X04BRFP

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

X04BRFP reads an m by n complex matrix A_s from an external file (stored in its natural, non-distributed form) into an array in a cyclic two-dimensional block distribution on a logical grid of processors; 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 $A_s = A$.

This routine distributes matrices in the form required by some routines in Chapter F07 and Chapter F08.

2 Specification

```
SUBROUTINE X04BRFP(NIN, M, N, A, IA, JA, IDESCA, IFAIL)
COMPLEX*16         A(*)
INTEGER            NIN, M, N, IA, JA, IDESCA(*), IFAIL
```

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, IFAIL, IDESCA(1), IDESCA(3:8)

Global output arguments: IFAIL

Note: NIN may have a different value on each processor but it is likely to be the same on each processor – the only constraint is that it must be attached to the same file (or identical files) on each processor.

3.3 Distribution Strategy

On exit, the resulting array, A, contains the local portion of the matrix A_s distributed in a cyclic two-dimensional block distribution for each processor. This data distribution is described in more detail in

the Essential Introduction of the NAG Parallel Library and in the F07 Chapter Introduction and the F08 Chapter Introduction. Array elements of A contained within the submatrix A_s will have been updated with the values read from the input file.

4 Arguments

1: NIN — INTEGER *Local Input*

On entry: the unit number on which the external file is to be read.

Constraint: $0 < \text{NIN} \leq 99$.

2: M — INTEGER *Global Input*

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

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

3: N — INTEGER *Global Input*

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

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

4: A(∗) — COMPLEX*16 array *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
$\gamma \geq \text{numroc(JA+ N } -1\text{,IDESCA(6)},p_c\text{,IDESCA(8)},n_p)$.

On exit: the local part of the matrix A, distributed in a cyclic two-dimensional block fashion.

5: IA — INTEGER *Global Input*

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

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

6: JA — INTEGER *Global Input*

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

Constraint: $1 \leq \text{JA} \leq \text{IDESCA(4)} - \text{N} + 1$.

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. 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;
$0 \leq$ 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)).

8: IFAIL — INTEGER *Global Input/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 entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this argument (described in the Essential Introduction) the recommended values are:

IFAIL = 0, if multigridding is **not** employed;
IFAIL = −1, if multigridding is employed.

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

5 Errors and Warnings

If on entry IFAIL = 0 or −1, 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).

5.1 Full Error Checking Mode Only

IFAIL = −2000

The routine has been called with a value of ICNTXT (stored in IDESCA(2)) which was not returned by a call to Z01AAFP on one or more processors.

IFAIL = −1000

The logical processor grid and library mechanism (Library Grid) have not been correctly defined, see Z01AAFP.

IFAIL < 0

On entry, one of the arguments was invalid:

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

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.

5.2 Any Error Checking Mode

IFAIL = 1

The routine has reached the end of the external file unexpectedly. This is probably due to supplying values for M and N which are too large for the amount of data actually stored in the file.

IFAIL = 2

An error has occurred while reading the data, probably due to the routine trying to read an incorrect data type. This may occur if there are non-numeric characters accidentally mixed with the matrix data or if the current read position is not at the beginning of the matrix data when the routine is called.

6 Further Comments

The routine needs exactly n matrix elements for each row of A_s and assumes that the next row of A_s begins on the next line of the external file.

All of the processors read from the data file concurrently. On exit from the routine all processors will have read data to the same point in the file. There is no way of checking that the data file is the same for all processors; if the file is not the same for all processors, the behaviour of the routine could be unpredictable.

7 References

None.

8 Example

This example reads in a matrix A from a data file, computes the LU factorization of the matrix, using routine F07ARFP, and prints the LU factors to standard output. 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

```
*     X04BRFP 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
      PARAMETER        (MB=2,NB=MB)
      INTEGER          MMAX, NMAX, LDA, IAROW, IACOL, NPMAX, MPMAX
      PARAMETER        (MMAX=8,NMAX=8,LDA=MMAX,IAROW=0,IACOL=0,MPMAX=2,
     +                 NPMAX=2)
      INTEGER          IW
      PARAMETER        (IW=NMAX)
*     .. Local Scalars ..
      INTEGER          IA, ICNTXT, IFAIL, INFO, JA, M, N, NCOLS, NROWS
      LOGICAL          ROOT
      CHARACTER*80     FORMAT
*     .. Local Arrays ..
      COMPLEX*16       A(LDA,NMAX), WORK(IW)
      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) THEN
         WRITE (NOUT,*) 'X04BRFP Example Program Results'
         WRITE (NOUT,*)
      END IF
*
      NROWS = MPMAX
      NCOLS = NPMAX
      IFAIL = 0
*
*   Define Library Grid
*
      CALL Z01AAFP(ICNTXT,NROWS,NCOLS,IFAIL)

      OPEN (NIN,FILE='x04brfpe.d')
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) M, N, FORMAT
*
*     Set the array descriptor of A
*
      IA = 1
      JA = 1
      IDESCA(1) = DT
      IDESCA(2) = ICNTXT
      IDESCA(3) = MMAX
      IDESCA(4) = NMAX
      IDESCA(5) = MB
      IDESCA(6) = NB
      IDESCA(7) = IAROW
      IDESCA(8) = IACOL
      IDESCA(9) = LDA
*
      IF (IA+M-1.LE.MMAX .AND. JA+N-1.LE.NMAX) THEN
*
*        Read A from the data file
*
         IFAIL = 0
         CALL X04BRFP(NIN,M,N,A,IA,JA,IDESCA,IFAIL)
*
*        Factorise the matrix
*
         CALL F07ARFP(M,N,A,IA,JA,IDESCA,IPIV,INFO)
         IF (INFO.EQ.0) THEN
*
*           Print factor
*
            IF (ROOT) THEN
               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) WRITE (NOUT,*) 'Matrix is singular'
         END IF
      END IF
*
      CLOSE (NIN)
*
*    Undefine Library Grid
*
      IFAIL = 0
      CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
      STOP
*
      END
```

8.2 Example Data

```
X04BRFP Example Program Data
 4 4 '(4(:,'' ('',F7.4,'','',F7.4,'')''))'             :Values of M, N and FORMAT
 (1.80,1.80)   (2.88,2.88)   (2.05,2.05)  (-0.89,-0.89)
 (5.25,5.25)  (-2.95,-2.95) (-0.95,-0.95) (-3.80,-3.80)
 (1.58,1.58)  (-2.69,-2.69) (-2.90,-2.90) (-1.04,-1.04)
(-1.11,-1.11) (-0.66,-0.66) (-0.59,-0.59)  (0.80,0.80) :End of matrix A
```

8.3 Example Results

```
 X04BRFP Example Program Results

 Details of the factorization

 ( 5.2500, 5.2500) (-2.9500,-2.9500) (-0.9500,-0.9500) (-3.8000,-3.8000)
 ( 0.3429, 0.0000) ( 3.8914, 3.8914) ( 2.3757, 2.3757) ( 0.4129, 0.4129)
 ( 0.3010, 0.0000) (-0.4631, 0.0000) (-1.5139,-1.5139) ( 0.2948, 0.2948)
 (-0.2114, 0.0000) (-0.3299, 0.0000) ( 0.0047, 0.0000) ( 0.1314, 0.1314)
```