NAG Library Function Document
nag_zgehrd (f08nsc)

1 Purpose
nag_zgehrd (f08nsc) reduces a complex general matrix to Hessenberg form.

2 Specification
#include <nag.h>
#include <nagf08.h>
void nag_zgehrd (Nag_OrderType order, Integer n, Integer ilo, Integer ihi,
Complex a[], Integer pda, Complex tau[], NagError *fail)

3 Description
nag_zgehrd (f08nsc) reduces a complex general matrix $A$ to upper Hessenberg form $H$ by a unitary similarity transformation: $A = QHQ^H$. $H$ has real subdiagonal elements.

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_zgebal (f08nvc), which may produce a matrix with the structure:

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

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_zgebal (f08nvc) can be supplied to the function directly. If nag_zgebal (f08nvc) 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: \text{ilo} – Integer \hspace{1cm} \textit{Input}
4: \text{ihi} – Integer \hspace{1cm} \textit{Input}

\textit{On entry:} if \( A \) has been output by \texttt{nag_zgebal (f08nvc)}, then \texttt{ilo} and \texttt{ihi} must contain the values returned by that function. Otherwise, \texttt{ilo} must be set to 1 and \texttt{ihi} to \( n \).

\textit{Constraints:}
\begin{align*}
\text{if } n > 0, & \quad 1 \leq \texttt{ilo} \leq \texttt{ihi} \leq n; \\
\text{if } n = 0, & \quad \texttt{ilo} = 1 \text{ and } \texttt{ihi} = 0.
\end{align*}

5: \( a[\text{dim}] \) – Complex \hspace{1cm} \textit{Input/Output}

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

The \((i,j)\)th element of the matrix \( A \) is stored in
\begin{align*}
\texttt{a}[(j-1) \times \texttt{pda} + i - 1] & \text{ when } \texttt{order} = \texttt{Nag\_ColMajor}; \\
\texttt{a}[(i-1) \times \texttt{pda} + j - 1] & \text{ when } \texttt{order} = \texttt{Nag\_RowMajor}.
\end{align*}

\textit{On entry:} the \( n \) by \( n \) general matrix \( A \).

\textit{On exit:} \( a \) is overwritten by the upper Hessenberg matrix \( H \) and details of the unitary matrix \( Q \). The subdiagonal elements of \( H \) are real.

6: \texttt{pda} – Integer \hspace{1cm} \textit{Input}

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

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

7: \( \texttt{tau}[\text{dim}] \) – Complex \hspace{1cm} \textit{Output}

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

\textit{On exit:} further details of the unitary matrix \( Q \).

8: \texttt{fail} – \texttt{NagError} * \hspace{1cm} \textit{Input/Output}

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

6 Error Indicators and Warnings

\textbf{NE_ALLOC_FAIL}

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

\textbf{NE_BAD_PARAM}

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

\textbf{NE_INT}

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

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

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

\textit{Constraint:} \( \texttt{pda} > 0 \).

\textbf{NE_INT_2}

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

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

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_zgehrd (f08nsc) is not threaded by NAG in any implementation.
nag_zgehrd (f08nsc) 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 approximately $\frac{8}{3}q(2q + 3n)$, where $q = \text{ihi} - \text{ilo}$; if
$\text{ilo} = 1$ and $\text{ihi} = n$, the number is approximately $\frac{40}{3}n^3$.

To form the unitary matrix $Q$ nag_zgehrd (f08nsc) may be followed by a call to nag_zunghr (f08ntc):
\[ \text{nag_zunghr(order, n, ilo, ihi, &a, pda, tau, &fail)} \]
To apply $Q$ to an $m$ by $n$ complex matrix $C$ nag_zgehrd (f08nsc) may be followed by a call to
nag_zunmhr (f08nuc). For example,
\[ \text{nag_zunmhr(order, Nag_LeftSide, Nag_NoTrans, m, n, ilo, ihi, &a, pda, tau, &c, pdc, &fail)} \]
forms the matrix product $QC$.
The real analogue of this function is nag_dgehrd (f08nec).
10 Example

This example computes the upper Hessenberg form of the matrix $A$, where

$$A = \begin{pmatrix}
-3.97 - 5.04i & -4.11 + 3.70i & -0.34 + 1.01i & 1.29 - 0.86i \\
0.34 - 1.50i & 1.52 - 0.43i & 1.88 - 5.38i & 3.36 + 0.65i \\
3.31 - 3.85i & 2.50 + 3.45i & 0.88 - 1.08i & 0.64 - 1.48i \\
-1.10 + 0.82i & 1.81 - 1.59i & 3.25 + 1.33i & 1.57 - 3.44i
\end{pmatrix}.$$ 

10.1 Program Text

/* nag_zgehrd (f08nsc) Example Program. */
* Copyright 2014 Numerical Algorithms Group.
*/

#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 */
    Complex *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_zgehrd (f08nsc) 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
    #ifdef NAG_COLUMN_MAJOR
    pda = n;
    #else
    pda = n;
    #endif
    tau_len = n - 1;

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

/* Read A from data file */
for (i = 1; i <= n; ++i)
{
    for (j = 1; j <= n; ++j)
#ifndef _WIN32
    scanf_s(" ( %lf , %lf )", &A(i, j).re, &A(i, j).im);
#else
    scanf(" ( %lf , %lf )", &A(i, j).re, &A(i, j).im);
#endif
#endif _WIN32
    scanf("%*\[^
"");
#endif
    scanf("%*[\n] ");
}

/* Reduce A to upper Hessenberg form */
/* nag_zgehrd (f08nsc). * Unitary reduction of complex general matrix to upper * Hessenberg form */
nag_zgehrd(order, n, 1, n, a, pda, tau, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_zgehrd (f08nsc).\n%s\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).re = 0.0, A(j, i).im = 0.0;
}

/* Print upper Hessenberg form */
/* nag_gen_complex_mat_print_comp (x04dbc). * Print complex general matrix (comprehensive) */
fflush(stdout);
nag_gen_complex_mat_print_comp(order, Nag_GeneralMatrix, Nag_NonUnitDiag, n, n, a, pda, Nag_BracketForm, "%7.4f", "Upper Hessenberg form", Nag_IntegerLabels, 0, Nag_IntegerLabels, 0, 80, 0, 0, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_gen_complex_mat_print_comp (x04dbc).\n%s\n", fail.message);
    exit_status = 1;
goto END;
}
#endif
END:
NAG_FREE(a);
NAG_FREE(tau);
return exit_status;
}
10.2 Program Data

nag_zgehrd (f08nsc) Example Program Data

4 :Value of N

(-3.97,-5.04) (-4.11, 3.70) (-0.34, 1.01) ( 1.29,-0.86)
( 0.34,-1.50) ( 1.52,-0.43) ( 1.88,-5.38) ( 3.36, 0.65)
( 3.31,-3.85) ( 2.50, 3.45) ( 0.88,-1.08) ( 0.64,-1.48)
(-1.10, 0.82) ( 1.81,-1.59) ( 3.25, 1.33) ( 1.57,-3.44) :End of matrix A

10.3 Program Results

nag_zgehrd (f08nsc) 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>(-3.9700, -5.0400)</td>
<td>(-1.1318, -2.5693)</td>
<td>(-4.6027, -0.1426)</td>
<td>(-1.4249, 1.7330)</td>
</tr>
<tr>
<td>2</td>
<td>(-5.4797, 0.0000)</td>
<td>(1.8585, -1.5502)</td>
<td>(4.4145, -0.7638)</td>
<td>(-0.4805, -1.1976)</td>
</tr>
<tr>
<td>3</td>
<td>(0.0000, 0.0000)</td>
<td>(6.2673, 0.0000)</td>
<td>(-0.4504, 0.0290)</td>
<td>(-1.3467, 1.6579)</td>
</tr>
<tr>
<td>4</td>
<td>(0.0000, 0.0000)</td>
<td>(0.0000, 0.0000)</td>
<td>(-3.5000, 0.0000)</td>
<td>(2.5619, -3.3708)</td>
</tr>
</tbody>
</table>