# NAG Fortran Library Routine Document

# F08NTF (CUNGHR/ZUNGHR)

**Note:** before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

## 1 Purpose

F08NTF (CUNGHR/ZUNGHR) generates the complex unitary matrix $Q$ which was determined by F08NSF (CGEHRD/ZGEHRD) when reducing a complex general matrix $A$ to Hessenberg form.

## 2 Specification

```
SUBROUTINE F08NTF(N, ILO, IHI, A, LDA, TAU, WORK, LWORK, INFO)
ENTRY      cunghr (N, ILO, IHI, A, LDA, TAU, WORK, LWORK, INFO)
INTEGER           N, ILO, IHI, LDA, LWORK, INFO
complex           A(LDA,*), TAU(*), WORK(*)
```

The ENTRY statement enables the routine to be called by its LAPACK name.

## 3 Description

This routine is intended to be used following a call to F08NSF (CGEHRD/ZGEHRD), which reduces a complex general matrix $A$ to upper Hessenberg form $H$ by a unitary similarity transformation: $A = QHQ^H$. F08NSF (CGEHRD/ZGEHRD)represents the matrix $Q$ as a product of $i_{hi} - i_{lo}$ elementary reflectors. Here $i_{lo}$ and $i_{hi}$ are values determined by F08NVF (CGEBAL/ZGEBAL) when balancing the matrix; if the matrix has not been balanced, $i_{lo} = 1$ and $i_{hi} = n$.

This routine may be used to generate $Q$ explicitly as a square matrix. $Q$ has the structure:

$$Q = \begin{pmatrix} I & 0 & 0 \\ 0 & Q_{22} & 0 \\ 0 & 0 & I \end{pmatrix}$$

where $Q_{22}$ occupies rows and columns $i_{lo}$ to $i_{hi}$.

## 4 References

Golub G H and van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

## 5 Parameters

1: N – INTEGER *Input*

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

*Constraint*: $N \geq 0$.

2: ILO – INTEGER *Input*
3: IHI – INTEGER *Input*

*On entry*: these **must** be the same parameters ILO and IHI, respectively, as supplied to F08NSF (CGEHRD/ZGEHRD).

*Constraints*:

$1 \leq \text{ILO} \leq \text{IHI} \leq \text{N}$ if $\text{N} > 0$,
$\text{ILO} = 1$ and $\text{IHI} = 0$ if $\text{N} = 0$.

4: A(LDA,*) – ***complex*** array *Input/Output*

**Note:** the second dimension of the array A must be at least $\max(1, \mathrm{N})$.

*On entry*: details of the vectors which define the elementary reflectors, as returned by F08NSF (CGEHRD/ZGEHRD).

*On exit*: the $n$ by $n$ unitary matrix $Q$.

5: LDA – INTEGER *Input*

*On entry*: the first dimension of the array A as declared in the (sub)program from which F08NTF (CUNGHR/ZUNGHR) is called.

*Constraint*: $\mathrm{LDA} \geq \max(1, \mathrm{N})$.

6: TAU($*$) – ***complex*** array *Input*

**Note:** the dimension of the array TAU must be at least $\max(1, \mathrm{N} - 1)$.

*On entry*: further details of the elementary reflectors, as returned by F08NSF (CGEHRD/ZGEHRD).

7: WORK($*$) – ***complex*** array *Workspace*

**Note:** the dimension of the array WORK must be at least $\max(1, \mathrm{LWORK})$.

*On exit*: if INFO $= 0$, the real part of WORK(1) contains the minimum value of LWORK required for optimum performance.

8: LWORK – INTEGER *Input*

*On entry*: the dimension of the array WORK as declared in the (sub)program from which F08NTF (CUNGHR/ZUNGHR) is called, unless LWORK $= -1$, in which case a workspace query is assumed and the routine only calculates the optimal dimension of WORK (using the formula given below).

*Suggested value*: for optimum performance LWORK should be at least (IHI − ILO) $\times$ $nb$, where $nb$ is the ***blocksize***.

*Constraint*: $\mathrm{LWORK} \geq \max(1, \mathrm{IHI} - \mathrm{ILO})$ or LWORK $= -1$.

9: INFO – INTEGER *Output*

*On exit*: INFO $= 0$ unless the routine detects an error (see Section 6).

# 6 Error Indicators and Warnings

Errors or warnings detected by the routine:

INFO $< 0$

If INFO $= -i$, the $i$th parameter had an illegal value. An explanatory message is output, and execution of the program is terminated.

# 7 Accuracy

The computed matrix $Q$ differs from an exactly unitary matrix by a matrix $E$ such that

$$\|E\|_2 = O(\epsilon),$$

where $\epsilon$ is the ***machine precision***.

## 8 Further Comments

The total number of real floating-point operations is approximately $\frac{16}{3}q^3$, where $q = i_{hi} - i_{lo}$.

The real analogue of this routine is F08NFF (SORGHR/DORGHR).

## 9 Example

To compute the Schur factorization of the matrix $A$, where

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

Here $A$ is general and must first be reduced to Hessenberg form by F08NSF (CGEHRD/ZGEHRD). The program then calls F08NTF (CUNGHR/ZUNGHR) to form $Q$, and passes this matrix to F08PSF (CHSEQR/ZHSEQR) which computes the Schur factorization of $A$.

### 9.1 Program Text

**Note:** the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     F08NTF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LDA, LDZ, LWORK
      PARAMETER        (NMAX=8,LDA=NMAX,LDZ=NMAX,LWORK=64*(NMAX-1))
*     .. Local Scalars ..
      INTEGER          I, IFAIL, INFO, J, N
*     .. Local Arrays ..
      complex          A(LDA,NMAX), TAU(NMAX), W(NMAX), WORK(LWORK),
     +                 Z(LDZ,NMAX)
      CHARACTER        CLABS(1), RLABS(1)
*     .. External Subroutines ..
      EXTERNAL         F06TFF, X04DBF, cgehrd, chseqr, cunghr
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F08NTF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
*
*        Read A from data file
*
         READ (NIN,*) ((A(I,J),J=1,N),I=1,N)
*
*        Reduce A to upper Hessenberg form H = (Q**H)*A*Q
*
         CALL cgehrd(N,1,N,A,LDA,TAU,WORK,LWORK,INFO)
*
*        Copy A into Z
*
         CALL F06TFF('General',N,N,A,LDA,Z,LDZ)
*
*        Form Q explicitly, storing the result in Z
*
         CALL cunghr(N,1,N,Z,LDZ,TAU,WORK,LWORK,INFO)
*
*        Calculate the Schur factorization of H = Y*T*(Y**H) and form
*        Q*Y explicitly, storing the result in Z
*
*        Note that A = Z*T*(Z**H), where Z = Q*Y
*
```

```
         CALL chseqr('Schur form','Vectors',N,1,N,A,LDA,W,Z,LDZ,WORK,
     +               LWORK,INFO)
*
*        Print Schur form
*
         WRITE (NOUT,*)
         IFAIL = 0
*
         CALL X04DBF('General',' ',N,N,A,LDA,'Bracketed','F7.4',
     +               'Schur form','Integer',RLABS,'Integer',CLABS,80,0,
     +               IFAIL)
*
*        Print Schur vectors
*
         WRITE (NOUT,*)
         IFAIL = 0
*
         CALL X04DBF('General',' ',N,N,Z,LDZ,'Bracketed','F7.4',
     +               'Schur vectors of A','Integer',RLABS,'Integer',
     +               CLABS,80,0,IFAIL)
*
      END IF
      STOP
*
      END
```

## 9.2 Program Data

```
F08NTF 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
```

## 9.3 Program Results

```
 F08NTF Example Program Results

 Schur form
                    1                  2                  3                  4
 1  (-6.0004,-6.9998) (-0.4701,-0.2119) ( 0.0438, 0.5124) (-0.9097,-0.0925)
 2  ( 0.0000, 0.0000) (-5.0000, 2.0060) ( 0.7150,-0.1028) (-0.0580, 0.2575)
 3  ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 7.9982,-0.9964) (-0.2232,-1.0549)
 4  ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 3.0023,-3.9998)

 Schur vectors of A
                    1                  2                  3                  4
 1  ( 0.8457, 0.0000) (-0.3613, 0.1351) (-0.1755, 0.2297) ( 0.1099,-0.2007)
 2  (-0.0177, 0.3036) (-0.3366, 0.4660) ( 0.7228, 0.0000) ( 0.0336, 0.2312)
 3  ( 0.0875, 0.3115) ( 0.6311, 0.0000) ( 0.2871, 0.4999) ( 0.0944,-0.3947)
 4  (-0.0561,-0.2906) (-0.1045,-0.3339) ( 0.2476, 0.0195) ( 0.8534, 0.0000)
```