NAG Library Function Document

nag_pack_complx_mat_print_comp (x04ddc)

1 Purpose

nag_pack_complx_mat_print_comp (x04ddc) prints a Complex triangular matrix stored in a packed one-dimensional array.

2 Specification

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

void nag_pack_complx_mat_print_comp (Nag_OrderType order, Nag_UploType uplo,
    Nag_DiagType diag, Integer n, const Complex a[],
    Nag_ComplexFormType cmplxform, 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_complx_mat_print_comp (x04ddc) prints a Complex 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.

Mark 25
diag = Nag_UnitDiag
   The diagonal elements of the matrix are not referenced, but are assumed all to be unity, and
   are printed as such.

diag = Nag_NonUnitDiag
   The diagonal elements of the matrix are referenced and printed.

Constraint: diag = Nag_NonRefDiag, Nag_UnitDiag or Nag_NonUnitDiag.

4: n – Integer

Input

On entry: the number of rows and columns of the matrix to be printed.

If n is less than 1, nag_pack_complx_mat_print_comp (x04ddc) will exit immediately after
printing title; no row or column labels are printed.

5: a[dim] – const Complex

Input

Note: the dimension, dim, of the array a must be at least max(1, n × (n + 1)/2).

On entry: the matrix to be printed. Note that 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 order and uplo arguments as follows:

- If order = Nag_ColMajor and uplo = Nag_Upper,
  \( A_{ij} \) is stored in \( a[(j - 1) \times j/2 + i - 1] \), for \( i \leq j \);

- if order = Nag_ColMajor and uplo = Nag_Lower,
  \( A_{ij} \) is stored in \( a[(2n - j) \times (j - 1)/2 + i - 1] \), for \( i \geq j \);

- if order = Nag_RowMajor and uplo = Nag_Upper,
  \( A_{ij} \) is stored in \( a[(2n - i) \times (i - 1)/2 + j - 1] \), for \( i \leq j \);

- if order = Nag_RowMajor and uplo = Nag_Lower,
  \( A_{ij} \) is stored in \( a[(i - 1) \times i/2 + j - 1] \), for \( i \geq j \).

If diag = Nag_UnitDiag, the diagonal elements of \( A \) are assumed to be 1, and are not referenced;
the same storage scheme is used whether diag = Nag_NonUnitDiag or diag = Nag_UnitDiag.

6: cmplxform – Nag_ComplexFormType

Input

On entry: indicates how the value of form is to be used to print matrix elements.

- cmplxform = Nag_AboveForm
  The format code in form is assumed to contain a single real edit-descriptor which is to be
  used to print the real and imaginary parts of each Complex number one above the other. Each row
  of the matrix is separated by a blank line, and any row labels are attached only to
  the real parts. This option means that about twice as many columns can be fitted into ncols
  characters than if any other cmplxform option is used. A typical value of form for this
  cmplxform option might be %11.4e.

- cmplxform = Nag_BracketForm
  The format code in form is assumed to contain a single edit-descriptor such as %13.4f, * or
  NULL, which is used to print the real and imaginary parts of each Complex number
  separated by a comma, and surrounded by brackets. Thus a matrix element printed with this
  cmplxform option might look like this: (12.345, −11.323).

- cmplxform = Nag_DirectForm
  The format code in form is used unaltered to print a Complex number. This cmplxform
  option allows you flexibility to specify exactly how the number is printed. With this option
  for cmplxform and a suitable value for form it is possible, for example, to print a Complex
  number in the form \((0.123 + 3.214i)\) or \((0.123e−02, 0.234e−01)\).

Constraint: cmplxform = Nag_AboveForm, Nag_BracketForm or Nag_DirectForm.
7: \textbf{form} \textbf{– 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 \texttt{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. \texttt{flag} can be one of \texttt{-, +, < space >}
or \texttt{#} and \texttt{format indicator} can be e, E, f, g or G. Thus, possible formats include \texttt{\%f}, \texttt{\%-11.4G},
\texttt{\%6e}. \textit{form} is used in conjunction with argument \textit{cmplxform}, to print elements of the matrix \textit{A}.

In addition, \texttt{nag_pack_complx_mat_print_comp (x04ddc)} chooses its own format code when \textit{form}
is \texttt{NULL} or \texttt{form = \textbf{\textquoteleft \textquoteleft \textquoteleft}}.

If \textit{form = NULL}, \texttt{nag_pack_complx_mat_print_comp (x04ddc)} will choose a format code such
that numbers will be printed with either a \texttt{\%8.4f}, a \texttt{\%11.4f} or a \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. Otherwise the \texttt{\%13.4e} code is chosen.

If \textit{form = \textbf{\textquoteleft \textquoteleft \textquoteleft}}, \texttt{nag_pack_complx_mat_print_comp (x04ddc)} 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.

More complicated values of \textit{form}, to print a Complex number in a desired form, may be used. See
the description of argument \textit{cmplxform} above for more details.

\textit{Constraint:} if \textit{cmplxform = Nag_AboveForm} or \textit{Nag_BracketForm}, \textit{form} must be of the form
\texttt{\%[flag]ww.pp[format indicator]}.

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

\textit{On entry:} a title to be printed above the matrix, or name of the matrix.

If \textit{title = NULL}, no title (and no blank line) will be printed.

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

Any trailing blank characters in \textit{title} are ignored.

9: \textbf{labrow} \textbf{– Nag_LabelType} \hspace{1cm} \textit{Input}

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

\texttt{labrow = Nag_NoLabels}
\hspace{0.5cm} Prints no row labels.

\texttt{labrow = Nag_IntegerLabels}
\hspace{0.5cm} Prints integer row labels.

\texttt{labrow = Nag_CharacterLabels}
\hspace{0.5cm} Prints character labels, which must be supplied in array \textit{rlabs}.

\textit{Constraint:} \textit{labrow = Nag_NoLabels, Nag_IntegerLabels} or \textit{Nag_CharacterLabels}.

10: \textbf{rlabs[dim]} \textbf{– const char *} \hspace{1cm} \textit{Input}

\textit{Note:} the dimension, \texttt{dim}, of the array \textit{rlabs} must be at least
\texttt{n} when \textit{labrow = Nag_CharacterLabels};
otherwise \textit{rlabs} may be \texttt{NULL}.

\textit{On entry:} if \textit{labrow = Nag_CharacterLabels}, \textit{rlabs} must contain labels for the rows of the matrix;
otherwise \textit{rlabs} is not referenced and may be \texttt{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, \texttt{ncols}.
labcol – Nag_LabelType  
*Input*

*On entry:* indicates the type of labelling to be applied to the columns of the matrix.

labcol = Nag_NoLabels
Prints no column labels.

labcol = Nag_IntegerLabels
Prints integer column labels.

labcol = Nag_CharacterLabels
Prints character labels, which must be supplied in array clabs.

*Constraint:* labcol = Nag_NoLabels, Nag_IntegerLabels or Nag_CharacterLabels.

clabs[dim] – const char *
*Input*

*Note:* the dimension, dim, of the array clabs must be at least

\[ n \]

when labcol = Nag_CharacterLabels;

otherwise clabs may be NULL.

*On entry:* if labcol = Nag_CharacterLabels, clabs must contain labels for the columns of the matrix; otherwise clabs is not referenced and may be NULL.

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

ncols – Integer  
*Input*

*On entry:* the maximum output record length. If the number of columns of the matrix is too large to be accommodated in 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.

indent – Integer  
*Input*

*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.

outfile – const char *  
*Input*

*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.

fail – NagError *  
*Input/Output*

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

See Section 10 in nag_zpptrf (f07grc).