NAG Library Routine Document

F08AWF (ZUNGLQ)

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

F08AWF (ZUNGLQ) generates all or part of the complex unitary matrix Q from an LQ factorization computed by F08AVF (ZGELQF).

2 Specification

```
SUBROUTINE F08AWF (M, N, K, A, LDA, TAU, WORK, LWORK, INFO)
INTEGER                 M, N, K, LDA, LWORK, INFO
COMPLEX (KIND=nag_wp) A(LDA,*), TAU(*), WORK(max(1,LWORK))
```

The routine may be called by its LAPACK name ***zunglq***.

3 Description

F08AWF (ZUNGLQ) is intended to be used after a call to F08AVF (ZGELQF), which performs an LQ factorization of a complex matrix A. The unitary matrix Q is represented as a product of elementary reflectors.

This routine may be used to generate Q explicitly as a square matrix, or to form only its leading rows.

Usually Q is determined from the LQ factorization of a p by n matrix A with $p \leq n$. The whole of Q may be computed by:

```
CALL ZUNGLQ(N,N,P,A,LDA,TAU,WORK,LWORK,INFO)
```

(note that the array A must have at least n rows) or its leading p rows by:

```
CALL ZUNGLQ(P,N,P,A,LDA,TAU,WORK,LWORK,INFO)
```

The rows of Q returned by the last call form an orthonormal basis for the space spanned by the rows of A; thus F08AVF (ZGELQF) followed by F08AWF (ZUNGLQ) can be used to orthogonalize the rows of A.

The information returned by the LQ factorization routines also yields the LQ factorization of the leading k rows of A, where $k < p$. The unitary matrix arising from this factorization can be computed by:

```
CALL ZUNGLQ(N,N,K,A,LDA,TAU,WORK,LWORK,INFO)
```

or its leading k rows by:

```
CALL ZUNGLQ(K,N,K,A,LDA,TAU,WORK,LWORK,INFO)
```

4 References

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

5 Parameters

1: M – INTEGER *Input*

On entry: m, the number of rows of the matrix Q.

Constraint: M $\geq$ 0.

2: N – INTEGER *Input*

On entry: n, the number of columns of the matrix Q.

Constraint: $\mathrm{N} \geq \mathrm{M}$.

3: K – INTEGER *Input*

On entry: k, the number of elementary reflectors whose product defines the matrix Q.

Constraint: $\mathrm{M} \geq \mathrm{K} \geq 0$.

4: A(LDA, $*$) – COMPLEX (KIND=nag_wp) 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 F08AVF (ZGELQF).

On exit: the m by n matrix Q.

5: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08AWF (ZUNGLQ) is called.

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

6: TAU($*$) – COMPLEX (KIND=nag_wp) array *Input*

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

On entry: further details of the elementary reflectors, as returned by F08AVF (ZGELQF).

7: WORK($\max(1, \mathrm{LWORK})$) – COMPLEX (KIND=nag_wp) array *Workspace*

On exit: if $\mathrm{INFO} = 0$, the real part of $\mathrm{WORK}(1)$ contains the minimum value of LWORK required for optimal performance.

8: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08AWF (ZUNGLQ) is called.

If $\mathrm{LWORK} = -1$, a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued.

Suggested value: for optimal performance, $\mathrm{LWORK} \geq \mathrm{M} \times nb$, where nb is the optimal ***block size***.

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

9: INFO – INTEGER *Output*

On exit: $\mathrm{INFO} = 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

$\mathrm{INFO} < 0$

If $\mathrm{INFO} = -i$, argument i 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 ϵ is the ***machine precision***.

8 Parallelism and Performance

F08AWF (ZUNGLQ) is not threaded by NAG in any implementation.

F08AWF (ZUNGLQ) 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 routine. 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 $16mnk - 8(m+n)k^2 + \frac{16}{3}k^3$; when $m = k$, the number is approximately $\frac{8}{3}m^2(3n-m)$.

The real analogue of this routine is F08AJF (DORGLQ).

10 Example

This example forms the leading 4 rows of the unitary matrix Q from the LQ factorization of the matrix A, where

$$A = \begin{pmatrix} 0.28-0.36i & 0.50-0.86i & -0.77-0.48i & 1.58+0.66i \\ -0.50-1.10i & -1.21+0.76i & -0.32-0.24i & -0.27-1.15i \\ 0.36-0.51i & -0.07+1.33i & -0.75+0.47i & -0.08+1.01i \end{pmatrix}.$$

The rows of Q form an orthonormal basis for the space spanned by the rows of A.

10.1 Program Text

```
    Program f08awfe

!     F08AWF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, x04dbf, zgelqf, zunglq
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, info, lda, lwork, m, n
      Character (30)                   :: title
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), tau(:), work(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Executable Statements ..
      Write (nout,*) 'F08AWF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) m, n
      lda = m
      lwork = 64*m
```

```
      Allocate (a(lda,n),tau(n),work(lwork))

!     Read A from data file

      Read (nin,*)(a(i,1:n),i=1,m)

!     Compute the LQ factorization of A
!     The NAG name equivalent of zgelqf is f08avf
      Call zgelqf(m,n,a,lda,tau,work,lwork,info)

!     Form the leading M rows of Q explicitly
!     The NAG name equivalent of zunglq is f08awf
      Call zunglq(m,n,m,a,lda,tau,work,lwork,info)

!     Print the leading M rows of Q only

      Write (nout,*)
      Write (title,99999) m
      Flush (nout)

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call x04dbf('General',' ',m,n,a,lda,'Bracketed','F7.4',title,'Integer', &
        rlabs,'Integer',clabs,80,0,ifail)

99999 Format ('The leading ',I2,' rows of Q')
    End Program f08awfe
```

10.2 Program Data

```
F08AWF Example Program Data
  3  4                                                    :Values of M and N
 ( 0.28,-0.36) ( 0.50,-0.86) (-0.77,-0.48) ( 1.58, 0.66)
 (-0.50,-1.10) (-1.21, 0.76) (-0.32,-0.24) (-0.27,-1.15)
 ( 0.36,-0.51) (-0.07, 1.33) (-0.75, 0.47) (-0.08, 1.01)   :End of matrix A
```

10.3 Program Results

```
 F08AWF Example Program Results

 The leading  3 rows of Q
                     1                  2                  3                  4
 1  (-0.1258, 0.1618) (-0.2247, 0.3864) ( 0.3460, 0.2157) (-0.7099,-0.2966)
 2  (-0.1163,-0.6380) (-0.3240, 0.4272) (-0.1995,-0.5009) (-0.0323,-0.0162)
 3  (-0.4607, 0.1090) ( 0.2171,-0.4062) ( 0.2733,-0.6106) (-0.0994,-0.3261)
```