F11JPF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

F11JPF solves a system of complex linear equations involving the incomplete Cholesky preconditioning matrix generated by F11JNF.

2 Specification

```
 SUBROUTINE F11JPF(N, A, LA, IROW, ICOL, IPIV, ISTR, CHECK, Y, X,
1                  IFAIL)
 INTEGER           N, LA, IROW(LA), ICOL(LA), IPIV(N), ISTR(N+1),
1                  IFAIL
 complex           A(LA), Y(N), X(N)
 CHARACTER*1       CHECK
```

3 Description

This routine solves a system of linear equations

$$Mx = y$$

involving the preconditioning matrix $M = PLDL^HP^T$, corresponding to an incomplete Cholesky decomposition of a complex sparse Hermitian matrix stored in symmetric coordinate storage (SCS) format (see Section 2.1.2 of the Chapter Introduction), as generated by F11JNF.

In the above decomposition L is a complex lower triangular sparse matrix with unit diagonal, D is a real diagonal matrix and P is a permutation matrix. L and D are supplied to F11JPF through the matrix

$$C = L + D^{-1} - I$$

which is a lower triangular n by n complex sparse matrix, stored in SCS format, as returned by F11JNF. The permutation matrix P is returned from F11JNF via the array IPIV.

F11JPF may also be used in combination with F11JNF to solve a sparse complex Hermitian positive-definite system of linear equations directly (see Section 8.4 of the document for F11JNF). This is illustrated in Section 9.

4 References

None.

5 Parameters

1: N — INTEGER *Input*

On entry: n, the order of the matrix M. This **must** be the same value as was supplied in the preceding call to F11JNF.

Constraint: N ≥ 1.

2: A(LA) — ***complex*** array *Input*

On entry: the values returned in array A by a previous call to F11JNF.

3: LA — INTEGER *Input*

On entry: the dimension of the arrays A, IROW and ICOL as declared in the (sub)program from which F11JPF is called. This **must** be the same value as was supplied in the preceding call to F11JNF.

4: IROW(LA) — INTEGER array *Input*
5: ICOL(LA) — INTEGER array *Input*
6: IPIV(N) — INTEGER array *Input*
7: ISTR(N+1) — INTEGER array *Input*

On entry: the values returned in arrays IROW, ICOL, IPIV and ISTR by a previous call to F11JNF.

8: CHECK — CHARACTER*1 *Input*

On entry: specifies whether or not the input data should be checked:

if CHECK = 'C', checks are carried out on the values of N, IROW, ICOL, IPIV and ISTR;
if CHECK = 'N', none of these checks are carried out.

Constraint: CHECK = 'C' or 'N'.

9: Y(N) — ***complex*** array *Input*

On entry: the right-hand side vector y.

10: X(N) — ***complex*** array *Output*

On exit: the solution vector x.

11: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, CHECK $\neq$ 'C' or 'N'.

IFAIL = 2

On entry, N < 1.

IFAIL = 3

On entry, the SCS representation of the preconditioning matrix M is invalid. Further details are given in the error message. Check that the call to F11JPF has been preceded by a valid call to F11JNF and that the arrays A, IROW, ICOL, IPIV and ISTR have not been corrupted between the two calls.

7 Accuracy

The computed solution x is the exact solution of a perturbed system of equations $(M+\delta M)x = y$, where

$$|\delta M| \leq c(n)\epsilon P|L||D||L^H|P^T,$$

$c(n)$ is a modest linear function of n, and ϵ is the ***machine precision***.

8 Further Comments

8.1 Timing

The time taken for a call to F11JPF is proportional to the value of NNZC returned from F11JNF.

9 Example

This example program reads in a complex sparse Hermitian positive-definite matrix A and a vector y. It then calls F11JNF, with LFILL $= -1$ and DTOL $= 0.0$, to compute the **complete** Cholesky decomposition of A:

$$A = PLDL^H P^T.$$

Finally it calls F11JPF to solve the system

$$PLDL^H P^T x = y.$$

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     F11JPF Example Program Text.
*     Mark 19 Release. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LA, LIWORK
      PARAMETER        (NMAX=1000,LA=10000,LIWORK=2*LA+7*NMAX+1)
*     .. Local Scalars ..
      real             DSCALE, DTOL
      INTEGER          I, IFAIL, LFILL, N, NNZ, NNZC, NPIVM
      CHARACTER        CHECK, MIC, PSTRAT
*     .. Local Arrays ..
      complex          A(LA), X(NMAX), Y(NMAX)
      INTEGER          ICOL(LA), IPIV(NMAX), IROW(LA), ISTR(NMAX+1),
     +                 IWORK(LIWORK)
*     .. External Subroutines ..
      EXTERNAL         F11JNF, F11JPF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F11JPF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
*
*     Read order of matrix and number of non-zero entries
*
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
         READ (NIN,*) NNZ
*
*        Read the matrix A
*
         DO 20 I = 1, NNZ
            READ (NIN,*) A(I), IROW(I), ICOL(I)
   20    CONTINUE
*
*        Read the vector y
*
         READ (NIN,*) (Y(I),I=1,N)
```

```
*
*        Calculate Cholesky factorization
*
         LFILL = -1
         DTOL = 0.0e0
         MIC = 'N'
         DSCALE = 0.0e0
         PSTRAT = 'M'
         IFAIL = 0
*
         CALL F11JNF(N,NNZ,A,LA,IROW,ICOL,LFILL,DTOL,MIC,DSCALE,PSTRAT,
     +               IPIV,ISTR,NNZC,NPIVM,IWORK,LIWORK,IFAIL)
*
*        Check the output value of NPIVM
*
         IF (NPIVM.NE.0) THEN
*
            WRITE (NOUT,*) 'Factorization is not complete'
*
         ELSE
*                          H T
*           Solve P L D L P x = y
*
            CHECK = 'C'
*
            CALL F11JPF(N,A,LA,IROW,ICOL,IPIV,ISTR,CHECK,Y,X,IFAIL)
*
*           Output results
*
            WRITE (NOUT,*) 'Solution of linear system'
            DO 40 I = 1, N
               WRITE (NOUT,99999) X(I)
   40       CONTINUE
*
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,'(',e16.4,',',e16.4,')')
      END
```

9.2 Program Data

```
F11JPF Example Program Data
  9                   N
 23                   NNZ
 ( 6., 0.)   1    1
 (-1., 1.)   2    1
 ( 6., 0.)   2    2
 ( 0., 1.)   3    2
 ( 5., 0.)   3    3
 ( 5., 0.)   4    4
 ( 2.,-2.)   5    1
 ( 4., 0.)   5    5
 ( 1., 1.)   6    3
 ( 2., 0.)   6    4
 ( 6., 0.)   6    6
```

```
(-4., 3.)   7    2
( 0., 1.)   7    5
(-1., 0.)   7    6
( 6., 0.)   7    7
(-1.,-1.)   8    4
( 0.,-1.)   8    6
( 9., 0.)   8    8
( 1., 3.)   9    1
( 1., 2.)   9    5
(-1., 0.)   9    6
( 1., 4.)   9    8
( 9., 0.)   9    9    A(I), IROW(I), ICOL(I), I=1,...,NNZ
( 8.,54.) (-10.,-92.)
(25.,27.) (26., -28.)
(54.,12.) (26.,-22.)
(47.,65.) (71.,-57.)
(60.,70.)             Y(I), I=1,...,N
```

9.3 Program Results

```
F11JPF Example Program Results
Solution of linear system
(      0.1000E+01,      0.9000E+01)
(      0.2000E+01,     -0.8000E+01)
(      0.3000E+01,      0.7000E+01)
(      0.4000E+01,     -0.6000E+01)
(      0.5000E+01,      0.5000E+01)
(      0.6000E+01,     -0.4000E+01)
(      0.7000E+01,      0.3000E+01)
(      0.8000E+01,     -0.2000E+01)
(      0.9000E+01,      0.1000E+01)
```