NAG Library Function Document

nag_opt_sparse_mps_free (e04myc)

1 Purpose

nag_opt_sparse_mps_free (e04myc) frees the memory allocated by nag_opt_sparse_mps_read (e04mzc).

2 Specification

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

```
void nag_opt_sparse_mps_free (double **a, Integer **ha, Integer **ka,
     double **bl, double **bu, double **xs)
```

3 Description

nag_opt_sparse_mps_free (e04myc) should be used in conjunction with nag_opt_sparse_mps_read (e04mzc), which reads data for a sparse linear or quadratic programming problem from an MPSX file, allocates several arrays, and initializes them with the data contained in the file. nag_opt_sparse_mps_free (e04myc) is a utility provided for the convenient freeing of this memory. It should be called in order to conserve memory which is no longer required, e.g., following a call to nag_opt_sparse_convex_qp (e04nkc) (which may be used to solve the problem defined by the MPSX file). Any memory not freed will, of course, be freed when your program terminates.

nag_opt_sparse_mps_free (e04myc) can be used to free a subset of the allocated arrays by passing null pointers for those arguments which you do not wish to free.

4 References

None.

5 Arguments

1: **a** – double ** *Input/Output*

On entry: the nonzeros of the sparse constraint matrix A, to be freed. If **a** or ***a** is a null pointer, no action is taken.

On exit: if **a** is not null, ***a** is set to the null pointer.

2: **ha** – Integer ** *Input/Output*

On entry: the row indices of the nonzero elements stored in **a**, to be freed. If **ha** or ***ha** is a null pointer, no action is taken.

On exit: if **ha** is not null, ***ha** is set to the null pointer.

3: **ka** – Integer ** *Input/Output*

On entry: the indices indicating the beginning of each column of A, to be freed. If **ka** or ***ka** is a null pointer, no action is taken.

On exit: if **ka** is not null, ***ka** is set to the null pointer.

4: **bl** – double ** *Input/Output*

On entry: the lower bounds of the problem variables and general constraints, to be freed. If **bl** or ***bl** is a null pointer, no action is taken.

On exit: if **bl** is not null, ***bl** is set to the null pointer.

5: **bu** – double ** *Input/Output*

On entry: the upper bounds of the problem variables and general constraints, to be freed. If **bu** or ***bu** is a null pointer, no action is taken.

On exit: if **bu** is not null, ***bu** is set to the null pointer.

6: **xs** – double ** *Input/Output*

On entry: a set of initial values for the variables and constraints, to be freed. If **xs** or ***xs** is a null pointer no action is taken.

On exit: if **xs** is not null, ***xs** is set to the null pointer.

6 Error Indicators and Warnings

None.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_opt_sparse_mps_free (e04myc) is not threaded in any implementation.

9 Further Comments

In addition to allocating the memory freed by this function, nag_opt_sparse_mps_read (e04mzc) also allocates memory to the **crnames** member of the **options** structure (if the structure is supplied as an argument). The function nag_opt_free (e04xzc) should be used to free this memory. You must **not** use the standard C function `free()` for this purpose.

10 Example

See Section 10 in nag_opt_sparse_mps_read (e04mzc).