# NAG Toolbox

# nag_lapack_dbdsdc (f08md)

## 1 Purpose

nag_lapack_dbdsdc (f08md) computes the singular values and, optionally, the left and right singular vectors of a real $n$ by $n$ (upper or lower) bidiagonal matrix $B$.

## 2 Syntax

```
[d, e, u, vt, q, iq, info] = nag_lapack_dbdsdc(uplo, compq, d, e, 'n', n, 'ldq', ldq, 'ldiq', ldiq)
[d, e, u, vt, q, iq, info] = f08md(uplo, compq, d, e, 'n', n, 'ldq', ldq, 'ldiq', ldiq)
```

## 3 Description

nag_lapack_dbdsdc (f08md) computes the singular value decomposition (SVD) of the (upper or lower) bidiagonal matrix $B$ as

$$B = USV^{\mathrm{T}},$$

where $S$ is a diagonal matrix with non-negative diagonal elements $s_{ii} = s_i$, such that

$$s_1 \geq s_2 \geq \cdots \geq s_n \geq 0,$$

and $U$ and $V$ are orthogonal matrices. The diagonal elements of $S$ are the singular values of $B$ and the columns of $U$ and $V$ are respectively the corresponding left and right singular vectors of $B$.

When only singular values are required the function uses the $QR$ algorithm, but when singular vectors are required a divide and conquer method is used. The singular values can optionally be returned in compact form, although currently no function is available to apply $U$ or $V$ when stored in compact form.

## 4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia http://www.netlib.org/lapack/lug

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

## 5 Parameters

### 5.1 Compulsory Input Parameters

1: **uplo** – CHARACTER(1)

Indicates whether $B$ is upper or lower bidiagonal.

**uplo** = 'U'
$B$ is upper bidiagonal.

**uplo** = 'L'
$B$ is lower bidiagonal.

*Constraint*: **uplo** = 'U' or 'L'.

2: **compq** – CHARACTER(1)

Specifies whether singular vectors are to be computed.

**compq** = 'N'
Compute singular values only.

**compq** = 'P'
Compute singular values and compute singular vectors in compact form.

**compq** = 'I'
Compute singular values and singular vectors.

*Constraint*: **compq** = 'N', 'P' or 'I'.

3: **d**(:) – REAL (KIND=nag_wp) array

The dimension of the array **d** must be at least $\max(1, \mathbf{n})$

The $n$ diagonal elements of the bidiagonal matrix $B$.

4: **e**(:) – REAL (KIND=nag_wp) array

The dimension of the array **e** must be at least $\max(1, \mathbf{n} - 1)$

The $(n - 1)$ off-diagonal elements of the bidiagonal matrix $B$.

## 5.2 Optional Input Parameters

1: **n** – INTEGER

*Default*: the dimension of the array **d**.

$n$, the order of the matrix $B$.

*Constraint*: $\mathbf{n} \geq 0$.

2: **ldq** – INTEGER

*Default*:

if **compq** = 'P', $\mathbf{n} \times (61 + 8 \times \text{int}(\log_2 \mathbf{n}/26))$;
otherwise 1.

The dimension of the array **q**. see the description of **q**.

3: **ldiq** – INTEGER

*Default*:

if **compq** = 'P', $\mathbf{n} \times (3 + 3 \times \text{int}\log_2 \mathbf{n}/(26))$;
otherwise 1.

The dimension of the array **iq**. see the description of **iq**.

## 5.3 Output Parameters

1: **d**(:) – REAL (KIND=nag_wp) array

The dimension of the array **d** will be $\max(1, \mathbf{n})$

If **info** = 0, the singular values of $B$.

2: **e**(:) – REAL (KIND=nag_wp) array

The dimension of the array **e** will be $\max(1, \mathbf{n} - 1)$

The contents of **e** are destroyed.

3: **u**$(ldu,:)$ – REAL (KIND=nag_wp) array

The first dimension, $ldu$, of the array **u** will be

if **compq** = 'I', $ldu = \max(1, \mathbf{n})$;
otherwise $ldu = 1$.

The second dimension of the array **u** will be $\max(1, \mathbf{n})$ if **compq** = 'I' and 1 otherwise.

If **compq** = 'I', then if **info** $= 0$, **u** contains the left singular vectors of the bidiagonal matrix $B$.

If **compq** $\neq$ 'I', **u** is not referenced.

4: **vt**$(ldvt,:)$ – REAL (KIND=nag_wp) array

The first dimension, $ldvt$, of the array **vt** will be

if **compq** = 'I', $ldvt = \max(1, \mathbf{n})$;
otherwise $ldvt = 1$.

The second dimension of the array **vt** will be $\max(1, \mathbf{n})$ if **compq** = 'I' and 1 otherwise.

If **compq** = 'I', then if **info** $= 0$, the rows of **vt** contain the right singular vectors of the bidiagonal matrix $B$.

If **compq** $\neq$ 'I', **vt** is not referenced.

5: **q**$(:)$ – REAL (KIND=nag_wp) array

The dimension of the array **q** will be **ldq**

If **compq** = 'P', then if **info** $= 0$, **q** and **iq** contain the left and right singular vectors in a compact form, requiring $O(\mathbf{n}\log_2 \mathbf{n})$ space instead of $2 \times \mathbf{n}^2$. In particular, **q** contains all the real data in the first **ldq** $= \mathbf{n} \times (11 + 2 \times smlsiz + 8 \times \text{int}(\log_2 (\mathbf{n}/(smlsiz + 1))))$ elements of **q**, where $smlsiz$ is equal to the maximum size of the subproblems at the bottom of the computation tree (usually about 25).

If **compq** $\neq$ 'P', **q** is not referenced.

6: **iq**$(:)$ – INTEGER array

The dimension of the array **iq** will be **ldiq**

If **compq** = 'P', then if **info** $= 0$, **q** and **iq** contain the left and right singular vectors in a compact form, requiring $O(\mathbf{n}\log_2 \mathbf{n})$ space instead of $2 \times \mathbf{n}^2$. In particular, **iq** contains all integer data in the first **ldiq** $= \mathbf{n} \times (3 + 3 \times \text{int}(\log_2 (\mathbf{n}/(smlsiz + 1))))$ elements of **iq**, where $smlsiz$ is equal to the maximum size of the subproblems at the bottom of the computation tree (usually about 25).

If **compq** $\neq$ 'P', **iq** is not referenced.

7: **info** – INTEGER

**info** $= 0$ unless the function detects an error (see Section 6).

## 6 Error Indicators and Warnings

**info** $= -i$

If **info** $= -i$, parameter $i$ had an illegal value on entry. The parameters are numbered as follows:

1: **uplo**, 2: **compq**, 3: **n**, 4: **d**, 5: **e**, 6: **u**, 7: **ldu**, 8: **vt**, 9: **ldvt**, 10: **q**, 11: **ldq**, 12: **iq**, 13: **ldiq**, 14: **work**, 15: **iwork**, 16: **info**.

It is possible that **info** refers to a parameter that is omitted from the MATLAB interface. This usually indicates that an error in one of the other input parameters has caused an incorrect value to be inferred.

**info** $> 0$

The algorithm failed to compute a singular value. The update process of divide-and-conquer failed.

## 7 Accuracy

Each computed singular value of $B$ is accurate to nearly full relative precision, no matter how tiny the singular value. The $i$th computed singular value, $\hat{s}_i$, satisfies the bound

$$|\hat{s}_i - s_i| \leq p(n)\epsilon s_i$$

where $\epsilon$ is the ***machine precision*** and $p(n)$ is a modest function of $n$.

For bounds on the computed singular values, see Section 4.9.1 of Anderson *et al.* (1999). See also nag_lapack_ddisna (f08fl).

## 8 Further Comments

If only singular values are required, the total number of floating-point operations is approximately proportional to $n^2$. When singular vectors are required the number of operations is bounded above by approximately the same number of operations as nag_lapack_dbdsqr (f08me), but for large matrices nag_lapack_dbdsdc (f08md) is usually much faster.

There is no complex analogue of nag_lapack_dbdsdc (f08md).

## 9 Example

This example computes the singular value decomposition of the upper bidiagonal matrix

$$B = \begin{pmatrix} 3.62 & 1.26 & 0 & 0 \\ 0 & -2.41 & -1.53 & 0 \\ 0 & 0 & 1.92 & 1.19 \\ 0 & 0 & 0 & -1.43 \end{pmatrix}.$$

### 9.1 Program Text

```
function f08md_example

fprintf('f08md example results\n\n');

% Bidiagonal matrix B stored as diagonal and off-diagonal
uplo = 'Upper';
d = [3.62;     -2.41;     1.92;     -1.43];
e = [1.26;     -1.53;     1.19];

% Compute singular values
compq = 'I';
[s, ~, ~, ~, ~, ~, info] = f08md( ...
                                  uplo, compq, d, e);

disp('Singular values');
disp(s');
```

### 9.2 Program Results

```
f08md example results

Singular values
    4.0001    3.0006    1.9960    0.9998
```