NAG Toolbox

nag_opt_nlp1_sparse_option_string (e04uj)

1 Purpose

To supply individual optional parameters to nag_opt_nlp1_sparse_solve (e04ug).

2 Syntax

```
[lwsav, iwsav, rwsav, inform] = nag_opt_nlp1_sparse_option_string(str, lwsav, iwsav, rwsav)
[lwsav, iwsav, rwsav, inform] = e04uj(str, lwsav, iwsav, rwsav)
```

3 Description

nag_opt_nlp1_sparse_option_string (e04uj) may be used to supply values for optional parameters to nag_opt_nlp1_sparse_solve (e04ug). It is only necessary to call nag_opt_nlp1_sparse_option_string (e04uj) for those arguments whose values are to be different from their default values. One call to nag_opt_nlp1_sparse_option_string (e04uj) 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_sparse_option_string (e04uj), 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] = e04uj('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_sparse_solve (e04ug) and may be turned on again at any time using the keyword **List**.

Optional parameter settings are preserved following a call to nag_opt_nlp1_sparse_solve (e04ug) 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_sparse_solve (e04ug).

A complete list of optional parameters, their abbreviations, synonyms and default values is given in Section 12 in nag_opt_nlp1_sparse_solve (e04ug).

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_sparse_solve (e04ug)).

2: **lwsav**(**20**) – LOGICAL array
3: **iwsav**(**550**) – INTEGER array
4: **rwsav**(**550**) – 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_sparse_option_string (e04uj), nag_opt_nlp1_sparse_solve (e04ug) or nag_opt_init (e04wb).

5.2 Optional Input Parameters

None.

5.3 Output Parameters

1: **lwsav**(**20**) – LOGICAL array
2: **iwsav**(**550**) – INTEGER array
3: **rwsav**(**550**) – 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 e04uj_example

fprintf('e04uj example results\n\n');

n     = nag_int(4);
m     = nag_int(6);
ncnln = nag_int(3);
nonln = nag_int(4);
njnln = nag_int(2);
iobj  = nag_int(6);
a  = [1e25; 1e25; 1e25;   1;   -1;
```

```
      1e25; 1e25; 1e25;  -1;    1;
         3;   -1;
        -1;    2];
ha = nag_int([ 1; 2; 3; 5; 4; 1; 2; 3; 5; 4; 6; 1; 2; 6]);
ka = nag_int([ 1;             6;            11;   13;    15]);
bl = [-0.55; -0.55;    0;    0;
                                -894.8; -894.8; -1294.8; -0.55; -0.55; -1e25];
bu = [ 0.55;  0.55; 1200; 1200;
                                -894.8; -894.8; -1294.8;  1e25;  1e25;  1e25];
start = 'C';
names = {''};
ns = nag_int(0);
xs     = [ 0; 0; 0; 0; 0; 0; 0; 0; 0; 0];
istate(1:10) = nag_int(0);
clamda = [ 0; 0; 0; 0; 0; 0; 0; 0; 0; 0];
leniz  = nag_int(1000);
lenz   = nag_int(1000);

[cwsav,lwsav,iwsav,rwsav,ifail] = e04wb('e04ug');
[lwsav,iwsav,rwsav,ifail] = e04uj(...
                              'Print level = 0', lwsav, iwsav, rwsav);

[a, ns, xs, istate, clamda, miniz, minz, ninf, sinf, ...
 obj, user, lwsav, iwsav, rwsav, ifail] = ...
     e04ug(...
           @confun, @objfun, n, m, ncnln, nonln, njnln, ...
           iobj, a, ha, ka, bl, bu, start, names, ns, xs, istate, clamda, ...
           lwsav, iwsav, rwsav, 'lenz', lenz, 'leniz', leniz);

fprintf('\nMinimum found at x: ');
fprintf(' %9.4f',xs(1:n));
fprintf('\nMinimum value     :  %9.4f\n\n',obj);

function [mode, f, fjac, user] = ...
   confun(mode, ncnln, njnln, nnzjac, x, fjac, nstate, user)
  f = zeros(ncnln, 1);

  if (mode == 0 || mode == 2)
    f(1) = 1000*sin(-x(1)-0.25) + 1000*sin(-x(2)-0.25);
    f(2) = 1000*sin(x(1)-0.25) + 1000*sin(x(1)-x(2)-0.25);
    f(3) = 1000*sin(x(2)-x(1)-0.25) + 1000*sin(x(2)-0.25);
  end

  if (mode == 1 || mode == 2)
%   nonlinear jacobian elements for column 1.
    fjac(1) = -1000*cos(-x(1)-0.25);
    fjac(2) = 1000*cos(x(1)-0.25) + 1000*cos(x(1)-x(2)-0.25);
    fjac(3) = -1000*cos(x(2)-x(1)-0.25);
%   nonlinear jacobian elements for column 2.
    fjac(4) = -1000*cos(-x(2)-0.25);
    fjac(5) = -1000*cos(x(1)-x(2)-0.25);
    fjac(6) = 1000*cos(x(2)-x(1)-0.25) + 1000*cos(x(2) -0.25);
  end

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

  if (mode == 0 || mode == 2)
    objf = 1.0e-6*x(3)^3 + 2.0e-6*x(4)^3/3;
  end

  if (mode == 1 || mode == 2)
    objgrd(1) = 0;
    objgrd(2) = 0;
    objgrd(3) = 3.0e-6*x(3)^2;
    objgrd(4) = 2.0e-6*x(4)^2;
  end
```

9.2 Program Results

```
     e04uj example results

Minimum found at x:     0.1189   -0.3962  679.9453 1026.0671
Minimum value     :  5126.4981
```