F08AFFP (PDORGQR)

NAG Parallel Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check for implementation-dependent details. You are advised to enclose any calls to NAG Parallel Library routines between calls to Z01AAFP and Z01ABFP.

1 Description

This routine is intended to be used after a call to F08AEFP (PDGEQRF), which performs a QR factorization of an m by r real matrix A_s. F08AEFP represents the m by m orthogonal matrix Q as a product of elementary reflectors.

F08AFFP (PDORGQR) may be used to generate Q explicitly as an m by m square matrix, or to form the n leading columns of Q, where $n \leq m$.

Alternatively, this routine may be called to compute the orthogonal matrix Q of the QR factorization of the k leading columns of the matrix A_s for $k \leq r$.

This routine returns the matrix Q in the array A which on entry must contain the details of the elementary reflectors computed by F08AEFP (PDGEQRF). The distribution of the matrix Q conforms to the details in the description array IDESCA.

2 Specification

```
 SUBROUTINE F08AFFP(M, N, K, A, IA, JA, IDESCA, TAU, WORK, LWORK,
1                   INFO)
 ENTRY      PDORGQR(M, N, K, A, IA, JA, IDESCA, TAU, WORK, LWORK,
1                   INFO)
 DOUBLE PRECISION   A(*), TAU(*), WORK(*)
 INTEGER            M, N, K, IA, JA, IDESCA(*), LWORK, INFO
```

The ENTRY statement enables the routine to be called by its ScaLAPACK name.

3 Usage

3.1 Definitions

The following definitions are used in describing the data distribution within this document:

m_p	–	the number of rows in the Library Grid.
n_p	–	the number of columns in the Library Grid.
p_r	–	the row grid coordinate of the calling processor.
p_c	–	the column grid coordinate of the calling processor.
M_b^X	–	the blocking factor for the distribution of the rows of a matrix X.
N_b^X	–	the blocking factor for the distribution of the columns of a matrix X.
numroc(α,b_ℓ,q,s,k)	–	a function which gives the **num**ber of **r**ows **o**r **c**olumns of a distributed matrix owned by the processor with the row or column coordinate q (p_r or p_c), where α is the total number of rows or columns of the matrix, b_ℓ is the blocking factor used (M_b^X or N_b^X), s is the row or column coordinate of the processor that possesses the first row or column of the distributed matrix and k is either m_p or n_p. The Library provides the function Z01CAFP (NUMROC) for the evaluation of this function.
indxg2p(i_g,b_ℓ,q,s,k)	–	a function which gives the processor row or column coordinate which possess the row or column index i_g of the distributed full matrix A. The arguments b_ℓ, q, s and k have the same meaning as in the function numroc. The Library provides the function Z01CDFP (INDXG2P) for the evaluation of this function.

3.2 Global and Local Arguments

The following global **input** arguments must have the same value on entry to the routine on each processor and the global **output** arguments will have the same value on exit from the routine on each processor:

Global input arguments: M, N, K, IA, JA, IDESCA(1), IDESCA(3:8)

Global output arguments: INFO

The remaining arguments are local.

3.3 Distribution Strategy

On entry to this routine, the input values of M, A, IA, JA, IDESCA and TAU must be identical to the output values of the corresponding arguments on exit from the QR factorization routine F08AEFP (PDGEQRF).

In F08AEFP (PDGEQRF), the matrix Q is represented as a set of elementary reflectors but in F08AFFP (PDORGQR), the matrix Q is explicitly computed. In both these routines, the details of Q are stored in the local arrays A but the minimal storage requirements (as specified by the second dimensions) of A on a particular processor are, in general, different. In particular, if $r < n$ then F08AFFP (PDORGQR) requires larger local arrays for A than in F08AEFP (PDGEQRF).

3.4 Related Routines

The Library provides many support routines for the generation, scattering/gathering and input/output of matrices/vectors in cyclic two-dimensional block form. The following routines may be used in conjunction with F08AFFP (PDORGQR):

Real matrix output: X04BDFP

Real matrix gather: F01WAFP

4 Arguments

1: M — INTEGER *Global Input*

On entry: m, the number of rows of the matrix Q. This should be identical to the number of rows of the matrix A_s as previously supplied to the QR factorization routine F08AEFP (PDGEQRF).

Constraint: $0 \leq \mathrm{M} \leq \mathrm{IDESCA}(3)$.

2: N — INTEGER *Global Input*

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

Constraints: $0 \leq \mathrm{N} \leq \mathrm{IDESCA}(4)$; $\mathrm{N} \leq \mathrm{M}$.

3: K — INTEGER *Global Input*

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

Constraint: $0 \leq \mathrm{K} \leq \mathrm{N}$.

4: A(∗) — DOUBLE PRECISION array *Local Input/Local Output*

Note: array A is formally defined as a vector. However, you may find it more convenient to consider A as a two-dimensional array of dimension (IDESCA(9),γ), where
$\gamma \geq \mathrm{numroc}(\mathrm{JA}+\mathrm{N}-1,\mathrm{IDESCA}(6),p_c,\mathrm{IDESCA}(8),n_p)$.

On entry: details of the vectors which define the elementary reflectors as returned by F08AEFP (PDGEQRF).

On exit: the local parts of the first n columns of the m by m matrix Q. The distribution of the matrix Q is defined by the indices IA and JA and the description array IDESCA.

5: IA — INTEGER *Global Input*

On entry: i_A, the row index of matrix A that identifies the first row of Q to be generated. This must be identical to the value previously used in F08AEFP (PDGEQRF).

Constraints: $1 \leq \text{IA} \leq \text{IDESCA}(3) - \text{M} + 1$.

6: JA — INTEGER *Global Input*

On entry: j_A, the column index of matrix A that identifies the first column of Q to be generated. This must be identical to the value previously used in F08AEFP (PDGEQRF).

Constraints: $1 \leq \text{JA} \leq \text{IDESCA}(4) - \max(\text{N},\text{K}) + 1$.

7: IDESCA(∗) — INTEGER array *Local Input*

Note: the dimension of the array IDESCA must be at least 9.

Distribution: the array elements IDESCA(1) and IDESCA(3)... IDESCA(8) must be global to the processor grid and the elements IDESCA(2) and IDESCA(9) are local to each processor.

On entry: the description array for the matrix A as defined in the QR factorization routine F08AEFP (PDGEQRF). This array must contain details of the distribution of the matrix A and the logical processor grid.

IDESCA(1), the descriptor type. For this routine, which uses a cyclic two-dimensional block distribution, IDESCA(1) = 1;
IDESCA(2), the Library context, usually returned by a call to the Library Grid initialisation routine Z01AAFP;
IDESCA(3), the number of rows, m_A, of the matrix A;
IDESCA(4), the number of columns, n_A, of the matrix A;
IDESCA(5), the blocking factor, M_b^A, used to distribute the rows of the matrix A;
IDESCA(6), the blocking factor, N_b^A, used to distribute the columns of the matrix A;
IDESCA(7), the processor row index over which the first row of the matrix A is distributed;
IDESCA(8), the processor column index over which the first column of the matrix A is distributed;
IDESCA(9), the leading dimension of the conceptual two-dimensional array A.

Constraints:

IDESCA(1) = 1;
IDESCA(3) $\geq$ 0; IDESCA(4) $\geq$ 0;
IDESCA(5) $\geq$ 1; IDESCA(6) $\geq$ 1;
$0 \leq \text{IDESCA}(7) \leq m_p - 1$; $0 \leq \text{IDESCA}(8) \leq n_p - 1$;
IDESCA(9) $\geq$ max(1,numroc(IDESCA(3),IDESCA(5),p_r,IDESCA(7),m_p)).

8: TAU(∗) — DOUBLE PRECISION array *Local Input*

Note: the dimension of the array TAU must be at least α, where
α = numroc(JA+K −1,IDESCA(6),p_c,IDESCA(8),n_p).

On entry: details of the elementary reflectors, as returned by a call to F08AEFP (PDGEQRF).

9: WORK(∗) — DOUBLE PRECISION array *Local Workspace/Local Output*

Note: the dimension of WORK must be at least max(1,LWORK). The minimum value of LWORK required to successfully call this routine can be obtained by setting LWORK = −1. The required size is returned in array element WORK(1).

On exit: WORK(1) contains the minimum dimension of the array WORK required to successfully complete the task.

10: LWORK — INTEGER *Local Input*

On entry: either -1 (see WORK) or the dimension of the array WORK required to successfully complete the task. If LWORK is set to -1 on entry this routine simply performs some initial error checking and then, if these checks are successful, calculates the minimum size of LWORK required.

Constraint:

either LWORK $= -1$
or LWORK $\geq$ IDESCA(6) $\times$ ($c_1 + c_2$+IDESCA(6)), where
c_1 = numroc(M+d_1,IDESCA(5),p_r,e_1,m_p);
c_2 = numroc(N+d_2,IDESCA(6),p_c,e_2,n_p);
d_1 = mod(IA$-$1,IDESCA(5));
d_2 = mod(JA$-$1,IDESCA(6));
e_1 = indxg2p(IA,IDESCA(5),p_r,IDESCA(7),m_p);
e_2 = indxg2p(JA,IDESCA(6),p_c,IDESCA(8),n_p).

11: INFO — INTEGER *Global Output*

The NAG Parallel Library provides a mechanism, via the routine Z02EAFP, to reduce the amount of parameter validation performed by this routine. For a full description refer to the Z02 Chapter Introduction.

On exit: INFO $= 0$ (or -9999 if reduced error checking is enabled) unless the routine detects an error (see Section 5).

5 Errors and Warnings

If INFO < 0 an explanatory message is output and control returned to the calling program.

INFO < 0

On entry, one of the arguments was invalid:

if the kth argument is a scalar INFO $= -k$;
if the kth argument is an array and the jth element is invalid, INFO $= -(100 + j)$.

This error occured either because a global argument did not have the same value on all logical processors, or because its value on one or more processors was incorrect. An explanatory message distinguishes between these two cases.

6 Further Comments

Often Q is determined from the QR factorization of an m by r matrix A with $m \geq r$. The matrix Q may be computed by calling:

```
CALL F08AFFP (M,M,r,A,IA,JA,IDESCA,TAU,WORK,LWORK,INFO)
```

The leading r columns of Q may be obtained by:

```
CALL F08AFFP (M,r,r,A,IA,JA,IDESCA,TAU,WORK,LWORK,INFO)
```

The columns of Q returned by the last call form an orthonormal basis for the space spanned by the columns of A; thus F08AEFP (PDGEQRF) followed by F08AFFP (PDORGQR) can be used to orthogonalise the columns of A.

6.1 Algorithmic Detail

See Anderson *et al.* [1] and Blackford *et al.* [2] for details of the block method used by the routine.

6.2 Parallelism Detail

The Level-3 BLAS operations are carried out in parallel within the routine.

6.3 Accuracy

The computed matrix Q differs from an exactly orthogonal matrix by a matrix E such that

$$\|E\|_2 \leq \epsilon p(m,n),$$

where ϵ is the ***machine precision***, $p(m,n)$ is a modest function of m and n.

7 References

[1] 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

[2] Blackford L S, Choi J, Cleary A, D'Azevedo E, Demmel J, Dhillon I, Dongarra J, Hammarling S, Henry G, Petitet A, Stanley K, Walker D and Whaley R C (1997) ScaLAPACK Users' Guide *SIAM* 3600 University City Science Center, Philadelpia, PA 19104-2688, USA. URL: http://www.netlib.org/scalapack/slug/scalapack_slug.html

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

8 Example

To form the leading four columns of the orthogonal matrix Q from the QR factorization of the matrix A, where

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

The columns of Q form an orthonormal basis for the space spanned by the columns of A. The example uses a 2 by 2 logical processor grid and a 2 by 2 block for A.

Note: the listing of the Example Program presented below does not give a full pathname for the data file being opened, but in general the user must give the full pathname in this and any other OPEN statement.

8.1 Example Text

```
*     F08AFFP Example Program Text
*     NAG Parallel Library Release 3 Revised. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          DT
      PARAMETER        (DT=1)
      INTEGER          MB, NB, LWORK
      PARAMETER        (MB=2,NB=MB,LWORK=100)
      INTEGER          MMAX, NMAX, LDA, IAROW, IACOL
      PARAMETER        (MMAX=6,NMAX=4,LDA=MMAX,IAROW=0,IACOL=0)
*     .. Local Scalars ..
      INTEGER          I, IA, ICNTXT, IFAIL, INFO, J, JA, M, MP, N, NP
      LOGICAL          ROOT
*     .. Local Arrays ..
```

```
      DOUBLE PRECISION A(LDA,NMAX), Q(MMAX,MMAX), TAU(MMAX), WORK(LWORK)
      INTEGER          IDESCA(9)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         F01WAFP, F08AEFP, F08AFFP, X04BCFP, Z01AAFP,
     +                 Z01ABFP
*     .. Executable Statements ..
      ROOT = Z01ACFP()
      IF (ROOT) THEN
         WRITE (NOUT,*) 'F08AFFP Example Program Results'
         WRITE (NOUT,*)
      END IF
*
      MP = 2
      NP = 2
      IFAIL = 0
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      OPEN (NIN,FILE='f08affpe.d')
*
*     Skip heading in data file
*
      READ (NIN,*)
      READ (NIN,*) M, N
*
      IF (M.LE.MMAX .AND. N.LE.NMAX) THEN
*
*        Set the starting address and array descriptor for A
*
         IA = 1
         JA = 1
         IDESCA(1) = DT
         IDESCA(2) = ICNTXT
         IDESCA(3) = M
         IDESCA(4) = N
         IDESCA(5) = MB
         IDESCA(6) = NB
         IDESCA(7) = IAROW
         IDESCA(8) = IACOL
         IDESCA(9) = LDA
*
*        Read the matrix A from data file
*
         IFAIL = 0
         CALL X04BCFP(NIN,M,N,A,IA,JA,IDESCA,IFAIL)
*
*        Factorize A
*
         CALL F08AEFP(M,N,A,IA,JA,IDESCA,TAU,WORK,LWORK,INFO)
*
         IF (ROOT) THEN
            WRITE (NOUT,*) 'F08AEFP INFO = ', INFO
            WRITE (NOUT,*)
         END IF
         IF (INFO.NE.0) GO TO 40
*
```

```
*        Generate Q
*
         CALL F08AFFP(M,N,N,A,IA,JA,IDESCA,TAU,WORK,LWORK,INFO)
*
         IF (ROOT) THEN
            WRITE (NOUT,*) 'F08AFFP INFO = ', INFO
            WRITE (NOUT,*)
         END IF
         IF (INFO.NE.0) GO TO 40
*
*        Gather the matrix Q to the (root) processor (0,0)
*
         IFAIL = 0
         CALL F01WAFP(M,N,A,IA,JA,IDESCA,0,0,Q,MMAX,WORK,LWORK,IFAIL)
*
*        Print the first N columns of the matrix Q
*
         IF (ROOT) THEN
            WRITE (NOUT,*) 'The first N columns of Q'
            WRITE (NOUT,*)
            DO 20 I = 1, M
               WRITE (NOUT,'(1X,4(F12.4,1X))') (Q(I,J),J=1,N)
   20       CONTINUE
         END IF
*
      END IF
*
   40 CONTINUE
      CLOSE (NIN)
*
      IFAIL = 0
      CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
      STOP
*
      END
```

8.2 Example Data

```
F08AFFP Example Program Data
 6 4 '(4F12.4)'           :Values of M, N and FORMAT
-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 of matrix A
```

8.3 Example Results

```
F08AFFP Example Program Results

F08AEFP INFO =            0

F08AFFP INFO =            0

The first N columns of Q

     -0.1576       0.6744      -0.4571       0.4489
     -0.5335      -0.3861       0.2583       0.3898
      0.6358      -0.2928       0.0165       0.1930
     -0.5335      -0.1692      -0.0834      -0.2350
      0.0415      -0.1593       0.1475       0.7436
     -0.0055      -0.5064      -0.8339       0.0335
```