F07ARFP (PZGETRF)
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

F07ARFP (PZGETRF) computes the LU factorization of a complex m by n matrix A_s, where A_s is a submatrix of a larger m_A by n_A matrix A, i.e.,

$$A_s(1:m,1:n) \equiv A(i_A:i_A+m-1, j_A:j_A+n-1).$$

Note: if $i_A = j_A = 1$, $m = m_A$ and $n = n_A$, then $A_s = A$.

F07ARFP (PZGETRF) factorizes A_s as $A_s = PLU$, where P is a permutation matrix, L is lower triangular with unit diagonal elements (lower trapezoidal if $m > n$) and U is upper triangular (upper trapezoidal if $m < n$). Usually A is square ($m = n$), and both L and U are triangular. The routine uses partial pivoting, with row interchanges.

2 Specification

```
SUBROUTINE F07ARFP(M, N, A, IA, JA, IDESCA, IPIV, INFO)
ENTRY      PZGETRF(M, N, A, IA, JA, IDESCA, IPIV, INFO)
COMPLEX*16         A(*)
INTEGER            M, N, IA, JA, IDESCA(*), IPIV(*), 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.

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, IA, JA, IDESCA(1), IDESCA(3:8)
Global output arguments:	INFO

The remaining arguments are local.

3.3 Distribution Strategy

The matrix A must be partitioned into M_b^A by N_b^A rectangular blocks (in this release $M_b^A = N_b^A$) and stored in an array A in a cyclic two-dimensional block distribution. This data distribution is described in more detail in the the F07 Chapter Introduction. The resulting LU factorization is stored in the same data distribution.

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 F07ARFP (PZGETRF):

Complex matrix generation:	F01ZVFP
Complex matrix input:	X04BRFP
Complex matrix output:	X04BSFP
Complex matrix gather:	F01WGFP
Complex matrix scatter:	F01WUFP

4 Arguments

1: M — INTEGER *Global Input*

On entry: m, the number of rows of the submatrix A_s.

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

2: N — INTEGER *Global Input*

On entry: n, the number of columns of the submatrix A_s.

Constraint: $0 \leq \text{N} \leq \text{IDESCA}(4)$.

3: A($*$) — COMPLEX*16 array *Local Input/Local Output*

Note: the 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
$\gamma \geq$ numroc(JA+N−1,IDESCA(6),p_c,IDESCA(8),n_p).

On entry: the local part of the matrix A which may contain parts of the m by n submatrix A_s to be factorized.

On exit: A is overwritten by the factors L and U distributed in the same cyclic two-dimensional block fashion; the unit diagonal elements of L are not stored.

4: IA — INTEGER *Global Input*

On entry: i_A, the row index of matrix A that identifies the first row of the submatrix A_s to be factorized.

Constraints: $1 \leq \text{IA} \leq \text{IDESCA}(3) - \text{M} + 1$ and mod(IA−1,IDESCA(5)) = 0.

5: JA — INTEGER *Global Input*

On entry: j_A, the column index of matrix A that identifies the first column of the submatrix A_s to be factorized.

Constraints: $1 \leq \text{JA} \leq \text{IDESCA}(4) - \text{N} + 1$ and mod(JA−1,IDESCA(6)) = 0.

6: 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 array elements IDESCA(2) and IDESCA(9) are local to each processor.

On entry: the description array for the matrix A. 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) = IDESCA(6); 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)).

7: IPIV($*$) — INTEGER array *Local Output*

Note: the dimension of the array IPIV must be at least β + IDESCA(5) where,
β = numroc(IDESCA(3),IDESCA(5),p_r,IDESCA(7),m_p).

On exit: the pivot indices. The global row IPIV(k) was interchanged with the local row k. This array is aligned with the distributed matrix A.

8: 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 $\neq 0$ explanatory error messages are output from the root processor (or processor $\{0, 0\}$ when the root processor is not available) on the current error message unit (as defined by X04AAF).

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 its jth 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.

INFO > 0

If INFO $= i$, the element u_{ii} of the upper triangular matrix U is exactly zero. The factorization has been completed but the factor U is exactly singular, and division by zero will occur if it is subsequently used to solve a system of linear equations or to compute A^{-1}.

6 Further Comments

The total number of floating-point operations is approximately $\frac{8}{3}n^3$ if $m = n$ (the usual case), $\frac{4}{3}n^2(3m-n)$ if $m > n$ and $\frac{4}{3}m^2(3n-m)$ if $m < n$.

A call to this routine with $m = n$ may be followed by a call to the routine F07ASFP (PZGETRS) to solve a system of equations $A_sX = B_s$.

6.1 Algorithmic Detail

The routine uses a block-partitioned *LU* factorization with partial pivoting. See Anderson *et al.* [1] for details of the method used by the routine.

6.2 Parallelism Detail

Each processor column performs an *LU* factorization with partial pivoting on successive column blocks of the matrix. Details of this factorization and pivoting are passed to all processors which perform the update of the remaining submatrix in parallel.

6.3 Accuracy

The computed factors L and U are the exact factors of a perturbed matrix $A + E$, where

$$|E| \leq c(\min(m,n))\epsilon P|L| \cdot |U|,$$

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

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 compute the *LU* factorization of the matrix A, where

$$A = \begin{pmatrix} -1.34+2.55i & 0.28+3.17i & -6.39-2.20i & 0.72-0.92i \\ -0.17-1.41i & 3.31-0.15i & -0.15+1.34i & 1.29+1.38i \\ -3.29-2.39i & -1.91+4.42i & -0.14-1.35i & 1.72+1.35i \\ 2.41+0.39i & -0.56+1.47i & -0.83-0.69i & -1.96+0.67i \end{pmatrix}.$$

The example uses a 2 by 2 logical processor grid and a block size of 2.

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

```
*     F07ARFP Example Program Text
*     NAG Parallel Library Release 2. NAG Copyright 1996.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          DT
      PARAMETER        (DT=1)
      INTEGER          MB, NB
      PARAMETER        (MB=2,NB=MB)
      INTEGER          MMAX, NMAX, IAROW, IACOL, LDA, LW
      PARAMETER        (MMAX=8,NMAX=8,IAROW=0,IACOL=0,LDA=MMAX,LW=NMAX)
*     .. Local Scalars ..
      INTEGER          IA, ICNTXT, IFAIL, INFO, JA, M, MP, N, NP
      LOGICAL          ROOT
      CHARACTER*80     FORMAT
*     .. Local Arrays ..
      COMPLEX*16       A(LDA,NMAX), WORK(LW)
      INTEGER          IDESCA(9), IPIV(MMAX+MB)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         F07ARFP, X04BRFP, X04BSFP, Z01AAFP, Z01ABFP
*     .. Executable Statements ..
      ROOT = Z01ACFP()
      IF (ROOT) WRITE (NOUT,*) 'F07ARFP Example Program Results'
*
      MP = 2
      NP = 2
      IFAIL = 0
*
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      OPEN (NIN,FILE='f07arfpe.d')
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) M, N, FORMAT
*
      IF (M.LE.MMAX .AND. N.LE.NMAX) THEN
*
*        Set the array descriptor of A
*
         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
         IA = 1
         JA = 1
```

```
*
*        Read A from the data file
*
         IFAIL = 0
         CALL X04BRFP(NIN,M,N,A,1,1,IDESCA,IFAIL)
*
*        Factorize the matrix
*
         CALL F07ARFP(M,N,A,IA,JA,IDESCA,IPIV,INFO)
*
         IF (INFO.EQ.0) THEN
*
*           Print factor
*
            IF (ROOT) THEN
               WRITE (NOUT,*)
               WRITE (NOUT,*) 'Details of the factorization'
               WRITE (NOUT,*)
            END IF
            IFAIL = 0
*
            CALL X04BSFP(NOUT,M,N,A,IA,JA,IDESCA,FORMAT,WORK,IFAIL)
*
         ELSE IF (INFO.GT.0) THEN
            IF (ROOT) THEN
               WRITE (NOUT,*)
               WRITE (NOUT,*) 'Matrix is singular'
            END IF
         END IF
*
      END IF
*
      CLOSE (NIN)
*
      IFAIL = 0
      CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
      STOP
      END
```

8.2 Example Data

```
F07ARFP Example Program Data
  4  4  '(4(:,'' ('',F7.4,'','',F7.4,'')''))'           :Values of M, N and FORMAT
 (-1.34, 2.55) ( 0.28, 3.17) (-6.39,-2.20) ( 0.72,-0.92)
 (-0.17,-1.41) ( 3.31,-0.15) (-0.15, 1.34) ( 1.29, 1.38)
 (-3.29,-2.39) (-1.91, 4.42) (-0.14,-1.35) ( 1.72, 1.35)
 ( 2.41, 0.39) (-0.56, 1.47) (-0.83,-0.69) (-1.96, 0.67)  :End of matrix A
```

8.3 Example Results

```
F07ARFP Example Program Results

Details of the factorization

(-3.2900,-2.3900) (-1.9100, 4.4200) (-0.1400,-1.3500) ( 1.7200, 1.3500)
( 0.2376, 0.2560) ( 4.8952,-0.7114) (-0.4623, 1.6966) ( 1.2269, 0.6190)
(-0.1020,-0.7010) (-0.6691, 0.3689) (-5.1414,-1.1300) ( 0.9983, 0.3850)
(-0.5359, 0.2707) (-0.2040, 0.8601) ( 0.0082, 0.1211) ( 0.1482,-0.1252)
```