NAG Library Routine Document

F08APF (ZGEQRT)

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

F08APF (ZGEQRT) recursively computes, with explicit blocking, the QR factorization of a complex m by n matrix.

2 Specification

```
SUBROUTINE F08APF (M, N, NB, A, LDA, T, LDT, WORK, INFO)

INTEGER                M, N, NB, LDA, LDT, INFO
COMPLEX (KIND=nag_wp) A(LDA,*), T(LDT,*), WORK(NB*N)
```

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

3 Description

F08APF (ZGEQRT) forms the QR factorization of an arbitrary rectangular complex m by n matrix. No pivoting is performed.

It differs from F08ASF (ZGEQRF) in that it: requires an explicit block size; stores reflector factors that are upper triangular matrices of the chosen block size (rather than scalars); and recursively computes the QR factorization based on the algorithm of Elmroth and Gustavson (2000).

If $m \geq n$, the factorization is given by:

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix},$$

where R is an n by n upper triangular matrix (with real diagonal elements) and Q is an m by m unitary matrix. It is sometimes more convenient to write the factorization as

$$A = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}\begin{pmatrix} R \\ 0 \end{pmatrix},$$

which reduces to

$$A = Q_1 R,$$

where Q_1 consists of the first n columns of Q, and Q_2 the remaining $m - n$ columns.

If $m < n$, R is upper trapezoidal, and the factorization can be written

$$A = Q\begin{pmatrix} R_1 & R_2 \end{pmatrix},$$

where R_1 is upper triangular and R_2 is rectangular.

The matrix Q is not formed explicitly but is represented as a product of $\min(m, n)$ elementary reflectors (see the F08 Chapter Introduction for details). Routines are provided to work with Q in this representation (see Section 9).

Note also that for any $k < n$, the information returned represents a QR factorization of the first k columns of the original matrix A.

4 References

Elmroth E and Gustavson F (2000) Applying Recursion to Serial and Parallel QR Factorization Leads to Better Performance *IBM Journal of Research and Development. (Volume 44)* **4** 605–624

Golub G H and Van Loan C F (2012) *Matrix Computations* (4th Edition) Johns Hopkins University Press, Baltimore

5 Parameters

1: M – INTEGER *Input*

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

Constraint: $M \geq 0$.

2: N – INTEGER *Input*

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

Constraint: $N \geq 0$.

3: NB – INTEGER *Input*

On entry: the explicitly chosen block size to be used in computing the QR factorization. See Section 9 for details.

Constraints:

$NB \geq 1$;
if $\min(M, N) > 0$, $NB \leq \min(M, N)$.

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: the m by n matrix A.

On exit: if $m \geq n$, the elements below the diagonal are overwritten by details of the unitary matrix Q and the upper triangle is overwritten by the corresponding elements of the n by n upper triangular matrix R.

If $m < n$, the strictly lower triangular part is overwritten by details of the unitary matrix Q and the remaining elements are overwritten by the corresponding elements of the m by n upper trapezoidal matrix R.

The diagonal elements of R are real.

5: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08APF (ZGEQRT) is called.

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

6: T(LDT, $*$) – COMPLEX (KIND=nag_wp) array *Output*

Note: the second dimension of the array T must be at least $\max(1, \min(M, N))$.

On exit: further details of the unitary matrix Q. The number of blocks is $b = \left\lceil \frac{k}{NB} \right\rceil$, where $k = \min(m, n)$ and each block is of order NB except for the last block, which is of order $k - (b - 1) \times NB$. For each of the blocks, an upper triangular block reflector factor is computed: $\boldsymbol{T}_1, \boldsymbol{T}_2, \ldots, \boldsymbol{T}_b$. These are stored in the NB by n matrix T as $\boldsymbol{T} = [\boldsymbol{T}_1 | \boldsymbol{T}_2 | \ldots | \boldsymbol{T}_b]$.

7: LDT – INTEGER *Input*

On entry: the first dimension of the array T as declared in the (sub)program from which F08APF (ZGEQRT) is called.

Constraint: LDT $\geq$ NB.

8: WORK(NB $\times$ N) – COMPLEX (KIND=nag_wp) array *Workspace*

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 factorization is the exact factorization of a nearby matrix $(A+E)$, where

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

and ϵ is the ***machine precision***.

8 Parallelism and Performance

F08APF (ZGEQRT) is not threaded by NAG in any implementation.

F08APF (ZGEQRT) 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 $\frac{8}{3}n^2(3m-n)$ if $m \geq n$ or $\frac{8}{3}m^2(3n-m)$ if $m < n$.

To apply Q to an arbitrary complex rectangular matrix C, F08APF (ZGEQRT) may be followed by a call to F08AQF (ZGEMQRT). For example,

```
CALL ZGEMQRT('Left','Conjugate Transpose',M,P,MIN(M,N),NB,A,LDA, &
             T,LDT,C,LDC,WORK,INFO)
```

forms $C = Q^{\mathrm{H}}C$, where C is m by p.

To form the unitary matrix Q explicitly, simply initialize the m by m matrix C to the identity matrix and form $C = QC$ using F08AQF (ZGEMQRT) as above.

The block size, NB, used by F08APF (ZGEQRT) is supplied explicitly through the interface. For moderate and large sizes of matrix, the block size can have a marked effect on the efficiency of the algorithm with the optimal value being dependent on problem size and platform. A value of NB $= 64 \ll \min(m,n)$ is likely to achieve good efficiency and it is unlikely that an optimal value would exceed 340.

To compute a QR factorization with column pivoting, use F08BPF (ZTPQRT) or F08BSF (ZGEQPF).

The real analogue of this routine is F08ABF (DGEQRT).

10 Example

This example solves the linear least squares problems

$$\text{minimize} \|Ax_i - b_i\|_2, \quad i = 1, 2$$

where b_1 and b_2 are the columns of the matrix B,

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

and

$$B = \begin{pmatrix} -2.09 + 1.93i & 3.26 - 2.70i \\ 3.34 - 3.53i & -6.22 + 1.16i \\ -4.94 - 2.04i & 7.94 - 3.13i \\ 0.17 + 4.23i & 1.04 - 4.26i \\ -5.19 + 3.63i & -2.31 - 2.12i \\ 0.98 + 2.53i & -1.39 - 4.05i \end{pmatrix}.$$

10.1 Program Text

```
    Program f08apfe

!     F08APF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: dznrm2, nag_wp, x04dbf, zgemqrt, zgeqrt, ztrtrs
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nbmax = 64, nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, info, j, lda, ldb, ldt,    &
                                          lwork, m, n, nb, nrhs
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), b(:,:), t(:,:), work(:)
      Real (Kind=nag_wp), Allocatable  :: rnorm(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, min
!     .. Executable Statements ..
      Write (nout,*) 'F08APF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) m, n, nrhs
      lda = m
      ldb = m
      nb = min(m,n,nbmax)
      ldt = nb
      lwork = nb*max(m,n)
      Allocate (a(lda,n),b(ldb,nrhs),t(ldt,min(m,n)),work(lwork),rnorm(nrhs))

!     Read A and B from data file

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

```
!     Compute the QR factorization of A
!     The NAG name equivalent of zgeqrf is f08apf
      Call zgeqrt(m,n,nb,a,lda,t,ldt,work,info)

!     Compute C = (C1) = (Q**H)*B, storing the result in B
!                 (C2)
!     The NAG name equivalent of zgemqrt is f08aqf
      Call zgemqrt('Left','Conjugate transpose',m,nrhs,n,nb,a,lda,t,ldt,b,ldb, &
        work,info)

!     Compute least-squares solutions by backsubstitution in
!     R*X = C1
!     The NAG name equivalent of ztrtrs is f07tsf
      Call ztrtrs('Upper','No transpose','Non-Unit',n,nrhs,a,lda,b,ldb,info)

      If (info>0) Then
        Write (nout,*) 'The upper triangular factor, R, of A is singular, '
        Write (nout,*) 'the least squares solution could not be computed'
      Else

!       Print least-squares solutions

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call x04dbf('General',' ',n,nrhs,b,ldb,'Bracketed','F7.4', &
          'Least-squares solution(s)','Integer',rlabs,'Integer',clabs,80,0, &
          ifail)

!       Compute and print estimates of the square roots of the residual
!       sums of squares
!       The NAG name equivalent of dznrm2 is f06jjf
        Do j = 1, nrhs
          rnorm(j) = dznrm2(m-n,b(n+1,j),1)
        End Do

        Write (nout,*)
        Write (nout,*) 'Square root(s) of the residual sum(s) of squares'
        Write (nout,99999) rnorm(1:nrhs)
      End If

99999 Format (3X,1P,7E11.2)
    End Program f08apfe
```

10.2 Program Data

```
F08APF Example Program Data

   6             4             2                          : m, n and nrhs

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

 (-2.09, 1.93) ( 3.26,-2.70)
 ( 3.34,-3.53) (-6.22, 1.16)
 (-4.94,-2.04) ( 7.94,-3.13)
 ( 0.17, 4.23) ( 1.04,-4.26)
 (-5.19, 3.63) (-2.31,-2.12)
 ( 0.98, 2.53) (-1.39,-4.05)                              : matrix B
```

10.3 Program Results

```
 F08APF Example Program Results

 Least-squares solution(s)
                    1                 2
 1  (-0.5044,-1.2179) ( 0.7629, 1.4529)
 2  (-2.4281, 2.8574) ( 5.1570,-3.6089)
 3  ( 1.4872,-2.1955) (-2.6518, 2.1203)
 4  ( 0.4537, 2.6904) (-2.7606, 0.3318)

 Square root(s) of the residual sum(s) of squares
      6.88E-02   1.87E-01
```