NAG Library Function Document

nag_opt_handle_set_nlnls (e04rmc)

1 Purpose

nag_opt_handle_set_nlnls (e04rmc) is a part of the NAG optimization modelling suite and defines the number of residuals in a sum of squares objective function (nonlinear least squares problems) and, optionally, the sparsity structure of their first derivatives.

2 Specification

```
#include <nag.h>
#include <nage04.h>
void nag_opt_handle_set_nlnls (void *handle, Integer nres, Integer isparse,
     Integer nnzrd, const Integer irowrd[], const Integer icolrd[],
     NagError *fail)
```

3 Description

After the initialization function **nag_opt_handle_init (e04rac)** has been called and unless the objective function has already been defined, **nag_opt_handle_set_nlnls (e04rmc)** may be used to declare the objective function of the optimization problem as a sum of squares. It will typically be used in data fitting or calibration problems of the form

$$\begin{aligned} \underset{x \in R^n}{\text{minimize}} \quad & f(x) = \sum_{j=1}^{m_r} r_j(x)^2 \\ \text{subject to} \quad & l_x \leq x \leq u_x, \end{aligned}$$

where x is an n-dimensional variable vector and $r_i(x)$ are nonlinear residuals (see Section 2.2.3 in the e04 Chapter Introduction). The values of the residuals, and possibly their derivatives, will be communicated to the solver by a user-supplied function. **nag_opt_handle_set_nlnls (e04rmc)** also allows the user to declare the structured first derivative matrix

$$\left[\frac{\partial r_j(x)}{\partial x_i}\right]_{i=1,\ldots,n,\ j=1,\ldots,m_r}$$

as being dense or sparse. If declared as sparse, its sparsity structure must be specified here.

See **nag_opt_handle_init (e04rac)** for more details.

4 References

None.

5 Arguments

1: **handle** – void * *Input*

On entry: the handle to the problem. It needs to be initialized by **nag_opt_handle_init (e04rac)** and **must not** be changed.

2: **nres** – Integer *Input*

On entry: m_r, the number of residuals in the objective function.

If **nres** $= 0$, no objective function will be defined and **irowrd** and **icolrd** will not be referenced and may be **NULL**.

Constraint: **nres** ≥ 0.

3: **isparse** – Integer *Input*

On entry: is a flag indicating if the nonzero structure of the first derivative matrix is dense or sparse.

isparse $= 0$
The first derivative matrix is considered dense and **irowrd** and **icolrd** will not be referenced and may be specified as **NULL**. The ordering is assumed to be column-wise, namely the function will behave as if **nnzrd** $= n \times m_r$ and the vectors **irowrd** and **icolrd** filled as:

$$\textbf{irowrd} = (1, 2, \ldots, n, 1, 2, \ldots, n, \ldots, 1, 2, \ldots, n);$$

$$\textbf{icolrd} = (1, 1, \ldots, 1, 2, 2, \ldots, 2, \ldots, m_r, m_r, \ldots, m_r).$$

isparse $= 1$
The sparsity structure of the first derivative matrix will be supplied by **nnzrd**, **irowrd** and **icolrd**.

Constraint: **isparse** $= 0$ or 1.

4: **nnzrd** – Integer *Input*

On entry: the number of nonzeros in the first derivative matrix.

Constraint: if **nres** > 0, **nnzrd** > 0.

5: **irowrd**[**nnzrd**] – const Integer *Input*
6: **icolrd**[**nnzrd**] – const Integer *Input*

On entry: arrays **irowrd** and **icolrd** store the sparsity structure (pattern) of the first derivative matrix as **nnzrd** nonzeros in coordinate storage (CS) format (see Section 2.1.1 in the f11 Chapter Introduction). The matrix has dimensions $n \times m_r$. **irowrd** specifies one-based row indices and **icolrd** specifies one-based column indices. No particular order of elements is expected, but elements should not repeat and the same order should be used when the first derivative matrix is evaluated for the solver.

Constraints:

$1 \leq \textbf{irowrd}[l-1] \leq n$, for $l = 1, 2, \ldots,$ **nnzrd**;
$1 \leq \textbf{icolrd}[l-1] \leq$ **nres**, for $l = 1, 2, \ldots,$ **nnzrd**.

7: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 3.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_ALREADY_DEFINED

The objective function has already been defined.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_HANDLE

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 **nag_opt_handle_init (e04rac)** or it has been corrupted.

NE_INT

On entry, **isparse** = ⟨*value*⟩.
Constraint: **isparse** = 0 or 1.

On entry, **nnzrd** = ⟨*value*⟩.
Constraint: **nnzrd** > 0.

On entry, **nres** = ⟨*value*⟩.
Constraint: **nres** ≥ 0.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.
See Section 2.7.6 in How to Use the NAG Library and its Documentation for further information.

NE_INVALID_CS

On entry, i = ⟨*value*⟩, **icolrd**$[i-1]$ = ⟨*value*⟩ and **nres** = ⟨*value*⟩.
Constraint: $1 \leq$ **icolrd**$[i-1] \leq$ **nres**.

On entry, i = ⟨*value*⟩, **irowrd**$[i-1]$ = ⟨*value*⟩ and n = ⟨*value*⟩.
Constraint: $1 \leq$ **irowrd**$[i-1] \leq n$.

On entry, more than one element of first derivative matrix has row index ⟨*value*⟩ and column index ⟨*value*⟩.
Constraint: each element of first derivative matrix must have a unique row and column index.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

NE_PHASE

The Hessians of nonlinear functions have already been defined, a nonlinear objective cannot be added.

The problem cannot be modified in this phase any more, the solver has already been called.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_opt_handle_set_nlnls (e04rmc) is not threaded in any implementation.

9 Further Comments

None.

10 Example

In this example, we demonstrate how to declare a least squares problem through **nag_opt_handle_set_nlnls (e04rmc)** and solve it with **nag_opt_handle_solve_dfls (e04ffc)** on a very simple example. Here $n = 2$, $m_r = 3$ and the residuals are computed by:

$$\begin{aligned} r_1(x) &= x(1) + x(2) - 0.9 \\ r_2(x) &= 2x(1) + x(2) - 1.9 \\ r_3(x) &= 3x(1) + x(2) - 3.0 \end{aligned}$$

The expected result is:

$$x = (0.95, 0.10)$$

with an objective value of 0.015.

10.1 Program Text

```
/* nag_opt_handle_set_nlnls (e04rmc) Example Program.
 *
 * Copyright 2017 Numerical Algorithms Group.
 *
 * Mark 26.1, 2017.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nage04.h>
#include <nagx04.h>
#include <assert.h>

#ifdef __cplusplus
extern "C"
{
#endif
static void NAG_CALL objfun(Integer nvar, const double x[],
                            Integer nres, double rx[],
                            Integer *inform, Nag_Comm *comm);
#ifdef __cplusplus
}
#endif

int main(void)
{
  int          nvar, nres, isparse, nnzrd;
  Integer      icolrd[] = {1,2,3,1,2,3}, irowrd[] = {1,2,1,2,1,2};
  double       x[] = { 2.0, 2.0 };
  double       rinfo[100], stats[100];
  double       *rx;
  void         *handle;
  int exit_status = 0;

  /* Nag Types */
  Nag_Comm     comm;
  NagError     fail;

  printf("nag_opt_handle_set_nlnls (e04rmc) Example Program Results\n\n");
  fflush(stdout);

  /* Fill the problem data structure */
  nvar = 2;
  nres = 3;

  /* nag_opt_handle_init (e04rac).
   * Initialize the handle
   */
  nag_opt_handle_init(&handle, nvar, NAGERR_DEFAULT);
```

```
  /* nag_opt_handle_set_nlnls (e04rmc)
   * Define residuals structure,
   */
  isparse = 1;
  nnzrd = 6;
  nag_opt_handle_set_nlnls(handle, nres, isparse, nnzrd, irowrd, icolrd,
                           NAGERR_DEFAULT);

  /* nag_opt_handle_opt_set (e04zmc)
   * Set options
   */
  /* Relax the main convergence criteria a bit */
  nag_opt_handle_opt_set(handle, "DFLS Trust Region Tolerance = 1.0e-3",
                         NAGERR_DEFAULT);
  /* Deactivate the slow convergence detection */
  nag_opt_handle_opt_set(handle, "DFLS Maximum slow steps = 0",
                         NAGERR_DEFAULT);
  /* Turn off option printing */
  nag_opt_handle_opt_set(handle, "Print Options = NO", NAGERR_DEFAULT);
  /* Print the solution */
  nag_opt_handle_opt_set(handle, "Print Solution = YES", NAGERR_DEFAULT);
  /* Deactivate iteration log */
  nag_opt_handle_opt_set(handle, "Print Level = 1", NAGERR_DEFAULT);

  /* nag_opt_handle_solve_dfls (e04ffc)
   * Call the solver
   */
  rx = NAG_ALLOC(nres, double); assert(rx);
  INIT_FAIL(fail);
  nag_opt_handle_solve_dfls(handle, objfun, NULL, nvar, x, nres, rx,
                            rinfo, stats, &comm, &fail);
  if (fail.code != NE_NOERROR){
    printf("Error from nag_opt_handle_solve_dfls (e04ffc).\n%s\n",
           fail.message);
    exit_status = 1;
  }

  /* Clean data */
  if (handle)
    /* nag_opt_handle_free (e04rzc).
     * Destroy the problem handle and deallocate all the memory used
     */
    nag_opt_handle_free(&handle, NAGERR_DEFAULT);
  if (rx)
    NAG_FREE(rx);

  return exit_status;
}

static void NAG_CALL objfun(Integer nvar, const double x[],
                            Integer nres, double rx[],
                            Integer *inform, Nag_Comm *comm)
{
  /* Interrupt the solver if the dimensions does not correspond to the
   * problem
   */
  if (nvar != 2 || nres != 3)
    {
      *inform = -1;
      return;
    }

  rx[0] = x[0] + x[1] - 1.1;
  rx[1] = 2.0*x[0] + x[1] - 1.9;
  rx[2] = 3.0*x[0] + x[1] - 3.0;
}
```

10.2 Program Data

None.

10.3 Program Results

```
nag_opt_handle_set_nlnls (e04rmc) Example Program Results

  ------------------------------------------
  E04FF, Derivative free solver for data fitting
         (nonlinear least-squares problems)
  ------------------------------------------

  Status: Converged, small trust region size.

  Value of the objective                    1.50000E-02
  Computed Solution:
    idx   Lower bound        Value      Upper bound
      1        -inf        9.50000E-01         inf
      2        -inf        1.00000E-01         inf
```