NAG Library Routine Document

E04NLF/E04NLA

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 E04NKF/E04NKA from an external file. More precisely, E04NLF must be used to supply optional parameters to E04NKF and E04NLA must be used to supply optional parameters to E04NKA.

E04NLA is a version of E04NLF 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 E04NLA.

2 Specification

2.1 Specification for E04NLF

```
SUBROUTINE E04NLF (IOPTNS, INFORM)
INTEGER IOPTNS, INFORM
```

2.2 Specification for E04NLA

```
SUBROUTINE E04NLA (IOPTNS, LWSAV, IWSAV, RWSAV, INFORM)
INTEGER          IOPTNS, IWSAV(380), INFORM
REAL (KIND=nag_wp) RWSAV(285)
LOGICAL          LWSAV(20)
```

3 Description

E04NLF/E04NLA may be used to supply values for optional parameters to E04NKF/E04NKA. E04NLF/E04NLA 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 E04NLF 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 E04NKF or E04NLF and may be turned on again at any time using the keyword **List**.

For E04NLA 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 E04NKF/E04NKA 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 E04NKF/E04NKA.

A complete list of optional parameters, their abbreviations, synonyms and default values is given in Section 12 in E04NKF/E04NKA.

4 References

None.

5 Arguments

1:     IOPTNS – INTEGER   *Input*

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

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

2:     INFORM – INTEGER   *Output*

Note: *for E04NLA, 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 E04NLA. Users of E04NLF therefore need not read the remainder of this description.*

3:     LWSAV(20) – LOGICAL array   *Communication Array*
4:     IWSAV(380) – INTEGER array   *Communication Array*
5:     RWSAV(285) – REAL (KIND=nag_wp) array   *Communication Array*

The arrays LWSAV, IWSAV and RWSAV **must not** be altered between calls to any of the routines E04NLA, E04NKA, E04NMA or 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

E04NLF/E04NLA is not threaded in any implementation.

9   Further Comments

E04NMF/E04NMA may also be used to supply optional parameters to E04NKF/E04NKA.

10   Example

This example solves the same problem as the example for E04NKF/E04NKA, but in addition illustrates the use of E04NLF/E04NLA and E04NMF/E04NMA to set optional parameters for E04NKF/E04NKA.

In this example the options file read by E04NLF/E04NLA 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 E04NLF. An equivalent program illustrating the use of E04NLA is available with the supplied Library and is also available from the NAG web site.

```
!   E04NLF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

    Module e04nlfe_mod

!     E04NLF 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                              :: qphx
!     .. Parameters ..
    Integer, Parameter, Public          :: iset = 1, nin = 5, ninopt = 7,      &
                                           nout = 6
  Contains
    Subroutine qphx(nstate,ncolh,x,hx)
```

```
!        Routine to compute H*x. (In this version of QPHX, the Hessian
!        matrix H is not referenced explicitly.)

!        .. Scalar Arguments ..
         Integer, Intent (In)            :: ncolh, nstate
!        .. Array Arguments ..
         Real (Kind=nag_wp), Intent (Out) :: hx(ncolh)
         Real (Kind=nag_wp), Intent (In) :: x(ncolh)
!        .. Executable Statements ..
         If (nstate==1) Then

!          First entry.

           Write (nout,*)
           Write (nout,99999) ncolh
           Flush (nout)
         End If

         hx(1) = 2.0E0_nag_wp*x(1)
         hx(2) = 2.0E0_nag_wp*x(2)
         hx(3) = 2.0E0_nag_wp*(x(3)+x(4))
         hx(4) = hx(3)
         hx(5) = 2.0E0_nag_wp*x(5)
         hx(6) = 2.0E0_nag_wp*(x(6)+x(7))
         hx(7) = hx(6)

         If (nstate>=2) Then

!          Final entry.

           Write (nout,*)
           Write (nout,99998)
           Flush (nout)
         End If

         Return

99999    Format (1X,'This is the E04NLF example.   NCOLH =',I4,'.')
99998    Format (1X,'Finished the E04NLF example.')
       End Subroutine qphx
     End Module e04nlfe_mod
     Program e04nlfe

!    E04NLF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: e04nkf, e04nlf, e04nmf, nag_wp, x04abf, x04acf
      Use e04nlfe_mod, Only: iset, nin, ninopt, nout, qphx
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Character (*), Parameter        :: fname = 'e04nlfe.opt'
!     .. Local Scalars ..
      Real (Kind=nag_wp)              :: obj, sinf
      Integer                         :: i, icol, ifail, inform, iobj, jcol, &
                                         leniz, lenz, m, miniz, minz, mode,  &
                                         n, ncolh, ninf, nname, nnz, ns,     &
                                         outchn
      Character (1)                   :: start
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable :: a(:), bl(:), bu(:), clamda(:),      &
                                         xs(:), z(:)
      Integer, Allocatable            :: ha(:), istate(:), iz(:), ka(:)
      Character (8), Allocatable      :: crname(:)
      Character (8)                   :: names(5)
!     .. Executable Statements ..
      Write (nout,99997) 'E04NLF Example Program Results'
      Flush (nout)

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

```
      Read (nin,*) n, m
      Read (nin,*) nnz, iobj, ncolh, start, nname

      Allocate (ha(nnz),ka(n+1),istate(n+m),a(nnz),bl(n+m),bu(n+m),xs(n+m),    &
        clamda(n+m),crname(nname))

      Read (nin,*) names(1:5)
      Read (nin,*) crname(1:nname)

!     Read the matrix A from data file. Set up KA.

      jcol = 1
      ka(jcol) = 1

      Do i = 1, nnz

!       Element ( HA( I ), ICOL ) is stored in A( I ).

        Read (nin,*) a(i), ha(i), icol

        If (icol<jcol) Then

!         Elements not ordered by increasing column index.

          Write (nout,*)
          Flush (nout)
          Write (nout,99998) 'Element in column', icol,                        &
            ' found after element in column', jcol, '. Problem', ' abandoned.'
          Flush (nout)
          Go To 100
        Else If (icol==jcol+1) Then

!         Index in A of the start of the ICOL-th column equals I.

          ka(icol) = i
          jcol = icol
        Else If (icol>jcol+1) Then

!         Index in A of the start of the ICOL-th column equals I,
!         but columns JCOL+1,JCOL+2,...,ICOL-1 are empty. Set the
!         corresponding elements of KA to I.

          ka((jcol+1):icol) = i
          jcol = icol
        End If

      End Do

      ka(n+1) = nnz + 1

!     Columns N,N-1,...,ICOL+1 are empty. Set the corresponding
!     elements of KA accordingly.

      Do i = n, icol + 1, -1
        ka(i) = ka(i+1)
      End Do

      Read (nin,*) bl(1:(n+m))
      Read (nin,*) bu(1:(n+m))

      If (start=='C') Then
        Read (nin,*) istate(1:n)
      Else If (start=='W') Then
        Read (nin,*) istate(1:(n+m))
      End If

      Read (nin,*) xs(1:n)

!     Set the unit number for advisory messages to OUTCHN.
```

```
      outchn = nout
      Call x04abf(iset,outchn)

!     Set three options using E04NMF.

      Call e04nmf(' Check Frequency = 10 ')

      Call e04nmf(' Crash Tolerance = 0.05 ')

      Call e04nmf(' Infinite Bound Size = 1.0E+25 ')

!     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 e04nlf(ninopt,inform)

      If (inform/=0) Then
        Write (nout,99999) 'E04NLF terminated with INFORM = ', inform
        Flush (nout)
        Go To 100
      End If

!     Solve the QP problem.
!     First call is a workspace query

      leniz = 1
      lenz = 1
      Allocate (iz(leniz),z(lenz))

      ifail = 1
      Call e04nkf(n,m,nnz,iobj,ncolh,qphx,a,ha,ka,bl,bu,start,names,nname,     &
        crname,ns,xs,istate,miniz,minz,ninf,sinf,obj,clamda,iz,leniz,z,lenz,  &
        ifail)

      If (ifail/=0 .And. ifail/=12 .And. ifail/=13) Then
        Write (nout,99999) 'Query call to E04NKF failed with IFAIL =', ifail
        Go To 100
      End If

      Deallocate (iz,z)

      lenz = minz
      leniz = miniz
      Allocate (iz(leniz),z(lenz))

      ifail = 0
      Call e04nkf(n,m,nnz,iobj,ncolh,qphx,a,ha,ka,bl,bu,start,names,nname,     &
        crname,ns,xs,istate,miniz,minz,ninf,sinf,obj,clamda,iz,leniz,z,lenz,  &
        ifail)

100   Continue

99999 Format (1X,A,I5)
99998 Format (1X,A,I5,A,I5,A,A)
99997 Format (1X,A)
      End Program e04nlfe
```

10.2 Program Data

```
Begin * Example options file for E04NLF
   Iteration Limit = 25  * (Default = 75)
       Print Level =  1  * (Default = 10)
End

E04NLF Example Program Data
 7  8                :Values of N and M
48  8  7  'C'  15 :Values of NNZ, IOBJ, NCOLH, START and NNAME
'       ' '       '       '  '       '  '       '  '       '       ' :End of NAMES
'...X1...' '...X2...' '...X3...' '...X4...' '...X5...'
'...X6...' '...X7...' '..ROW1..' '..ROW2..' '..ROW3..'
'..ROW4..' '..ROW5..' '..ROW6..' '..ROW7..' '..COST..' :End of CRNAME
    0.02  7  1
    0.02  5  1
    0.03  3  1
    1.00  1  1
    0.70  6  1
    0.02  4  1
    0.15  2  1
 -200.00  8  1
    0.06  7  2
    0.75  6  2
    0.03  5  2
    0.04  4  2
    0.05  3  2
    0.04  2  2
    1.00  1  2
-2000.00  8  2
    0.02  2  3
    1.00  1  3
    0.01  4  3
    0.08  3  3
    0.08  7  3
    0.80  6  3
-2000.00  8  3
    1.00  1  4
    0.12  7  4
    0.02  3  4
    0.02  4  4
    0.75  6  4
    0.04  2  4
-2000.00  8  4
    0.01  5  5
    0.80  6  5
    0.02  7  5
    1.00  1  5
    0.02  2  5
    0.06  3  5
    0.02  4  5
-2000.00  8  5
    1.00  1  6
    0.01  2  6
    0.01  3  6
    0.97  6  6
    0.01  7  6
  400.00  8  6
    0.97  7  7
    0.03  2  7
    1.00  1  7
  400.00  8  7                                                :End of matrix A
 0.0       0.0       4.0E+02   1.0E+02   0.0       0.0       0.0       2.0E+03
-1.0E+26  -1.0E+26  -1.0E+26  -1.0E+26   1.5E+03   2.5E+02  -1.0E+26 :End of BL
 2.0E+02   2.5E+03   8.0E+02   7.0E+02   1.5E+03   1.0E+26   1.0E+26   2.0E+03
 6.0E+01   1.0E+02   4.0E+01   3.0E+01   1.0E+26   3.0E+02   1.0E+26 :End of BU
 0   0   0   0   0   0   0   0  :End of ISTATE
 0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0 :End of XS
```

10.3 Program Results

```
E04NLF Example Program Results


Calls to E04NMF
---------------

      Check Frequency = 10
      Crash Tolerance = 0.05
      Infinite Bound Size = 1.0E+25


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

      Begin * Example options file for E04NLF
         Iteration Limit = 25  * (Default = 75)
             Print Level =  1  * (Default = 10)
      End

Workspace provided is                   IZ(       1),  Z(        1).
To start solving the problem we need  IZ(     428),  Z(      358).

Exit E04NKF - Not enough integer workspace to start solving the problem.

*** E04NKF

Parameters
----------


Frequencies.
Check frequency.........       10       Expand frequency........     10000
Factorization frequency.      100

LP Parameters.
Scale tolerance.........  9.00E-01       Feasibility tolerance...  1.00E-06
Iteration limit.........       25        Scale option............         2
Optimality tolerance....  1.00E-06       Partial price...........        10
Crash tolerance.........  5.00E-02       Pivot tolerance.........  2.04E-11
Crash option............        2

QP objective.
Objective variables.....        7       Hessian columns........         7
Superbasics limit.......        7

Miscellaneous.
Variables...............        7       Linear constraints......         8
LU factor tolerance.....  1.00E+02       LU update tolerance.....  1.00E+01
LU singularity tolerance  2.04E-11       Monitoring file.........        -1
EPS (machine precision).  1.11E-16       Print level.............         1
Infinite bound size.....  1.00E+25       Infinite step size......  1.00E+25
COLD start..............               MINIMIZE................

Workspace provided is                   IZ(     428),  Z(      358).
To start solving the problem we need  IZ(     428),  Z(      358).

This is the E04NLF example.   NCOLH =   7.


Variable State     Value      Lower Bound  Upper Bound  Lagr Mult   Residual

...X1...    LL    0.00000         .          200.00       2361.          .
...X2...    BS    349.399         .          2500.0     -1.2975E-12    349.4
...X3...    SBS   648.853       400.00       800.00     -5.7329E-13    151.1
...X4...    SBS   172.847       100.00       700.00      6.4970E-13    72.85
...X5...    BS    407.521         .          1500.0      9.1881E-13    407.5
...X6...    BS    271.356         .          None       -1.1928E-12    271.4
...X7...    BS    150.023         .          None       -1.4130E-12    150.0
```

Constrnt	State	Value	Lower Bound	Upper Bound	Lagr Mult	Residual
..ROW1..	EQ	2000.00	2000.0	2000.0	-1.2901E+04	.
..ROW2..	BS	49.2316	None	60.000	.	-10.77
..ROW3..	UL	100.000	None	100.00	-2325.	.
..ROW4..	BS	32.0719	None	40.000	.	-7.928
..ROW5..	BS	14.5572	None	30.000	.	-15.44
..ROW6..	LL	1500.00	1500.0	None	1.4455E+04	.
..ROW7..	LL	250.000	250.00	300.00	1.4581E+04	.
..COST..	BS	-2.988690E+06	None	None	-1.000	-2.9887E+06

Finished the E04NLF example.

Exit E04NKF - Optimal QP solution found.

Final QP objective value =   -1847785.

Exit from QP problem after   10 iterations.