NAG Library Routine Document

E04UDF/E04UDA

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/E04UCA from an external file. More precisely, E04UDF must be used to supply optional parameters to E04UCF and E04UDA must be used to supply optional parameters to E04UCA.

E04UDA is a version of E04UDF that has additional arguments in order to make it safe for use in multithreaded applications (see Section 5). The initialization routine E04WBF **must** have been called before calling E04UDA.

E04UDF/E04UDA can also be used to supply optional parameters to E04UFF/E04UFA.

2 Specification

2.1 Specification for E04UDF

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

2.2 Specification for E04UDA

```
SUBROUTINE E04UDA (IOPTNS, LWSAV, IWSAV, RWSAV, INFORM)
INTEGER            IOPTNS, IWSAV(610), INFORM
REAL (KIND=nag_wp) RWSAV(475)
LOGICAL            LWSAV(120)
```

3 Description

E04UDF/E04UDA may be used to supply values for optional parameters to E04UCF/E04UCA. E04UDF/E04UDA reads an external file and each line of the file defines a single optional parameter. It is only necessary to supply values for those arguments 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 equals 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 mandatory keyword;

– a phrase that qualifies the keyword;

– a number that specifies an integer or real value. Such numbers may be up to 40 contiguous characters in Fortran'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
```

For E04UDF each line of the file is normally 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 or E04UDF and may be turned on again at any time using the keyword **List**.

For E04UDA printing is turned off by default, but may be turned on at any time using the keyword **List**.

Optional parameter settings are preserved following a call to E04UCF/E04UCA and so the keyword **Defaults** is provided to allow you to reset all the optional parameters to their default values before a subsequent call to E04UCF/E04UCA.

A complete list of optional parameters, their abbreviations, synonyms and default values is given in Section 12 in E04UCF/E04UCA.

4 References

None.

5 Arguments

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*

Note: *for E04UDA, INFORM does not occur in this position in the argument list. See the additional arguments described below.*

On exit: contains zero if the options file has been successfully read and a value > 0 otherwise (see Section 6).

Note: *the following are additional arguments for specific use with E04UDA. Users of E04UDF therefore need not read the remainder of this description.*

3: LWSAV(120) – LOGICAL array *Communication Array*

4: IWSAV(610) – INTEGER array *Communication Array*

5: RWSAV(475) – REAL (KIND=nag_wp) array *Communication Array*

The arrays LWSAV, IWSAV and RWSAV **must not** be altered between calls to any of the routines E04UDA, E04UCA, E04UEA and E04WBF.

6: INFORM – INTEGER *Output*

Note: see the argument description for INFORM above.

6 Error Indicators and Warnings

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.

INFORM $= 4$

Not used.

INFORM $= 5$

One or more lines of the options file is invalid. Check that all keywords are neither ambiguous nor misspelt.

7 Accuracy

Not applicable.

8 Parallelism and Performance

E04UDF/E04UDA is not threaded in any implementation.

9 Further Comments

E04UEF/E04UEA may also be used to supply optional parameters to E04UCF/E04UCA.

10 Example

This example solves the same problem as the example for E04UCF/E04UCA, but in addition illustrates the use of E04UDF/E04UDA and E04UEF/E04UEA to set optional parameters for E04UCF/E04UCA.

In this example the options file read by E04UDF/E04UDA is appended to the data file for the program (see Section 10.2). It would usually be more convenient in practice to keep the data file and the options file separate.

10.1 Program Text

the following program illustrates the use of E04UDF. An equivalent program illustrating the use of E04UDA is available with the supplied Library and is also available from the NAG web site.

```
!   E04UDF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

    Module e04udfe_mod

!     E04UDF Example Program Module:
!            Parameters and User-defined Routines

!     .. Use Statements ..
      Use nag_library, Only: nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Accessibility Statements ..
      Private
      Public                            :: confun, objfun
!     .. Parameters ..
      Integer, Parameter, Public        :: iset = 1, nin = 5, ninopt = 7,       &
                                           nout = 6
    Contains
      Subroutine objfun(mode,n,x,objf,objgrd,nstate,iuser,ruser)
!       Routine to evaluate objective function and its 1st derivatives.
```

```
!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (Out) :: objf
        Integer, Intent (Inout)        :: mode
        Integer, Intent (In)           :: n, nstate
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Inout) :: objgrd(n), ruser(*)
        Real (Kind=nag_wp), Intent (In) :: x(n)
        Integer, Intent (Inout)        :: iuser(*)
!       .. Executable Statements ..
        If (mode==0 .Or. mode==2) Then
          objf = x(1)*x(4)*(x(1)+x(2)+x(3)) + x(3)
        End If

        If (mode==1 .Or. mode==2) Then
          objgrd(1) = x(4)*(2.0E0_nag_wp*x(1)+x(2)+x(3))
          objgrd(2) = x(1)*x(4)
          objgrd(3) = x(1)*x(4) + 1.0E0_nag_wp
          objgrd(4) = x(1)*(x(1)+x(2)+x(3))
        End If

        Return

      End Subroutine objfun
      Subroutine confun(mode,ncnln,n,ldcj,needc,x,c,cjac,nstate,iuser,ruser)
!       Routine to evaluate the nonlinear constraints and their 1st
!       derivatives.

!       .. Scalar Arguments ..
        Integer, Intent (In)           :: ldcj, n, ncnln, nstate
        Integer, Intent (Inout)        :: mode
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Out) :: c(ncnln)
        Real (Kind=nag_wp), Intent (Inout) :: cjac(ldcj,n), ruser(*)
        Real (Kind=nag_wp), Intent (In) :: x(n)
        Integer, Intent (Inout)        :: iuser(*)
        Integer, Intent (In)           :: needc(ncnln)
!       .. Executable Statements ..
        If (nstate==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).

          cjac(1:ncnln,1:n) = 0.0E0_nag_wp
        End If

        If (needc(1)>0) Then

          If (mode==0 .Or. mode==2) Then
            c(1) = x(1)**2 + x(2)**2 + x(3)**2 + x(4)**2
          End If

          If (mode==1 .Or. mode==2) Then
            cjac(1,1) = 2.0E0_nag_wp*x(1)
            cjac(1,2) = 2.0E0_nag_wp*x(2)
            cjac(1,3) = 2.0E0_nag_wp*x(3)
            cjac(1,4) = 2.0E0_nag_wp*x(4)
          End If

        End If

        If (needc(2)>0) Then

          If (mode==0 .Or. mode==2) Then
            c(2) = x(1)*x(2)*x(3)*x(4)
          End If

          If (mode==1 .Or. mode==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 Subroutine confun
    End Module e04udfe_mod
    Program e04udfe

!     E04UDF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: e04ucf, e04udf, e04uef, nag_wp, x04abf, x04acf,   &
                             x04baf
      Use e04udfe_mod, Only: confun, iset, nin, ninopt, nout, objfun
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Character (*), Parameter         :: fname = 'e04udfe.opt'
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: objf
      Integer                          :: i, ifail, inform, iter, lda, ldcj,   &
                                          ldr, liwork, lwork, mode, n, nclin,  &
                                          ncnln, outchn, sda, sdcjac
      Character (80)                   :: rec
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:), bl(:), bu(:), c(:),          &
                                          cjac(:,:), clamda(:), objgrd(:),     &
                                          r(:,:), work(:), x(:)
      Real (Kind=nag_wp)               :: ruser(1)
      Integer, Allocatable             :: istate(:), iwork(:)
      Integer                          :: iuser(1)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max
!     .. Executable Statements ..
      Write (rec,99998) 'E04UDF Example Program Results'
      Call x04baf(nout,rec)

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

      Read (nin,*) n, nclin, ncnln
      liwork = 3*n + nclin + 2*ncnln
      lda = max(1,nclin)

      If (nclin>0) Then
        sda = n
      Else
        sda = 1
      End If

      ldcj = max(1,ncnln)

      If (ncnln>0) Then
        sdcjac = n
      Else
        sdcjac = 1
      End If

      ldr = n

      If (ncnln==0 .And. nclin>0) Then
        lwork = 2*n**2 + 20*n + 11*nclin
      Else If (ncnln>0 .And. nclin>=0) Then
        lwork = 2*n**2 + n*nclin + 2*n*ncnln + 20*n + 11*nclin + 21*ncnln
      Else
        lwork = 20*n
      End If
```

```
      Allocate (istate(n+nclin+ncnln),iwork(liwork),a(lda,sda),                &
        bl(n+nclin+ncnln),bu(n+nclin+ncnln),c(max(1,                           &
        ncnln)),cjac(ldcj,sdcjac),clamda(n+nclin+ncnln),objgrd(n),r(ldr,n),    &
        x(n),work(lwork))

      If (nclin>0) Then
        Read (nin,*)(a(i,1:sda),i=1,nclin)
      End If

      Read (nin,*) bl(1:(n+nclin+ncnln))
      Read (nin,*) bu(1:(n+nclin+ncnln))
      Read (nin,*) x(1:n)

!     Set the unit number for advisory messages to OUTCHN

      outchn = nout
      Call x04abf(iset,outchn)

!     Set three options using E04UEF

      Call e04uef(' Infinite Bound Size = 1.0D+25 ')

      Call e04uef(' Print Level = 1 ')

      Call e04uef(' Verify Level = -1 ')

!     Open the options file for reading

      mode = 0

      ifail = 0
      Call x04acf(ninopt,fname,mode,ifail)

!     Read the options file for the remaining options

      Call e04udf(ninopt,inform)

      If (inform/=0) Then
        Write (rec,99999) 'E04UDF terminated with ' // 'INFORM = ', inform
        Call x04baf(nout,rec)
      End If

!     Solve the problem

      ifail = 0
      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,    &
        ruser,ifail)

99999 Format (1X,A,I5)
99998 Format (1X,A)
    End Program e04udfe
```

10.2 Program Data

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

```
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
```

10.3 Program Results

```
E04UDF Example Program Results


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

      Infinite Bound Size = 1.0D+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

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

Start point
   1.000000E+00   5.000000E+00   5.000000E+00   1.000000E+00

Workspace provided is     IWORK(      17),  WORK(     185).
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
```