# F08AGFP (PDORMQR)
# 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 (PDGEQRF) represents the real orthogonal matrix $Q$ as a product of elementary reflectors.

F08AGFP (PDORMQR) multiplies an $m$ by $n$ real matrix $C_s$ by $Q$ from a $QR$ factorization procedure, where $C_s$ is a submatrix of a larger $m_C$ by $n_C$ matrix $C$, i.e.,

$$C_s(1:m,1:n) \equiv C(i_C:i_C+m-1, j_C:j_C+n-1).$$

This routine may be used to form one of the matrix products

$$QC_s, Q^TC_s, C_sQ \text{ or } C_sQ^T,$$

overwriting the result on $C_s$.

## 2 Specification

```
      SUBROUTINE F08AGFP(SIDE, TRANS, M, N, K, A, IA, JA, IDESCA, TAU,
     1                   C, IC, JC, IDESCC, WORK, LWORK, INFO)
      ENTRY      PDORMQR(SIDE, TRANS, M, N, K, A, IA, JA, IDESCA, TAU,
     1                   C, IC, JC, IDESCC, WORK, LWORK, INFO)
      DOUBLE PRECISION   A(*), TAU(*), C(*), WORK(LWORK)
      INTEGER            M, N, K, IA, JA, IDESCA(9), IC, JC, IDESCC(9),
     1                   LWORK, INFO
      CHARACTER*1        SIDE, TRANS
```

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

## 3 Data Distribution

### 3.1 Definitions

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

| | | |
|---|---|---|
| $m_p$ | – | the number of rows in the logical processor grid. |
| $n_p$ | – | the number of columns in the logical processor 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$(\alpha, b_\ell, q, s, k)$ | – | 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 $\alpha$ is the total number of rows or columns of the matrix, $b_\ell$ 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 $n_p$ or $m_p$. The Library provides the function Z01CAFP (NUMROC) for the evaluation of this function. |
| indxg2p$(i_g, b_\ell, q, s, k)$ | – | 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 $b_\ell, q, s$ 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 input arguments SIDE, TRANS, M, N, K, IA, JA, IC, JC and the array elements IDESCA(1), IDESCA(3),…,IDESCA(8), IDESCC(1) and IDESCC(3),…,IDESCC(8) are all global and so must have the same values on entry to the routine on every processor. The output argument INFO is global and so will have the same value on exit from the routine on each processor. 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 identically equal to the output values of the corresponding arguments on exit from the $QR$ factorization routine F08AEFP (PDGEQRF).

The matrix $C$ should be partitioned into $M_b^C$ by $N_b^C$ rectangular blocks, which are stored in the array C in a cyclic 2-d block distribution. This data distribution is described in more detail in the the F08 Chapter Introduction.

This routine assumes that the data has already been correctly distributed, and if this is not the case will fail to produce correct results. However, the Library provides some utility routines which assist you in distributing data correctly. Descriptions of these routines can be found in Chapters F01 and X04 of the NAG Parallel Library Manual.

## 4 Arguments

**Warning:** This routine is derived from ScaLAPACK and accurately reflects the specification of the equivalent ScaLAPACK routine. The current release (1.2) of ScaLAPACK imposed a global change in the specification of descriptor arrays. Consequently any applications developed using this routine from Release 1 of the Library will not run correctly, without change, using this Release.

**1:** SIDE — CHARACTER*1 *Global Input*

*On entry:* indicates how $Q$ or $Q^T$ is to be applied to $C_s$ as follows:

if SIDE = 'L', then $Q$ or $Q^T$ is applied to $C_s$ from the left;
if SIDE = 'R', then $Q$ or $Q^T$ is applied to $C_s$ from the right.

*Constraint:* SIDE = 'L' or 'R'.

**2:** TRANS — CHARACTER*1 *Global Input*

*On entry:* indicates whether $Q$ or $Q^T$ is to be applied to $C_s$ as follows:

if TRANS = 'N', then $Q$ is applied to $C_s$;
if TRANS = 'T', then $Q^T$ is applied to $C_s$.

*Constraint:* TRANS = 'N' or 'T'.

**3:** M — INTEGER *Global Input*

*On entry:* the number of rows, $m$, of the matrix $C_s$.

*Constraint:*

if SIDE = 'L', $0 \leq$ M$\leq$ min(IDESCA(3), IDESCC(3));
if SIDE = 'R', $0 \leq$ M $\leq$ IDESCC(3).

**4:** N — INTEGER *Global Input*

*On entry:* the number of columns, $n$, of the matrix $C_s$.

*Constraint:*

if SIDE = 'R', $0 \leq$ N $\leq$ min(IDESCA(3), IDESCC(4));
if SIDE = 'L', $0 \leq$ N $\leq$ IDESCC(4).

**5:** K — INTEGER *Global Input*

*On entry:* the number of elementary reflectors, $k$, whose product defines $Q$. These reflectors are computed by a call to F08AEFP (PDGEQRF).

*Constraint:*

if SIDE = 'L', $0 \leq$ K $\leq$ M;
if SIDE = 'R', $0 \leq$ K $\leq$ N.

**6:** 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 2-d array of dimension (IDESCA(9),$\gamma$), where
$\gamma \geq$ numroc(JA+K$-$1,IDESCA(6),$p_c$,IDESCA(8),$n_p$). See ? (PDGEQRF).

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

*On exit:* A is used as workspace, but restored on exit.

**7:** IA — INTEGER *Global Input*

*On entry:* the row index of matrix $A$, $i_A$, that identified the first row of the submatrix $A_s$ as defined in F08AEFP (PDGEQRF).

*Constraints:* $1 \leq$ IA $\leq$ IDESCA(3) $-$ M + 1.

**8:** JA — INTEGER *Global Input*

*On entry:* the column index of matrix $A$, $j_A$, that identified the first column of the submatrix $A_s$ as defined in F08AEFP (PDGEQRF).

*Constraint:* $1 \leq$ JA $\leq$ IDESCA(4) $-$ K + 1.

**9:** IDESCA(9) — INTEGER array *Local Input*

*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 2-d block distribution, IDESCA(1) = 1;
IDESCA(2), the BLACS context (ICNTXT) for the processor grid, usually returned by 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 2-d array A.

*Constraints:*

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

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

**Note:** the dimension of the array TAU must be at least 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).

**11:** C(∗) — DOUBLE PRECISION array *Local Input/Local Output*

**Note:** array C is formally defined as a vector. However, you may find it more convenient to consider C as a 2-d array of dimension (IDESCC(9),$\gamma$), where
$\gamma \geq$ numroc(JC+N−1,IDESCC(6),$p_c$,IDESCC(8),$n_p$). See ? (PDGEQRF).

*On entry:* the local part of the matrix $C$.

*On exit:* overwritten by $QC_s, Q^TC_s, C_sQ$ or $C_sQ^T$.

**12:** IC — INTEGER *Global Input*

*On entry:* the row index of matrix $C$, $i_C$, that identifies the first row of the submatrix $C_s$.

*Constraint:* 1 ≤ IC ≤ IDESCC(3) − M + 1.

**13:** JC — INTEGER *Global Input*

*On entry:* the column index of matrix $C$, $j_C$, that identifies the first column of the submatrix $C_s$.

*Constraint:* 1 ≤ JC ≤ IDESCC(4) − N + 1.

**14:** IDESCC(9) — INTEGER array *Local Input*

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

*On entry:* the description array for the matrix $C$. This array must contain details of the distribution of the matrix $C$ and the logical processor grid.

IDESCC(1), the descriptor type. For this routine, which uses a cyclic 2-d block distribution, IDESCC(1) = 1;
IDESCC(2), the BLACS context (ICNTXT) for the processor grid, usually returned by Z01AAFP;
IDESCC(3), the number of rows, $m_C$, of the matrix $C$;
IDESCC(4), the number of columns, $n_C$, of the matrix $C$;
IDESCC(5), the blocking factor, $M_b^C$, used to distribute the rows of the matrix $C$;
IDESCC(6), the blocking factor, $N_b^C$, used to distribute the columns of the matrix $C$;
IDESCC(7), the processor row index over which the first row of the matrix $C$ is distributed;
IDESCC(8), the processor column index over which the first column of the matrix $C$ is distributed;
IDESCC(9), the leading dimension of the conceptual 2-d array C.

*Constraints:*

IDESCC(1) = 1;
IDESCC(2) = IDESCA(2)
IDESCC(3) ≥ 0; IDESCC(4) ≥ 0;
IDESCC(5) ≥ 1; IDESCC(6) ≥ 1;
0 ≤ IDESCC(7) ≤ $m_p$ − 1; 0 ≤ IDESCC(8) ≤ $n_p$ − 1;
IDESCC(9) ≥ max(1,numroc(IDESCC(3),IDESCC(5),$p_r$,IDESCC(7),$m_p$));
if SIDE = 'L',
    IDESCC(5) = IDESCA(5);
    mod(IA−1,IDESCA(5)) = mod(IC−1,IDESCC(5));
    mod(IDESCA(7)+(IA−1)/IDESCA(5),$n_p$) =
    mod(IDESCC(7)+(IC−1)/IDESCC(5),$n_p$);
if SIDE = 'R',
    IDESCC(6) = IDESCA(5); IDESCC(7) = IDESCA(7);
    mod(IA−1,IDESCA(5)) = mod(JC−1,IDESCC(6)).

**15:** WORK(LWORK) — DOUBLE PRECISION array *Local Workspace*

*On exit:* WORK(1) contains the minimum value of LWORK.

**16:** LWORK — INTEGER *Local Input*

*On entry:* the dimension of the array WORK as declared in the (sub)program from which F08AGFP (PDORMQR) is called.

*Constraint:* LWORK $\geq$ max((IDESCA(6)$\times$(IDESCA(6)$-$1))/2,$\theta\times$IDESCA(6)) + IDESCA(6) $\times$ IDESCA(6), where

if SIDE = 'L', $\theta = d_1 + d_2$;
if SIDE = 'R', $\theta = d_2 + \max(f_1 + \text{numroc}(\lambda,\text{IDESCA}(6),0,0,l_{mn}/n_p),d_1)$; where
$\lambda = \text{numroc}(\text{N}+e_2,\text{IDESCA}(6),0,0,n_p)$,
and $l_{mn}$ is the least common multiple of $m_p$ and $n_p$;

with,

$d_1 = \text{numroc}(\text{M}+e_1,\text{IDESCC}(5),p_r,e_3,m_p)$;
$d_2 = \text{numroc}(\text{N}+e_2\text{IDESCC}(6),p_c,e_4,n_p)$;
$e_1 = \text{mod}(\text{IC}-1,\text{IDESCC}(5))$;
$e_2 = \text{mod}(\text{JC}-1,\text{IDESCC}(6))$;
$e_3 = \text{indxg2p}(\text{IC},\text{IDESCC}(5),p_r,\text{IDESCC}(7),m_p)$;
$e_4 = \text{indxg2p}(\text{JC},\text{IDESCC}(6),p_c,\text{IDESCC}(8),n_p)$.
$f_1 = \text{numroc}(\text{N}+f_2,\text{IDESCA}(5),p_r,f_3,m_p)$;
$f_2 = \text{mod}(\text{IA}-1,\text{IDESCA}(5))$;
$f_3 = \text{indxg2p}(\text{IA},\text{IDESCA}(5),p_r,\text{IDESCA}(7), m_p)$.

This value of LWORK can be calculated by using the Library function Z01CCFP; i.e.,

LWORK = Z01CCFP(SIDE,M,N,IA,JA,IC,JC, IDESCA,IDESCC)

**17:** INFO — INTEGER *Global Output*

*On exit:* INFO = 0 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 $k$th argument is a scalar INFO $= -k$;
if the $k$th argument is an array and the $j$th element is invalid, INFO $= -(100 \times k + j)$.

This error occurred 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

## 6.1 Algorithmic Detail

See Golub and Van Loan [2] and Anderson *et al.* [1].

## 6.2 Parallelism Detail

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

# 7 References

**[1]** Anderson E, Bai Z, Bischof C, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A, Ostrouchov S and Sorensen D (1995) *LAPACK Users' Guide* (2nd Edition) SIAM, Philadelphia

**[2]** Golub G H and Van Loan C F (1989) *Matrix Computations* Johns Hopkins University Press (2nd Edition), Baltimore

# 8 Example

See ? (PDGEQRF).