

# F01WFPF

## 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

F01WFPF distributes an  $m$  by  $n$  real matrix  $B$  available in its natural form on the  $\{0, 0\}$  logical processor to the processors in the Library Grid. The distributed form of the matrix, denoted by  $A$ , conforms to the cyclic two-dimensional block distribution as required by routines in Chapter F04. On exit, the  $(1, 1)$  element of the matrix  $A$  will be located on the  $\{0, 0\}$  logical processor.

### 2 Specification

```

SUBROUTINE F01WFPF(ICNTXT, M, N, A, LDA, MB, NB, B, LDB, WORK,
1                LWORK, IFAIL)
DOUBLE PRECISION A(LDA,*), B(LDB,*), WORK(*)
INTEGER          ICNTXT, M, N, LDA, MB, NB, LDB, LWORK, 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 $X$ .
$\text{numroc}(\alpha, b_\ell, q, s, k)$	–	a function which gives the <b>number of rows or columns</b> 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 $m_p$ 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, MB, NB, IFAIL

Global output arguments: IFAIL

The remaining arguments are local.

#### 3.3 Distribution Strategy

On exit, the distributed matrix  $A$  is in the cyclic two-dimensional block distribution on the processors in the Library Grid. Each rectangular block is  $M_b^A$  by  $N_b^A$  and each local array A on each logical processor contains the relevant blocks of the matrix. This data distribution is described in more detail in the F04 Chapter Introduction.

## 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:** M — INTEGER *Global Input*  
*On entry:*  $m$ , the number of rows of the matrix  $B$ .  
*Constraint:*  $M \geq 0$ .
- 3:** N — INTEGER *Global Input*  
*On entry:*  $n$ , the number of columns of the matrix  $B$ .  
*Constraint:*  $N \geq 0$ .
- 4:** A(LDA,\*) — DOUBLE PRECISION array *Local Output*  
**Note:** the size of the second dimension of the array A must be at least  $\max(1, \text{numroc}(N, \text{NB}, p_c, 0, n_p))$ .  
*On exit:* the relevant blocks of the distributed matrix  $A$  in the cyclic two-dimensional block distribution format.
- 5:** LDA — INTEGER *Local Input*  
*On entry:* the size of the first dimension of the array A as declared in the (sub)program from which F01WFPF is called.  
*Constraint:*  $LDA \geq \max(1, \text{numroc}(M, \text{MB}, p_r, 0, m_p))$ .
- 6:** MB — INTEGER *Global Input*  
*On entry:*  $M_b^A$ , the blocking factor, for distributing the rows of the matrix  $A$ .  
*Constraint:*  $MB \geq 1$ .
- 7:** NB — INTEGER *Global Input*  
*On entry:*  $N_b^A$ , the blocking factor, for distributing the columns of the matrix  $A$ .  
*Constraint:*  $NB \geq 1$ .
- 8:** B(LDB,\*) — DOUBLE PRECISION array *Local Input*  
**Note:** the size of the second dimension of the array B must be at least  $\max(1, N)$ . B is only referenced on logical processor  $\{0, 0\}$ .  
*On entry:* the matrix  $B$  in its natural form.
- 9:** LDB — INTEGER *Local Input*  
*On entry:* the size of the first dimension of the array B as declared in the (sub)program from which F01WFPF is called.  
*Constraints:*  
 $LDB \geq \max(1, M)$  on logical processor  $\{0, 0\}$ ;  
 $LDB \geq 1$  elsewhere.

**10:** WORK(\*) — DOUBLE PRECISION array *Local Workspace/Global Output*

**Note:** the dimension of the array WORK must be at least  $\max(3, \text{LWORK})$ . WORK is used as a workspace only by the logical processor  $\{0, 0\}$ .

*On exit:*  $\text{WORK}(i) = l_i$ ,  $i = 1, 2, 3$ . See LWORK for the definitions of  $l_i$ . If one of the following conditions are satisfied

IFAIL = 0 on exit or

IFAIL = -10 on exit and  $\text{LWORK} \geq 4$  on entry

then  $l_i$ ,  $i = 1, 2, 3$  give the workspace requirements.

**11:** LWORK — INTEGER *Local Input*

*On entry:* the dimension of the array WORK as declared in the (sub)program from which F01WFPF is called. The minimum requirement for LWORK is  $\max(4, \min(l_1, l_2))$ , but the higher value of  $\max(4, l_3)$  is recommended for higher efficiency where

$$\begin{aligned} l_1 &= \max_{i=0, \dots, m_p-1} \alpha(i) \\ \alpha(i) &= \text{numroc}(M, \text{MB}, i, 0, m_p) \\ l_2 &= \max_{j=0, \dots, n_p-1} \beta(j) \\ \beta(j) &= \text{numroc}(N, \text{NB}, j, 0, n_p) \\ l_3 &= \max_{i=0, \dots, m_p-1} \max_{j=0, \dots, n_p-1} \alpha(i)\beta(j) \end{aligned}$$

**Note:** if  $\text{LWORK} = -1$ , then a workspace query for LWORK is assumed; the routine only calculates the required minimum sizes of the array WORK as defined by  $l_1$ ,  $l_2$  and  $l_3$ . These values are returned in the first three elements of the array WORK.

*Constraint:*  $\text{LWORK} \geq \max[4, \min(l_1, l_2)]$  or  $\text{LWORK} = -1$ .

**12:** 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 which was not returned by a call to Z01AAFP on one or more processors.

IFAIL = -1000

The utility routine Z01AAFP has not been called to define the logical processor grid and initialise the internal variables used by the Library.

IFAIL =  $-i$

On entry, the  $i$ th argument was invalid. 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

For distributing matrices from any arbitrary logical processor, the routine F01WNFP should be used instead of F01WFPF. The routine F01WNFP can also position the (1, 1) element of the matrix  $A$  on any arbitrary logical processor on the Library Grid.

### 6.1 Algorithmic Detail

The performance of the algorithm depends upon the size of LWORK. The critical values of LWORK are  $l_i$ ,  $i = 1, 2, 3$ . See LWORK for the definitions of  $l_i$ . For higher efficiency, LWORK should be set to  $\max(l_3, 4)$  (or greater). However, this routine will work with a workspace size of  $\max(4, \min(l_1, l_2))$ . Note that  $l_3 \geq \max(l_1, l_2)$ .

### 6.2 Parallelism Detail

The logical processor {0,0} sequentially distributes the relevant parts of  $B$  to other processors.

## 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, Philadelphia, PA 19104-2688, USA. URL: [http://www.netlib.org/scalapack/slug/scalapack\\_slug.html](http://www.netlib.org/scalapack/slug/scalapack_slug.html)

## 8 Example

The example program illustrates the distribution of a matrix  $B$ .

### 8.1 Example Text

```
*      F01WFPF Example Program Text
*      NAG Parallel Library Release 3. NAG Copyright 1999.
*      .. Parameters ..
INTEGER          NOUT
PARAMETER       (NOUT=6)
INTEGER          M, N
PARAMETER       (M=10,N=5)
INTEGER          MB, NB
PARAMETER       (MB=2,NB=MB)
INTEGER          NA
PARAMETER       (NA=25)
INTEGER          LDA, TDA, LWORK, LDB, LDC
PARAMETER       (LDA=NA, TDA=NA, LWORK=LDA, LDB=LDA, LDC=LDA)
*      .. Local Scalars ..
INTEGER          I, I1, I2, I3, ICNTXT, IFAIL, J, MP, NP
LOGICAL          ROOT
*      .. Local Arrays ..
DOUBLE PRECISION A(LDA,TDA), B(LDB,TDA), C(LDC,TDA), WORK(LWORK)
*      .. External Functions ..
LOGICAL          Z01ACFP
EXTERNAL         Z01ACFP
```

```

* .. External Subroutines ..
EXTERNAL          F01WBFP, F01WPFP, Z01AAFP, Z01ABFP
* .. Intrinsic Functions ..
INTRINSIC          DBLE, NINT
* .. Executable Statements ..
ROOT = Z01ACFP()
IF (ROOT) THEN
    WRITE (NOUT,*) 'F01WPFP Example Program Results'
    WRITE (NOUT,*)
END IF

*
MP = 2
NP = 2
IFAIL = 0
CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)

*
* Generate a matrix on the root processor
*
IF (ROOT) THEN
    DO 20 J = 1, N
        DO 20 I = 1, M
            B(I,J) = DBLE(I) + DBLE(J)/10000.0D0
20    CONTINUE
END IF

*
* Distribute the matrix from the root processor
*
IFAIL = 0
CALL F01WPFP(ICNTXT,M,N,A,LDA,MB,NB,B,LDB,WORK,LWORK,IFAIL)

*
* Store the values of l(1), l(2) and l(3) in I1, I2 and I3
*
I1 = NINT(WORK(1))
I2 = NINT(WORK(2))
I3 = NINT(WORK(3))

*
* Gather the matrix back to the root processor as the matrix C, and
* print the matrix
*
IFAIL = 0
CALL F01WBFP(ICNTXT,M,N,A,LDA,MB,NB,C,LDC,WORK,LWORK,IFAIL)

*
IF (ROOT) THEN
    WRITE (NOUT,'(1X,"The matrix",/)' )
    DO 40 I = 1, M
        WRITE (NOUT,'(1X,5(F10.4,2X))' ) (C(I,J),J=1,N)
40    CONTINUE
    WRITE (NOUT,*)
END IF

*
* Print the values l(1), l(2) and l(3) that determine the
* recommended dimension of WORK
*
IF (ROOT) THEN
    WRITE (NOUT,
+    '(1X,"The values of l(1), l(2) and l(3) are:",/)' )
    WRITE (NOUT,'(1X,"WORK(1) = ",I3)' ) I1
    WRITE (NOUT,'(1X,"WORK(2) = ",I3)' ) I2

```

```
        WRITE (NOUT, '(1X,"WORK(3) = ",I3)') I3
    END IF

    IFAIL = 0
    CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
    STOP
    END
```

## 8.2 Example Data

None.

## 8.3 Example Results

F01WPF Example Program Results

The matrix

1.0001	1.0002	1.0003	1.0004	1.0005
2.0001	2.0002	2.0003	2.0004	2.0005
3.0001	3.0002	3.0003	3.0004	3.0005
4.0001	4.0002	4.0003	4.0004	4.0005
5.0001	5.0002	5.0003	5.0004	5.0005
6.0001	6.0002	6.0003	6.0004	6.0005
7.0001	7.0002	7.0003	7.0004	7.0005
8.0001	8.0002	8.0003	8.0004	8.0005
9.0001	9.0002	9.0003	9.0004	9.0005
10.0001	10.0002	10.0003	10.0004	10.0005

The values of l(1), l(2) and l(3) are:

```
WORK(1) = 6
WORK(2) = 3
WORK(3) = 18
```

---