NAG C Library Function Document

nag_gen_complx_mat_print (x04dac)

1 Purpose

nag_gen_complx_mat_print (x04dac) is an easy-to-use function to print a complex matrix.

2 Specification

```
void nag_gen_complx_mat_print (Nag_OrderType order, Nag_MatrixType matrix,
     Nag_DiagType diag, Integer m, Integer n, const Complex a[], Integer pda,
     const char *title, const char *outfile, NagError *fail)
```

3 Description

nag_gen_complx_mat_print (x04dac) prints a complex matrix. It is an easy-to-use driver for nag_gen_complx_mat_print_comp (x04dbc). The function uses default values for the format in which numbers are printed, for labelling the rows and columns, and for output record length.

nag_gen_complx_mat_print (x04dac) will choose a format code such that numbers will be printed with 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. The chosen code is used to print each complex element of the matrix with the real part above the imaginary part.

The matrix is printed with integer row and column labels, and with a maximum record length of 80.

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 *Input*

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: **matrix** – Nag_MatrixType *Input*

On entry: indicates the part of the matrix to be printed, as follows:

if **matrix** = **Nag_GeneralMatrix**, the whole of the rectangular matrix;

if **matrix** = **Nag_LowerMatrix**, the lower triangle of the matrix, or the lower trapezium if the matrix has more rows than columns;

if **matrix** = **Nag_UpperMatrix**, the upper triangle of the matrix, or the upper trapezium if the matrix has more columns than rows.

Constraint: **matrix** = **Nag_GeneralMatrix**, **Nag_LowerMatrix** or **Nag_UpperMatrix**.

3: **diag** – Nag_DiagType *Input*

On entry: unless **matrix** = **Nag_GeneralMatrix**, **diag** must specify whether the diagonal elements of the matrix are to be printed, as follows:

if **diag** = **Nag_NonRefDiag**, the diagonal elements of the matrix are not referenced and not printed;

if **diag** = **Nag_UnitDiag**, the diagonal elements of the matrix are not referenced, but are assumed all to be unity, and are printed as such;

if **diag** = **Nag_NonUnitDiag**, the diagonal elements of the matrix are referenced and printed.

If **matrix** = **Nag_GeneralMatrix**, then **diag** must be set to **Nag_NonUnitDiag**.

Constraints:

if **matrix** $\neq$ **Nag_GeneralMatrix**, **diag** = **Nag_NonRefDiag**, **Nag_UnitDiag** or **Nag_NonUnitDiag**;
if **matrix** = **Nag_GeneralMatrix**, **diag** = **Nag_NonUnitDiag**.

4: **m** – Integer *Input*
5: **n** – Integer *Input*

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

If either **m** or **n** is less than 1, nag_gen_complx_mat_print (x04dac) will exit immediately after printing **title**; no row or column labels are printed.

6: **a**[dim] – const Complex *Input*

Note: the dimension, dim, of the array **a** must be at least $\max(1, \textbf{pda} \times \textbf{n})$ when **order** = **Nag_ColMajor** and at least $\max(1, \textbf{pda} \times \textbf{m})$ when **order** = **Nag_RowMajor**.

If **order** = **Nag_ColMajor**, the (i, j)th element of the matrix A is stored in **a**$[(j-1) \times \textbf{pda} + i - 1]$ and if **order** = **Nag_RowMajor**, the (i, j)th element of the matrix A is stored in **a**$[(i-1) \times \textbf{pda} + j - 1]$.

On entry: the matrix to be printed. Only the elements that will be referred to, as specified by parameters **matrix** and **diag**, need be set.

7: **pda** – Integer *Input*

On entry: the stride separating matrix row or column elements (depending on the value of **order**) in the array **a**.

Constraints:

if **order** = **Nag_ColMajor**, **pda** $\geq \max(1, \textbf{m})$;
if **order** = **Nag_RowMajor**, **pda** $\geq \max(1, \textbf{n})$.

8: **title** – char * *Input*

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 80 characters, the contents of **title** will be wrapped onto more than one line, with the break after 80 characters.

Any trailing blank characters in **title** are ignored.

9: **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.

10: **fail** – NagError * *Input/Output*

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

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

A call to nag_gen_complx_mat_print (x04dac) is equivalent to a call to nag_gen_complx_mat_print_comp (x04dbc) with the following argument values:

```
ncols = 80
indent = 0
labrow = Nag_IntegerLabels
labcol = Nag_IntegerLabels
form = 0
cmplxform = Nag_AboveForm
```

9 Example

None.