NAG Library Routine Document

F04CCF

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

F04CCF computes the solution to a complex system of linear equations $AX = B$, where A is an n by n tridiagonal matrix 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 F04CCF (N, NRHS, DL, D, DU, DU2, IPIV, B, LDB, RCOND, ERRBND,        &
                   IFAIL)

INTEGER               N, NRHS, IPIV(N), LDB, IFAIL
REAL (KIND=nag_wp)    RCOND, ERRBND
COMPLEX (KIND=nag_wp) DL(*), D(*), DU(*), DU2(N-2), B(LDB,*)
```

3 Description

The LU decomposition with partial pivoting and row interchanges is used to factor A as $A = PLU$, where P is a permutation matrix, L is unit lower triangular with at most one nonzero subdiagonal element, and U is an upper triangular band matrix with two superdiagonals. The factored form of A is then used to solve the system of equations $AX = B$.

Note that the equations $A^{\mathrm{T}}X = B$ may be solved by interchanging the order of the arguments DU and DL.

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 http://www.netlib.org/lapack/lug

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

5 Parameters

1: N – INTEGER *Input*

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

Constraint: $\mathrm{N} \geq 0$.

2: NRHS – INTEGER *Input*

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

Constraint: $\mathrm{NRHS} \geq 0$.

3: DL($*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

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

On entry: must contain the $(n-1)$ subdiagonal elements of the matrix A.

On exit: if $\mathrm{IFAIL} \geq 0$, DL is overwritten by the $(n-1)$ multipliers that define the matrix L from the LU factorization of A.

4: D($*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

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

On entry: must contain the n diagonal elements of the matrix A.

On exit: if IFAIL ≥ 0, D is overwritten by the n diagonal elements of the upper triangular matrix U from the LU factorization of A.

5: DU($*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

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

On entry: must contain the $(n-1)$ superdiagonal elements of the matrix A

On exit: if IFAIL ≥ 0, DU is overwritten by the $(n-1)$ elements of the first superdiagonal of U.

6: DU2$(\mathrm{N} - 2)$ – COMPLEX (KIND=nag_wp) array *Output*

On exit: if IFAIL ≥ 0, DU2 returns the $(n-2)$ elements of the second superdiagonal of U.

7: IPIV(N) – INTEGER array *Output*

On exit: if IFAIL ≥ 0, the pivot indices that define the permutation matrix P; at the ith step row i of the matrix was interchanged with row IPIV(i). IPIV(i) will always be either i or $(i+1)$; IPIV$(i) = i$ indicates a row interchange was not required.

8: B(LDB,$*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

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

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.

9: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F04CCF is called.

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

10: RCOND – REAL (KIND=nag_wp) *Output*

On exit: if no constraints are violated, an estimate of the reciprocal of the condition number of the matrix A, computed as $\mathrm{RCOND} = 1/\left(\|A\|_1 \|A^{-1}\|_1\right)$.

11: ERRBND – REAL (KIND=nag_wp) *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 $\|\hat{x} - x\|_1 / \|x\|_1 \leq \mathrm{ERRBND}$, 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.

12: 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 < 0 and IFAIL $\neq -999$

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

IFAIL $= -999$

Allocation of memory failed. The complex allocatable memory required is $2 \times$ N. In this case the factorization and the solution X have been computed, but RCOND and ERRBND have not been computed.

IFAIL > 0 and IFAIL $\leq$ N

If IFAIL $= i$, u_{ii} is exactly zero. The factorization has been completed, but the factor U 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 $\kappa(A) = \|A^{-1}\|_1\|A\|_1$, the condition number of A with respect to the solution of the linear equations. F04CCF 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 total number of floating point operations required to solve the equations $AX = B$ is proportional to nr. 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.

The real analogue of F04CCF is F04BCF.

9 Example

This example solves the equations

$$AX = B,$$

where A is the tridiagonal matrix

$$A = \begin{pmatrix} -1.3+1.3i & 2.0-1.0i & 0 & 0 & 0 \\ 1.0-2.0i & -1.3+1.3i & 2.0+1.0i & 0 & 0 \\ 0 & 1.0+1.0i & -1.3+3.3i & -1.0+1.0i & 0 \\ 0 & 0 & 2.0-3.0i & -0.3+4.3i & 1.0-1.0i \\ 0 & 0 & 0 & 1.0+1.0i & -3.3+1.3i \end{pmatrix}$$

and

$$B = \begin{pmatrix} 2.4-\ 5.0i & 2.7+\ 6.9i \\ 3.4+18.2i & -6.9-\ 5.3i \\ -14.7+\ 9.7i & -6.0-\ 0.6i \\ 31.9-\ 7.7i & -3.9+\ 9.3i \\ -1.0+\ 1.6i & -3.0+12.2i \end{pmatrix}.$$

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

9.1 Program Text

```
    Program f04ccfe

!     F04CCF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: f04ccf, nag_wp, x04dbf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: errbnd, rcond
      Integer                          :: i, ierr, ifail, ldb, n, nrhs
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: b(:,:), d(:), dl(:), du(:), du2(:)
      Integer, Allocatable             :: ipiv(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Executable Statements ..
      Write (nout,*) 'F04CCF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, nrhs
      ldb = n
      Allocate (b(ldb,nrhs),d(n),dl(n-1),du(n-1),du2(n-2),ipiv(n))
!     Read A and B from data file

      Read (nin,*) du(1:n-1)
      Read (nin,*) d(1:n)
      Read (nin,*) dl(1:n-1)
      Read (nin,*)(b(i,1:nrhs),i=1,n)

!     Solve the equations AX = B for X

!     ifail: behaviour on error exit
!            =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 1
      Call f04ccf(n,nrhs,dl,d,du,du2,ipiv,b,ldb,rcond,errbnd,ifail)

      If (ifail==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) 1.0E0_nag_wp/rcond
        Write (nout,*)
        Write (nout,*) 'Estimate of error bound for computed solutions'
        Write (nout,99999) errbnd
      Else If (ifail==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,*)
        Flush (nout)

        ierr = 0
        Call x04dbf('General',' ',n,nrhs,b,ldb,'Bracketed',' ','Solution', &
          'Integer',rlabs,'Integer',clabs,80,0,ierr)

      Else If (ifail>0 .And. ifail<=n) Then
!       The upper triangular matrix U is exactly singular.  Print
!       details of factorization
        Write (nout,*) 'Details of factorization'
        Write (nout,*)
        Write (nout,*) ' Second super-diagonal of U'
        Write (nout,99998) du2(1:n-2)
        Write (nout,*)
        Write (nout,*) ' First super-diagonal of U'
        Write (nout,99998) du(1:n-1)
        Write (nout,*)
        Write (nout,*) ' Main diagonal of U'
        Write (nout,99998) d(1:n)
        Write (nout,*)
        Write (nout,*) ' Multipliers'
        Write (nout,99998) dl(1:n-1)
        Write (nout,*)
        Write (nout,*) ' Vector of interchanges'
        Write (nout,99997) ipiv(1:n)
      Else
        Write (nout,99996) ifail
      End If

99999 Format (8X,1P,E9.1)
99998 Format (4(1X,'(',F7.4,',',F7.4,')':))
99997 Format (1X,8I9)
99996 Format (1X,' ** F04CCF returned with IFAIL = ',I5)
    End Program f04ccfe
```

9.2 Program Data

```
F04CCF Example Program Data

    5            2                                     : n, nrhs

 (  2.0, -1.0) (  2.0,  1.0) ( -1.0,  1.0) (  1.0, -1.0) : du

 ( -1.3,  1.3) ( -1.3,  1.3) ( -1.3,  3.3) ( -0.3,  4.3)
 ( -3.3,  1.3)                                         : d

 (  1.0, -2.0) (  1.0 , 1.0) (  2.0, -3.0) (  1.0,  1.0) : dl

 (  2.4, -5.0) (  2.7,  6.9)
 (  3.4, 18.2) ( -6.9, -5.3)
 (-14.7,  9.7) ( -6.0, -0.6)
 ( 31.9, -7.7) ( -3.9,  9.3)
 ( -1.0,  1.6) ( -3.0, 12.2)                           : b
```

9.3 Program Results

```
F04CCF Example Program Results

Solution
                          1                         2
1  (     1.0000,     1.0000) (     2.0000,    -1.0000)
2  (     3.0000,    -1.0000) (     1.0000,     2.0000)
3  (     4.0000,     5.0000) (    -1.0000,     1.0000)
4  (    -1.0000,    -2.0000) (     2.0000,     1.0000)
5  (     1.0000,    -1.0000) (     2.0000,    -2.0000)

Estimate of condition number
         1.8E+02

Estimate of error bound for computed solutions
         2.0E-14
```