NAG Fortran Library Routine Document

X04DDF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

X04DDF prints a ***complex*16*** triangular matrix stored in a packed one-dimensional array.

2 Specification

```
SUBROUTINE X04DDF (UPLO, DIAG, N, A, USEFRM, FORMAT, TITLE, LABROW,
1                  RLABS, LABCOL, CLABS, NCOLS, INDENT, IFAIL)

INTEGER            N, NCOLS, INDENT, IFAIL
complex*16         A(*)
CHARACTER*(*)      FORMAT, TITLE, RLABS(*), CLABS(*)
CHARACTER*1        UPLO, DIAG, USEFRM, LABROW, LABCOL
```

3 Description

X04DDF prints a ***complex*16*** triangular matrix stored in packed form, using a format specifier supplied by you. The matrix must be packed by column. The matrix is output to the unit defined by X04ABF.

4 References

None.

5 Parameters

1: UPLO – CHARACTER*1 *Input*

On entry: indicates the type of the matrix to be printed

UPLO = 'L' (Lower)

The matrix is lower triangular. In this case, the packed array A holds the matrix elements in the following order: $(1,1),(2,1),\ldots,(\mathrm{N},1),(2,2),(3,2),\ldots,(\mathrm{N},2),\cdot$

UPLO = 'U' (Upper)

The matrix is upper triangular. In this case, the packed array A holds the matrix elements in the following order: $(1,1),(1,2),(2,2),(1,3),(2,3),(3,3),(1,4),\cdot$

Constraint: UPLO = 'L' or 'U'.

2: DIAG – CHARACTER*1 *Input*

On entry: indicates whether the diagonal elements of the matrix are to be printed.

DIAG = 'B' (Blank)

The diagonal elements of the matrix are not referenced and not printed.

DIAG = 'U' (Unit diagonal)

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

DIAG = 'N' (Non-unit diagonal)

The diagonal elements of the matrix are referenced and printed.

Constraint: DIAG = 'B', 'U' or 'N'.

3: N – INTEGER *Input*

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

If N is less than 1, X04DDF will exit immediately after printing TITLE; no row or column labels are printed.

4: A(∗) – ***complex*16*** array *Input*

Note: the dimension of the array A must be at least $\max(1, \mathrm{N} \times (\mathrm{N}+1)/2)$.

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.

5: USEFRM – CHARACTER*1 *Input*

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

USEFRM = 'A' (Above)

The format code in 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*16*** 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 USEFRM option is used. A typical value of FORMAT for this USEFRM option might be 'E13.4', '*' or ' '.

USEFRM = 'B' (Bracketed)

The format code in FORMAT is assumed to contain a single edit-descriptor such as 'E13.4', '*' or ' ' which is used to print the real and imaginary parts of each ***complex*16*** number separated by a comma, and surrounded by brackets. Thus a matrix element printed with this USEFRM option might look like this: $(12.345, -11.323)$.

USEFRM = 'D' (Direct)

The format code in FORMAT is used unaltered to print a ***complex*16*** number. This USEFRM option allows you flexibility to specify exactly how the number is printed. With this option for USEFRM and a suitable value for FORMAT it is possible, for example, to print a ***complex*16*** number in the form $(0.123 + 3.214i)$ or $(0.123\mathrm{D}-02, 0.234\mathrm{D}-01)$. See Section 9 for an example illustrating this option.

Constraint: USEFRM = 'A', 'B' or 'D'.

6: FORMAT – CHARACTER*(*) *Input*

On entry: a valid Fortran format code. This may be any format code allowed on the system, whether it is standard Fortran or not. FORMAT is used in conjunction with parameter USEFRM, to print elements of the matrix *A*. It may or may not be enclosed in brackets. Examples of valid values for FORMAT are '(F11.4)', '1P,2E13.5'.

In addition, there are two special codes which force X04DDF to choose its own format code:

FORMAT = ' '

X04DDF will choose a format code such that numbers will be printed with an F8.4, an F11.4 or a 1PE13.4 format. The F8.4 code is chosen if the sizes of the real and imaginary parts of all the matrix elements to be printed lie between 0.001 and 1.0. The F11.4 code is chosen if the sizes of all the numbers to be printed lie between 0.001 and 9999.9999. Otherwise the 1PE13.4 code is chosen.

FORMAT = ∗

X04DDF 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. Whether they do in fact look different will depend on the run-time library of the Fortran compiler in use.

More complicated values of FORMAT, to print a ***complex*16*** number in a desired form, may be used. See the description of parameter USEFRM for more details.

Constraint: the character length of FORMAT must be ≤ 80.

7: TITLE – CHARACTER*(*) *Input*

On entry: a title to be printed above the matrix.

If TITLE = ' ', 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.

8: LABROW – CHARACTER*1 *Input*

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

LABROW = 'N'

Prints no row labels.

LABROW = 'I'

Prints integer row labels.

LABROW = 'C'

Prints character labels, which must be supplied in array RLABS.

Constraint: LABROW = 'N', 'I' or 'C'.

9: RLABS(∗) – CHARACTER*(*) array *Input*

Note: the dimension of the array RLABS must be at least N if LABROW = 'C' and at least 1 otherwise.

On entry: if LABROW = 'C', RLABS must be dimensioned at least of length N and must contain labels for the rows of the matrix, otherwise RLABS may be dimensioned of length 1.

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.

10: LABCOL – CHARACTER*1 *Input*

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

LABCOL = 'N'

Prints no column labels.

LABCOL = 'I'

Prints integer column labels.

LABCOL = 'C'

Prints character labels, which must be supplied in array CLABS.

Constraint: LABCOL = 'N', 'I' or 'C'.

11: CLABS(∗) – CHARACTER*(*) array *Input*

Note: the dimension of the array CLABS must be at least N if LABCOL = 'C' and at least 1 otherwise.

On entry: if LABCOL = 'C', must contain labels for the columns of the matrix.

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

12: 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 FORMAT. If a value less than 0 or greater than 132 is supplied for NCOLS, then the value 80 is used instead.

13: 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.

14: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. If you are unfamiliar with this parameter you should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

For environments where it might be inappropriate to halt program execution when an error is detected, the value −1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter the recommended value is 0. **When the value −1 or 1 is used it is essential to test the value of IFAIL on exit.**

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

On entry, UPLO ≠ 'L' or 'U'.

IFAIL = 2

On entry, DIAG ≠ 'N', 'U' or 'B'.

IFAIL = 3

On entry, USEFRM ≠ 'A', 'B' or 'D'.

IFAIL = 4

On entry, variable FORMAT is more than 80 characters long.

IFAIL = 5

The code supplied in FORMAT cannot be used to output a number. FORMAT probably has too wide a field width or contains an illegal edit descriptor.

IFAIL = 6

On entry, either LABROW or LABCOL $\neq$ 'N', 'I' or 'C'.

IFAIL = 7

The quantity NCOLS − INDENT − *labwid* (where *labwid* is the width needed for the row labels) is not large enough to hold at least one column of the matrix.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

This example program calls X04DDF twice, first to print a 4 by 4 lower triangular matrix, and then to print a 4 by 4 triangular matrix, various options for labelling and formatting are illustrated.

9.1 Program Text

```
*     X04DDF Example Program Text
*     Mark 14 Release. NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          NMAX, LA
      PARAMETER        (NMAX=4,LA=(NMAX*(NMAX+1))/2)
*     .. Local Scalars ..
      DOUBLE PRECISION AA
      INTEGER          I, IFAIL, INDENT, NCOLS
      CHARACTER*19     FORMAT
*     .. Local Arrays ..
      COMPLEX *16      A(LA)
      CHARACTER*7      CLABS(NMAX), RLABS(NMAX)
*     .. External Subroutines ..
      EXTERNAL         X04DDF
*     .. Intrinsic Functions ..
      INTRINSIC        CMPLX
*     .. Data statements ..
      DATA             CLABS/'Un', 'Deux', 'Trois', 'Quatre'/
      DATA             RLABS/'Uno', 'Duo', 'Tre', 'Quattro'/
*     .. Executable Statements ..
      WRITE (NOUT,*) 'X04DDF Example Program Results'
      WRITE (NOUT,*)
*
*     Generate an array of data
      DO 20 I = 1, LA
         AA = I
         A(I) = CMPLX(AA,-AA,KIND=KIND(AA))
   20 CONTINUE
*
      NCOLS = 80
      INDENT = 0
      IFAIL = 0
      FORMAT = ' '
*
*     Print order 4 lower triangular matrix with default format and
*     integer row and column labels, and bracketed complex elements
      CALL X04DDF('Lower','Non-unit',4,A,'Bracketed',FORMAT,
     +            'Example 1:','Integer',RLABS,'Integer',CLABS,NCOLS,
     +            INDENT,IFAIL)
*
```

```
      WRITE (NOUT,*)
      FORMAT = 'SS,F7.1,SP,F6.1,''i'''
*
*     Print order 4 upper triangular matrix with user-supplied format
*     and row and column labels, using the supplied format directly
      CALL X04DDF('Upper','Unit',4,A,'Direct',FORMAT,'Example 2:',
     +            'Character',RLABS,'Character',CLABS,NCOLS,INDENT,
     +            IFAIL)
*
      STOP
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 X04DDF Example Program Results

 Example 1:
                             1                          2
 1  (     1.0000,    -1.0000)
 2  (     2.0000,    -2.0000) (     5.0000,    -5.0000)
 3  (     3.0000,    -3.0000) (     6.0000,    -6.0000)
 4  (     4.0000,    -4.0000) (     7.0000,    -7.0000)

                             3                          4
 1
 2
 3  (     8.0000,    -8.0000)
 4  (     9.0000,    -9.0000) (    10.0000,   -10.0000)

 Example 2:
                    Un           Deux          Trois         Quatre
     Uno     1.0  +0.0i    2.0  -2.0i    4.0  -4.0i    7.0  -7.0i
     Duo                  1.0  +0.0i    5.0  -5.0i    8.0  -8.0i
     Tre                               1.0  +0.0i    9.0  -9.0i
 Quattro                                           1.0  +0.0i
```