NAG Library Function Document

nag_opt_read (e04xyc)

1 Purpose

nag_opt_read (e04xyc) reads a set of optional argument values from a file and assigns those values to a given options structure of type Nag_E04_Opt. Values supplied are checked as being of correct type for the specified optional argument.

2 Specification

```
#include <nag.h>
#include <nage04.h>
```

```
void nag_opt_read (const char *name, const char *opt_file,
     Nag_E04_Opt *options, Nag_Boolean print, const char *outfile,
     NagError *fail)
```

3 Description

The optimization functions of Chapter e04 have a number of optional arguments, which are set by means of a structure of type Nag_E04_Opt. Optional argument values may be assigned to members of the options structure directly in the program text and/or by supplying the optional values in a file which can be read by the function nag_opt_read (e04xyc).

When optional argument values are read from a file using nag_opt_read (e04xyc) then the options structure will be initialized automatically if this has not already been done. It is only necessary to call nag_opt_init (e04xxc) if direct assignments to the options structure are made in your program before calling nag_opt_read (e04xyc).

As well as reading from a file, nag_opt_read (e04xyc) will also read from `stdin`. This allows redirection to be used to supply the file; it also permits nag_opt_read (e04xyc) to be used interactively by supplying values from the keyboard.

Checks are made that the values read in are of valid type for the optional argument specified and (except for nag_opt_nlp_sparse (e04ugc)) that the value is within the range for that argument. If a value is accepted, a printed confirmation of the setting of the relevant argument will be output if **print** = Nag_TRUE. An unacceptable argument name or value will give an error message if **fail**.**print** = Nag_TRUE.

4 References

None.

5 Arguments

1: **name** – const char * *Input*

On entry: a character string specifying either the NAG six character name or the NAG long name of the proposed optimization function. The case of the character string is disregarded.

2: **opt_file** – const char * *Input*

On entry: the name of the file which specifies the optional argument values. If `stdin` is to be used, the string `"stdin"` should be supplied. The set of option values must be preceded by the keyword `begin` followed by the function name for which the set of options is being supplied. The function name may be the six character NAG name of an optimization function or its associated long name.

Each option value specified in the file must be preceded by the name of the optional argument. The argument name and value must be separated by at least one blank space or an equals symbol. nag_opt_read (e04xyc) will read to the end of file or until the keyword `end` is found or until another `begin` is found. C style comments may be placed within a set of option values to aid your documentation. Outside the option value sets, text need not be within C style comment delimiters.

Note: assignment to function pointers in the options structure, memory allocation to array pointers and assignment of trailing array dimensions cannot be performed from an options file. These must be assigned directly to the options structure in your calling program.

3: **options** – Nag_E04_Opt * *Input/Output*

On entry: the options structure may or may not have previously been initialized, and had values assigned to its members.

On exit: the options structure, initialized and with values assigned according to the values found in the options file.

4: **print** – Nag_Boolean *Input*

On entry: if Nag_TRUE a message confirming the setting of each option will be output.

5: **outfile** – const char * *Input*

On entry: a character string specifying the name of the file to which confirmation messages should be output. If `stdout` is required then the string `"stdout"` should be given. When **print** = Nag_FALSE the empty string `""` can be supplied as **outfile** will be ignored.

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

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_STOP_LT_START and NE_CHECK_LT_ONE are specific to option setting for nag_opt_conj_grad (e04dgc), nag_opt_nlin_lsq (e04unc) and nag_opt_nlp_solve (e04wdc).

NE_CHECK_LT_ONE

Value $\langle\mathit{value}\rangle$ given to $\langle\mathit{string}\rangle$ is less than 1.

NE_FIELD_UNKNOWN

(line $\langle\mathit{value}\rangle$) '$\langle\mathit{string}\rangle$' is not a permitted structure member or option for $\langle\mathit{string}\rangle$.

NE_INVALID_BEGIN

The Begin statement occurring in data file from which options are being read is not valid.

NE_INVALID_ENUM_RANGE

Enum value $\langle\mathit{string}\rangle$ given to $\langle\mathit{option}\rangle$ is not valid for this function.

NE_INVALID_INT_RANGE_1

Value $\langle\mathit{value}\rangle$ given to $\langle\mathit{option}\rangle$ is not valid. Correct range is $\langle\mathit{option}\rangle\langle\mathit{value}\rangle$.

NE_INVALID_INT_RANGE_2

Value $\langle\mathit{value}\rangle$ given to $\langle\mathit{option}\rangle$ is not valid. Correct range is $\langle\mathit{value}\rangle\langle\mathit{option}\rangle\langle\mathit{value}\rangle$.

NE_INVALID_OPTION

(line ⟨*value*⟩) ⟨*string*⟩ cannot be assigned to using an options file.

NE_INVALID_OPTION_NAME

(line ⟨*value*⟩) '⟨*string*⟩' is not a valid name for a structure member or option.
This error message is output if, for example, the specified string contains characters which are not permitted in a variable name in the C programming language.

NE_INVALID_REAL_RANGE_CONS

Value ⟨*value*⟩ given to ⟨*option*⟩ not valid. The argument ⟨*option*⟩ must satisfy ⟨*constraint*⟩.

NE_INVALID_REAL_RANGE_E

Value ⟨*value*⟩ given to ⟨*option*⟩ is not valid. Correct range is ⟨*option*⟩⟨*value*⟩.

NE_INVALID_REAL_RANGE_EF

Value ⟨*value*⟩ given to ⟨*option*⟩ is not valid. Correct range is ⟨*value*⟩⟨*option*⟩⟨*value*⟩.

NE_INVALID_REAL_RANGE_F

Value ⟨*value*⟩ given to ⟨*option*⟩ is not valid. Correct range is ⟨*option*⟩⟨*value*⟩.

NE_INVALID_REAL_RANGE_FF

Value ⟨*value*⟩ given to ⟨*option*⟩ is not valid. Correct range is ⟨*value*⟩⟨*option*⟩⟨*value*⟩.

NE_INVALID_TEXT_RANGE

Value ⟨*string*⟩ given to ⟨*option*⟩ not valid.

NE_INVALID_VALUE

(line ⟨*value*⟩) is not a permitted structure member or option for ⟨*string*⟩.

NE_NO_VALUE

(line ⟨*value*⟩) no value found for option ⟨*string*⟩.

NE_NOT_APPEND_FILE

Cannot open file ⟨*string*⟩ for appending.

NE_NOT_CLOSE_FILE

Cannot close file ⟨*string*⟩.

NE_NOT_FUN_NAME

The string, ⟨*string*⟩, supplied in the argument name is not the name of any C Library function with option setting facilities.

NE_NOT_READ_FILE

Cannot open file ⟨*string*⟩ for reading.

NE_STOP_LT_START

Value given to obj_check_stop, ⟨*value*⟩, is less than value given to obj_check_start, ⟨*value*⟩.

NE_UNBALANCED_COMMENT

Unbalanced comment starting on line ⟨*value*⟩ found in options file.

NE_WRITE_ERROR

Error occurred when writing to file $\langle string \rangle$.

7 Accuracy

Not applicable.

8 Parallelism and Performance

Not applicable.

9 Further Comments

None.

10 Example

See Section 10 in nag_opt_conj_grad (e04dgc), nag_opt_lsq_no_deriv (e04fcc), nag_opt_lsq_deriv (e04gbc), nag_opt_bounds_2nd_deriv (e04lbc), nag_opt_lp (e04mfc), nag_opt_qp (e04nfc), nag_opt_nlp_sparse (e04ugc) and nag_opt_nlp_solve (e04wdc).