F04ECFP

NAG Parallel Library Routine Document

Note: Before using this routine, please read the Users' Note for your implementation to check for implementation-dependent details. You are advised to enclose any calls to NAG Parallel Library routines between calls to Z01AAFP and Z01ABFP.

1 Description

F04ECFP calculates the solution of a set of complex linear equations

$$AX = B,\ A^TX = B \text{ or } A^HX = B,$$

with multiple right-hand sides, using an LU factorization, where A and B are n by n and n by r matrices respectively.

The routine first computes an LU factorization of A as $A = PLU$, where P is a permutation matrix, L is lower triangular with unit diagonal elements and U is upper triangular. The routine uses partial pivoting, with row interchanges. An approximation to X is found by forward and backward substitution.

2 Specification

```
 SUBROUTINE F04ECFP(ICNTXT, TRANS, N, NB, A, LDA, NRHS, B, LDB,
1                   IPIV, IFAIL)
 COMPLEX*16         A(LDA,*), B(LDB,*)
 INTEGER            ICNTXT, N, NB, LDA, NRHS, LDB, IPIV(*), IFAIL
 CHARACTER*1        TRANS
```

3 Data Distribution

3.1 Definitions

The following definitions are used in describing the data distribution within this document:

m_p	–	the number of rows in the logical processor grid.
n_p	–	the number of columns in the logical processor grid.
p_r	–	the row grid coordinate of the calling processor.
p_c	–	the column grid coordinate of the calling processor.
N_b	–	the blocking factor for the distribution of the rows and columns of the matrix.
numroc$(\alpha, b_\ell, q, s, k)$	–	a function which gives the **num**ber of **r**ows **o**r **c**olumns of a distributed matrix owned by the processor with the row or column coordinate q (p_r or p_c), where α is the total number of rows or columns of the matrix, b_ℓ is the blocking factor used (N_b), s is the row or column coordinate of the processor that possesses the first row or column of the distributed matrix, and k is either n_p or m_p. The Library provides the function Z01CAFP (NUMROC) for the evaluation of this function.

3.2 Global and Local Arguments

The input arguments TRANS, N, NB, NRHS and IFAIL are all global and so must have the same values on entry to the routine on each processor. The output argument IFAIL is global, and will have the same value on exit from the routine on each processor. The remaining arguments are local.

3.3 Distribution Strategy

The matrix A must be partitioned into N_b by N_b square blocks and stored in an array A in a cyclic 2-d block distribution. In this routine, the logical processor {0,0} of the processor grid must always possess the first block of the distributed matrix (i.e., $s = 0$ in the function numroc). This data distribution is

described in more detail in the F04 Chapter Introduction. The right-hand sides of the equation, B, must be stored in the array B, also in a cyclic 2-d block distribution.

This routine assumes that the data has already been correctly distributed, and if this is not the case will fail to produce correct results. However, the Library provides some utility routines which assist you in distributing data correctly. Descriptions of these routines can be found in Chapters F01 and X04 of the NAG Parallel Library Manual.

4 Arguments

1: ICNTXT — INTEGER *Local Input*

On entry: the BLACS context used by the communication mechanism, usually returned by a call to Z01AAFP.

2: TRANS — CHARACTER*1 *Global Input*

On entry: indicates the form of the equations as follows:

if TRANS = 'N', then $AX = B$ is solved for X;
if TRANS = 'T', then $A^TX = B$ is solved for X.
if TRANS = 'C', then $A^HX = B$ is solved for X.

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

3: N — INTEGER *Global Input*

On entry: the order of the matrix A, n.

Constraint: N ≥ 0.

4: NB — INTEGER *Global Input*

On entry: the blocking factor, N_b, used to distribute the rows and columns of the matrices A and B.

Constraints: NB ≥ 1.

5: A(LDA,$*$) — COMPLEX*16 array *Local Input/Local Output*

Note: the second dimension of the array A must be at least max(1,numroc(N,NB,p_c,0,n_p)).

On entry: the local part of the matrix A.

On exit: A is overwritten by the factors L and U distributed in the same cyclic 2-d block fashion; the unit diagonal elements of L are not stored.

6: LDA — INTEGER *Local Input*

On entry: the size of the first dimension of the array A as declared in the (sub)program from which F04ECFP is called.

Constraint: LDA $\geq$ max(1,numroc(N,NB,p_r,0,m_p)).

7: NRHS — INTEGER *Global Input*

On entry: the number of right-hand sides, r.

Constraint: NRHS ≥ 0.

8: B(LDB,$*$) — COMPLEX*16 array *Local Input/Local Output*

Note: the second dimension of the array B must be at least max(1,numroc(NRHS,NB,p_c,0,n_p)).

On entry: the local part of the the n by r right-hand side matrix B.

On exit: the n by r solution matrix X distributed in the same cyclic 2-d block distribution.

9: LDB — INTEGER *Local Input*

On entry: the size of the first dimension of the array B as declared in the (sub)program from which F04ECFP is called.

Constraint: LDB $\geq$ max(1,numroc(N,NB,p_r,0,m_p)).

10: IPIV($*$) — INTEGER array *Local Output*

Note: the dimension of the array IPIV must be at least NB + numroc(N,NB,p_r,0,m_p).

On exit: contains the pivot indices. The global row IPIV(k) of the matrix A was interchanged with the local row k. This array is aligned with the distributed matrix A.

11: IFAIL — INTEGER *Global Input/Global Output*

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this parameter (described in the Essential Introduction) the recommended values are:

IFAIL = 0, if multigridding is **not** employed;
IFAIL = -1, if multigridding is employed.

On exit: IFAIL = 0 unless the routine detects an error (see Section 5).

5 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 = -2000

The routine has been called with an invalid value of ICNTXT on one or more processors.

IFAIL = -1000

The logical processor grid and library mechanism (Library Grid) have not been correctly defined, see Z01AAFP.

IFAIL = $-i$

On entry, the ith argument had an invalid value. This error occurred either because a global argument did not have the same value on all the logical processors (see Section 3.2), or because its value was incorrect. An explanatory message distinguishes between these two cases.

IFAIL = 1

A diagonal element u_{ii} of U is exactly zero. The factorization has been completed but the factor U is exactly singular, and division by zero will occur if it is used to solve a system of linear equations.

6 Further Comments

The total number of floating-point operations is approximately $\frac{8}{3}n^3 + 8n^2r$.

6.1 Algorithmic Detail

The routine uses a block-partitioned LU factorization with partial pivoting. See [2] for further details. To compute X, forward and backward substitution are used.

6.2 Parallelism Detail

Each processor column performs an LU factorization with partial pivoting on successive column blocks of the matrix. Details of this factorization and pivoting are passed to all processors which perform the update of the remaining matrix in parallel. Forward and backward substitution are applied to each column block of the right-hand sides in parallel.

6.3 Accuracy

For each right-hand side vector b, the computed solution x is the exact solution of a perturbed system of equations $(A + E)x = b$, where

$$\| E \| \leq \epsilon g c(n) \| A \|,$$

$c(n)$ is a modest function of n, ϵ is the ***machine precision*** and g is $\max |u_{ij}|$.

If $\hat{x}$ is the true solution, then the computed solution x satisfies the bound

$$\frac{\|x - \hat{x}\|}{\|x\|} \leq g c(n) \mathrm{cond}(A) \epsilon$$

where $\mathrm{cond}(A) = \|A\| \cdot \|A^{-1}\|$.

7 References

[1] Golub G H and Van Loan C F (1989) *Matrix Computations* Johns Hopkins University Press (2nd Edition), Baltimore

[2] Anderson E, Bai Z, Bischof C, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A, Ostrouchov S and Sorensen D (1995) *LAPACK Users' Guide* (2nd Edition) SIAM, Philadelphia

8 Example

To solve the system of equations $AX = B$, where

$$A = \begin{pmatrix} -1.34 + 2.55i & 0.28 + 3.17i & -6.39 - 2.20i & 0.72 - 0.92i \\ -0.17 - 1.41i & 3.31 - 0.15i & -0.15 + 1.34i & 1.29 + 1.38i \\ -3.29 - 2.39i & -1.91 + 4.42i & -0.14 - 1.35i & 1.72 + 1.35i \\ 2.41 + 0.39i & -0.56 + 1.47i & -0.83 - 0.69i & -1.96 + 0.67i \end{pmatrix} \text{ and}$$

$$B = \begin{pmatrix} 26.26 + 51.78i & 31.32 - 6.70i \\ 6.43 - 8.68i & 15.86 - 1.42i \\ -5.75 + 25.31i & -2.15 + 30.19i \\ 1.16 + 2.57i & -2.56 + 7.55i \end{pmatrix}.$$

The example uses a 2 by 2 logical processor grid and a block size of 2 for both A and B.

Note: the listing of the Example Program presented below does not give a full pathname for the data file being opened, but in general the user must give the full pathname in this and any other OPEN statement.

8.1 Example Text

```
*     F04ECFP Example Program Text
*     NAG Parallel Library Release 2. NAG Copyright 1996.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NB
      PARAMETER        (NB=2)
      INTEGER          NMAX, LDA, LDB, NRHMAX, LW
      PARAMETER        (NMAX=8,LDA=NMAX,LDB=NMAX,NRHMAX=2,LW=NMAX)
*     .. Local Scalars ..
      INTEGER          ICNTXT, IFAIL, MP, N, NP, NRHS
      LOGICAL          ROOT
      CHARACTER        TRANS
      CHARACTER*80     FORMAT
*     .. Local Arrays ..
      COMPLEX*16       A(LDA,NMAX), B(LDB,NRHMAX), WORK(LW)
```

```
      INTEGER          IPIV(NMAX+NB)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         F04ECFP, X04BVFP, X04BWFP, Z01AAFP, Z01ABFP
*     .. Executable Statements ..
      ROOT = Z01ACFP()
      IF (ROOT) WRITE (NOUT,*) 'F04ECFP Example Program Results'
*
      MP = 2
      NP = 2
      IFAIL = 0
*
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      OPEN (NIN,FILE='f04ecfpe.d')
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, NRHS
      READ (NIN,*) FORMAT
*
      IF (N.LE.NMAX .AND. NRHS.LE.NRHMAX) THEN
*
         IFAIL = 0
*
*        Read in matrices A and B
*
         CALL X04BVFP(ICNTXT,NIN,N,N,NB,A,LDA,IFAIL)
*
         CALL X04BVFP(ICNTXT,NIN,N,NRHS,NB,B,LDB,IFAIL)
*
         TRANS = 'N'
*
         CALL F04ECFP(ICNTXT,TRANS,N,NB,A,LDA,NRHS,B,LDB,IPIV,IFAIL)
*
*        Print solution(s)
*
         IF (ROOT) THEN
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Solution(s)'
            WRITE (NOUT,*)
         END IF
*
         CALL X04BWFP(ICNTXT,NOUT,N,NRHS,NB,B,LDB,FORMAT,WORK,IFAIL)
*
      END IF
*
      CLOSE (NIN)
*
      IFAIL = 0
      CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
      STOP
      END
```

8.2 Example Data

```
F07ECFP Example Program Data
 4  2                                                    :Values of N and NRHS
 'F7.4'                                                  :Value of Format
 (-1.34, 2.55) ( 0.28, 3.17) (-6.39,-2.20) ( 0.72,-0.92)
 (-0.17,-1.41) ( 3.31,-0.15) (-0.15, 1.34) ( 1.29, 1.38)
 (-3.29,-2.39) (-1.91, 4.42) (-0.14,-1.35) ( 1.72, 1.35)
 ( 2.41, 0.39) (-0.56, 1.47) (-0.83,-0.69) (-1.96, 0.67)  :End of matrix A
 (26.26, 51.78) (31.32, -6.70)
 ( 6.43, -8.68) (15.86, -1.42)
 (-5.75, 25.31) (-2.15, 30.19)
 ( 1.16,  2.57) (-2.56,  7.55)                           :End of matrix B
```

8.3 Example Results

```
 F04ECFP Example Program Results

 Solution(s)

 ( 1.0000, 1.0000) (-1.0000,-2.0000)
 ( 2.0000,-3.0000) ( 5.0000, 1.0000)
 (-4.0000,-5.0000) (-3.0000, 4.0000)
 ( 0.0000, 6.0000) ( 2.0000,-3.0000)
```