NAG Library Routine Document

F08ZFF (DGGRQF)

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

F08ZFF (DGGRQF) computes a generalized RQ factorization of a real matrix pair (A, B), where A is an m by n matrix and B is a p by n matrix.

2 Specification

```
SUBROUTINE F08ZFF (M, P, N, A, LDA, TAUA, B, LDB, TAUB, WORK, LWORK,       &
                   INFO)
INTEGER            M, P, N, LDA, LDB, LWORK, INFO
REAL (KIND=nag_wp) A(LDA,*), TAUA(min(M,N)), B(LDB,*), TAUB(min(P,N)),     &
                   WORK(max(1,LWORK))
```

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

3 Description

F08ZFF (DGGRQF) forms the generalized RQ factorization of an m by n matrix A and a p by n matrix B

$$A = RQ, \quad B = ZTQ,$$

where Q is an n by n orthogonal matrix, Z is a p by p orthogonal matrix and R and T are of the form

$$R = \begin{cases} \begin{array}{cc} & \begin{array}{cc} n-m & m \end{array} \\ m & \begin{pmatrix} 0 & R_{12} \end{pmatrix} \end{array}; & \text{if } m \leq n, \\ \\ \begin{array}{cc} & n \\ m-n & \begin{pmatrix} R_{11} \\ R_{21} \end{pmatrix} \\ n & \end{array}; & \text{if } m > n, \end{cases}$$

with R_{12} or R_{21} upper triangular,

$$T = \begin{cases} \begin{array}{cc} & n \\ \begin{array}{c} n \\ p-n \end{array} & \begin{pmatrix} T_{11} \\ 0 \end{pmatrix} \end{array}; & \text{if } p \geq n, \\ \begin{array}{cc} & \begin{array}{cc} p & n-p \end{array} \\ p & \begin{pmatrix} T_{11} & T_{12} \end{pmatrix} \end{array}; & \text{if } p < n, \end{cases}$$

with T_{11} upper triangular.

In particular, if B is square and nonsingular, the generalized RQ factorization of A and B implicitly gives the RQ factorization of AB^{-1} as

$$AB^{-1} = \left(RT^{-1}\right)Z^{\mathrm{T}}.$$

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

Anderson E, Bai Z and Dongarra J (1992) Generalized *QR* factorization and its applications *Linear Algebra Appl. (Volume 162–164)* 243–271

Hammarling S (1987) The numerical solution of the general Gauss-Markov linear model *Mathematics in Signal Processing* (eds T S Durrani, J B Abbiss, J E Hudson, R N Madan, J G McWhirter and T A Moore) 441–456 Oxford University Press

Paige C C (1990) Some aspects of generalized QR factorizations . *In Reliable Numerical Computation* (eds M G Cox and S Hammarling) 73–91 Oxford University Press

5 Parameters

1: M – INTEGER *Input*

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

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

2: P – INTEGER *Input*

On entry: p, the number of rows of the matrix B.

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

3: N – INTEGER *Input*

On entry: n, the number of columns of the matrices A and B.

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

4: A(LDA, $*$) – REAL (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 m by n matrix A.

On exit: if $m \leq n$, the upper triangle of the subarray $\mathrm{A}(1:m, n-m+1:n)$ contains the m by m upper triangular matrix R_{12}.

If $m \geq n$, the elements on and above the $(m-n)$th subdiagonal contain the m by n upper trapezoidal matrix R; the remaining elements, with the array TAUA, represent the orthogonal matrix Q as a product of $\min(m, n)$ elementary reflectors (see Section 3.3.6 in the F08 Chapter Introduction).

5: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08ZFF (DGGRQF) is called.

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

6: TAUA($\min(\mathrm{M}, \mathrm{N})$) – REAL (KIND=nag_wp) array *Output*

On exit: the scalar factors of the elementary reflectors which represent the orthogonal matrix Q.

7: B(LDB, $*$) – REAL (KIND=nag_wp) array *Input/Output*

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

On entry: the p by n matrix B.

On exit: the elements on and above the diagonal of the array contain the $\min(p, n)$ by n upper trapezoidal matrix T (T is upper triangular if $p \geq n$); the elements below the diagonal, with the array TAUB, represent the orthogonal matrix Z as a product of elementary reflectors (see Section 3.3.6 in the F08 Chapter Introduction).

8: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F08ZFF (DGGRQF) is called.

Constraint: $\mathrm{LDB} \geq \max(1, \mathrm{P})$.

9: TAUB$(\min(\mathrm{P}, \mathrm{N}))$ – REAL (KIND=nag_wp) array *Output*

On exit: the scalar factors of the elementary reflectors which represent the orthogonal matrix Z.

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

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

11: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08ZFF (DGGRQF) 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, $\mathrm{LWORK} \geq \max(\mathrm{N}, \mathrm{M}, \mathrm{P}) \times \max(nb1, nb2, nb3)$, where $nb1$ is the optimal ***block size*** for the RQ factorization of an m by n matrix by F08CHF (DGERQF), $nb2$ is the optimal ***block size*** for the QR factorization of a p by n matrix by F08AEF (DGEQRF), and $nb3$ is the optimal ***block size*** for a call of F08CKF (DORMRQ).

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

12: 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 generalized RQ factorization is the exact factorization for nearby matrices $(A + E)$ and $(B + F)$, where

$$\|E\|_2 = O\epsilon\|A\|_2 \quad \text{and} \quad \|F\|_2 = O\epsilon\|B\|_2,$$

and ϵ is the ***machine precision***.

8 Parallelism and Performance

F08ZFF (DGGRQF) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F08ZFF (DGGRQF) 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 orthogonal matrices Q and Z may be formed explicitly by calls to F08CJF (DORGRQ) and F08AFF (DORGQR) respectively. F08CKF (DORMRQ) may be used to multiply Q by another matrix and F08AGF (DORMQR) may be used to multiply Z by another matrix.

The complex analogue of this routine is F08ZTF (ZGGRQF).

10 Example

This example solves the least squares problem

$$\underset{x}{\text{minimize}}\|c - Ax\|_2 \quad \text{subject to} \quad Bx = d$$

where

$$A = \begin{pmatrix} -0.57 & -1.28 & -0.39 & 0.25 \\ -1.93 & 1.08 & -0.31 & -2.14 \\ 2.30 & 0.24 & 0.40 & -0.35 \\ -1.93 & 0.64 & -0.66 & 0.08 \\ 0.15 & 0.30 & 0.15 & -2.13 \\ -0.02 & 1.03 & -1.43 & 0.50 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix},$$

$$c = \begin{pmatrix} -1.50 \\ -2.14 \\ 1.23 \\ -0.54 \\ -1.68 \\ 0.82 \end{pmatrix} \quad \text{and} \quad d = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

The constraints $Bx = d$ correspond to $x_1 = x_3$ and $x_2 = x_4$.

The solution is obtained by first computing a generalized RQ factorization of the matrix pair (B, A). The example illustrates the general solution process.

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 f08zffe

!     F08ZFF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: dgemv, dggrqf, dnrm2, dormqr, dormrq, dtrmv,      &
                             dtrtrs, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter    :: one = 1.0E0_nag_wp
      Integer, Parameter               :: nb = 64, nin = 5, nout = 6
!     .. Local Scalars ..
```

```
      Real (Kind=nag_wp)                 :: rnorm
      Integer                            :: i, info, lda, ldb, lwork, m, n, p
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable    :: a(:,:), b(:,:), c(:), d(:), taua(:), &
                                            taub(:), work(:), x(:)
!     .. Intrinsic Procedures ..
      Intrinsic                          :: min
!     .. Executable Statements ..
      Write (nout,*) 'F08ZFF Example Program Results'
      Write (nout,*)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) m, n, p
      lda = m
      ldb = p
      lwork = nb*(p+n)
      Allocate (a(lda,n),b(ldb,n),c(m),d(p),taua(n),taub(n),work(lwork),x(n))

!     Read B, A, C and D from data file
      Read (nin,*)(a(i,1:n),i=1,m)
      Read (nin,*)(b(i,1:n),i=1,p)
      Read (nin,*) c(1:m)
      Read (nin,*) d(1:p)

!     Compute the generalized RQ factorization of (B,A) as
!     B = (0 T12)*Q,   A = Z*(R11 R12)*Q, where T12, R11
!                            ( 0  R22)
!     are upper triangular
!     The NAG name equivalent of dggrqf is f08zff
      Call dggrqf(p,m,n,b,ldb,taub,a,lda,taua,work,lwork,info)

!     Set Qx = y. The problem then reduces to:
!                minimize (Ry - Z^Tc) subject to Ty = d
!     Update c = Z^T*c -> minimize (Ry-c)
!     The NAG name equivalent of dormqr is f08agf
      Call dormqr('Left','Transpose',m,1,min(m,n),a,lda,taua,c,m,work,lwork, &
        info)

!     Putting y = (y1), solve T12*w = d for w, storing result in d
!                 (w )
!     The NAG name equivalent of dtrtrs is f07tef
      Call dtrtrs('Upper','No transpose','Non-unit',p,1,b(1,n-p+1),ldb,d,p, &
        info)

      If (info>0) Then
        Write (nout,*) 'The upper triangular factor of B is singular, '
        Write (nout,*) 'the least squares solution could not be computed'
        Go To 100
      End If

!     From first n-p rows of (Ry-c) we have: R11*y1 + R12*w = c(1:n-p) = c1
!     Form c1 = c1 - R12*w = R11*y1
!     The NAG name equivalent of dgemv is f06raf
      Call dgemv('No transpose',n-p,p,-one,a(1,n-p+1),lda,d,1,one,c,1)

!     Solve R11*y1 = c1 for y1, storing result in c(1:n-p)
!     The NAG name equivalent of dtrtrs is f07tef
      Call dtrtrs('Upper','No transpose','Non-unit',n-p,1,a,lda,c,n-p,info)

      If (info>0) Then
        Write (nout,*) 'The upper triangular factor of A is singular, '
        Write (nout,*) 'the least squares solution could not be computed'
        Go To 100
      End If

!     Copy y into X (first y1, then w)
      x(1:n-p) = c(1:n-p)
      x(n-p+1:n) = d(1:p)

!     Compute x = (Q**T)*y
!     The NAG name equivalent of dormrq is f08ckf
```

```
      Call dormrq('Left','Transpose',n,1,p,b,ldb,taub,x,n,work,lwork,info)

!     The least squares solution is in x, the remainder here is to compute
!     the residual, which equals c2 - R22*w.

!     Upper triangular part of R22 first
!     The NAG name equivalent of dtrmv is f06pff
      Call dtrmv('Upper','No transpose','Non-unit',min(m,n)-n+p, &
        a(n-p+1,n-p+1),lda,d,1)
      Do i = 1, min(m,n) - n + p
        c(n-p+i) = c(n-p+i) - d(i)
      End Do

      If (m<n) Then

!       Additional rectangular part of R22
!       The NAG name equivalent of dgemv is f06paf
        Call dgemv('No transpose',m-n+p,n-m,-one,a(n-p+1,m+1),lda,d(m-n+p+1), &
          1,one,c(n-p+1),1)
      End If

!     Compute norm of residual sum of squares.
      rnorm = dnrm2(m-(n-p),c(n-p+1),1)

!     Print least squares solution x
      Write (nout,*) 'Constrained least squares solution'
      Write (nout,99999) x(1:n)

!     Print estimate of the square root of the residual sum of squares
      Write (nout,*)
      Write (nout,*) 'Square root of the residual sum of squares'
      Write (nout,99998) rnorm

100   Continue

99999 Format (1X,7F11.4)
99998 Format (3X,1P,E11.2)
    End Program f08zffe
```

10.2 Program Data

```
F08ZFF Example Program Data

  6      4      2           : m, n and p

 -0.57  -1.28  -0.39   0.25
 -1.93   1.08  -0.31  -2.14
  2.30   0.24   0.40  -0.35
 -1.93   0.64  -0.66   0.08
  0.15   0.30   0.15  -2.13
 -0.02   1.03  -1.43   0.50 : A

  1.00   0.00  -1.00   0.00
  0.00   1.00   0.00  -1.00 : B

 -1.50
 -2.14
  1.23
 -0.54
 -1.68
  0.82                      : c

  0.00
  0.00                      : d
```

10.3 Program Results

```
F08ZFF Example Program Results

Constrained least squares solution
     0.4890     0.9975     0.4890     0.9975

Square root of the residual sum of squares
     2.51E-02
```