NAG Library Function Document

nag_zsteqr (f08jsc)

1 Purpose

nag_zsteqr (f08jsc) computes all the eigenvalues and, optionally, all the eigenvectors of a complex Hermitian matrix which has been reduced to tridiagonal form.

2 Specification

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

void nag_zsteqr (Nag_OrderType order, Nag_ComputeZType compz, Integer n, double d[], double e[], Complex z[], Integer pdz, NagError *fail)

3 Description

nag_zsteqr (f08jsc) computes all the eigenvalues and, optionally, all the eigenvectors of a real symmetric tridiagonal matrix $T$. In other words, it can compute the spectral factorization of $T$ as

$$T = Z \Lambda^T,$$

where $\Lambda$ is a diagonal matrix whose diagonal elements are the eigenvalues $\lambda_i$, and $Z$ is the orthogonal matrix whose columns are the eigenvectors $z_i$. Thus

$$Tz_i = \lambda_i z_i, \quad i = 1, 2, \ldots, n.$$ 

The function stores the real orthogonal matrix $Z$ in a complex array, so that it may also be used to compute all the eigenvalues and eigenvectors of a complex Hermitian matrix $A$ which has been reduced to tridiagonal form $T$:

$$A = QTQ^H,$$

where $Q$ is unitary

$$= (QZ)\Lambda(QZ)^H.$$ 

In this case, the matrix $Q$ must be formed explicitly and passed to nag_zsteqr (f08jsc), which must be called with compz = Nag_UpdateZ. The functions which must be called to perform the reduction to tridiagonal form and form $Q$ are:

- full matrix: nag_zhetrd (f08fsc) and nag_zungtr (f08ftc)
- full matrix, packed storage: nag_zhptrd (f08gsc) and nag_zupgtr (f08gtc)
- band matrix: nag_zhbtrd (f08hsc) with vect = Nag_FormQ.

nag_zsteqr (f08jsc) uses the implicitly shifted $QR$ algorithm, switching between the $QR$ and $QL$ variants in order to handle graded matrices effectively (see Greenbaum and Dongarra (1980)). The eigenvectors are normalized so that $||z_i||_2 = 1$, but are determined only to within a complex factor of absolute value 1. If only the eigenvalues of $T$ are required, it is more efficient to call nag_dsterf (f08jfc) instead. If $T$ is positive definite, small eigenvalues can be computed more accurately by nag_zpteqr (f08juc).
4 References


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 3.2.1.3 in the Essential Introduction for a more detailed explanation of the use of this argument.

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

2:   **compz** – Nag_ComputeZType

*Input*

*On entry:* indicates whether the eigenvectors are to be computed.

**compz** = Nag_NotZ

Only the eigenvalues are computed (and the array **z** is not referenced).

**compz** = Nag_UpdateZ

The eigenvalues and eigenvectors of **A** are computed (and the array **z** must contain the matrix **Q** on entry).

**compz** = Nag_InitZ

The eigenvalues and eigenvectors of **T** are computed (and the array **z** is initialized by the function).

*Constraint:* **compz** = Nag_NotZ, Nag_UpdateZ or Nag_InitZ.

3:   **n** – Integer

*Input*

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

*Constraint:* **n** ≥ 0.

4:   **d**[dim] – double

*Input/Output*

*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**.

*On exit:* the **n** eigenvalues in ascending order, unless **fail.code** = NE_CONVERGENCE (in which case see Section 6).

5:   **e**[dim] – double

*Input/Output*

*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**.

*On exit:* **e** is overwritten.

6:   **z**[dim] – Complex

*Input/Output*

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

max(1, pdz × **n**) when **compz** = Nag_UpdateZ or Nag_InitZ;

1 when **compz** = Nag_NotZ.
The \((i, j)\)th element of the matrix \(Z\) is stored in
\[
Z[(j - 1) \times pdz + i - 1] \text{ when } \text{order} = \text{Nag\_ColMajor};
\]
\[
Z[(i - 1) \times pdz + j - 1] \text{ when } \text{order} = \text{Nag\_RowMajor}.
\]

On entry: if \(\text{compz} = \text{Nag\_UpdateZ}\), \(Z\) must contain the unitary matrix \(Q\) from the reduction to tridiagonal form.

If \(\text{compz} = \text{Nag\_InitZ}\), \(Z\) need not be set.

On exit: if \(\text{compz} = \text{Nag\_UpdateZ}\) or \(\text{Nag\_InitZ}\), the \(n\) required orthonormal eigenvectors stored as columns of \(Z\); the \(i\)th column corresponds to the \(i\)th eigenvalue, where \(i = 1, 2, \ldots, n\), unless \text{fail\_code} = \text{NE\_CONVERGENCE}.

If \(\text{compz} = \text{Nag\_NotZ}\), \(Z\) is not referenced.

7: \(pdz\) – Integer  \hspace{1cm} \text{Input}

On entry: the stride separating row or column elements (depending on the value of \text{order}) in the array \(Z\).

Constraints:
\[
\text{if } \text{compz} = \text{Nag\_UpdateZ} \text{ or } \text{Nag\_InitZ}, \text{ pdz} \geq \max(1, n);
\]
\[
\text{if } \text{compz} = \text{Nag\_NotZ}, \text{ pdz} \geq 1.
\]

8: \(\text{fail}\) – NagError * \hspace{1cm} \text{Input/Output}

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

\text{NE\_ALLOC\_FAIL}

Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

\text{NE\_BAD\_PARAM}

On entry, argument \langle\text{value}\rangle had an illegal value.

\text{NE\_CONVERGENCE}

The algorithm has failed to find all the eigenvalues after a total of \(30 \times n\) iterations. In this case, \(d\) and \(e\) contain on exit the diagonal and off-diagonal elements, respectively, of a tridiagonal matrix unitarily similar to \(T\). \langle\text{value}\rangle off-diagonal elements have not converged to zero.

\text{NE\_ENUM\_INT\_2}

On entry, \(\text{compz} = \langle\text{value}\rangle, \text{ pdz} = \langle\text{value}\rangle\) and \(n = \langle\text{value}\rangle\).
Constraint: if \(\text{compz} = \text{Nag\_UpdateZ}\) or \(\text{Nag\_InitZ}\), \(\text{pdz} \geq \max(1, n)\);
\[
\text{if } \text{compz} = \text{Nag\_NotZ}, \text{ pdz} \geq 1.
\]

\text{NE\_INT}

On entry, \(n = \langle\text{value}\rangle\).
Constraint: \(n \geq 0\).

On entry, \(\text{pdz} = \langle\text{value}\rangle\).
Constraint: \(\text{pdz} > 0\).

\text{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 the Essential Introduction 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 the Essential Introduction for further information.

### 7 Accuracy

The computed eigenvalues and eigenvectors are exact for a nearby matrix \((T + E)\), where

\[
\|E\|_2 = O(\epsilon)\|T\|_2,
\]

and \(\epsilon\) is the *machine precision*.

If \(\lambda_i\) is an exact eigenvalue and \(\tilde{\lambda}_i\) is the corresponding computed value, then

\[
|\tilde{\lambda}_i - \lambda_i| \leq c(n)\epsilon\|T\|_2,
\]

where \(c(n)\) is a modestly increasing function of \(n\).

If \(z_i\) is the corresponding exact eigenvector, and \(\tilde{z}_i\) is the corresponding computed eigenvector, then the angle \(\theta(\tilde{z}_i, z_i)\) between them is bounded as follows:

\[
\theta(\tilde{z}_i, z_i) \leq \frac{c(n)\epsilon\|T\|_2}{\min_{i \neq j}|\lambda_i - \lambda_j|}.
\]

Thus the accuracy of a computed eigenvector depends on the gap between its eigenvalue and all the other eigenvalues.

### 8 Parallelism and Performance

*nag_zsteqr* (f08jsc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

*nag_zsteqr* (f08jsc) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

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’ Note for your implementation for any additional implementation-specific information.

### 9 Further Comments

The total number of real floating-point operations is typically about \(24n^2\) if \(\text{compz} = \text{Nag\_NotZ}\) and about \(14n^3\) if \(\text{compz} = \text{Nag\_UpdateZ}\) or \(\text{Nag\_InitZ}\), but depends on how rapidly the algorithm converges. When \(\text{compz} = \text{Nag\_NotZ}\), the operations are all performed in scalar mode; the additional operations to compute the eigenvectors when \(\text{compz} = \text{Nag\_UpdateZ}\) or \(\text{Nag\_InitZ}\) can be vectorized and on some machines may be performed much faster.

The real analogue of this function is nag_dsteqr (f08jec).

### 10 Example

See Section 10 in nag_zungtr (f08ftc), nag_zupgtr (f08gtc) or nag_zhbtrd (f08hsc), which illustrate the use of this function to compute the eigenvalues and eigenvectors of a full or band Hermitian matrix.