F11DAFP

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.

F11DAFP computes incomplete LU factorizations of the local diagonal blocks (see Sections 2.6 and 2.7 of the F11 Chapter Introduction) of an n by n real sparse matrix A, represented in coordinate storage format, distributed on a logical grid of processors in cyclic row block form. The resulting factorized block diagonal matrix may be used as a preconditioner in combination with F11DCFP or F11BBFP.

The local diagonal blocks A_k, $k = 1, 2, \ldots, n_{\mathrm{LB}}$, on each logical processor are decomposed in the form

$$A_k = M_k + R_k$$

where

$$M_k = P_k L_k D_k U_k Q_k$$

and L_k is lower triangular with unit diagonal elements, D_k is diagonal, U_k is upper triangular with unit diagonal elements, P_k and Q_k are permutation matrices, and R_k is a remainder matrix.

The amount of fill-in occurring in each factorization can vary from zero to complete fill, and can be controlled by specifying either the maximum level of fill using the argument LFILL, or the drop tolerance using the argument DTOL.

The argument PSTRAT defines the pivoting strategy to be used within each local diagonal block. The options currently available are no pivoting, user-defined pivoting, partial pivoting by rows for stability, and complete pivoting by columns for sparsity and by rows for stability. The factorizations may optionally be modified to preserve the row-sums of the original local diagonal blocks.

The preconditioning matrices M_k, $k = 1, 2, \ldots, n_{\mathrm{LB}}$, are stored in terms of the coordinate storage representation of the matrices

$$C_k = L_k + D_k^{-1} + U_k - 2I_k,$$

where I_k is the identity matrix of the same order as A_k.

2 Specification

```
 SUBROUTINE F11DAFP(ICNTXT, N, NNZ, A, IROW, ICOL, LFILL, DTOL,
1                   PSTRAT, MILU, IPIVP, IPIVQ,NNZC, C, LC, IROWC,
2                   ICOLC, NPIVM, IAINFO, LIA, IFAIL)
 DOUBLE PRECISION   A(*), DTOL(*), C(LC)
 INTEGER            ICNTXT, N, NNZ, IROW(*), ICOL(*),
1                   LFILL(*), IPIVP(*), IPIVQ(*), NNZC, LC,
2                   IROWC(LC), ICOLC(LC), NPIVM(*), IAINFO(LIA),
3                   LIA, IFAIL
 CHARACTER*1        PSTRAT(*), MILU(*)
```

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.
m_l	–	the number of rows of the matrix assigned to the calling processor (= IAINFO(3), see IAINFO).
n_{LB}	–	the number of row blocks assigned to the calling processor (= IAINFO(8), see IAINFO).

3.2 Global and Local Arguments

The input arguments N 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 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 data correctly. A description of this routine can be found in Chapter F01 of the NAG Parallel Library

The pivot vectors IPIVP and IPIVQ are distributed conformally to the matrix A, i.e., IPIVP and IPIVQ are distributed across the logical processor grid in the same way as each of the columns of the matrix A is.

The aforementioned data distributions are described in more detail in Section 2.5 of the F11 Chapter Introduction.

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: $\mathrm{N} \geq 1$.

3: NNZ — INTEGER *Local Input*

On entry: the number of non-zero elements of 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: $\mathrm{NNZ} \geq 0$.

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. These 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: LFILL(∗) — INTEGER array *Local Input*

Note: the dimension of the array LFILL must be at least $\max(1, n_{\mathrm{LB}})$.

On entry: if $\mathrm{LFILL}(k) \geq 0$, for $k = 1, 2, \ldots, n_{\mathrm{LB}}$, LFILL($k$) is the maximum level of fill allowed in the decomposition of the local diagonal block A_k assigned to the calling processor (see Section 6.3). A negative value of LFILL(k) indicates that DTOL(k) will be used to control the fill instead.

8: DTOL(∗) — DOUBLE PRECISION array *Local Input*

Note: the dimension of the array DTOL must be at least $\max(1, n_{\mathrm{LB}})$.

On entry: if LFILL(k) < 0, for $k = 1, 2, \ldots, n_{\mathrm{LB}}$, then DTOL($k$) is used as a drop tolerance to control the fill-in allowed in the decomposition of the local diagonal block A_k assigned to the calling processor (see Section 6.3). Otherwise DTOL(k) is not referenced.

Constraint: DTOL(k) ≥ 0.0 if LFILL(k) < 0, for $k = 1, 2, \ldots, n_{\mathrm{LB}}$.

9: PSTRAT(∗) — CHARACTER*1 array *Local Input*

Note: the dimension of the array PSTRAT must be at least $\max(1, n_{\mathrm{LB}})$.

On entry: PSTRAT(k), for $k = 1, 2, \ldots, n_{\mathrm{LB}}$, specifies the pivoting strategy to be adopted for the diagonal block A_k as follows:

if PSTRAT(k) = 'N', then no pivoting is carried out;
if PSTRAT(k) = 'U', then pivoting is carried out according to the user-defined input values of IPIVP and IPIVQ;
if PSTRAT(k) = 'P', then partial pivoting by rows for stability is carried out;
if PSTRAT(k) = 'C', then complete pivoting by columns for sparsity, and by rows for stability, is carried out.

Suggested value: PSTRAT(k) = 'C', for $k = 1, 2, \ldots, n_{\mathrm{LB}}$.

Constraint: PSTRAT(k) = 'N', 'U', 'P', or 'C', for $k = 1, 2, \ldots, n_{\mathrm{LB}}$.

10: MILU(∗) — CHARACTER*1 array *Local Input*

Note: the dimension of the array MILU must be at least $\max(1, n_{\mathrm{LB}})$.

On entry: MILU(k), for $k = 1, 2, \ldots, n_{\mathrm{LB}}$, indicates whether or not the factorization of A_k should be modified to preserve row-sums (see Section 6.3):

if MILU(k) = 'M', the factorization is modified;
if MILU(k) = 'N', the factorization is not modified.

Constraint: MILU(k) = 'M' or 'N', for $k = 1, 2, \ldots, n_{\mathrm{LB}}$.

11: IPIVP(∗) — INTEGER array *Local Input/Local Output*

12: IPIVQ(∗) — INTEGER array *Local Input/Local Output*

Note: the dimension of the arrays IPIVP and IPIVQ must be at least $\max(1, m_l)$.

On entry: if PSTRAT(k) = 'U', for $k = 1, 2, \ldots, n_{\mathrm{LB}}$, then the corresponding parts of IPIVP and IPIVQ must specify the local row and column indices of the elements to be used as pivots in the factorization of A_k. Otherwise the parts of IPIVP and IPIVQ corresponding to the k-th diagonal block need not be initialized.

Constraint: if PSTRAT(k) = U, for $k = 1, 2, \ldots, n_{\mathrm{LB}}$, the corresponding parts of IPIVP and IPIVQ must both hold valid permutations of the index set associated with the diagonal block A_k.

On exit: the local row and column indices of the elements used as pivots.

13: NNZC — INTEGER array *Local Output*

On exit: the number of non-zero elements in the matrices C_k, $k = 1, 2, \ldots, n_{\mathrm{LB}}$, assigned to the calling processor.

14: C(LC) — DOUBLE PRECISION array *Local Output*

On exit: the first NNZC entries of C store the non-zero elements of the matrices C_k, $k = 1, 2, \ldots, n_{\mathrm{LB}}$, assigned to the calling processor.

15: LC — INTEGER *Local Input*

On entry: the dimension of the arrays C, IROWC and ICOLC as declared in the (sub)program from which F11DAFP is called.

Suggested value: depending on the arguments LFILL and DTOL, a small multiple of NNZ, LZ ≤ 5 $\times$ NNZ, should be adequate for most problems.

Constraint: LC $\geq \max(1, m_l)$.

16: IROWC(LC) — INTEGER array *Local Output*

17: ICOLC(LC) — INTEGER array *Local Output*

On exit: the first NNZC entries of IROWC and ICOLC store the local row and column indices of the non-zero elements of the matrices C_k, $k = 1, 2, \ldots, n_{\mathrm{LB}}$, returned in C.

18: NPIVM($*$) — INTEGER array *Local Output*

On exit: NPIVM(k), for $k = 1, 2, \ldots, n_{\mathrm{LB}}$, specifies the number of pivots which were modified during the factorization of A_k to ensure that the matrix M_k assigned to the calling processor exists. The quality of the local contribution to the preconditioner will generally depend on the returned value of NPIVM(k). If NPIVM(k) > 0, the local contribution to the preconditioner may not be satisfactory. In this case it may be advantageous to call F11DAFP again with an increased value of LFILL(k), a reduced value of DTOL(k), or PSTRAT(k) set to 'C'.

19: 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.
Note: On exit from F11ZAFP, the elements IAINFO(3) and IAINFO(8) contain m_l and n_{LB}, the number of rows and row blocks of the matrix assigned to the calling processor, respectively.

On exit: auxiliary information about the matrix A including information needed to efficiently solve the equations $Mx = y$ and $M^T x = y$, where M is the block diagonal matrix formed by the diagonal blocks M_k. IAINFO(1) contains the minimum required value of LIA. 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 F11DAFP and thus provides the minimum required value of LIA. If IFAIL = 5, F11DAFP 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.

20: LIA — INTEGER *Local Input*

On entry: the dimension of the array IAINFO as declared in the (sub)program from which F11DAFP is called.

Suggested value: a value of IAINFO(2) + $9m_l$ + 3 should be adequate for most problems, where IAINFO(2) is the number of elements of IAINFO used prior to the call of F11DAFP.

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

21: 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

For at least one $k = 1, 2, \ldots, n_{\mathrm{LB}}$, PSTRAT($k$) = 'U', but the corresponding part of IPIVP and/or IPIVQ does not hold a valid permutation of the index set associated with the diagonal block A_k. An input value of IPIVP or IPIVQ is either out of range or repeated.

IFAIL = 4

LC is too small, resulting in insufficient storage space for the non-zero elements in the local diagonal blocks C_k.

IFAIL = 5

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

IFAIL = 6

A serious error has occurred in an internal call to an auxiliary routine. Check all subroutine calls and array sizes. Seek expert help.

6 Further Comments

6.1 Parallelism Detail

The routine performs all operations on each logical processor independently.

6.2 Accuracy

The accuracy of each factorization will be determined by the size of the elements that are dropped and the size of any modifications made to the pivot elements. If these sizes are small then the computed factors will correspond to matrices close to the local diagonal blocks A_k. The factorizations can generally be made more accurate by increasing LFILL(k), or by reducing DTOL(k) with LFILL(k) < 0.

If F11DAFP is used in combination with F11BBFP or F11DCFP, the more accurate the factorizations the fewer iterations will be required. However, the cost of the decompositions will also generally increase.

6.3 Control of Fill-in

If LFILL(k) ≥ 0 the amount of fill-in occurring in the incomplete factorization of A_k is controlled by limiting the maximum **level** of fill-in to LFILL(k). The original non-zero elements of A_k are defined to be of level 0. The fill level of a new non-zero location occurring during the factorization is defined as:

$$l = \max(l_e, l_c) + 1,$$

where l_e is the level of fill of the element being eliminated, and l_c is the level of fill of the element causing the fill-in.

If LFILL(k) < 0 the fill-in is controlled by means of the **drop tolerance** DTOL(k). A potential fill-in element $a_{ij}^{(k)}$ occurring in row i and column j will not be included if:

$$|a_{ij}^{(k)}| < \text{DTOL}(k) \times \alpha_k,$$

where α_k is the maximum absolute value of any element in the matrix A_k.

For either method of control, any elements which are not included are discarded unless MILU(k) = 'M', in which case their contributions are subtracted from the pivot element in the relevant elimination row, to preserve the row-sums of the original matrix A_k.

6.4 Choice of Parameters

There is unfortunately no choice of the various algorithmic parameters which is optimal for all types of matrices, and some experimentation will generally be required for each new matrix type encountered.

If the local diagonal blocks A_k are not known to have any particular special properties, the following strategy is recommended. Start with LFILL(k) = 0 and PSTRAT(k) = 'C', for $k = 1, 2, \ldots, n_{\text{LB}}$. If a value returned for NPIVM(k) is significantly larger than zero, i.e., a large number of pivot modifications were required to ensure that the local M_k existed, the preconditioner is not likely to be satisfactory. In this case increase LFILL(k) until NPIVM(k) falls to a value close to zero.

If a local diagonal block A_k has non-positive off-diagonal elements, is non-singular, and has only non-negative elements in its inverse, it is called an 'M-matrix'. It can be shown that no pivot modifications are required in the incomplete LU factorization of an M-matrix [1]. In this case a good preconditioner can generally be expected by setting LFILL(k) = 0, PSTRAT(k) = 'N' and MILU(k) = 'M'.

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.