NAG Fortran Library Routine Document

F11DPF

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

F11DPF solves a system of complex linear equations involving the incomplete LU preconditioning matrix generated by F11DNF.

2 Specification

```
 SUBROUTINE F11DPF(TRANS, N, A, LA, IROW, ICOL, IPIVP, IPIVQ, ISTR,
1                  IDIAG, CHECK, Y, X, IFAIL)
 INTEGER           N, LA, IROW(LA), ICOL(LA), IPIVP(N), IPIVQ(N),
1                  ISTR(N+1), IDIAG(N), IFAIL
 complex           A(LA), Y(N), X(N)
 CHARACTER*1       TRANS, CHECK
```

3 Description

This routine solves a system of complex linear equations

$$Mx = y, \quad \text{or} \quad M^T x = y,$$

according to the value of the argument TRANS, where the matrix $M = PLDUQ$ corresponds to an incomplete LU decomposition of a complex sparse matrix stored in coordinate storage (CS) format (see Section 2.1.1 of the F11 Chapter Introduction), as generated by F11DNF.

In the above decomposition L is a lower triangular sparse matrix with unit diagonal elements, D is a diagonal matrix, U is an upper triangular sparse matrix with unit diagonal elements and, P and Q are permutation matrices. L, D and U are supplied to F11DPF through the matrix

$$C = L + D^{-1} + U - 2I$$

which is an N by N sparse matrix, stored in CS format, as returned by F11DNF. The permutation matrices P and Q are returned from F11DNF via the arrays IPIVP and IPIVQ.

It is envisaged that a common use of F11DPF will be to carry out the preconditioning step required in the application of F11BSF to sparse complex linear systems. F11DPF is used for this purpose by the black-box routine F11DQF.

F11DPF may also be used in combination with F11DNF to solve a sparse system of complex linear equations directly (see Section 8.5 of the document for F11DNF). This use of F11DPF is illustrated in Section 9.

4 References

None.

5 Parameters

1: TRANS – CHARACTER*1 *Input*

On entry: specifies whether or not the matrix M is transposed:

if TRANS = 'N', then $Mx = y$ is solved;

if TRANS = 'T', then $M^T x = y$ is solved.

Constraint: TRANS = 'N' or 'T'.

2: 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 F11DNF.

Constraint: N $\geq$ 1.

3: A(LA) – ***complex*** array *Input*

On entry: the values returned in the array A by a previous call to F11DNF.

4: LA – INTEGER *Input*

On entry: the dimension of the arrays A, IROW and ICOL as declared in the (sub)program from which F11DPF is called. This **must** be the same value as was supplied in the preceding call to F11DNF.

5: IROW(LA) – INTEGER array *Input*
6: ICOL(LA) – INTEGER array *Input*
7: IPIVP(N) – INTEGER array *Input/Output*
8: IPIVQ(N) – INTEGER array *Input/Output*
9: ISTR(N+1) – INTEGER array *Input*
10: IDIAG(N) – INTEGER array *Input*

On entry: the values returned in arrays IROW, ICOL, IPIVP, IPIVQ, ISTR and IDIAG by a previous call to F11DAF.

On exit: IPIVP and IPIVQ are used as internal workspace prior to being restored and hence are unchanged.

11: CHECK – CHARACTER*1 *Input*

On entry: specifies whether or not the CS representation of the matrix M should be checked:

if CHECK = 'C', checks are carried on the values of N, IROW, ICOL, IPIVP, IPIVQ, ISTR and IDIAG;

if CHECK = 'N', none of these checks are carried out.

See also Section 8.2.

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

12: Y(N) – ***complex*** array *Input*

On entry: the right-hand side vector y.

13: X(N) – ***complex*** array *Output*

On exit: the solution vector x.

14: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

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

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, for users 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.**

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, TRANS $\neq$ 'N' or 'T',
or CHECK $\neq$ 'C' or 'N'.

IFAIL $= 2$

On entry, N < 1.

IFAIL $= 3$

On entry, the CS representation of the preconditioning matrix M is invalid. Further details are given in the error message. Check that the call to F11DPF has been preceded by a valid call to F11DNF and that the arrays A, IROW, ICOL, IPIVP, IPIVQ, ISTR and IDIAG have not been corrupted between the two calls.

7 Accuracy

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

$$|\delta M| \leq c(n)\epsilon P|L||D||U|Q,$$

$c(n)$ is a modest linear function of n, and ϵ is the ***machine precision***. An equivalent result holds when TRANS $=$ 'T'.

8 Further Comments

8.1 Timing

The time taken for a call to F11DPF is proportional to the value of NNZC returned from F11DNF.

8.2 Use of CHECK

It is expected that a common use of F11DPF will be to carry out the preconditioning step required in the application of F11BSF to sparse complex linear systems. In this situation F11DPF is likely to be called many times with the same matrix M. In the interests of both reliability and efficiency, you are recommended to set CHECK to 'C' for the first of such calls, and to 'N' for all subsequent calls.

9 Example

This example program reads in a complex sparse non-Hermitian matrix A and a vector y. It then calls F11DNF, with LFILL $= -1$ and DTOL $= 0.0$, to compute the **complete** LU decomposition

$$A = PLDUQ.$$

Finally it calls F11DPF to solve the system

$$PLDUQx = y.$$

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.

```
*     F11DPF Example Program Text.
*     Mark 20 Revised. NAG Copyright 2001.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LA, LIWORK
      PARAMETER        (NMAX=1000,LA=10000,LIWORK=7*NMAX+2)
*     .. Local Scalars ..
      real             DTOL
      INTEGER          I, IFAIL, LFILL, N, NNZ, NNZC, NPIVM
      CHARACTER        CHECK, MILU, PSTRAT, TRANS
*     .. Local Arrays ..
      complex          A(LA), X(NMAX), Y(NMAX)
      INTEGER          ICOL(LA), IDIAG(NMAX), IPIVP(NMAX), IPIVQ(NMAX),
     +                 IROW(LA), ISTR(NMAX+1), IWORK(LIWORK)
*     .. External Subroutines ..
      EXTERNAL         F11DNF, F11DPF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F11DPF Example Program Results'
      WRITE (NOUT,*)
*     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 LU factorization
*
         LFILL = -1
         DTOL = 0.0e0
         PSTRAT = 'C'
         MILU = 'N'
         IFAIL = 0
*
         CALL F11DNF(N,NNZ,A,LA,IROW,ICOL,LFILL,DTOL,PSTRAT,MILU,IPIVP,
     +               IPIVQ,ISTR,IDIAG,NNZC,NPIVM,IWORK,LIWORK,IFAIL)
*
*        Check value of NPIVM
*
         IF (NPIVM.GT.0) THEN
*
            WRITE (NOUT,*) 'Factorization is not complete'
*
         ELSE
*
*           Solve P L D U x = y
*
            TRANS = 'N'
            CHECK = 'C'
*
            CALL F11DPF(TRANS,N,A,LA,IROW,ICOL,IPIVP,IPIVQ,ISTR,IDIAG,
     +                  CHECK,Y,X,IFAIL)
```

```
*
*           Output results
*
            WRITE (NOUT,*) 'Solution of linear system'
            DO 40 I = 1, N
               WRITE (NOUT,'(1X,''('',D16.4,'','',D16.4,'')'')') X(I)
   40       CONTINUE
*
         END IF
*
      END IF
      STOP
      END
```

9.2 Program Data

```
F11DPF Example Program Data
  4                       N
 11                       NNZ
 ( 1., 2.)   1    2
 ( 1., 3.)   1    3
 (-1.,-3.)   2    1
 ( 2., 0.)   2    3
 ( 0., 4.)   2    4
 ( 3., 4.)   3    1
 (-2., 0.)   3    4
 ( 1.,-1.)   4    1
 (-2.,-1.)   4    2
 ( 1., 0.)   4    3
 ( 1., 3.)   4    4      A(I), IROW(I), ICOL(I), I=1,...,NNZ
 (  5.0, 14.0)
 ( 21.0,  5.0)
 (-21.0, 18.0)
 ( 14.0,  4.0)           Y(I), I=1,...,N
```

9.3 Program Results

```
 F11DPF Example Program Results

 Solution of linear system
 (      0.1000E+01,      0.4000E+01)
 (      0.2000E+01,      0.3000E+01)
 (      0.3000E+01,     -0.2000E+01)
 (      0.4000E+01,     -0.1000E+01)
```