+5*N.

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the ith argument had an illegal value.
> 0: if INFO = i, the algorithm failed to converge; i off-diagonal elements of E did not converge to zero.

SSTEVR

```
      SUBROUTINE SSTEVR( JOBZ, RANGE, N, D, E, VL, VU, IL, IU, ABSTOL,
     $                   M, W, Z, LDZ, ISUPPZ, WORK, LWORK, IWORK,
     $                   LIWORK, INFO )
      CHARACTER          JOBZ, RANGE
      INTEGER            IL, INFO, IU, LDZ, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            ISUPPZ( * ), IWORK( * )
      REAL               D( * ), E( * ), W( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSTEVR computes selected eigenvalues and, optionally, eigenvectors of a real symmetric tridiagonal matrix T. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

Whenever possible, SSTEVR calls SSTEGR to compute the eigenspectrum using Relatively Robust Representations. SSTEGR computes eigenvalues by the dqds algorithm, while orthogonal eigenvectors are computed from various "good" LDL^T representations (also known as Relatively Robust Representations). Gram-Schmidt orthogonalization is avoided as far as possible. More specifically, the various steps of the algorithm are as follows. For the ith unreduced block of T,

(a) Compute $T - \sigma_i = L_i D_i L_i^T$, such that $L_i D_i L_i^T$ is a relatively robust representation,

(b) Compute the eigenvalues, λ_j, of $L_i D_i L_i^T$ to high relative accuracy by the dqds algorithm,

(c) If there is a cluster of close eigenvalues, "choose" σ_i close to the cluster, and go to step (a),

(d) Given the approximate eigenvalue λ_j of $L_i D_i L_i^T$, compute the corresponding eigenvector by forming a rank-revealing twisted factorization.

The desired accuracy of the output can be specified by the input parameter ABSTOL.

Note 1: SSTEVR calls SSTEGR when the full spectrum is requested on machines which conform to the ieee-754 floating point standard. SSTEVR calls SSTEBZ and SSTEIN on non-ieee machines and when partial spectrum requests are made.

Normal execution of SSTEGR may create NaNs and infinities and hence may abort due to a floating point exception in environments which do not handle NaNs and infinities in the ieee standard default manner.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

RANGE (input) CHARACTER*1
= 'A': all eigenvalues will be found;
= 'V': all eigenvalues in the half-open interval (VL,VU] will be found;
= 'I': the ILth through IUth eigenvalues will be found.

For RANGE = 'V' or 'I', SSTEBZ and SSTEIN are called.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

D (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the tridiagonal matrix T.
On exit, D may be multiplied by a constant factor chosen to avoid over/underflow in computing the eigenvalues.

E (input/output) REAL array, dimension (N)
On entry, the (n−1) subdiagonal elements of the tridiagonal matrix T in elements 1 to N−1 of E; E(N) need not be set.
On exit, E may be multiplied by a constant factor chosen to avoid over/underflow in computing the eigenvalues.

VL,VR (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be searched for eigenvalues. VL < VU.
Not referenced if RANGE = 'A' or 'I'.

IL,IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned. 1 $\leq$ IL $\leq$ IU $\leq$ N, if N > 0; IL = 1 and IU = 0 if N = 0.
Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute error tolerance for the eigenvalues/eigenvectors.
IF JOBZ = 'V', the eigenvalues and eigenvectors output have residual norms bounded by ABSTOL, and the dot products between different eigenvectors are bounded by ABSTOL. If ABSTOL < N*EPS*$\|T\|_1$, then N*EPS*$\|T\|_1$ will be used in its place, where EPS is the machine precision. The eigenvalues are computed to an accuracy of EPS*$\|T\|_1$ irrespective of ABSTOL. If high relative accuracy is important, set ABSTOL to SLAMCH('S'). Doing so will guarantee that eigenvalues are computed to high relative accuracy when possible in future releases. The current code does not make any guarantees about high relative accuracy, but future releases will.

M (output) INTEGER
The total number of eigenvalues found. 0 $\leq$ M $\leq$ N.
If RANGE = 'A', M = N, and if RANGE = 'I', M = IU−IL+1.

W (output) REAL array, dimension (N)
The first M elements contain the selected eigenvalues in ascending order.

Z (output) REAL array, dimension (LDZ, max(1,M))
If JOBZ = 'V', then if INFO = 0, the first M columns of Z contain the orthonormal eigenvectors of the matrix T corresponding to the selected eigenvalues, with the i^{th} column of Z holding the eigenvector associated with W(i).
If JOBZ = 'N', then Z is not referenced.
Note: the user must ensure that at least max(1,M) columns are supplied in the array Z; if RANGE = 'V', the exact value of M is not known in advance and an upper bound must be used.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

ISUPPZ (output) INTEGER array, dimension (2*max(1,M))
The support of the eigenvectors in Z, i.e., the indices indicating the nonzero elements in Z. The i^{th} eigenvector is nonzero only in elements ISUPPZ(2*i−1) through ISUPPZ(2*i).

WORK (workspace/output) REAL array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal (and minimal) LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ max(1,20*N).

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array IWORK. LIWORK $\geq$ max(1, 10*N).

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: Internal error

SSTEVX

```
      SUBROUTINE SSTEVX( JOBZ, RANGE, N, D, E, VL, VU, IL, IU, ABSTOL,
     $                   M, W, Z, LDZ, WORK, IWORK, IFAIL, INFO )
      CHARACTER          JOBZ, RANGE
      INTEGER            IL, INFO, IU, LDZ, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               D( * ), E( * ), W( * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSTEVX computes selected eigenvalues and, optionally, eigenvectors of a real symmetric tridiagonal matrix A. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

RANGE (input) CHARACTER*1
= 'A': all eigenvalues will be found.
= 'V': all eigenvalues in the half-open interval (VL,VU] will be found.
= 'I': the IL^{th} through IU^{th} eigenvalues will be found.

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

D (input/output) REAL array, dimension (N)
On entry, the n diagonal elements of the tridiagonal matrix A.
On exit, D may be multiplied by a constant factor chosen to avoid over/underflow in computing the eigenvalues.

E (input/output) REAL array, dimension (N)
On entry, the (n−1) subdiagonal elements of the tridiagonal matrix A, stored in elements 1 to n−1 of E; E(n) need not be set.
On exit, E may be multiplied by a constant factor chosen to avoid over/underflow in computing the eigenvalues.

VL, VU (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be searched for eigenvalues. VL < VU.
Not referenced if RANGE = 'A' or 'I'.

IL, IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned.
$1 \leq IL \leq IU \leq N$, if $N > 0$; IL = 1 and IU = 0 if N = 0.
Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute error tolerance for the eigenvalues. An approximate eigenvalue is accepted as converged when it is determined to lie in an interval [a,b] of width less than or equal to ABSTOL + EPS*max(|a|,|b|), where EPS is the machine precision. If ABSTOL is less than or equal to zero, then $EPS*||A||_1$ will be used in its place.
Eigenvalues will be computed most accurately when ABSTOL is set to twice the underflow threshold 2*SLAMCH('S'), not zero. If this routine returns with INFO>0, indicating that some eigenvectors did not converge, try setting ABSTOL to 2*SLAMCH('S').

M (output) INTEGER
The total number of eigenvalues found. $0 \leq M \leq N$.
If RANGE = 'A', M = N, and if RANGE = 'I', M = IU−IL+1.

W (output) REAL array, dimension (N)
The first M elements contain the selected eigenvalues in ascending order.

Z (output) REAL array, dimension (LDZ,max(1,M))
If JOBZ = 'V', then if INFO = 0, the first M columns of Z contain the orthonormal eigenvectors of the matrix A corresponding to the selected eigenvalues, with the i^{th} column of Z holding the eigenvector associated with W(i). If an eigenvector fails to converge (INFO > 0), then that column of Z contains the latest approximation to the eigenvector, and the index of the eigenvector is returned in IFAIL.
If JOBZ = 'N', then Z is not referenced.
Note: the user must ensure that at least max(1,M) columns are supplied in the array Z; if RANGE = 'V', the exact value of M is not known in advance and an upper bound must be used.

LDZ (input) INTEGER
The leading dimension of the array Z. $LDZ \geq 1$, and if JOBZ = 'V', $LDZ \geq max(1,N)$.

WORK (workspace) REAL array, dimension (5*N)

IWORK (workspace) INTEGER array, dimension (5*N)

IFAIL (output) INTEGER array, dimension (N)
If JOBZ = 'V', then if INFO = 0, the first M elements of IFAIL are zero; if INFO > 0, then IFAIL contains the indices of the eigenvectors that failed to converge.
If JOBZ = 'N', then IFAIL is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, then i eigenvectors failed to converge. Their indices are stored in array IFAIL.

SSYCON/CSYCON/CHECON

```
      SUBROUTINE SSYCON( UPLO, N, A, LDA, IPIV, ANORM, RCOND, WORK,
     $                   IWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, N
      REAL               ANORM, RCOND
      INTEGER            IPIV( * ), IWORK( * )
      REAL               A( LDA, * ), WORK( * )
```

```
      SUBROUTINE CSYCON( UPLO, N, A, LDA, IPIV, ANORM, RCOND, WORK,
     $                   INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, N
      REAL               ANORM, RCOND
      INTEGER            IPIV( * )
      COMPLEX            A( LDA, * ), WORK( * )

      SUBROUTINE CHECON( UPLO, N, A, LDA, IPIV, ANORM, RCOND, WORK,
     $                   INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, N
      REAL               ANORM, RCOND
      INTEGER            IPIV( * )
      COMPLEX            A( LDA, * ), WORK( * )
```

Purpose

SSYCON/CSYCON estimates the reciprocal of the condition number (in the 1-norm) of a real/complex symmetric matrix A using the factorization A = U*D*U^T or A = L*D*L^T computed by SSYTRF/CSYTRF.

CHECON estimates the reciprocal of the condition number of a complex Hermitian matrix A using the factorization A = U*D*U^H or A = L*D*L^H computed by CHETRF.

An estimate is obtained for $\|A^{-1}\|$, and the reciprocal of the condition number is computed as RCOND = $1/(\|A\| * \|A^{-1}\|)$.

Arguments

UPLO (input) CHARACTER*1
Specifies whether the details of the factorization are stored as an upper or lower triangular matrix.
= 'U': Upper triangular, form is A = U*D*U^T (SSYCON/CSYCON) or A = U*D*U^H (CHECON);
= 'L': Lower triangular, form is A = L*D*L^T (SSYCON/CSYCON) or A = L*D*L^H (CHECON).

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input) REAL/COMPLEX/COMPLEX array, dimension (LDA,N)
The block diagonal matrix D and the multipliers used to obtain the factor U or L as computed by SSYTRF/CSYTRF/CHETRF.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

IPIV (input) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D as determined by SSYTRF/CSYTRF/CHETRF.

ANORM (input) REAL
The 1-norm of the original matrix A.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as RCOND = $1/(\|A\| * \|A^{-1}\|)$.

WORK (workspace) REAL/COMPLEX/COMPLEX array, dimension (2*N)

IWORK *SSYCON only* (workspace) INTEGER array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the ith argument had an illegal value.

SSYEV/CHEEV

```
      SUBROUTINE SSYEV( JOBZ, UPLO, N, A, LDA, W, WORK, LWORK, INFO )
      CHARACTER         JOBZ, UPLO
      INTEGER           INFO, LDA, LWORK, N
      REAL              A( LDA, * ), W( * ), WORK( * )

      SUBROUTINE CHEEV( JOBZ, UPLO, N, A, LDA, W, WORK, LWORK, RWORK,
     $                  INFO )
      CHARACTER         JOBZ, UPLO
      INTEGER           INFO, LDA, LWORK, N
      REAL              RWORK( * ), W( * )
      COMPLEX           A( LDA, * ), WORK( * )
```

Purpose

SSYEV/CHEEV computes all eigenvalues and, optionally, eigenvectors of a real/complex symmetric/Hermitian matrix A.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A.
On exit, if JOBZ = 'V', then if INFO = 0, A contains the orthonormal eigenvectors of the matrix A.

If JOBZ = 'N', then on exit the lower triangle (if UPLO='L') or the upper triangle (if UPLO='U') of A, including the diagonal, is destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The length of the array WORK.
SSYEV
LWORK $\geq$ max(1,3*N−1). For optimal efficiency, LWORK $\geq$ (NB+2)*N, where NB is the block size for SSYTRD returned by ILAENV.
CHEEV
LWORK $\geq$ max(1,2*N−1). For optimal efficiency, LWORK $\geq$ (NB+1)*N, where NB is the block size for CHETRD returned by ILAENV.

RWORK *CHEEV only* (workspace) REAL array, dimension (max(1,3*N−2))

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value
> 0: the algorithm failed to converge; if INFO = i, i off-diagonal elements of an intermediate tridiagonal form did not converge to zero.

SSYEVD/CHEEVD

```
      SUBROUTINE SSYEVD( JOBZ, UPLO, N, A, LDA, W, WORK, LWORK, IWORK,
     $                   LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, LDA, LIWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), W( * ), WORK( * )

      SUBROUTINE CHEEVD( JOBZ, UPLO, N, A, LDA, W, WORK, LWORK, RWORK,
     $                   LRWORK, IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, LDA, LIWORK, LRWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            A( LDA, * ), WORK( * )
```

Purpose

SSYEVD/CHEEVD computes all eigenvalues and, optionally, eigenvectors of a real/complex symmetric/Hermitian matrix A. If eigenvectors are desired, it uses a divide and conquer algorithm.

The divide and conquer algorithm makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A.
On exit, if JOBZ = 'V', then if INFO = 0, A contains the orthonormal eigenvectors of the matrix A.
If JOBZ = 'N', then on exit the lower triangle (if UPLO='L') or the upper triangle (if UPLO='U') of A, including the diagonal, is destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK.
If N $\leq$ 1, LWORK $\geq$ 1.
SSYEVD
If JOBZ = 'N' and N > 1, LWORK $\geq$ 2*N+1.
If JOBZ = 'V' and N > 1, LWORK $\geq$ (1 + 6*N + 2*N^2).
CHEEVD
If JOBZ = 'N' and N > 1, LWORK $\geq$ N + 1.
If JOBZ = 'V' and N > 1, LWORK $\geq$ 2*N + N^2.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CHEEVD only* (workspace/output) REAL array, dimension (LRWORK)
On exit, if INFO = 0, RWORK(1) returns the optimal LRWORK.

LRWORK *CHEEVD only* (input) INTEGER
The dimension of the array RWORK.
If N ≤ 1, LRWORK ≥ 1.
If JOBZ = 'N' and N > 1, LRWORK ≥ N.
If JOBZ = 'V' and N > 1, LRWORK ≥ (1 + 5∗N + 2∗N^2).

If LRWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the RWORK array, returns this value as the first entry of the RWORK array, and no error message related to LRWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array IWORK.
If N ≤ 1, LIWORK ≥ 1.
If JOBZ = 'N' and N > 1, LIWORK ≥ 1.
If JOBZ = 'V' and N > 1, LIWORK ≥ 3 + 5∗N.

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the ith argument had an illegal value.
> 0: if INFO = i, the algorithm failed to converge; i off-diagonal elements of an intermediate tridiagonal form did not converge to zero.

SSYEVR/CHEEVR

```
      SUBROUTINE SSYEVR( JOBZ, RANGE, UPLO, N, A, LDA, VL, VU, IL, IU,
     $                   ABSTOL, M, W, Z, LDZ, ISUPPZ, WORK, LWORK,
     $                   IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, IU, LDA, LDZ, LIWORK, LWORK,
     $                   M, N
      REAL               ABSTOL, VL, VU
      INTEGER            ISUPPZ( * ), IWORK( * )
      REAL               A( LDA, * ), W( * ), WORK( * ), Z( LDZ, * )

      SUBROUTINE CHEEVR( JOBZ, RANGE, UPLO, N, A, LDA, VL, VU, IL, IU,
     $                   ABSTOL, M, W, Z, LDZ, ISUPPZ, WORK, LWORK,
     $                   RWORK, LRWORK, IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, IU, LDA, LDZ, LIWORK, LWORK,
     $                   M, N, LRWORK
      REAL               ABSTOL, VL, VU
      INTEGER            ISUPPZ( * ), IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            A( LDA, * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSYEVR/CHEEVR computes selected eigenvalues and, optionally, eigenvectors of a real/complex symmetric/Hermitian tridiagonal matrix T. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

Whenever possible, SSYEVR/CHEEVR calls SSTEGR/CSTEGR to compute the eigenspectrum using Relatively Robust Representations. SSTEGR/CSTEGR computes eigenvalues by the dqds algorithm, while orthogonal eigenvectors are computed from various "good" LDL^T representations (also known as Relatively Robust Representations). Gram-Schmidt orthogonalization is avoided as far as possible. More specifically, the various steps of the algorithm are as follows. For the ith unreduced block of T,

(a) Compute $T - \sigma_i = L_i D_i L_i^T$, such that $L_i D_i L_i^T$ is a relatively robust representation,

(b) Compute the eigenvalues, λ_j, of $L_i D_i L_i^T$ to high relative accuracy by the dqds algorithm,

(c) If there is a cluster of close eigenvalues, "choose" σ_i close to the cluster, and go to step (a),

(d) Given the approximate eigenvalue λ_j of $L_i D_i L_i^T$, compute the corresponding eigenvector by forming a rank-revealing twisted factorization.

The desired accuracy of the output can be specified by the input parameter ABSTOL.

Note 1: SSYEVR/CHEEVR calls SSTEGR/CSTEGR when the full spectrum is requested on machines which conform to the ieee-754 floating point standard. SSYEVR/CHEEVR calls SSTEBZ and SSTEIN/CSTEIN on non-ieee machines and when partial spectrum requests are made.

Normal execution of SSTEGR/CSTEGR may create NaNs and infinities and hence may abort due to a floating point exception in environments which do not handle NaNs and infinities in the ieee standard default manner.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

RANGE (input) CHARACTER*1
= 'A': all eigenvalues will be found;
= 'V': all eigenvalues in the half-open interval (VL,VU] will be found;
= 'I': the IL^{th} through IU^{th} eigenvalues will be found.
For RANGE = 'V' or 'I', SSTEBZ and SSTEIN are called.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A.
On exit, the lower triangle (if UPLO='L') or the upper triangle (if UPLO='U') of A, including the diagonal, is destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

VL,VR (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be searched for eigenvalues. VL < VU.
Not referenced if RANGE = 'A' or 'I'.

IL,IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned. 1 $\leq$ IL $\leq$ IU $\leq$ N, if N > 0; IL = 1 and IU = 0 if N = 0.
Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute error tolerance for the eigenvalues/eigenvectors. IF JOBZ = 'V', the eigenvalues and eigenvectors output have residual norms bounded by ABSTOL, and the dot products between different eigenvectors are bounded by ABSTOL. If ABSTOL < N*EPS*$\|T\|_1$, then N*EPS*$\|T\|_1$ will be used in its place, where EPS is the machine precision. The eigenvalues are computed to an accuracy of EPS*$\|T\|_1$ irrespective of ABSTOL. If high relative accuracy is important, set ABSTOL to SLAMCH('S'). Doing so will guarantee that eigenvalues are computed to high relative accuracy when possible in future releases. The current code does not make any guarantees about high relative accuracy, but future releases will.

M (output) INTEGER
The total number of eigenvalues found. 0 $\leq$ M $\leq$ N.
If RANGE = 'A', M = N, and if RANGE = 'I', M = IU−IL+1.

W (output) REAL array, dimension (N)
The first M elements contain the selected eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, max(1,M))
If JOBZ = 'V', then if INFO = 0, the first M columns of Z contain the orthonormal eigenvectors of the matrix T corresponding to the selected eigenvalues, with the i^{th} column of Z holding the eigenvector associated with W(i).
If JOBZ = 'N', then Z is not referenced.
Note: the user must ensure that at least max(1,M) columns are supplied in the array Z; if RANGE = 'V', the exact value of M is not known in advance and an upper bound must be used.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

ISUPPZ (output) INTEGER array, dimension (2*max(1,M))
The support of the eigenvectors in Z, i.e., the indices indicating the nonzero elements in Z. The i^{th} eigenvector is nonzero only in elements ISUPPZ(2*i−1) through ISUPPZ(2*i).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal (and minimal) LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. *SSYEVR*
LWORK $\geq$ max(1,26*N).
For optimal efficiency, LWORK $\geq$ (NB+6)*N, where NB is the max of the blocksize for SSYTRD and SORMTR returned by ILAENV.
CHEEVR
LWORK $\geq$ max(1,2*N).
For optimal efficiency, LWORK $\geq$ (NB+1)*N, where NB is the max of the blocksize for CHETRD and for CUNMTR as returned by ILAENV.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CHEEVR only* (workspace/output) REAL array, dimension (LRWORK)
On exit, if INFO = 0, RWORK(1) returns the optimal (and minimal) LRWORK.

LRWORK *CHEEVR only* (input) INTEGER
The length of the array RWORK. LRWORK $\geq$ max(1,24*N).

If LRWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the RWORK array, returns this value as the first entry of the RWORK array, and no error message related to LRWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array IWORK. LIWORK $\geq$ max(1, 10*N).

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: Internal error

SSYEVX/CHEEVX

```
      SUBROUTINE SSYEVX( JOBZ, RANGE, UPLO, N, A, LDA, VL, VU, IL, IU,
     $                   ABSTOL, M, W, Z, LDZ, WORK, LWORK, IWORK,
     $                   IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, IU, LDA, LDZ, LWORK, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               A( LDA, * ), W( * ), WORK( * ), Z( LDZ, * )
```

```
      SUBROUTINE CHEEVX( JOBZ, RANGE, UPLO, N, A, LDA, VL, VU, IL, IU,
     $                   ABSTOL, M, W, Z, LDZ, WORK, LWORK, RWORK,
     $                   IWORK, IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, IU, LDA, LDZ, LWORK, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            A( LDA, * ), WORK( * ), Z( LDZ, * )
```

Purpose

SSYEVX/CHEEVX computes selected eigenvalues and, optionally, eigenvectors of a real/complex symmetric/Hermitian matrix A. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

Arguments

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

RANGE (input) CHARACTER*1
= 'A': all eigenvalues will be found.
= 'V': all eigenvalues in the half-open interval (VL,VU] will be found.
= 'I': the IL^{th} through IU^{th} eigenvalues will be found.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A.
On exit, the lower triangle (if UPLO='L') or the upper triangle (if UPLO='U') of A, including the diagonal, is destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

VL, VU (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be searched for eigenvalues. VL < VU.
Not referenced if RANGE = 'A' or 'I'.

IL, IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned.

$1 \leq IL \leq IU \leq N$, if $N > 0$; IL = 1 and IU = 0 if N = 0.
Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute error tolerance for the eigenvalues. An approximate eigenvalue is accepted as converged when it is determined to lie in an interval [a,b] of width less than or equal to ABSTOL + EPS*max(|a|,|b|), where EPS is the machine precision. If ABSTOL is less than or equal to zero, then $EPS*\|T\|_1$ will be used in its place, where T is the tridiagonal matrix obtained by reducing A to tridiagonal form.
Eigenvalues will be computed most accurately when ABSTOL is set to twice the underflow threshold 2*SLAMCH('S'), not zero. If this routine returns with INFO>0, indicating that some eigenvectors did not converge, try setting ABSTOL to 2*SLAMCH('S').

M (output) INTEGER
The total number of eigenvalues found. $0 \leq M \leq N$.
If RANGE = 'A', M = N, and if RANGE = 'I', M = IU−IL+1.

W (output) REAL array, dimension (N)
The first m elements contain the selected eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, max(1,M))
If JOBZ = 'V', then if INFO = 0, the first M columns of Z contain the orthonormal eigenvectors of the matrix A corresponding to the selected eigenvalues, with the i^{th} column of Z holding the eigenvector associated with W(i). If an eigenvector fails to converge, then that column of Z contains the latest approximation to the eigenvector, and the index of the eigenvector is returned in IFAIL.
If JOBZ = 'N', then Z is not referenced.
Note: the user must ensure that at least max(1,M) columns are supplied in the array Z; if RANGE = 'V', the exact value of M is not known in advance and an upper bound must be used.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The length of the array WORK.
SSYEVX
LWORK $\geq$ max(1,8*N). For optimal efficiency, LWORK $\geq$ (NB+3)*N, where NB is the max of the block size for SSYTRD and SORMTR returned by ILAENV.
CHEEVX
LWORK $\geq$ max(1,2*N−1). For optimal efficiency, LWORK $\geq$ (NB+1)*N, where NB is the max of the block size for CHETRD and for CUNMTR as returned by ILAENV.

RWORK *CHEEVX only* (workspace) REAL array, dimension (7*N)

IWORK (workspace) INTEGER array, dimension (5*N)

IFAIL (output) INTEGER array, dimension (N)
If JOBZ = 'V', then if INFO = 0, the first M elements of IFAIL are zero; if INFO > 0, then IFAIL contains the indices of the eigenvectors that failed to converge.
If JOBZ = 'N', then IFAIL is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, then i eigenvectors failed to converge. Their indices are stored in array IFAIL.

SSYGST/CHEGST

```
      SUBROUTINE SSYGST( ITYPE, UPLO, N, A, LDA, B, LDB, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, ITYPE, LDA, LDB, N
      REAL               A( LDA, * ), B( LDB, * )

      SUBROUTINE CHEGST( ITYPE, UPLO, N, A, LDA, B, LDB, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, ITYPE, LDA, LDB, N
      COMPLEX            A( LDA, * ), B( LDB, * )
```

Purpose

SSYGST/CHEGST reduces a real/complex symmetric/Hermitian definite generalized eigenproblem to standard form.

If ITYPE = 1, the problem is $A*x = \lambda*B*x$, and A is overwritten by $(U^H)^{-1}*A*U^{-1}$ or $L^{-1}*A*(L^H)^{-1}$.

If ITYPE = 2 or 3, the problem is $A*B*x = \lambda*x$ or $B*A*x = \lambda*x$, and A is overwritten by $U*A*U^H$ or L^H*A*L.

B must have been previously factorized as U^H*U or $L*L^H$ by SPOTRF/CPOTRF.

Arguments

ITYPE (input) INTEGER
= 1: compute $(U^H)^{-1}*A*U^{-1}$ or $L^{-1}*A*(L^H)^{-1}$;
= 2 or 3: compute $U*A*U^H$ or L^H*A*L.

UPLO (input) CHARACTER
= 'U': Upper triangle of A is stored and B is factored as U^H*U;
= 'L': Lower triangle of A is stored and B is factored as $L*L^H$.

N (input) INTEGER
The order of the matrices A and B. $N \geq 0$.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A, and the strictly upper triangular part of A is not referenced.
On exit, if INFO = 0, the transformed matrix, stored in the same format as A.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

B (input) REAL/COMPLEX array, dimension (LDB,N)
The triangular factor from the Cholesky factorization of B, as returned by SPOTRF/CPOTRF.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SSYGV/CHEGV

```
      SUBROUTINE SSYGV( ITYPE, JOBZ, UPLO, N, A, LDA, B, LDB, W, WORK,
     $                  LWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, ITYPE, LDA, LDB, LWORK, N
      REAL               A( LDA, * ), B( LDB, * ), W( * ), WORK( * )

      SUBROUTINE CHEGV( ITYPE, JOBZ, UPLO, N, A, LDA, B, LDB, W, WORK,
     $                  LWORK, RWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, ITYPE, LDA, LDB, LWORK, N
      REAL               RWORK( * ), W( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), WORK( * )
```

Purpose

SSYGV/CHEGV computes all the eigenvalues, and optionally, the eigenvectors of a real/complex generalized symmetric/Hermitian definite eigenproblem, of the form

$A*x = \lambda*B*x$, $A*Bx = \lambda*x$, or $B*A*x = \lambda*x$.

Here A and B are assumed to be symmetric/Hermitian and B is also positive definite.

Arguments

ITYPE (input) INTEGER
Specifies the problem type to be solved:
= 1: $A*x = \lambda*B*x$
= 2: $A*B*x = \lambda*x$
= 3: $B*A*x = \lambda*x$

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangles of A and B are stored;
= 'L': Lower triangles of A and B are stored.

N (input) INTEGER
The order of the matrices A and B. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A.
On exit, if JOBZ = 'V', then if INFO = 0, A contains the matrix Z of eigenvectors. The eigenvectors are normalized as follows:
if ITYPE = 1 or 2, $Z^H*B*Z = I$;
if ITYPE = 3, $Z^H*B^{-1}*Z = I$.
If JOBZ = 'N', then on exit the upper triangle (if UPLO='U') or the lower triangle (if UPLO='L') of A, including the diagonal, is destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB, N)
On entry, the symmetric/Hermitian positive definite matrix B. If UPLO = 'U', the leading n-by-n upper triangular part of B contains the upper triangular part of the matrix B. If UPLO = 'L', the leading n-by-n lower triangular part of B contains the lower triangular part of the matrix B.
On exit, if INFO $\leq$ N, the part of B containing the matrix is overwritten by the triangular factor U or L from the Cholesky factorization $B = U^H*U$ or $B = L*L^H$.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The length of the array WORK.
SSYGV

LWORK ≥ max(1,3*N−1). For optimal efficiency, LWORK ≥ (NB+2)*N, where NB is the block size for SSYTRD returned by ILAENV.
CHEGV
LWORK ≥ max(1,2*N−1). For optimal efficiency, LWORK ≥ (NB+1)*N, where NB is the block size for CHETRD returned by ILAENV.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CHEGV only* (workspace) REAL array, dimension (max(1,3*N−2))

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value
> 0: SPOTRF/CPOTRF or SSYEV/CHEEV returned an error code:
≤ N: SSYEV/CHEEV failed to converge; if INFO = i, i off-diagonal elements of an intermediate tridiagonal form did not converge to zero.
> N: if INFO = N + i, for 1 ≤ i ≤ N, then the leading minor of order i of B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

SSYGVD/CHEGVD

```
      SUBROUTINE SSYGVD( ITYPE, JOBZ, UPLO, N, A, LDA, B, LDB, W, WORK,
     $                   LWORK, IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, ITYPE, LDA, LDB, LIWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), B( LDB, * ), W( * ), WORK( * )

      SUBROUTINE CHEGVD( ITYPE, JOBZ, UPLO, N, A, LDA, B, LDB, W, WORK,
     $                   LWORK, RWORK, LRWORK, IWORK, LIWORK, INFO )
      CHARACTER          JOBZ, UPLO
      INTEGER            INFO, ITYPE, LDA, LDB, LIWORK, LRWORK, LWORK, N
      INTEGER            IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), WORK( * )
```

Purpose

SSYGVD/CHEGVD computes all the eigenvalues, and optionally, the eigenvectors of a real/complex generalized symmetric-definite/Hermitian-definite eigenproblem, of the form $A*x = \lambda*B*x$, $A*Bx = \lambda*x$, or $B*A*x = \lambda*x$. Here A and B are assumed to be symmetric/Hermitian and B is also positive definite. If eigenvectors are desired, it uses a divide and conquer algorithm.

The divide and conquer algorithm makes very mild assumptions about floating point arithmetic. It will work on machines with a guard digit in add/subtract, or on those binary machines without guard digits which subtract like the Cray X-MP, Cray Y-MP, Cray C-90, or Cray-2. It could conceivably fail on hexadecimal or decimal machines without guard digits, but we know of none.

Arguments

ITYPE (input) INTEGER
Specifies the problem type to be solved:
= 1: A*x = λ*B*x
= 2: A*B*x = λ*x
= 3: B*A*x = λ*x

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

UPLO (input) CHARACTER*1
= 'U': Upper triangles of A and B are stored;
= 'L': Lower triangles of A and B are stored.

N (input) INTEGER
The order of the matrices A and B. N ≥ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading N-by-N upper triangular part of A contains the upper triangular part of the matrix A. If UPLO = 'L', the leading N-by-N lower triangular part of A contains the lower triangular part of the matrix A.

On exit, if JOBZ = 'V', then if INFO = 0, A contains the matrix Z of eigenvectors. The eigenvectors are normalized as follows:
if ITYPE = 1 or 2, $Z^H * B * Z = I$;
if ITYPE = 3, $Z^H * B^{-1} * Z = I$.
If JOBZ = 'N', then on exit the upper triangle (if UPLO='U') or the lower triangle (if UPLO='L') of A, including the diagonal, is destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB, N)
On entry, the symmetric/Hermitian matrix B. If UPLO = 'U', the leading N-by-N upper triangular part of B contains the upper triangular part of the matrix B. If UPLO = 'L', the leading N-by-N lower triangular part of B contains the lower triangular part of the matrix B.

On exit, if INFO ≤ N, the part of B containing the matrix is overwritten by the triangular factor U or L from the Cholesky factorization

$B = U^H * U$ or $B = L * L^H$.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

W (output) REAL array, dimension (N)
If INFO = 0, the eigenvalues in ascending order.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The length of the array WORK. If N $\leq$ 1, LWORK $\geq$ 1.
SSYGVD
If JOBZ = 'N' and N > 1, LWORK $\geq$ 2*N+1.
If JOBZ = 'V' and N > 1, LWORK $\geq$ 1 + 6*N + 2*N^2.
CHEGVD
If JOBZ = 'N' and N > 1, LWORK $\geq$ N + 1. If JOBZ = 'V' and N > 1, LWORK $\geq$ 2*N + N^2.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CHEGVD only* (workspace/output) REAL array, dimension (LRWORK)
On exit, if INFO = 0, RWORK(1) returns the optimal LRWORK.

LRWORK *CHEGVD only* (input) INTEGER
The dimension of the array RWORK.
If N $\leq$ 1, LRWORK $\geq$ 1.
If JOBZ = 'N' and N > 1, LRWORK $\geq$ N.
If JOBZ = 'V' and N > 1, LRWORK $\geq$ 1 + 5*N + 2*N^2.

If LRWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the RWORK array, returns this value as the first entry of the RWORK array, and no error message related to LRWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
On exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
The dimension of the array IWORK.
If N $\leq$ 1, LIWORK $\geq$ 1.
If JOBZ = 'N' and N > 1, LIWORK $\geq$ 1.
If JOBZ = 'V' and N > 1, LIWORK $\geq$ 3 + 5*N.

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the ith argument had an illegal value.
> 0: SPOTRF/CPOTRF or SSYEVD/CHEEVD returned an error code:
$\leq$ N: if INFO = i, SSYEVD/CHEEVD failed to converge; i off-diagonal elements of an intermediate tridiagonal form did not converge to zero;
> N: if INFO = N + i, for 1 $\leq$ i $\leq$ N, then the leading minor of order i of B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

SSYGVX/CHEGVX

```
      SUBROUTINE SSYGVX( ITYPE, JOBZ, RANGE, UPLO, N, A, LDA, B, LDB,
     $                   VL, VU, IL, IU, ABSTOL, M, W, Z, LDZ, WORK,
     $                   LWORK, IWORK, IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, ITYPE, IU, LDA, LDB, LDZ, LWORK, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               A( LDA, * ), B( LDB, * ), W( * ), WORK( * ),
     $                   Z( LDZ, * )

      SUBROUTINE CHEGVX( ITYPE, JOBZ, RANGE, UPLO, N, A, LDA, B, LDB,
     $                   VL, VU, IL, IU, ABSTOL, M, W, Z, LDZ, WORK,
     $                   LWORK, RWORK, IWORK, IFAIL, INFO )
      CHARACTER          JOBZ, RANGE, UPLO
      INTEGER            IL, INFO, ITYPE, IU, LDA, LDB, LDZ, LWORK, M, N
      REAL               ABSTOL, VL, VU
      INTEGER            IFAIL( * ), IWORK( * )
      REAL               RWORK( * ), W( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), WORK( * ),
     $                   Z( LDZ, * )
```

Purpose

SSYGVX/CHEGVX computes selected eigenvalues, and optionally, eigenvectors of a real/complex generalized symmetric-definite/Hermitian-definite eigenproblem, of the form $A * x = \lambda * B * x$, $A * Bx = \lambda * x$, or $B * A * x = \lambda * x$. Here A and B are assumed to be symmetric/Hermitian and B is also positive definite. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

Arguments

ITYPE (input) INTEGER
Specifies the problem type to be solved:
= 1: A*x = λ*B*x
= 2: A*B*x = λ*x
= 3: B*A*x = λ*x

JOBZ (input) CHARACTER*1
= 'N': Compute eigenvalues only;
= 'V': Compute eigenvalues and eigenvectors.

RANGE (input) CHARACTER*1
= 'A': all eigenvalues will be found.
= 'V': all eigenvalues in the half-open interval (VL,VU] will be found.
= 'I': the ILth through IUth eigenvalues will be found.

UPLO (input) CHARACTER*1
= 'U': Upper triangles of A and B are stored;
= 'L': Lower triangles of A and B are stored.

N (input) INTEGER
The order of the matrices A and B. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA, N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading N-by-N upper triangular part of A contains the upper triangular part of the matrix A. If UPLO = 'L', the leading N-by-N lower triangular part of A contains the lower triangular part of the matrix A.

On exit, the lower triangle (if UPLO='L') or the upper triangle (if UPLO='U') of A, including the diagonal, is destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB, N)
On entry, the symmetric/Hermitian matrix B. If UPLO = 'U', the leading N-by-N upper triangular part of B contains the upper triangular part of the matrix B. If UPLO = 'L', the leading N-by-N lower triangular part of B contains the lower triangular part of the matrix B.

On exit, if INFO $\leq$ N, the part of B containing the matrix is overwritten by the triangular factor U or L from the Cholesky factorization $B = U^H * U$ or $B = L * L^H$.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

VL,VU (input) REAL
If RANGE='V', the lower and upper bounds of the interval to be searched for eigenvalues. VL < VU.
Not referenced if RANGE = 'A' or 'I'.

IL,IU (input) INTEGER
If RANGE='I', the indices (in ascending order) of the smallest and largest eigenvalues to be returned. 1 $\leq$ IL $\leq$ IU $\leq$ N, if N > 0; IL = 1 and IU = 0 if N = 0. Not referenced if RANGE = 'A' or 'V'.

ABSTOL (input) REAL
The absolute error tolerance for the eigenvalues. An approximate eigenvalue is accepted as converged when it is determined to lie in an interval [a,b] of width less than or equal to ABSTOL + EPS*max(|a|,|b|), where EPS is the machine precision. If ABSTOL is less than or equal to zero, then EPS*$\|T\|_1$ will be used in its place, where T is the tridiagonal matrix obtained by reducing A to tridiagonal form.
Eigenvalues will be computed most accurately when ABSTOL is set to twice the underflow threshold 2*SLAMCH('S'), not zero. If this routine returns with INFO>0, indicating that some eigenvectors did not converge, try setting ABSTOL to 2*SLAMCH('S').

M (output) INTEGER
The total number of eigenvalues found. 0 $\leq$ M $\leq$ N.
If RANGE = 'A', M = N, and if RANGE = 'I', M = IU−IL+1.

W (output) REAL array, dimension (N)
The first m elements contain the selected eigenvalues in ascending order.

Z (output) REAL/COMPLEX array, dimension (LDZ, max(1,M))
If JOBZ = 'N', then Z is not referenced.
If JOBZ = 'V', then if INFO = 0, the first M columns of Z contain the orthonormal eigenvectors of the matrix A corresponding to the selected eigenvalues, with the ith column of Z holding the eigenvector associated with W(i). The eigenvectors are normalized as follows:
if ITYPE = 1 or 2, $Z^H * B * Z = I$;
if ITYPE = 3, $Z^H * B^{-1} * Z = I$.

If an eigenvector fails to converge, then that column of Z contains the latest approximation to the eigenvector, and the index of the eigenvector is returned in IFAIL. Note: the user must ensure that at least max(1,M) columns are supplied in the array Z; if RANGE = 'V', the exact value of M is not known in advance and an upper bound must be used.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1, and if JOBZ = 'V', LDZ $\geq$ max(1,N).

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The length of the array WORK.
SSYGVX
LWORK $\geq$ max(1,8*N). For optimal efficiency, LWORK $\geq$ (NB+3)*N, where NB is the blocksize for SSYTRD returned by ILAENV.
CHEGVX
LWORK $\geq$ max(1,2*N−1).

For optimal efficiency, LWORK ¿= (NB+1)*N, where NB is the blocksize for CHETRD returned by ILAENV.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

RWORK *CHEGVX only* (workspace) REAL array, dimension (7*N)

IWORK (workspace) INTEGER array, dimension (5*N)

IFAIL (output) INTEGER array, dimension (N)
If JOBZ = 'V', then if INFO = 0, the first M elements of IFAIL are zero. If INFO > 0, then IFAIL contains the indices of the eigenvectors that failed to converge. If JOBZ = 'N', then IFAIL is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = -i, the i-th argument had an illegal value.
> 0: SPOTRF/CPOTRF or SSYEVX/CHEEVX returned an error code:
$\le$ N: if INFO = i, SSYEVX/CHEEVX failed to converge; i eigenvectors failed to converge. Their indices are stored in array IFAIL.
> N: if INFO = N + i, for $1 \le i \le n$, then the leading minor of order i of B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

SSYRFS/CSYRFS/CHERFS

```
      SUBROUTINE SSYRFS( UPLO, N, NRHS, A, LDA, AF, LDAF, IPIV, B, LDB,
     $                   X, LDX, FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      INTEGER            IPIV( * ), IWORK( * )
      REAL               A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   BERR( * ), FERR( * ), WORK( * ), X( LDX, * )

      SUBROUTINE CSYRFS( UPLO, N, NRHS, A, LDA, AF, LDAF, IPIV, B, LDB,
     $                   X, LDX, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )

      SUBROUTINE CHERFS( UPLO, N, NRHS, A, LDA, AF, LDAF, IPIV, B, LDB,
     $                   X, LDX, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDAF, LDB, LDX, N, NRHS
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SSYRFS/CSYRFS/CHERFS improves the computed solution to a system of linear equations when the coefficient matrix is real/complex/complex symmetric/-symmetric/Hermitian indefinite, and provides error bounds and backward error estimates for the solution.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\ge$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\ge$ 0.

A (input) REAL/COMPLEX/COMPLEX array, dimension (LDA,N)
The symmetric/symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A, and the strictly upper triangular part of A is not referenced.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\ge$ max(1,N).

AF (input) REAL/COMPLEX/COMPLEX array, dimension (LDAF,N)
The factored form of the matrix A. AF contains the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $A = U*D*U^T$ or $A = L*D*L^T$ as computed by SSYTRF/CSYTRF or the factorization $A = U*D*U^H$ or $A = L*D*L^H$ as computed by CHETRF.

LDAF (input) INTEGER
The leading dimension of the array AF. LDAF $\ge$ max(1,N).

IPIV (input) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D as determined by SSYTRF/CSYTRF/CHETRF.

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (input/output)
REAL/COMPLEX/COMPLEX array, dimension (LDX,NRHS)
On entry, the solution matrix X, as computed by SSYTRS/CSYTRS/CHETRS.
On exit, the improved solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *SSYRFS* (workspace) REAL array, dimension (3*N)
CSYRFS/CHERFS (workspace) COMPLEX array, dimension (2*N)

IWORK *SSYRFS only* (workspace) INTEGER array, dimension (N)

RWORK *CSYRFS/CHERFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SSYSV/CSYSV/CHESV

```
      SUBROUTINE SSYSV( UPLO, N, NRHS, A, LDA, IPIV, B, LDB, WORK,
     $                  LWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDB, LWORK, N, NRHS
      INTEGER            IPIV( * )
      REAL               A( LDA, * ), B( LDB, * ), WORK( LWORK )

      SUBROUTINE CSYSV( UPLO, N, NRHS, A, LDA, IPIV, B, LDB, WORK,
     $                  LWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDB, LWORK, N, NRHS
      INTEGER            IPIV( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), WORK( LWORK )

      SUBROUTINE CHESV( UPLO, N, NRHS, A, LDA, IPIV, B, LDB, WORK,
     $                  LWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDB, LWORK, N, NRHS
      INTEGER            IPIV( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), WORK( LWORK )
```

Purpose

SSYSV/CSYSV/CHESV computes the solution to a real/complex/complex system of linear equations A*X = B, where A is an n-by-n symmetric/symmetric/Hermitian matrix and X and B are n-by-nrhs matrices.

The diagonal pivoting method is used to factor A as

$A = U*D*U^T$ or $A = L*D*L^T$ (SSYSV/CSYSV) or

$A = U*D*U^H$ or $A = L*D*L^H$ (CHESV),

where U (or L) is a product of permutation and unit upper (lower) triangular matrices, and D is symmetric (SSYSV/CSYSV) or Hermitian (CHESV) and block diagonal with 1-by-1 and 2-by-2 diagonal blocks. The factored form of A is then used to solve the system of equations A*X = B.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

A (input/output) REAL/COMPLEX/COMPLEX array, dimension (LDA,N)
On entry, the symmetric/symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A, and the strictly upper triangular part of A is not referenced.
On exit, if INFO = 0, the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $A = U*D*U^T$ or $A = L*D*L^T$ as computed by SSYTRF/CSYTRF or the factorization $A = U*D*U^H$ or $A = L*D*L^H$ as computed by CHETRF.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

IPIV (output) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D, as deter-

mined by SSYTRF/CSYTRF/CHETRF. If IPIV(k) > 0, then rows and columns k and IPIV(k) were interchanged, and D(k,k) is a 1-by-1 diagonal block. If UPLO = 'U' and IPIV(k) = IPIV(k−1) < 0, then rows and columns k−1 and −IPIV(k) were interchanged and D(k−1:k,k−1:k) is a 2-by-2 diagonal block. If UPLO = 'L' and IPIV(k) = IPIV(k+1) < 0, then rows and columns k+1 and −IPIV(k) were interchanged and D(k:k+1,k:k+1) is a 2-by-2 diagonal block.

B (input/output)
REAL/COMPLEX/COMPLEX array, dimension (LDB,NRHS)
On entry, the n-by-nrhs right hand side matrix B.
On exit, if INFO = 0, the n-by-nrhs solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,N).

WORK (workspace/output)
REAL/COMPLEX/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The length of WORK. LWORK ≥ 1, and for best performance LWORK ≥ N*NB, where NB is the optimal blocksize for SSYTRF/CSYTRF/CHETRF.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, D(i,i) is exactly zero. The factorization has been completed, but the block diagonal matrix D is exactly singular, so the solution could not be computed.

SSYSVX/CSYSVX/CHESVX

```
      SUBROUTINE SSYSVX( FACT, UPLO, N, NRHS, A, LDA, AF, LDAF, IPIV, B,
     $                   LDB, X, LDX, RCOND, FERR, BERR, WORK, LWORK,
     $                   IWORK, INFO )
      CHARACTER          FACT, UPLO
      INTEGER            INFO, LDA, LDAF, LDB, LDX, LWORK, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * ), IWORK( * )
      REAL               A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   BERR( * ), FERR( * ), WORK( * ), X( LDX, * )
```

```
      SUBROUTINE CSYSVX( FACT, UPLO, N, NRHS, A, LDA, AF, LDAF, IPIV, B,
     $                   LDB, X, LDX, RCOND, FERR, BERR, WORK, LWORK,
     $                   RWORK, INFO )
      CHARACTER          FACT, UPLO
      INTEGER            INFO, LDA, LDAF, LDB, LDX, LWORK, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

```
      SUBROUTINE CHESVX( FACT, UPLO, N, NRHS, A, LDA, AF, LDAF, IPIV, B,
     $                   LDB, X, LDX, RCOND, FERR, BERR, WORK, LWORK,
     $                   RWORK, INFO )
      CHARACTER          FACT, UPLO
      INTEGER            INFO, LDA, LDAF, LDB, LDX, LWORK, N, NRHS
      REAL               RCOND
      INTEGER            IPIV( * )
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            A( LDA, * ), AF( LDAF, * ), B( LDB, * ),
     $                   WORK( * ), X( LDX, * )
```

Purpose

SSYSVX/CSYSVX/CHESVX uses the diagonal pivoting factorization to compute the solution to a real/complex/complex system of linear equations A*X = B, where A is an n-by-n symmetric (SSYSVX/CSYSVX) or Hermitian (CHESVX) matrix and X and B are n-by-nrhs matrices.

Error bounds on the solution and a condition estimate are also computed.

Description

The following steps are performed:

1. If FACT = 'N', the diagonal pivoting method is used to factor A. The form of the factorization is

   $A = U*D*U^T$ or $A = L*D*L^T$ (SSYSVX/CSYSVX) or

   $A = U*D*U^H$ or $A = L*D*L^H$ (CHESVX),

   where U (or L) is a product of permutation and unit upper (lower) triangular matrices, and D is symmetric (SSYSVX/CSYSVX) or Hermitian (CHESVX) and block diagonal with 1-by-1 and 2-by-2 diagonal blocks.

2. If some D(i,i)=0, so that D is exactly singular, then the routine returns with INFO = i. Otherwise, the factored form of A is used to estimate the condition number of the matrix A. If the reciprocal of the condition number is less than machine precision, INFO = N+1 is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.

3. The system of equations is solved for X using the factored form of A.

4. Iterative refinement is applied to improve the computed solution matrix and calculate error bounds and backward error estimates for it.

Arguments

FACT (input) CHARACTER*1
Specifies whether or not the factored form of A has been supplied on entry.
= 'F': On entry, AF and IPIV contain the factored form of A. A, AF and IPIV will not be modified.
= 'N': The matrix A will be copied to AF and factored.

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The number of linear equations, i.e., the order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

A (input) REAL/COMPLEX/COMPLEX array, dimension (LDA,N)
The symmetric/symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A, and the strictly upper triangular part of A is not referenced.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

AF (input or output)
REAL/COMPLEX/COMPLEX array, dimension (LDAF,N)
If FACT = 'F', then AF is an input argument and on entry contains the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization A = $U*D*U^T$ or A = $L*D*L^T$ as computed by SSYTRF/CSYTRF or the factorization A = $U*D*U^H$ or A = $L*D*L^H$ as computed by CHETRF.
If FACT = 'N', then AF is an output argument and on exit returns the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization.

LDAF (input) INTEGER
The leading dimension of the array AF. LDAF $\geq$ max(1,N).

IPIV (input or output) INTEGER array, dimension (N)
If FACT = 'F', then IPIV is an input argument and on entry contains details of the interchanges and the block structure of D, as determined by SSYTRF/CSYTRF/CHETRF.
If IPIV(k) > 0, then rows and columns k and IPIV(k) were interchanged and D(k,k) is a 1-by-1 diagonal block.
If UPLO = 'U' and IPIV(k) = IPIV(k−1) < 0, then rows and columns k−1 and −IPIV(k) were interchanged and D(k−1:k,k−1:k) is a 2-by-2 diagonal block. If UPLO = 'L' and IPIV(k) = IPIV(k+1) < 0, then rows and columns k+1 and −IPIV(k) were interchanged and D(k:k+1,k:k+1) is a 2-by-2 diagonal block.
If FACT = 'N', then IPIV is an output argument and on exit contains details of the interchanges and the block structure of D, as determined by SSYTRF/CSYTRF/CHETRF.

B (input) REAL/COMPLEX/COMPLEX array, dimension (LDB,NRHS)
The n-by-nrhs right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (output)
REAL/COMPLEX/COMPLEX array, dimension (LDX,NRHS)
If INFO = 0 or INFO = N+1, the N-by-NRHS solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

RCOND (output) REAL
The estimate of the reciprocal condition number of the matrix A. If RCOND is less than the machine precision (in particular, if RCOND = 0), the matrix is singular to working precision. This condition is indicated by a return code of INFO > 0.

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK (workspace/output) REAL/COMPLEX/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The length of the array WORK.
SSYSVX
LWORK $\geq$ 3*N, and for best performance LWORK $\geq$ N*NB, where NB is the optimal block size for SSYTRF.
CSYSVX/CHESVX
LWORK $\geq$ 2*N, and for best performance LWORK $\geq$ N*NB, where

NB is the optimal block size for CSYTRF/CHETRF.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

IWORK *SSYSVX only* (workspace) INTEGER array, dimension (N)

RWORK *CSYSVX/CHESVX only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, and i is
≤ N: D(i,i) is exactly zero. The factorization has has been completed, but the block diagonal matrix D is exactly singular, so the solution and error bounds could not be computed. RCOND = 0 is returned.
= N+1: D is nonsingular, but RCOND is less than machine precision, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of RCOND would suggest.

SSYTRD/CHETRD

```
      SUBROUTINE SSYTRD( UPLO, N, A, LDA, D, E, TAU, WORK, LWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LWORK, N
      REAL               A( LDA, * ), D( * ), E( * ), TAU( * ),
     $                   WORK( * )

      SUBROUTINE CHETRD( UPLO, N, A, LDA, D, E, TAU, WORK, LWORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LWORK, N
      REAL               D( * ), E( * )
      COMPLEX            A( LDA, * ), TAU( * ), WORK( * )
```

Purpose

SSYTRD/CHETRD reduces a real/complex symmetric/Hermitian matrix A to real symmetric tridiagonal form T by an orthogonal/unitary similarity transformation: $Q^H*A*Q = T$.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N ≥ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A, and the strictly upper triangular part of A is not referenced.
On exit, if UPLO = 'U', the diagonal and first superdiagonal of A are overwritten by the corresponding elements of the tridiagonal matrix T, and the elements above the first superdiagonal, with the array TAU, represent the orthogonal/unitary matrix Q as a product of elementary reflectors; if UPLO = 'L', the diagonal and first subdiagonal of A are over- written by the corresponding elements of the tridiagonal matrix T, and the elements below the first subdiagonal, with the array TAU, represent the orthogonal/unitary matrix Q as a product of elementary reflectors.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,N).

D (output) REAL array, dimension (N)
The diagonal elements of the tridiagonal matrix T: D(i) = A(i,i).

E (output) REAL array, dimension (N−1)
The off-diagonal elements of the tridiagonal matrix T: E(i) = A(i,i+1) if UPLO = 'U', E(i) = A(i+1,i) if UPLO = 'L'.

TAU (output) REAL/COMPLEX array, dimension (N−1)
The scalar factors of the elementary reflectors.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK ≥ 1. For optimum performance LWORK ≥ N*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

SSYTRF/CSYTRF/CHETRF

```
SUBROUTINE SSYTRF( UPLO, N, A, LDA, IPIV, WORK, LWORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, LWORK, N
INTEGER            IPIV( * )
REAL               A( LDA, * ), WORK( LWORK )

SUBROUTINE CSYTRF( UPLO, N, A, LDA, IPIV, WORK, LWORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, LWORK, N
INTEGER            IPIV( * )
COMPLEX            A( LDA, * ), WORK( LWORK )

SUBROUTINE CHETRF( UPLO, N, A, LDA, IPIV, WORK, LWORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, LWORK, N
INTEGER            IPIV( * )
COMPLEX            A( LDA, * ), WORK( LWORK )
```

Purpose

SSYTRF/CSYTRF/CHETRF computes the factorization of a real/complex/complex symmetric/symmetric/Hermitian matrix A using the diagonal pivoting method. The form of the factorization is

$A = U*D*U^T$ or $A = L*D*L^T$ (SSYTRF/CSYTRF) or

$A = U*D*U^H$ or $A = L*D*L^H$ (CHETRF),

where U (or L) is a product of permutation and unit upper (lower) triangular matrices, and D is symmetric (SSYTRF/CSYTRF) or Hermitian (CHETRF) and block diagonal with 1-by-1 and 2-by-2 diagonal blocks.

Arguments

UPLO (input) CHARACTER*1
= 'U': Upper triangle of A is stored;
= 'L': Lower triangle of A is stored.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input/output)
REAL/COMPLEX/COMPLEX array, dimension (LDA,N)
On entry, the symmetric/symmetric/Hermitian matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of A contains the upper triangular part of the matrix A, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of A contains the lower triangular part of the matrix A, and the strictly upper triangular part of A is not referenced.
On exit, the block diagonal matrix D and the multipliers used to obtain the factor U or L.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

IPIV (output) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D.
If IPIV(k) > 0, then rows and columns k and IPIV(k) were interchanged and D(k,k) is a 1-by-1 diagonal block.
If UPLO = 'U' and IPIV(k) = IPIV(k−1) < 0, then rows and columns k−1 and −IPIV(k) were interchanged and D(k−1:k,k−1:k) is a 2-by-2 diagonal block. If UPLO = 'L' and IPIV(k) = IPIV(k+1) < 0, then rows and columns k+1 and −IPIV(k) were interchanged and D(k:k+1,k:k+1) is a 2-by-2 diagonal block.

WORK (workspace/output)
REAL/COMPLEX/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The length of WORK. LWORK $\geq$1. For best performance LWORK $\geq$ N*NB, where NB is the block size returned by ILAENV.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, D(i,i) is exactly zero. The factorization has been completed, but the block diagonal matrix D is exactly singular, and division by zero will occur if it is used to solve a system of equations.

SSYTRI/CSYTRI/CHETRI

```
SUBROUTINE SSYTRI( UPLO, N, A, LDA, IPIV, WORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, N
INTEGER            IPIV( * )
REAL               A( LDA, * ), WORK( * )

SUBROUTINE CSYTRI( UPLO, N, A, LDA, IPIV, WORK, INFO )
CHARACTER          UPLO
INTEGER            INFO, LDA, N
INTEGER            IPIV( * )
COMPLEX            A( LDA, * ), WORK( * )
```

```
      SUBROUTINE CHETRI( UPLO, N, A, LDA, IPIV, WORK, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, N
      INTEGER            IPIV( * )
      COMPLEX            A( LDA, * ), WORK( * )
```

Purpose

SSYTRI/CSYTRI/CHETRI computes the inverse of a real/complex/complex symmetric/symmetric/Hermitian indefinite matrix A using the factorization $A = U*D*U^T$ or $A = L*D*L^T$ computed by SSYTRF/CSYTRF or the factorization $A = U*D*U^H$ or $A = L*D*L^H$ computed by CHETRF.

Arguments

UPLO (input) CHARACTER*1
Specifies whether the details of the factorization are stored as an upper or lower triangular matrix.
= 'U': Upper triangular, form is $A = U*D*U^T$ (SSYTRI/CSYTRI) or $A = U*D*U^H$ (CHETRI);
= 'L': Lower triangular, form is $A = L*D*L^T$ (SSYTRI/CSYTRI) or $A = L*D*L^H$ (CHETRI)

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

A (input/output)
REAL/COMPLEX/COMPLEX array, dimension (LDA,N)
On entry, the block diagonal matrix D and the multipliers used to obtain the factor U or L as computed by SSYTRF/CSYTRF/CHETRF.
On exit, if INFO = 0, the (symmetric/symmetric/Hermitian) inverse of the original matrix. If UPLO = 'U', the upper triangular part of the inverse is formed and the part of A below the diagonal is not referenced; if UPLO = 'L' the lower triangular part of the inverse is formed and the part of A above the diagonal is not referenced.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

IPIV (input) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D as determined by SSYTRF/CSYTRF/CHETRF.

WORK *SSYTRI* (workspace) REAL array, dimension (N)
CSYTRI/CHETRI (workspace) COMPLEX array, dimension (2*N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, D(i,i) = 0; the matrix is singular and its inverse could not be computed.

SSYTRS/CSYTRS/CHETRS

```
      SUBROUTINE SSYTRS( UPLO, N, NRHS, A, LDA, IPIV, B, LDB, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDB, N, NRHS
      INTEGER            IPIV( * )
      REAL               A( LDA, * ), B( LDB, * )

      SUBROUTINE CSYTRS( UPLO, N, NRHS, A, LDA, IPIV, B, LDB, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDB, N, NRHS
      INTEGER            IPIV( * )
      COMPLEX            A( LDA, * ), B( LDB, * )

      SUBROUTINE CHETRS( UPLO, N, NRHS, A, LDA, IPIV, B, LDB, INFO )
      CHARACTER          UPLO
      INTEGER            INFO, LDA, LDB, N, NRHS
      INTEGER            IPIV( * )
      COMPLEX            A( LDA, * ), B( LDB, * )
```

Purpose

SSYTRS/CSYTRS/CHETRS solves a system of linear equations $A*X = B$ with a real/complex/complex symmetric/symmetric/Hermitian matrix A using the factorization $A = U*D*U^T$ or $A = L*D*L^T$ computed by SSYTRF/CSYTRF or the factorization $A = U*D*U^H$ or $A = L*D*L^H$ computed by CHETRF.

Arguments

UPLO (input) CHARACTER*1
Specifies whether the details of the factorization are stored as an upper or lower triangular matrix.
= 'U': Upper triangular, form is $A = U*D*U^T$ (SSYTRS/CSYTRS) or $A = U*D*U^H$ (CHETRS);
= 'L': Lower triangular, form is $A = L*D*L^T$ (SSYTRS/CSYTRS) or $A = L*D*L^H$ (CHETRS).

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS ≥ 0.

A (input) REAL/COMPLEX/COMPLEX array, dimension (LDA,N)
The block diagonal matrix D and the multipliers used to obtain the factor U or L as computed by SSYTRF/CSYTRF/CHETRF.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

IPIV (input) INTEGER array, dimension (N)
Details of the interchanges and the block structure of D as determined by SSYTRF/CSYTRF/CHETRF.

B (input/output)
REAL/COMPLEX/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = $-$i, the ith argument had an illegal value.

STBCON/CTBCON

```
      SUBROUTINE STBCON( NORM, UPLO, DIAG, N, KD, AB, LDAB, RCOND, WORK,
     $                   IWORK, INFO )
      CHARACTER          DIAG, NORM, UPLO
      INTEGER            INFO, KD, LDAB, N
      REAL               RCOND
      INTEGER            IWORK( * )
      REAL               AB( LDAB, * ), WORK( * )

      SUBROUTINE CTBCON( NORM, UPLO, DIAG, N, KD, AB, LDAB, RCOND, WORK,
     $                   RWORK, INFO )
      CHARACTER          DIAG, NORM, UPLO
      INTEGER            INFO, KD, LDAB, N
      REAL               RCOND
      REAL               RWORK( * )
      COMPLEX            AB( LDAB, * ), WORK( * )
```

Purpose

STBCON/CTBCON estimates the reciprocal of the condition number of a triangular band matrix A, in either the 1-norm or the infinity-norm.

The norm of A is computed and an estimate is obtained for $||A^{-1}||$, then the reciprocal of the condition number is computed as RCOND = $1/(||A|| * ||A^{-1}||)$.

Arguments

NORM (input) CHARACTER*1
Specifies whether the 1-norm condition number or the infinity-norm condition number is required:
= '1' or 'O': 1-norm;
= 'I': Infinity-norm.

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

KD (input) INTEGER
The number of superdiagonals or subdiagonals of the triangular band matrix A. KD $\geq$ 0.

AB (input) REAL/COMPLEX array, dimension (LDAB,N)
The upper or lower triangular band matrix A, stored in the first kd+1 rows of the array. The jth column of A is stored in the jth column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i$-$j,j) = A(i,j) for max(1,j$-$kd)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i$-$j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+kd).
If DIAG = 'U', the diagonal elements of A are not referenced and are assumed to be 1.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KD+1.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as RCOND = $1/(||A|| * ||A^{-1}||)$.

WORK *STBCON* (workspace) REAL array, dimension (3*N)
CTBCON (workspace) COMPLEX array, dimension (2*N)

IWORK *STBCON only* (workspace) INTEGER array, dimension (N)

RWORK *CTBCON only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = $-$i, the ith argument had an illegal value.

STBRFS/CTBRFS

```
      SUBROUTINE STBRFS( UPLO, TRANS, DIAG, N, KD, NRHS, AB, LDAB, B,
     $                   LDB, X, LDX, FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, KD, LDAB, LDB, LDX, N, NRHS
      INTEGER            IWORK( * )
      REAL               AB( LDAB, * ), B( LDB, * ), BERR( * ),
     $                   FERR( * ), WORK( * ), X( LDX, * )
```

```
      SUBROUTINE CTBRFS( UPLO, TRANS, DIAG, N, KD, NRHS, AB, LDAB, B,
     $                   LDB, X, LDX, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, KD, LDAB, LDB, LDX, N, NRHS
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            AB( LDAB, * ), B( LDB, * ), WORK( * ),
     $                   X( LDX, * )
```

Purpose

STBRFS/CTBRFS provides error bounds and backward error estimates for the solution to a system of linear equations with a triangular band coefficient matrix.

The solution matrix X must be computed by STBTRS/CTBTRS or some other means before entering this routine. STBRFS/CTBRFS does not do iterative refinement because doing so cannot improve the backward error.

Arguments

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)
= 'C': A^H*X = B (Conjugate transpose)

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

KD (input) INTEGER
The number of superdiagonals or subdiagonals of the triangular band matrix A. KD $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

AB (input) REAL/COMPLEX array, dimension (LDAB,N)
The upper or lower triangular band matrix A, stored in the first kd+1 rows of the array. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for max(1,j−kd)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+kd).
If DIAG = 'U', the diagonal elements of A are not referenced and are assumed to be 1.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KD+1.

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (input) REAL/COMPLEX array, dimension (LDX,NRHS)
The solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *STBRFS* (workspace) REAL array, dimension (3*N)
CTBRFS (workspace) COMPLEX array, dimension (2*N)

IWORK *STBRFS only* (workspace) INTEGER array, dimension (N)

RWORK *CTBRFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

STBTRS/CTBTRS

```
      SUBROUTINE STBTRS( UPLO, TRANS, DIAG, N, KD, NRHS, AB, LDAB, B,
     $                   LDB, INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, KD, LDAB, LDB, N, NRHS
      REAL               AB( LDAB, * ), B( LDB, * )

      SUBROUTINE CTBTRS( UPLO, TRANS, DIAG, N, KD, NRHS, AB, LDAB, B,
     $                   LDB, INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, KD, LDAB, LDB, N, NRHS
      COMPLEX            AB( LDAB, * ), B( LDB, * )
```

Purpose

STBTRS/CTBTRS solves a triangular system of the form $A*X = B$, $A^T*X = B$, or $A^H*X = B$, where A is a triangular band matrix of order n, and B is an n-by-nrhs matrix. A check is made to verify that A is nonsingular.

Arguments

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': $A*X = B$ (No transpose)
= 'T': $A^T*X = B$ (Transpose)
= 'C': $A^H*X = B$ (Conjugate transpose)

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

KD (input) INTEGER
The number of superdiagonals or subdiagonals of the triangular band matrix A. KD $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS $\geq$ 0.

AB (input) REAL/COMPLEX array, dimension (LDAB,N)
The upper or lower triangular band matrix A, stored in the first kd+1 rows of AB. The j^{th} column of A is stored in the j^{th} column of the array AB as follows:
if UPLO = 'U', AB(kd+1+i−j,j) = A(i,j) for max(1,j−kd)$\leq$i$\leq$j;
if UPLO = 'L', AB(1+i−j,j) = A(i,j) for j$\leq$i$\leq$min(n,j+kd).
If DIAG = 'U', the diagonal elements of A are not referenced and are assumed to be 1.

LDAB (input) INTEGER
The leading dimension of the array AB. LDAB $\geq$ KD+1.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, if INFO = 0, the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the i^{th} diagonal element of A is zero, indicating that the matrix is singular and the solutions X have not been computed.

STGEVC/CTGEVC

```
      SUBROUTINE STGEVC( SIDE, HOWMNY, SELECT, N, A, LDA, B, LDB, VL,
     $                   LDVL, VR, LDVR, MM, M, WORK, INFO )
      CHARACTER          HOWMNY, SIDE
      INTEGER            INFO, LDA, LDB, LDVL, LDVR, M, MM, N
      LOGICAL            SELECT( * )
      REAL               A( LDA, * ), B( LDB, * ), VL( LDVL, * ),
     $                   VR( LDVR, * ), WORK( N, * )

      SUBROUTINE CTGEVC( SIDE, HOWMNY, SELECT, N, A, LDA, B, LDB, VL,
     $                   LDVL, VR, LDVR, MM, M, WORK, RWORK, INFO )
      CHARACTER          HOWMNY, SIDE
      INTEGER            INFO, LDA, LDB, LDVL, LDVR, M, MM, N
      LOGICAL            SELECT( * )
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), VL( LDVL, * ),
     $                   VR( LDVR, * ), WORK( * )
```

Purpose

STGEVC/CTGEVC computes some or all of the right and/or left generalized eigenvectors of a pair of real/complex upper triangular matrices (A,B).

The right generalized eigenvector x and the left generalized eigenvector y of (A,B) corresponding to a generalized eigenvalue w are defined by:

$$(A - w*B)*x = 0 \quad \text{and} \quad y^H(A - w*B) = 0$$

If an eigenvalue w is determined by zero diagonal elements of both A and B, a unit vector is returned as the corresponding eigenvector.

If all eigenvectors are requested, the routine may either return the matrices X and/or Y of right or left eigenvectors of (A,B), or the products Z*X and/or Q*Y, where Z and Q are input orthogonal/unitary matrices. If (A,B) was obtained from the generalized real-Schur/Schur factorization of an original pair of matrices $(A_0,B_0) = (Q*A*Z^H, Q*B*Z^H)$, then Z*X and Q*Y are the matrices of right or left eigenvectors of A.

STGEVC only
A must be block upper triangular, with 1-by-1 and 2-by-2 diagonal blocks. Corresponding to each 2-by-2 diagonal block is a complex conjugate pair of eigenvalues and eigenvectors; only one eigenvector of the pair is computed, namely the one corresponding to the eigenvalue with positive imaginary part.

Arguments

SIDE (input) CHARACTER*1
= 'R': compute right eigenvectors only;
= 'L': compute left eigenvectors only;
= 'B': compute both right and left eigenvectors.

HOWMNY (input) CHARACTER*1
= 'A': compute all right and/or left eigenvectors;
= 'B': compute all right and/or left eigenvectors, and backtransform them using the input matrices supplied in VR and/or VL;
= 'S': compute selected right and/or left eigenvectors, specified by the logical array SELECT.

SELECT (input) LOGICAL array, dimension (N)
If HOWMNY='S', SELECT specifies the eigenvectors to be computed. If HOWMNY='A' or 'B', SELECT is not referenced.
STGEVC
To select the real eigenvector corresponding to a real eigenvalue w(j), SELECT(j) must be set to .TRUE.. To select the complex eigenvector corresponding to a complex conjugate pair w(j) and w(j+1), either SELECT(j) or SELECT(j+1) must be set to .TRUE..
CTGEVC
To select the eigenvector corresponding to the j^{th} eigenvalue, SELECT(j) must be set to .TRUE..

N (input) INTEGER
The order of the matrices A and B. N $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
STGEVC
The upper quasi-triangular matrix A.
CTGEVC
The upper triangular matrix A.

LDA (input) INTEGER
The leading dimension of array A. LDA $\geq$ max(1,N).

B (input) REAL/COMPLEX array, dimension (LDB,N)
The upper triangular matrix B.
STGEVC
If A has a 2-by-2 diagonal block, then the corresponding 2-by-2 block of B must be diagonal with positive elements.
CTGEVC
B must have real diagonal elements.

LDB (input) INTEGER
The leading dimension of array B. LDB $\geq$ max(1,N).

VL (input/output) REAL/COMPLEX array, dimension (LDVL,MM)
On entry, if SIDE = 'L' or 'B' and HOWMNY = 'B', VL must contain an n-by-n matrix Q (usually the orthogonal/unitary matrix Q of left Schur vectors returned by SHGEQZ/CHGEQZ).
On exit, if SIDE = 'L' or 'B', VL contains:
if HOWMNY = 'A', the matrix Y of left eigenvectors of (A,B);
if HOWMNY = 'B', the matrix Q*Y;
if HOWMNY = 'S', the left eigenvectors of (A,B) specified by SELECT, stored consecutively in the columns of VL, in the same order as their eigenvalues.
If SIDE = 'R', VL is not referenced.
STGEVC only
A complex eigenvector corresponding to a complex eigenvalue is stored in two consecutive columns, the first holding the real part, and the second the imaginary part.

LDVL (input) INTEGER
The leading dimension of array VL.
LDVL $\geq$ max(1,N) if SIDE = 'L' or 'B'; LDVL $\geq$ 1 otherwise.

VR (input/output) REAL/COMPLEX array, dimension (LDVR,MM)
On entry, if SIDE = 'R' or 'B' and HOWMNY = 'B', VR must contain an n-by-n matrix Q (usually the orthogonal/unitary matrix Z of right Schur vectors returned by SHGEQZ/CHGEQZ).
On exit, if SIDE = 'R' or 'B', VR contains:
if HOWMNY = 'A', the matrix X of right eigenvectors of (A,B);
if HOWMNY = 'B', the matrix Z*X;
if HOWMNY = 'S', the right eigenvectors of (A,B) specified by SELECT, stored consecutively in the columns of VR, in the same order as their eigenvalues.
If SIDE = 'L', VR is not referenced.
STGEVC only
A complex eigenvector corresponding to a complex eigenvalue is stored in two consecutive columns, the first holding the real part and the second the imaginary part.

LDVR (input) INTEGER
The leading dimension of the array VR.
LDVR $\geq$ max(1,N) if SIDE = 'R' or 'B'; LDVR $\geq$ 1 otherwise.

MM (input) INTEGER
The number of columns in the arrays VL and/or VR. MM $\geq$ M.

M (output) INTEGER
The number of columns in the arrays VL and/or VR actually used to store the eigenvectors. If HOWMNY = 'A' or 'B', M is set to N.
STGEVC
Each selected real eigenvector occupies one column and each selected complex eigenvector occupies two columns.
CTGEVC
Each selected eigenvector occupies one column.

WORK *STGEVC* (workspace) REAL array, dimension (6*N)
CTGEVC (workspace) COMPLEX array, dimension (2*N)

RWORK *CTGEVC only* (workspace) REAL array, dimension (2*N)

INFO (output) INTEGER

= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: (*STGEVC only*) the 2-by-2 block (INFO:INFO+1) does not have a complex eigenvalue.

STGEXC/CTGEXC

```
      SUBROUTINE STGEXC( WANTQ, WANTZ, N, A, LDA, B, LDB, Q, LDQ, Z,
     $                   LDZ, IFST, ILST, WORK, LWORK, INFO )
      LOGICAL            WANTQ, WANTZ
      INTEGER            IFST, ILST, INFO, LDA, LDB, LDQ, LDZ, LWORK, N
      REAL               A( LDA, * ), B( LDB, * ), Q( LDQ, * ),
     $                   WORK( * ), Z( LDZ, * )

      SUBROUTINE CTGEXC( WANTQ, WANTZ, N, A, LDA, B, LDB, Q, LDQ, Z,
     $                   LDZ, IFST, ILST, INFO )
      LOGICAL            WANTQ, WANTZ
      INTEGER            IFST, ILST, INFO, LDA, LDB, LDQ, LDZ, N
      COMPLEX            A( LDA, * ), B( LDB, * ), Q( LDQ, * ),
     $                   Z( LDZ, * )
```

Purpose

STGEXC/CTGEXC reorders the generalized real-Schur/Schur decomposition of a real/complex matrix pair (A,B) using an orthogonal/unitary equivalence transformation

$$(A, B) = Q * (A, B) * Z',$$

so that the diagonal block of (A, B) with row index IFST is moved to row ILST.

(A, B) must be in generalized real-Schur/Schur canonical form (as returned by SGGES/CGGES), i.e. A is block upper triangular with 1-by-1 and 2-by-2 diagonal blocks. B is upper triangular.

Optionally, the matrices Q and Z of generalized Schur vectors are updated.

$$Q(in) * A(in) * Z(in)' = Q(out) * A(out) * Z(out)'$$
$$Q(in) * B(in) * Z(in)' = Q(out) * B(out) * Z(out)'$$

Arguments

WANTQ (input) LOGICAL
= .TRUE.: update the left transformation matrix Q;
= .FALSE.: do not update Q.

WANTZ (input) LOGICAL
= .TRUE.: update the right transformation matrix Z;
= .FALSE.: do not update Z.

N (input) INTEGER
The order of the matrices A and B. N ≥ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
STGEXC
On entry, the matrix A in generalized real Schur canonical form.
On exit, the updated matrix A, again in generalized real Schur canonical form.
CTGEXC
On entry, the upper triangular matrix A in the pair (A, B).
On exit, th : updated matrix A.

LDA (input) INTEGER
The leading dimension of array A. LDA ≥ max(1,N).

B (input) REAL/COMPLEX array, dimension (LDB,N)
STGEXC
On entry, the matrix B in generalized real Schur canonical form (A,B).
On exit, the updated matrix B, again in generalized real Schur canonical form (A,B).
CTGEXC
On entry, the upper triangular matrix B in the pair (A, B). On exit, the updated matrix B.

LDB (input) INTEGER
The leading dimension of array B. LDB ≥ max(1,N).

Q (input/output) REAL/COMPLEX array, dimension (LDZ,N)
On entry, if WANTQ = .TRUE., the orthogonal/unitary matrix Q.
On exit, the updated matrix Q.
If WANTQ = .FALSE., Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q. LDQ ≥ 1.
If WANTQ = .TRUE., LDQ ≥ N.

Z (input/output) REAL/COMPLEX array, dimension (LDZ,N)
On entry, if WANTZ = .TRUE., the orthogonal/unitary matrix Z.
On exit, the updated matrix Z.
If WANTZ = .FALSE., Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ ≥ 1.
If WANTZ = .TRUE., LDZ ≥ N.

IFST (input/output) INTEGER

ILST (input/output) INTEGER
Specify the reordering of the diagonal blocks of (A, B). The block with row index IFST is moved to row ILST, by a sequence of swapping between adjacent blocks.
STGEXC

On exit, if IFST pointed on entry to the second row of a 2-by-2 block, it is changed to point to the first row; ILST always points to the first row of the block in its final position (which may differ from its input value by +1 or -1). $1 \leq$ IFST, ILST $\leq$ N.

WORK — *STGEXC only* (workspace) REAL array, dimension (LWORK)

LWORK — *STGEXC only* (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ 4*N + 16.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO — (output) INTEGER
- = 0: successful exit
- < 0: if INFO = −i, the i^{th} argument had an illegal value.
- = 1: The transformed matrix pair (A, B) would be too far from generalized Schur form; the problem is ill- conditioned. (A, B) may have been partially reordered, and ILST points to the first row of the current position of the block being moved.

STGSEN/CTGSEN

```
      SUBROUTINE STGSEN( IJOB, WANTQ, WANTZ, SELECT, N, A, LDA, B, LDB,
     $                   ALPHAR, ALPHAI, BETA, Q, LDQ, Z, LDZ, M, PL,
     $                   PR, DIF, WORK, LWORK, IWORK, LIWORK, INFO )
      LOGICAL            WANTQ, WANTZ
      INTEGER            IJOB, INFO, LDA, LDB, LDQ, LDZ, LIWORK, LWORK,
     $                   M, N
      REAL               PL, PR
      LOGICAL            SELECT( * )
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), ALPHAI( * ), ALPHAR( * ),
     $                   B( LDB, * ), BETA( * ), DIF( * ), Q( LDQ, * ),
     $                   WORK( * ), Z( LDZ, * )
```

```
      SUBROUTINE CTGSEN( IJOB, WANTQ, WANTZ, SELECT, N, A, LDA, B, LDB,
     $                   ALPHA, BETA, Q, LDQ, Z, LDZ, M, PL, PR, DIF,
     $                   WORK, LWORK, IWORK, LIWORK, INFO )
      LOGICAL            WANTQ, WANTZ
      INTEGER            IJOB, INFO, LDA, LDB, LDQ, LDZ, LIWORK, LWORK,
     $                   M, N
      REAL               PL, PR
      LOGICAL            SELECT( * )
      INTEGER            IWORK( * )
      REAL               DIF( * )
      COMPLEX            A( LDA, * ), ALPHA( * ), B( LDB, * ),
     $                   BETA( * ), Q( LDQ, * ), WORK( * ), Z( LDZ, * )
```

Purpose

STGSEN/CTGSEN reorders the generalized real-Schur/Schur decomposition of a real/complex matrix pair (A, B) (in terms of an orthogonal/unitary equivalence transformation Q' * (A, B) * Z), so that a selected cluster of eigenvalues appears in the leading diagonal blocks of the pair (A,B). The leading columns of Q and Z form orthonormal/unitary bases of the corresponding left and right eigenspaces (deflating subspaces). (A, B) must be in generalized real-Schur/Schur canonical form (as returned by SGGES/CGGES), that is, A and B are both upper triangular.

STGSEN/CTGSEN also computes the generalized eigenvalues

$$w(j) = (ALPHAR(j) + i * ALPHAI(j))/BETA(j)(STGSEN)$$
$$w(j) = ALPHA(j)/BETA(j)(CTGSEN)$$

of the reordered matrix pair (A, B).

Optionally, the routine computes estimates of reciprocal condition numbers for eigenvalues and eigenspaces. These are Difu[(A11,B11), (A22,B22)] and Difl[(A11,B11), (A22,B22)], i.e. the separation(s) between the matrix pairs (A11, B11) and (A22,B22) that correspond to the selected cluster and the eigenvalues outside the cluster, resp., and norms of "projections" onto left and right eigenspaces w.r.t. the selected cluster in the (1,1)-block.

Arguments

IJOB — (input) INTEGER
Specifies whether condition numbers are required for the cluster of eigenvalues (PL and PR) or the deflating subspaces (Difu and Difl):
- = 0: Only reorder w.r.t. SELECT. No extras.
- = 1: Reciprocal of norms of "projections" onto left and right eigenspaces w.r.t. the selected cluster (PL and PR).
- = 2: Upper bounds on Difu and Difl. F-norm-based estimate (DIF(1:2)).
- = 3: Estimate of Difu and Difl. 1-norm-based estimate (DIF(1:2)). About 5 times as expensive as IJOB = 2.
- = 4: Compute PL, PR and DIF (i.e. 0, 1 and 2 above): Economic version to get it all.

= 5: Compute PL, PR and DIF (i.e. 0, 1 and 3 above)

WANTQ (input) LOGICAL
= .TRUE.: update the left transformation matrix Q;
= .FALSE.: do not update Q.

WANTZ (input) LOGICAL
= .TRUE.: update the right transformation matrix Z;
= .FALSE.: do not update Z.

SELECT (input) LOGICAL array, dimension (N)
STGSEN
SELECT specifies the eigenvalues in the selected cluster. To select a real eigenvalue w(j), SELECT(j) must be set to .TRUE.. To select a complex conjugate pair of eigenvalues w(j) and w(j+1), corresponding to a 2-by-2 diagonal block, either SELECT(j) or SELECT(j+1) or both must be set to .TRUE.; a complex conjugate pair of eigenvalues must be either both included in the cluster or both excluded.
CTGSEN
SELECT specifies the eigenvalues in the selected cluster. To select an eigenvalue w(j), SELECT(j) must be set to .TRUE..

N (input) INTEGER
The order of the matrices A and B. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
STGSEN
On entry, the upper quasi-triangular matrix A, with (A, B) in generalized real Schur canonical form.
On exit, A is overwritten by the reordered matrix A.
CTGSEN
On entry, the upper triangular matrix A, in generalized Schur canonical form.
On exit, A is overwritten by the reordered matrix A.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB,N)
STGSEN
On entry, the upper triangular matrix B, with (A, B) in generalized real Schur canonical form.
On exit, B is overwritten by the reordered matrix B.
CTGSEN
On entry, the upper triangular matrix B, in generalized Schur canonical form.
On exit, B is overwritten by the reordered matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

ALPHAR *STGSEN only* (output) REAL array, dimension (N)

ALPHAI *STGSEN only* (output) REAL array, dimension (N)

ALPHA *CTGSEN only* (output) COMPLEX array, dimension (N)

BETA (output) REAL/COMPLEX array, dimension (N)
STGSEN
On exit, (ALPHAR(j) + ALPHAI(j)*i)/BETA(j), j=1,...,N, will be the generalized eigenvalues. ALPHAR(j) + ALPHAI(j)*i and BETA(j),j=1,...,N are the diagonals of the complex Schur form (S,T) that would result if the 2-by-2 diagonal blocks of the real generalized Schur form of (A,B) were further reduced to triangular form using complex unitary transformations. If ALPHAI(j) is zero, then the j^{th} eigenvalue is real; if positive, then the j^{th} and $(j+1)^{st}$ eigenvalues are a complex conjugate pair, with ALPHAI(j+1) negative.
CTGSEN
The diagonal elements of A and B, respectively, when the pair (A,B) has been reduced to generalized Schur form. ALPHA(i)/BETA(i) i=1,...,N are the generalized eigenvalues.

Q (input/output) REAL/COMPLEX array, dimension (LDQ,N)
On entry, if WANTQ = .TRUE., Q is an N-by-N matrix. On exit, Q has been postmultiplied by the left orthogonal/unitary transformation matrix which reorder (A, B); The leading M columns of Q form orthonormal/unitary bases for the specified pair of left eigenspaces (deflating subspaces).
If WANTQ = .FALSE., Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q. LDQ $\geq$ 1;
and if WANTQ = .TRUE., LDQ $\geq$ N.

Z (input/output) REAL/COMPLEX array, dimension (LDZ,N)
On entry, if WANTZ = .TRUE., Z is an N-by-N matrix.
On exit, Z has been postmultiplied by the left orthogonal/unitary transformation matrix which reorder (A, B); The leading M columns of Z form orthonormal/unitary bases for the specified pair of left eigenspaces (deflating subspaces).
If WANTZ = .FALSE., Z is not referenced.

LDZ (input) INTEGER
The leading dimension of the array Z. LDZ $\geq$ 1;
If WANTZ = .TRUE., LDZ $\geq$ N.

M (output) INTEGER
The dimension of the specified pair of left and right eigen- spaces (deflating subspaces). 0 $\leq$ M $\leq$ N.

PL, PR (output) REAL
If IJOB = 1, 4 or 5, PL, PR are lower bounds on the reciprocal of the norm of "projections" onto left and right eigenspaces with respect to the selected cluster. 0 < PL, PR $\leq$ 1. If M = 0 or M = N, PL = PR = 1. If IJOB = 0, 2 or 3, PL and PR are not referenced.

DIF (output) REAL array, dimension (2)
If IJOB $\geq$ 2, DIF(1:2) store the estimates of Difu and Difl. If IJOB =

2 or 4, DIF(1:2) are F-norm-based upper bounds on Difu and Difl. If IJOB = 3 or 5, DIF(1:2) are 1-norm-based estimates of Difu and Difl. If M = 0 or N, DIF(1:2) = F-norm([A, B]). If IJOB = 0 or 1, DIF is not referenced.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
If IJOB = 0, WORK is not referenced. Otherwise, on exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
STGSEN
The dimension of the array WORK. LWORK $\geq$ 4*N+16.
If IJOB = 1, 2 or 4, LWORK $\geq$ MAX(4*N+16, 2*M*(N−M)).
If IJOB = 3 or 5, LWORK $\geq$ MAX(4*N+16, 4*M*(N−M)).
CTGSEN
The dimension of the array WORK. LWORK $\geq$ 1.
If IJOB = 1, 2 or 4, LWORK $\geq$ 2*M*(N−M).
If IJOB = 3 or 5, LWORK $\geq$ 4*M*(N−M).

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

IWORK (workspace/output) INTEGER array, dimension (LIWORK)
IF IJOB = 0, IWORK is not referenced. Otherwise, on exit, if INFO = 0, IWORK(1) returns the optimal LIWORK.

LIWORK (input) INTEGER
STGSEN
The dimension of the array IWORK. LIWORK $\geq$ 1. If IJOB = 1, 2 or 4, LIWORK $\geq$ N+6. If IJOB = 3 or 5, LIWORK $\geq$ MAX(2*M*(N−M), N+6).
CTGSEN
The dimension of the array IWORK. LIWORK $\geq$ 1.
If IJOB = 1, 2 or 4, LIWORK $\geq$ N+2.
If IJOB = 3 or 5, LIWORK $\geq$ MAX(N+2, 2*M*(N−M)).

If LIWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the IWORK array, returns this value as the first entry of the IWORK array, and no error message related to LIWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
= 1: Reordering of (A, B) failed because the transformed matrix pair (A, B) would be too far from generalized Schur form; the problem is very ill-conditioned. (A, B) may have been partially reordered. If requested, 0 is returned in DIF(*), PL and PR.

STGSJA/CTGSJA

```
      SUBROUTINE STGSJA( JOBU, JOBV, JOBQ, M, P, N, K, L, A, LDA, B,
     $                   LDB, TOLA, TOLB, ALPHA, BETA, U, LDU, V, LDV,
     $                   Q, LDQ, WORK, NCYCLE, INFO )
      CHARACTER          JOBQ, JOBU, JOBV
      INTEGER            INFO, K, L, LDA, LDB, LDQ, LDU, LDV, M, N,
     $                   NCYCLE, P
      REAL               TOLA, TOLB
      REAL               ALPHA( * ), BETA( * ), A( LDA, * ),
     $                   B( LDB, * ), Q( LDQ, * ), U( LDU, * ),
     $                   V( LDV, * ), WORK( * )

      SUBROUTINE CTGSJA( JOBU, JOBV, JOBQ, M, P, N, K, L, A, LDA, B,
     $                   LDB, TOLA, TOLB, ALPHA, BETA, U, LDU, V, LDV,
     $                   Q, LDQ, WORK, NCYCLE, INFO )
      CHARACTER          JOBQ, JOBU, JOBV
      INTEGER            INFO, K, L, LDA, LDB, LDQ, LDU, LDV, M, N,
     $                   NCYCLE, P
      REAL               TOLA, TOLB
      REAL               ALPHA( * ), BETA( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), Q( LDQ, * ),
     $                   U( LDU, * ), V( LDV, * ), WORK( * )
```

Purpose

STGSJA/CTGSJA computes the generalized singular value decomposition (GSVD) of two real/complex upper triangular (or trapezoidal) matrices A and B.

On entry, it is assumed that matrices A and B have the following forms, which may be obtained by the preprocessing subroutine SGGSVP/CGGSVP from a general m-by-n matrix A and p-by-n matrix B:

$$A = \begin{array}{c} \\ k \\ l \\ m-k-l \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & A_{12} & A_{13} \\ 0 & 0 & A_{23} \\ 0 & 0 & 0 \end{pmatrix} \end{array} \text{ if } m-k-l \geq 0;$$

$$= \begin{array}{c} \\ k \\ m-k \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & A_{12} & A_{13} \\ 0 & 0 & A_{23} \end{pmatrix} \end{array} \text{ if } m-k-l < 0;$$

$$B = \begin{array}{c} \\ l \\ p-l \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & 0 & B_{13} \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

where the k-by-k matrix A_{12} and l-by-l matrix B_{13} are nonsingular upper triangular; A_{23} is l-by-l upper triangular if m−k−l $\geq$ 0, otherwise A_{23} is (m−k)-by-l upper trapezoidal.

On exit,

$$U^H * A * Q = D_1 * \begin{pmatrix} 0 & R \end{pmatrix}, \quad V^H * B * Q = D_2 * \begin{pmatrix} 0 & R \end{pmatrix},$$

where U, V and Q are orthogonal/unitary matrices, R is a nonsingular upper triangular matrix, and D_1 and D_2 are "diagonal" matrices, which are of the following structures:

If $m-k-l \geq 0$,

$$D_1 = \begin{array}{c} \\ k \\ l \\ m-k-l \end{array} \begin{array}{c} \begin{array}{cc} k & l \end{array} \\ \begin{pmatrix} I & 0 \\ 0 & C \\ 0 & 0 \end{pmatrix} \end{array}$$

$$D_2 = \begin{array}{c} \\ l \\ p-l \end{array} \begin{array}{c} \begin{array}{cc} k & l \end{array} \\ \begin{pmatrix} 0 & S \\ 0 & 0 \end{pmatrix} \end{array}$$

$$\begin{pmatrix} 0 & R \end{pmatrix} = \begin{array}{c} \\ k \\ l \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & R_{11} & R_{12} \\ 0 & 0 & R_{22} \end{pmatrix} \end{array}$$

where

C = diag(ALPHA(k+1), ... , ALPHA(k+l)),
S = diag(BETA(k+1), ... , BETA(k+l)),
$C^2 + S^2 = I$;

R is stored in A(1:k+l, n−k−l+1:n) on exit.

If $m-k-l < 0$,

$$D_1 = \begin{array}{c} \\ k \\ m-k \end{array} \begin{array}{c} \begin{array}{ccc} k & m-k & k+l-m \end{array} \\ \begin{pmatrix} I & 0 & 0 \\ 0 & C & 0 \end{pmatrix} \end{array}$$

$$D_2 = \begin{array}{c} \\ m-k \\ k+l-m \\ p-l \end{array} \begin{array}{c} \begin{array}{ccc} k & m-k & k+l-m \end{array} \\ \begin{pmatrix} 0 & S & 0 \\ 0 & 0 & I \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

$$\begin{pmatrix} 0 & R \end{pmatrix} = \begin{array}{c} \\ k \\ m-k \\ k+l-m \end{array} \begin{array}{c} \begin{array}{cccc} n-k-l & k & m-k & k+l-m \end{array} \\ \begin{pmatrix} 0 & R_{11} & R_{12} & R_{13} \\ 0 & 0 & R_{22} & R_{23} \\ 0 & 0 & 0 & R_{33} \end{pmatrix} \end{array}$$

where

C = diag(ALPHA(k+1), ... , ALPHA(m)),
S = diag(BETA(k+1), ... , BETA(m)),
$C^2 + S^2 = I$;

$\begin{pmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \end{pmatrix}$ is stored in A(1:m, n−k−l+1:n), and R_{33} is stored in B(m−k+1:l, n+m−k−l+1:n) on exit.

The computation of the orthogonal/unitary transformation matrices U, V or Q is optional. The matrices may either be formed explicitly, or they may be postmultiplied into input matrices U_1, V_1 or Q_1.

Arguments

JOBU (input) CHARACTER*1
- = 'U': U must contain a orthogonal/unitary matrix U_1 on entry, and the product U_1*U is returned;
- = 'I': U is initialized to the unit matrix, and the orthogonal/unitary matrix U is returned;
- = 'N': U is not computed.

JOBV (input) CHARACTER*1
- = 'V': V must contain a orthogonal/unitary matrix V_1 on entry, and the product V_1*V is returned;
- = 'I': V is initialized to the unit matrix, and the orthogonal/unitary matrix V is returned;
- = 'N': V is not computed.

JOBQ (input) CHARACTER*1
- = 'Q': Q must contain a orthogonal/unitary matrix Q_1 on entry, and the product Q_1*Q is returned;
- = 'I': Q is initialized to the unit matrix, and the orthogonal/unitary matrix Q is returned;
- = 'N': Q is not computed.

M (input) INTEGER
The number of rows of the matrix A. M $\geq$ 0.

P (input) INTEGER
The number of rows of the matrix B. P $\geq$ 0.

N (input) INTEGER
The number of columns of the matrices A and B. N $\geq$ 0.

K, L (input) INTEGER
K and L specify the subblocks in the input matrices A and B:
A_{23} = A(K+1:min(K+L,M),N−L+1:N) and B_{13} = B(1:L,N−L+1:N) of A and B, whose GSVD is going to be computed by STGSJA/CTGSJA.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the m-by-n matrix A.
On exit, A(N−K+1:N,1:min(K+L,M)) contains the triangular matrix R or part of R.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

B (input/output) REAL/COMPLEX array, dimension (LDB,N)
On entry, the p-by-n matrix B.

On exit, if necessary, B(M−K+1:L,N+M−K-L+1:N) contains a part of R.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,P).

TOLA (input) REAL

TOLB (input) REAL
TOLA and TOLB are the convergence criteria for the Jacobi-Kogbetliantz iteration procedure. Generally, they are the same as used in the preprocessing step, say
TOLA = max(m,n)*$\|A\|$*MACHEPS,
TOLB = max(p,n)*$\|B\|$*MACHEPS.

ALPHA (output) REAL array, dimension (N)

BETA (output) REAL array, dimension (N)
On exit, ALPHA and BETA contain the generalized singular value pairs of A and B;
ALPHA(1:K)=1,
BETA(1:K)=0,
and if M−K−L $\geq$ 0,
ALPHA(K+1:K+L)=diag(C),
BETA(K+1:K+L)=diag(S),
or if M−K−L < 0,
ALPHA(K+1:M)=C, ALPHA(M+1:K+L)=0,
BETA(K+1:M)=S, BETA(M+1:K+L)=1.
Furthermore, if K+L < N,
ALPHA(K+L+1:N) = 0 and BETA(K+L+1:N) = 0.

U (input/output) REAL/COMPLEX array, dimension (LDU,M)
On entry, if JOBU = 'U', U must contain a matrix U_1 (usually the orthogonal/unitary matrix returned by SGGSVP/CGGSVP).
On exit,
if JOBU = 'I', U contains the orthogonal/unitary matrix U;
if JOBU = 'U', U contains the product U_1*U.
If JOBU = 'N': U is not referenced.

LDU (input) INTEGER
The leading dimension of the array U.
LDU $\geq$ max(1,M) if JOBU = 'U'; LDU $\geq$ 1 otherwise.

V (input/output) REAL/COMPLEX array, dimension (LDV,P)
On entry, if JOBV = 'V', V must contain a matrix V_1 (usually the orthogonal/unitary matrix returned by SGGSVP/CGGSVP).
On exit,
if JOBV = 'I', V contains the orthogonal/unitary matrix V;
if JOBV = 'V', V contains the product V_1*V.
If JOBV = 'N': V is not referenced.

LDV (input) INTEGER
The leading dimension of the array V.
LDV $\geq$ max(1,P) if JOBV = 'V'; LDV $\geq$ 1 otherwise.

Q (input/output) REAL/COMPLEX array, dimension (LDQ,N)
On entry, if JOBQ = 'Q', Q must contain a matrix Q_1 (usually the orthogonal/unitary matrix returned by SGGSVP/CGGSVP).
On exit,
if JOBQ = 'I', Q contains the orthogonal/unitary matrix Q;
if JOBQ = 'Q', Q contains the product Q_1*Q.
If JOBQ = 'N': Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q.
LDQ $\geq$ max(1,N) if JOBQ = 'Q'; LDQ $\geq$ 1 otherwise.

WORK (workspace) REAL/COMPLEX array, dimension (2*N)

NCYCLE (output) INTEGER
The number of cycles required for convergence.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
= 1: the procedure does not converge after MAXIT cycles.

STGSNA/CTGSNA

```
      SUBROUTINE STGSNA( JOB, HOWMNY, SELECT, N, A, LDA, B, LDB, VL,
     $                   LDVL, VR, LDVR, S, DIF, MM, M, WORK, LWORK,
     $                   IWORK, INFO )
      CHARACTER          HOWMNY, JOB
      INTEGER            INFO, LDA, LDB, LDVL, LDVR, LWORK, M, MM, N
      LOGICAL            SELECT( * )
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), B( LDB, * ), DIF( * ), S( * ),
     $                   VL( LDVL, * ), VR( LDVR, * ), WORK( * )

      SUBROUTINE CTGSNA( JOB, HOWMNY, SELECT, N, A, LDA, B, LDB, VL,
     $                   LDVL, VR, LDVR, S, DIF, MM, M, WORK, LWORK,
     $                   IWORK, INFO )
      CHARACTER          HOWMNY, JOB
      INTEGER            INFO, LDA, LDB, LDVL, LDVR, LWORK, M, MM, N
      LOGICAL            SELECT( * )
      INTEGER            IWORK( * )
      REAL               DIF( * ), S( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), VL( LDVL, * ),
     $                   VR( LDVR, * ), WORK( * )
```

Purpose

STGSNA estimates reciprocal condition numbers for specified eigenvalues and/or eigenvectors of a matrix pair (A, B) in generalized real Schur canonical form (or of any matrix pair (Q*A*Z^T, Q*B*Z^T) with orthogonal matrices Q and Z.

(A,B) must be in generalized real Schur form (as returned by SGGES), i.e. A is block upper triangular with 1-by-1 and 2-by-2 diagonal blocks. B is upper triangular.

CTGSNA estimates reciprocal condition numbers for specified eigenvalues and/or eigenvectors of a matrix pair (A, B).

(A,B) must be in generalized Schur canonical form, that is, A and B are both upper triangular.

Arguments

JOB (input) CHARACTER*1
Specifies whether condition numbers are required for eigenvalues (S) or eigenvectors (DIF):
= 'E': for eigenvalues only (S);
= 'V': for eigenvectors only (DIF);
= 'B': for both eigenvalues and eigenvectors (S and DIF).

HOWMNY (input) CHARACTER*1
= 'A': compute condition numbers for all eigenpairs;
= 'S': compute condition numbers for selected eigenpairs specified by the array SELECT.

SELECT (input) LOGICAL array, dimension (N)
STGSNA
If HOWMNY = 'S', SELECT specifies the eigenpairs for which condition numbers are required. To select condition numbers for the eigenpair corresponding to a real eigenvalue w(j), SELECT(j) must be set to .TRUE.. To select condition numbers corresponding to a complex conjugate pair of eigenvalues w(j) and w(j+1), either SELECT(j) or SELECT(j+1) or both, must be set to .TRUE..
If HOWMNY = 'A', SELECT is not referenced.
CTGSNA
If HOWMNY = 'S', SELECT specifies the eigenpairs for which condition numbers are required. To select condition numbers for the corresponding j-th eigenvalue and/or eigenvector, SELECT(j) must be set to .TRUE..
If HOWMNY = 'A', SELECT is not referenced.

N (input) INTEGER
The order of the square matrix pair (A, B). N $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The upper quasi-triangular/triangular matrix A in the pair (A,B).

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

B (input) REAL/COMPLEX array, dimension (LDB,N)
The upper triangular matrix B in the pair (A,B).

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

VL (input) REAL/COMPLEX array, dimension (LDVL,M)
IF JOB = 'E' or 'B', VL must contain left eigenvectors of (A, B), corresponding to the eigenpairs specified by HOWMNY and SELECT. The eigenvectors must be stored in consecutive columns of VL, as returned by STGEVC/CTGEVC. If JOB = 'V', VL is not referenced.

LDVL (input) INTEGER
The leading dimension of the array VL. LDVL $\geq$ 1; and If JOB = 'E' or 'B', LDVL $\geq$ N.

VR (input) REAL/COMPLEX array, dimension (LDVR,M)
IF JOB = 'E' or 'B', VR must contain right eigenvectors of (A, B), corresponding to the eigenpairs specified by HOWMNY and SELECT. The eigenvectors must be stored in consecutive columns of VR, as returned by STGEVC/CTGEVC. If JOB = 'V', VR is not referenced.

LDVR (input) INTEGER
The leading dimension of the array VR. LDVR $\geq$ 1; If JOB = 'E' or 'B', LDVR $\geq$ N.

S (output) REAL array, dimension (MM)
STGSNA
If JOB = 'E' or 'B', the reciprocal condition numbers of the selected eigenvalues, stored in consecutive elements of the array. For a complex conjugate pair of eigenvalues two consecutive elements of S are set to the same value. Thus S(j), DIF(j), and the j^{th} columns of VL and VR all correspond to the same eigenpair (but not in general the j^{th} eigenpair, unless all eigenpairs are selected). If JOB = 'V', S is not referenced.
CTGSNA
If JOB = 'E' or 'B', the reciprocal condition numbers of the selected eigenvalues, stored in consecutive elements of the array.
If JOB = 'V', S is not referenced.

DIF (output) REAL array, dimension (MM)
STGSNA
If JOB = 'V' or 'B', the estimated reciprocal condition numbers of the selected eigenvectors, stored in consecutive elements of the array. For a complex eigenvector two consecutive elements of DIF are set to the same value. If the eigenvalues cannot be reordered to compute DIF(j), DIF(j) is set to 0; this can only occur when the true value would be very small anyway.
If JOB = 'E', DIF is not referenced.
CTGSNA
If JOB = 'V' or 'B', the estimated reciprocal condition numbers of the selected eigenvectors, stored in consecutive elements of the array. If the eigenvalues cannot be reordered to compute DIF(j), DIF(j) is set to 0; this can only occur when the true value would be very small anyway. For each eigenvalue/vector specified by SELECT, DIF stores a Frobenius norm-based estimate of Difl.
If JOB = 'E', DIF is not referenced.

MM (input) INTEGER
The number of elements in the arrays S and DIF. MM $\geq$ M.

M (output) INTEGER
The number of elements of the arrays S and DIF used to store the specified condition numbers; for each selected eigenvalue one element is used. If HOWMNY = 'A', M is set to N.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
If JOB = 'E', WORK is not referenced. Otherwise, on exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
STGSNA
The dimension of the array WORK. LWORK $\geq$ N.
If JOB = 'V' or 'B' LWORK $\geq$ 2*N*(N+2)+16.
CTGSNA
The dimension of the array WORK. LWORK $\geq$ 1.
If JOB = 'V' or 'B', LWORK $\geq$ 2*N*N.

IWORK *STGSNA* (workspace) INTEGER array, dimension (N+6)
CTGSNA (workspace) INTEGER array, dimension (N+2)
IF JOB = 'E', IWORK is not referenced.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

STGSYL/CTGSYL

```
      SUBROUTINE STGSYL( TRANS, IJOB, M, N, A, LDA, B, LDB, C, LDC, D,
     $                   LDD, E, LDE, F, LDF, SCALE, DIF, WORK, LWORK,
     $                   IWORK, INFO )
      CHARACTER          TRANS
      INTEGER            IJOB, INFO, LDA, LDB, LDC, LDD, LDE, LDF,
     $                   LWORK, M, N
      REAL               DIF, SCALE
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), B( LDB, * ), C( LDC, * ),
     $                   D( LDD, * ), E( LDE, * ), F( LDF, * ),
     $                   WORK( * )
```

```
      SUBROUTINE CTGSYL( TRANS, IJOB, M, N, A, LDA, B, LDB, C, LDC, D,
     $                   LDD, E, LDE, F, LDF, SCALE, DIF, WORK, LWORK,
     $                   IWORK, INFO )
      CHARACTER          TRANS
      INTEGER            IJOB, INFO, LDA, LDB, LDC, LDD, LDE, LDF,
     $                   LWORK, M, N
      REAL               DIF, SCALE
      INTEGER            IWORK( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), C( LDC, * ),
     $                   D( LDD, * ), E( LDE, * ), F( LDF, * ),
     $                   WORK( * )
```

Purpose

STGSYL/CTGSYL solves the generalized Sylvester equation:

$$\begin{aligned} A * R - L * B &= scale * C \\ D * R - L * E &= scale * F \end{aligned} \qquad (1)$$

where R and L are unknown m-by-n matrices, (A, D), (B, E) and (C, F) are given matrix pairs of size m-by-m, n-by-n and m-by-n, respectively, with real/complex entries. (A, D) and (B, E) must be in generalized real-Schur/Schur canonical form, i.e. A, B are upper quasi-triangular/triangular and D, E are upper triangular.

The solution (R, L) overwrites (C, F). $0 \leq$ SCALE ≤ 1 is an output scaling factor chosen to avoid overflow.

In matrix notation (1) is equivalent to solve Z * x = scale * b, where Z is defined as

$$Z = \begin{pmatrix} kron(In, A) & -kron(B', Im) \\ kron(In, D) & -kron(E', Im) \end{pmatrix} \qquad (2)$$

Here Ik is the identity matrix of size k and X' is the transpose/conjugate-transpose of X. kron(X, Y) is the Kronecker product between the matrices X and Y.

If TRANS = 'T' (*STGSYL*) or TRANS = 'C' (*CTGSYL*), STGSYL/CTGSYL solves the transposed/conjugate-transposed system Z' * y = scale * b, which is equivalent to solve for R and L in

$$\begin{aligned} A' * R + D' * L &= scale * C \\ R * B' + L * E' &= scale * (-F) \end{aligned} \qquad (3)$$

This case (TRANS = 'T' for STGSYL or TRANS = 'C' for CTGSYL) is used to compute an one-norm-based estimate of Dif[(A,D), (B,E)], the separation between the matrix pairs (A,D) and (B,E), using SLACON/CLACON.

If IJOB $\geq$ 1, STGSYL/CTGSYL computes a Frobenius norm-based estimate of Dif[(A,D),(B,E)]. That is, the reciprocal of a lower bound on the reciprocal of the smallest singular value of Z. See [1-2] for more information.

This is a level 3 BLAS algorithm.

Arguments

TRANS (input) CHARACTER*1
= 'N': solve the generalized Sylvester equation (1).
= 'T': solve the 'transposed' system (3). (*STGSYL only*)
= 'C': solve the "conjugate transposed" system (3). (*CTGSYL only*)

IJOB (input) INTEGER
Specifies what kind of functionality to be performed.
= 0: solve (1) only.
= 1: The functionality of 0 and 3.
= 2: The functionality of 0 and 4.
= 3: Only an estimate of Dif[(A,D), (B,E)] is computed. (look ahead strategy is used).
= 4: Only an estimate of Dif[(A,D), (B,E)] is computed. (SGECON/CGECON on sub-systems is used).
Not referenced if TRANS = 'T' (STGSYL) or TRANS = 'C' (CTGSYL).

M (input) INTEGER
The order of the matrices A and D, and the row dimension of the matrices C, F, R and L.

N (input) INTEGER
The order of the matrices B and E, and the column dimension of the matrices C, F, R and L.

A (input) REAL/COMPLEX array, dimension (LDA,M)
The upper quasi-triangular/triangular matrix A.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

B (input) REAL/COMPLEX array, dimension (LDB,N)
The upper quasi-triangular/triangular matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, C contains the right-hand-side of the first matrix equation in (1) or (3).
On exit, if IJOB = 0, 1 or 2, C has been overwritten by the solution R. If IJOB = 3 or 4 and TRANS = 'N', C holds R, the solution achieved during the computation of the Dif-estimate.

LDC (input) INTEGER
The leading dimension of the array C. LDC $\geq$ max(1,M).

D (input) REAL/COMPLEX array, dimension (LDD,M)
The upper triangular matrix D.

LDD (input) INTEGER
The leading dimension of the array D. LDD $\geq$ max(1, M).

E (input) REAL/COMPLEX array, dimension (LDE,N)
The upper triangular matrix E.

LDE (input) INTEGER
The leading dimension of the array E. LDE $\geq$ max(1, N).

F (input/output) REAL/COMPLEX array, dimension (LDF,N)
On entry, F contains the right-hand-side of the second matrix equation in (1) or (3).
On exit, if IJOB = 0, 1 or 2, F has been overwritten by the solution L. If IJOB = 3 or 4 and TRANS = 'N', F holds L, the solution achieved during the computation of the Dif-estimate.

LDF (input) INTEGER
The leading dimension of the array F. LDF $\geq$ max(1, M).

DIF (output) REAL
On exit DIF is the reciprocal of a lower bound of the reciprocal of the Dif-function, i.e. DIF is an upper bound of Dif[(A,D), (B,E)] = sigma_min(Z), where Z as in (2).
IF IJOB = 0 or TRANS = 'T' (STGSYL) or TRANS = 'C' (CTGSYL), DIF is not touched.

SCALE (output) REAL
On exit SCALE is the scaling factor in (1) or (3). If 0 < SCALE < 1, C and F hold the solutions R and L, resp., to a slightly perturbed system but the input matrices A, B, D and E have not been changed. If SCALE = 0, C and F hold the solutions R and L, respectively, to the homogeneous system with C = F = 0. Normally, SCALE = 1.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
If IJOB = 0, WORK is not referenced. Otherwise, on exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK $\geq$ 1.
If IJOB = 1 or 2 and TRANS = 'N', LWORK $\geq$ 2*M*N.

IWORK *STGSYL* (workspace) INTEGER array, dimension (M+N+6)
CTGSYL (workspace) INTEGER array, dimension (M+N+2)
If IJOB = 0, IWORK is not referenced.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: (A, D) and (B, E) have common or close eigenvalues.

STPCON/CTPCON

```
      SUBROUTINE STPCON( NORM, UPLO, DIAG, N, AP, RCOND, WORK, IWORK,
     $                   INFO )
      CHARACTER          DIAG, NORM, UPLO
      INTEGER            INFO, N
      REAL               RCOND
      INTEGER            IWORK( * )
      REAL               AP( * ), WORK( * )

      SUBROUTINE CTPCON( NORM, UPLO, DIAG, N, AP, RCOND, WORK, RWORK,
     $                   INFO )
      CHARACTER          DIAG, NORM, UPLO
      INTEGER            INFO, N
      REAL               RCOND
      REAL               RWORK( * )
      COMPLEX            AP( * ), WORK( * )
```

Purpose

STPCON/CTPCON estimates the reciprocal of the condition number of a packed triangular matrix A, in either the 1-norm or the infinity-norm.

The norm of A is computed and an estimate is obtained for $\|A^{-1}\|$, then the reciprocal of the condition number is computed as RCOND $= 1/(\|A\| * \|A^{-1}\|)$.

Arguments

NORM (input) CHARACTER*1
Specifies whether the 1-norm condition number or the infinity-norm condition number is required:
= '1' or 'O': 1-norm;
= 'I': Infinity-norm.

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

AP (input) REAL/COMPLEX array, dimension (N*(N+1)/2)
The upper or lower triangular matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.
If DIAG = 'U', the diagonal elements of A are not referenced and are assumed to be 1.

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as RCOND $= 1/(\|A\| * \|A^{-1}\|)$.

WORK *STPCON* (workspace) REAL array, dimension (3*N)
CTPCON (workspace) COMPLEX array, dimension (2*N)

IWORK *STPCON only* (workspace) INTEGER array, dimension (N)

RWORK *CTPCON only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

STPRFS/CTPRFS

```
      SUBROUTINE STPRFS( UPLO, TRANS, DIAG, N, NRHS, AP, B, LDB, X, LDX,
     $                   FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      INTEGER            IWORK( * )
      REAL               AP( * ), B( LDB, * ), BERR( * ), FERR( * ),
     $                   WORK( * ), X( LDX, * )

      SUBROUTINE CTPRFS( UPLO, TRANS, DIAG, N, NRHS, AP, B, LDB, X, LDX,
     $                   FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, LDB, LDX, N, NRHS
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            AP( * ), B( LDB, * ), WORK( * ), X( LDX, * )
```

Purpose

STPRFS/CTPRFS provides error bounds and backward error estimates for the solution to a system of linear equations with a triangular packed coefficient matrix.

The solution matrix X must be computed by STPTRS/CTPTRS or some other means before entering this routine. STPRFS/CTPRFS does not do iterative refinement because doing so cannot improve the backward error.

Arguments

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)

= 'C': $A^H*X = B$ (Conjugate transpose)

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

AP (input) REAL/COMPLEX array, dimension (N*(N+1)/2)
The upper or lower triangular matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.
If DIAG = 'U', the diagonal elements of A are not referenced and are assumed to be 1.

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (input) REAL/COMPLEX array, dimension (LDX,NRHS)
The solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *STPRFS* (workspace) REAL array, dimension (3*N)
CTPRFS (workspace) COMPLEX array, dimension (2*N)

IWORK *STPRFS only* (workspace) INTEGER array, dimension (N)

RWORK *CTPRFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

STPTRI/CTPTRI

```
SUBROUTINE STPTRI( UPLO, DIAG, N, AP, INFO )
CHARACTER          DIAG, UPLO
INTEGER            INFO, N
REAL               AP( * )

SUBROUTINE CTPTRI( UPLO, DIAG, N, AP, INFO )
CHARACTER          DIAG, UPLO
INTEGER            INFO, N
COMPLEX            AP( * )
```

Purpose

STPTRI/CTPTRI computes the inverse of a real/complex upper or lower triangular matrix A stored in packed format.

Arguments

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

AP (input/output) REAL/COMPLEX array, dimension (N*(N+1)/2)
On entry, the upper or lower triangular matrix A, stored columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', AP(i + (j−1)*j/2) = A(i,j) for $1 \leq i \leq j$;
if UPLO = 'L', AP(i + (j−1)*(2*n−j)/2) = A(i,j) for $j \leq i \leq n$.
On exit, the (triangular) inverse of the original matrix, in the same packed storage format.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, A(i,i) is exactly zero. The triangular matrix is singular and its inverse can not be computed.

STPTRS/CTPTRS

```
      SUBROUTINE STPTRS( UPLO, TRANS, DIAG, N, NRHS, AP, B, LDB, INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, LDB, N, NRHS
      REAL               AP( * ), B( LDB, * )

      SUBROUTINE CTPTRS( UPLO, TRANS, DIAG, N, NRHS, AP, B, LDB, INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, LDB, N, NRHS
      COMPLEX            AP( * ), B( LDB, * )
```

Purpose

STPTRS/CTPTRS solves a triangular system of the form $A*X = B$, $A^T*X = B$, or $A^H*X = B$, where A is a triangular matrix of order n stored in packed format, and B is an n-by-nrhs matrix. A check is made to verify that A is nonsingular.

Arguments

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': $A*X = B$ (No transpose)
= 'T': $A^T*X = B$ (Transpose)
= 'C': $A^H*X = B$ (Conjugate transpose)

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. $NRHS \geq 0$.

AP (input) REAL/COMPLEX array, dimension (N*(N+1)/2)
The upper or lower triangular matrix A, packed columnwise in a linear array. The j^{th} column of A is stored in the array AP as follows:
if UPLO = 'U', $AP(i + (j-1)*j/2) = A(i,j)$ for $1 \leq i \leq j$;
if UPLO = 'L', $AP(i + (j-1)*(2*n-j)/2) = A(i,j)$ for $j \leq i \leq n$.

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, if INFO = 0, the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. $LDB \geq \max(1,N)$.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the i^{th} diagonal element of A is zero, indicating that the matrix is singular and the solutions X have not been computed.

STRCON/CTRCON

```
      SUBROUTINE STRCON( NORM, UPLO, DIAG, N, A, LDA, RCOND, WORK,
     $                   IWORK, INFO )
      CHARACTER          DIAG, NORM, UPLO
      INTEGER            INFO, LDA, N
      REAL               RCOND
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), WORK( * )

      SUBROUTINE CTRCON( NORM, UPLO, DIAG, N, A, LDA, RCOND, WORK,
     $                   RWORK, INFO )
      CHARACTER          DIAG, NORM, UPLO
      INTEGER            INFO, LDA, N
      REAL               RCOND
      REAL               RWORK( * )
      COMPLEX            A( LDA, * ), WORK( * )
```

Purpose

STRCON/CTRCON estimates the reciprocal of the condition number of a triangular matrix A, in either the 1-norm or the infinity-norm.

The norm of A is computed and an estimate is obtained for $\|A^{-1}\|$, then the reciprocal of the condition number is computed as $RCOND = 1/(\|A\| * \|A^{-1}\|)$.

Arguments

NORM (input) CHARACTER*1
Specifies whether the 1-norm condition number or the infinity-norm condition number is required:
= '1' or 'O': 1-norm;
= 'I': Infinity-norm.

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. $N \geq 0$.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The triangular matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of the array A contains the upper triangular matrix, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of the array A contains the lower triangular matrix, and the strictly upper triangular part of A is not referenced. If DIAG = 'U', the diagonal elements of A are also not referenced and are assumed to be 1.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

RCOND (output) REAL
The reciprocal of the condition number of the matrix A, computed as RCOND = $1/(\|A\| * \|A^{-1}\|)$.

WORK *STRCON* (workspace) REAL array, dimension (3*N)
CTRCON (workspace) COMPLEX array, dimension (2*N)

IWORK *STRCON only* (workspace) INTEGER array, dimension (N)

RWORK *CTRCON only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

STREVC/CTREVC

```
      SUBROUTINE STREVC( SIDE, HOWMNY, SELECT, N, T, LDT, VL, LDVL, VR,
     $                   LDVR, MM, M, WORK, INFO )
      CHARACTER          HOWMNY, SIDE
      INTEGER            INFO, LDT, LDVL, LDVR, M, MM, N
      LOGICAL            SELECT( * )
      REAL               T( LDT, * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   WORK( * )

      SUBROUTINE CTREVC( SIDE, HOWMNY, SELECT, N, T, LDT, VL, LDVL, VR,
     $                   LDVR, MM, M, WORK, RWORK, INFO )
      CHARACTER          HOWMNY, SIDE
      INTEGER            INFO, LDT, LDVL, LDVR, M, MM, N
      LOGICAL            SELECT( * )
      REAL               RWORK( * )
      COMPLEX            T( LDT, * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   WORK( * )
```

Purpose

STREVC/CTREVC computes some or all of the right and/or left eigenvectors of a real/complex upper quasi-triangular/triangular matrix T.

The right eigenvector x and the left eigenvector y of T corresponding to an eigenvalue w are defined by:

$$T*x = w*x \quad \text{and} \quad y^H*T = w*y^H.$$

If all eigenvectors are requested, the routine may either return the matrices X and/or Y of right or left eigenvectors of T, or the products Q*X and/or Q*Y, where Q is an input orthogonal/unitary matrix. If T was obtained from the real-Schur/Schur factorization of an original matrix A = $Q*T*Q^H$, then Q*X and Q*Y are the matrices of right or left eigenvectors of A.

STREVC only
T must be in Schur canonical form (as returned by SHSEQR), that is, block upper triangular with 1-by-1 and 2-by-2 diagonal blocks; each 2-by-2 diagonal block has its diagonal elements equal and its off-diagonal elements of opposite sign. Corresponding to each 2-by-2 diagonal block is a complex conjugate pair of eigenvalues and eigenvectors; only one eigenvector of the pair is computed, namely the one corresponding to the eigenvalue with positive imaginary part.

Arguments

SIDE (input) CHARACTER*1
= 'R': compute right eigenvectors only;
= 'L': compute left eigenvectors only;
= 'B': compute both right and left eigenvectors.

HOWMNY (input) CHARACTER*1
= 'A': compute all right and/or left eigenvectors;
= 'B': compute all right and/or left eigenvectors, and backtransform them using the input matrices supplied in VR and/or VL;
= 'S': compute selected right and/or left eigenvectors, specified by the logical array SELECT.

SELECT *STREVC* (input/output) LOGICAL array, dimension (N)
CTREVC (input) LOGICAL array, dimension (N)
If HOWMNY = 'S', SELECT specifies the eigenvectors to be computed.
If HOWMNY = 'A' or 'B', SELECT is not referenced.
STREVC
To select the real eigenvector corresponding to a real eigenvalue w(j), SELECT(j) must be set to .TRUE.. To select the complex eigenvector corresponding to a complex conjugate pair w(j) and w(j+1), either SELECT(j) or SELECT(j+1) must be set to .TRUE.; then on exit SELECT(j) is .TRUE. and SELECT(j+1) is .FALSE..
CTREVC
To select the eigenvector corresponding to the j^{th} eigenvalue, SELECT(j) must be set to .TRUE..

N (input) INTEGER
The order of the matrix T. N $\geq$ 0.

T (input) REAL array, dimension (LDT,N)
STREVC

The upper quasi-triangular matrix T in Schur canonical form.
CTREVC
The upper triangular matrix T. T is modified by the routine, but restored on exit.

LDT (input) INTEGER
The leading dimension of the array T. LDT $\geq$ max(1,N).

VL (input/output) REAL/COMPLEX array, dimension (LDVL,MM)
On entry, if SIDE = 'L' or 'B' and HOWMNY = 'B', VL must contain an n-by-n matrix Q (usually the orthogonal/unitary matrix Q of Schur vectors returned by SHSEQR/CHSEQR).
On exit, if SIDE = 'L' or 'B', VL contains:
if HOWMNY = 'A', the matrix Y of left eigenvectors of T;
STREVC
VL has the same quasi-lower triangular form as T'. If T(i,i) is a real eigenvalue, then the i^{th} column VL(i) of VL is its corresponding eigenvector. If T(i:i+1,i:i+1) is a 2-by-2 block whose eigenvalues are complex-conjugate eigenvalues of T, then VL(i)+$\sqrt{-1}$*VL(i+1) is the complex eigenvector corresponding to the eigenvalue with positive real part.
CTREVC
VL is lower triangular. The i^{th} column VL(i) of VL is the eigenvector corresponding to T(i,i).
if HOWMNY = 'B', the matrix Q*Y;
if HOWMNY = 'S', the left eigenvectors of T specified by SELECT, stored consecutively in the columns of VL, in the same order as their eigenvalues.
If SIDE = 'R', VL is not referenced.
STREVC only
A complex eigenvector corresponding to a complex eigenvalue is stored in two consecutive columns, the first holding the real part, and the second the imaginary part.

LDVL (input) INTEGER
The leading dimension of the array VL.
LDVL $\geq$ max(1,N) if SIDE = 'L' or 'B'; LDVL $\geq$ 1 otherwise.

VR (input/output) REAL/COMPLEX array, dimension (LDVR,MM)
On entry, if SIDE = 'R' or 'B' and HOWMNY = 'B', VR must contain an n-by-n matrix Q (usually the orthogonal/unitary matrix Q of Schur vectors returned by SHSEQR/CHSEQR).
On exit, if SIDE = 'R' or 'B', VR contains:
if HOWMNY = 'A', the matrix X of right eigenvectors of T;
STREVC
VR has the same quasi-upper triangular form as T. If T(i,i) is a real eigenvalue, then the i^{th} column VR(i) of VR is its corresponding eigenvector. If T(i:i+1,i:i+1) is a 2-by-2 block whose eigenvalues are complex-conjugate eigenvalues of T, then VR(i)+$\sqrt{-1}$*VR(i+1) is the complex eigenvector corresponding to the eigenvalue with positive real part.
CTREVC
VR is upper triangular. The i^{th} column VR(i) of VR is the eigenvector corresponding to T(i,i).
if HOWMNY = 'B', the matrix Q*X;
if HOWMNY = 'S', the right eigenvectors of T specified by SELECT, stored consecutively in the columns of VR, in the same order as their eigenvalues.
If SIDE = 'L', VR is not referenced.
STREVC only
A complex eigenvector corresponding to a complex eigenvalue is stored in two consecutive columns, the first holding the real part and the second the imaginary part.

LDVR (input) INTEGER
The leading dimension of the array VR.
LDVR $\geq$ max(1,N) if SIDE = 'R' or 'B'; LDVR $\geq$ 1 otherwise.

MM (input) INTEGER
The number of columns in the arrays VL and/or VR. MM $\geq$ M.

M (output) INTEGER
The number of columns in the arrays VL and/or VR actually used to store the eigenvectors. If HOWMNY = 'A' or 'B', M is set to N.
STREVC
Each selected real eigenvector occupies one column and each selected complex eigenvector occupies two columns.
CTREVC
Each selected eigenvector occupies one column.

WORK *STREVC* (workspace) REAL array, dimension (3*N)
CTREVC (workspace) COMPLEX array, dimension (2*N)

RWORK *CTREVC only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

STREXC/CTREXC

```
      SUBROUTINE STREXC( COMPQ, N, T, LDT, Q, LDQ, IFST, ILST, WORK,
     $                   INFO )
      CHARACTER          COMPQ
      INTEGER            IFST, ILST, INFO, LDQ, LDT, N
      REAL               Q( LDQ, * ), T( LDT, * ), WORK( * )

      SUBROUTINE CTREXC( COMPQ, N, T, LDT, Q, LDQ, IFST, ILST, INFO )
      CHARACTER          COMPQ
      INTEGER            IFST, ILST, INFO, LDQ, LDT, N
      COMPLEX            Q( LDQ, * ), T( LDT, * )
```

Purpose

STREXC/CTREXC reorders the real-Schur/Schur factorization of a real/complex matrix $A = Q*T*Q^H$, so that the diagonal block/element of T with row index IFST is moved to row ILST.

The real-Schur/Schur form T is reordered by an orthogonal/unitary similarity transformation Z^H*T*Z, and optionally the matrix Q of Schur vectors is updated by postmultiplying it with Z.

STREXC only
T must be in Schur canonical form (as returned by SHSEQR), that is, block upper triangular with 1-by-1 and 2-by-2 diagonal blocks; each 2-by-2 diagonal block has its diagonal elements equal and its off-diagonal elements of opposite sign.

Arguments

COMPQ (input) CHARACTER*1
= 'V': update the matrix Q of Schur vectors;
= 'N': do not update Q.

N (input) INTEGER
The order of the matrix T. $N \geq 0$.

T (input/output) REAL/COMPLEX array, dimension (LDT,N)
STREXC
On entry, the upper quasi-triangular matrix T, in Schur canonical form.
On exit, the reordered upper quasi-triangular matrix, again in Schur canonical form.
CTREXC
On entry, the upper triangular matrix T.
On exit, the reordered upper triangular matrix.

LDT (input) INTEGER
The leading dimension of the array T. LDT $\geq$ max(1,N).

Q (input/output) REAL/COMPLEX array, dimension (LDQ,N)
On entry, if COMPQ = 'V', the matrix Q of Schur vectors.
On exit, if COMPQ = 'V', Q has been postmultiplied by the orthogonal/unitary transformation matrix Z which reorders T.
If COMPQ = 'N', Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q. LDQ $\geq$ max(1,N).

IFST *STREXC* (input/output) INTEGER
CTREXC (input) INTEGER

ILST *STREXC* (input/output) INTEGER
CTREXC (input) INTEGER
Specify the reordering of the diagonal blocks/elements of T. The block/element with row index IFST is moved to row ILST by a sequence of transpositions between adjacent blocks/elements.
$1 \leq IFST \leq N$; $1 \leq ILST \leq N$.

STREXC only
On exit, if IFST pointed on entry to the second row of a 2-by-2 block, it is changed to point to the first row; ILST always points to the first row of the block in its final position (which may differ from its input value by ± 1).

WORK *STREXC only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = $-i$, the i^{th} argument had an illegal value.

STRRFS/CTRRFS

```
      SUBROUTINE STRRFS( UPLO, TRANS, DIAG, N, NRHS, A, LDA, B, LDB, X,
     $                   LDX, FERR, BERR, WORK, IWORK, INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, LDA, LDB, LDX, N, NRHS
      INTEGER            IWORK( * )
      REAL               A( LDA, * ), B( LDB, * ), BERR( * ), FERR( * ),
     $                   WORK( * ), X( LDX, * )

      SUBROUTINE CTRRFS( UPLO, TRANS, DIAG, N, NRHS, A, LDA, B, LDB, X,
     $                   LDX, FERR, BERR, WORK, RWORK, INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, LDA, LDB, LDX, N, NRHS
      REAL               BERR( * ), FERR( * ), RWORK( * )
      COMPLEX            A( LDA, * ), B( LDB, * ), WORK( * ),
     $                   X( LDX, * )
```

Purpose

STRRFS/CTRRFS provides error bounds and backward error estimates for the solution to a system of linear equations with a triangular coefficient matrix.

The solution matrix X must be computed by STRTRS/CTRTRS or some other means before entering this routine. STRRFS/CTRRFS does not do iterative refinement because doing so cannot improve the backward error.

Arguments

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': $A*X = B$ (No transpose)
= 'T': $A^T*X = B$ (Transpose)
= 'C': $A^H*X = B$ (Conjugate transpose)

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrices B and X. NRHS $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The triangular matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of the array A contains the upper triangular matrix, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of the array A contains the lower triangular matrix, and the strictly upper triangular part of A is not referenced. If DIAG = 'U', the diagonal elements of A are also not referenced and are assumed to be 1.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

B (input) REAL/COMPLEX array, dimension (LDB,NRHS)
The right hand side matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

X (input) REAL/COMPLEX array, dimension (LDX,NRHS)
The solution matrix X.

LDX (input) INTEGER
The leading dimension of the array X. LDX $\geq$ max(1,N).

FERR (output) REAL array, dimension (NRHS)
The estimated forward error bound for each solution vector X(j) (the j^{th} column of the solution matrix X). If XTRUE is the true solution corresponding to X(j), FERR(j) is an estimated upper bound for the magnitude of the largest element in (X(j) − XTRUE) divided by the magnitude of the largest element in X(j). The estimate is as reliable as the estimate for RCOND, and is almost always a slight overestimate of the true error.

BERR (output) REAL array, dimension (NRHS)
The componentwise relative backward error of each solution vector X(j) (i.e., the smallest relative change in any element of A or B that makes X(j) an exact solution).

WORK *STRRFS* (workspace) REAL array, dimension (3*N)
CTRRFS (workspace) COMPLEX array, dimension (2*N)

IWORK *STRRFS only* (workspace) INTEGER array, dimension (N)

RWORK *CTRRFS only* (workspace) REAL array, dimension (N)

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

STRSEN/CTRSEN

```
      SUBROUTINE STRSEN( JOB, COMPQ, SELECT, N, T, LDT, Q, LDQ, WR, WI,
     $                   M, S, SEP, WORK, LWORK, IWORK, LIWORK, INFO )
      CHARACTER          COMPQ, JOB
      INTEGER            INFO, LDQ, LDT, LIWORK, LWORK, M, N
      REAL               S, SEP
      LOGICAL            SELECT( * )
      INTEGER            IWORK( * )
      REAL               Q( LDQ, * ), T( LDT, * ), WI( * ), WORK( * ),
     $                   WR( * )

      SUBROUTINE CTRSEN( JOB, COMPQ, SELECT, N, T, LDT, Q, LDQ, W, M, S,
     $                   SEP, WORK, LWORK, INFO )
      CHARACTER          COMPQ, JOB
      INTEGER            INFO, LDQ, LDT, LWORK, M, N
      REAL               S, SEP
      LOGICAL            SELECT( * )
      COMPLEX            Q( LDQ, * ), T( LDT, * ), W( * ), WORK( * )
```

Purpose

STRSEN/CTRSEN reorders the real-Schur/Schur factorization of a real/complex matrix $A = Q*T*Q^H$, so that a selected cluster of eigenvalues appears in the leading blocks/elements on the diagonal of the upper quasi-triangular/triangular matrix T, and the leading columns of Q form a(n) orthonormal/unitary basis of the corresponding right invariant subspace.

Optionally, the routine computes the reciprocal condition numbers of the cluster of eigenvalues and/or the invariant subspace.

STRSEN only
T must be in Schur canonical form (as returned by SHSEQR), that is, block upper triangular with 1-by-1 and 2-by-2 diagonal blocks; each 2-by-2 diagonal block has its diagonal elements equal and its off-diagonal elements of opposite sign.

Arguments

JOB (input) CHARACTER*1
Specifies whether condition numbers are required for the cluster of eigenvalues (S) or the invariant subspace (SEP):
= 'N': none;
= 'E': for eigenvalues only (S);
= 'V': for invariant subspace only (SEP);
= 'B': for both eigenvalues and invariant subspace (S and SEP).

COMPQ (input) CHARACTER*1
= 'V': update the matrix Q of Schur vectors;
= 'N': do not update Q.

SELECT (input) LOGICAL array, dimension (N)
SELECT specifies the eigenvalues in the selected cluster.
STRSEN
To select a real eigenvalue w(j), SELECT(j) must be set to .TRUE.. To select a complex conjugate pair of eigenvalues w(j) and w(j+1), corresponding to a 2-by-2 diagonal block, either SELECT(j) or SELECT(j+1) or both must be set to .TRUE.; a complex conjugate pair of eigenvalues must be either both included in the cluster or both excluded.
CTRSEN
To select the j^{th} eigenvalue, SELECT(j) must be set to .TRUE..

N (input) INTEGER
The order of the matrix T. N $\geq$ 0.

T (input/output) REAL/COMPLEX array, dimension (LDT,N)
STRSEN
On entry, the upper quasi-triangular matrix T, in Schur canonical form. On exit, T is overwritten by the reordered matrix T, again in Schur canonical form, with the selected eigenvalues in the leading diagonal blocks.
CTRSEN
On entry, the upper triangular matrix T.
On exit, T is overwritten by the reordered matrix T, with the selected eigenvalues as the leading diagonal elements.

LDT (input) INTEGER
The leading dimension of the array T. LDT $\geq$ max(1,N).

Q (input/output) REAL/COMPLEX array, dimension (LDQ,N)
On entry, if COMPQ = 'V', the matrix Q of Schur vectors.
On exit, if COMPQ = 'V', Q has been postmultiplied by the orthogonal/unitary transformation matrix which reorders T; the leading M columns of Q form a(n) orthonormal/unitary basis for the specified invariant subspace.
If COMPQ = 'N', Q is not referenced.

LDQ (input) INTEGER
The leading dimension of the array Q.
LDQ $\geq$ 1; and if COMPQ = 'V', LDQ $\geq$ N.

WR, WI *STRSEN only* (output) REAL array, dimension (N)
The real and imaginary parts, respectively, of the reordered eigenvalues of T. The eigenvalues are stored in the same order as on the diagonal of T, with WR(i) = T(i,i) and, if T(i:i+1,i:i+1) is a 2-by-2 diagonal block, WI(i) > 0 and WI(i+1) = −WI(i). Note that if a complex eigenvalue is sufficiently ill-conditioned, then its value may differ significantly from its value before reordering.

W *CTRSEN only* (output) COMPLEX array, dimension (N)
The reordered eigenvalues of T, in the same order as they appear on the diagonal of T.

M (output) INTEGER
The dimension of the specified invariant subspace (0 $\leq$ M $\leq$ N).

S (output) REAL
If JOB = 'E' or 'B', S is a lower bound on the reciprocal condition number of the selected cluster of eigenvalues. S cannot underestimate the true reciprocal condition number by more than a factor of sqrt(N).
If M = 0 or N, S = 1.
If JOB = 'N' or 'V', S is not referenced.

SEP (output) REAL
If JOB = 'V' or 'B', SEP is the estimated reciprocal condition number of the specified invariant subspace. If M = 0 or N, SEP = $\|T\|$.
If JOB = 'N' or 'E', SEP is not referenced.

WORK (workspace) REAL/COMPLEX array, dimension (LWORK)
CTRSEN only
If JOB = 'N', WORK is not referenced.

LWORK (input) INTEGER
The dimension of the array WORK.
If JOB = 'N', LWORK $\geq$ max(1,N); (*STRSEN only*)
if JOB = 'N', LWORK $\geq$ 1; (*CTRSEN only*)
if JOB = 'E', LWORK $\geq$ max(1,M*(N−M));
if JOB = 'V' or 'B', LWORK $\geq$ max(1,2*M*(N−M)).

IWORK *STRSEN only* (workspace) INTEGER array, dimension (LIWORK)
If JOB = 'N' or 'E', IWORK is not referenced.

LIWORK *STRSEN only* (input) INTEGER
The dimension of the array IWORK.
If JOB = 'N' or 'E', LIWORK $\geq$ 1;
if JOB = 'V' or 'B', LIWORK $\geq$ max(1,M*(N−M)).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

STRSNA/CTRSNA

```
      SUBROUTINE STRSNA( JOB, HOWMNY, SELECT, N, T, LDT, VL, LDVL, VR,
     $                   LDVR, S, SEP, MM, M, WORK, LDWORK, IWORK,
     $                   INFO )
      CHARACTER          HOWMNY, JOB
      INTEGER            INFO, LDT, LDVL, LDVR, LDWORK, M, MM, N
      LOGICAL            SELECT( * )
      INTEGER            IWORK( * )
      REAL               S( * ), SEP( * ), T( LDT, * ), VL( LDVL, * ),
     $                   VR( LDVR, * ), WORK( LDWORK, * )

      SUBROUTINE CTRSNA( JOB, HOWMNY, SELECT, N, T, LDT, VL, LDVL, VR,
     $                   LDVR, S, SEP, MM, M, WORK, LDWORK, RWORK,
     $                   INFO )
      CHARACTER          HOWMNY, JOB
      INTEGER            INFO, LDT, LDVL, LDVR, LDWORK, M, MM, N
      LOGICAL            SELECT( * )
      REAL               RWORK( * ), S( * ), SEP( * )
      COMPLEX            T( LDT, * ), VL( LDVL, * ), VR( LDVR, * ),
     $                   WORK( LDWORK, * )
```

Purpose

STRSNA/CTRSNA estimates reciprocal condition numbers for specified eigenvalues and/or right eigenvectors of a real/complex upper quasi-triangular/triangular matrix T (or of any matrix A = Q*T*Q^H with Q orthogonal/unitary).

STRSNA only
T must be in Schur canonical form (as returned by SHSEQR), that is, block upper triangular with 1-by-1 and 2-by-2 diagonal blocks; each 2-by-2 diagonal block has its diagonal elements equal and its off-diagonal elements of opposite sign.

Arguments

JOB (input) CHARACTER*1
Specifies whether condition numbers are required for eigenvalues (S) or eigenvectors (SEP):
= 'E': for eigenvalues only (S);
= 'V': for eigenvectors only (SEP);
= 'B': for both eigenvalues and eigenvectors (S and SEP).

HOWMNY (input) CHARACTER*1
= 'A': compute condition numbers for all eigenpairs;
= 'S': compute condition numbers for selected eigenpairs specified by the array SELECT.

SELECT (input) LOGICAL array, dimension (N)
If HOWMNY = 'S', SELECT specifies the eigenpairs for which condition numbers are required.
If HOWMNY = 'A', SELECT is not referenced.
STRSNA
To select condition numbers for the eigenpair corresponding to a real eigenvalue w(j), SELECT(j) must be set to .TRUE.. To select condition numbers corresponding to a complex conjugate pair of eigenvalues w(j) and w(j+1), either SELECT(j) or SELECT(j+1) or both must be set to .TRUE..
CTRSNA
To select condition numbers for the j^{th} eigenpair, SELECT(j) must be set to .TRUE..

N (input) INTEGER
The order of the matrix T. N $\geq$ 0.

T (input) REAL/COMPLEX array, dimension (LDT,N)
STRSNA
The upper quasi-triangular matrix T, in Schur canonical form.
CTRSNA
The upper triangular matrix T.

LDT (input) INTEGER
The leading dimension of the array T. LDT $\geq$ max(1,N).

VL (input) REAL/COMPLEX array, dimension (LDVL,M)
If JOB = 'E' or 'B', VL must contain left eigenvectors of T (or of any matrix Q*T*Q^H with Q orthogonal/unitary), corresponding to the eigenpairs specified by HOWMNY and SELECT. The eigenvectors must be stored in consecutive columns of VL, as returned by STREVC/CTREVC or SHSEIN/CHSEIN.
If JOB = 'V', VL is not referenced.

LDVL (input) INTEGER
The leading dimension of the array VL.
LDVL $\geq$ 1; and if JOB = 'E' or 'B', LDVL $\geq$ N.

VR (input) REAL/COMPLEX array, dimension (LDVR,M)
If JOB = 'E' or 'B', VR must contain right eigenvectors of T (or of any matrix Q*T*Q^H with Q orthogonal/unitary), corresponding to the eigenpairs specified by HOWMNY and SELECT. The eigenvectors must be stored in consecutive columns of VR, as returned by STREVC/CTREVC or SHSEIN/CHSEIN.
If JOB = 'V', VR is not referenced.

LDVR (input) INTEGER
The leading dimension of the array VR.
LDVR $\geq$ 1; and if JOB = 'E' or 'B', LDVR $\geq$ N.

S (output) REAL array, dimension (MM)
If JOB = 'E' or 'B', the reciprocal condition numbers of the selected eigenvalues, stored in consecutive elements of the array. Thus S(j), SEP(j), and the j^{th} columns of VL and VR all correspond to the same eigenpair (but not in general the j^{th} eigenpair unless all eigenpairs have been selected).
If JOB = 'V', S is not referenced.
STRSNA only

For a complex conjugate pair of eigenvalues, two consecutive elements of S are set to the same value.

SEP (output) REAL array, dimension (MM)
If JOB = 'V' or 'B', the estimated reciprocal condition numbers of the selected right eigenvectors, stored in consecutive elements of the array.
If JOB = 'E', SEP is not referenced.
STRSNA only
For a complex eigenvector, two consecutive elements of SEP are set to the same value. If the eigenvalues cannot be reordered to compute SEP(j), SEP(j) is set to zero; this can only occur when the true value would be very small anyway.

MM (input) INTEGER
The number of elements in the arrays S (if JOB = 'E' or 'B') and/or SEP (if JOB = 'V' or 'B'). MM $\geq$ M.

M (output) INTEGER
The number of elements of the arrays S and/or SEP actually used to store the estimated condition numbers.
If HOWMNY = 'A', M is set to N.

WORK (workspace) REAL/COMPLEX array, dimension (LDWORK,N+1)
If JOB = 'E', WORK is not referenced.

LDWORK (input) INTEGER
The leading dimension of the array WORK. LDWORK $\geq$ 1; and if JOB = 'V' or 'B', LDWORK $\geq$ N.

IWORK *STRSNA only* (workspace) INTEGER array, dimension (N)
If JOB = 'E', IWORK is not referenced.

RWORK *CTRSNA only* (workspace) REAL array, dimension (N)
If JOB = 'E', RWORK is not referenced.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.

STRSYL/CTRSYL

```
      SUBROUTINE STRSYL( TRANA, TRANB, ISGN, M, N, A, LDA, B, LDB, C,
     $                   LDC, SCALE, INFO )
      CHARACTER          TRANA, TRANB
      INTEGER            INFO, ISGN, LDA, LDB, LDC, M, N
      REAL               SCALE
      REAL               A( LDA, * ), B( LDB, * ), C( LDC, * )
```

```
      SUBROUTINE CTRSYL( TRANA, TRANB, ISGN, M, N, A, LDA, B, LDB, C,
     $                   LDC, SCALE, INFO )
      CHARACTER          TRANA, TRANB
      INTEGER            INFO, ISGN, LDA, LDB, LDC, M, N
      REAL               SCALE
      COMPLEX            A( LDA, * ), B( LDB, * ), C( LDC, * )
```

Purpose

STRSYL/CTRSYL solves the real/complex Sylvester matrix equation:

$$op(A)*X \pm X*op(B) = scale*C,$$

where op(A) = A or A^H, and A and B are both upper quasi-triangular/triangular. A is m-by-m and B is n-by-n; the right hand side C and the solution X are m-by-n; and scale is an output scale factor, set ≤ 1 to avoid overflow in X.

STRSYL only
A and B must be in Schur canonical form (as returned by SHSEQR), that is, block upper triangular with 1-by-1 and 2-by-2 diagonal blocks; each 2-by-2 diagonal block has its diagonal elements equal and its off-diagonal elements of opposite sign.

Arguments

TRANA (input) CHARACTER*1
Specifies the option op(A):
= 'N': op(A) = A (No transpose)
= 'T': op(A) = A^T (Transpose) (*STRSYL only*)
= 'C': op(A) = A^H (Conjugate transpose)

TRANB (input) CHARACTER*1
Specifies the option op(B):
= 'N': op(B) = B (No transpose)
= 'T': op(B) = B^T (Transpose) (*STRSYL only*)
= 'C': op(B) = B^H (Conjugate transpose)

ISGN (input) INTEGER
Specifies the sign in the equation:
= +1: solve op(A)*X + X*op(B) = scale*C
= −1: solve op(A)*X − X*op(B) = scale*C

M (input) INTEGER
The order of the matrix A, and the number of rows in the matrices X and C. M $\geq$ 0.

N (input) INTEGER
The order of the matrix B, and the number of columns in the matrices X and C. N $\geq$ 0.

A (input) REAL/COMPLEX array, dimension (LDA,M)
STRSYL
The upper quasi-triangular matrix A, in Schur canonical form.
CTRSYL
The upper triangular matrix A.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,M).

B (input) REAL/COMPLEX array, dimension (LDB,N)
STRSYL
The upper quasi-triangular matrix B, in Schur canonical form.
CTRSYL
The upper triangular matrix B.

LDB (input) INTEGER
The leading dimension of the array B. LDB $\geq$ max(1,N).

C (input/output) REAL/COMPLEX array, dimension (LDC,N)
On entry, the m-by-n right hand side matrix C.
On exit, C is overwritten by the solution matrix X.

LDC (input) INTEGER
The leading dimension of the array C. LDC $\geq$ max(1,M)

SCALE (output) REAL
The scale factor, scale, set $\leq$ 1 to avoid overflow in X.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
= 1: A and B have common or very close eigenvalues; perturbed values were used to solve the equation (but the matrices A and B are unchanged).

STRTRI/CTRTRI

```
SUBROUTINE STRTRI( UPLO, DIAG, N, A, LDA, INFO )
CHARACTER          DIAG, UPLO
INTEGER            INFO, LDA, N
REAL               A( LDA, * )

SUBROUTINE CTRTRI( UPLO, DIAG, N, A, LDA, INFO )
CHARACTER          DIAG, UPLO
INTEGER            INFO, LDA, N
COMPLEX            A( LDA, * )
```

Purpose

STRTRI/CTRTRI computes the inverse of a real/complex upper or lower triangular matrix A.

Arguments

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. N $\geq$ 0.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the triangular matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of the array A contains the upper triangular matrix, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of the array A contains the lower triangular matrix, and the strictly upper triangular part of A is not referenced. If DIAG = 'U', the diagonal elements of A are also not referenced and are assumed to be 1.
On exit, the (triangular) inverse of the original matrix, in the same storage format.

LDA (input) INTEGER
The leading dimension of the array A. LDA $\geq$ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, A(i,i) is exactly zero. The triangular matrix is singular and its inverse can not be computed.

STRTRS/CTRTRS

```
      SUBROUTINE STRTRS( UPLO, TRANS, DIAG, N, NRHS, A, LDA, B, LDB,
     $                   INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, LDA, LDB, N, NRHS
      REAL               A( LDA, * ), B( LDB, * )

      SUBROUTINE CTRTRS( UPLO, TRANS, DIAG, N, NRHS, A, LDA, B, LDB,
     $                   INFO )
      CHARACTER          DIAG, TRANS, UPLO
      INTEGER            INFO, LDA, LDB, N, NRHS
      COMPLEX            A( LDA, * ), B( LDB, * )
```

Purpose

STRTRS/CTRTRS solves a triangular system of the form $A*X = B$, $A^T*X = B$, or $A^H*X = B$, where A is a triangular matrix of order n, and B is an n-by-nrhs matrix. A check is made to verify that A is nonsingular.

Arguments

UPLO (input) CHARACTER*1
= 'U': A is upper triangular;
= 'L': A is lower triangular.

TRANS (input) CHARACTER*1
Specifies the form of the system of equations:
= 'N': A*X = B (No transpose)
= 'T': A^T*X = B (Transpose)
= 'C': A^H*X = B (Conjugate transpose)

DIAG (input) CHARACTER*1
= 'N': A is non-unit triangular;
= 'U': A is unit triangular.

N (input) INTEGER
The order of the matrix A. N ≥ 0.

NRHS (input) INTEGER
The number of right hand sides, i.e., the number of columns of the matrix B. NRHS ≥ 0.

A (input) REAL/COMPLEX array, dimension (LDA,N)
The triangular matrix A. If UPLO = 'U', the leading n-by-n upper triangular part of the array A contains the upper triangular matrix, and the strictly lower triangular part of A is not referenced. If UPLO = 'L', the leading n-by-n lower triangular part of the array A contains the lower triangular matrix, and the strictly upper triangular part of A is not referenced. If DIAG = 'U', the diagonal elements of A are also not referenced and are assumed to be 1.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,N).

B (input/output) REAL/COMPLEX array, dimension (LDB,NRHS)
On entry, the right hand side matrix B.
On exit, if INFO = 0, the solution matrix X.

LDB (input) INTEGER
The leading dimension of the array B. LDB ≥ max(1,N).

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = −i, the i^{th} argument had an illegal value.
> 0: if INFO = i, the i^{th} diagonal element of A is zero, indicating that the matrix is singular and the solutions X have not been computed.

STZRZF/CTZRZF

```
SUBROUTINE STZRZF( M, N, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, LDA, LWORK, M, N
REAL               A( LDA, * ), TAU( * ), WORK( LWORK )

SUBROUTINE CTZRZF( M, N, A, LDA, TAU, WORK, LWORK, INFO )
INTEGER            INFO, LDA, LWORK, M, N
COMPLEX            A( LDA, * ), TAU( * ), WORK( LWORK )
```

Purpose

STZRZF/CTZRZF reduces the m-by-n (m ≤ n) real/complex upper trapezoidal matrix A to upper triangular form by m :ans of orthogonal/unitary transformations.

The upper trapezoidal matrix A is factorized as

$$A = \begin{pmatrix} R & 0 \end{pmatrix} * Z,$$

where Z is an n-by-n orthogonal/unitary matrix and R is an m-by-m upper triangular matrix.

Arguments

M (input) INTEGER
The number of rows of the matrix A. M ≥ 0.

N (input) INTEGER
The number of columns of the matrix A. N ≥ M.

A (input/output) REAL/COMPLEX array, dimension (LDA,N)
On entry, the leading m-by-n upper trapezoidal part of the array A must contain the matrix to be factorized.
On exit, the leading m-by-m upper triangular part of A contains the upper triangular matrix R, and elements m+1 to n of the first m rows of A, with the array TAU, represent the orthogonal/unitary matrix Z as a product of m elementary reflectors.

LDA (input) INTEGER
The leading dimension of the array A. LDA ≥ max(1,M).

TAU (output) REAL/COMPLEX array, dimension (M)
The scalar factors of the elementary reflectors.

WORK (workspace/output) REAL/COMPLEX array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal WORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK ≥ max(1,M).
For optimum performance LWORK ≥ M*NB, where NB is the optimal blocksize.

If LWORK = −1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.

INFO (output) INTEGER
= 0: successful exit
< 0: if INFO = $-i$, the i^{th} argument had an illegal value.