

# NAG Library Function Document

## nag\_opt\_read (e04xyc)

### 1 Purpose

nag\_opt\_read (e04xyc) reads a set of optional parameter 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 parameter.

### 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 parameters, which are set by means of a structure of type Nag\_E04\_Opt. Optional parameter 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 parameter 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 parameter 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 parameter 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 parameter. 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 2.7 in How to Use the NAG Library and its Documentation).

## 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  $\langle value \rangle$  given to  $\langle string \rangle$  is less than 1.

### NE\_FIELD\_UNKNOWN

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

### 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  $\langle string \rangle$  given to  $\langle option \rangle$  is not valid for this function.

### NE\_INVALID\_INT\_RANGE\_1

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

### NE\_INVALID\_INT\_RANGE\_2

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

**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 *<string>*.

**7 Accuracy**

Not applicable.

**8 Parallelism and Performance**

nag\_opt\_read (e04xyc) is not threaded in any implementation.

**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).

---