

# NAG Library Function Document

## nag\_gen\_complex\_mat\_print (x04dac)

### 1 Purpose

nag\_gen\_complex\_mat\_print (x04dac) is an easy-to-use function to print a Complex matrix.

### 2 Specification

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

void nag_gen_complex_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\_complex\_mat\_print (x04dac) prints a Complex matrix. It is an easy-to-use driver for nag\_gen\_complex\_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\_complex\_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 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: **matrix** – Nag\_MatrixType *Input*  
*On entry:* indicates the part of the matrix to be printed.  
**matrix** = Nag\_GeneralMatrix  
 The whole of the rectangular matrix.  
**matrix** = Nag\_LowerMatrix  
 The lower triangle of the matrix, or the lower trapezium if the matrix has more rows than columns.

**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:* 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.

**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.

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\_complex\_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

$\max(1, \mathbf{pda} \times \mathbf{n})$  when **order** = Nag\_ColMajor;

$\max(1, \mathbf{m} \times \mathbf{pda})$  when **order** = Nag\_RowMajor.

The (*i*, *j*)th element of the matrix *A* is stored in

**a**[(*j* – 1)  $\times$  **pda** + *i* – 1] when **order** = Nag\_ColMajor;

**a**[(*i* – 1)  $\times$  **pda** + *j* – 1] when **order** = Nag\_RowMajor.

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

7: **pda** – Integer *Input*

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

*Constraints:*

if **order** = Nag\_ColMajor, **pda**  $\geq$   $\max(1, \mathbf{m})$ ;

if **order** = Nag\_RowMajor, **pda**  $\geq$   $\max(1, \mathbf{n})$ .

8: **title** – const 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** – 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.

10: **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  $\langle value \rangle$  had an illegal 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\_NOT\_APPEND\_FILE

Cannot open file  $\langle value \rangle$  for appending.

### NE\_NOT\_CLOSE\_FILE

Cannot close file  $\langle value \rangle$ .

### NE\_NOT\_WRITE\_FILE

Cannot open file  $\langle value \rangle$  for writing.

## 7 Accuracy

Not applicable.

## 8 Parallelism and Performance

Not applicable.

## 9 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
```

## 10 Example

None.

---