NAG Library Routine Document

F08BVF (ZTZRZF)

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

F08BVF (ZTZRZF) reduces the m by n ($m \leq n$) complex upper trapezoidal matrix A to upper triangular form by means of unitary transformations.

2 Specification

```
SUBROUTINE F08BVF (M, N, A, LDA, TAU, WORK, LWORK, INFO)
INTEGER                  M, N, LDA, LWORK, INFO
COMPLEX (KIND=nag_wp) A(LDA,*), TAU(*), WORK(max(1,LWORK))
```

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

3 Description

The m by n ($m \leq n$) complex upper trapezoidal matrix A given by

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

where R_1 is an m by m upper triangular matrix and R_2 is an m by $(n-m)$ matrix, is factorized as

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

where R is also an m by m upper triangular matrix and Z is an n by n unitary matrix.

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

5 Parameters

1: M – INTEGER *Input*

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

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

2: N – INTEGER *Input*

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

Constraint: $\mathrm{N} \geq 0$.

3: 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: the leading m by n upper trapezoidal part of the array A must contain the matrix to be factorized.

On exit: the leading m by m upper triangular part of A contains the upper triangular matrix R, and elements $\mathrm{M}+1$ to N of the first m rows of A, with the array TAU, represent the unitary matrix Z as a product of m elementary reflectors (see Section 3.3.6 in the F08 Chapter Introduction).

4: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08BVF (ZTZRZF) is called.

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

5: TAU($*$) – COMPLEX (KIND=nag_wp) array *Output*

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

On exit: the scalar factors of the elementary reflectors.

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

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

7: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08BVF (ZTZRZF) 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 $\mathrm{LWORK} = -1$.

8: 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 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

F08BVF (ZTZRZF) is not threaded by NAG in any implementation.

F08BVF (ZTZRZF) 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 $16m^2(n-m)$.

The real analogue of this routine is F08BHF (DTZRZF).

10 Example

This example solves the linear least squares problems

$$\min_x \left\| b_j - Ax_j \right\|_2, \quad j = 1, 2$$

for the minimum norm solutions x_1 and x_2, where b_j is the jth column of the matrix B,

$$A = \begin{pmatrix} 0.47 - 0.34i & -0.40 + 0.54i & 0.60 + 0.01i & 0.80 - 1.02i \\ -0.32 - 0.23i & -0.05 + 0.20i & -0.26 - 0.44i & -0.43 + 0.17i \\ 0.35 - 0.60i & -0.52 - 0.34i & 0.87 - 0.11i & -0.34 - 0.09i \\ 0.89 + 0.71i & -0.45 - 0.45i & -0.02 - 0.57i & 1.14 - 0.78i \\ -0.19 + 0.06i & 0.11 - 0.85i & 1.44 + 0.80i & 0.07 + 1.14i \end{pmatrix}$$

and

$$B = \begin{pmatrix} -1.08 - 2.59i & 2.22 + 2.35i \\ -2.61 - 1.49i & 1.62 - 1.48i \\ 3.13 - 3.61i & 1.65 + 3.43i \\ 7.33 - 8.01i & -0.98 + 3.08i \\ 9.12 + 7.63i & -2.84 + 2.78i \end{pmatrix}.$$

The solution is obtained by first obtaining a QR factorization with column pivoting of the matrix A, and then the RZ factorization of the leading k by k part of R is computed, where k is the estimated rank of A. A tolerance of 0.01 is used to estimate the rank of A from the upper triangular factor, R.

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 f08bvfe

!     F08BVF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: dznrm2, nag_wp, x04dbf, zgeqp3, ztrsm, ztzrzf,    &
                             zunmqr, zunmrz
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Complex (Kind=nag_wp), Parameter :: one = (1.0_nag_wp,0.0_nag_wp)
      Complex (Kind=nag_wp), Parameter :: zero = (0.0_nag_wp,0.0_nag_wp)
      Integer, Parameter               :: inc1 = 1, nb = 64, nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: tol
      Integer                          :: i, ifail, info, j, k, lda, ldb,      &
                                          lwork, m, n, nrhs
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), b(:,:), tau(:), work(:)
      Real (Kind=nag_wp), Allocatable  :: rnorm(:), rwork(:)
      Integer, Allocatable             :: jpvt(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Intrinsic Procedures ..
```

```
      Intrinsic                        :: abs
!     .. Executable Statements ..
      Write (nout,*) 'F08BVF Example Program Results'
      Write (nout,*)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) m, n, nrhs
      lda = m
      ldb = m
      lwork = (n+1)*nb
      Allocate (a(lda,n),b(ldb,nrhs),tau(n),work(lwork),rnorm(n),rwork(2*n), &
        jpvt(n))

!     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)

!     Initialize JPVT to be zero so that all columns are free

      jpvt(1:n) = 0

!     Compute the QR factorization of A with column pivoting as
!     A = Q*(R11 R12)*(P**T)
!            ( 0  R22)
!     The NAG name equivalent of zgeqp3 is f08btf
      Call zgeqp3(m,n,a,lda,jpvt,tau,work,lwork,rwork,info)

!     Compute C = (C1) = (Q**H)*B, storing the result in B
!                 (C2)
!     The NAG name equivalent of zunmqr is f08auf
      Call zunmqr('Left','Conjugate transpose',m,nrhs,n,a,lda,tau,b,ldb,work, &
        lwork,info)

!     Choose TOL to reflect the relative accuracy of the input data

      tol = 0.01_nag_wp

!     Determine and print the rank, K, of R relative to TOL

loop: Do k = 1, n
        If (abs(a(k,k))<=tol*abs(a(1,1))) Exit loop
      End Do loop
      k = k - 1

      Write (nout,*) 'Tolerance used to estimate the rank of A'
      Write (nout,99999) tol
      Write (nout,*) 'Estimated rank of A'
      Write (nout,99998) k
      Write (nout,*)
      Flush (nout)

!     Compute the RZ factorization of the K by K part of R as
!     (R1 R2) = (T 0)*Z
!     The NAG name equivalent of ztzrzf is f08bvf
      Call ztzrzf(k,n,a,lda,tau,work,lwork,info)

!     Compute least-squares solutions of triangular problems by
!     back substitution in T*Y1 = C1, storing the result in B
!     The NAG name equivalent of ztrsm is f06zjf
      Call ztrsm('Left','Upper','No transpose','Non-Unit',k,nrhs,one,a,lda,b, &
        ldb)

!     Compute estimates of the square roots of the residual sums of
!     squares (2-norm of each of the columns of C2)

!     The NAG name equivalent of dznrm2 is f06jjf
      Do j = 1, nrhs
        rnorm(j) = dznrm2(m-k,b(k+1,j),inc1)
      End Do
```

```
!     Set the remaining elements of the solutions to zero (to give
!     the minimum-norm solutions), Y2 = 0

      b(k+1:n,1:nrhs) = zero

!     Form W = (Z**H)*Y
!     The NAG name equivalent of zunmrz is f08bxf
      Call zunmrz('Left','Conjugate transpose',n,nrhs,k,n-k,a,lda,tau,b,ldb, &
        work,lwork,info)

!     Permute the least-squares solutions stored in B to give X = P*W

      Do j = 1, nrhs
        Do i = 1, n
          work(jpvt(i)) = b(i,j)
        End Do
        b(1:n,j) = work(1:n)
      End Do

!     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)

!     Print the square roots of the residual sums of squares

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

99999 Format (3X,1P,7E11.2)
99998 Format (1X,I6)
    End Program f08bvfe
```

10.2 Program Data

```
F08BVF Example Program Data

   5             4             2                  :Values of M, N and NRHS

 ( 0.47,-0.34) (-0.40, 0.54) ( 0.60, 0.01) ( 0.80,-1.02)
 (-0.32,-0.23) (-0.05, 0.20) (-0.26,-0.44) (-0.43, 0.17)
 ( 0.35,-0.60) (-0.52,-0.34) ( 0.87,-0.11) (-0.34,-0.09)
 ( 0.89, 0.71) (-0.45,-0.45) (-0.02,-0.57) ( 1.14,-0.78)
 (-0.19, 0.06) ( 0.11,-0.85) ( 1.44, 0.80) ( 0.07, 1.14) :End of matrix A

 (-1.08,-2.59) ( 2.22, 2.35)
 (-2.61,-1.49) ( 1.62,-1.48)
 ( 3.13,-3.61) ( 1.65, 3.43)
 ( 7.33,-8.01) (-0.98, 3.08)
 ( 9.12, 7.63) (-2.84, 2.78)                              :End of matrix B
```

10.3 Program Results

```
 F08BVF Example Program Results

 Tolerance used to estimate the rank of A
      1.00E-02
 Estimated rank of A
      3

 Least-squares solution(s)
                    1                 2
 1  ( 1.1669,-3.3224) (-0.5023, 1.8323)
 2  ( 1.3486, 5.5027) (-1.4418,-1.6465)
```

```
 3  ( 4.1764, 2.3435) ( 0.2908, 1.4900)
 4  ( 0.6467, 0.0107) (-0.2453, 0.3951)

 Square root(s) of the residual sum(s) of squares
      2.51E-01   8.10E-02
```