# NAG Library Function Document

# nag_dstebz (f08jjc)

## 1 Purpose

nag_dstebz (f08jjc) computes some (or all) of the eigenvalues of a real symmetric tridiagonal matrix, by bisection.

## 2 Specification

```
#include <nag.h>
#include <nagf08.h>
```

```
void nag_dstebz (Nag_RangeType range, Nag_EigValRankType rank, Integer n,
     double vl, double vu, Integer il, Integer iu, double abstol,
     const double d[], const double e[], Integer *m, Integer *nsplit,
     double w[], Integer iblock[], Integer isplit[], NagError *fail)
```

## 3 Description

nag_dstebz (f08jjc) uses bisection to compute some or all of the eigenvalues of a real symmetric tridiagonal matrix $T$.

It searches for zero or negligible off-diagonal elements of $T$ to see if the matrix splits into block diagonal form:

$$T = \begin{pmatrix} T_1 & & & & & \\ & T_2 & & & & \\ & & \cdot & & & \\ & & & \cdot & & \\ & & & & \cdot & \\ & & & & & T_p \end{pmatrix}.$$

It performs bisection on each of the blocks $T_i$ and returns the block index of each computed eigenvalue, so that a subsequent call to nag_dstein (f08jkc) to compute eigenvectors can also take advantage of the block structure.

## 4 References

Kahan W (1966) Accurate eigenvalues of a symmetric tridiagonal matrix *Report CS41* Stanford University

## 5 Arguments

1: **range** – Nag_RangeType *Input*

*On entry*: indicates which eigenvalues are required.

**range** = Nag_AllValues
All the eigenvalues are required.

**range** = Nag_Interval
All the eigenvalues in the half-open interval (**vl**,**vu**] are required.

**range** = Nag_Indices
Eigenvalues with indices **il** to **iu** are required.

*Constraint*: **range** = Nag_AllValues, Nag_Interval or Nag_Indices.

2: **rank** – Nag_EigValRankType *Input*

*On entry*: indicates the order in which the eigenvalues and their block numbers are to be stored.

**rank** = Nag_ByBlock
The eigenvalues are to be grouped by split-off block and ordered from smallest to largest within each block.

**rank** = Nag_Entire
The eigenvalues for the entire matrix are to be ordered from smallest to largest.

*Constraint*: **rank** = Nag_ByBlock or Nag_Entire.

3: **n** – Integer *Input*

*On entry*: $n$, the order of the matrix $T$.

*Constraint*: **n** $\geq 0$.

4: **vl** – double *Input*
5: **vu** – double *Input*

*On entry*: if **range** = Nag_Interval, the lower and upper bounds, respectively, of the half-open interval (**vl**,**vu**] within which the required eigenvalues lie.

If **range** = Nag_AllValues or Nag_Indices, **vl** is not referenced.

*Constraint*: if **range** = Nag_Interval, **vl** < **vu**.

6: **il** – Integer *Input*
7: **iu** – Integer *Input*

*On entry*: if **range** = Nag_Indices, the indices of the first and last eigenvalues, respectively, to be computed (assuming that the eigenvalues are in ascending order).

If **range** = Nag_AllValues or Nag_Interval, **il** is not referenced.

*Constraint*: if **range** = Nag_Indices, $1 \leq$ **il** $\leq$ **iu** $\leq$ **n**.

8: **abstol** – double *Input*

*On entry*: the absolute tolerance to which each eigenvalue is required. An eigenvalue (or cluster) is considered to have converged if it lies in an interval of width $\leq$ **abstol**. If **abstol** $\leq 0.0$, then the tolerance is taken as ***machine precision*** $\times \|T\|_1$.

9: **d**[*dim*] – const double *Input*

**Note**: the dimension, *dim*, of the array **d** must be at least max(1, **n**).

*On entry*: the diagonal elements of the tridiagonal matrix $T$.

10: **e**[*dim*] – const double *Input*

**Note**: the dimension, *dim*, of the array **e** must be at least max(1, **n** − 1).

*On entry*: the off-diagonal elements of the tridiagonal matrix $T$.

11: **m** – Integer * *Output*

*On exit*: $m$, the actual number of eigenvalues found.

12: **nsplit** – Integer * *Output*

*On exit*: the number of diagonal blocks which constitute the tridiagonal matrix $T$.

13: **w**[**n**] – double *Output*

*On exit*: the required eigenvalues of the tridiagonal matrix $T$ stored in **w**[0] to **w**[$m-1$].

14: **iblock**[**n**] – Integer *Output*

*On exit*: at each row/column $j$ where **e**$[j-1]$ is zero or negligible, $T$ is considered to split into a block diagonal matrix and **iblock**$[i-1]$ contains the block number of the eigenvalue stored in **w**$[i-1]$, for $i = 1, 2, \ldots, m$. Note that **iblock**$[i-1] < 0$ for some $i$ whenever **fail.code** = NE_CONVERGENCE (see Section 6) and **range** = Nag_AllValues or Nag_Interval.

15: **isplit**[**n**] – Integer *Output*

*On exit*: the leading **nsplit** elements contain the points at which $T$ splits up into sub-matrices as follows. The first sub-matrix consists of rows/columns 1 to **isplit**[0], the second sub-matrix consists of rows/columns **isplit**[0] + 1 to **isplit**[1], . . ., and the **nsplit**(th) sub-matrix consists of rows/columns **isplit**[**nsplit** − 2] + 1 to **isplit**[**nsplit** − 1] ( = $n$).

16: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

## 6 Error Indicators and Warnings

**NE_ALLOC_FAIL**

Dynamic memory allocation failed.
See Section 3.2.1.2 in How to Use the NAG Library and its Documentation for further information.

**NE_BAD_PARAM**

On entry, argument ⟨*value*⟩ had an illegal value.

**NE_CONVERGENCE**

If **range** = Nag_AllValues or Nag_Interval, the algorithm failed to compute some (or all) of the required eigenvalues to the required accuracy. More precisely, **iblock**[⟨*value*⟩] < 0 indicates that eigenvalue ⟨*value*⟩ (stored in **w**[⟨*value*⟩]) failed to converge.

If **range** = Nag_Indices, the algorithm failed to compute some (or all) of the required eigenvalues. Try calling the function again with **range** = Nag_AllValues.

If **range** = Nag_Indices, the algorithm failed to compute some (or all) of the required eigenvalues. Try calling the function again with **range** = Nag_AllValues. If **range** = Nag_AllValues or Nag_Interval, the algorithm failed to compute some (or all) of the required eigenvalues to the required accuracy. More precisely, **iblock**[⟨*value*⟩] < 0 indicates that eigenvalue ⟨*value*⟩ (stored in **w**[⟨*value*⟩]) failed to converge.

No eigenvalues have been computed. The floating-point arithmetic on the computer is not behaving as expected.

**NE_ENUM_INT_3**

On entry, **range** = ⟨*value*⟩, **n** = ⟨*value*⟩, **il** = ⟨*value*⟩ and **iu** = ⟨*value*⟩.
Constraint: if **range** = Nag_Indices, 1 ≤ **il** ≤ **iu** ≤ **n**.

**NE_ENUM_REAL_2**

On entry, **range** = ⟨*value*⟩, **vl** = ⟨*value*⟩ and **vu** = ⟨*value*⟩.
Constraint: if **range** = Nag_Interval, **vl** < **vu**.

**NE_INT**

On entry, **n** = ⟨*value*⟩.
Constraint: **n** ≥ 0.

**NE_INTERNAL_ERROR**

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in How to Use the NAG Library and its Documentation for further information.

**NE_NO_LICENCE**

Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in How to Use the NAG Library and its Documentation for further information.

## 7 Accuracy

The eigenvalues of $T$ are computed to high relative accuracy which means that if they vary widely in magnitude, then any small eigenvalues will be computed more accurately than, for example, with the standard $QR$ method. However, the reduction to tridiagonal form (prior to calling the function) may exclude the possibility of obtaining high relative accuracy in the small eigenvalues of the original matrix if its eigenvalues vary widely in magnitude.

## 8 Parallelism and Performance

nag_dstebz (f08jjc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

Please consult the x06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users' Notefor your implementation for any additional implementation-specific information.

## 9 Further Comments

There is no complex analogue of this function.

## 10 Example

See Section 10 in nag_dormtr (f08fgc).