F07FSFP (PZPOTRS)

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

F07FSFP (PZPOTRS) solves an n by n complex Hermitian positive-definite system of linear equations with multiple right hand-sides, i.e., $A_sX = B_s$, where A_s is a submatrix of a larger m_A by n_A matrix A, i.e.,

$$A_s(1:n,1:n) \equiv A(i_A:i_A+n-1, j_A:j_A+n-1),$$

and B_s is a (r right-hand sides) submatrix of a larger m_B by n_B matrix B, i.e.,

$$B_s(1:n,1:r) \equiv B(i_B:i_B+n-1, j_B:j_B+r-1).$$

The matrix A_s must have been previously factorized by a call to F07FRFP (PZPOTRF). F07FRFP (PZPOTRF) performs a Cholesky factorization and F07FSFP (PZPOTRS) solves the system of equations by forward and backward substitution.

2 Specification

```
      SUBROUTINE F07FSFP(UPLO, N, NRHS, A, IA, JA, IDESCA, B, IB, JB,
     1                   IDESCB, INFO)
      ENTRY      PZPOTRS(UPLO, N, NRHS, A, IA, JA, IDESCA, B, IB, JB,
     1                   IDESCB, INFO)
      COMPLEX*16         A(*), B(*)
      INTEGER            N, NRHS, IA, JA, IDESCA(9), IB, JB, IDESCB(9),
     1                   INFO
      CHARACTER*1        UPLO
```

The ENTRY statement enables the routine to be called by its ScaLAPACK name.

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.
M_b^X	–	the blocking factor for the distribution of the rows of a matrix X.
N_b^X	–	the blocking factor for the distribution of the columns of a matrix X.
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 (M_b^X or N_b^X), 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 UPLO, N, NRHS, IA, JA, IB, JB and the array elements IDESCA(1), IDESCA(3),...,IDESCA(8), IDESCB(1) and IDESCB(3),...,IDESCB(8) are all global and so must have the same value on entry to the routine on each processor. The output argument INFO is global and so will have the same value on exit from the routine on each processor. The remaining arguments are local.

3.3 Distribution Strategy

The array A must contain the Cholesky factorization of the matrix A_s, previously factorized by F07FRFP (PZPOTRF). The Cholesky factors must be stored in a cyclic 2-d block distribution (described in the F07 Chapter Introduction), as returned by F07FRFP (PZPOTRF). The right-hand sides of the equation, B_s are stored in the array B, 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: UPLO — CHARACTER*1 *Global Input*

On entry: indicates whether A_s has been factorized as $U^H U$ or LL^H as follows:

if UPLO = 'U', then $A_s = U^H U$, where U is upper triangular;
if UPLO = 'L', then $A_s = LL^H$, where L is lower triangular.

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

2: N — INTEGER *Global Input*

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

Constraint: $0 \le$ N $\le$ min(IDESCA(3) , IDESCA(4), IDESCB(3)).

3: NRHS — INTEGER *Global Input*

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

Constraint: $0 \le$ NRHS $\le$ IDESCB(4).

4: A($*$) — COMPLEX*16 array *Local Input*

Note: the array A is formally defined as a vector. However, you may find it more convenient to consider A as a 2-d array of dimension (IDESCA(9),γ), where $\gamma \ge$ numroc(JA + N $-$ 1, IDESCA(6), p_c, IDESCA(8), n_p). See the Example Program.

On entry: the local part of the Cholesky factorization of the matrix A_s as returned by F07FRFP (PZPOTRF).

5: IA — INTEGER *Global Input*

On entry: the row index of matrix A, i_A, that identifies the first row of the Cholesky factorization of A_s.

Constraints: $1 \le$ IA $\le$ IDESCA(3) $-$ N + 1 and mod(IA $-$ 1, IDESCA(5)) = 0.

6: JA — INTEGER *Global Input*

On entry: the column index of matrix A, j_A, that identifies the first column of the Cholesky factorization of A_s.

Constraints: $1 \le$ JA $\le$ IDESCA(4) $-$ N + 1 and mod(JA $-$ 1, IDESCA(6)) = 0.

7: IDESCA(9) — INTEGER array *Local Input*

Distribution: the array elements IDESCA(1) and IDESCA(3),...,IDESCA(8) must be global to the processor grid and the array elements IDESCA(2) and IDESCA(9) are local to each processor.

On entry: the description array for the matrix A. This array must contain details of the distribution of the matrix A and the logical processor grid.

IDESCA(1), the descriptor tyep. For this routine, which uses a cyclic 2-d block distribution, IDESCA(1) = 1;
IDESCA(2), the BLACS context (ICNTXT) for the processor grid, usually returned by Z01AAFP;
IDESCA(3), the number of rows, m_A, of the matrix A;
IDESCA(4), the number of columns, n_A, of the matrix A;
IDESCA(5), the blocking factor, M_b^A, used to distribute the rows of the matrix A;
IDESCA(6), the blocking factor, N_b^A, used to distribute the columns of the matrix A;
IDESCA(7), the processor row index over which the first row of the matrix A is distributed;
IDESCA(8), the processor column index over which the first column of the matrix A is distributed;
IDESCA(9), the leading dimension of the conceptual 2-d array A.

Constraints:

IDESCA(1) = 1;
IDESCA(3) $\geq$ 0; IDESCA(4) $\geq$ 0;
IDESCA(5) = IDESCA(6); IDESCA(5) $\geq$ 1; IDESCA(6) $\geq$ 1;
$0 \leq$ IDESCA(7) $\leq m_p - 1$; $0 \leq$ IDESCA(8) $\leq n_p - 1$;
IDESCA(9) $\geq$ max(1,numroc(IDESCA(3),IDESCA(5),p_r,IDESCA(7),m_p)).

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

Note: array B is formally defined as a vector. However, you may find it more convenient to consider B as a 2-d array of dimension (IDESCB(9),γ), where
$\gamma \geq$ numroc(JB + NRHS $-$ 1, IDESCB(6), p_c, IDESCB(8), n_p). See the Example Program.

On entry: the local part of the right-hand side matrix B which may contain parts of the n by r submatrix B_s.

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

9: IB — INTEGER *Global Input*

On entry: the row index of matrix B, i_B, that identifies the first column of the submatrix B_s.

Constraints: $1 \leq$ IB $\leq$ IDESCB(3) $-$ N + 1 and mod(IB $-$ 1, IDESCB(5)) = 0.
The IAth row of the array A and the IBth row of the array B must be located on the same row of the processor grid, i.e.,

$$\text{mod(IDESCA(7) + (IA} - 1\text{)/IDESCA(5)}, \ n_p) = \text{mod(IDESCB(7) + (IB} - 1\text{)/IDESCB(5)}, \ n_p).$$

10: JB — INTEGER *Global Input*

On entry: the column index of matrix B, j_B, that identifies the first column of the submatrix B_s.

Constraint: $1 \leq$ JB $\leq$ IDESCB(4) $-$ NRHS + 1.

11: IDESCB(9) — INTEGER array *Local Input*

Distribution: the array elements IDESCB(1) and IDESCB(3),...,IDESCB(8) must be global to the processor grid and the array elements IDESCB(2) and IDESCB(9) are local to each processor.

On entry: the description array for the matrix B. This array must contain details of the distribution of the matrix B and the logical processor grid.

IDESCB(1), the descriptor type. For this routine, which uses a cyclic 2-d block distribution, IDESCB(1) = 1;
IDESCB(2), the BLACS context (ICNTXT) for the processor grid, usually returned by Z01AAFP;
IDESCB(3), the number of rows, m_B, of the matrix B;
IDESCB(4), the number of columns, n_B, of the matrix B;
IDESCB(5), the blocking factor, M_b^B, used to distribute the rows of the matrix B;
IDESCB(6), the blocking factor, N_b^B, used to distribute the columns of the matrix B;
IDESCB(7), the processor row index over which the first row of the matrix B is distributed;
IDESCB(8), the processor column index over which the first column of the matrix B is distributed;
IDESCB(9), the leading dimension of the conceptual 2-d array B.

Constraints:

IDESCB(1) = 1;
IDESCB(3) $\geq$ 0; IDESCB(4) $\geq$ 0;
IDESCB(2) = IDESCA(2);
IDESCB(5) = IDESCB(6); IDESCB(5) $\geq$ 1; IDESCB(6) $\geq$ 1;
$0 \leq$ IDESCB(7) $\leq m_p - 1$; $0 \leq$ IDESCB(8) $\leq n_p - 1$;
IDESCB(9) $\geq$ max(1,numroc(IDESCB(3),IDESCB(5),p_r,IDESCB(7),m_p)).

12: INFO — INTEGER *Global Output*

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

5 Errors and Warnings

If INFO $\neq$ 0 an explanatory message is output and control returned to the calling program.

INFO < 0

On entry, one of the arguments was invalid:

if the kth argument is a scalar INFO = $-k$;
if the kth argument is an array and its jth element is invalid, INFO = $-(100 \times k + j)$.

This error occurred either because a global argument did not have the same value on all logical processors, or because its value on one or more processors was incorrect.

6 Further Comments

The total number of floating-point operations is approximately $8n^2r$.

6.1 Algorithmic Detail

Forward and backward substitution is used.

If UPLO = 'U', $A = U^HU$, where U is upper triangular; the solution X is computed by solving $U^HY = B$ and then $UX = Y$.

If UPLO = 'L', $A = LL^H$, where L is lower triangular; the solution X is computed by solving $LY = B$ and then $L^HX = Y$.

6.2 Parallelism Detail

The Level 3 BLAS operations are carried out 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

$$|E| \leq c(n)\epsilon|U^H| \cdot |U| \text{ if UPLO = 'U'},$$

$$|E| \leq c(n)\epsilon|L| \cdot |L^H| \text{ if UPLO = 'L'},$$

$c(n)$ is a modest linear function of n and ϵ is the ***machine precision***. If x is the true solution, then the computed solution $\hat{x}$ satisfies a forward error bound of the form

$$\frac{\|x-\hat{x}\|_\infty}{\|x\|_\infty} \leq \epsilon c(n)\kappa(A),$$

where $\kappa(A)$ is the condition number of A. See the F07 Chapter Introduction.

7 References

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

8 Example

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

$$A = \begin{pmatrix} 3.23+0.00i & 1.51-1.92i & 1.90+0.84i & 0.42+2.50i \\ 1.51+1.92i & 3.58+0.00i & -0.23+1.11i & -1.18+1.37i \\ 1.90-0.84i & -0.23-1.11i & 4.09+0.00i & 2.33-0.14i \\ 0.42-2.50i & -1.18-1.37i & 2.33+0.14i & 4.29+0.00i \end{pmatrix} \text{ and}$$

$$B = \begin{pmatrix} 3.93-6.14i & 1.48+6.58i \\ 6.17+9.42i & 4.65-4.75i \\ -7.17-21.83i & -4.91+2.29i \\ 1.99-14.38i & 7.64-10.79i \end{pmatrix}.$$

Here A is Hermitian positive-definite and must first be factorized by F07FRFP. The example uses a 2 by 2 logical processor grid and a block size of 2.

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

```
*     F07FSFP Example Program Text
*     NAG Parallel Library Release 2. NAG Copyright 1996.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          DT
      PARAMETER        (DT=1)
      INTEGER          MB, NB
      PARAMETER        (MB=2,NB=MB)
      INTEGER          NMAX, IAROW, IACOL, LDA, LDB, NRHMAX, LW
      PARAMETER        (NMAX=8,IAROW=0,IACOL=0,LDA=NMAX,LDB=NMAX,
     +                 NRHMAX=2,LW=NMAX)
```

```
*     .. Local Scalars ..
      INTEGER          IA, IB, ICNTXT, IFAIL, INFO, JA, JB, MP, N, NP,
     +                 NRHS
      LOGICAL          ROOT
      CHARACTER        UPLO
      CHARACTER*80     FORMAT
*     .. Local Arrays ..
      COMPLEX*16       A(LDA,NMAX), B(LDB,NRHMAX), WORK(LW)
      INTEGER          IDESCA(9), IDESCB(9)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         F07FRFP, F07FSFP, X04BRFP, X04BSFP, Z01AAFP,
     +                 Z01ABFP
*     .. Executable Statements ..
      ROOT = Z01ACFP()
      IF (ROOT) THEN
         WRITE (NOUT,*) 'F07FSFP Example Program Results'
         WRITE (NOUT,*)
      END IF
*
      MP = 2
      NP = 2
      IFAIL = 0
*
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      OPEN (NIN,FILE='f07fsfpe.d')
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, NRHS, UPLO, FORMAT
*
      IF (N.LE.NMAX .AND. NRHS.LE.NRHMAX) THEN
*
*        Set the array descriptor of A
*
         IDESCA(1) = DT
         IDESCA(2) = ICNTXT
         IDESCA(3) = N
         IDESCA(4) = N
         IDESCA(5) = MB
         IDESCA(6) = NB
         IDESCA(7) = IAROW
         IDESCA(8) = IACOL
         IDESCA(9) = LDA
         IA = 1
         JA = 1
*
*        Read A from the data file
*
         IFAIL = 0
         CALL X04BRFP(NIN,N,N,A,1,1,IDESCA,IFAIL)
*
*        Factorize the matrix
*
         CALL F07FRFP(UPLO,N,A,IA,JA,IDESCA,INFO)
*
```

```
         IF (INFO.EQ.0) THEN
*
*           Set the array descriptor of B
*
            IDESCB(1) = DT
            IDESCB(2) = IDESCA(2)
            IDESCB(3) = N
            IDESCB(4) = NRHS
            IDESCB(5) = MB
            IDESCB(6) = NB
            IDESCB(7) = IAROW
            IDESCB(8) = IACOL
            IDESCB(9) = LDB
            IB = 1
            JB = 1
*
*           Read B from data file
*
            IFAIL = 0
            CALL X04BRFP(NIN,N,NRHS,B,1,1,IDESCB,IFAIL)
*
            CALL F07FSFP(UPLO,N,NRHS,A,IA,JA,IDESCA,B,IB,JB,IDESCB,INFO)
            IF (INFO.EQ.0) THEN
*
*               Print solution(s)
*
               IF (ROOT) THEN
                  WRITE (NOUT,*) 'Solution(s)'
                  WRITE (NOUT,*)
               END IF
               IFAIL = 0
*
               CALL X04BSFP(NOUT,N,NRHS,B,IB,JB,IDESCB,FORMAT,WORK,
     +                      IFAIL)
*
            ELSE
               IF (ROOT) WRITE (NOUT,*)
     +             'Unable to solve triangular system'
            END IF
         ELSE
            IF (ROOT) WRITE (NOUT,*)
     +          'Matrix is not positive-definite'
         END IF
*
      END IF
*
      CLOSE (NIN)
*
      IFAIL = 0
      CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
      STOP
      END
```

8.2 Example Data

```
F07FSFP Example Program Data
 4 2 'L' 'F7.4'                        :Value of N, NRHS, UPLO and FORMAT
 (3.23, 0.00) ( 0.0 , 0.0 ) ( 0.0 , 0.0 ) ( 0.0 , 0.0 )
 (1.51, 1.92) ( 3.58, 0.00) ( 0.0 , 0.0 ) ( 0.0 , 0.0 )
 (1.90,-0.84) (-0.23,-1.11) ( 4.09, 0.00) ( 0.0 , 0.0 )
 (0.42,-2.50) (-1.18,-1.37) ( 2.33, 0.14) ( 4.29, 0.00)  :End of matrix A
 ( 3.93, -6.14) ( 1.48,  6.58)
 ( 6.17,  9.42) ( 4.65, -4.75)
 (-7.17,-21.83) (-4.91,  2.29)
 ( 1.99,-14.38) ( 7.64,-10.79)                           :End of matrix B
```

8.3 Example Results

```
 F07FSFP Example Program Results

 Solution(s)

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