NAG Library Routine Document

E02ZAF

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

E02ZAF sorts two-dimensional data into rectangular panels.

2 Specification

```
SUBROUTINE E02ZAF (PX, PY, LAMDA, MU, M, X, Y, POINT, NPOINT, ADRES,       &
                   NADRES, IFAIL)
INTEGER            PX, PY, M, POINT(NPOINT), NPOINT, ADRES(NADRES),        &
                   NADRES, IFAIL
REAL (KIND=nag_wp) LAMDA(PX), MU(PY), X(M), Y(M)
```

3 Description

A set of m data points with rectangular Cartesian coordinates x_r, y_r are sorted into panels defined by lines parallel to the y and x axes. The intercepts of these lines on the x and y axes are given in LAMDA(i), for $i = 5, 6, \ldots, \mathrm{PX} - 4$ and MU(j), for $j = 5, 6, \ldots, \mathrm{PY} - 4$, respectively. The subroutine orders the data so that all points in a panel occur before data in succeeding panels, where the panels are numbered from bottom to top and then left to right, with the usual arrangement of axes, as shown in the diagram. Within a panel the points maintain their original order.

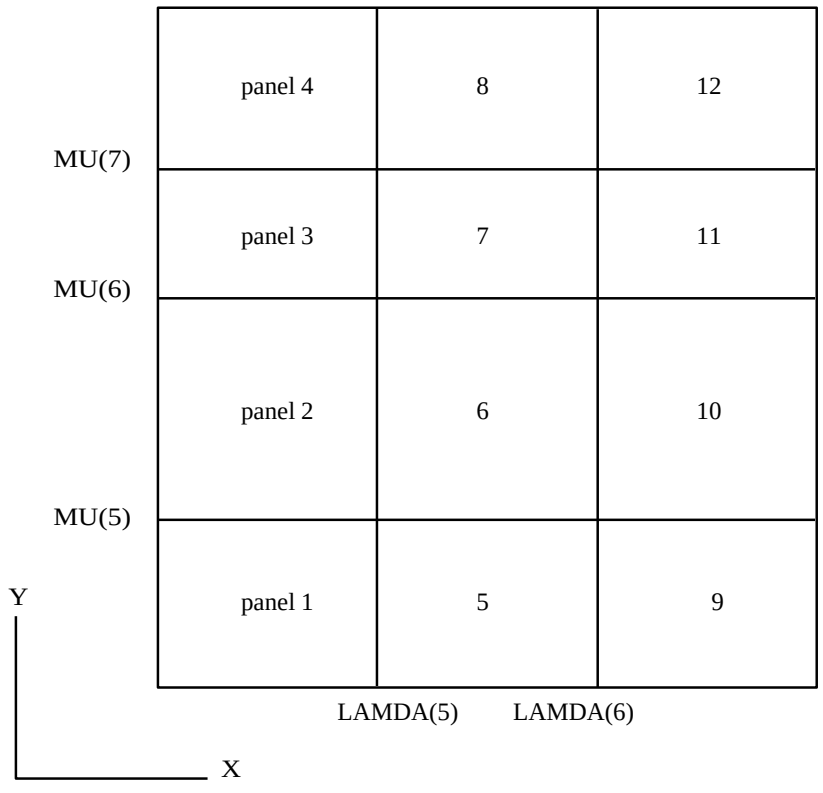

Figure 1

A data point lying exactly on one or more panel sides is taken to be in the highest-numbered panel adjacent to the point. The subroutine does not physically rearrange the data, but provides the array POINT which contains a linked list for each panel, pointing to the data in that panel. The total number of panels is $(\mathrm{PX}-7)\times(\mathrm{PY}-7)$.

4 References

None.

5 Arguments

1: PX – INTEGER *Input*

2: PY – INTEGER *Input*

On entry: PX and PY must specify eight more than the number of intercepts on the x axis and y axis, respectively.

Constraint: $\mathrm{PX}\geq 8$ and $\mathrm{PY}\geq 8$.

3: LAMDA(PX) – REAL (KIND=nag_wp) array *Input*

On entry: LAMDA(5) to LAMDA(PX – 4) must contain, in nondecreasing order, the intercepts on the x axis of the sides of the panels parallel to the y axis.

4: MU(PY) – REAL (KIND=nag_wp) array *Input*

On entry: MU(5) to MU(PY – 4) must contain, in nondecreasing order, the intercepts on the y axis of the sides of the panels parallel to the x axis.

5: M – INTEGER *Input*

On entry: the number m of data points.

6: X(M) – REAL (KIND=nag_wp) array *Input*

7: Y(M) – REAL (KIND=nag_wp) array *Input*

On entry: the coordinates of the rth data point (x_r, y_r), for $r = 1, 2, \ldots, m$.

8: POINT(NPOINT) – INTEGER array *Output*

On exit: for $i = 1, 2, \ldots,$ NPOINT, $\mathrm{POINT}(m+i) = \mathrm{I1}$ is the index of the first point in panel i, $\mathrm{POINT}(\mathrm{I1}) = \mathrm{I2}$ is the index of the second point in panel i and so on.

$\mathrm{POINT}(\mathrm{In}) = 0$ indicates that $\mathrm{X}(\mathrm{In}), \mathrm{Y}(\mathrm{In})$ was the last point in the panel.

The coordinates of points in panel i can be accessed in turn by means of the following instructions:

```
IN = M + I
 10 IN = POINT(IN)
 IF (IN.EQ. 0) GOTO 20
 XI = X(IN)
 YI = Y(IN)
 .
 .
 .
 GOTO 10
 20 ...
```

9: NPOINT – INTEGER *Input*

On entry: the dimension of the array POINT as declared in the (sub)program from which E02ZAF is called.

Constraint: $\mathrm{NPOINT}\geq \mathrm{M}+(\mathrm{PX}-7)\times(\mathrm{PY}-7)$.

10: ADRES(NADRES) – INTEGER array *Workspace*

11: NADRES – INTEGER *Input*

On entry: the value $(\mathrm{PX}-7)\times(\mathrm{PY}-7)$, the number of panels into which the (x,y) plane is divided.

12: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

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, if you are not familiar with this argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: $\mathrm{IFAIL}=0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry $\mathrm{IFAIL}=0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

$\mathrm{IFAIL}=1$

The intercepts in the array LAMDA, or in the array MU, are not in nondecreasing order.

$\mathrm{IFAIL}=2$

On entry, $\mathrm{PX}<8$,
or $\mathrm{PY}<8$,
or $\mathrm{M}\leq 0$,
or $\mathrm{NADRES}\neq(\mathrm{PX}-7)\times(\mathrm{PY}-7)$,
or $\mathrm{NPOINT}<\mathrm{M}+(\mathrm{PX}-7)\times(\mathrm{PY}-7)$.

$\mathrm{IFAIL}=-99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

$\mathrm{IFAIL}=-399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

$\mathrm{IFAIL}=-999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

E02ZAF is not threaded in any implementation.

9 Further Comments

The time taken is approximately proportional to $m \times \log(\text{NPOINT})$.

This subroutine was written to sort two-dimensional data in the manner required by routine E02DAF. The first 9 arguments of E02ZAF are the same as the arguments in E02DAF which have the same name.

10 Example

This example reads in data points and the intercepts of the panel sides on the x and y axes; it calls E02ZAF to set up the index array POINT; and finally it prints the data points in panel order.

10.1 Program Text

```
    Program e02zafe

!     E02ZAF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: e02zaf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, iadres, ifail, m, nadres, npoint, &
                                          px, py
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: lamda(:), mu(:), x(:), y(:)
      Integer, Allocatable             :: adres(:), point(:)
!     .. Executable Statements ..
      Write (nout,*) 'E02ZAF Example Program Results'

!     Skip heading in data file
      Read (nin,*)

      Read (nin,*) m
      Read (nin,*) px, py
      nadres = (px-7)*(py-7)
      npoint = m + nadres
      Allocate (adres(nadres),lamda(px),mu(py),x(m),y(m),point(npoint))

!     Read data points and intercepts of panel sides

      Read (nin,*)(x(i),y(i),i=1,m)

      If (px>8) Then
        Read (nin,*) lamda(5:(px-4))
      End If

      If (py>8) Then
        Read (nin,*) mu(5:(py-4))
      End If

!     Sort points into panel order

      ifail = 0
      Call e02zaf(px,py,lamda,mu,m,x,y,point,npoint,adres,nadres,ifail)

!     Output points in panel order

      Do i = 1, nadres
        Write (nout,*)
        Write (nout,99999) 'Panel', i
        iadres = m + i
```

```
loop:   Do
          iadres = point(iadres)

          If (iadres<=0) Then
            Exit loop
          End If

          Write (nout,99998) x(iadres), y(iadres)
        End Do loop

      End Do

99999 Format (1X,A,I4)
99998 Format (1X,2F7.2)
    End Program e02zafe
```

10.2 Program Data

```
E02ZAF Example Program Data
  10
   9
  10
0       0.77
0.70    1.06
1.44    0.33
0.21    0.44
1.01    0.50
1.84    0.02
0.71    1.95
1.00    1.20
0.54    0.04
1.53    0.18
1.00
0.80
1.20
0
```

10.3 Program Results

```
 E02ZAF Example Program Results

 Panel   1
    0.00   0.77
    0.21   0.44
    0.54   0.04

 Panel   2
    0.70   1.06

 Panel   3
    0.71   1.95

 Panel   4
    1.44   0.33
    1.01   0.50
    1.84   0.02
    1.53   0.18

 Panel   5

 Panel   6
    1.00   1.20
```