# NAG C Library Function Document

# nag_dstein (f08jkc)

## 1 Purpose

nag_dstein (f08jkc) computes the eigenvectors of a real symmetric tridiagonal matrix corresponding to specified eigenvalues, by inverse iteration.

## 2 Specification

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

void nag_dstein (Nag_OrderType order, Integer n, const double d[],
      const double e[], Integer m, const double w[], const Integer iblock[],
      const Integer isplit[], double z[], Integer pdz, Integer ifailv[], NagError *fail)
```

## 3 Description

nag_dstein (f08jkc) computes the eigenvectors of a real symmetric tridiagonal matrix $T$ corresponding to specified eigenvalues, by inverse iteration (see Jessup and Ipsen (1992)). It is designed to be used in particular after the specified eigenvalues have been computed by nag_dstebz (f08jjc) with **rank** = **Nag_ByBlock**, but may also be used when the eigenvalues have been computed by other functions in Chapters f08 or f02.

If $T$ has been formed by reduction of a full real symmetric matrix $A$ to tridiagonal form, then eigenvectors of $T$ may be transformed to eigenvectors of $A$ by a call to nag_dormtr (f08fgc) or nag_dopmtr (f08ggc).

nag_dstebz (f08jjc) determines whether the matrix $T$ splits into block diagonal form:

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

and passes details of the block structure to this function in the arrays **iblock** and **isplit**. This function can then take advantage of the block structure by performing inverse iteration on each block $T_i$ separately, which is more efficient than using the whole matrix.

## 4 References

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

Jessup E and Ipsen I C F (1992) Improving the accuracy of inverse iteration *SIAM J. Sci. Statist. Comput.* **13** 550–572

## 5 Arguments

1: **order** – Nag_OrderType *Input*

*On entry*: the **order** argument specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by **order** = **Nag_RowMajor**. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this argument.

*Constraint*: **order** = **Nag_RowMajor** or **Nag_ColMajor**.

2: **n** – Integer *Input*

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

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

3: **d**[$dim$] – const double *Input*

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

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

4: **e**[$dim$] – const double *Input*

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

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

5: **m** – Integer *Input*

*On entry*: $m$, the number of eigenvectors to be returned.

*Constraint*: $0 \leq \mathbf{m} \leq \mathbf{n}$.

6: **w**[$dim$] – const double *Input*

**Note**: the dimension, *dim*, of the array **w** must be at least $\max(1, \mathbf{n})$.

*On entry*: the eigenvalues of the tridiagonal matrix $T$ stored in $\mathbf{w}[0]$ to $\mathbf{w}[m]$, as returned by nag_dstebz (f08jjc) with **rank** = **Nag_ByBlock**. Eigenvalues associated with the first sub-matrix must be supplied first, in non-decreasing order; then those associated with the second sub-matrix, again in non-decreasing order; and so on.

*Constraint*: $\mathbf{w}[i] \leq \mathbf{w}[i+1]$ if $\mathbf{iblock}[i] = \mathbf{iblock}[i+1]$, for $i = 0, 1, \ldots, \mathbf{m} - 2$.

7: **iblock**[$dim$] – const Integer *Input*

**Note**: the dimension, *dim*, of the array **iblock** must be at least $\max(1, \mathbf{n})$.

*On entry*: the first $m$ elements must contain the sub-matrix indices associated with the specified eigenvalues, as returned by nag_dstebz (f08jjc) with **rank** = **Nag_ByBlock**. If the eigenvalues were not computed by nag_dstebz (f08jjc) with **rank** = **Nag_ByBlock**, set $\mathbf{iblock}[i-1]$ to 1 for $i = 1, 2, \ldots, m$.

*Constraint*: $\mathbf{iblock}[i] \leq \mathbf{iblock}[i+1]$, for $i = 0, 1, \ldots, \mathbf{m} - 2$.

8: **isplit**[$dim$] – const Integer *Input*

**Note**: the dimension, *dim*, of the array **isplit** must be at least $\max(1, \mathbf{n})$.

*On entry*: the points at which $T$ breaks up into sub-matrices, as returned by nag_dstebz (f08jjc) with **rank** = **Nag_ByBlock**. If the eigenvalues were not computed by nag_dstebz (f08jjc) with **rank** = **Nag_ByBlock**, set $\mathbf{isplit}[0]$ to **n**.

9: **z**[$dim$] – double *Output*

**Note**: the dimension, *dim*, of the array **z** must be at least

$\max(1, \mathbf{pdz} \times \mathbf{m})$ when **order** = **Nag_ColMajor**;
$\max(1, \mathbf{pdz} \times \mathbf{n})$ when **order** = **Nag_RowMajor**.

If **order** = **Nag_ColMajor**, the $(i,j)$th element of the matrix $Z$ is stored in $\mathbf{z}[(j-1) \times \mathbf{pdz} + i - 1]$.

If **order** = **Nag_RowMajor**, the $(i,j)$th element of the matrix $Z$ is stored in $\mathbf{z}[(i-1) \times \mathbf{pdz} + j - 1]$.

*On exit*: the $m$ eigenvectors, stored as columns of $Z$; the $i$th column corresponds to the $i$th specified eigenvalue, unless **fail.code** $> 0$ (in which case see Section 6).

10: **pdz** – Integer *Input*

*On entry*: the stride separating matrix row or column elements (depending on the value of **order**) in the array **z**.

*Constraints*:

if **order** = **Nag_ColMajor**, **pdz** $\geq \max(1, \mathbf{n})$;
if **order** = **Nag_RowMajor**, **pdz** $\geq \max(1, \mathbf{m})$.

11: **ifailv**[*dim*] – Integer *Output*

**Note**: the dimension, *dim*, of the array **ifailv** must be at least $\max(1, \mathbf{m})$.

*On exit*: if **fail.code** $= i > 0$, the first $i$ elements of **ifailv** contain the indices of any eigenvectors which have failed to converge. The rest of the first **m** elements of **ifailv** are set to 0.

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

The NAG error argument (see Section 2.6 of the Essential Introduction).

# 6 Error Indicators and Warnings

**NE_ALLOC_FAIL**

Dynamic memory allocation failed.

**NE_BAD_PARAM**

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

**NE_CONSTRAINT**

On entry, **m** = ⟨*value*⟩, **iblock**[$i$]**iblock**[$i+1$] = ⟨*value*⟩, **w**[$i$]**w**[$i+1$] = ⟨*value*⟩.
Constraint: if **iblock**[$i$] = **iblock**[$i+1$], **w**[$i$] $\leq$ **w**[$i+1$], for $i = 0, \ldots, \mathbf{m} - 2$.

**NE_CONVERGENCE**

⟨*value*⟩ eigenvectors (as indicated by argument **ifailv**) each failed to converge in 5 iterations. The current iterate after 5 iterations is stored in the corresponding column of **z**.

**NE_INT**

On entry, **m** = ⟨*value*⟩.
Constraint: $\max(1, \mathbf{m}) > 0$.

On entry, **n** = ⟨*value*⟩.
Constraint: **n** $\geq 0$.

On entry, **pdz** = ⟨*value*⟩.
Constraint: **pdz** $> 0$.

**NE_INT_2**

On entry, **m** = ⟨*value*⟩, **n** = ⟨*value*⟩.
Constraint: $0 \leq \mathbf{m} \leq \mathbf{n}$.

On entry, **pdz** = ⟨*value*⟩, **m** = ⟨*value*⟩.
Constraint: **pdz** $\geq \max(1, \mathbf{m})$.

On entry, **pdz** = ⟨*value*⟩, **n** = ⟨*value*⟩.
Constraint: **pdz** $\geq \max(1, \mathbf{n})$.

**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 consult NAG for assistance.

## 7 Accuracy

Each computed eigenvector $z_i$ is the exact eigenvector of a nearby matrix $A + E_i$, such that

$$\|E_i\| = O(\epsilon)\|A\|,$$

where $\epsilon$ is the ***machine precision***. Hence the residual is small:

$$\|Az_i - \lambda_i z_i\| = O(\epsilon)\|A\|.$$

However, a set of eigenvectors computed by this function may not be orthogonal to so high a degree of accuracy as those computed by nag_dsteqr (f08jec).

## 8 Further Comments

The complex analogue of this function is nag_zstein (f08jxc).

## 9 Example

See Section 9 of the document for nag_dormtr (f08fgc).