NAG Fortran Library Routine Document

F08MSF (CBDSQR/ZBDSQR)

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

Warning. The specification of the parameter WORK changed at Mark 20: the length of WORK needs to be increased.

1 Purpose

F08MSF (CBDSQR/ZBDSQR) computes the singular value decomposition of a complex general matrix which has been reduced to bidiagonal form.

2 Specification

```
 SUBROUTINE F08MSF(UPLO, N, NCVT, NRU, NCC, D, E, VT, LDVT, U, LDU, C,
1                  LDC, WORK, INFO)
 ENTRY      cbdsqr (UPLO, N, NCVT, NRU, NCC, D, E, VT, LDVT, U, LDU, C,
1                  LDC, WORK, INFO)
 INTEGER           N, NCVT, NRU, NCC, LDVT, LDU, LDC, INFO
 real              D(*), E(*), WORK(*)
 complex           VT(LDVT,*), U(LDU,*), C(LDC,*)
 CHARACTER*1       UPLO
```

The ENTRY statement enables the routine to be called by its LAPACK name.

3 Description

This routine computes the singular values, and optionally, the left or right singular vectors of a real upper or lower bidiagonal matrix B. In other words, it can compute the singular value decomposition (SVD) of B as

$$B = U\Sigma V^T.$$

Here Σ is a diagonal matrix with real diagonal elements σ_i (the singular values of B), such that

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0;$$

U is an orthogonal matrix whose columns are the left singular vectors u_i; V is an orthogonal matrix whose rows are the right singular vectors v_i. Thus

$$Bu_i = \sigma_i v_i \quad \text{and} \quad B^T v_i = \sigma_i u_i, \quad i = 1, 2, \ldots, n.$$

To compute U and/or V^T, the arrays U and/or VT must be initialised to the unit matrix before F08MSF (CBDSQR/ZBDSQR) is called.

The routine stores the real orthogonal matrices U and V^T in complex arrays U and VT, so that it may also be used to compute the SVD of a complex general matrix A which has been reduced to bidiagonal form by a unitary transformation: $A = QBP^H$. If A is m by n with $m \geq n$, then Q is m by n and P^H is n by n; if A is n by p with $n < p$, then Q is n by n and P^H is n by p. In this case, the matrices Q and/or P^H must be formed explicitly by F08KTF (CUNGBR/ZUNGBR) and passed to F08MSF (CBDSQR/ZBDSQR) in the arrays U and/or VT respectively.

F08MSF (CBDSQR/ZBDSQR) also has the capability of forming $U^H C$, where C is an arbitrary complex matrix; this is needed when using the SVD to solve linear least-squares problems.

F08MSF (CBDSQR/ZBDSQR) uses two different algorithms. If any singular vectors are required (that is, if NCVT > 0 or NRU > 0 or NCC > 0), the bidiagonal QR algorithm is used, switching between zero-shift and implicitly shifted forms to preserve the accuracy of small singular values, and switching between QR and QL variants in order to handle graded matrices effectively (see Demmel and Kahan (1990)). If only singular values are required (that is, if NCVT $=$ NRU $=$ NCC $= 0$), they are computed by the differential qd algorithm (see Fernando and Parlett (1994)), which is faster and can achieve even greater accuracy.

The singular vectors are normalized so that $\|u_i\| = \|v_i\| = 1$, but are determined only to within a complex factor of absolute value 1.

4 References

Demmel J W and Kahan W (1990) Accurate singular values of bidiagonal matrices *SIAM J. Sci. Statist. Comput.* **11** 873–912

Fernando K V and Parlett B N (1994) Accurate singular values and differential qd algorithms *Numer. Math.* **67** 191–229

Golub G H and van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

5 Parameters

1: UPLO – CHARACTER*1 *Input*

On entry: indicates whether B is an upper or lower bidiagonal matrix as follows:

if UPLO = 'U', B is an upper bidiagonal matrix;

if UPLO = 'L', B is a lower bidiagonal matrix.

Constraint: UPLO = 'U' or 'L'.

2: N – INTEGER *Input*

On entry: n, the order of the matrix B.

Constraint: N $\geq$ 0.

3: NCVT – INTEGER *Input*

On entry: $ncvt$, the number of columns of the matrix V^H of right singular vectors. Set NCVT = 0 if no right singular vectors are required.

Constraint: NCVT $\geq$ 0.

4: NRU – INTEGER *Input*

On entry: nru, the number of rows of the matrix U of left singular vectors. Set NRU = 0 if no left singular vectors are required.

Constraint: NRU $\geq$ 0.

5: NCC – INTEGER *Input*

On entry: ncc, the number of columns of the matrix C. Set NCC = 0 if no matrix C is supplied.

Constraint: NCC $\geq$ 0.

6: D(*) – ***real*** array *Input/Output*

Note: the dimension of the array D must be at least max(1, N).

On entry: the diagonal elements of the bidiagonal matrix B.

On exit: the singular values in decreasing order of magnitude, unless INFO > 0 (in which case see Section 6).

7: E(*) – ***real*** array *Input/Output*

Note: the dimension of the array E must be at least max(1, N − 1).

On entry: the off-diagonal elements of the bidiagonal matrix B.

On exit: the array is overwritten, but if INFO > 0 see Section 6.

8: VT(LDVT,*) – ***complex*** array *Input/Output*

Note: the second dimension of the array VT must be at least max(1, NCVT).

On entry: if NCVT > 0, VT must contain an n by $ncvt$ matrix. If the right singular vectors of B are required, $ncvt = n$ and VT must contain the unit matrix; if the right singular vectors of A are required, VT must contain the unitary matrix P^H returned by F08KTF (CUNGBR/ZUNGBR) with VECT = 'P' .

On exit: the n by $ncvt$ matrix V^H or $V^H P^H$ of right singular vectors, stored by rows.

VT is not referenced if NCVT = 0.

9: LDVT – INTEGER *Input*

On entry: the first dimension of the array VT as declared in the (sub)program from which F08MSF (CBDSQR/ZBDSQR) is called.

Constraints:

LDVT ≥ max(1, N) if NCVT > 0,
LDVT ≥ 1 otherwise.

10: U(LDU,*) – ***complex*** array *Input/Output*

Note: the second dimension of the array U must be at least max(1, N).

On entry: if NRU > 0, U must contain an nru by n matrix. If the left singular vectors of B are required, $nru = n$ and U must contain the unit matrix; if the left singular vectors of A are required, U must contain the unitary matrix Q returned by F08KTF (CUNGBR/ZUNGBR) with VECT = 'Q'.

On exit: the nru by n matrix U or QU of left singular vectors, stored by columns.

U is not referenced if NRU = 0.

11: LDU – INTEGER *Input*

On entry: the first dimension of the array U as declared in the (sub)program from which F08MSF (CBDSQR/ZBDSQR) is called.

Constraint: LDU ≥ max(1, NRU).

12: C(LDC,*) – ***complex*** array *Input/Output*

Note: the second dimension of the array C must be at least max(1, NCC).

On entry: the n by ncc matrix C if NCC > 0.

On exit: C is overwritten by the matrix $U^H C$.

C is not referenced if NCC = 0.

13: LDC – INTEGER *Input*

On entry: the first dimension of the array C as declared in the (sub)program from which F08MSF (CBDSQR/ZBDSQR) is called.

Constraints:

LDC ≥ max(1, N) if NCC > 0,
LDC ≥ 1 otherwise.

14: WORK(*) – ***real*** array *Workspace*

Note: the dimension of the array WORK must be at least $\max(1, 4 * \mathrm{N})$.

15: INFO – INTEGER *Output*

On exit: INFO $= 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

INFO < 0

If INFO $= -i$, the ith parameter had an illegal value. An explanatory message is output, and execution of the program is terminated.

INFO > 0

The algorithm failed to converge and INFO specifies how many off-diagonals did not converge. In this case, D and E contain on exit the diagonal and off-diagonal elements, respectively, of a bidiagonal matrix orthogonally equivalent to B.

7 Accuracy

Each singular value and singular vector is computed to high relative accuracy. However, the reduction to bidiagonal form (prior to calling the routine) may exclude the possibility of obtaining high relative accuracy in the small singular values of the original matrix if its singular values vary widely in magnitude.

If σ_i is an exact singular value of B, and $\tilde{\sigma}_i$ is the corresponding computed value, then

$$|\tilde{\sigma}_i - \sigma_i| \leq p(m,n)\epsilon\sigma_i$$

where $p(m,n)$ is a modestly increasing function of m and n, and ϵ is the ***machine precision***. If only singular values are computed, they are computed more accurately (that is, the function $p(m,n)$ is smaller), than when some singular vectors are also computed.

If u_i is an exact left singular vector of B, and $\tilde{u}_i$ is the corresponding computed left singular vector, then the angle $\theta(\tilde{u}_i, u_i)$ between them is bounded as follows:

$$\theta(\tilde{u}_i, u_i) \leq \frac{p(m,n)\epsilon}{relgap_i}$$

where $relgap_i$ is the relative gap between σ_i and the other singular values, defined by

$$relgap_i = \min_{i \neq j} \frac{|\sigma_i - \sigma_j|}{(\sigma_i + \sigma_j)}.$$

A similar error bound holds for the right singular vectors.

8 Further Comments

The total number of real floating-point operations is roughly proportional to n^2 if only the singular values are computed. About $12n^2 \times nru$ additional operations are required to compute the left singular vectors and about $12n^2 \times ncvt$ to compute the right singular vectors. The operations to compute the singular values must all be performed in scalar mode; the additional operations to compute the singular vectors can be vectorized and on some machines may be performed much faster.

The real analogue of this routine is F08MEF (SBDSQR/DBDSQR).

9 Example

See Section 9 of the document for F08KTF (CUNGBR/ZUNGBR), which illustrates the use of the routine to compute the singular value decomposition of a general matrix.