E04UDF – 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

To supply optional parameters to E04UCF from an external file.

2 Specification

```
SUBROUTINE E04UDF(IOPTNS, INFORM)
INTEGER           IOPTNS, INFORM
```

3 Description

E04UDF may be used to supply values for optional parameters to E04UCF. E04UDF reads an external file and each line of the file defines a single optional parameter. It is only necessary to supply values for those parameters whose values are to be different from their default values.

Each optional parameter is defined by a single character string of up to 72 characters, consisting of one or more items. The items associated with a given option must be separated by spaces, or equal signs [=]. Alphabetic characters may be upper or lower case. The string

```
Print level = 1
```

is an example of a string used to set an optional parameter. For each option the string contains one or more of the following items:

(a) A mandatory keyword.

(b) A phrase that qualifies the keyword.

(c) A number that specifies an INTEGER or ***real*** value. Such numbers may be up to 16 contiguous characters in Fortran 77's I, F, E or D formats, terminated by a space if this is not the last item on the line.

Blank strings and comments are ignored. A comment begins with an asterisk (*) and all subsequent characters in the string are regarded as part of the comment.

The file containing the options must start with **begin** and must finish with **end**. An example of a valid options file is:

```
Begin * Example options file
  Print level = 5
End
```

Normally each line of the file is printed as it is read, on the current advisory message unit (see X04ABF), but printing may be suppressed using the keyword **nolist**. To suppress printing of **begin**, **nolist** must be the first option supplied as in the file:

```
Begin
  Nolist
  Print level = 5
End
```

Printing will automatically be turned on again after a call to E04UCF and may be turned on again at any time by the user by using the keyword **list**.

Optional parameter settings are preserved following a call to E04UCF, and so the keyword **defaults** is provided to allow the user to reset all the optional parameters to their default values prior to a subsequent call to E04UCF.

A complete list of optional parameters, their abbreviations, synonyms and default values is given in Section 11 of the document for E04UCF.

4 References

None.

5 Parameters

1: IOPTNS — INTEGER *Input*

On entry: the unit number of the options file to be read.

Constraint: $0 \leq \text{IOPTNS} \leq 99$.

2: INFORM — INTEGER *Output*

On exit: contains zero if an options file with the correct structure has been read and a value > 0 otherwise, as indicated below.

INFORM = 1

IOPTNS is not in the range $[0, 99]$.

INFORM = 2

begin was found, but end-of-file was found before **end** was found.

INFORM = 3

end-of-file was found before **begin** was found.

6 Error Indicators and Warnings

If a line is not recognized as a valid option, then a warning message is output on the current advisory message unit (see X04ABF).

7 Accuracy

Not applicable.

8 Further Comments

E04UEF may also be used to supply optional parameters to E04UCF.

9 Example

This example solves the same problen as the example for E04UCF, but in addition illustrates the use of E04UDF and E04UEF to set optional parameters for E04UCF.

In this example the options file read by E04UDF is appended to the data file for the program (see Section 9.2). It would usually be more convenient in practice to keep the data file and the options file separate.

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.

```
*     E04UDF Example Program Text
*     Mark 16 Release. NAG Copyright 1993.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, NCLMAX, NCNMAX
      PARAMETER        (NMAX=10,NCLMAX=10,NCNMAX=10)
      INTEGER          LDA, LDCJ, LDR
      PARAMETER        (LDA=NCLMAX,LDCJ=NCNMAX,LDR=NMAX)
      INTEGER          LIWORK, LWORK
      PARAMETER        (LIWORK=100,LWORK=1000)
*     .. Local Scalars ..
      real             OBJF
      INTEGER          I, IFAIL, INFORM, ITER, J, N, NCLIN, NCNLN
*     .. Local Arrays ..
      real             A(LDA,NMAX), BL(NMAX+NCLMAX+NCNMAX),
     +                 BU(NMAX+NCLMAX+NCNMAX), C(NCNMAX),
     +                 CJAC(LDCJ,NMAX), CLAMDA(NMAX+NCLMAX+NCNMAX),
     +                 OBJGRD(NMAX), R(LDR,NMAX), USER(1), WORK(LWORK),
     +                 X(NMAX)
      INTEGER          ISTATE(NMAX+NCLMAX+NCNMAX), IUSER(1),
     +                 IWORK(LIWORK)
*     .. External Subroutines ..
      EXTERNAL         CONFUN, E04UCF, E04UDF, E04UEF, OBJFUN, X04ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'E04UDF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, NCLIN, NCNLN
      IF (N.LE.NMAX .AND. NCLIN.LE.NCLMAX .AND. NCNLN.LE.NCNMAX) THEN
*
*        Read A, BL, BU and X from data file
*
         IF (NCLIN.GT.0) READ (NIN,*) ((A(I,J),J=1,N),I=1,NCLIN)
         READ (NIN,*) (BL(I),I=1,N+NCLIN+NCNLN)
         READ (NIN,*) (BU(I),I=1,N+NCLIN+NCNLN)
         READ (NIN,*) (X(I),I=1,N)
*
*        Set three options using E04UEF
*
         CALL E04UEF(' Infinite Bound Size = 1.0D+25 ')
*
         CALL E04UEF(' Print Level = 1 ')
*
         CALL E04UEF(' Verify Level = -1 ')
*
*        Set the unit number for advisory messages to NOUT
*
         CALL X04ABF(1,NOUT)
*
*        Read the options file for the remaining options
*
         CALL E04UDF(NIN,INFORM)
*
```

```
         IF (INFORM.NE.0) THEN
            WRITE (NOUT,99999) 'E04UDF terminated with INFORM = ',
     +        INFORM
            STOP
         END IF
*
*        Solve the problem
*
         IFAIL = -1
*
         CALL E04UCF(N,NCLIN,NCNLN,LDA,LDCJ,LDR,A,BL,BU,CONFUN,OBJFUN,
     +               ITER,ISTATE,C,CJAC,CLAMDA,OBJF,OBJGRD,R,X,IWORK,
     +               LIWORK,WORK,LWORK,IUSER,USER,IFAIL)
*
      END IF
      STOP
*
99999 FORMAT (1X,A,I3)
      END
      SUBROUTINE OBJFUN(MODE,N,X,OBJF,OBJGRD,NSTATE,IUSER,USER)
*     Routine to evaluate objective function and its 1st derivatives.
*     .. Parameters ..
      real              ONE, TWO
      PARAMETER         (ONE=1.0e0,TWO=2.0e0)
*     .. Scalar Arguments ..
      real              OBJF
      INTEGER           MODE, N, NSTATE
*     .. Array Arguments ..
      real              OBJGRD(N), USER(*), X(N)
      INTEGER           IUSER(*)
*     .. Executable Statements ..
      IF (MODE.EQ.0 .OR. MODE.EQ.2) OBJF = X(1)*X(4)*(X(1)+X(2)+X(3)) +
     +    X(3)
*
      IF (MODE.EQ.1 .OR. MODE.EQ.2) THEN
         OBJGRD(1) = X(4)*(TWO*X(1)+X(2)+X(3))
         OBJGRD(2) = X(1)*X(4)
         OBJGRD(3) = X(1)*X(4) + ONE
         OBJGRD(4) = X(1)*(X(1)+X(2)+X(3))
      END IF
*
      RETURN
      END
*
      SUBROUTINE CONFUN(MODE,NCNLN,N,LDCJ,NEEDC,X,C,CJAC,NSTATE,IUSER,
     +                  USER)
*     Routine to evaluate the nonlinear constraints and their 1st
*     derivatives.
*     .. Parameters ..
      real              ZERO, TWO
      PARAMETER         (ZERO=0.0e0,TWO=2.0e0)
*     .. Scalar Arguments ..
      INTEGER           LDCJ, MODE, N, NCNLN, NSTATE
*     .. Array Arguments ..
      real              C(*), CJAC(LDCJ,*), USER(*), X(N)
      INTEGER           IUSER(*), NEEDC(*)
*     .. Local Scalars ..
      INTEGER           I, J
```

```
*     .. Executable Statements ..
      IF (NSTATE.EQ.1) THEN
*        First call to CONFUN.  Set all Jacobian elements to zero.
*        Note that this will only work when 'Derivative Level = 3'
*        (the default; see Section 11.2).
         DO 40 J = 1, N
            DO 20 I = 1, NCNLN
               CJAC(I,J) = ZERO
   20       CONTINUE
   40    CONTINUE
      END IF
*
      IF (NEEDC(1).GT.0) THEN
         IF (MODE.EQ.0 .OR. MODE.EQ.2) C(1) = X(1)**2 + X(2)**2 + X(3)
     +       **2 + X(4)**2
         IF (MODE.EQ.1 .OR. MODE.EQ.2) THEN
            CJAC(1,1) = TWO*X(1)
            CJAC(1,2) = TWO*X(2)
            CJAC(1,3) = TWO*X(3)
            CJAC(1,4) = TWO*X(4)
         END IF
      END IF
*
      IF (NEEDC(2).GT.0) THEN
         IF (MODE.EQ.0 .OR. MODE.EQ.2) C(2) = X(1)*X(2)*X(3)*X(4)
         IF (MODE.EQ.1 .OR. MODE.EQ.2) THEN
            CJAC(2,1) = X(2)*X(3)*X(4)
            CJAC(2,2) = X(1)*X(3)*X(4)
            CJAC(2,3) = X(1)*X(2)*X(4)
            CJAC(2,4) = X(1)*X(2)*X(3)
         END IF
      END IF
*
      RETURN
      END
```

9.2 Program Data

```
E04UDF Example Program Data
  4   1   2                            :Values of N, NCLIN and NCNLN
  1.0   1.0   1.0   1.0                              :End of matrix A
  1.0   1.0   1.0   1.0  -1.0E+25  -1.0E+25  25.0       :End of BL
  5.0   5.0   5.0   5.0  20.0      40.0       1.0E+25   :End of BU
  1.0   5.0   5.0   1.0                              :End of X
Begin   Example options file for  E04UDF
   Major Iteration Limit   =  15      *  (Default =  50)
   Minor Iteration Limit   =  10      *  (Default =  50)
End
```

9.3 Program Results

```
E04UDF Example Program Results


Calls to E04UEF
---------------

      Infinite Bound Size = 1.0E+25
      Print Level = 1
      Verify Level = -1


OPTIONS file
------------

     Begin   Example options file for  E04UDF
        Major Iteration Limit   =  15      *  (Default =  50)
        Minor Iteration Limit   =  10      *  (Default =  50)
     End

*** E04UCF
*** Start of NAG Library implementation details ***

Implementation title: Generalised Base Version
           Precision: FORTRAN double precision
        Product Code: FLBAS19D
                Mark: 19A

*** End of NAG Library implementation details ***

Parameters
----------

Linear constraints.....         1       Variables.............         4
Nonlinear constraints..         2

Infinite bound size....  1.00E+25       COLD start............
Infinite step size.....  1.00E+25       EPS (machine precision)  1.11E-16
Step limit............  2.00E+00       Hessian...............        NO

Linear feasibility.....  1.05E-08       Crash tolerance........  1.00E-02
Nonlinear feasibility..  1.05E-08       Optimality tolerance...  3.26E-12
Line search tolerance..  9.00E-01       Function precision.....  4.37E-15

Derivative level.......         3       Monitoring file........        -1
Verify level..........        -1

Major iterations limit.        15       Major print level......         1
Minor iterations limit.        10       Minor print level......         0

Workspace provided is     IWORK(     100),  WORK(    1000).
To solve problem we need  IWORK(      17),  WORK(     185).

Exit from NP problem after     5 major iterations,
                               9 minor iterations.
```

```
Varbl State     Value       Lower Bound   Upper Bound    Lagr Mult   Slack

V   1    LL    1.00000       1.00000       5.00000       1.088         .
V   2    FR    4.74300       1.00000       5.00000         .       0.2570
V   3    FR    3.82115       1.00000       5.00000         .        1.179
V   4    FR    1.37941       1.00000       5.00000         .       0.3794


L Con State     Value       Lower Bound   Upper Bound    Lagr Mult   Slack

L   1    FR    10.9436          None       20.0000         .        9.056


N Con State     Value       Lower Bound   Upper Bound    Lagr Mult   Slack

N   1    UL    40.0000          None       40.0000     -0.1615     -3.5264E-11
N   2    LL    25.0000       25.0000          None      0.5523     -2.8791E-11

Exit E04UCF - Optimal solution found.

Final objective value =    17.01402
```