NAG Library Function Document

nag_pack_real_mat_print_comp (x04cdc)

1 Purpose

nag_pack_real_mat_print_comp (x04cdc) prints a double triangular matrix stored in a packed one-dimensional array.

2 Specification

```c
#include <nag.h>
#include <nagx04.h>

void nag_pack_real_mat_print_comp (Nag_OrderType order, Nag_UploType uplo,
   Nag_DiagType diag, Integer n, const double a[], const char *form,
   const char *title, Nag_LabelType labrow, const char *rlabs[],
   Nag_LabelType labcol, const char *clabs[], Integer ncols,
   Integer indent, const char *outfile, NagError *fail)
```

3 Description

nag_pack_real_mat_print_comp (x04cdc) prints a double triangular matrix stored in packed form, using a format specifier supplied by you. The matrix is output to the file specified by `outfile` or, by default, to standard output.

4 References

None.

5 Arguments

1:  **order** – Nag_OrderType  
   *Input*  
   *On entry:* the `order` argument specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by `order = Nag_RowMajor`. See Section 3.2.1.3 in the Essential Introduction for a more detailed explanation of the use of this argument.
   *Constraint:* `order = Nag_RowMajor` or `Nag_ColMajor`.

2:  **uplo** – Nag_UploType  
   *Input*  
   *On entry:* indicates the type of the matrix to be printed  
   `uplo = Nag_Lower`  
   The matrix is lower triangular  
   `uplo = Nag_Upper`  
   The matrix is upper triangular  
   *Constraint:* `uplo = Nag_Lower` or `Nag_Upper`.

3:  **diag** – Nag_DiagType  
   *Input*  
   *On entry:* indicates whether the diagonal elements of the matrix are to be printed.
   `diag = Nag_NonRefDiag`  
   The diagonal elements of the matrix are not referenced and not printed.
The diagonal elements of the matrix are not referenced, but are assumed all to be unity, and are printed as such.

\textbf{diag} = \texttt{Nag\_UnitDiag}

The diagonal elements of the matrix are referenced and printed.

\textbf{Constraint: diag} = \texttt{Nag\_NonRefDiag}, \texttt{Nag\_UnitDiag} or \texttt{Nag\_NonUnitDiag}.

4: \textbf{n} – Integer \hspace{1cm} \textit{Input}

\textit{On entry:} the order of the matrix to be printed.

If \textbf{n} is less than 1, \texttt{nag\_pack\_real\_mat\_print\_comp (x04cdc)} will exit immediately after printing \texttt{title}; no row or column labels are printed.

5: \textbf{a}[\textit{dim}] – const double \hspace{1cm} \textit{Input}

\textit{Note:} the dimension, \textit{dim}, of the array \textbf{a} must be at least \texttt{max(1,n \times (n+1)/2)}.

\textit{On entry:} the matrix to be printed. Note that \textbf{a} must have space for the diagonal elements of the matrix, even if these are not stored.

The storage of elements \( A_{ij} \) depends on the \textbf{order} and \textbf{uplo} arguments as follows:

- If \textbf{order} = \texttt{Nag\_ColMajor} and \textbf{uplo} = \texttt{Nag\_Upper}, \( A_{ij} \) is stored in \( \textbf{a}[(j-1) \times j/2 + i - 1] \), for \( i \leq j \);
- If \textbf{order} = \texttt{Nag\_ColMajor} and \textbf{uplo} = \texttt{Nag\_Lower}, \( A_{ij} \) is stored in \( \textbf{a}[(2n - j) \times (j - 1)/2 + i - 1] \), for \( i \geq j \);
- If \textbf{order} = \texttt{Nag\_RowMajor} and \textbf{uplo} = \texttt{Nag\_Upper}, \( A_{ij} \) is stored in \( \textbf{a}[(2n - i) \times (i - 1)/2 + j - 1] \), for \( i \leq j \);
- If \textbf{order} = \texttt{Nag\_RowMajor} and \textbf{uplo} = \texttt{Nag\_Lower}, \( A_{ij} \) is stored in \( \textbf{a}[(i - 1) \times i/2 + j - 1] \), for \( i \geq j \).

If \textbf{diag} = \texttt{Nag\_UnitDiag}, the diagonal elements of \( A \) are assumed to be 1, and are not referenced; the same storage scheme is used whether \textbf{diag} = \texttt{Nag\_NonUnitDiag} or \textbf{diag} = \texttt{Nag\_UnitDiag}.

6: \textbf{form} – const char * \hspace{1cm} \textit{Input}

\textit{On entry:} a valid C format code. This should be of the form \texttt{\%[flag]ww:pp[format indicator]}, where \textit{ww:pp} indicates that up to two digits may be used to specify the field width and precision respectively. Only \texttt{\%} and \texttt{format indicator} must be present. \textit{flag} can be one of \texttt{-}, \texttt{+}, \texttt{< space >} or \texttt{#} and \texttt{format indicator} can be \texttt{e}, \texttt{E}, \texttt{f}, \texttt{g} or \texttt{G}. Thus, possible formats include \texttt{\%f}, \texttt{\%+23.15G}, \texttt{\%.6e}. \textbf{form} is used to print elements of the matrix \textit{A}.

In addition, \texttt{nag\_pack\_real\_mat\_print\_comp (x04cdc)} chooses its own format code when \textbf{form} is \texttt{NULL} or \textit{form} = \texttt{\'*\}'.

\texttt{form = NULL}

\texttt{nag\_pack\_real\_mat\_print\_comp (x04cdc)} will choose a format code such that numbers will be printed with either \texttt{\%8.4f}, \texttt{\%11.4f} or \texttt{\%13.4e} format. The \texttt{\%8.4f} code is chosen if the sizes of all the matrix elements to be printed lie between 0.001 and 1.0. The \texttt{\%11.4f} code is chosen if the sizes of all the matrix elements to be printed lie between 0.001 and 9999.9999. The \texttt{\%13.4e} code is chosen if the sizes of all the matrix elements to be printed lie between 0.001 and 9999.9999.

\texttt{form = \texttt{\'*\'}}

\texttt{nag\_pack\_real\_mat\_print\_comp (x04cdc)} will choose a format code such that numbers will be printed to as many significant digits as are necessary to distinguish between neighbouring machine numbers. Thus any two numbers that are stored with different internal representations should look different on output.

\textbf{Constraint:} \textbf{form} must be of the form \texttt{\%[flag]ww:pp[format indicator]}.

7: \textbf{title} – const char * \hspace{1cm} \textit{Input}

\textit{On entry:} a title to be printed above the matrix, or name of the matrix.
If \( \text{title} = \text{NULL} \), no title (and no blank line) will be printed.

If \( \text{title} \) contains more than \( \text{ncols} \) characters, the contents of \( \text{title} \) will be wrapped onto more than one line, with the break after \( \text{ncols} \) characters.

Any trailing blank characters in \( \text{title} \) are ignored.

8: \( \text{labrow} \) – Nag_LabelType \hspace{1cm} \text{Input}

\textit{On entry}: indicates the type of labelling to be applied to the rows of the matrix.

\( \text{labrow} = \text{Nag} \_\text{NoLabels} \)
Prints no row labels.

\( \text{labrow} = \text{Nag} \_\text{IntegerLabels} \)
Prints integer row labels.

\( \text{labrow} = \text{Nag} \_\text{CharacterLabels} \)
Prints character labels, which must be supplied in array \( \text{rlabs} \).

\textit{Constraint}: \( \text{labrow} = \text{Nag} \_\text{NoLabels}, \text{Nag} \_\text{IntegerLabels} \) or \( \text{Nag} \_\text{CharacterLabels} \).

9: \( \text{rlabs}[\text{dim}] \) – const char * \hspace{1cm} \text{Input}

\textit{Note}: the dimension, \( \text{dim} \), of the array \( \text{rlabs} \) must be at least

\( n \) when \( \text{labrow} = \text{Nag} \_\text{CharacterLabels} \);
otherwise \( \text{rlabs} \) may be \text{NULL}.

\textit{On entry}: if \( \text{labrow} = \text{Nag} \_\text{CharacterLabels} \), \( \text{rlabs} \) must contain labels for the rows of the matrix; otherwise \( \text{rlabs} \) is not referenced and may be \text{NULL}.

Labels are right-justified when output, in a field which is as wide as necessary to hold the longest row label. Note that this field width is subtracted from the number of usable columns, \( \text{ncols} \).

10: \( \text{labcol} \) – Nag_LabelType \hspace{1cm} \text{Input}

\textit{On entry}: indicates the type of labelling to be applied to the columns of the matrix.

\( \text{labcol} = \text{Nag} \_\text{NoLabels} \)
Prints no column labels.

\( \text{labcol} = \text{Nag} \_\text{IntegerLabels} \)
Prints integer column labels.

\( \text{labcol} = \text{Nag} \_\text{CharacterLabels} \)
Prints character labels, which must be supplied in array \( \text{clabs} \).

\textit{Constraint}: \( \text{labcol} = \text{Nag} \_\text{NoLabels}, \text{Nag} \_\text{IntegerLabels} \) or \( \text{Nag} \_\text{CharacterLabels} \).

11: \( \text{clabs}[\text{dim}] \) – const char * \hspace{1cm} \text{Input}

\textit{Note}: the dimension, \( \text{dim} \), of the array \( \text{clabs} \) must be at least

\( n \) when \( \text{labcol} = \text{Nag} \_\text{CharacterLabels} \);
otherwise \( \text{clabs} \) may be \text{NULL}.

\textit{On entry}: if \( \text{labcol} = \text{Nag} \_\text{CharacterLabels} \), \( \text{clabs} \) must contain labels for the columns of the matrix; otherwise \( \text{clabs} \) is not referenced and may be \text{NULL}.

Labels are right-justified when output. Any label that is too long for the column width, which is determined by \( \text{form} \), is truncated.

12: \( \text{ncols} \) – Integer \hspace{1cm} \text{Input}

\textit{On entry}: the maximum output record length. If the number of columns of the matrix is too large to be accommodated in \( \text{ncols} \) characters, the matrix will be printed in parts, containing the largest possible number of matrix columns, and each part separated by a blank line.
ncols must be large enough to hold at least one column of the matrix using the format specifier in form. If a value less than or equal to 0 or greater than 132 is supplied for ncols, then the value 80 is used instead.

13: indent – Integer

On entry: the number of columns by which the matrix (and any title and labels) should be indented. The effective value of ncols is reduced by indent columns. If a value less than 0 or greater than ncols is supplied for indent, the value 0 is used instead.

14: outfile – const char *

On entry: the name of a file to which output will be directed. If outfile is NULL the output will be directed to standard output.

15: fail – NagError *

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

6 Error Indicators and Warnings

NE_ALLOC_FAIL
Memory allocation failed.

NE_BAD_PARAM
On entry, argument ⟨value⟩ had an illegal value.

NE_COL_WIDTH
⟨value⟩ is not wide enough to hold at least one matrix column. ncols = ⟨value⟩ and indent = ⟨value⟩.

NE_INTERNAL_ERROR
An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

NE_INVALID_FORMAT
The string ⟨value⟩ has not been recognized as a valid format.

NE_NOT_APPEND_FILE
Cannot open file ⟨value⟩ for appending.

NE_NOT_CLOSE_FILE
Cannot close file ⟨value⟩.

NE_NOT_WRITE_FILE
Cannot open file ⟨value⟩ for writing.

7 Accuracy
Not applicable.

8 Parallelism and Performance
Not applicable.
9 Further Comments
None.

10 Example
None.