

NAG Library Routine Document

F08CWF (ZUNGRQ)

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

F08CWF (ZUNGRQ) generates all or part of the complex n by n unitary matrix Q from an RQ factorization computed by F08CVF (ZGERQF).

2 Specification

```
SUBROUTINE F08CWF (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 *zungrq*.

3 Description

F08CWF (ZUNGRQ) is intended to be used following a call to F08CVF (ZGERQF), which performs an RQ factorization of a complex matrix A and represents the unitary matrix Q as a product of k elementary reflectors of order n .

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

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

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

(note that the matrix A must have at least n rows), or its trailing p rows as:

```
CALL ZUNGRQ(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 F08CVF (ZGERQF) followed by F08CWF (ZUNGRQ) can be used to orthogonalize the rows of A .

The information returned by F08CVF (ZGERQF) also yields the RQ factorization of the trailing k rows of A , where $k < p$. The unitary matrix arising from this factorization can be computed by:

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

or its leading k columns by:

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

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia <http://www.netlib.org/lapack/lug>

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

5 Arguments

- 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: $N \geq M$.
- 3: K – INTEGER *Input*
On entry: k , the number of elementary reflectors whose product defines the matrix Q .
Constraint: $M \geq K \geq 0$.
- 4: A(LDA,*) – COMPLEX (KIND=nag_wp) array *Input/Output*
Note: the second dimension of the array A must be at least $\max(1, N)$.
On entry: details of the vectors which define the elementary reflectors, as returned by F08CVF (ZGERQF).
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 F08CWF (ZUNGRQ) is called.
Constraint: $LDA \geq \max(1, M)$.
- 6: TAU(*) – COMPLEX (KIND=nag_wp) array *Input*
Note: the dimension of the array TAU must be at least $\max(1, K)$.
On entry: $TAU(i)$ must contain the scalar factor of the elementary reflector H_i , as returned by F08CVF (ZGERQF).
- 7: WORK($\max(1, LWORK)$) – COMPLEX (KIND=nag_wp) array *Workspace*
On exit: if $INFO = 0$, the real part of $WORK(1)$ 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 F08CWF (ZUNGRQ) is called.
 If $LWORK = -1$, 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, $LWORK \geq N \times nb$, where nb is the optimal **block size**.
Constraint: $LWORK \geq \max(1, M)$ 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

INFO < 0

If 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

$$\|E\|_2 = O\epsilon$$

and ϵ is the *machine precision*.

8 Parallelism and Performance

F08CWF (ZUNGQR) 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 floating-point operations is approximately $16mnk - 8(m+n)k^2 + \frac{16}{3}k^3$; when $m = k$ this becomes $\frac{8}{3}m^2(3n - m)$.

The real analogue of this routine is F08CJF (DORGRQ).

10 Example

This example generates the first four rows of the matrix Q of the RQ factorization of A as returned by F08CVF (ZGERQF), where

$$A = \begin{pmatrix} 0.96 - 0.81i & -0.98 + 1.98i & 0.62 - 0.46i & -0.37 + 0.38i & 0.83 + 0.51i & 1.08 - 0.28i \\ -0.03 + 0.96i & -1.20 + 0.19i & 1.01 + 0.02i & 0.19 - 0.54i & 0.20 + 0.01i & 0.20 - 0.12i \\ -0.91 + 2.06i & -0.66 + 0.42i & 0.63 - 0.17i & -0.98 - 0.36i & -0.17 - 0.46i & -0.07 + 1.23i \\ -0.05 + 0.41i & -0.81 + 0.56i & -1.11 + 0.60i & 0.22 - 0.20i & 1.47 + 1.59i & 0.26 + 0.26i \end{pmatrix}.$$

Note that the block size (NB) of 64 assumed in this example is not realistic for such a small problem, but should be suitable for large problems.

10.1 Program Text

```

Program f08cwfe

!      F08CWF Example Program Text

!      Mark 26 Release. NAG Copyright 2016.

!      .. Use Statements ..
      Use nag_library, Only: nag_wp, x04dbf, zgerqf, zungqr
!      .. Implicit None Statement ..
      Implicit None
!      .. Parameters ..
      Integer, Parameter          :: nb = 64, nin = 5, nout = 6
!      .. Local Scalars ..
      Integer                      :: i, ifail, info, lda, lwork, m, n
      Character (26)               :: title
!      .. Local Arrays ..

```

```

Complex (Kind=nag_wp), Allocatable :: a(:,,:), tau(:), work(:)
Character (1)                        :: clabs(1), rlabs(1)
! .. Executable Statements ..
Write (nout,*) 'F08CWF Example Program Results'
Write (nout,*)
! Skip heading in data file
Read (nin,*)
Read (nin,*) m, n
lda = m
lwork = nb*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 RQ factorization of A
! The NAG name equivalent of zgerqf is f08cvf
Call zgerqf(m,n,a,lda,tau,work,lwork,info)

! Form the leading M rows of Q explicitly
! The NAG name equivalent of zungrq is f08cwf
Call zungrq(m,n,m,a,lda,tau,work,lwork,info)

! Form the heading for X04DBF

Write (title,99999) m
Flush (nout)

! Print the leading M rows of Q

! 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 ',I4,' rows of Q')
End Program f08cwf

```

10.2 Program Data

F08CWF Example Program Data

```

4           6                               :Values of M and N

( 0.96,-0.81) (-0.98, 1.98) ( 0.62,-0.46) (-0.37, 0.38) ( 0.83, 0.51)
( 1.08,-0.28)
(-0.03, 0.96) (-1.20, 0.19) ( 1.01, 0.02) ( 0.19,-0.54) ( 0.20, 0.01)
( 0.20,-0.12)
(-0.91, 2.06) (-0.66, 0.42) ( 0.63,-0.17) (-0.98,-0.36) (-0.17,-0.46)
(-0.07, 1.23)
(-0.05, 0.41) (-0.81, 0.56) (-1.11, 0.60) ( 0.22,-0.20) ( 1.47, 1.59)
( 0.26, 0.26)                               :End of matrix A

```

10.3 Program Results

F08CWF Example Program Results

```

The leading 4 rows of Q
1           1           2           3           4
1 ( 0.2810, 0.5020) ( 0.2707,-0.3296) (-0.2864,-0.0094) ( 0.2262,-0.3854)
2 (-0.2051,-0.1092) ( 0.5711, 0.0432) (-0.5416, 0.0454) (-0.3387, 0.2228)
3 ( 0.3083,-0.6874) ( 0.2251,-0.1313) (-0.2062, 0.0691) ( 0.3259, 0.1178)
4 ( 0.0181,-0.1483) ( 0.2930,-0.2025) ( 0.4015,-0.2170) (-0.0796, 0.0723)

```

		5		6
1	(0.0341,-0.0760)	(-0.3936,-0.2083)
2	(0.0098,-0.0712)	(-0.1296, 0.3691)
3	(0.0753, 0.1412)	(0.0264,-0.4134)
4	(-0.5317,-0.5751)	(-0.0940,-0.0940)
