F01XTFP
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

F01XTFP distributes a complex dense vector x of length n, stored locally on one or more source processors, conformally to a sparse matrix A (see Section 2.5 of the F11 Chapter Introduction).

An appropriate Chapter F11 routine must have been called prior to F01XTFP in order to set up auxiliary information about the sparse matrix A in the array IAINFO. See Section 3.3 of the F11 Chapter Introduction for further information, particularly Section 3.3.2.

2 Specification

```
 SUBROUTINE F01XTFP(ICNTXT, IS, JS, N, XG, XL, IAINFO, IWORK, WORK,
1                   IFAIL)
 COMPLEX*16         XG(*), XL(*), WORK(*)
 INTEGER            ICNTXT, IS, JS, N, IAINFO(*), IWORK(*), IFAIL
```

3 Usage

3.1 Definitions

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

m_l – the number of vector elements to be stored locally (m_l = IAINFO(3), see IAINFO).
$m_l^{\max}$ – the maximum number of vector elements stored on any processor of the Library Grid ($m_l^{\max}$ = IAINFO(5), see IAINFO).

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: N, IS, JS, IFAIL

Global output arguments: IFAIL

The remaining arguments are local.

3.3 Distribution Strategy

On entry to F01XTFP, the whole vector x must be stored on the 'source' processors, specified by the input parameters IS and JS.

On exit from F01XTFP, the vector x is distributed conformally to the sparse matrix A, i.e., the vector x is distributed across the logical processors in the Library Grid in the same way as each of the columns of the matrix A. The local part of the vector x is stored in the array XL. This data distribution is described in more detail in Section 2.5 of the F11 Chapter Introduction.

3.4 Related Routines

The vectors distributed by F01XTFP can be gathered using the Library routine F01XUFP.

3.5 Requisites

The sparse matrix A must have been preprocessed to set up the auxiliary array IAINFO by an appropriate Chapter F11 routine.

Cyclic row block distribution: F11ZBFP or F11ZPFP for real or complex A, respectively.

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: IS — INTEGER *Global Input*

3: JS — INTEGER *Global Input*

On entry: the Library Grid coordinates of the source processor(s) on which the vector x is stored.

If JS $= -1$, then all processors in row IS of the Library Grid must store on entry a copy of x. Similarly, if IS $= -1$, then all processors in column JS of the Library Grid must store on entry a copy of x.
If IS $=$ JS $= -1$, then all processors in the Library Grid must store on entry a copy of x.

Constraint: $-1 \leq$ IS $\leq m_p - 1$ and $-1 \leq$ JS $\leq n_p - 1$.

4: N — INTEGER *Global Input*

On entry: n, the order of the vector x. It must contain the same value as the parameter N used in a prior call to the Chapter F11 routine in which the array IAINFO was initialised.

Constraint: N ≥ 1.

5: XG($*$) — COMPLEX*16 array *Local Input*

Note: the dimension of the array XG must be at least N on the source processors, as specified by the input parameters IS and JS, on all other processors, it must be at least 1.

On entry: on the source processors, as specified by the input parameters IS and JS, XG must contain the vector x; on all other processors, XG is not referenced.

6: XL($*$) — COMPLEX*16 array *Local Output*

Note: the dimension of the array XL must be at least max(1,m_l).

On exit: the local part of the vector x.

7: IAINFO($*$) — INTEGER array *Local Input*

Note: the dimension of the array IAINFO must be at least max(30,IAINFO(2)).

On entry: the first IAINFO(2) elements of IAINFO contain information about the matrix A. The array IAINFO must have been initialised by a prior call to an appropriate Chapter F11 routine. The first IAINFO(2) elements of IAINFO **must not** be changed between successive calls to library routines involving the sparse matrix A.

Note: on exit from the Chapter F11 routine, the element IAINFO(3) contains m_l, the number of rows of the matrix stored locally, and IAINFO(5) contains $m_l^{\max}$, the maximum number of vector elements stored on any processor of the Library Grid.

8: IWORK($*$) — INTEGER array *Local Workspace*

Note: the dimension of the array IWORK must be at least IAINFO(5) on the source processors, as specified by the input parameters IS and JS, and at least max(1,IAINFO(3)) on all other processors.

9: WORK($*$) — COMPLEX*16 array *Local Workspace*

Note: the dimension of the array WORK must be at least IAINFO(5) on the source processors, as specified by the input parameters IS and JS, and at least 1 on all other processors. The array WORK is only referenced on the source processor(s).

10: 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.

5.2 Any Error Checking Mode

IFAIL = 1

IAINFO was not initialised by a prior call to a Chapter F11 routine.

IFAIL = 2

On entry, the data stored in the arguments N and IAINFO are inconsistent. This could indicate that, after the array IAINFO was initialised, at least one of these arguments was changed between successive calls to library routines.

6 Further Comments

This routine may be used to distribute the data in the form required by a number of the routines in Chapter F11.

6.1 Algorithmic Detail

Each source processor is assigned a corresponding set of (non-source) destination processors. A source processor extracts the elements of the vector x assigned to different destination processors from the array XG and sends them to each destination processor in turn. The set of destination processors assigned to a source processor is determined as follows:

if IS ≥ 0 and JS ≥ 0, then any processor in the Library Grid other than that specified by IS and JS is a destination processor associated to it;

if IS ≥ 0 and JS $= -1$, then any processor in each column of the Library Grid other than that in row IS is a destination processor associated to it;

if IS $= -1$ and JS ≥ 0, then any processor in each row of the Library Grid other than that in column JS is a destination processor associated to it;

if IS $= -1$ and JS $= -1$, then there are no associated destination processors.

Additionally, each source processor extracts the elements of the vector x assigned to it from the array XG and stores them in the array XL.

6.2 Parallelism Detail

The degree to which the distribution of the vector x, described in Section 6.1, can be performed in parallel depends on the specific set of source processors:

if IS ≥ 0 and JS ≥ 0, then execution is essentially sequential, each destination processor being served in turn;

if IS ≥ 0 and JS $= -1$, then different columns of the Library Grid can operate in parallel, wholly independently of each other;

if IS $= -1$ and JS ≥ 0, then different rows of the Library Grid can operate in parallel, wholly independently of each other;

if IS $=$ JS $= -1$, then all processors of the Library Grid can operate in parallel, wholly independently of each other.

7 References

None.

8 Example

See Section 8 of the document for F01XPFP.