NAG C Library Function Document

nag_dstevd (f08jcc)

1 Purpose

nag_dstevd (f08jcc) computes all the eigenvalues, and optionally all the eigenvectors, of a real symmetric
tridiagonal matrix. If the eigenvectors are requested, then it uses a divide and conquer algorithm to
compute eigenvalues and eigenvectors. However, if only eigenvalues are required, then it uses the
Pal–Walker–Kahan variant of the QL or QR algorithm.

2 Specification

void nag_dstevd (Nag_OrderType order, Nag_JobType job, Integer n, double d[], double e[], double z[],
                Integer pdz, NagError *fail)

3 Description

nag_dstevd (f08jcc) 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 $A$ 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

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

4 References

Baltimore

5 Parameters

1: $order$ – Nag_OrderType

On entry: the $order$ 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
$order = Nag_RowMajor$. See Section 2.2.1.4 of the Essential Introduction for a more detailed
explanation of the use of this parameter.

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

2: $job$ – Nag_JobType

On entry: indicates whether eigenvectors are computed as follows:

if $job = Nag_DoNothing$, only eigenvalues are computed;

if $job = Nag_EigVecs$, eigenvalues and eigenvectors are computed.

Constraint: $job = Nag_DoNothing$ or $Nag_EigVecs$.

3: $n$ – Integer

On entry: $n$, the order of the matrix $A$.

Constraint: $n \geq 0$. 

[NP3645/7]
4: \(d[\text{dim}]\) – double

**Input/Output**

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

On entry: the \(n\) diagonal elements of the tridiagonal matrix \(T\).

On exit: the eigenvalues of the matrix \(T\) in ascending order.

5: \(e[\text{dim}]\) – double

**Input/Output**

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

On entry: the \(n-1\) off-diagonal elements of the tridiagonal matrix \(T\). The \(n\)th element of this array is used as workspace.

On exit: the array is overwritten with intermediate results.

6: \(z[\text{dim}]\) – double

**Output**

Note: the dimension, \(\text{dim}\), of the array \(z\) must be at least
\(\max(1, \text{pdz} \times n)\) when \(\text{job} = \text{Nag EigVecs}\);
1 when \(\text{job} = \text{Nag DoNothing}\).

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

On exit: if \(\text{job} = \text{Nag EigVecs}\), \(z\) is overwritten by the orthogonal matrix \(Z\) which contains the eigenvectors of \(T\).

If \(\text{job} = \text{Nag DoNothing}\), \(z\) is not referenced.

7: \(\text{pdz}\) – Integer

**Input**

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

Constraints:

- if \(\text{job} = \text{Nag EigVecs}\), \(\text{pdz} \geq \max(1, n)\);
- if \(\text{job} = \text{Nag DoNothing}\), \(\text{pdz} \geq 1\).

8: \(\text{fail}\) – NagError *

**Output**

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

**NE_INT**

On entry, \(n = \langle\text{value}\rangle\).

Constraint: \(n \geq 0\).

On entry, \(\text{pdz} = \langle\text{value}\rangle\).

Constraint: \(\text{pdz} > 0\).

**NE_ENUM_INT_2**

On entry, \(\text{job} = \langle\text{value}\rangle\), \(n = \langle\text{value}\rangle\), \(\text{pdz} = \langle\text{value}\rangle\).

Constraint: if \(\text{job} = \text{Nag EigVecs}\), \(\text{pdz} \geq \max(1, n)\);
if \(\text{job} = \text{Nag DoNothing}\), \(\text{pdz} \geq 1\).

**NE_CONVERGENCE**

The algorithm failed to converge, \(\langle\text{value}\rangle\) elements of an intermediate tridiagonal form did not converge to zero.
NE_ALLOC_FAIL

Memory allocation failed.

NE_BAD_PARAM

On entry, parameter \( h \) value \( i \) 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 \textit{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 Further Comments

There is no complex analogue of this function.

9 Example

To compute all the eigenvalues and eigenvectors of the symmetric tridiagonal matrix \( T \), where

\[
T = \begin{pmatrix}
1.0 & 1.0 & 0.0 & 0.0 \\
1.0 & 4.0 & 1.0 & 0.0 \\
0.0 & 1.0 & 9.0 & 1.0 \\
0.0 & 0.0 & 1.0 & 16.0
\end{pmatrix}
\]

9.1 Program Text

/* nag_dstevd (f08jcc) Example Program. */
* * Copyright 2001 Numerical Algorithms Group. *
* * Mark 7, 2001. */
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf08.h>
#include <nagx04.h>
int main(void)
{ /* Scalars */
  Integer i, n, pdz, d_len, e_len;
  NagError fail;
  Nag_JobType job;
  Nag_OrderType order;
  /* Arrays */
  char job_char[2];
  double *z=0, *d=0, *e=0;

  #ifdef NAG_COLUMN_MAJOR
  order = Nag_ColMajor;
  #else
  order = Nag_RowMajor;
  #endif

  INIT_FAIL(fail);
  Vprintf("f08jcc Example Program Results\n\n");

  /* Skip heading in data file */
  Vscanf("%*[\n\] ");
  Vscanf("%ld%*[\n\] ", &n);
  pdz = n;
  d_len = n;
  e_len = n-1;

  /* Allocate memory */
  if ( !(z = NAG_ALLOC(n * n, double)) ||
      !(d = NAG_ALLOC(d_len, double)) ||
      !(e = NAG_ALLOC(e_len, double)) )
  {
    Vprintf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }

  /* Read T from data file */
  for (i = 0; i < d_len; ++i)
    Vscanf("%lf", &d[i]);
  for (i = 0; i < e_len; ++i)
    Vscanf("%lf", &e[i]);

  /* Read type of job to be performed */
  Vscanf("%*[\n\] ");
  Vscanf("%ls %*[\n\] ", job_char);
  if (*((unsigned char *)job_char == 'V')
    job = Nag_EigVecs;
  else
    job = Nag_DoNothing;

  /* Calculate all the eigenvalues and eigenvectors of T */
  f08jcc(order, job, n, d, e, z, pdz, &fail);
  if (fail.code != NE_NOERROR)
  {
    Vprintf("Error from f08jcc.\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }

  /* Print eigenvalues and eigenvectors */
  Vprintf(" Eigenvalues\n");
  for (i = 0; i < n; ++i)
    Vprintf(" %7.4lf", d[i]);
  Vprintf("\n\n");
  x04cac(order, Nag_GeneralMatrix, Nag_NonUnitDiag, n, n,
         z, pdz, "Eigenvectors", 0, &fail);
  if (fail.code != NE_NOERROR)
  {
    Vprintf("Error from x04cac.\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }

  END:
  if (d) NAG_FREE(d);
if (e) NAG_FREE(e);
if (z) NAG_FREE(z);
return exit_status;
}

9.2 Program Data
f08jcc Example Program Data
4 :Value of N
  1.0  4.0  9.0  16.0 :End of T
  1.0  2.0  3.0 :Value of JOB

9.3 Program Results
f08jcc Example Program Results

Eigenvalues
  0.6476  3.5470  8.6578  17.1477

Eigenvectors
1     2     3     4
1  0.9396  0.3388  0.0494  0.0034
2 -0.3311  0.8628  0.3781  0.0545
3  0.0853 -0.3648  0.8558  0.3568
4 -0.0167  0.0879 -0.3497  0.9326