NAG Library Routine Document

E04RYF

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

E04RYF is a part of the NAG optimization modelling suite. It allows you to print information about the problem, stored as a handle, such as which parts have already been defined or details of the matrix constraints.

2 Specification

```
SUBROUTINE E04RYF (HANDLE, NOUT, CMDSTR, IFAIL)
INTEGER      NOUT, IFAIL
CHARACTER(*) CMDSTR
TYPE (C_PTR) HANDLE
```

3 Description

E04RYF prints information on a problem handle which has been previously initialized by E04RAF. Various pieces of information can be retrieved and printed to the given output unit. This can be helpful when the routine is interfaced from interactive environments, for debugging purposes or to help familiarize you with the NAG optimization modelling suite.

The printer is guided by a command string which contains one or more of the following keywords:

Overview
: Gives a brief overview of the problem handle, particularly, which phase it is in, if the problem or optional parameters can be edited and which parts of the problem have already been set. This might be helpful to clarify situations when IFAIL $= 2$ or 3 (the problem cannot be altered anymore, a certain part has already been defined) is obtained from routines, such as, E04REF, E04RFF, E04RHF and E04RJF.

Objective
: Prints the objective function as it was defined by E04REF or E04RFF if it is linear or quadratic. Prints the sparsity structure of the objective function as it was defined by E04RGF if it is nonlinear.

Simple bounds
: Prints the variable bounds as defined by E04RHF. This might help you understand the effect of the optional parameter **Infinite Bound Size** on the bounds.

Linear constraints bounds
Linear constraints detailed
: Print bounds or linear constraint matrix as defined by E04RJF.

Matrix constraints
: Gives a list of the matrix constraints as defined by E04RNF and E04RPF. For each matrix constraint its IDBLK, dimension and order (e.g., linear, bilinear) are printed.

Matrix constraints detailed
: Prints all the matrix constraints including all nonzeros of all the matrices as formulated by E04RNF and E04RPF.

Nonlinear constraints bounds
Nonlinear constraints detailed
: Print bounds or sparsity structure of the nonlinear constraints as defined by E04RKF.

Multipliers sizes
Prints the expected dimensions of array arguments U and UA of the solver E04SVF which store the Lagrangian multipliers for standard and matrix constraints, respectively. This might be helpful in particular in connection with **Overview** and **Matrix constraints** to check the way the sizes of the arrays are derived.

Options
Prints all the current optional parameters. It flags whether the argument is at its default choice, whether you have set it or whether it is chosen by the solver (for example, options left on 'AUTO' setting after the solver has been called).

Note that the output data might not match your input exactly. The sparse matrices are typically transposed, sorted and explicit zeros removed and in certain cases transformed as needed (for example, matrices Q_{ij} and Q_{ji} are merged by E04RPF).

4 References

None.

5 Arguments

1: HANDLE – TYPE (C_PTR) *Input*

On entry: the handle to the problem. It needs to be initialized by E04RAF and **must not** be changed.

2: NOUT – INTEGER *Input*

On entry: the Fortran unit number which identifies the file to be written to.

Constraint: NOUT ≥ 0.

3: CMDSTR – CHARACTER(*) *Input*

On entry: a command string which contains one or more keywords which identify the piece of information about the handle to be printed. Keywords are case-insensitive and space tolerant. Multiple keywords in CMDSTR must be separated by commas or semicolons.

Constraint: CMDSTR can only contain one or more of the following accepted keywords: overview, objective, simple bounds, linear constraints bounds, linear constraints detailed, matrix constraints, matrix constraints detailed, nonlinear constraints bounds, nonlinear constraints detailed, multipliers sizes, options.

4: 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 supplied HANDLE does not define a valid handle to the data structure for the NAG optimization modelling suite. It has not been initialized by E04RAF or it has been corrupted.

IFAIL $= 6$

On entry, NOUT $= \langle value \rangle$.
Constraint: NOUT ≥ 0.

IFAIL $= 7$

CMDSTR does not contain any keywords or is empty.

Keyword number $\langle value \rangle$ is not recognized, it is too long.

Keyword number $\langle value \rangle$ is not recognized.

IFAIL $= 199$

An error occurred when writing to output.

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

E04RYF is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example shows the life-cycle of a handle of a typical (BMI-SDP) problem by printing the overview of the handle in various stages of the problem formulation and after the solution is found. It is also helpful to notice how a linear matrix inequality is extended with the bilinear term, see E04RAF and E04RPF for further details.

The problem is as follows:

$$\begin{array}{ll} \underset{x,y \in \mathbb{R}}{\text{minimize}} & y \\ \text{subject to} & \begin{pmatrix} 1 & x-1 & y \\ x-1 & 3/4 & 0 \\ y & 0 & 16 \end{pmatrix} \succeq 0 \\ & \begin{pmatrix} x & -xy \\ -xy & 1 \end{pmatrix} \succeq 0 \\ & x \geq 0 \\ & -3 \leq y \leq 3 \end{array}$$

The solution is $x = 1/4$, $y = -2$.

Note that the matrix constraints need to be supplied in the form of equation (3) in E04RPF, i.e.,

$$\sum_{i,j=1}^{n} x_i x_j Q_{ij}^k + \sum_{i=1}^{n} x_i A_i^k - A_0^k \succeq 0, \quad k = 1, \ldots, m_A.$$

Therefore the first constraint is defined by matrices

$$A_0^1 = \begin{pmatrix} -1 & 1 & 0 \\ & -3/4 & 0 \\ & & -16 \end{pmatrix}, \quad A_1^1 = \begin{pmatrix} 0 & 1 & 0 \\ & 0 & 0 \\ & & 0 \end{pmatrix}, \quad A_2^1 = \begin{pmatrix} 0 & 0 & 1 \\ & 0 & 0 \\ & & 0 \end{pmatrix}$$

and the second one by

$$A_0^2 = \begin{pmatrix} 0 & 0 \\ & -1 \end{pmatrix}, \quad A_1^2 = \begin{pmatrix} 1 & 0 \\ & 0 \end{pmatrix}, \quad A_2^2 \text{ empty}, \quad Q_{12}^2 = \begin{pmatrix} 0 & -1 \\ 0 & 0 \end{pmatrix}.$$

See also Section 10 in E04RAF for links to further examples in the suite.

10.1 Program Text

```
    Program e04ryfe

!     E04RYF Example Program Text

!     Demonstrate the life-cycle of a handle of a typical BMI-SDP problem
!     by printing the evolution of the HANDLE in certain stages.

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: e04raf, e04ref, e04rhf, e04rnf, e04rpf, e04ryf,   &
                             e04rzf, e04svf, e04zmf, nag_wp
      Use, Intrinsic                   :: iso_c_binding, Only: c_null_ptr,     &
                                          c_ptr
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: maxnnz = 6, maxnvar = 2, nout = 6
!     .. Local Scalars ..
      Type (c_ptr)                     :: h
      Integer                          :: dima, idblk, ifail, inform, nblk,    &
                                          nnzasum, nvar
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: a(maxnnz), rdummy(1), rinfo(32),     &
                                          stats(32), x(maxnvar)
      Integer                          :: icola(maxnnz), idummy(1),            &
```

```
                                        irowa(maxnnz), nnza(maxnvar+1)
!     .. Executable Statements ..
      Continue

      Write (nout,*) 'E04RYF Example Program Results'
      Write (nout,*)
      Flush (nout)

      h = c_null_ptr

!     Start a problem formulation with 2 variables.
      nvar = 2
      ifail = 0
      Call e04raf(h,nvar,ifail)

!     Anything can be defined at this phase.
      Write (nout,*) 'Freshly created handle'
      Flush (nout)
      ifail = 0
      Call e04ryf(h,nout,'Overview',ifail)

!     Define linear objective (min y).
      ifail = 0
      Call e04ref(h,nvar,(/0.0_nag_wp,1.0_nag_wp/),ifail)

!     Add simple bounds (x>=0, -3<=y<=3).
      ifail = 0
      Call e04rhf(h,nvar,(/0.0_nag_wp,-3.0_nag_wp/),                          &
        (/1E20_nag_wp,3.0_nag_wp/),ifail)

!     The simple bounds and the objective are set and cannot be changed.
      Write (nout,*)
      Write (nout,*)                                                          &
        'Handle after definition of simple bounds and the objective'
      Flush (nout)
      ifail = 0
      Call e04ryf(h,nout,'Overview,Objective,Simple Bounds',ifail)

!     Definition of the first (linear) matrix constraint
!        ( 1   x-1   y )
!        (x-1  3/4   0 )  >= 0
!        ( y    0   16 )
!     only upper triangles, thus we have matrices
!           ( 1  -1  0 )        ( 0  1  0 )       ( 0  0  1 )
!     A0 = -(   3/4  0 ),  A1 = (    0  0 ), A2 = (    0  0 )
!           (       16 )        (       0 )       (       0 )
!     Note: don't forget the minus at A0 term!
      dima = 3
      nnzasum = 6
      nblk = 1
!     A0
      irowa(1:4) = (/1,1,2,3/)
      icola(1:4) = (/1,2,2,3/)
      a(1:4) = (/-1.0_nag_wp,1.0_nag_wp,-0.75_nag_wp,-16.0_nag_wp/)
      nnza(1) = 4
!     A1
      irowa(5:5) = (/1/)
      icola(5:5) = (/2/)
      a(5:5) = (/1.0_nag_wp/)
      nnza(2) = 1
!     A2
      irowa(6:6) = (/1/)
      icola(6:6) = (/3/)
      a(6:6) = (/1.0_nag_wp/)
      nnza(3) = 1

      idblk = 0
      ifail = 0
      Call e04rnf(h,nvar,dima,nnza,nnzasum,irowa,icola,a,nblk,idummy,idblk,   &
        ifail)
```

```
!     It is possible to add or extend existing matrix constraints.
      Write (nout,*)
      Write (nout,*) 'Handle after definition of the 1st matrix constraint'
      Flush (nout)
      ifail = 0
      Call e04ryf(h,nout,'Overview,Matrix Constraints',ifail)

!     Definition of the absolute term and linear part of BMI
!        (  x   -xy )
!        (-xy    1  ) >= 0
!     thus
!           ( 0  0 )        ( 1  0 )
!     A0 = -(    1 ),  A1 = (    0 ), A2 = zero
!     Note: don't forget the minus at A0 term!
      dima = 2
      nnzasum = 2
      nblk = 1
!     A0
      irowa(1:1) = (/2/)
      icola(1:1) = (/2/)
      a(1:1) = (/-1.0_nag_wp/)
      nnza(1) = 1
!     A1
      irowa(2:2) = (/1/)
      icola(2:2) = (/1/)
      a(2:2) = (/1.0_nag_wp/)
      nnza(2) = 1
!     A2
      nnza(3) = 0

      idblk = 0
      ifail = 0
      Call e04rnf(h,nvar,dima,nnza,nnzasum,irowa,icola,a,nblk,idummy,idblk,    &
        ifail)

!     It is possible to add or extend existing matrix constraints.
      Write (nout,*)
      Write (nout,*)                                                           &
        'Handle after partial definition of the 2nd matrix constraint'
      Flush (nout)
      ifail = 0
      Call e04ryf(h,nout,'Matrix Constraints',ifail)

!     Extending current matrix constraint (with IDBLK) by bilinear term
!           ( 0  -1 )
!     Q12 = ( 0   0 ).
      dima = 2
      nnzasum = 1
      nnza(1) = 1
      irowa(1:1) = (/1/)
      icola(1:1) = (/2/)
      a(1:1) = (/-1.0_nag_wp/)
      ifail = 0
      Call e04rpf(h,1,(/1/),(/2/),dima,nnza,nnzasum,irowa,icola,a,idblk,ifail)

!     Our problem completely defined.
      Write (nout,*)
      Write (nout,*) 'Handle with the complete problem formulation'
      Flush (nout)
      ifail = 0
      Call e04ryf(h,nout,'Overview,Matrix Constraints,Multipliers Sizes',      &
        ifail)
      ifail = 0
      Call e04ryf(h,nout,'Matrix Constraints Detailed',ifail)

!     Set optional arguments for the solver.
      ifail = 0
      Call e04zmf(h,'Print Options = No',ifail)
      ifail = 0
      Call e04zmf(h,'Initial X = Automatic',ifail)
```

```
!     Options can be printed even outside the solver.
      Write (nout,*)
      Flush (nout)
      ifail = 0
      Call e04ryf(h,nout,'Options',ifail)

!     Call the solver.
      ifail = 0
      Call e04svf(h,nvar,x,0,rdummy,0,rdummy,0,rdummy,rinfo,stats,inform,      &
        ifail)

!     After solver finished.
      Write (nout,*)
      Write (nout,*) 'Problem solved'
      Flush (nout)
      ifail = 0
      Call e04ryf(h,nout,'Overview',ifail)

!     Print result.
      Write (nout,*)
      Write (nout,'(1X,A,F9.2)') 'Final objective value = ', rinfo(1)
      Write (nout,'(1X,A,2F9.2)') 'Final X = ', x(1:nvar)

!     release all memory held in the handle
      ifail = 0
      Call e04rzf(h,ifail)

    End Program e04ryfe
```

10.2 Program Data

None.

10.3 Program Results

```
 E04RYF Example Program Results

 Freshly created handle
 Overview
   Status:                  Problem and option settings are editable.
   No of variables:         2
   Objective function:      not defined yet
   Simple bounds:           not defined yet
   Linear constraints:      not defined yet
   Nonlinear constraints:   not defined yet
   Matrix constraints:      not defined yet

 Handle after definition of simple bounds and the objective
 Overview
   Status:                  Problem and option settings are editable.
   No of variables:         2
   Objective function:      linear
   Simple bounds:           defined
   Linear constraints:      not defined yet
   Nonlinear constraints:   not defined yet
   Matrix constraints:      not defined yet
 Objective function
   linear part
   c(      2) =  1.00E+00,
 Simple bounds
    0.000E+00 <= X_      1
   -3.000E+00 <= X_      2 <=  3.000E+00

 Handle after definition of the 1st matrix constraint
 Overview
   Status:                  Problem and option settings are editable.
   No of variables:         2
   Objective function:      linear
   Simple bounds:           defined
   Linear constraints:      not defined yet
```

```
  Nonlinear constraints:   not defined yet
  Matrix constraints:      1
Matrix constraints
  IDblk =     1, size =      3 x     3, linear

Handle after partial definition of the 2nd matrix constraint
Matrix constraints
  IDblk =     1, size =      3 x     3, linear
  IDblk =     2, size =      2 x     2, linear

Handle with the complete problem formulation
Overview
  Status:                  Problem and option settings are editable.
  No of variables:         2
  Objective function:      linear
  Simple bounds:           defined
  Linear constraints:      not defined yet
  Nonlinear constraints:   not defined yet
  Matrix constraints:      2
Matrix constraints
  IDblk =     1, size =      3 x     3, linear
  IDblk =     2, size =      2 x     2, polynomial of order 2
Lagrangian multipliers sizes
  (Standard) multipliers U: 4 + 0 + 0
  Matrix multipliers UA:    9
Matrix constraints (detailed)
  Matrix inequality IDBLK =     1, dimension     3
    multiindex k =     0
      A_k(     1,     1) = -1.000E+00
      A_k(     2,     1) =  1.000E+00
      A_k(     2,     2) = -7.500E-01
      A_k(     3,     3) = -1.600E+01

    multiindex k =     1
      A_k(     2,     1) =  1.000E+00

    multiindex k =     2
      A_k(     3,     1) =  1.000E+00

  Matrix inequality IDBLK =     2, dimension     2
    multiindex k =     0
      A_k(     2,     2) = -1.000E+00

    multiindex k =     1
      A_k(     1,     1) =  1.000E+00

    multiindex k =     1,     2
      Q_k(     2,     1) = -1.000E+00


Option settings
Begin of Options
    Outer Iteration Limit         =                100     * d
    Inner Iteration Limit         =                100     * d
    Infinite Bound Size           =         1.00000E+20     * d
    Initial X                     =          Automatic      * U
    Initial U                     =          Automatic      * d
    Initial P                     =          Automatic      * d
    Hessian Density               =               Auto      * d
    Init Value P                  =         1.00000E+00     * d
    Init Value Pmat               =         1.00000E+00     * d
    Presolve Block Detect         =                Yes      * d
    Print File                    =                  6      * d
    Print Level                   =                  2      * d
    Print Options                 =                 No      * U
    Monitoring File               =                 -1      * d
    Monitoring Level              =                  4      * d
    Monitor Frequency             =                  0      * d
    Stats Time                    =                 No      * d
    P Min                         =         1.05367E-08     * d
    Pmat Min                      =         1.05367E-08     * d
```

```
    U Update Restriction          =         5.00000E-01     * d
    Umat Update Restriction       =         3.00000E-01     * d
    Preference                    =               Speed     * d
    Transform Constraints         =                Auto     * d
    Dimacs Measures               =               Check     * d
    Stop Criteria                 =                Soft     * d
    Stop Tolerance 1              =         1.00000E-06     * d
    Stop Tolerance 2              =         1.00000E-07     * d
    Stop Tolerance Feasibility    =         1.00000E-07     * d
    Linesearch Mode               =                Auto     * d
    Inner Stop Tolerance          =         1.00000E-02     * d
    Inner Stop Criteria           =           Heuristic     * d
    Task                          =            Minimize     * d
    P Update Speed                =                  12     * d
    Hessian Mode                  =                Auto     * d
    Verify Derivatives            =                Auto     * d
    Time Limit                    =         1.00000E+06     * d
End of Options
E04SV, NLP-SDP Solver (Pennon)
------------------------------
Number of variables             2                 [eliminated            0]
                         simple  linear  nonlin
(Standard) inequalities       3       0       0
(Standard) equalities                 0       0
Matrix inequalities                   1       1 [dense    2, sparse    0]
                                                [max dimension         3]

------------------------------------------------------------------
 it|  objective |  optim  |   feas  |  compl  | pen min |inner
------------------------------------------------------------------
  0  0.00000E+00  4.56E+00  1.23E-01  4.41E+01  1.00E+00   0
  1 -3.01854E-01  1.21E-03  0.00E+00  1.89E+00  1.00E+00   7
  2 -6.21230E-01  2.58E-03  0.00E+00  6.72E-01  4.65E-01   2
  3 -2.11706E+00  4.31E-03  3.39E-02  6.07E-02  2.16E-01   5
  4 -2.01852E+00  5.71E-03  6.05E-03  8.55E-03  1.01E-01   3
  5 -2.00164E+00  3.36E-03  6.26E-04  1.02E-03  4.68E-02   2
  6 -2.00022E+00  4.45E-03  8.37E-05  1.82E-04  2.18E-02   1
  7 -2.00001E+00  4.73E-04  4.01E-06  3.96E-05  1.01E-02   1
  8 -2.00000E+00  4.77E-06  2.25E-07  9.20E-06  4.71E-03   1
  9 -2.00000E+00  4.52E-08  3.61E-08  2.14E-06  2.19E-03   1
 10 -2.00000E+00  6.63E-09  3.19E-08  4.98E-07  1.02E-03   1
 11 -2.00000E+00  8.80E-10  5.34E-09  1.16E-07  4.74E-04   1
 12 -2.00000E+00  1.02E-10  5.41E-09  2.69E-08  2.21E-04   1
------------------------------------------------------------------
Status: converged, an optimal solution found
------------------------------------------------------------------
Final objective value              -2.000000E+00
Relative precision                  9.839057E-10
Optimality                          1.019125E-10
Feasibility                         5.406175E-09
Complementarity                     2.693704E-08
Iteration counts
  Outer iterations                            12
  Inner iterations                            26
  Linesearch steps                            37
Evaluation counts
  Augm. Lagr. values                          50
  Augm. Lagr. gradient                        39
  Augm. Lagr. hessian                         26
------------------------------------------------------------------

Problem solved
Overview
  Status:                 Solver finished, only options can be changed.
  No of variables:        2
  Objective function:     linear
  Simple bounds:          defined
  Linear constraints:     not defined
```

```
   Nonlinear constraints:   not defined
   Matrix constraints:      2

 Final objective value =     -2.00
 Final X =      0.25    -2.00
```