NAG C Library Function Document

nag_band_complx_mat_print_comp (x04dfc)

1 Purpose

nag_band_complx_mat_print_comp (x04dfc) prints a complex band matrix.

2 Specification

```c
void nag_band_complx_mat_print_comp (Nag_OrderType order, Integer m, Integer n,
        Integer kl, Integer ku, const Complex a[], Integer pda,
        Nag_ComplexFormType cmplxform, const char *format, 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_band_complx_mat_print_comp (x04dfc) prints a complex band matrix stored in packed form, using a format specifier supplied by the user. The matrix is output to the file specified by outfile or, by default, to standard output.

4 References

None.

5 Parameters

1: order – Nag_OrderType
   
   On entry: the order parameter 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 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.

   Constraint: order = Nag_RowMajor or Nag_ColMajor.

2: m – Integer
   
   On entry: the number of rows of the band matrix, respectively, to be printed.

   If either m or n is less than 1, nag_band_complx_mat_print_comp (x04dfc) will exit immediately after printing title; no row or column labels are printed.

3: n – Integer
   
   On entry: the number of columns of the band matrix, respectively, to be printed.

4: kl – Integer
   
   On entry: the number of sub-diagonals of the band matrix A.

   Constraint: kl \geq 0.

5: ku – Integer
   
   On entry: the number of super-diagonals of the band matrix A.

   Constraint: ku \geq 0.
6: \( a[\text{dim}] \) – const Complex

\textbf{Input}

\textbf{Note:} the dimension, \( \text{dim} \), of the array \( a \) must be at least \( \max(1, \text{pda} \times n) \) when \( \text{order} = \text{Nag}_\text{ColMajor} \) and at least \( \max(1, \text{pda} \times m) \) when \( \text{order} = \text{Nag}_\text{RowMajor} \).

On entry: the \( m \) by \( n \) matrix \( A \). This is stored as a notional two-dimensional array with row elements or column elements stored contiguously. The storage of elements \( a_{ij} \), for \( i = 1, \ldots, m \) and \( j = \max(1, i - k), \ldots, \min(n, i + k) \), depends on the \text{order} parameter as follows:

\begin{align*}
\text{if} \; \text{order} = \text{Nag}_\text{ColMajor}, \; a_{ij} \; \text{is stored as} \; a[(j - 1) \times \text{pda} + \text{kl} + \text{ku} + i - j]; \\
\text{if} \; \text{order} = \text{Nag}_\text{RowMajor}, \; a_{ij} \; \text{is stored as} \; a[(i - 1) \times \text{pda} + \text{kl} + j - i].
\end{align*}

7: \( \text{pda} \) – Integer

\textbf{Input}

On entry: the stride separating row or column elements (depending on the value of \text{order}) of the matrix \( A \) in the array \( a \).

\textbf{Constraint:} \( \text{pda} \geq \text{kl} + \text{ku} + 1 \).

8: \( \text{cmplxform} \) – \text{Nag}_\text{ComplexFormType}

\textbf{Input}

On entry: indicates how the value of \text{format} is to be used to print matrix elements.

If \text{cmplxform} = \text{Nag}_\text{AboveForm}, the format code in \text{format} 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 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 \( \text{ncols} \) characters than if any other \text{cmplxform} option is used. A typical value of \text{format} for this \text{cmplxform} option might be \%11.4e.

If \text{cmplxform} = \text{Nag}_\text{BracketForm}, the format code in \text{format} 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 \text{cmplxform} option might look like this: \((12.345, -11.323)\).

If \text{cmplxform} = \text{Nag}_\text{DirectForm}, the format code in \text{format} is used unaltered to print a complex number. This \text{cmplxform} option allows the user flexibility to specify exactly how the number is printed. With this option for \text{cmplxform} and a suitable value for \text{format} it is possible, for example, to print a complex number in the form \((0.123 + 3.214i)\) or \((0.123e-02, 0.234e-01)\).

\textbf{Constraint:} \text{cmplxform} = \text{Nag}_\text{AboveForm}, \text{Nag}_\text{BracketForm} or \text{Nag}_\text{DirectForm}.

9: \( \text{format} \) – char *

\textbf{Input}

\textbf{On entry:} a valid C format code. This should be of the form \%[\text{flag}]ww.pp[\text{format indicator}], where \text{ww.pp} indicates that up to 2 digits may be used to specify the field width and precision respectively. Only \% and \text{format indicator} must be present. \text{flag} can be one of \(-, +, < \text{space} > \) or \# and \text{format indicator} can be e, E, f, g or G. Thus, possible formats include \%f, \%e, \%g and \%G. \text{format} is used in conjunction with parameter \text{cmplxform}, described above, to print elements of the matrix \( A \).

In addition, \text{nag}_\text{band}_\text{complx}_\text{mat}_\text{print}_\text{comp} (x04dfc) chooses its own format code when \text{format} is \text{NULL} or \text{format} = *.

If \text{format} = \text{NULL}, \text{nag}_\text{band}_\text{complx}_\text{mat}_\text{print}_\text{comp} (x04dfc) will choose a format code such that numbers will be printed with either a \%8.4f, a \%11.4f or a \%13.4e format. The \%8.4f code is chosen if the sizes of all the matrix elements to be printed lie between 0.001 and 1.0. The \%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 \%13.4e code is chosen.

If \text{format} = *, \text{nag}_\text{band}_\text{complx}_\text{mat}_\text{print}_\text{comp} (x04dfc) 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 format, to print a complex number in a desired form, may be used. See the description of parameter cmplxform above for more details.

Constraint:

if cmplxform = Nag_AboveForm or Nag_BracketForm, format must be of the form \%[flag]ww.pp[format indicator].

10:  title – char *  
On entry: a title to be printed above the matrix. If title = NULL, no title (and no blank line) will be printed.

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

Any trailing blank characters in title are ignored.

11:  labrow – Nag_LabelType  
On entry: the type of labelling to be applied to the rows of the matrix, as follows:

if labrow = Nag_NoLabels, nag_band_complx_mat_print_comp (x04dfc) prints no row labels;

if labrow = Nag_IntegerLabels, nag_band_complx_mat_print_comp (x04dfc) prints integer row labels;

if labrow = Nag_CharacterLabels, nag_band_complx_mat_print_comp (x04dfc) prints character labels, which must be supplied in array rlabs.

Constraint: labrow = Nag_NoLabels, Nag_IntegerLabels or Nag_CharacterLabels.

12:  rlabs[dim] – const char *  
On entry: if labrow = Nag_CharacterLabels, rlabs must be dimensioned at least of length m and must contain labels for the rows of the matrix, otherwise rlabs should be 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, ncols.

13:  labcol – Nag_LabelType  
On entry: the type of labelling to be applied to the columns of the matrix, as follows:

if labcol = Nag_NoLabels, nag_band_complx_mat_print_comp (x04dfc) prints no column labels;

if labcol = Nag_IntegerLabels, nag_band_complx_mat_print_comp (x04dfc) prints integer column labels;

if labcol = Nag_CharacterLabels, nag_band_complx_mat_print_comp (x04dfc) prints character labels, which must be supplied in array clabs.

Constraint: labcol = Nag_NoLabels, Nag_IntegerLabels or Nag_CharacterLabels.

14:  clabs[dim] – const char *  
On entry: if labcol = Nag_CharacterLabels, clabs must be dimensioned at least of length n and must contain labels for the columns of the matrix, otherwise clabs should be NULL.

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

15:  ncols – Integer  
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 **format**. If a value less than or equal to 0 or greater than 132 is supplied for **ncols**, then the value 80 is used instead.

16:  **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.

17:  **outfile** — 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.

18:  **fail** — NagError *

   *Input/Output*

   The NAG error parameter (see the Essential Introduction).

### 6 Error Indicators and Warnings

**NE_COL_WIDTH**

   \(<value>\) is not wide enough to hold at least one matrix column. **ncols** = \(<value>\), **indent** = \(<value>\).

**NE_INVALID_FORMAT**

   The string \(<value>\), has not been recognised as a valid format.

**NE_ALLOC_FAIL**

   Memory allocation failed.

**NE_BAD_PARAM**

   On entry, parameter \(<value>\) had an illegal value.

**NE_NOT_WRITE_FILE**

   Cannot open file \(<value>\) for writing.

**NE_NOT_APPEND_FILE**

   Cannot open file \(<value>\) for appending.

**NE_NOT_CLOSE_FILE**

   Cannot close file \(<value>\).

**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 consult NAG for assistance.

### 7 Accuracy

Not applicable.

### 8 Further Comments

None.
9 Example

See Section 9 of the document for nag_zgbtrf (f07brc).