nag_complex_form_q (f01rec)

1. Purpose

nag_complex_form_q (f01rec) returns the first ncolq columns of the m by m unitary matrix Q, where Q is given as the product of Householder transformation matrices.

This function is intended for use following nag_complex_qr (f01rcc).

2. Specification

```c
#include <nag.h>
#include <nagf01.h>

void nag_complex_form_q(Nag_WhereElements wheret, Integer m, Integer n,
                        Integer ncolq, Complex a[], Integer tda, Complex theta[], NagError *fail)
```

3. Description

The unitary matrix Q is assumed to be given by

\[ Q = (Q_nQ_{n-1}\ldots Q_1)^H, \]

\[ Q_k \text{ being given in the form} \]

\[ Q_k = \begin{pmatrix} I & 0 \\ 0 & T_k \end{pmatrix}, \]

where

\[ T_k = I - \gamma_k u_k u_k^H \]
\[ u_k = \begin{pmatrix} \zeta_k \\ z_k \end{pmatrix}, \]

\( \gamma_k \) is a scalar for which \( \text{Re}\, \gamma_k = 1.0 \), \( \zeta_k \) is a real scalar and \( z_k \) is an \((m - k)\) element vector.

\( z_k \) must be supplied in the \((k - 1)\)th column of \( a \) in elements \( a[k][k-1], \ldots, a[m-1][k-1] \) and \( \theta_k \), given by

\[ \theta_k = (\zeta_k, \text{Im}\, \gamma_k), \]

must be supplied either in \( a[k-1][k-1] \) or in \( \text{theta}[k-1] \) depending upon the parameter \( \text{wheret} \).

4. Parameters

**wheret**

Input: the elements of \( \theta \) are to be found as follows:

- \( \text{wheret} = \text{Nag\_ElementsIn} \), the elements of \( \theta \) are in \( A \).
- \( \text{wheret} = \text{Nag\_ElementsSeparate} \), the elements of \( \theta \) are separate from \( A \), in \( \text{theta} \).

Constraint: \( \text{wheret} \) must be one of \( \text{Nag\_ElementsIn} \) or \( \text{Nag\_ElementsSeparate} \).

**m**

Input: \( m \), the number of rows of \( A \).
Constraint: \( m \geq n \).

**n**

Input: \( n \), the number of columns of \( A \).
Constraint: \( n \geq 0 \).

**ncolq**

Input: \( \text{ncolq} \), the required number of columns of \( Q \).
When \( \text{ncolq} = 0 \) then an immediate return is effected.
Constraint: \( 0 \leq \text{ncolq} \leq m \).
nag_complex_form_q

\[ a[m][tda] \]
Input: the leading \( m \) by \( n \) strictly lower triangular part of the array \( a \) must contain details of the matrix \( Q \). In addition, when \( \text{where} = \text{Nag\_ElementsIn} \), then the diagonal elements of \( a \) must contain the elements of \( \theta \) as described under the parameter \( \text{theta} \) below.

Output: the first \( n\text{colm} \) columns of the array \( a \) are overwritten by the first \( n\text{colm} \) columns of the \( m \) by \( m \) unitary matrix \( Q \). When \( n = 0 \) then the first \( n\text{colm} \) columns of \( a \) are overwritten by the first \( n\text{colm} \) columns of the unit matrix.

\[ tda \]
Input: the second dimension of the array \( a \) as declared in the function from which \( \text{nag\_complex\_form\_q} \) is called.
Constraint: \( tda \geq \max(n, n\text{colm}) \).

\[ \text{theta}[n] \]
Input: if \( \text{where} = \text{Nag\_ElementsSeparate} \), the array \( \text{theta} \) must contain the elements of \( \theta \). If \( \text{theta}[k-1] = 0.0 \) then \( T_k \) is assumed to be \( I \); if \( \text{theta}[k-1] = \alpha \), with \( \text{Re} \alpha < 0.0 \), then \( T_k \) is assumed to be of the form

\[
T_k = \begin{pmatrix} \alpha & 0 \\ 0 & I \end{pmatrix};
\]

otherwise \( \text{theta}[k-1] \) is assumed to contain \( \theta_k \) given by \( \theta_k = (\zeta_k, \text{Im} \gamma_k) \).
When \( \text{where} = \text{Nag\_ElementsIn} \), the array \( \text{theta} \) is not referenced and may be set to the null pointer, i.e., \((\text{Complex} *)0\).

\[ \text{fail} \]
The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

**NE\_BAD\_PARAM**
On entry, parameter \( \text{where} \) had an illegal value.

**NE\_2\_INT\_ARG\_LT**
On entry, \( m = \langle \text{value} \rangle \) while \( n = \langle \text{value} \rangle \). These parameters must satisfy \( m \geq n \).
On entry, \( tda = \langle \text{value} \rangle \) while \( \max(n, n\text{colm}) = \langle \text{value} \rangle \). These parameters must satisfy \( tda \geq n \).

**NE\_INT\_ARG\_LT**
On entry, \( n \) must not be less than 0: \( n = \langle \text{value} \rangle \).
On entry, \( n\text{colm} \) must not be less than 0: \( n\text{colm} = \langle \text{value} \rangle \).

**NE\_2\_INT\_ARG\_GT**
On entry, \( n\text{colm} = \langle \text{value} \rangle \) while \( m = \langle \text{value} \rangle \). These parameters must satisfy \( n\text{colm} \leq m \).

**NE\_ALLOC\_FAIL**
Memory allocation failed.

6. Further Comments

The approximate number of real floating-point operations required is given by

\[
\frac{2}{3} n(3m - n)(2n\text{colm} - n) - n(n\text{colm} - n) \quad n\text{colm} > n;
\]

\[
\frac{2}{3} n\text{colm}^2(3m - n\text{colm}) \quad n\text{colm} \leq n.
\]

6.1. Accuracy

The computed matrix \( Q \) satisfies the relation

\[ Q = P + E, \]
where \( P \) is an exactly unitary matrix and

\[ \|E\| \leq c\epsilon, \]
\( \epsilon \) being the **machine precision**, \( c \) is a modest function of \( m \) and \( \|\cdot\| \) denotes the spectral (two) norm. See also Section 6.1 of nag\_complex\_qr (f01rec).
6.2. References


7. See Also

nag_complex_qr (f01rec)
nag_complex_apply_q (f01rdc)

8. Example

To obtain the 5 by 5 unitary matrix $Q$ following the $QR$ factorization of the 5 by 3 matrix $A$ given by

$$A = \begin{pmatrix}
0.5i & -0.5 + 1.5i & -1.0 + 1.4i \\
0.4 + 0.3i & 0.9 + 1.3i & 0.2 + 1.4i \\
0.4 & -0.4 + 0.4i & 1.8 \\
0.3 - 0.4i & 0.1 + 0.7i & 0.0 \\
-0.3i & 0.3 + 0.3i & 2.4i
\end{pmatrix}.$$ 

8.1. Program Text

/* nag_complex_form_q (f01rec) Example Program */
/* Copyright 1990 Numerical Algorithms Group. */
/* Mark 1, 1990. */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagf01.h>

#define MMAX 20
#define NMAX 10
#define TDA NMAX
#define TDQ MMAX
#define COMPLEX(A) A.re, A.im

main()
{
    Integer i, j, m, n, ncolq;
    Complex a[MMAX][TDA], q[MMAX][TDQ], theta[NMAX];
    static NagError fail;
    Vprintf("f01rec Example Program Results\n");
    /* Skip heading in data file */
    Vscanf("%*[\n]");
    Vscanf("%ld%ld", &m, &n);
    if (m>MMAX || n>NMAX)
    {
        Vfprintf(stderr,"m or n is out of range. m = %ld, n = %ld\n", m, n);
        exit (EXIT_FAILURE);
    }
    for (i=0; i<m; ++i)
        for(j=0; j<n; ++j)
            Vscanf(" ( %lf , %lf ) ", COMPLEX(&a[i][j]));
    fail.print = TRUE;
    /* Find the QR factorization of A. */
    f01rcc(m, n, (Complex *)a, (Integer)TDA, theta, &fail);
    if (fail.code != NE_NOERROR)
        exit(EXIT_FAILURE);
    /* Copy the array A into Q and form the m by m matrix Q. */
    for (j=0; j<n; ++j)
        for (i=0; i<m; ++i)
            q[i][j].re = a[i][j].re, q[i][j].im = a[i][j].im;

}
ncolq = m;
f01rec(Nag_ElementsSeparate, m, n, ncolq, (Complex *)q, (Integer)TDQ, theta, &fail);
if (fail.code != NE_NOERROR)
    exit(EXIT_FAILURE);
Vprintf("\nMatrix Q\n");
for (i=0; i<m; ++i)
    {
        for (j=0; j<ncolq; ++j)
            Vprintf(" (%5.2f,%5.2f)%s", COMPLEX(q[i][j]),
                (j%5==4 || j==ncolq-1) ? "\n" : " ");
    }
exit (EXIT_SUCCESS);

8.2. Program Data

f01rec Example Program Data

5 3

( 0.0, 0.5 ) ( -0.5, 1.5 ) ( -1.0, 1.0 )
( 0.4, 0.3 ) ( 0.9, 1.3 ) ( 0.2, 1.4 )
( 0.4, 0.0 ) ( -0.4, 0.4 ) ( 1.8, 0.0 )
( 0.3, -0.4 ) ( 0.1, 0.7 ) ( 0.0, 0.0 )
( 0.0, -0.3 ) ( 0.3, 0.3 ) ( 0.0, 2.4 )

8.3. Program Results

f01rec Example Program Results

Matrix Q

( 0.00, 0.50 ) ( 0.00,-0.50 ) ( 0.00, 0.00 ) ( 0.50, 0.00 ) ( 0.40, 0.30 )
( 0.40, 0.30 ) ( -0.40,-0.30 ) ( 0.00, 0.00 ) ( -0.30, 0.40 ) ( -0.48, 0.14 )
( 0.40, 0.00 ) ( 0.40, 0.00 ) ( -0.60, 0.00 ) ( -0.24,-0.32 ) ( 0.00, 0.40 )
( 0.30,-0.40 ) ( 0.30,-0.40 ) ( 0.00, 0.00 ) ( 0.50, 0.00 ) ( -0.40,-0.30 )
( 0.00,-0.30 ) ( 0.00,-0.30 ) ( 0.00,-0.80 ) ( -0.24, 0.18 ) ( 0.30, 0.00 )