NAG Fortran 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 parameters in order to make it safe for use in multithreaded applications (see Section 5 below). The initialisation routine E04WBF **must** have been called prior to calling E04NLA.

2 Specifications

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              RWSAV(285)
LOGICAL           LWSAV(20)
```

3 Description

E04NLF/E04NLA may be used to supply values for optional parameters to the corresponding routines 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 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 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) 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'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 prior to a subsequent call to E04NKF/E04NKA.

A complete list of optional parameters, their abbreviations, synonyms and default values is given in Section 11 of the document for E04NKF/E04NKA.

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*

Note: *for E04NLA, INFORM does not occur in this position in the parameter list. See the additional parameters 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 parameters for specific use with E04NLA. Users of E04NLF therefore need not read the remainder of this section.*

2: LWSAV(20) – LOGICAL array *Workspace*

3: IWSAV(380) – INTEGER array *Workspace*

4: RWSAV(285) – ***real*** array *Workspace*

The arrays LWSAV, IWSAV and RWSAV **must not** be altered between calls to any of the routines E04WBF, E04NKA, E04NLA or E04NMA.

5: INFORM – INTEGER *Output*

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

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

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 Further Comments

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

9 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 E04NLF/E04NLA). 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.

Note: *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 19 Revised. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          IDUMMY
      PARAMETER        (IDUMMY=-11111)
      INTEGER          NMAX, MMAX, NNZMAX, LENIZ, LENZ
      PARAMETER        (NMAX=100,MMAX=100,NNZMAX=100,LENIZ=10000,
     +                 LENZ=10000)
*     .. Local Scalars ..
      real             OBJ, SINF
      INTEGER          I, ICOL, IFAIL, INFORM, IOBJ, J, JCOL, M, MINIZ,
```

```
     +                 MINZ, N, NCOLH, NINF, NNAME, NNZ, NS
      CHARACTER        START
*     .. Local Arrays ..
      real             A(NNZMAX), BL(NMAX+MMAX), BU(NMAX+MMAX),
     +                 CLAMDA(NMAX+MMAX), XS(NMAX+MMAX), Z(LENZ)
      INTEGER          HA(NNZMAX), ISTATE(NMAX+MMAX), IZ(LENIZ),
     +                 KA(NMAX+1)
      CHARACTER*8      CRNAME(NMAX+MMAX), NAMES(5)
*     .. External Subroutines ..
      EXTERNAL         E04NKF, E04NLF, E04NMF, QPHX, X04ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'E04NLF Example Program Results'
*     Skip heading in data file.
      READ (NIN,*)
      READ (NIN,*) N, M
      IF (N.LE.NMAX .AND. M.LE.MMAX) THEN
*
*        Read NNZ, IOBJ, NCOLH, START and NNAME from data file.
*
         READ (NIN,*) NNZ, IOBJ, NCOLH, START, NNAME
*
*        Read NAMES and CRNAME from data file.
*
         READ (NIN,*) (NAMES(I),I=1,5)
         READ (NIN,*) (CRNAME(I),I=1,NNAME)
*
*        Read the matrix A from data file. Set up KA.
*
         JCOL = 1
         KA(JCOL) = 1
         DO 40 I = 1, NNZ
*
*           Element ( HA( I ), ICOL ) is stored in A( I ).
*
            READ (NIN,*) A(I), HA(I), ICOL
*
            IF (ICOL.LT.JCOL) THEN
*
*              Elements not ordered by increasing column index.
*
               WRITE (NOUT,99999) 'Element in column', ICOL,
     +           ' found after element in column', JCOL, '. Problem',
     +           ' abandoned.'
               STOP
            ELSE IF (ICOL.EQ.JCOL+1) THEN
*
*              Index in A of the start of the ICOL-th column equals I.
*
               KA(ICOL) = I
               JCOL = ICOL
            ELSE IF (ICOL.GT.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.
*
               DO 20 J = JCOL + 1, ICOL - 1
                  KA(J) = I
   20          CONTINUE
               KA(ICOL) = I
               JCOL = ICOL
            END IF
   40    CONTINUE
*
         KA(N+1) = NNZ + 1
*
         IF (N.GT.ICOL) THEN
*
*           Columns N,N-1,...,ICOL+1 are empty. Set the corresponding
*           elements of KA accordingly.
*
```

```
            DO 60 I = N, ICOL + 1, -1
               IF (KA(I).EQ.IDUMMY) KA(I) = KA(I+1)
   60       CONTINUE
         END IF
*
*        Read BL, BU, ISTATE and XS from data file.
*
         READ (NIN,*) (BL(I),I=1,N+M)
         READ (NIN,*) (BU(I),I=1,N+M)
         IF (START.EQ.'C') THEN
            READ (NIN,*) (ISTATE(I),I=1,N)
         ELSE IF (START.EQ.'W') THEN
            READ (NIN,*) (ISTATE(I),I=1,N+M)
         END IF
         READ (NIN,*) (XS(I),I=1,N)
*
*        Set three options using E04NMF.
*
         CALL E04NMF(' Check Frequency = 10 ')
*
         CALL E04NMF(' Crash Tolerance = 0.05 ')
*
         CALL E04NMF(' Infinite Bound Size = 1.0E+25 ')
*
*        Set the unit number for advisory messages to NOUT.
*
         CALL X04ABF(1,NOUT)
*
*        Read the options file for the remaining options.
*
         CALL E04NLF(NIN,INFORM)
*
         IF (INFORM.NE.0) THEN
            WRITE (NOUT,99998) 'E04NLF terminated with INFORM = ',
     +        INFORM
            STOP
         END IF
*
*        Solve the QP problem.
*
         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)
*
      END IF
      STOP
*
99999 FORMAT (/1X,A,I5,A,I5,A,A)
99998 FORMAT (1X,A,I3)
      END
*
      SUBROUTINE QPHX(NSTATE,NCOLH,X,HX)
*
*     Routine to compute H*x. (In this version of QPHX, the Hessian
*     matrix H is not referenced explicitly.)
*
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      real             TWO
      PARAMETER        (TWO=2.0e+0)
*     .. Scalar Arguments ..
      INTEGER          NCOLH, NSTATE
*     .. Array Arguments ..
      real             HX(NCOLH), X(NCOLH)
*     .. Executable Statements ..
      IF (NSTATE.EQ.1) THEN
*
*        First entry.
```

```
*
         WRITE (NOUT,99999) NCOLH
*
      END IF
*
*     Normal entry.
*
      HX(1) = TWO*X(1)
      HX(2) = TWO*X(2)
      HX(3) = TWO*(X(3)+X(4))
      HX(4) = HX(3)
      HX(5) = TWO*X(5)
      HX(6) = TWO*(X(6)+X(7))
      HX(7) = HX(6)
*
      IF (NSTATE.GE.2) THEN
*
*        Final entry.
*
         WRITE (NOUT,99998)
*
      END IF
*
      RETURN
*
99999 FORMAT (/' This is the E04NKF example.   NCOLH =',I4,'.')
99998 FORMAT (/' Finished the E04NKF example.')
      END
```

9.2 Program Data

```
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+25  -1.0E+25  -1.0E+25  -1.0E+25  1.5E+03  2.5E+02 -1.0E+25 :End of BL
 2.0E+02   2.5E+03   8.0E+02   7.0E+02  1.5E+03  1.0E+25  1.0E+25   2.0E+03
 6.0E+01   1.0E+02   4.0E+01   3.0E+01  1.0E+25  3.0E+02  1.0E+25 :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
Begin * Example options file for E04NLF
   Iteration Limit = 25  * (Default = 75)
       Print Level =  1  * (Default = 10)
End
```

9.3 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

 *** E04NKF
 *** Start of NAG Library implementation details ***

 Implementation title: Generalised Base Version
            Precision: FORTRAN double precision
         Product Code: FLBAS20D
                 Mark: 20A

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

 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.........  2.05E-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  2.05E-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(   10000),  Z(   10000).
 To start solving the problem we need  IZ(     428),  Z(     358).

 This is the E04NKF 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     -1.0616E-12   349.4
 ...X3...   SBS    648.853        400.00      800.00     -4.3952E-12   151.1
 ...X4...   SBS    172.847        100.00      700.00     -2.2739E-12   72.85
 ...X5...    BS    407.521           .        1500.0     -2.0673E-12   407.5
 ...X6...    BS    271.356           .           None     7.4549E-13   271.4
 ...X7...    BS    150.023           .           None     4.7101E-13   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     -1.3490E-11  -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 E04NKF example.

 Exit E04NKF - Optimal QP solution found.

 Final QP objective value =    -1847785.

 Exit from QP problem after      10 iterations.
```