F02FQFP

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

F02FQFP computes the eigenvalues and eigenvectors (the spectral decomposition) of a real symmetric matrix whose columns are distributed on a two-dimensional Library Grid.

The spectral decomposition of an $n \times n$ real symmetric matrix A is defined as

$$A = Z \Lambda Z^T$$

where Z is an $n \times n$ orthogonal matrix of eigenvectors and Λ is an $n \times n$ diagonal matrix of eigenvalues of A.

2 Specification

```
SUBROUTINE F02FQFP(ICNTXT, N, A, LDA, NX, IFAIL)
DOUBLE PRECISION   A(0:LDA-1,0:*)
INTEGER            ICNTXT, N, LDA, NX, IFAIL
```

3 Usage

3.1 Definitions

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

m_p	–	the number of rows in the Library Grid.
n_p	–	the number of columns in the Library Grid.
p	–	$m_p \times n_p$, the total number of processors in the Library Grid.
p_d	–	the number of logical processors which hold columns of the matrix A.
N_b	–	the maximum number of columns of the matrix A held locally on a logical processor.
N_x	–	the actual number of columns of the matrix A held locally on a logical processor where $0 \leq N_x \leq N_b$.
$\lceil x \rceil$	–	the ceiling function of x, which gives the smallest integer which is not less than x.

3.2 Global and Local Arguments

The following global **input** arguments must have the same value on entry to the routine on each processor and the global **output** arguments will have the same value on exit from the routine on each processor:

Global input arguments: N, IFAIL

Global output arguments: IFAIL

The remaining arguments are local.

3.3 Distribution Strategy

Columns of the matrix A are allocated to logical processors in the Library Grid row by row (i.e., in row major ordering of the grid) starting from the {0,0} logical processor. Each logical processor that contains columns of the matrix contains $N_b = \lceil n/p \rceil$ columns, except the last processor that actually contains data, for which the number of columns held may be less than N_b. This processor will contain $\mathrm{mod}(n, N_b)$ columns if $\mathrm{mod}(n, N_b) \neq 0$, and will contain N_b columns otherwise. Some logical processors may not contain any columns of the matrix if n is not large relative to p, but if $n > (p-1)^2$ then all processors will certainly contain columns of the matrix.

The number of logical processors that contain columns of the matrix is given by $p_d = \lceil n/N_b \rceil$.

The following example illustrates a case where the last processor with data is not the last processor of the grid. Furthermore the number of columns on the last processor with data is not equal to the number of columns on other processors.

If $m_p = 2$, $n_p = 4$ then $p = m_p \times n_p = 8$. If $n = 41$ then $N_b = \lceil n/p \rceil = \lceil 5.125 \rceil = 6$, $\mathrm{mod}(n, N_b) = 5 \neq 0$ and $p_d = \lceil n/N_b \rceil = \lceil 6.833 \rceil = 7$.

processor {0,0} $N_x = 6$ columns (1:6)	processor {0,1} $N_x = 6$ columns (7:12)	processor {0,2} $N_x = 6$ columns (13:18)	processor {0,3} $N_x = 6$ columns (19:24)
processor {1,0} $N_x = 6$ columns (25:30)	processor {1,1} $N_x = 6$ columns (31:36)	processor {1,2} $N_x = 5$ columns (37:41)	processor {1,3} $N_x = 0$

If the data is distributed incorrectly, the routine may fail to produce correct results or will exit with an error flag.

3.4 Related Routines

The Library provides support routines for the generation, scattering/gathering and input/output of matrices whose columns are distributed on the Library Grid. The following routines may be used in conjunction with F02FQFP:

Real matrix generation:	F01ZRFP
Real matrix output:	X04BFFP

4 Arguments

1: ICNTXT — INTEGER *Local Input*

On entry: the Library context, usually returned by a call to the Library Grid initialisation routine Z01AAFP.

Note: the value of ICNTXT **must not** be changed.

2: N — INTEGER *Global Input*

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

Constraint: N ≥ 0.

3: A(0:LDA−1,0:∗) — DOUBLE PRECISION array *Local Input/Local Output*

Note: the size of the second dimension of the array A must be at least $N_x + 1$ where N_x is the number of columns of A held locally by the logical processor. The array A is not referenced if $N_x = 0$.

On entry: A(1:n,1:N_x) must contain columns of the matrix A as defined by the distribution strategy (see Section 3.3).

On exit: A(0,1:N_x) contains N_x eigenvalues of the matrix A stored on this logical processor. They are ordered locally and globally (in the row major ordering of the processors) in non-decreasing order of magnitude.

A(1:n,1:N_x) contains the eigenvectors corresponding to the eigenvalues kept on this logical processor.

The remainder of the array is used as workspace and contains no useful information.

4: LDA — INTEGER *Local Input*

On entry: the size of the first dimension of the array A as declared in the (sub)program from which F02FQFP is called.

Constraint: LDA $\geq 2\times$N + 2.

5: NX — INTEGER *Local Output*

On exit: N_x, the number of columns of the matrix A held on the logical processor. This is also equal to the number of eigenvalues held by the logical processor.

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

The NAG Parallel Library provides a mechanism, via the routine Z02EAFP, to reduce the amount of parameter validation performed by this routine. For a full description refer to the Z02 Chapter Introduction.

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this argument (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 (or -9999 if reduced error checking is enabled) 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 from the root processor (or processor {0,0} when the root processor is not available) on the current error message unit (as defined by X04AAF).

5.1 Full Error Checking Mode Only

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 was invalid. This error occured 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. An explanatory message distinguishes between these two cases.

5.2 Any Error Checking Mode

IFAIL = 1

The Jacobi algorithm has not converged.

6 Further Comments

6.1 Algorithmic Detail

The algorithm is based on an one-sided Jacobi method, see Hestenes [1].

6.2 Parallelism Detail

The algorithm uses a linear array of logical processors. This linear array is mapped to the Library Grid based on the row major ordering beginning from the {0,0} logical processor on the two-dimensional array. Most of the communication is between neighbours on the linear array of processors.

6.3 Accuracy

The computed factors Λ and Z satisfy the relation

$$Z\Lambda Z^T = A + E,$$

where

$$\|E\| \leq c\epsilon\|A\|,$$

ϵ being the ***machine precision***, c is a modest function of n and $\|.\|$ denotes the 2-norm.

7 References

[1] Hestenes M R (1958) Inversion of matrices by biorthogonalization and related results *J. SIAM* **6** 51–90

8 Example

To find the eigenvalues and eigenvectors of the 7 by 7 matrix A given by

$$A = \left(\begin{array}{cc|cc|cc|c} 1.0 & 2.0 & 3.0 & 4.0 & 5.0 & 6.0 & 7.0 \\ 2.0 & 2.0 & 3.0 & 4.0 & 5.0 & 6.0 & 7.0 \\ 3.0 & 3.0 & 3.0 & 4.0 & 5.0 & 6.0 & 7.0 \\ 4.0 & 4.0 & 4.0 & 4.0 & 5.0 & 6.0 & 7.0 \\ 5.0 & 5.0 & 5.0 & 5.0 & 5.0 & 6.0 & 7.0 \\ 6.0 & 6.0 & 6.0 & 6.0 & 6.0 & 6.0 & 7.0 \\ 7.0 & 7.0 & 7.0 & 7.0 & 7.0 & 7.0 & 7.0 \end{array}\right)$$

and to print the results on the root processor. Routine F01ZRFP is used to generate the matrix A on a 2 by 2 logical processor grid. The number of columns of the matrix A on each logical processor, N_x, is equal to 2 on logical processors {0,0}, {0,1}, and {1,0}. On the final logical processor {1,1}, $N_x = 1$. This blocking is indicated by the vertical lines in the matrix A above.

The routine X04BFFP is used to bring eigenvalues to the root processor and print them. The same routine is called again to print the eigenvectors.

8.1 Example Text

```
*     F02FQFP Example Program Text
*     NAG Parallel Library Release 2. NAG Copyright 1996.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          N
      PARAMETER        (N=7)
      INTEGER          MG, NG
      PARAMETER        (MG=2,NG=2)
      INTEGER          LDA, TDA
      PARAMETER        (LDA=N+N+2,TDA=(N/(MG*NG)+2))
      CHARACTER*20     FORMT
      PARAMETER        (FORMT='F12.4')
*     .. Local Scalars ..
      INTEGER          ICNTXT, ICOFF, IFAIL, MP, NP, NX
      LOGICAL          ROOT
      CHARACTER        CNUMOP, TITOP
*     .. Local Arrays ..
      DOUBLE PRECISION A(0:LDA-1,0:TDA-1)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
```

```
*     .. External Subroutines ..
      EXTERNAL         F01ZRFP, F02FQFP, GMATA, X04BFFP, Z01AAFP,
     +                 Z01ABFP
*     .. Executable Statements ..
      ROOT = Z01ACFP()
      IF (ROOT) THEN
         WRITE (NOUT,*) 'F02FQFP Example Program Results'
         WRITE (NOUT,*)
      END IF
*
*     Define the 2D processor grid
*
      MP = MG
      NP = NG
      IFAIL = 0
*
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      IFAIL = 0
*
*     Generate matrix A
*
      CALL F01ZRFP(ICNTXT,GMATA,N,N,A(1,1),LDA,NX,IFAIL)
*
      IFAIL = 0
*
*     Solution of the symmetric eigenvalue problem
*
      CALL F02FQFP(ICNTXT,N,A,LDA,NX,IFAIL)
*
*     Print the eigenvalues
*
      IF (ROOT) THEN
         WRITE (NOUT,*) 'Eigenvalues'
         WRITE (NOUT,*)
         TITOP = 'N'
         CNUMOP = 'G'
      END IF
      ICOFF = 0
      IFAIL = 0
*
      CALL X04BFFP(ICNTXT,NOUT,1,NX,A(0,1),LDA,FORMT,TITOP,CNUMOP,ICOFF,
     +             A(N+1,1),LDA,IFAIL)
*
*     Print the eigenvectors
*
      IF (ROOT) THEN
         WRITE (NOUT,*) 'Eigenvectors'
         WRITE (NOUT,*)
      END IF
      IFAIL = 0
*
      CALL X04BFFP(ICNTXT,NOUT,N,NX,A(1,1),LDA,FORMT,TITOP,CNUMOP,ICOFF,
     +             A(N+1,1),LDA,IFAIL)
*
      IFAIL = 0
*
*     Undefine the 2D grid
```

```
*
      CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
      STOP
      END
*
      SUBROUTINE GMATA(M,J1,J2,AL,LDAL)
*
*     GMATA generates the block A( 1: M, J1: J2 ) of the matrix A such
*     that
*       a(i,j) = max(i,j)
*     in the array AL.
*
*     .. Scalar Arguments ..
      INTEGER          J1, J2, LDAL, M
*     .. Array Arguments ..
      DOUBLE PRECISION AL(LDAL,*)
*     .. Local Scalars ..
      INTEGER          I, J, L
*     .. Intrinsic Functions ..
      INTRINSIC        MAX
*     .. Executable Statements ..
      L = 1
      DO 40 J = J1, J2
         DO 20 I = 1, M
            AL(I,L) = MAX(I,J)
   20    CONTINUE
         L = L + 1
   40 CONTINUE
*
*     End of GMATA.
*
      RETURN
      END
```

8.2 Example Data

None.

8.3 Example Results

```
F02FQFP Example Program Results

Eigenvalues

           1          2
     -6.2163    -1.3855

           3          4
     -0.6530    -0.4115

           5          6
     -0.3087    -0.2632

           7
     37.2381

Eigenvectors

           1          2
      0.5542    -0.4903
      0.4650    -0.1364
      0.3011     0.3159
      0.0887     0.5402
     -0.1380     0.3746
     -0.3424    -0.0613
     -0.4918    -0.4530

           3          4
     -0.4225     0.3362
      0.2246    -0.4808
      0.5277    -0.1294
      0.0227     0.5364
     -0.5171    -0.1013
     -0.2650    -0.4928
      0.3930     0.3133

           5          6
     -0.2337    -0.1198
      0.5235     0.3355
     -0.4153    -0.4841
     -0.0087     0.5359
      0.4260    -0.4805
     -0.5195     0.3290
      0.2180    -0.1118

           7
      0.3033
      0.3115
      0.3280
      0.3533
      0.3881
      0.4333
      0.4902
```