NAG Fortran Library Routine Document

M01ZCF

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

M01ZCF decomposes a permutation into cycles, as an aid to re-ordering ranked data.

2 Specification

```
SUBROUTINE M01ZCF(IPERM, M1, M2, ICYCL, IFAIL)
INTEGER           IPERM(M2), M1, M2, ICYCL(M2), IFAIL
```

3 Description

M01ZCF is provided as an aid to re-ordering arbitrary data structures without using additional storage. However users should consider carefully whether it is necessary to rearrange their data, or whether it would be simpler and more efficient to refer to the data in sorted order using an index vector, or to create a copy of the data in sorted order.

To rearrange data into a different order without using additional storage, the simplest method is to decompose the permutation which specifies the new order into cycles and then to do a cyclic permutation of the data items in each cycle. (This is the method used by the M01E re-ordering routines.) Given a vector IRANK which specifies the ranks of the data (as generated by the M01D routines), M01ZCF generates a new vector ICYCL, in which the permutation is represented in its component cycles, with the first element of each cycle negated. For example, the permutation

$$5 \quad 7 \quad 4 \quad 2 \quad 1 \quad 6 \quad 3$$

is composed of the cycles

$$(1 \quad 5) \quad (2 \quad 7 \quad 3 \quad 4) \quad (6)$$

and the vector ICYCL generated by M01ZCF contains

$$-1 \quad 5 \quad -2 \quad 7 \quad 3 \quad 4 \quad -6$$

In order to rearrange the data according to the specified ranks:

item 6 must be left in place;

items 1 and 5 must be interchanged;

items 4, 2, 7 and 3 must be moved right one place round the cycle.

The complete rearrangement can be achieved by the following code:

```
      DO 10 K = M1, M2
         I = ICYCL(K)
         IF (I.LT.0) THEN
            J = -I
         ELSE
            [swap items I and J]
         ENDIF
   10 CONTINUE
```

4 References

None.

5 Parameters

1: IPERM(M2) – INTEGER array *Input/Output*

On entry: elements M1 to M2 of IPERM must contain a permutation of the integers M1 to M2.

On exit: IPERM is used as internal workpsace prior to being restored and hence is unchanged.

2: M1 – INTEGER *Input*

3: M2 – INTEGER *Input*

On entry: M1 and M2 must specify the range of elements used in the array IPERM and the range of values in the permutation, as specified under IPERM.

Constraint: $0 < \mathrm{M1} \leq \mathrm{M2}$.

4: ICYCL(M2) – INTEGER array *Output*

On exit: elements M1 to M2 of ICYCL contain a representation of the permutation as a list of cycles, with the first integer in each cycle negated. (See Section 3.)

5: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL $= 0$ unless the routine detects an error (see Section 6).

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, for users 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.**

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 $= 1$

On entry, $\mathrm{M2} < 1$,
or $\mathrm{M1} < 1$,
or $\mathrm{M1} > \mathrm{M2}$.

IFAIL $= 2$

Elements M1 to M2 of IPERM contain a value outside the range M1 to M2.

IFAIL $= 3$

Elements M1 to M2 of IPERM contain a repeated value.

If IFAIL $= 2$ or 3, elements M1 to M2 of IPERM do not contain a permutation of the integers M1 to M2.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

The example program reads a matrix of real numbers and rearranges its columns so that the elements of the lth row are in ascending order. To do this, the program first calls M01DJF to rank the elements of the lth row, and then calls M01ZCF to decompose the rank vector into cycles. It then rearranges the columns using the framework of code suggested in Section 3. The value of l is read from the data file.

9.1 Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     M01ZCF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          MMAX, NMAX
      PARAMETER        (MMAX=20,NMAX=20)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             T
      INTEGER          I, IFAIL, II, J, K, L, M, N
*     .. Local Arrays ..
      real             RM(MMAX,NMAX)
      INTEGER          ICYCL(NMAX), IRANK(NMAX)
*     .. External Subroutines ..
      EXTERNAL         M01DJF, M01ZCF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'M01ZCF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) M, N, L
      IF (M.GE.1 .AND. M.LE.MMAX .AND. N.GE.1 .AND. N.LE.NMAX .AND.
     +    L.GE.1 .AND. L.LE.M) THEN
         DO 20 I = 1, M
            READ (NIN,*) (RM(I,J),J=1,N)
   20    CONTINUE
         IFAIL = 0
*
         CALL M01DJF(RM,MMAX,L,L,1,N,'Ascending',IRANK,IFAIL)
         CALL M01ZCF(IRANK,1,N,ICYCL,IFAIL)
*
         DO 60 K = 1, N
            I = ICYCL(K)
            IF (I.LT.0) THEN
               J = -I
            ELSE
*              Swap columns I and J
               DO 40 II = 1, M
                  T = RM(II,J)
                  RM(II,J) = RM(II,I)
                  RM(II,I) = T
   40          CONTINUE
            END IF
   60    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'Matrix sorted on row', L
         WRITE (NOUT,*)
         DO 80 I = 1, M
            WRITE (NOUT,99998) (RM(I,J),J=1,N)
   80    CONTINUE
      END IF
      STOP
*
99999 FORMAT (1X,A,I3)
99998 FORMAT (1X,12F6.1)
      END
```

9.2 Program Data

```
M01ZCF Example Program Data
3 12 3
5.0 4.0 3.0 2.0 2.0 1.0 9.0 4.0 4.0 2.0 2.0 1.0
3.0 8.0 2.0 5.0 5.0 6.0 9.0 8.0 9.0 5.0 4.0 1.0
9.0 1.0 6.0 1.0 2.0 4.0 8.0 1.0 2.0 2.0 6.0 2.0
```

9.3 Program Results

```
 M01ZCF Example Program Results

 Matrix sorted on row  3

    4.0   2.0   4.0   2.0   4.0   2.0   1.0   1.0   3.0   2.0   9.0   5.0
    8.0   5.0   8.0   5.0   9.0   5.0   1.0   6.0   2.0   4.0   9.0   3.0
    1.0   1.0   1.0   2.0   2.0   2.0   2.0   4.0   6.0   6.0   8.0   9.0
```