NAG Library Function Document

nag_dgehrd (f08nec)

1 Purpose

nag_dgehrd (f08nec) reduces a real general matrix to Hessenberg form.

2 Specification

```c
#include <nag.h>
#include <nagf08.h>
void nag_dgehrd (Nag_OrderType order, Integer n, Integer ilo, Integer ihi,
                  double a[], Integer pda, double tau[], NagError *fail)
```

3 Description

nag_dgehrd (f08nec) reduces a real general matrix $A$ to upper Hessenberg form $H$ by an orthogonal similarity transformation: $A = QHQ^T$.

The matrix $Q$ is not formed explicitly, but is represented as a product of elementary reflectors (see the f08 Chapter Introduction for details). Functions are provided to work with $Q$ in this representation (see Section 9).

The function can take advantage of a previous call to nag_dgebal (f08nhc), which may produce a matrix with the structure:

\[
\begin{pmatrix}
A_{11} & A_{12} & A_{13} \\
A_{22} & A_{23} & \\
A_{33} & & \\
\end{pmatrix}
\]

where $A_{11}$ and $A_{33}$ are upper triangular. If so, only the central diagonal block $A_{22}$, in rows and columns $i_{lo}$ to $i_{hi}$, needs to be reduced to Hessenberg form (the blocks $A_{12}$ and $A_{23}$ will also be affected by the reduction). Therefore the values of $i_{lo}$ and $i_{hi}$ determined by nag_dgebal (f08nhc) can be supplied to the function directly. If nag_dgebal (f08nhc) has not previously been called however, then $i_{lo}$ must be set to 1 and $i_{hi}$ to $n$.

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:   **n**  --  Integer  

   *Input*

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

   **Constraint:** $n \geq 0$. 

Mark 25
3:  ilo – Integer  
Input

4:  ihi – Integer  
Input

On entry: if $A$ has been output by nag_dgebal (f08nhc), then ilo and ihi must contain the values returned by that function. Otherwise, ilo must be set to 1 and ihi to $n$.

Constraints:

if $n > 0$, $1 \leq ilo \leq ihi \leq n$;
if $n = 0$, ilo = 1 and ihi = 0.

5:  $a[dim]$ – double  
Input/Output

Note: the dimension, dim, of the array $a$ must be at least max(1, pda $\times n$).

The $(i,j)$th element of the matrix $A$ is stored in

$a[(j-1) \times pda + i - 1]$ when order = Nag_ColMajor;

$a[(i-1) \times pda + j - 1]$ when order = Nag_RowMajor.

On entry: the $n$ by $n$ general matrix $A$.

On exit: $a$ is overwritten by the upper Hessenberg matrix $H$ and details of the orthogonal matrix $Q$.

6:  pda – Integer  
Input

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

Constraint: $pda \geq \max(1, n)$.

7:  $tau[dim]$ – double  
Output

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

On exit: further details of the orthogonal matrix $Q$.

8:  fail – NagError *  
Input/Output

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

6   Error Indicators and Warnings

NE_ALLOC_FAIL

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

NE_BAD_PARAM

On entry, argument $\langle value \rangle$ had an illegal value.

NE_INT

On entry, $n = \langle value \rangle$.
Constraint: $n \geq 0$.

On entry, $pda = \langle value \rangle$.
Constraint: $pda > 0$.

NE_INT_2

On entry, $pda = \langle value \rangle$ and $n = \langle value \rangle$.
Constraint: $pda \geq \max(1, n)$.
NE_INT_3
On entry, \( n = \langle \text{value} \rangle, \ ilo = \langle \text{value} \rangle \) and \( ihi = \langle \text{value} \rangle \).
Constraint: if \( n > 0 \), \( 1 \leq ilo \leq ihi \leq n \);
if \( n = 0 \), \( ilo = 1 \) and \( ihi = 0 \).

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 Hessenberg matrix \( H \) 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 \( H \) 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, eigenvectors or Schur factorization.

8 Parallelism and Performance
nag_dgehrd (f08nec) is not threaded by NAG in any implementation.
nag_dgehrd (f08nec) 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 floating-point operations is approximately \( \frac{2}{3}q^2(2q + 3n) \), where \( q = i_{hi} - i_{lo} \); if
\( i_{lo} = 1 \) and \( i_{hi} = n \), the number is approximately \( 10n^3 \).
To form the orthogonal matrix \( Q \) nag_dgehrd (f08nec) may be followed by a call to nag_dorgrhr (f08nfc):
\[
nag_dorgrhr(order,n,ilo,ihi,\&a,pda,\tau,\&fail)
\]
To apply \( Q \) to an \( m \) by \( n \) real matrix \( C \) nag_dgehrd (f08nec) may be followed by a call to nag_dormhr
(f08nge).

Forms the matrix product \( QC \).
The complex analogue of this function is nag_zgehrd (f08nsc).
10 Example

This example computes the upper Hessenberg form of the matrix \( A \), where

\[
A = \begin{pmatrix}
0.35 & 0.45 & -0.14 & -0.17 \\
0.09 & 0.07 & -0.54 & 0.35 \\
-0.44 & -0.33 & -0.03 & 0.17 \\
0.25 & -0.32 & -0.13 & 0.11 \\
\end{pmatrix}
\]

10.1 Program Text

/* nag_dgehrd (f08nec) Example Program. *
 * Copyright 2014 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, j, n, pda, tau_len;
    Integer exit_status = 0;
    NagError fail;
    Nag_OrderType order;
    /* Arrays */
    double *a = 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
    INIT_FAIL(fail);

    printf("nag_dgehrd (f08nec) Example Program Results\n\n");

    /* Skip heading in data file */
    #ifdef _WIN32
    scanf_s("%*[^\n] ");
    #else
    scanf("%*[^\n] ");
    #endif
    #ifdef _WIN32
    scanf("%"NAG_IFMT"%*[^\n] ", &n);
    #else
    scanf("%"NAG_IFMT"%*[^\n] ", &n);
    #endif
    pda = n;
    tau_len = n - 1;

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

    /* Compute Hessenberg form */
    f08nec...

    /* Print results */
    printf("Results:\n\n");
    ...
/* Read A from data file */
for (i = 1; i <= n; ++i)
{
    for (j = 1; j <= n; ++j)
        #ifdef _WIN32
            scanf_s("%lf", &A(i, j));
        #else
            scanf("%lf", &A(i, j));
        #endif
    #ifdef _WIN32
        scanf_s("%*[\n ] ");
    #else
        scanf("%*[\n ] ");
    #endif
}

/* nag_dgehrd (f08nec): Reduce A to upper Hessenberg form */
nag_dgehrd(order, n, 1, n, a, pda, tau, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_dgehrd (f08nec).\n\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Set the elements below the first sub-diagonal to zero */
for (i = 1; i <= n - 2; ++i)
{
    for (j = i + 2; j <= n; ++j)
        A(j, i) = 0.0;
}

/* nag_gen_real_mat_print (x04cac): Print upper Hessenberg form. */
fflush(stdout);
nag_gen_real_mat_print(order, Nag_GeneralMatrix, Nag_NonUnitDiag, n, n, a, pda, "Upper Hessenberg form", 0, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_gen_real_mat_print (x04cac).\n\n", fail.message);
    exit_status = 1;
    goto END;
}
END:
NAG_FREE(a);
NAG_FREE(tau);
return exit_status;
#endif A

10.2 Program Data

nag_dgehrd (f08nec) Example Program Data

Value of N
4
0.35 0.45 -0.14 -0.17
0.09 0.07 -0.54 0.35
-0.44 -0.33 -0.03 0.17
0.25 -0.32 -0.13 0.11

End of matrix A
## 10.3 Program Results

nag_dgehrd (f08nec) Example Program Results

Upper Hessenberg form

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.3500</td>
<td>-0.1160</td>
<td>-0.3886</td>
<td>-0.2942</td>
</tr>
<tr>
<td>2</td>
<td>-0.5140</td>
<td>0.1225</td>
<td>0.1004</td>
<td>0.1126</td>
</tr>
<tr>
<td>3</td>
<td>0.0000</td>
<td>0.6443</td>
<td>-0.1357</td>
<td>-0.0977</td>
</tr>
<tr>
<td>4</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.4262</td>
<td>0.1632</td>
</tr>
</tbody>
</table>