**NAG C Library Function Document**

**nag_dsytrd (f08fec)**

1 Purpose

`nag_dsytrd (f08fec)` reduces a real symmetric matrix to tridiagonal form.

2 Specification

```c
void nag_dsytrd (Nag_OrderType order, Nag_UploType uplo, Integer n, double a[],
  Integer pda, double d[], double e[], double tau[], NagError *fail)
```

3 Description

`nag_dsytrd (f08fec)` reduces a real symmetric matrix $A$ to symmetric tridiagonal form $T$ by an orthogonal similarity transformation: $A = QTQ^T$.

The matrix $Q$ is not formed explicitly but is represented as a product of $n - 1$ elementary reflectors (see the f08 Chapter Introduction for details). Functions are provided to work with $Q$ in this representation (see Section 8).

4 References


5 Parameters

1. `order` – Nag_OrderType
   - **Input**
   - 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. `uplo` – Nag_UploType
   - **Input**
   - On entry: indicates whether the upper or lower triangular part of $A$ is stored as follows:
     - If `uplo` = Nag_Upper, the upper triangular part of $A$ is stored;
     - If `uplo` = Nag_Lower, the lower triangular part of $A$ is stored.
   - **Constraint**: `uplo` = Nag_Upper or Nag_Lower.

3. `n` – Integer
   - **Input**
   - On entry: $n$, the order of the matrix $A$.
   - **Constraint**: $n \geq 0$.

4. `a[dim]` – double
   - **Input/Output**
   - **Note**: the dimension, `dim`, of the array `a` must be at least max(1, `pda` × `n`).
   - If `order` = Nag_ColMajor, the $(i,j)$th element of the matrix $A$ is stored in `a[(j-1) × pda + i - 1]` and if `order` = Nag_RowMajor, the $(i,j)$th element of the matrix $A$ is stored in `a[(i-1) × pda + j - 1]`.

[f08 fec.1]
On entry: the \( n \) by \( n \) symmetric matrix \( A \). If \texttt{uplo = Nag\_Upper}, the upper triangular part of \( A \) must be stored and the elements of the array below the diagonal are not referenced; if \texttt{uplo = Nag\_Lower}, the lower triangular part of \( A \) must be stored and the elements of the array above the diagonal are not referenced.

On exit: \( a \) is overwritten by the tridiagonal matrix \( T \) and details of the orthogonal matrix \( Q \) as specified by \texttt{uplo}.

5: \( \text{pda} \) – Integer \hspace{1cm} \text{Input}

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

Constraint: \( \text{pda} \geq \max(1, n) \).

6: \( \text{d}[\text{dim}] \) – double \hspace{1cm} \text{Output}

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

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

7: \( \text{e}[\text{dim}] \) – double \hspace{1cm} \text{Output}

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

On exit: the off-diagonal elements of the tridiagonal matrix \( T \).

8: \( \text{tau}[\text{dim}] \) – double \hspace{1cm} \text{Output}

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

On exit: further details of the orthogonal matrix \( Q \).

9: \( \text{fail} \) – NagError * \hspace{1cm} \text{Output}

The NAG error parameter (see the Essential Introduction).

6 \hspace{1cm} \textbf{Error Indicators and Warnings}

**NE\_INT**

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

Constraint: \( n \geq 0 \).

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

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

**NE\_INT\_2**

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

Constraint: \( \text{pda} \geq \max(1, n) \).

**NE\_ALLOC\_FAIL**

Memory allocation failed.

**NE\_BAD\_PARAM**

On entry, parameter \( \langle \text{value} \rangle \) 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 tridiagonal matrix $T$ is exactly similar to a nearby matrix $A + E$, where
\[
\|E\|_2 \leq c(n)\epsilon\|A\|_2,
\]
c($n$) is a modestly increasing function of $n$, and $\epsilon$ is the machine precision.

The elements of $T$ themselves may be sensitive to small perturbations in $A$ or to rounding errors in the computation, but this does not affect the stability of the eigenvalues and eigenvectors.

8 Further Comments

The total number of floating-point operations is approximately $\frac{4}{3}n^3$.

To form the orthogonal matrix $Q$ this function may be followed by a call to nag_dorgtr (f08ffc):
\[
\text{nag_dorgtr (order, uplo, n, a, pda, tau, &fail)}
\]
To apply $Q$ to an $n$ by $p$ real matrix $C$ this function may be followed by a call to nag_dormtr (f08fgc).

For example,
\[
\text{nag_dormtr (order, Nag_LeftSide, uplo, Nag_NoTrans, n, p, a, pda, tau, &c, pdc, &fail)}
\]
forms the matrix product $QC$.

The complex analogue of this function is nag_zhetrd (f08fsc).

9 Example

To reduce the matrix $A$ to tridiagonal form, where
\[
A = \begin{pmatrix}
2.07 & 3.87 & 4.20 & -1.15 \\
3.87 & -0.21 & 1.87 & 0.63 \\
4.20 & 1.87 & 1.15 & 2.06 \\
-1.15 & 0.63 & 2.06 & -1.81
\end{pmatrix}
\]

9.1 Program Text

/* nag_dsytrd (f08fec) Example Program. 
 * Copyright 2001 Numerical Algorithms Group. 
 * Mark 7, 2001. */

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

int main(void)
{
    /* Scalars */
    Integer i, j, n, pda, d_len, e_len, tau_len;
    Integer exit_status=0;
    NagError fail;
    Nag_UploType uplo;
    Nag_OrderType order;
    /* Arrays */
    char uplo_char[2];
    double *a=0, *d=0, *e=0, *tau=0;
    #ifdef NAG_COLUMN_MAJOR
    #define A(I,J) a[(J-1)*pda + I - 1]
    order = Nag_ColMajor;
    #else
    #define A(I,J) a[(I-1)*pda + J - 1]
    order = Nag_RowMajor;
    #endif

    /* Main body of program */
    ...

}
order = Nag_RowMajor;
#endif

INIT_FAIL(fail);
Vprintf("f08fec Example Program Results\n");

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

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

/* Read A from data file */
Vscanf(" ' %1s '%*[\n"]", uplo_char);
if (*(unsigned char *)uplo_char == 'L')
    uplo = Nag_Lower;
else if (*(unsigned char *)uplo_char == 'U')
    uplo = Nag_Upper;
else
{
    Vprintf("Unrecognised character for Nag_UploType type\n");
    exit_status = -1;
    goto END;
}
if (uplo == Nag_Upper)
{
    for (i = 1; i <= n; ++i)
    {
        for (j = i; j <= n; ++j)
            Vscanf("%lf", &A(i,j));
    }
    Vscanf("%*[\n"]");
}
else
{
    for (i = 1; i <= n; ++i)
    {
        for (j = 1; j <= i; ++j)
            Vscanf("%lf", &A(i,j));
    }
    Vscanf("%*[\n"]");
}

/* Reduce A to tridiagonal form */
f08fec(order, uplo, n, a, pda, d, e, tau, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from f08fec.\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Print tridiagonal form */
Vprintf("\nnDiagonal\n")
for (i = 1; i <= n; ++i)
    Vprintf("%9.4f", d[i-1], i%8==0 ?"\n": "");
Vprintf("\nOff-diagonal\n")
for (i = 1; i <= n - 1; ++i)
Vprintf("%9.4f%s", e[i-1], i%8==0 ?"\n":" ");
Vprintf("\n");
END:
if (a) NAG_FREE(a);
if (d) NAG_FREE(d);
if (e) NAG_FREE(e);
if (tau) NAG_FREE(tau);
return exit_status;
}

9.2 Program Data
f08fec Example Program Data
4 :Value of N
'L' :Value of UPLO
2.07
3.87 -0.21
4.20 1.87 1.15
-1.15 0.63 2.06 -1.81 :End of matrix A

9.3 Program Results
f08fec Example Program Results
Diagonal
2.0700 1.4741 -0.6492 -1.6949
Off-diagonal
-5.8258 2.6240 0.9163