NAG Library Routine Document

E04CBF

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

E04CBF minimizes a general function $F(x)$ of $n$ independent variables $x = (x_1, x_2, \ldots, x_n)^T$ by the Nelder and Mead simplex method (see Nelder and Mead (1965)). Derivatives of the function need not be supplied.

2 Specification

SUBROUTINE E04CBF (N, X, F, TOLF, TOLX, FUNCT, MONIT, MAXCAL, IUSER, & RUSER, IFAIL)

INTEGER N, MAXCAL, IUSER(*), IFAIL
REAL (KIND=nag_wp) X(N), F, TOLF, TOLX, RUSER(*)
EXTERNAL FUNCT, MONIT

3 Description

E04CBF finds an approximation to a minimum of a function $F$ of $n$ variables. You must supply a subroutine to calculate the value of $F$ for any set of values of the variables.

The method is iterative. A simplex of $n + 1$ points is set up in the $n$-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 you have provided is the first vertex of the simplex, the remaining $n$ vertices are generated by E04CBF. 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. See Nelder and Mead (1965) and Parkinson and Hutchinson (1972) for more details. When no further progress can be made the sides of the simplex are reduced in length and the method is repeated.

The method can be slow, but computational bottlenecks have been reduced following Singer and Singer (2004). However, E04CBF is robust, and therefore very useful for functions that are subject to inaccuracies.

There are the following options for successful termination of the method: based only on the function values at the vertices of the current simplex (see (1)); based only on a volume ratio between the current simplex and the initial one (see (2)); or based on which one of the previous two tests passes first. The volume test may be useful if $F$ is discontinuous, while the function-value test should be sufficient on its own if $F$ is continuous.

4 References

5 Parameters

1: \( N \) – INTEGER  
\( \text{Input} \)

\( On \ entry: n \), the number of variables.

\( Constraint: N \geq 1. \)

2: \( X(N) \) – REAL (KIND=nag_wp) array  
\( \text{Input/Output} \)

\( On \ entry: \) 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 (KIND=nag_wp)  
\( \text{Output} \)

\( On \ exit: \) the lowest function value found.

4: \( TOLF \) – REAL (KIND=nag_wp)  
\( \text{Input} \)

\( On \ entry: \) the error tolerable in the function values, in the following sense. If \( f_i \), for \( i = 1, 2, \ldots, n + 1 \), are the individual function values at the vertices of the current simplex, and if \( f_m \) is the mean of these values, then you can request that E04CBF should terminate if

\[
\sqrt{\frac{1}{n+1} \sum_{i=1}^{n+1} (f_i - f_m)^2} < TOLF. \quad (1)
\]

You may specify \( TOLF = 0 \) if you wish to use only the termination criterion (2) on the spatial values: see the description of TOLX.

\( Constraint: TOLF \) must be greater than or equal to the \textit{machine precision} (see Chapter X02), or if \( TOLF \) equals zero then TOLX must be greater than or equal to the \textit{machine precision}.

5: \( TOLX \) – REAL (KIND=nag_wp)  
\( \text{Input} \)

\( On \ entry: \) the error tolerable in the spatial values, in the following sense. If \( LV \) denotes the ‘linearized’ volume of the current simplex, and if \( LV_{\text{init}} \) denotes the ‘linearized’ volume of the initial simplex, then you can request that E04CBF should terminate if

\[
\frac{LV}{LV_{\text{init}}} < TOLX. \quad (2)
\]

You may specify \( TOLX = 0 \) if you wish to use only the termination criterion (1) on function values: see the description of TOLF.

\( Constraint: TOLX \) must be greater than or equal to the \textit{machine precision} (see Chapter X02), or if \( TOLX \) equals zero then TOLF must be greater than or equal to the \textit{machine precision}.

6: \( \text{FUNCT} \) – SUBROUTINE, supplied by the user.  
\( \text{External Procedure} \)

\( \text{FUNCT} \) must evaluate the function \( F \) at a specified point. It should be tested separately before being used in conjunction with E04CBF.

The specification of \( \text{FUNCT} \) is:

```
SUBROUTINE FUNCT (N, XC, FC, IUSER, RUSER)
INTEGER N, IUSER(*)
REAL (KIND=nag_wp) XC(N), FC, RUSER(*)
```

INUENT

On entry: \( n \), the number of variables.

2: \( \text{XC(N)} \) – REAL (KIND=nag_wp) array

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

3: \( \text{FC} \) – REAL (KIND=nag_wp)

Output

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

4: \( \text{IUSER(*)} \) – INTEGER array

User Workspace

5: \( \text{RUSER(*)} \) – REAL (KIND=nag_wp) array

User Workspace

FUNCT is called with the parameters IUSER and RUSER as supplied to E04CBF. You are free to use the arrays IUSER and RUSER to supply information to FUNCT as an alternative to using COMMON global variables.

FUNCT must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub)program from which E04CBF is called. Parameters denoted as Input must not be changed by this procedure.

7: \( \text{MONIT} \) – SUBROUTINE, supplied by the NAG Library or the user. External Procedure

MONIT may be used to monitor the optimization process. It is invoked once every iteration. If no monitoring is required, MONIT may be the dummy monitoring routine E04CBK supplied by the NAG Library.

The specification of MONIT is:

```
SUBROUTINE MONIT (FMIN, FMAX, SIM, N, NCALL, SERROR, VRATIO, &
                   IUSER, RUSER)
   INTEGER N, NCALL, IUSER(*)
   REAL (KIND=nag_wp) FMIN, FMAX, SIM(N+1,N), SERROR, VRATIO, &
                   RUSER(*)
```

1: \( \text{FMIN} \) – REAL (KIND=nag_wp)

Input

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

2: \( \text{FMAX} \) – REAL (KIND=nag_wp)

Input

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

3: \( \text{SIM(N+1,N)} \) – REAL (KIND=nag_wp) array

Input

On entry: the \( n + 1 \) position vectors of the current simplex.

4: \( \text{N} \) – INTEGER

Input

On entry: \( n \), the number of variables.

5: \( \text{NCALL} \) – INTEGER

Input

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

6: \( \text{SERROR} \) – REAL (KIND=nag_wp)

Input

On entry: the current value of the standard deviation in function values used in termination test (1).
MONIT must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub)program from which E04CBF is called. Parameters denoted as Input must not be changed by this procedure.

8: MAXCAL – INTEGER
   Input
   On entry: the maximum number of function evaluations to be allowed.
   Constraint: MAXCAL ≥ 1.

9: IUSER(*) – INTEGER array
   User Workspace
10: RUSER(*) – REAL (KIND=nag_wp) array
    User Workspace

IUSER and RUSER are not used by E04CBF, but are passed directly to FUNCT and MONIT and may be used to pass information to these routines as an alternative to using COMMON global variables.

11: IFAIL – INTEGER
    Input/Output
    On entry: IFAIL must be set to 0, −1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction 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 undesirabile, then the value 1 is recommended. Otherwise, if you are 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.
    On exit: 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 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, MAXCAL = ⟨value⟩.
Constraint: MAXCAL ≥ 1.
On entry, N = ⟨value⟩.
Constraint: N ≥ 1.
On entry, TOLF = 0.0 and TOLX = ⟨value⟩.
Constraint: if TOLF = 0.0 then TOLX is greater than or equal to the machine precision.
On entry, TOLF = ⟨value⟩ and TOLX = ⟨value⟩.
Constraint: if TOLF ≠ 0.0 and TOLX ≠ 0.0 then both should be greater than or equal to the machine precision.
On entry, TOLX = 0.0 and TOLF = \langle value\rangle.
Constraint: if TOLX = 0.0 then TOLF is greater than or equal to the machine precision.

IFAIL = 2
MAXCAL function evaluations have been completed without any other termination test passing.
Check the coding of FUNCT before increasing the value of MAXCAL.

IFAIL = −99
An unexpected error has been triggered by this routine. Please contact NAG.
See Section 3.8 in the Essential Introduction for further information.

IFAIL = −399
Your licence key may have expired or may not have been installed correctly.
See Section 3.7 in the Essential Introduction for further information.

IFAIL = −999
Dynamic memory allocation failed.
See Section 3.6 in the Essential Introduction for further information.

7 Accurate
On a successful exit the accuracy will be as defined by TOLF or TOLX, depending on which criterion
was satisfied first.

8 Parallelism and Performance
Not applicable.

9 Further Comments
Local workspace arrays of fixed lengths (depending on N) are allocated internally by E04CBF. The total
size of these arrays amounts to \( N^2 + 6N + 2 \) real elements.
The time taken by E04CBF 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.

10 Example
This example finds a minimum of the function
\[
F(x_1, x_2) = e^{x_1} (4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1).
\]
This example uses the initial point \((-1, 1)\) (see Section 10.3), and we expect to reach the minimum at
\((0.5, -1)\).

10.1 Program Text
1 E04CBF Example Program Text
1 Mark 25 Release. NAG Copyright 2014.
Module e04cbfe_mod

1 E04CBF Example Program Module:
1 Parameters and User-defined Routines

1 .. Use Statements ..
Use nag_library, Only: nag_wp
Implicit None

Public :: funct, monit

Contains

Subroutine funct(n,xc,fc,iuser,ruser)

Real (Kind=nag_wp), Intent (Out) :: fc
Integer, Intent (In) :: n

Real (Kind=nag_wp), Intent (Inout) :: ruser(*)
Real (Kind=nag_wp), Intent (In) :: xc(n)
Integer, Intent (Inout) :: iuser(*)

Intrinsic :: exp

fc = exp(xc(1))*(4.0_nag_wp*xc(1)*(xc(1)+xc(2))+2.0_nag_wp*xc(2)*(xc(2 & )+1.0_nag_wp)+1.0_nag_wp)

Return

End Subroutine funct

Subroutine monit(fmin,fmax,sim,n,ncall,serror,vratio,iuser,ruser)

Real (Kind=nag_wp), Intent (In) :: fmax, fmin, serror, vratio
Integer, Intent (In) :: n, ncall

Real (Kind=nag_wp), Intent (Inout) :: ruser(*)
Real (Kind=nag_wp), Intent (In) :: sim(n+1,n)
Integer, Intent (Inout) :: iuser(*)

Write (nout,*)
Write (nout,99999) ncall
Write (nout,99998) fmin
Write (nout,99997) fmax
Write (nout,99996) sim(1:(n+1),1:n)
Write (nout,99995) serror
Write (nout,99994) vratio

Return

99999 Format (1X,'There have been',I5,' function calls'
99998 Format (1X,'The smallest function value is',F10.4)
99997 Format (1X,'The simplex is'
99996 Format (1X,2F10.4)
99995 Format (1X,'The standard deviation in function values at the ', &
   'vertices of the simplex is',F10.4)
99994 Format (1X,'The linearized volume ratio of the current simplex', &
   ' to the starting one is',F10.4)

End Subroutine monit

End Module e04cbfe_mod

Program e04cbfe

E04CBF Example Main Program

Use nag_library, Only: e04cbf, e04cbk, nag_wp, x02ajf
Use e04cbfe_mod, Only: funct, monit, nout

Implicit None

Integer, Parameter :: n = 2

Real (Kind=nag_wp) :: f, tolf, tolx
Integer :: ifail, maxcal
Logical :: monitoring
! .. Local Arrays ..
Real (Kind=nag_wp) :: ruser(1), x(n)
Integer :: iuser(1)
! .. Intrinsic Procedures ..
Intrinsic :: sqrt
! .. Executable Statements ..
Write (nout,*), 'E04CBF Example Program Results'
Set MONITORING to .TRUE. to obtain monitoring information
monitoring = .False.
x(1:n) = (-1.0_nag_wp, 1.0_nag_wp/)
tolf = sqrt(x02ajf())
tolx = sqrt(tolx)
maxcal = 100
ifail = 0
If (.Not. monitoring) Then
   Call e04cbf(n,x,f,tolf,tolx,funct,e04cbk,maxcal,iuser,ruser,ifail)
Else
   Call e04cbf(n,x,f,tolf,tolx,funct,monit,maxcal,iuser,ruser,ifail)
End If
Write (nout,*), f
Write (nout,99999) x(1:n)
End Program e04cbfe

10.2 Program Data
None.

10.3 Program Results
E04CBF Example Program Results
The final function value is 0.0000
at the point 0.5000 -0.9999
Example Program
Contours of $F$ Showing the Initial Point (X) and Local Minimum (*)