F11YPFP

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

F11YPFP permutes the local part of a complex dense vector x of length n, distributed conformally to a sparse matrix A (see Section 2.5 of the F11 Chapter Introduction), from distribution-based order into matrix-based order (see Section 2.6.2 of the F11 Chapter Introduction).

One of the routines F11ZBFP or F11ZPFP must be called prior to F11YPFP to set up auxiliary information about the sparse matrix A in the array IAINFO.

2 Specification

```
SUBROUTINE F11YPFP(ICNTXT, N, XD, XL, IAINFO, IFAIL)
COMPLEX*16         XD(*), XL(*)
INTEGER            ICNTXT, N, IAINFO(*), 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 stored on the calling processor (m_l = IAINFO(3), see IAINFO).

3.2 Global and Local Arguments

Global input arguments: N, IFAIL

Global output arguments: IFAIL

The remaining arguments are local.

3.3 Distribution Strategy

The vector x must be distributed conformally to the sparse matrix A, i.e., x must be distributed across the Library Grid in the same way as each of the columns of the matrix A is. This data distribution is described in more detail in Section 2.5 of the F11 Chapter Introduction.

3.4 Related Routines

Some Library routines can be used to generate or distribute real dense vectors conformally to a given real or complex sparse matrix.

Real vector generation: F01YTFP

Real vector scatter: F01XTFP

3.5 Requisites

The sparse matrix A must have been preprocessed to set up the auxiliary information vector IAINFO by F11ZBFP (or F11ZPFP if A is complex).

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: N — INTEGER *Global Input*

On entry: n, the order of the matrix A. It must contain the same value as the parameter N used in a prior call to F11ZBFP or F11ZPFP in which the array IAINFO was initialised.

Constraint: N ≥ 1.

3: XD($*$) — COMPLEX*16 array *Local Input*

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

On entry: the local part of the vector x in distribution-based order.

4: 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 in matrix-based order.

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

Note: the dimension of the array IAINFO must be at least max(200,IAINFO(2)).

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 to F11ZBFP or F11ZPFP. The first IAINFO(2) elements of IAINFO **must not** be changed between successive calls to library routines involving the matrix A.

Note: on exit from F11ZBFP or F11ZPFP, the element IAINFO(3) contains m_l, the number of rows of the matrix assigned to the calling processor.

6: 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.

IFAIL = 1

IAINFO was not initialised by a prior call to F11ZBFP or F11ZPFP.

IFAIL = 2

On entry, the data stored in the arguments N and IAINFO is inconsistent. This indicates that, after the array IAINFO was set up by a call to F11ZBFP or F11ZPFP, at least one of these arguments was changed between successive calls to library routines.

6 Further Comments

The routine performs all operations on each logical processor independently.

7 References

None.

8 Example

This example generates a vector x, $x(i) = i$, for $i = 1, \ldots n$, of length $n = n_x^2$ in distribution-based order. It then converts it to matrix-based order and prints the vector on the root processor. The matrix-based order is determined by a sparse matrix A representing a five-point finite-difference approximation to the partial differential equation:

$$c_1 \frac{\partial^2 w}{\partial x^2} + c_2 \frac{\partial^2 w}{\partial y^2} + c_3 \frac{\partial w}{\partial x} + c_4 \frac{\partial w}{\partial y} + c_5 w = f$$

defined on the unit square.

8.1 Example Text

```
*     F11YPFP Example Program Text
*     NAG Parallel Library Release 3. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MLMAX
      PARAMETER        (MLMAX=1000)
      INTEGER          LA
      PARAMETER        (LA=5*MLMAX)
      INTEGER          LIA, LWORK
      PARAMETER        (LIA=-1,LWORK=2*MLMAX)
*     .. Scalars in Common ..
      COMPLEX*16       C1, C2, C3, C4, C5
```

```
      INTEGER          NX
*     .. Local Scalars ..
      INTEGER          ICNTXT, IFAIL, LEVEL, MB, ML, MP, N, NNZ, NP
      LOGICAL          ROOT, ZGRID
      CHARACTER        DUP, KIND, SYMM, ZERO
      CHARACTER*80     FORMAT
*     .. Local Arrays ..
      COMPLEX*16       A(LA), WORK(LWORK), XD(MLMAX), XL(MLMAX)
      INTEGER          CA(1), IAINFO(200), ICOL(LA), IERR(1), IROW(LA),
     +                 RA(1)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         F01CPFP, F01YPFP, F01ZXFP, F11YPFP, F11ZPFP,
     +                 F11ZZFP, GMAT, GVEC, X04YPFP, Z01AAFP, Z01ABFP,
     +                 Z01BBFP, Z02EAFP
*     .. Common blocks ..
      COMMON           /PROB/C1, C2, C3, C4, C5, NX
*     .. Executable Statements ..
      ROOT = Z01ACFP()
      IF (ROOT) WRITE (NOUT,*) 'F11YPFP Example Program Results'
*
*     Open input file on all processors
*
      OPEN (NIN,FILE='f11ypfpe.d')
*
*     Skip heading in data file
*     Read size of processor grid
*
      READ (NIN,*)
      READ (NIN,*) MP
      NP = 1
*
*     Read problem parameters
*
      READ (NIN,*) NX
      N = NX**2
*
*     Read algorithmic parameters
*
      READ (NIN,*) MB
      READ (NIN,*) FORMAT
      READ (NIN,*) LEVEL
*
*     Read complex coefficients in PDE
*
      READ (NIN,*) C1, C2, C3, C4, C5
*
*     Close the input file
*
      CLOSE (NIN)
*
*     Initialize Library Grid
*
      IFAIL = 0
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
```

```
*     Check whether processor is part of the Library Grid
*
      CALL Z01BBFP(ICNTXT,ZGRID,IFAIL)
      IF ( .NOT. ZGRID) GO TO 40
*
*     Set error checking level
*
      CALL Z02EAFP(ICNTXT,LEVEL,IFAIL)
*
*     Generate sparse matrix
*
      CALL F01YPFP(ICNTXT,GMAT,N,MB,NNZ,A,LA,IROW,ICOL,IFAIL)
*
*     Set up auxiliary data for subsequent operations
*
      DUP = 'F'
      ZERO = 'R'
      SYMM = 'S'
      KIND = 'N'
      CALL F11ZPFP(ICNTXT,N,MB,NNZ,A,IROW,ICOL,DUP,ZERO,SYMM,KIND,
     +             IAINFO,LIA,IFAIL)
*
*     Check whether number of rows is less than the corresponding
*     maximum possible value determined by MLMAX
*
      ML = IAINFO(3)
      IERR(1) = 0
      IF (ML.GT.MLMAX) IERR(1) = 1
      CALL F01CPFP(ICNTXT,'X','All',1,1,IERR,1,RA,CA,1,0,-1,-1,IFAIL)
      IF (IERR(1).NE.0) THEN
         IF (ROOT) WRITE (NOUT,99999)
         GO TO 20
      END IF
*
*     Generate vector in array XD in ScaLAPACK/distribution-based order
*
      CALL F01ZXFP(ICNTXT,GVEC,N,1,MB,XD,MLMAX,IFAIL)
*
*     Permute to matrix-based order
*
      CALL F11YPFP(ICNTXT,N,XD,XL,IAINFO,IFAIL)
*
*     Produce report
*
      IF (ROOT) WRITE (NOUT,'(/1X,''Result vector''/1X,13(''-'')/)')
      CALL X04YPFP(ICNTXT,NOUT,N,XL,FORMAT,IAINFO,WORK,IFAIL)
*
*     Release internally allocated memory if necessary
*
   20 IF (LIA.EQ.-1) CALL F11ZZFP(ICNTXT,IAINFO,IFAIL)
*
*     Finalize Library Grid
*
   40 CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
*     End of example program
*
      STOP
```

```
*
99999 FORMAT (1X,'** ERROR: Number of rows per processor too large')
      END


      SUBROUTINE GMAT(I1,I2,N,NNZL,AL,LAL,IROWL,ICOLL)
*
*     This routine generates a block tridiagonal matrix
*     representing the five-point finite difference
*     approximation to the equation:
*
*     c1*w_xx + c2*w_yy + c3*w_x + c4*w_y + c5*w = f
*
*     where the ci are complex coefficients.
*     The right-hand side vector is set up in the
*     routine GVEC.
*
*     .. Scalar Arguments ..
      INTEGER          I1, I2, LAL, N, NNZL
*     .. Array Arguments ..
      COMPLEX*16       AL(LAL)
      INTEGER          ICOLL(LAL), IROWL(LAL)
*     .. Scalars in Common ..
      COMPLEX*16       C1, C2, C3, C4, C5
      INTEGER          NX
*     .. Local Scalars ..
      COMPLEX*16       D1, D2, D3, D4, D5
      DOUBLE PRECISION H, RH, RH2
      INTEGER          I, IX, IY
*     .. Intrinsic Functions ..
      INTRINSIC        DBLE, MOD
*     .. Common blocks ..
      COMMON           /PROB/C1, C2, C3, C4, C5, NX
*     .. Executable Statements ..
*
*     Calculate details of mesh
*
      H = 1/DBLE(NX+1)
      RH = 1.D0/H
      RH2 = RH*RH
*
*     Define stencil coefficient
*
      D1 = -2*RH2*(C1+C2) + C5
      D2 = RH2*C1 + 0.5*RH*C3
      D3 = RH2*C1 - 0.5*RH*C3
      D4 = RH2*C2 + 0.5*RH*C4
      D5 = RH2*C2 - 0.5*RH*C4
*
*     Check whether there is sufficient storage space
*
      IF (LAL.LT.5*(I2-I1+1)) THEN
         NNZL = -1
         RETURN
      END IF
*
      NNZL = 0
      DO 20 I = I1, I2
```

```
*
*     Calculate indices of mesh node
*
         IX = 1 + MOD(I-1,NX)
         IY = 1 + (I-1)/NX
*
*     Set up diagonal elements of matrix first
*
         NNZL = NNZL + 1
         IROWL(NNZL) = I
         ICOLL(NNZL) = I
         AL(NNZL) = D1
*
*     Now add off-diagonal elements where necessary
*
         IF (IX.GT.1) THEN
            NNZL = NNZL + 1
            IROWL(NNZL) = I
            ICOLL(NNZL) = I - 1
            AL(NNZL) = D3
         END IF
*
         IF (IX.LT.NX) THEN
            NNZL = NNZL + 1
            IROWL(NNZL) = I
            ICOLL(NNZL) = I + 1
            AL(NNZL) = D2
         END IF
*
         IF (IY.GT.1) THEN
            NNZL = NNZL + 1
            IROWL(NNZL) = I
            ICOLL(NNZL) = I - NX
            AL(NNZL) = D5
         END IF
         IF (IY.LT.NX) THEN
            NNZL = NNZL + 1
            IROWL(NNZL) = I
            ICOLL(NNZL) = I + NX
            AL(NNZL) = D4
         END IF
*
   20 CONTINUE
*
      RETURN
      END


      SUBROUTINE GVEC(I1,I2,J1,J2,X,LDX)
*
*     .. Scalar Arguments ..
      INTEGER         I1, I2, J1, J2, LDX
*     .. Array Arguments ..
      COMPLEX*16      X(LDX,*)
*     .. Local Scalars ..
      INTEGER         I
*     .. Intrinsic Functions ..
      INTRINSIC       CMPLX, DBLE
```

```
*     .. Executable Statements ..
      DO 20 I = I1, I2
         X(I-I1+1,1) = CMPLX(DBLE(I),-DBLE(I))
   20 CONTINUE
*
      RETURN
      END
```

8.2 Example Data

```
F11YPFP Example Program Data
    4                                   :  MP
    4                                   :  NX
    4                                   :  MB
'(4(:,'' ('',F7.3,'','',F7.3,'')''))'   : FORMAT
   0                                    :  LEVEL
  (1.0, 2.0) (1.0,-1.0)
  (0.0, 3.0) (1.0, 0.0)
  (1.3,-2.2)                            :  C1, C2, C3, C4, C5
```

8.3 Example Results

```
 F11YPFP Example Program Results

 Result vector
 -------------

 (  1.000, -1.000) (  2.000, -2.000) (  3.000, -3.000) (  4.000, -4.000)
 (  5.000, -5.000) (  6.000, -6.000) (  7.000, -7.000) (  8.000, -8.000)
 (  9.000, -9.000) ( 10.000,-10.000) ( 11.000,-11.000) ( 12.000,-12.000)
 ( 13.000,-13.000) ( 14.000,-14.000) ( 15.000,-15.000) ( 16.000,-16.000)
```