F01ZWFP

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

F01ZWFP generates a distributed m by n complex matrix A on a grid of logical processors in column block form. This routine distributes matrices in the form required by a number of routines in Chapter F02. A user-supplied subroutine is required to generate a block of the matrix A.

2 Specification

```
SUBROUTINE F01ZWFP(ICNTXT, GMAT, M, N, A, LDA, NX, IFAIL)
COMPLEX*16         A(LDA,*)
INTEGER            ICNTXT, M, N, LDA, NX, IFAIL
EXTERNAL           GMAT
```

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	–	$m_p \times n_p$, the total number of processors in the Library Grid.
p_d	–	the number of logical processors which hold columns of the matrix A.
N_b	–	the blocking factor for the distribution of the columns of the matrix.
N_x	–	the actual number of columns of the matrix A held locally on a logical processor where $0 \leq N_x \leq N_b$.
$\lceil x \rceil$	–	the ceiling function of x, which gives the smallest integer greater than or equal to x.

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, IFAIL

Global output arguments: IFAIL

The remaining arguments are local.

3.3 Distribution Strategy

Columns of the matrix A are allocated to logical processors on the two-dimensional Library Grid row by row (i.e., in the row major ordering of the grid) starting from the {0,0} logical processor. Each logical processor that contains columns of the matrix contains $N_b = \lceil n/p \rceil$ columns, except the last processor that actually contains data, for which the number of columns held may be less than N_b. This processor will contain $\mathrm{mod}(n, N_b)$ columns if $\mathrm{mod}(n, N_b) \neq 0$, and will contain N_b columns otherwise. Some logical processors may not contain any columns of the matrix if n is not large relative to p, but if $n > (p-1)^2$ then all processors will certainly contain columns of the matrix.

The number of logical processors that contain columns of the matrix is given by $p_d = \lceil n/N_b \rceil$.

The following example illustrates a case where the last processor with data is not the last processor of the grid. Furthermore the number of columns on the last processor with data is not equal to the number of columns on other processors.

If $m_p = 2$, $n_p = 4$ then $p = m_p \times n_p = 8$. If $n = 41$ then $N_b = \lceil n/p \rceil = \lceil 5.125 \rceil = 6$, $\mathrm{mod}(n, N_b) = 5$ and $p_d = \lceil n/N_b \rceil = \lceil 6.833 \rceil = 7$.

processor {0,0} $N_x = 6$ columns (1:6)	processor {0,1} $N_x = 6$ columns (7:12)	processor {0,2} $N_x = 6$ columns (13:18)	processor {0,3} $N_x = 6$ columns (19:24)
processor {1,0} $N_x = 6$ columns (25:30)	processor {1,1} $N_x = 6$ columns (31:36)	processor {1,2} $N_x = 5$ columns (37:41)	processor {1,3} $N_x = 0$

4 Arguments

1: ICNTXT — INTEGER *Local Input*

On entry: the Library context, usually returned by a call to the Library Grid initialisation routine Z01AAFP.

Note: the value of ICNTXT **must not** be changed.

2: GMAT — SUBROUTINE, supplied by the user. *External Procedure*

GMAT must return the block $A(1 : m, j_1 : j_2)$ of the matrix to be distributed, in the array AL. That is, GMAT must return columns j_1 to j_2 of A.

Its specification is:

```
SUBROUTINE          GMAT(M, J1, J2, AL, LDAL)
COMPLEX*16          AL(LDAL,*)
INTEGER             M, J1, J2, LDAL
```

1: M — INTEGER *Global Input*

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

2: J1 — INTEGER *Local Input*

On entry: j_1, the first column of the block of A to be generated.

3: J2 — INTEGER *Local Input*

On entry: j_2, the last column of the block of A to be generated.

4: AL(LDAL,*) — COMPLEX*16 array *Local Output*

On exit: AL must contain the block $A(1 : m, j_1 : j_2)$ of the matrix A in its first m rows and $(j_2 - j_1 + 1)$ columns.

5: LDAL — INTEGER *Local Input*

On entry: the size of the first dimension of the array AL as declared in the (sub)program from which F01ZWFP is called.

GMAT must be declared as EXTERNAL in the (sub)program from which F01ZWFP is called. Arguments denoted as *Input* must **not** be changed by this procedure.

3: M — INTEGER *Global Input*

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

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

4: N — INTEGER *Global Input*

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

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

5: A(LDA,*) — COMPLEX*16 array *Local Output*

Note: the dimension of the array A must be at least N_x, the number of columns held on the local processor.

On exit: the local processor's part of the matrix A.

6: LDA — INTEGER *Local Input*

On entry: the size of the first dimension of the array A as declared in the (sub)program from which F01ZWFP is called.

Constraint: LDA $\geq$ max(1,m).

7: NX — INTEGER *Local Output*

On exit: N_x, the number of columns of the matrix A held by the logical processor.

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 an invalid value of ICNTXT on one or more processors.

IFAIL = −1000

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

IFAIL = $-i$

On entry, the ith argument was invalid. 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

This routine may be used to distribute the data in the form required by a number of routines in Chapter F02.

6.1 Algorithmic Detail

The routine generates the matrix on a logical processor by column block.

6.2 Parallelism Detail

The routine generates the column blocks on each logical processor independently.

7 References

[1] 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

8 Example

To generate the 6 by 7 matrix A given by

$$A = \left(\begin{array}{cc|cc|cc|c} 2.0+2.0i & 1.0+2.0i & 1.0+3.0i & 1.0+4.0i & 1.0+5.0i & 1.0+6.0i & 1.0+7.0i \\ 2.0+1.0i & 3.0+3.0i & 2.0+3.0i & 2.0+4.0i & 2.0+5.0i & 2.0+6.0i & 2.0+7.0i \\ 3.0+1.0i & 3.0+2.0i & 4.0+4.0i & 3.0+4.0i & 3.0+5.0i & 3.0+6.0i & 3.0+7.0i \\ 4.0+1.0i & 4.0+2.0i & 4.0+3.0i & 5.0+5.0i & 4.0+5.0i & 4.0+6.0i & 4.0+7.0i \\ 5.0+1.0i & 5.0+2.0i & 5.0+3.0i & 5.0+4.0i & 6.0+6.0i & 5.0+6.0i & 5.0+7.0i \\ 6.0+1.0i & 6.0+2.0i & 6.0+3.0i & 6.0+4.0i & 6.0+5.0i & 7.0+7.0i & 6.0+7.0i \end{array}\right)$$

on a two-dimensional processor grid and to print the matrix on the root processor. Routine F01ZWFP is used to generate the matrix A on a 2 by 2 logical processor grid. The vertical lines in the matrix A indicate the column blocks of the matrix. Routine X04BUFP is used to output the matrix.

8.1 Example Text

```
*     F01ZWFP Example Program Text
*     NAG Parallel Library Release 2. NAG Copyright 1996.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          M, N
      PARAMETER        (M=6,N=7)
      INTEGER          MG, NG
      PARAMETER        (MG=2,NG=2)
      INTEGER          LDA, TDA
      PARAMETER        (LDA=M,TDA=(N/(MG*NG)+1))
      CHARACTER*20     FORMT
      PARAMETER        (FORMT='F7.4')
*     .. Local Scalars ..
      INTEGER          ICNTXT, ICOFF, IFAIL, MP, NP, NX
      LOGICAL          ROOT
      CHARACTER        CNUMOP, TITOP
*     .. Local Arrays ..
      COMPLEX*16       A(LDA,TDA), W(LDA,TDA)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         F01ZWFP, GMATA, X04BUFP, Z01AAFP, Z01ABFP
*     .. Executable Statements ..
      ROOT = Z01ACFP()
      IF (ROOT) THEN
         WRITE (NOUT,*) 'F01ZWFP Example Program Results'
         WRITE (NOUT,*)
      END IF
```

```
*
*     Define the 2D processor grid
*
      MP = MG
      NP = NG
      IFAIL = 0
*
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      IFAIL = 0
*
*     Generate matrix A
*
      CALL F01ZWFP(ICNTXT,GMATA,M,N,A,LDA,NX,IFAIL)
*
*     Print matrix A
*
      IF (ROOT) THEN
         WRITE (NOUT,*) 'Generated Matrix'
         WRITE (NOUT,*)
         TITOP = 'N'
         CNUMOP = 'G'
      END IF
      ICOFF = 0
      IFAIL = 0
*
      CALL X04BUFP(ICNTXT,NOUT,M,NX,A,LDA,FORMT,TITOP,CNUMOP,ICOFF,W,
     +             LDA,IFAIL)
*
*     Undefine the 2D grid
*
      CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
      STOP
      END
*
      SUBROUTINE GMATA(M,J1,J2,AL,LDAL)
*
*     GMATA generates the block A( 1: M, J1: J2 ) of the matrix A such
*     that
*       a(i,j) = cmplx(i+1,i+1)    if i.eq.j
*       a(i,j) = cmplx(i,j) else
*     in the array AL.
*
*     .. Scalar Arguments ..
      INTEGER          J1, J2, LDAL, M
*     .. Array Arguments ..
      COMPLEX*16       AL(LDAL,*)
*     .. Local Scalars ..
      INTEGER          I, J, L
*     .. Intrinsic Functions ..
      INTRINSIC        DCMPLX
*     .. Executable Statements ..
      L = 1
      DO 40 J = J1, J2
         DO 20 I = 1, M
            IF (I.NE.J) THEN
               AL(I,L) = DCMPLX(I,J)
```

```
               ELSE
                  AL(I,L) = DCMPLX(I+1,I+1)
               END IF
   20       CONTINUE
            L = L + 1
   40    CONTINUE
*
*     End of GMATA.
*
      RETURN
      END
```

8.2 Example Data

None.

8.3 Example Results

```
F01ZWFP Example Program Results

Generated Matrix

                    1                   2
  ( 2.0000, 2.0000) ( 1.0000, 2.0000)
  ( 2.0000, 1.0000) ( 3.0000, 3.0000)
  ( 3.0000, 1.0000) ( 3.0000, 2.0000)
  ( 4.0000, 1.0000) ( 4.0000, 2.0000)
  ( 5.0000, 1.0000) ( 5.0000, 2.0000)
  ( 6.0000, 1.0000) ( 6.0000, 2.0000)

                    3                   4
  ( 1.0000, 3.0000) ( 1.0000, 4.0000)
  ( 2.0000, 3.0000) ( 2.0000, 4.0000)
  ( 4.0000, 4.0000) ( 3.0000, 4.0000)
  ( 4.0000, 3.0000) ( 5.0000, 5.0000)
  ( 5.0000, 3.0000) ( 5.0000, 4.0000)
  ( 6.0000, 3.0000) ( 6.0000, 4.0000)

                    5                   6
  ( 1.0000, 5.0000) ( 1.0000, 6.0000)
  ( 2.0000, 5.0000) ( 2.0000, 6.0000)
  ( 3.0000, 5.0000) ( 3.0000, 6.0000)
  ( 4.0000, 5.0000) ( 4.0000, 6.0000)
  ( 6.0000, 6.0000) ( 5.0000, 6.0000)
  ( 6.0000, 5.0000) ( 7.0000, 7.0000)

                    7
  ( 1.0000, 7.0000)
  ( 2.0000, 7.0000)
  ( 3.0000, 7.0000)
  ( 4.0000, 7.0000)
  ( 5.0000, 7.0000)
  ( 6.0000, 7.0000)
```