E04CCF – NAG Fortran Library Routine Document

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

E04CCF minimizes a general function $F(x)$ of n independent variables $x = (x_1, x_2, \ldots, x_n)^T$ by the Simplex method. Derivatives of the function need not be supplied.

2 Specification

```
 SUBROUTINE E04CCF(N, X, F, TOL, IW, W1, W2, W3, W4, W5, W6, FUNCT,
1                  MONIT, MAXCAL, IFAIL)
 INTEGER           N, IW, MAXCAL, IFAIL
 real              X(N), F, TOL, W1(N), W2(N), W3(N), W4(N),
1                  W5(IW), W6(IW,N)
 EXTERNAL          FUNCT, MONIT
```

3 Description

The routine finds an approximation to a minimum of a function of n variables. The user must supply a routine to calculate the value of the function for any set of values of the variables.

The method is iterative. A simplex of $n + 1$ points is set up in the dimensional space of the variables (for example in 2 dimensions the simplex is a triangle) under the assumption that the problem has been scaled so that the values of the independent variables at the minimum are of order unity. The starting point provided by the user is the first vertex of the simplex, the remaining n vertices are generated by the routine (see Parkinson and Hutchinson [2]). The vertex of the simplex with the largest function value is reflected in the centre of gravity of the remaining vertices and the function value at this new point is compared with the remaining function values. Depending on the outcome of this test the new point is accepted or rejected, a further expansion move may be made, or a contraction may be carried out [1], [2]. When no further progress can be made the sides of the simplex are reduced in length and the method is repeated.

The method tends to be slow, but it is robust and therefore very useful for functions that are subject to inaccuracies.

4 References

[1] Nelder J A and Mead R (1965) A simplex method for function minimization *Comput. J.* **7** 308–313

[2] Parkinson J M and Hutchinson D (1972) An investigation into the efficiency of variants of the simplex method *Numerical Methods for Nonlinear Optimization* (ed F A Lootsma) Academic Press

5 Parameters

1: N — INTEGER *Input*

On entry: the number n of independent variables.

Constraint: $N > 0$.

2: X(N) — ***real*** array *Input/Output*

On entry: a guess at the position of the minimum. Note that the problem should be scaled so that the values of the X(i) are of order unity.

On exit: the value of x corresponding to the function value in F.

3: F — ***real*** *Output*

On exit: the lowest function value found.

4: TOL — ***real*** *Input*

On entry: the error tolerable in the result, as follows:

If f_i, for $i = 1, 2, \ldots, n+1$, are the individual function values at the vertices of a simplex and f_m is the mean of these values, then the routine will terminate when

$$\sqrt{\frac{1}{n+1}\sum_{i=1}^{n+1}(f_i - f_m)^2} < \mathrm{TOL}.$$

Constraint: TOL must be greater than or equal to the ***machine precision***.

5: IW — INTEGER *Input*

On entry: the value N + 1.

Constraint: IW = N + 1.

6: W1(N) — ***real*** array *Workspace*
7: W2(N) — ***real*** array *Workspace*
8: W3(N) — ***real*** array *Workspace*
9: W4(N) — ***real*** array *Workspace*
10: W5(IW) — ***real*** array *Workspace*
11: W6(IW,N) — ***real*** array *Workspace*

12: FUNCT — SUBROUTINE, supplied by the user. *External Procedure*

FUNCT must calculate the value of the function at XC and assign this value to FC. It should be tested separately before being used in conjunction with E04CCF (see the Chapter Introduction).

Its specification is:

```
SUBROUTINE FUNCT(N, XC, FC)
INTEGER          N
real             XC(N), FC
```

1: N — INTEGER *Input*

On entry: the number n of variables.

2: XC(N) — ***real*** array *Input*

On entry: the point x at which the function is required.

3: FC — ***real*** *Output*

On exit: the value of the function $F(x)$ at the current point x.

FUNCT must be declared as EXTERNAL in the (sub)program from which E04CCF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

13: MONIT — SUBROUTINE, supplied by the user. *External Procedure*

MONIT is called once every iteration in E04CCF. It can be used to print out the current values of any selection of its parameters but must not be used to change the values of the parameters.

Its specification is:

```
SUBROUTINE MONIT(FMIN, FMAX, SIM, N, IS, NCALL)
INTEGER          N, IS, NCALL
real             FMIN, FMAX, SIM(IS,N)
```

1: FMIN — ***real*** *Input*

On entry: the smallest function value in the current simplex.

2: FMAX — ***real*** *Input*

On entry: the largest function value in the current simplex.

3: SIM(IS,N) — ***real*** array *Input*

On entry: the rows of SIM contain the position vectors of the vertices of the current simplex.

4: N — INTEGER *Input*

On entry: the number of variables.

5: IS — INTEGER *Input*

On entry: the first dimension of the array SIM.

6: NCALL — INTEGER *Input*

On entry: the number of times that FUNCT has been called so far.

MONIT must be declared as EXTERNAL in the (sub)program from which E04CCF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

14: MAXCAL — INTEGER *Input*

On entry: the maximum number of function evaluations to be allowed.

Constraint: MAXCAL ≥ 1.

15: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

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

6 Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, N < 1,
or TOL < ***machine precision***,
or IW $\neq$ N + 1,
or MAXCAL < 1.

IFAIL = 2

MAXCAL function evaluations have been completed, E04CCF has been terminated prematurely. Check the coding of the routine FUNCT before increasing the value of MAXCAL.

7 Accuracy

On a successful exit the accuracy will be as defined by TOL.

8 Further Comments

The time taken by the routine depends on the number of variables, the behaviour of the function and the distance of the starting point from the minimum. Each iteration consists of 1 or 2 function evaluations unless the size of the simplex is reduced, in which case $n+1$ function evaluations are required.

9 Example

A simple program to locate a minimum of the function:

$$F = e^{x_1}(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1).$$

The input parameters to E04CCF are all set before the routine is called.

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.

```
*     E04CCF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          N, IW
      PARAMETER        (N=2,IW=N+1)
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Scalars in Common ..
      INTEGER          IMONIT
*     .. Local Scalars ..
      real             F, TOL
      INTEGER          I, IFAIL, MAXCAL
*     .. Local Arrays ..
      real             SIM(IW,N), W1(N), W2(N), W3(N), W4(N), W5(IW),
     +                 X(N)
*     .. External Functions ..
      real             X02AJF
      EXTERNAL         X02AJF
*     .. External Subroutines ..
      EXTERNAL         E04CCF, FUNCT, MONIT
*     .. Intrinsic Functions ..
      INTRINSIC        SQRT
*     .. Common blocks ..
      COMMON           /OUTP/IMONIT
*     .. Executable Statements ..
      WRITE (NOUT,*) 'E04CCF Example Program Results'
*     ** Set IMONIT to 1 to obtain monitoring information **
      IMONIT = 0
      X(1) = -1.0e0
      X(2) = 1.0e0
      TOL = SQRT(X02AJF())
      MAXCAL = 100
      IFAIL = 0
*
      CALL E04CCF(N,X,F,TOL,IW,W1,W2,W3,W4,W5,SIM,FUNCT,MONIT,MAXCAL,
     +            IFAIL)
*
      WRITE (NOUT,*)
      WRITE (NOUT,99999) 'Final function value is ', F
      WRITE (NOUT,99999) 'at the point', (X(I),I=1,N)
      WRITE (NOUT,99998) 'This has error number', IFAIL
      STOP
*
99999 FORMAT (1X,A,2F12.4)
99998 FORMAT (1X,A,I3)
      END
*
      SUBROUTINE FUNCT(N,XC,FC)
*     .. Scalar Arguments ..
      real              FC
      INTEGER           N
*     .. Array Arguments ..
      real              XC(N)
*     .. Intrinsic Functions ..
      INTRINSIC         EXP
*     .. Executable Statements ..
      FC = EXP(XC(1))*(4.0e0*XC(1)*(XC(1)+XC(2))+2.0e0*XC(2)*(XC(2)
     +     +1.0e0)+1.0e0)
      RETURN
      END
*
      SUBROUTINE MONIT(FMIN,FMAX,SIM,N,N1,NCALL)
*     .. Parameters ..
      INTEGER           NOUT
      PARAMETER         (NOUT=6)
*     .. Scalar Arguments ..
      real              FMAX, FMIN
      INTEGER           N, N1, NCALL
*     .. Array Arguments ..
      real              SIM(N1,N)
*     .. Scalars in Common ..
      INTEGER           IMONIT
*     .. Local Scalars ..
      INTEGER           I, J
*     .. Common blocks ..
      COMMON            /OUTP/IMONIT
*     .. Executable Statements ..
      IF (IMONIT.NE.0) THEN
         WRITE (NOUT,99999) 'After', NCALL,
     +     '  function calls, the value is', FMIN, ' with simplex'
         WRITE (NOUT,99998) ((SIM(I,J),J=1,N),I=1,N1)
      END IF
      RETURN
*
99999 FORMAT (1X,A,I5,A,F10.4,A)
99998 FORMAT (1X,2F12.4)
      END
```

9.2 Program Data

None.

9.3 Program Results

```
E04CCF Example Program Results

Final function value is       0.0000
at the point      0.5000     -0.9999
This has error number  0
```