NAG Toolbox

nag_opt_nlp1_option_string (e04ue)

1 Purpose

To supply individual optional parameters to nag_opt_nlp1_solve (e04uc)

nag_opt_nlp1_option_string (e04ue) can also be used to supply individual optional parameters to nag_opt_nlp1_rcomm (e04uf).

2 Syntax

```
[lwsav, iwsav, rwsav, inform] = nag_opt_nlp1_option_string(str, lwsav, iwsav, rwsav)
[lwsav, iwsav, rwsav, inform] = e04ue(str, lwsav, iwsav, rwsav)
```

3 Description

nag_opt_nlp1_option_string (e04ue) may be used to supply values for optional parameters to nag_opt_nlp1_solve (e04uc). It is only necessary to call nag_opt_nlp1_option_string (e04ue) for those arguments whose values are to be different from their default values. One call to nag_opt_nlp1_option_string (e04ue) sets one argument value.

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 double 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.

For nag_opt_nlp1_option_string (e04ue), each user-specified option is normally printed as it is defined, on the current advisory message unit (see nag_file_set_unit_advisory (x04ab)), but this printing may be suppressed using the keyword **Nolist**. Thus the statement

```
[lwsav, iwsav, rwsav, inform] = e04ue('Nolist', lwsav, iwsav, rwsav);
```

suppresses printing of this and subsequent options. Printing will automatically be turned on again after a call to nag_opt_nlp1_solve (e04uc) and may be turned on again at any time using the keyword **List**.

For nag_opt_nlp1_option_string (e04ue) 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 nag_opt_nlp1_solve (e04uc) 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 nag_opt_nlp1_solve (e04uc).

A complete list of optional parameters, their abbreviations, synonyms and default values is given in Section 12 in nag_opt_nlp1_solve (e04uc).

4 References

None.

5 Parameters

5.1 Compulsory Input Parameters

1: **str** – CHARACTER(*)

A single valid option string (as described in Section 3 and in Section 12 in nag_opt_nlp1_solve (e04uc)).

2: **lwsav**(**120**) – LOGICAL array
3: **iwsav**(**610**) – INTEGER array
4: **rwsav**(**475**) – REAL (KIND=nag_wp) array

The arrays **lwsav**, **iwsav** and **rwsav** **must not** be altered between calls to any of the functions nag_opt_nlp1_option_string (e04ue), nag_opt_nlp1_solve (e04uc) and nag_opt_init (e04wb).

5.2 Optional Input Parameters

None.

5.3 Output Parameters

1: **lwsav**(**120**) – LOGICAL array
2: **iwsav**(**610**) – INTEGER array
3: **rwsav**(**475**) – REAL (KIND=nag_wp) array

4: **inform** – INTEGER

Contains zero if a valid option string has been supplied and a value > 0 otherwise (see Section 6).

6 Error Indicators and Warnings

inform $= 5$

The supplied option is invalid. Check that the keywords are neither ambiguous nor misspelt.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

9.1 Program Text

```
function e04ue_example

fprintf('e04ue example results\n\n');

n   = 4;
n_l = 1;
n_n = 2;
a   = [1, 1, 1, 1];
```

```
bl  = [1; 1; 1; 1; -10e+24; -10e+24; 25];
bu  = [5; 5; 5; 5;  20;      40;     10e+24];
istate = zeros(7, 1, nag_int_name);
cjac   = zeros(2, 4);
clamda = zeros(7, 1);
r      = zeros(4, 4);

%Initialize and set option
[cwsav,lwsav,iwsav,rwsav,ifail] = e04wb('e04uc');
[lwsav, iwsav, rwsav, inform] = e04ue(...
                                      'Verify Level = 0', lwsav, iwsav, rwsav);
x = [1; 5; 5; 1];

%Solve
[iter, istate, c, cjac, clamda, objf, objgrd, r, x] = ...
  e04uc(...
        a, bl, bu, @confun, @objfun, istate, cjac, ...
        clamda, r, x, lwsav, iwsav, rwsav);

fprintf('Minimum value     :  %9.4f\n\n',objf);
fprintf('Found after %3d iterations at x:\n   ',iter);
fprintf(' %9.4f',x);
fprintf('\nObjective gradients:\n   ');
fprintf(' %9.4f',objgrd);
fprintf('\nNonlinear contraint values c(x):\n   ');
fprintf(' %9.4f',c);
fprintf('\n');

function [mode, c, cjac, user] = ...
    confun(mode, ncnln, n, ldcj, needc, x, cjac, nstate, user)
  c = zeros(ncnln, 1);

  if (nstate == 1)
   %  first call to confun.  set all jacobian elements to zero.
   %  note that this will only work when 'derivative level = 3'
   %  (the default; see section 11.2).
   cjac=zeros(ldcj, n);
  end

  if (needc(1) > 0)
   if (mode == 0 || mode == 2)
     c(1) = x(1)^2 + x(2)^2 + x(3)^2 + x(4)^2;
   end
   if (mode == 1 || mode == 2)
      cjac(1,1) = 2*x(1);
      cjac(1,2) = 2*x(2);
      cjac(1,3) = 2*x(3);
      cjac(1,4) = 2*x(4);
   end
  end

  if (needc(2) > 0)
   if (mode == 0 || mode == 2)
     c(2) = x(1)*x(2)*x(3)*x(4);
   end
   if (mode == 1 || mode == 2)
      cjac(2,1) = x(2)*x(3)*x(4);
      cjac(2,2) = x(1)*x(3)*x(4);
      cjac(2,3) = x(1)*x(2)*x(4);
      cjac(2,4) = x(1)*x(2)*x(3);
   end
  end

function [mode, objf, objgrd, user] = objfun(mode, n, x, objgrd, nstate, user)

  if (mode == 0 || mode == 2)
    objf = x(1)*x(4)*(x(1)+x(2)+x(3)) + x(3);
  else
    objf = 0;
  end
```

```
  if (mode == 1 || mode == 2)
    objgrd(1) = x(4)*(2*x(1)+x(2)+x(3));
    objgrd(2) = x(1)*x(4);
    objgrd(3) = x(1)*x(4) + 1;
    objgrd(4) = x(1)*(x(1)+x(2)+x(3));
  end
```

9.2 Program Results

```
     e04ue example results

Minimum value     :    17.0140

Found after   5 iterations at x:
       1.0000    4.7430    3.8211    1.3794
Objective gradients:
      14.5723    1.3794    2.3794    9.5641
Nonlinear contraint values c(x):
      40.0000   25.0000
```