NAG C 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

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

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

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 $T$:

$$A = QTQ^H, \quad \text{where } Q \text{ is unitary,}$$

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

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) + nag_zungtr (f08ftc)
- full matrix, packed storage: nag_zhptrd (f08gsc) + 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 andQL 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_dstef (f08jfc) instead. If $T$ is positive-definite, small eigenvalues can be computed more accurately by nag_zpteqr (f08juc).

4 References


5 Parameters

1: \texttt{order} – Nag_OrderType \hspace{1cm} \textit{Input}

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

\textit{Constraint}: \texttt{order = Nag_RowMajor} or \texttt{Nag_ColMajor}.

2: \texttt{compz} – Nag_ComputeZType \hspace{1cm} \textit{Input}

\textit{On entry}: indicates whether the eigenvectors are to be computed as follows:

- if \texttt{compz = Nag_NotZ}, only the eigenvalues are computed (and the array \texttt{z} is not referenced);
- if \texttt{compz = Nag_InitZ}, the eigenvalues and eigenvectors of \( T \) are computed (and the array \texttt{z} is initialised by the routine);
- if \texttt{compz = Nag_UpdateZ}, the eigenvalues and eigenvectors of \( A \) are computed (and the array \texttt{z} must contain the matrix \( Q \) on entry).

\textit{Constraint}: \texttt{compz = Nag_NotZ}, \texttt{Nag_UpdateZ} or \texttt{Nag_InitZ}.

3: \texttt{n} – Integer \hspace{1cm} \textit{Input}

\textit{On entry}: \( n \), the order of the matrix \( T \).

\textit{Constraint}: \( n \geq 0 \).

4: \texttt{d[dim]} – double \hspace{1cm} \textit{Input/Output}

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

\textit{On entry}: the diagonal elements of the tridiagonal matrix \( T \).

\textit{On exit}: the \( n \) eigenvalues in ascending order, unless \texttt{fail} > 0 (in which case see Section 6).

5: \texttt{e[dim]} – double \hspace{1cm} \textit{Input/Output}

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

\textit{On entry}: the off-diagonal elements of the tridiagonal matrix \( T \).

\textit{On exit}: the array is overwritten.

6: \texttt{z[dim]} – Complex \hspace{1cm} \textit{Input/Output}

\textit{Note}: the dimension, \texttt{dim}, of the array \texttt{z} must be at least

- \( \max(1, \texttt{pdz} \times n) \) when \texttt{compz = Nag_UpdateZ} or \texttt{Nag_InitZ};
- 1 when \texttt{compz = Nag_NotZ}.

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

\textit{On entry}: if \texttt{compz = Nag_UpdateZ}, \texttt{z} must contain the unitary matrix \( Q \) from the reduction to tridiagonal form. If \texttt{compz = Nag_InitZ}, \texttt{z} need not be set.

\textit{On exit}: if \texttt{compz = Nag_InitZ} or \texttt{Nag_UpdateZ}, 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 \texttt{fail} > 0.

\( z \) is not referenced if \texttt{compz = Nag_NotZ}.
7:  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 compz = Nag_UpdateZ or Nag_InitZ, pdz ≥ max(1, n);
if compz = Nag_NotZ, pdz ≥ 1.

8:  fail – NagError *

Output

The NAG error parameter (see the Essential Introduction).

6  Error Indicators and Warnings

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

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

NE_ENUM_INT_2

On entry, compz = ⟨value⟩, n = ⟨value⟩, pdz = ⟨value⟩.
Constraint: if compz = Nag_UpdateZ or Nag_InitZ, pdz ≥ max(1, n);
if compz = Nag_NotZ, pdz ≥ 1.

NE_CONVERGENCE
The algorithm has failed to find all the eigenvalues after a total of 30 × n iterations. In this case, d
and e contain the diagonal and off-diagonal elements, respectively, of a tridiagonal matrix
orthogonally similar to T. ⟨value⟩ off-diagonal elements have not converged to zero.

NE_ALLOC_FAIL
Memory allocation failed.

NE_BAD_PARAM
On entry, parameter ⟨value⟩ had an illegal value.

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

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

8 Further Comments

The total number of real floating-point operations is typically about $24n^2$ if `compz = Nag_NotZ` and about $14n^3$ if `compz = Nag_UpdateZ` or `Nag_InitZ`, but depends on how rapidly the algorithm converges. When `compz = Nag_NotZ`, the operations are all performed in scalar mode; the additional operations to compute the eigenvectors when `compz = Nag_UpdateZ` or `Nag_InitZ` can be vectorized and on some machines may be performed much faster.

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

9 Example

See Section 9 of the documents for `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.