NAG Library Routine Document

F11JPF

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, IFAIL)

INTEGER              N, LA, IROW(LA), ICOL(LA), IPIV(N), ISTR(N+1), IFAIL
COMPLEX (KIND=nag_wp) A(LA), Y(N), X(N)
CHARACTER(1)         CHECK
```

3   Description

F11JPF solves a system of linear equations

$$Mx = y$$

involving the preconditioning matrix $M = PLDL^{\mathrm{H}}P^{\mathrm{T}}$, corresponding to an incomplete Cholesky decomposition of a complex sparse Hermitian matrix stored in symmetric coordinate storage (SCS) format (see Section 2.1.2 in the F11 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 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 $\geq$ 1.

2:    A(LA) – COMPLEX (KIND=nag_wp) array   *Input*

On entry: the values returned in the 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 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.
>
> CHECK = 'C'
> > Checks are carried out on the values of N, IROW, ICOL, IPIV and ISTR.
>
> CHECK = 'N'
> > None of these checks are carried out.
>
> *Constraint*: CHECK = 'C' or 'N'.

9:     Y(N) – COMPLEX (KIND=nag_wp) array   *Input*

> *On entry*: the right-hand side vector y.

10:    X(N) – COMPLEX (KIND=nag_wp) array   *Output*

> *On exit*: the solution vector x.

11:    IFAIL – INTEGER   *Input/Output*

> *On entry*: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.
>
> For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**
>
> *On exit*: IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

6   Error Indicators 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 or warnings 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

$$|\delta M| \le c(n)\epsilon P|L||D|\big|L^{\mathrm{H}}\big|P^{\mathrm{T}},$$

$c(n)$ 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 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:

$$A = PLDL^{\mathrm{H}}P^{\mathrm{T}}.$$

Finally it calls F11JPF to solve the system

$$PLDL^{\mathrm{H}}P^{\mathrm{T}}x = y.$$

9.1 Program Text

```
      Program f11jpfe

!     F11JPF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: f11jnf, f11jpf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter             :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)             :: dscale, dtol
      Integer                        :: i, ifail, la, lfill, liwork, n, nnz, &
                                        nnzc, npivm
      Character (1)                  :: check, mic, pstrat
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:), x(:), y(:)
      Integer, Allocatable           :: icol(:), ipiv(:), irow(:), istr(:),  &
                                        iwork(:)
!     .. 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
      Read (nin,*) nnz
      la = 3*nnz
      liwork = 2*la + 7*n + 1
      Allocate (a(la),x(n),y(n),icol(la),ipiv(n),irow(la),istr(n+1), &
        iwork(liwork))

!     Read the matrix A
```

```
        Do i = 1, nnz
          Read (nin,*) a(i), irow(i), icol(i)
        End Do

!     Read the vector y

        Read (nin,*) y(1:n)

!     Calculate Cholesky factorization

        lfill = -1
        dtol = 0.0E0_nag_wp
        mic = 'N'
        dscale = 0.0E0_nag_wp
        pstrat = 'M'

!     ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        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/=0) Then

          Write (nout,*) 'Factorization is not complete'

        Else

!       Solve P L D L^H P^T x = y

          check = 'C'

          ifail = 0
          Call f11jpf(n,a,la,irow,icol,ipiv,istr,check,y,x,ifail)

!       Output results

          Write (nout,*) 'Solution of linear system'
          Write (nout,99999) x(1:n)
        End If

99999 Format (1X,'(',E16.4,',',E16.4,')')
    End Program f11jpfe
```

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)
```