NAG Library Routine Document

E04NRF

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

E04NRF may be used to supply optional parameters to E04NQF from an external file. The initialization routine E04NPF **must** have been called before calling E04NRF.

2 Specification

```
SUBROUTINE E04NRF (ISPECS, CW, IW, RW, IFAIL)
INTEGER            ISPECS, IW(*), IFAIL
REAL (KIND=nag_wp) RW(*)
CHARACTER(8)       CW(*)
```

3 Description

E04NRF may be used to supply values for optional parameters to E04NQF. E04NRF 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
```

Optional parameter settings are preserved following a call to E04NQF 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 E04NQF.

A complete list of optional parameters, their abbreviations, synonyms and default values is given in Section 12 in E04NQF.

4 References

None.

5 Arguments

1: ISPECS – INTEGER *Input*

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

Constraint: ISPECS is a valid unit open for reading.

2: CW(∗) – CHARACTER(8) array *Communication Array*

Note: the dimension of the array CW must be at least LENCW (see E04NPF).

3: IW(∗) – INTEGER array *Communication Array*

Note: the dimension of the array IW must be at least LENIW (see E04NPF).

4: RW(∗) – REAL (KIND=nag_wp) array *Communication Array*

Note: the dimension of the array RW must be at least LENRW (see E04NPF).

5: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation 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 undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, 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

The initialization routine E04NPF has not been called.

IFAIL = 2

At least one line of the options file is invalid.

Could not read options file on unit ISPECS = ⟨*value*⟩.

Could not read options file on unit ISPECS. This may be due to:

(a) ISPECS is not a valid unit number;

(b) a file is not associated with unit ISPECS, or if it is, is unavailable for read access;

(c) one or more lines of the options file is invalid. Check that all keywords are neither ambiguous nor misspelt;

(d) `Begin` was found, but end-of-file was found before `End` was found;

(e) end-of-file was found before `Begin` was found.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

E04NRF is not threaded in any implementation.

9 Further Comments

E04NSF, E04NTF or E04NUF may also be used to supply optional parameters to E04NQF.

10 Example

This example minimizes the quadratic function $f(x) = c^{\mathrm{T}}x + \frac{1}{2}x^{\mathrm{T}}Hx$, where

$$c = (-200.0, -2000.0, -2000.0, -2000.0, -2000.0, 400.0, 400.0)^{\mathrm{T}}$$

and

$$H = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 2 & 2 \end{pmatrix}$$

subject to the bounds

$$\begin{aligned} 0 &\leq x_1 \leq 200 \\ 0 &\leq x_2 \leq 2500 \\ 400 &\leq x_3 \leq 800 \\ 100 &\leq x_4 \leq 700 \\ 0 &\leq x_5 \leq 1500 \\ 0 &\leq x_6 \\ 0 &\leq x_7 \end{aligned}$$

and to the linear constraints

$$\begin{array}{rcrcrcrcrcrcrcr} & & x_1 & + & x_2 & + & x_3 & + & x_4 & + & x_5 & + & x_6 & + & x_7 & = & 2000 \\ & & 0.15x_1 & + & 0.04x_2 & + & 0.02x_3 & + & 0.04x_4 & + & 0.02x_5 & + & 0.01x_6 & + & 0.03x_7 & \leq & 60 \\ & & 0.03x_1 & + & 0.05x_2 & + & 0.08x_3 & + & 0.02x_4 & + & 0.06x_5 & + & 0.01x_6 & & & \leq & 100 \\ & & 0.02x_1 & + & 0.04x_2 & + & 0.01x_3 & + & 0.02x_4 & + & 0.02x_5 & & & & & \leq & 40 \\ & & 0.02x_1 & + & 0.03x_2 & + & & & & & 0.01x_5 & & & & & \leq & 30 \\ 1500 & \leq & 0.70x_1 & + & 0.75x_2 & + & 0.80x_3 & + & 0.75x_4 & + & 0.80x_5 & + & 0.97x_6 & & & & \\ 250 & \leq & 0.02x_1 & + & 0.06x_2 & + & 0.08x_3 & + & 0.12x_4 & + & 0.02x_5 & + & 0.01x_6 & + & 0.97x_7 & \leq & 300 \end{array}$$

The initial point, which is infeasible, is

$$x_0 = (0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)^{\mathrm{T}}.$$

The optimal solution (to five figures) is

$$x^* = (0.0, 349.40, 648.85, 172.85, 407.52, 271.36, 150.02)^{\mathrm{T}}.$$

One bound constraint and four linear constraints are active at the solution. Note that the Hessian matrix H is positive semidefinite.

10.1 Program Text

```
!   E04NRF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

    Module e04nrfe_mod

!     E04NRF 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         :: lencw = 600, leniw = 600,            &
                                            lenrw = 600, nin = 5, ninopt = 7,    &
                                            nout = 6
    Contains
      Subroutine qphx(ncolh,x,hx,nstate,cuser,iuser,ruser)
!       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 (Inout) :: ruser(*)
        Real (Kind=nag_wp), Intent (In) :: x(ncolh)
        Integer, Intent (Inout)          :: iuser(*)
        Character (8), Intent (Inout)    :: cuser(*)
!       .. Executable Statements ..
        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)

        Return

      End Subroutine qphx
    End Module e04nrfe_mod
    Program e04nrfe

!     E04NRF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: e04npf, e04nqf, e04nrf, e04nsf, e04ntf, e04nuf,   &
                             e04nxf, e04nyf, nag_wp, x04acf
      Use e04nrfe_mod, Only: lencw, leniw, lenrw, nin, ninopt, nout, qphx
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Character (*), Parameter         :: fname = 'e04nrfe.opt'
```

```
!     .. Local Scalars ..
      Real (Kind=nag_wp)              :: bndinf, featol, obj, objadd, sinf
      Integer                         :: elmode, i, icol, ifail, iobj, jcol,  &
                                         lenc, m, mode, n, ncolh, ne, ninf,   &
                                         nname, ns
      Logical                         :: verbose_output
      Character (8)                   :: prob
      Character (1)                   :: start
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable :: acol(:), bl(:), bu(:), c(:), pi(:),  &
                                         rc(:), x(:)
      Real (Kind=nag_wp)              :: ruser(1), rw(lenrw)
      Integer, Allocatable            :: helast(:), hs(:), inda(:), loca(:)
      Integer                         :: iuser(1), iw(leniw)
      Character (8)                   :: cuser(1), cw(lencw)
      Character (8), Allocatable      :: names(:)
!     .. Intrinsic Procedures ..
      Intrinsic                       :: max
!     .. Executable Statements ..
      Write (nout,*) 'E04NRF Example Program Results'

!     This program demonstrates the use of routines to set and
!     get values of optional parameters associated with E04NQF.

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

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

      Allocate (inda(ne),loca(n+1),helast(n+m),hs(n+m),acol(ne),bl(n+m),       &
        bu(n+m),x(n+m),pi(m),rc(n+m),names(nname))

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

!     Read the matrix ACOL from data file. Set up LOCA.

      jcol = 1
      loca(jcol) = 1

      Do i = 1, ne

!       Element ( INDA( I ), ICOL ) is stored in ACOL( I ).

        Read (nin,*) acol(i), inda(i), icol

        If (icol<jcol) Then

!         Elements not ordered by increasing column index.

          Write (nout,99999) 'Element in column', icol,                        &
            ' found after element in column', jcol, '. Problem', ' abandoned.'
          Go To 100
        Else If (icol==jcol+1) Then

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

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

!         Index in ACOL 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 LOCA to I.

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

      End Do
```

```
      loca(n+1) = ne + 1

      If (n>icol) Then

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

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

      End If

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

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

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

!     We have no explicit objective vector so set LENC = 0; the
!     objective vector is stored in row IOBJ of ACOL.

      lenc = 0
      Allocate (c(max(1,lenc)))

      objadd = 0.0E0_nag_wp
      prob = ' '

      Write (nout,99998) n, m

!     Call E04NPF to initialize E04NQF.

      ifail = 0
      Call e04npf(cw,lencw,iw,leniw,rw,lenrw,ifail)

!     Set this to .True. to cause e04nqf to produce intermediate
!     progress output
      verbose_output = .False.

      If (verbose_output) Then
!       By default E04NQF does not print monitoring information.
!       Use E04NTF to set the integer-valued option 'Print file'
!       unit number to get information.
        ifail = 0
        Call e04ntf('Print file',nout,cw,iw,rw,ifail)
      End If

!     Open the options file for reading

      mode = 0

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

!     Use E04NRF to read the options file for the remaining
!     options

      ifail = 0
      Call e04nrf(ninopt,cw,iw,rw,ifail)

      Write (nout,*)

!     Use E04NXF to find the value of integer-valued option
!     'Elastic mode'.

      ifail = 0
```

```
      Call e04nxf('Elastic mode',elmode,cw,iw,rw,ifail)

      Write (nout,99997) elmode

!     If Elastic Mode is nonzero, set HELAST.

      If (elmode/=0) Then
        helast(1:(n+m)) = 0
      End If

!     Use E04NUF to set the value of real-valued option
!     'Infinite bound size'.

      bndinf = 1.0E10_nag_wp

      ifail = 0
      Call e04nuf('Infinite bound size',bndinf,cw,iw,rw,ifail)

!     Use E04NYF to find the value of real-valued option
!     'Feasibility tolerance'.

      ifail = 0
      Call e04nyf('Feasibility tolerance',featol,cw,iw,rw,ifail)

      Write (nout,99996) featol

!     Use E04NSF to set the option 'Iterations limit'.

      ifail = 0
      Call e04nsf('Iterations limit 50',cw,iw,rw,ifail)

!     Solve the QP problem.

      ifail = 0
      Call e04nqf(start,qphx,m,n,ne,nname,lenc,ncolh,iobj,objadd,prob,acol,    &
        inda,loca,bl,bu,c,names,helast,hs,x,pi,rc,ns,ninf,sinf,obj,cw,lencw,   &
        iw,leniw,rw,lenrw,cuser,iuser,ruser,ifail)

      Write (nout,*)
      Write (nout,99995) obj
      Write (nout,99994) x(1:n)

100   Continue

99999 Format (1X,A,I5,A,I5,A,A)
99998 Format (1X,/,1X,'QP problem contains ',I3,' variables and ',I3,          &
        ' linear constraints')
99997 Format (1X,'Option ''Elastic mode'' has the value ',I3,'.')
99996 Format (1X,'Option ''Feasibility tolerance'' has the value ',1P,E11.3,   &
        '.')
99995 Format (1X,'Final objective value = ',1P,E11.3)
99994 Format (1X,'Optimal X = ',7F9.2)
    End Program e04nrfe
```

10.2 Program Data

```
Begin example options file
* Comment lines like this begin with an asterisk.
* Switch off output of timing information:
Timing level 0
* Allow elastic variables:
Elastic mode 1
* Set the feasibility tolerance:
Feasibility tolerance 1.0D-4
End
```

```
E04NRF Example Program Data
 7  8                      : Values of N and M
48  8  7  'C'  15          : Values of NNZ, IOBJ, NCOLH, START and NNAME

'...X1...'  '...X2...'  '...X3...'  '...X4...'  '...X5...'
'...X6...'  '...X7...'  '..ROW1..'  '..ROW2..'  '..ROW3..'
'..ROW4..'  '..ROW5..'  '..ROW6..'  '..ROW7..'  '..COST..' : End of array NAMES

    0.02   7   1   : Sparse matrix A, ordered by increasing column index;
    0.02   5   1   : each row contains ACOL(i), INDA(i), ICOL (= column index)
    0.03   3   1   : The row indices may be in any order. In this example
    1.00   1   1   : row 8 defines the linear objective term transpose(C)*X.
    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 lower bounds array 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 upper bounds array BU

 0    0    0    0    0    0    0         : Initial array HS
 0.0  0.0  0.0  0.0  0.0  0.0  0.0       : Initial vector X
```

10.3 Program Results

```
 E04NRF Example Program Results

 QP problem contains   7 variables and   8 linear constraints

 Option 'Elastic mode' has the value   1.
 Option 'Feasibility tolerance' has the value   1.000E-04.

 Final objective value =  -1.848E+06
 Optimal X =      0.00   349.40   648.85   172.85   407.52   271.36   150.02
```