F11XAFP

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

Note: you should read the F11 Chapter Introduction before using this routine.

F11XAFP is a set-up routine for the matrix-vector product $y = Ax$, or the transposed matrix-vector product $y = A^T x$, where A is an n by n real sparse matrix stored in coordinate storage format, distributed on a logical grid of processors in cyclic row block form. F11XAFP must be called before F11XBFP, the main computational matrix-vector product routine. A call to F11XAFP must be preceded by a call to F11ZAFP.

2 Specification

```
 SUBROUTINE F11XAFP(ICNTXT, N, NNZ, A, IROW, ICOL, SYMM, OPTIM,
1                   IAINFO, LIA, IFAIL)
 DOUBLE PRECISION   A(*)
 INTEGER            ICNTXT, N, NNZ, IROW(*), ICOL(*), IAINFO(LIA),
1                   LIA, IFAIL
 CHARACTER*1        SYMM, OPTIM
```

3 Data Distribution

3.1 Definitions

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

M_b – the blocking factor for the distribution of the rows of the matrix.

3.2 Global and Local Arguments

The input arguments N, SYMM, OPTIM and IFAIL are global and so must have the same value on entry to the routine on each processor. The output argument IFAIL is global and so will have the same value on exit from the routine on each processor. The remaining arguments are local.

3.3 Distribution Strategy

Blocks of M_b contiguous rows of the matrix A are stored in coordinate storage format on a logical grid of processors cyclically row by row (i.e., in the row major ordering of the grid) starting from the $\{0, 0\}$ logical processor. This data distribution is described in more detail in Section 2.5 of the F11 Chapter Introduction.

This routine assumes that the data has already been correctly distributed, and if this is not the case will fail to produce correct results. However, the Library provides a utility routine, F01YAFP, which assists you in distributing the coordinate storage representation of A correctly. A description of this routine can be found in Chapter F01 of the NAG Parallel Library.

4 Arguments

1: ICNTXT — INTEGER *Local Input*

On entry: the BLACS context used by the communication mechanism, usually returned by a call to Z01AAFP.

2: N — INTEGER *Global Input*

On entry: the order of the matrix A, n. It must contain the same value as the parameter N used in a prior call of F11ZAFP in which the array IAINFO was initialised.

Constraint: N ≥ 1.

3: NNZ — INTEGER *Local Input*

On entry: the number of non-zero elements in the matrix A stored on the calling processor. It must contain the same value as the parameter NNZ returned from a prior call of F11ZAFP in which the array IAINFO was initialised.

Constraint: NNZ ≥ 0.

4: A(∗) — DOUBLE PRECISION array *Local Input*

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

On entry: the non-zero elements in the blocks of the matrix A assigned to the calling processor. The local non-zero elements must have been reordered by a prior call of F11ZAFP.

5: IROW(∗) — INTEGER array *Local Input*

6: ICOL(∗) — INTEGER array *Local Input*

Note: the dimension of the arrays IROW and ICOL must be at least max(1,NNZ).

On entry: the local row and column indices of the non-zero elements supplied in A. The contents of the arrays IROW and ICOL **must not** be changed between successive calls to library routines involving the matrix A.

7: SYMM — CHARACTER*1 *Global Input*

On entry: specifies whether the sparsity pattern (see Section 2.4.2 of the F11 Chapter Introduction) of the matrix A is symmetric or unsymmetric (see Section 6):

if SYMM = 'S', the sparsity pattern of the matrix A is symmetric;
if SYMM} = 'U', the sparsity pattern of the matrix A is unsymmetric.

Note: No attempt is made to verify that the sparsity pattern of A is indeed symmetric or unsymmetric.

Constraint: SYMM = 'S' or 'U'.

8: OPTIM — CHARACTER*1 *Global Input*

On entry: specifies the criterion used for optimising the communication operations required in the matrix-vector product. The only possible choice at this release is:

OPTIM = 'S', the communication operations are optimised such that the overall amount of information transmitted is minimised.

Constraint: OPTIM = 'S'.

9: IAINFO(LIA) — INTEGER array *Local Input/Local Output*

On entry: the first IAINFO(2) elements of IAINFO contain auxiliary information about the matrix A. The array IAINFO must be initialised by a prior call of F11ZAFP.

On exit: auxiliary information about the matrix A including information needed to efficiently perform matrix-vector products $y = Ax$ and transposed matrix-vector products $y = A^T x$. IAINFO(1) contains the maximum number of elements of IAINFO the routine F11ZAFP used or attempted to use. The first IAINFO(2) elements of IAINFO contain information required by other library routines and therefore must not be changed between successive calls to library routines involving the matrix A. See Section 3.2 of the F11 Chapter Introduction

Note: If, on exit, IFAIL = 0, IAINFO(1) contains the maximum number of elements of IAINFO used by F11XAFP and thus provides the minimum required value of LIA. If IFAIL = 3, F11XAFP

attempted to use IAINFO(1) elements of IAINFO at a particular computational stage but found that LIA was too small to accomodate this storage requirement. In this case IAINFO(1) may be **smaller** than the minimum required value of LIA because more than IAINFO(1) elements might have been required to perform subsequent computational steps. Nevertheless, IAINFO(1) still gives a valuable indication of the minimum value of LIA to be used in future program runs.

10: LIA — INTEGER *Local Input*

On entry: the dimension of the array IAINFO as declared in the (sub)program from which F11XAFP is called.

Suggested value: a value in the range IAINFO(2) + $6m_l$ + n_{LB} + 40 to IAINFO(2) + $10m_l$ + n_{LB} + 40 should be adequate for matrices with symmetric sparsity pattern. For matrices with unsymmetric sparsity pattern a value in the range IAINFO(2) + $2m_l + n_{\mathrm{LB}} + 3p + 30$ to IAINFO(2) + $4m_l + n_{\mathrm{LB}} + 3p + 30$ should be adequate. In either case IAINFO(2) is the number of elements of IAINFO used prior to the call of F11XAFP.

Constraint: LIA $\geq$ max(2,IAINFO(2)).

11: IFAIL — INTEGER *Global Input/Global Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (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 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).

Errors detected by the routine:

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 had an invalid value. For global arguments, this may also be caused by an argument not having the same value on all logical processors. An explanatory message distinguishes between these two cases.

IFAIL = 1

IAINFO was not initialised by a prior call of F11ZAFP.

IFAIL = 2

On entry, the data stored in the arguments N, NNZ, IROW, ICOL and IAINFO is inconsistent. This indicates that, after the array IAINFO was initialised by a call of F11ZAFP, at least one of these arguments was changed between successive calls to library routines.

IFAIL = 3

LIA is too small, resulting in insufficient space to store the required auxiliary information in the array IAINFO.

6 Further Comments

If the sparsity pattern of the matrix A is symmetric, the information needed for performing distributed matrix-vector products can be set up independently on each logical processor. In contrast, setting up the necessary information requires communication between logical processors if the sparsity pattern of the matrix A is unsymmetric. Therfore, setting SYMM = 'S' for matrices with symmetric sparsity pattern can lead to a performance improvement.

7 References

[1] Saad Y (1996) *Iterative Methods for Sparse Linear Systems* PWS Publishing Company, Boston, MA

8 Example

See the Example Program for F11BAFP.