NAG Fortran Library Routine Document

F04CJF

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

F04CJF computes the solution to a complex system of linear equations $AX = B$, where A is an n by n complex Hermitian matrix, stored in packed format and X and B are n by r matrices. An estimate of the condition number of A and an error bound for the computed solution are also returned.

2 Specification

```
      SUBROUTINE F04CJF (UPLO, N, NRHS, AP, IPIV, B, LDB, RCOND, ERRBND,
     1                   IFAIL)
      INTEGER            N, NRHS, IPIV(*), LDB, IFAIL
      double precision   RCOND, ERRBND
      complex*16         AP(*), B(LDB,*)
      CHARACTER*1        UPLO
```

3 Description

The diagonal pivoting method is used to factor A as $A = UDU^H$, if UPLO = 'U', or $A = LDL^H$, if UPLO = 'L', where U (or L) is a product of permutation and unit upper (lower) triangular matrices, and D is Hermitian and block diagonal with 1 by 1 and 2 by 2 diagonal blocks. The factored form of A is then used to solve the system of equations $AX = B$.

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia URL: http://www.netlib.org/lapack/lug

Higham N J (2002) *Accuracy and Stability of Numerical Algorithms* (2nd Edition) SIAM, Philadelphia

5 Parameters

1: UPLO – CHARACTER*1 *Input*

On entry: if UPLO = 'U', the upper triangle of the matrix A is stored, if UPLO = 'L', the lower triangle of the matrix A is stored.

Constraint: UPLO = 'U' or 'L'.

2: N – INTEGER *Input*

On entry: the number of linear equations n, i.e., the order of the matrix A.

Constraint: N ≥ 0.

3: NRHS – INTEGER *Input*

On entry: the number of right-hand sides r, i.e., the number of columns of the matrix B.

Constraint: NRHS ≥ 0.

4: AP($*$) – ***complex*16*** array *Input/Output*

Note: the dimension of the array AP must be at least $\max(1, \mathrm{N} \times (\mathrm{N}+1)/2)$.

On entry: the n by n Hermitian matrix A, packed columnwise in a linear array. The jth column of the matrix A is stored in the array AP as follows:

if UPLO = 'U', $\mathrm{AP}(i + (j-1)j/2) = a_{ij}$ for $1 \le i \le j$;
if UPLO = 'L', $\mathrm{AP}(i + (j-1)(2n-j)/2) = a_{ij}$ for $j \le i \le n$.

See Section 8 below for further details.

On exit: if IFAIL ≥ 0, the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $A = UDU^H$ or $A = LDL^H$ as computed by F07PRF (ZHPTRF), stored as a packed triangular matrix in the same storage format as A.

5: IPIV($*$) – INTEGER array *Output*

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

On exit: if IFAIL ≥ 0, details of the interchanges and the block structure of D, as determined by F07PRF (ZHPTRF).

If $\mathrm{IPIV}(k) > 0$, then rows and columns k and $\mathrm{IPIV}(k)$ were interchanged, and d_{kk} is a 1 by 1 diagonal block;
if UPLO = 'U' and $\mathrm{IPIV}(k) = \mathrm{IPIV}(k-1) < 0$, then rows and columns $k-1$ and $-\mathrm{IPIV}(k)$ were interchanged and $d_{k-1:k,k-1:k}$ is a 2 by 2 diagonal block;
if UPLO = 'L' and $\mathrm{IPIV}(k) = \mathrm{IPIV}(k+1) < 0$, then rows and columns $k+1$ and $-\mathrm{IPIV}(k)$ were interchanged and $d_{k:k+1,k:k+1}$ is a 2 by 2 diagonal block.

6: B(LDB,$*$) – ***complex*16*** array *Input/Output*

Note: the second dimension of the array B must be at least $\max(1, \mathrm{NRHS})$. To solve the equations $Ax = b$, where b is a single right-hand side, B may be supplied as a one-dimensional array with length $\mathrm{LDB} = \max(1, \mathrm{N})$.

On entry: the n by r matrix of right-hand sides B.

On exit: if IFAIL $= 0$ or $\mathrm{N}+1$, the n by r solution matrix X.

7: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F04CJF is called.

Constraint: $\mathrm{LDB} \ge \max(1, \mathrm{N})$.

8: RCOND – ***double precision*** *Output*

On exit: if IFAIL ≥ 0, an estimate of the reciprocal of the condition number of the matrix A, computed as $\mathrm{RCOND} = 1/\left(\|A\|_1 \|A^{-1}\|_1\right)$.

9: ERRBND – ***double precision*** *Output*

On exit: if IFAIL $= 0$ or $\mathrm{N}+1$, an estimate of the forward error bound for a computed solution $\hat{x}$, such that $\|\hat{x} - x\|_1 / \|x\|_1 \le \mathrm{ERRBND}$, where $\hat{x}$ is a column of the computed solution returned in the array B and x is the corresponding column of the exact solution X. If RCOND is less than ***machine precision***, then ERRBND is returned as unity.

10: 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 < 0 and IFAIL $\neq -999$

If IFAIL $= -i$, the ith argument had an illegal value.

IFAIL $= -999$

Allocation of memory failed. The ***double precision*** allocatable memory required is N, and the ***complex*16*** allocatable memory required is $2 \times$ N. Allocation failed before the solution could be computed.

IFAIL > 0 and IFAIL $\leq$ N

If IFAIL $= i$, d_{ii} is exactly zero. The factorization has been completed, but the block diagonal matrix D is exactly singular, so the solution could not be computed.

IFAIL $=$ N $+ 1$

RCOND is less than ***machine precision***, so that the matrix A is numerically singular. A solution to the equations $AX = B$ has nevertheless been computed.

7 Accuracy

The computed solution for a single right-hand side, $\hat{x}$, satisfies an equation of the form

$$(A + E)\hat{x} = b,$$

where

$$\|E\|_1 = O(\epsilon)\|A\|_1$$

and ϵ is the ***machine precision***. An approximate error bound for the computed solution is given by

$$\frac{\|\hat{x} - x\|_1}{\|x\|_1} \leq \kappa(A)\frac{\|E\|_1}{\|A\|_1},$$

where $\kappa(A) = \left\|A^{-1}\right\|_1\|A\|_1$, the condition number of A with respect to the solution of the linear equations. F04CJF uses the approximation $\|E\|_1 = \epsilon\|A\|_1$ to estimate ERRBND. See Section 4.4 of Anderson *et al.* (1999) for further details.

8 Further Comments

The packed storage scheme is illustrated by the following example when $n = 4$ and UPLO = 'U'. Two-dimensional storage of the Hermitian matrix A:

$$\begin{array}{cccc} a_{11} & a_{12} & a_{13} & a_{14} \\ & a_{22} & a_{23} & a_{24} \\ & & a_{33} & a_{34} \\ & & & a_{44} \end{array} \quad \left(a_{ij} = \bar{a}_{ji}\right)$$

Packed storage of the upper triangle of A:

$$AP = [\,a_{11},\quad a_{12},\quad a_{22},\quad a_{13},\quad a_{23},\quad a_{33},\quad a_{14},\quad a_{24},\quad a_{34},\quad a_{44}\,]$$

The total number of floating-point operations required to solve the equations $AX = B$ is proportional to $\left(\frac{1}{3}n^3 + 2n^2 r\right)$. The condition number estimation typically requires between four and five solves and never more than eleven solves, following the factorization.

In practice the condition number estimator is very reliable, but it can underestimate the true condition number; see Section 15.3 of Higham (2002) for further details.

Routine F04DJF is for complex symmetric matrices, and the real analogue of F04CJF is F04BJF.

9 Example

To solve the equations

$$AX = B,$$

where A is the Hermitian indefinite matrix

$$A = \begin{pmatrix} -1.84 & 0.11 - 0.11i & -1.78 - 1.18i & 3.91 - 1.50i \\ 0.11 + 0.11i & -4.63 & -1.84 + 0.03i & 2.21 + 0.21i \\ -1.78 + 1.18i & -1.84 - 0.03i & -8.87 & 1.58 - 0.90i \\ 3.91 + 1.50i & 2.21 - 0.21i & 1.58 + 0.90i & -1.36 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 2.98 - 10.18i & 28.68 - 39.89i \\ -9.58 + 3.88i & -24.79 - 8.40i \\ -0.77 - 16.05i & 4.23 - 70.02i \\ 7.79 + 5.48i & -35.39 + 18.01i \end{pmatrix}.$$

An estimate of the condition number of A and an approximate error bound for the computed solutions are also printed.

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.

```
*     F04CJF Example Program Text
*     Mark 21 Release. NAG Copyright 2004.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, NRHSMX
      PARAMETER        (NMAX=8,NRHSMX=2)
      INTEGER          LDB
      PARAMETER        (LDB=NMAX)
      CHARACTER        UPLO
      PARAMETER        (UPLO='U')
*     .. Local Scalars ..
      DOUBLE PRECISION ERRBND, RCOND
      INTEGER          I, IERR, IFAIL, J, N, NRHS
*     .. Local Arrays ..
      COMPLEX *16      AP((NMAX*(NMAX+1))/2), B(LDB,NRHSMX)
      INTEGER          IPIV(NMAX)
      CHARACTER        CLABS(1), RLABS(1)
*     .. External Subroutines ..
      EXTERNAL         F04CJF, X04DBF, X04DDF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F04CJF Example Program Results'
      WRITE (NOUT,*)
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, NRHS
```

```
      IF (N.LE.NMAX .AND. NRHS.LE.NRHSMX) THEN
*
*        Read the upper or lower triangular part of the matrix A from
*        data file
*
         IF (UPLO.EQ.'U') THEN
            READ (NIN,*) ((AP(I+(J*(J-1))/2),J=I,N),I=1,N)
         ELSE IF (UPLO.EQ.'L') THEN
            READ (NIN,*) ((AP(I+((2*N-J)*(J-1))/2),J=1,I),I=1,N)
         END IF
*
*        Read B from data file
*
         READ (NIN,*) ((B(I,J),J=1,NRHS),I=1,N)
*
*        Solve the equations AX = B for X
*
         IFAIL = -1
         CALL F04CJF(UPLO,N,NRHS,AP,IPIV,B,LDB,RCOND,ERRBND,IFAIL)
*
         IF (IFAIL.EQ.0) THEN
*
*           Print solution, estimate of condition number and approximate
*           error bound
*
            IERR = 0
            CALL X04DBF('General',' ',N,NRHS,B,LDB,'Bracketed',' ',
     +                  'Solution','Integer',RLABS,'Integer',CLABS,80,0,
     +                  IERR)
*
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Estimate of condition number'
            WRITE (NOUT,99999) 1.0D0/RCOND
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     +        'Estimate of error bound for computed solutions'
            WRITE (NOUT,99999) ERRBND
         ELSE IF (IFAIL.EQ.N+1) THEN
*
*           Matrix A is numerically singular.  Print estimate of
*           reciprocal of condition number and solution
*
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Estimate of reciprocal of condition number'
            WRITE (NOUT,99999) RCOND
*
            WRITE (NOUT,*)
            IERR = 0
            CALL X04DBF('General',' ',N,NRHS,B,LDB,'Bracketed',' ',
     +                  'Solution','Integer',RLABS,'Integer',CLABS,80,0,
     +                  IERR)
*
         ELSE IF (IFAIL.GT.0 .AND. IFAIL.LE.N) THEN
*
*           The upper triangular matrix U is exactly singular.  Print
*           details of factorization
*
            WRITE (NOUT,*)
            IERR = 0
            CALL X04DDF(UPLO,'Non-unit diagonal',N,AP,'Bracketed',' ',
     +                  'Details of factorization','Integer',RLABS,
     +                  'Integer',CLABS,80,0,IERR)
*
*           Print pivot indices
*
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Pivot indices'
            WRITE (NOUT,99998) (IPIV(I),I=1,N)
         END IF
      ELSE
         WRITE (NOUT,*) 'NMAX and/or NRHSMX too small'
```

```
      END IF
      STOP
*
99999 FORMAT (8X,1P,E9.1)
99998 FORMAT ((1X,7I11))
      END
```

9.2 Program Data

```
F04CJF Example Program Data

   4              2                                                :N and NRHS

 ( -1.84,  0.00) (  0.11, -0.11) ( -1.78, -1.18) (  3.91, -1.50)
                 ( -4.63 , 0.00) ( -1.84,  0.03) (  2.21,  0.21)
                                 ( -8.87,  0.00) (  1.58, -0.90)
                                                 ( -1.36 , 0.00) :End matrix A

 (  2.98,-10.18) ( 28.68,-39.89)
 ( -9.58,  3.88) (-24.79, -8.40)
 ( -0.77,-16.05) (  4.23,-70.02)
 (  7.79,  5.48) (-35.39, 18.01)                                 :End matrix B
```

9.3 Program Results

```
 F04CJF Example Program Results

 Solution
                          1                         2
 1  (      2.0000,     1.0000) (     -8.0000,     6.0000)
 2  (      3.0000,    -2.0000) (      7.0000,    -2.0000)
 3  (     -1.0000,     2.0000) (     -1.0000,     5.0000)
 4  (      1.0000,    -1.0000) (      3.0000,    -4.0000)

 Estimate of condition number
          6.7E+00

 Estimate of error bound for computed solutions
          7.4E-16
```