F08FEFP (PDSYTRD)

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

F08FEFP (PDSYTRD) reduces an n by n real symmetric matrix A_s to tridiagonal form T by an orthogonal similarity transformation Q

$$Q^T A_s Q = T,$$

where A_s is a submatrix of an m_A by n_A matrix A, i.e.,

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

Note: if $i_A = j_A = 1$ and $n = m_A = n_A$, then $A_s \equiv A$.

Since the matrix A_s is real symmetric, only the the upper triangular part or the lower triangular part is required.

The diagonal elements of the tridiagonal matrix T are represented by a vector d of length n and the off-diagonal elements by a vector e. On exit, the vector d is distributed in the one-dimensional block cyclic form across each logical processor row of the two-dimensional logical processor grid. The vector e is similarly distributed.

The orthogonal matrix Q is not formed explicitly but is represented as a product of $n - 1$ elementary reflectors. See the F08 Chapter Introduction for details of the distributions of Q, d and e.

The routine is designed to be used as the first step in computing the eigenvalues of A_s. It should be followed by calls to F01ZPFP to gather the vectors d and e onto each logical processor. Eigenvalues of T, which are the same as the eigenvalues of A_s, can then be computed by calling F08JFFP (PDSTEBZ).

2 Specification

```
 SUBROUTINE F08FEFP(UPLO, N, A, IA, JA, IDESCA, D, E, TAU, WORK,
1                   LWORK, INFO)
 ENTRY      PDSYTRD(UPLO, N, A, IA, JA, IDESCA, D, E, TAU, WORK,
1                   LWORK, INFO)
 DOUBLE PRECISION   A(*), D(*), E(*), TAU(*), WORK(LWORK)
 INTEGER            N, IA, JA, IDESCA(9), LWORK, 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.
s_r^X	–	the row coordinate of the processor that possesses the first row of a distributed matrix X.
s_c^X	–	the column coordinate of the processor that possesses the first column of a distributed matrix X.
$\mathrm{numroc}(\hat{\ell}, L_b^X, p, s^X, \ell_p)$	–	a function which gives the **num**ber of **r**ows **o**r **c**olumns of a distributed matrix X owned by the processor with the row or column coordinate p (p_r or p_c), where $\hat{\ell}$ is the total number of rows or columns of the matrix, L_b^X is the blocking factor used (M_b^X or N_b^X), s^X is the row or column coordinate (s_r^X or s_c^X) of the processor that possesses the first row or column of the distributed matrix and ℓ_p is either m_p or n_p. The Library provides the utility function Z01CAFP (NUMROC) for the evaluation of this function.
$\mathrm{indxg2p}(k, L_b^X, p, s^X, \ell_p)$	–	a function which gives the processor row or column coordinate which possess the row or column index k of a distributed matrix. The arguments L_b^X, s^X and ℓ_p have the same meaning as in the function numroc. However, the argument p is a dummy integer. The Library provides the utility function Z01CDFP (INDXG2P) for the evaluation of this function.

3.2 Global and Local Arguments

The input arguments UPLO, N, IA, JA and the array elements IDESCA(1) and IDESCA(3),...,IDESCA(8) are all global and so must have the same values 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

On entry, the matrix A must be stored in the cyclic 2-d block format and its descriptor array IDESCA must contain the details of the distributed matrix. The indices i_A and j_A and the order n identify the submatrix A_s. See the F08 Chapter Introduction for further details. The matrices A and A_s must satisfy the following requirements:

$$M_b^A = N_b^A;$$
$$\mathrm{mod}(i_A - 1,\ M_b^A) = 0;$$
$$\mathrm{mod}(j_A - 1,\ N_b^A) = 0.$$

Any further constraints, including the above, are stated in Section 4 under each argument.

This routine is the first step in the solution of the eigenvalue problem of a dense matrix A. However, before proceeding to eigenroutines, it is often necessary to gather complete copies of the vectors d and e to every logical processor on the grid. The Library provides the routine F01ZPFP for this particular gathering operation. The following prerequisites are mandatory on F08FEFP for post-processing by F01ZPFP:

$$i_A = j_A = 1;$$
$$s_r^A = s_c^A = 0.$$

It is assumed that the data has already been correctly distributed, and if this is not the case, then this routine will fail to produce correct results. However, the Library provides some utility routines to assist 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 the upper or lower triangular part of A_s is stored, as follows:

if UPLO = 'U', then the upper triangular part of A_s is stored;
if UPLO = 'L', then the lower triangular part of A_s is stored.

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

2: N — INTEGER *Global Input*

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

Constraints: $0 \leq$ N $\leq$ min(IDESCA(3),IDESCA(4)).

3: A($*$) — DOUBLE PRECISION array *Local Input/Local Output*

Note: 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
$\gamma \geq$ numroc(JA $-$ 1 + N, IDESCA(6), p_c, IDESCA(8), n_p).

On entry: the local part of the matrix A which may contain parts of the n by n submatrix A_s to be factorized.

If UPLO = 'U', the upper triangle of A_s must be stored and the elements of the matrix below the diagonal are not referenced;
if UPLO = 'L', the lower triangle of A_s must be stored and the elements of the matrix above the diagonal are not referenced.

On exit: The locally held parts of A corresponding to the matrix A_s are overwritten by the elements of the tridiagonal matrix T and the details of the orthogonal matrix Q.

4: IA — INTEGER *Global Input*

On entry: the row index of matrix A, i_A, that identifies the first row of the submatrix A_s to be factorized.

Note: i_A must be equal to 1 if this routine is to be followed by a call to F01ZPFP.

Constraints:

mod(IA $-$ 1, IDESCA(5)) = 0;
1 $\leq$ IA $\leq$ IDESCA(3) $-$ N + 1.

5: JA — INTEGER *Global Input*

On entry: the column index of matrix A, j_A, that identifies the first column of the submatrix A_s to be factorized.

Note: j_A must be equal to 1 if this routine is to be followed by a call to F01ZPFP.

Constraints:

mod(JA $-$ 1, IDESCA(6)) = 0;
1 $\leq$ JA $\leq$ IDESCA(4) $-$ N + 1.

6: 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 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 type. 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, s_r^A, over which the first row of the matrix A is distributed;

IDESCA(8), the processor column index, s_c^A, 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) ≥ 0; IDESCA(4) ≥ 0;

IDESCA(5) = IDESCA(6) ≥ 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)).

7: D($*$) — DOUBLE PRECISION array *Local Output*

Note: the dimension of the array D must be at least
numroc(JA − 1 + N, IDESCA(6), p_c, IDESCA(8), n_p).

On exit: the local parts of the distributed vector d which contains the diagonal elements of the tridiagonal matrix T. The vector d is distributed in the one-dimensional block cyclic form across each logical processor row of the two-dimensional logical processor grid. See the F08 Chapter Introduction for further details.

8: E($*$) — DOUBLE PRECISION array *Local Output*

Note: the dimension of the array E must be at least
numroc(JA − 1 + N, IDESCA(6), p_c, IDESCA(8), n_p).

On exit: the local parts of the distributed vector e, which contains the off-diagonal elements of the tridiagonal matrix T: if UPLO = 'U', $e_1 = 0$ and $e_i = t_{i-1,i}$ for $i = 2, \ldots, n$; if UPLO = 'L', $e_i = t_{i+1,i}$ for $i = 1, \ldots, n-1$ and $e_n = 0$. The vector e is distributed in the same manner as d.

9: TAU($*$) — DOUBLE PRECISION array *Local Output*

Note: the dimension of the array TAU must be at least
numroc(JA − 1 + N, IDESCA(6), p_c, IDESCA(8), n_p).

On exit: the scalar factors τ of the elementory reflectors which define the orthogonal matrix Q. See the F08 Chapter Introduction for further details.

10: WORK(LWORK) — DOUBLE PRECISION array *Local Workspace/Local Output*

On exit: WORK(1) contains the expected minimum value for LWORK.

11: LWORK — INTEGER *Local Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08FEFP (PDSYTRD) is called.

Constraint: LWORK $\geq$ IDESCA(5) $\times \max(\alpha+1, 3)$, where

α = numroc(N, IDESCA(5), p_r, β, m_p);
β = indxg2p(IA, IDESCA(5), p_r, IDESCA(7), m_p).

This value of LWORK can be calculated by using the Library function Z01CEFP; i.e.,

LWORK = Z01CEFP(N, IA, IDESCA)

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

6.1 Algorithmic Detail

The total number of floating-point operations is approximately $\frac{4}{3}n^3$.

6.2 Parallelism Detail

The BLAS operations used in this routine are carried out in parallel.

6.3 Accuracy

The computed tridiagonal matrix T is exactly similar to a nearby matrix $A+E$, where

$$\|E\|_2 \leq p(n)\epsilon\|A\|_2,$$

$p(n)$ is a modestly increasing function of n, and ϵ is the ***machine precision***.

The elements of T themselves may be sensitive to small perturbations in A or to rounding errors in the computation, but this does not affect the stability of the eigenvalues and eigenvectors.

6.4 Pre-processing

Not applicable.

6.5 Post-processing

To gather the vectors d and e to every logical processor on the grid, the routine F01ZPFP can be used.

7 References

[1] 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

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

[3] Choi J, Dongarra J J and Walker D W (1995) The design of a parallel dense linear algebra software library: reduction to Hessenberg, tridiagonal, and bidiagonal form *Technical Report ORNL/TM-12472* Department of Computer Science, University of Tennessee, 107 Ayres Hall, Knoxville, TN 37996-1301, USA

8 Example

The example program illustrates the reduction of a 9 by 9 symmetric matrix A_s to the tridiagonal form T using F08FEFP (PDSYTRD). The (i,j)th element of this matrix A_s is $\max(i,j)$ and the matrix is generated using the routine F01ZQFP. The argument UPLO is set to 'U' and hence only the upper triangular part of the matrix A_s is used.

The diagonal vector d and the off-diagonal vector e are gathered to every logical processor using the routine F01ZPFP. Finally, the vector d (denoted locally by the array DL) and the the vector e (denoted locally by the array EL) are printed on the root processor.

This example program also gives the required minimum sizes of the workspace array WORK on each logical processor for the routines F08FEFP (PDSYTRD) and F01ZPFP. This information is printed on the root processor using the routine X04BFFP. A header identifies the logical processor concerned.

8.1 Example Text

```
*     F08FEFP Example Program Text
*     NAG Parallel Library Release 2. NAG Copyright 1996.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          N
      PARAMETER        (N=9)
      INTEGER          DT
      PARAMETER        (DT=1)
      INTEGER          NA
      PARAMETER        (NA=20)
      INTEGER          NB
      PARAMETER        (NB=2)
      INTEGER          MPG, NPG
      PARAMETER        (MPG=2,NPG=2)
      INTEGER          LDA, TDA
      PARAMETER        (LDA=NA/MPG+NB,TDA=NA/NPG+NB)
      INTEGER          LWORK
      PARAMETER        (LWORK=25)
      CHARACTER*20     FORMT
      PARAMETER        (FORMT='F12.0')
*     .. Local Scalars ..
      INTEGER          I, IA, ICNTXT, ICOFF, IFAIL, INFO, JA, MP, NP
      LOGICAL          ROOT
      CHARACTER        CNUMOP, TITOP, UPLO
*     .. Local Arrays ..
      DOUBLE PRECISION A(LDA,TDA), D(TDA), DL(N), E(TDA), EL(N),
     +                 TAU(TDA), WORK(LWORK)
      INTEGER          IDESCA(9)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         F01ZPFP, F01ZQFP, F08FEFP, GMATA, X04BFFP,
```

```
     +                   Z01AAFP, Z01ABFP
*     .. Executable Statements ..
*
      ROOT = Z01ACFP()
      IF (ROOT) THEN
         WRITE (NOUT,*) 'F08FEFP Example Program Results'
         WRITE (NOUT,*)
      END IF
*
      MP = 2
      NP = 2
      IFAIL = 0
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      IA = 1
      JA = 1
      IDESCA(1) = DT
      IDESCA(2) = ICNTXT
      IDESCA(3) = NA
      IDESCA(4) = NA
      IDESCA(5) = NB
      IDESCA(6) = NB
      IDESCA(7) = 0
      IDESCA(8) = 0
      IDESCA(9) = LDA
*
*        Distribute the matrix A
*
      IFAIL = 0
      CALL F01ZQFP(GMATA,N,N,A,IA,JA,IDESCA,IFAIL)
*
*        Reduce to the tridiagonal form
*
      UPLO = 'U'
      CALL F08FEFP(UPLO,N,A,IA,JA,IDESCA,D,E,TAU,WORK,LWORK,INFO)
*
      IF (ROOT) THEN
         WRITE (NOUT,*) 'Minimum value required for LWORK'
         WRITE (NOUT,*) 'by F08FEFP on each logical processor'
         TITOP = 'Y'
         CNUMOP = 'N'
      END IF
*
      IFAIL = 0
      CALL X04BFFP(ICNTXT,NOUT,1,1,WORK,1,FORMT,TITOP,CNUMOP,ICOFF,
     +              WORK(2),1,IFAIL)

*
*        Gather the diagonal D to each logical processor
*
      IFAIL = 0
      CALL F01ZPFP(N,IA,JA,IDESCA,D,DL,WORK,LWORK,IFAIL)
*
*        Gather the off-diagonal E to each logical processor
*
      IFAIL = 0
      CALL F01ZPFP(N,IA,JA,IDESCA,E,EL,WORK,LWORK,IFAIL)
*
```

```
      IF (ROOT) THEN
         WRITE (NOUT,*) 'Minimum value required for LWORK'
         WRITE (NOUT,*) 'by F01ZPFP on each logical processor'
         TITOP = 'Y'
         CNUMOP = 'N'
      END IF
*
      IFAIL = 0
      CALL X04BFFP(ICNTXT,NOUT,1,1,WORK,1,FORMT,TITOP,CNUMOP,ICOFF,
     +             WORK(2),1,IFAIL)
*
*     Print the diagonal and the off-diagonal
*     of the tridiagonal matrix on the root process
*
      IF (ROOT) THEN
         WRITE (NOUT,*) 'Elements of the tridigonal matrix T'
         WRITE (NOUT,*)
         IF (UPLO.EQ.'U') THEN
            WRITE (NOUT,'(3X,"I",12X,"DL(I)",9X,"EL(I)"/)')
            WRITE (NOUT,'(1X,I3,5X,2(F12.4,2X))') 1, DL(1)
            DO 20 I = 2, N
               WRITE (NOUT,'(1X,I3,5X,2(F12.4,2X))') I, DL(I), EL(I)
   20       CONTINUE
         ELSE IF (UPLO.EQ.'L') THEN
            WRITE (NOUT,'(3X,"I",12X,"DL(I)",9X,"EL(I)"/)')
            DO 40 I = 1, N - 1
               WRITE (NOUT,'(1X,I3,5X,2(F12.4,2X))') I, DL(I), EL(I)
   40       CONTINUE
            WRITE (NOUT,'(1X,I3,5X,2(F12.4,2X))') N, DL(N)
         END IF
      END IF
*
      IFAIL = 0
      CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
      STOP
      END
*
      SUBROUTINE GMATA(I1,I2,J1,J2,AL,LDAL)
*     GMATA generates the block A(I1: I2, J1: J2) of the matrix A such
*     that
*
*        a(i,j) = max(i,j)
*
*     in the array AL.
*
*     .. Scalar Arguments ..
      INTEGER          I1, I2, J1, J2, LDAL
*     .. Array Arguments ..
      DOUBLE PRECISION AL(LDAL,*)
*     .. Local Scalars ..
      INTEGER          I, J, K, L
*     .. Intrinsic Functions ..
      INTRINSIC        MAX
*     .. Executable Statements ..
      L = 1
      DO 40 J = J1, J2
         K = 1
```

```
         DO 20 I = I1, I2
            AL(K,L) = MAX(I,J)
            K = K + 1
   20    CONTINUE
         L = L + 1
   40 CONTINUE
      RETURN
      END
```

8.2 Example Data

None.

8.3 Example Results

```
F08FEFP Example Program Results

Minimum value required for LWORK
by F08FEFP on each logical processor
  Array from logical processor {   0,   0}

          12.

  Array from logical processor {   0,   1}

          12.

  Array from logical processor {   1,   0}

          10.

  Array from logical processor {   1,   1}

          10.

Minimum value required for LWORK
by F01ZPFP on each logical processor
  Array from logical processor {   0,   0}

           9.

  Array from logical processor {   0,   1}

           9.

  Array from logical processor {   1,   0}

           9.

  Array from logical processor {   1,   1}

           9.

Elements of the tridigonal matrix T

  I            DL(I)         EL(I)

  1          -0.2778
```

```
   2          -0.3309       -0.0248
   3          -0.4176       -0.0583
   4          -0.5737       -0.1151
   5          -0.9000       -0.2235
   6          -1.7857       -0.4737
   7          -6.2143       -1.2963
   8          46.5000        9.4472
   9           9.0000      -25.4558
```
