NAG Library Routine Document

X04CFF

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

X04CFF prints a real band matrix stored in a packed two-dimensional array.

2 Specification

```
SUBROUTINE X04CFF (M, N, KL, KU, A, LDA, FORM, TITLE, LABROW, RLABS,       &
                   LABCOL, CLABS, NCOLS, INDENT, IFAIL)

INTEGER             M, N, KL, KU, LDA, NCOLS, INDENT, IFAIL
REAL (KIND=nag_wp) A(LDA,*)
CHARACTER(*)        FORM, TITLE, RLABS(*), CLABS(*)
CHARACTER(1)        LABROW, LABCOL
```

3 Description

X04CFF prints a real band matrix stored in a packed two-dimensional array, using a format specifier supplied by you. The matrix is output to the unit defined by X04ABF.

4 References

None.

5 Arguments

1: M – INTEGER *Input*

2: N – INTEGER *Input*

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

If either M or N is less than 1, X04CFF will exit immediately after printing TITLE; no row or column labels are printed.

3: KL – INTEGER *Input*

On entry: the number of subdiagonals of the band matrix A.

Constraint: KL ≥ 0.

4: KU – INTEGER *Input*

On entry: the number of superdiagonals of the band matrix A.

Constraint: KU ≥ 0.

5: A(LDA, $*$) – REAL (KIND=nag_wp) array *Input*

Note: the second dimension of the array A must be at least $\max(1, \min(\mathrm{M}+\mathrm{KU}, \mathrm{N}))$.

On entry: the band matrix to be printed.

The matrix is stored in rows 1 to $k_l + k_u + 1$, more precisely, the element A_{ij} must be stored in

$$\mathrm{A}(k_u + 1 + i - j, j) \quad \text{for } \max(1, j - k_u) \leq i \leq \min(m, j + k_l).$$

6: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which X04CFF is called.

Constraint: LDA $\geq$ KL + KU + 1.

7: FORM – CHARACTER(*) *Input*

On entry: describes the Fortran format code for printing the elements of the matrix A. The format code may be any allowed on the system, whether it is standard Fortran or not. It may or may not be enclosed in brackets.

In addition, there are the following special codes which force X04CFF to choose its own format code:

FORM = ' '
: X04CFF 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 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 matrix elements to be printed lie between 0.001 and 9999.9999. Otherwise the 1PE13.4 code is chosen.

FORM = '*'
: X04CFF 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.

By preceding the desired format code by the string 'MATLAB', X04CFF will print the matrix such that it can be input into MATLAB, and TITLE will be used as the name of the matrix.

Examples of valid values for FORM are 'F11.4', '1PE13.5', 'G14.5', 'MATLABF11.4', 'MATLAB*'.

Constraint: the character length of the format specifier in FORM must be ≤ 80.

8: TITLE – CHARACTER(*) *Input*

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

If printing in MATLAB mode, TITLE will be used as the name of the matrix.

9: LABROW – CHARACTER(1) *Input*

On entry: indicates the type of labelling to be applied to the rows of the matrix, except in MATLAB mode where LABROW is ignored.

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'.

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

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

On entry: if LABROW = 'C', RLABS must contain labels for the rows of the matrix, except in MATLAB mode where RLABS is ignored.

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.

11: LABCOL – CHARACTER(1) *Input*

On entry: indicates the type of labelling to be applied to the columns of the matrix, except in MATLAB mode where LABCOL is ignored.

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'.

12: 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', CLABS must contain labels for the columns of the matrix, except in MATLAB mode where CLABS is ignored.

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

13: 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 0 or greater than 132 is supplied for NCOLS, then the value 80 is used instead.

14: 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.

15: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

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 argument, the recommended value is 0. **When the value −1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

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, KL < 0.

IFAIL $= 2$

On entry, KU < 0.

IFAIL $= 3$

On entry, LDA $<$ KL $+$ KU $+ 1$.

IFAIL $= 4$

On entry, the format specifier in FORM is more than 80 characters long.

IFAIL $= 5$

The format specifier in FORM cannot be used to output a number. The specifier 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.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

X04CFF is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example calls X04CFF three times, to print 5 by 5 matrices of different bandwidths; various options for labelling and formatting are illustrated.

10.1 Program Text

```
    Program x04cffe

!     X04CFF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: f01zcf, nag_wp, x04cff
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: n = 5, nout = 6
      Character (7), Parameter         :: clabs(n) = (/'Un     ','Deux   ',    &
                                          'Trois  ','Quatre ','Cinq   '/)
      Character (7), Parameter         :: rlabs(n) = (/'Uno    ','Due    ',    &
                                          'Tre    ','Quattro','Cinque '/)
!     .. Local Scalars ..
      Integer                          :: i, ifail, indent, j, kl, ku, ku_a,   &
                                          lda, ldab, m, ncols
      Character (1)                    :: job
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:), ab(:,:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, min, real
!     .. Executable Statements ..
      Write (nout,*) 'X04CFF Example Program Results'

      Write (nout,*)
      Flush (nout)

!     Generate a square array of data.

      m = n
      kl = 1
      ku_a = 2

      lda = m
      Allocate (a(lda,n))

      Do j = 1, n

        Do i = max(1,j-ku_a), min(m,j+kl)
          a(i,j) = real(10*i+j,kind=nag_wp)
        End Do

      End Do

!     Convert a to packed storage, ignoring the second superdiagonal.

      ldab = kl + ku_a + 1
      Allocate (ab(ldab,n))

      ku = 1

      job = 'P'

      ifail = 0
```

```
      Call f01zcf(job,m,n,kl,ku,a,lda,ab,ldab,ifail)

      ncols = 80
      indent = 0

!     Print m by n band matrix with kl subdiagonals, 1 superdiagonal,
!     default format and integer row and column labels

      ifail = 0
      Call x04cff(m,n,kl,ku,ab,ldab,' ','Example 1:','Integer',rlabs,          &
        'Integer',clabs,ncols,indent,ifail)

      Write (nout,*)
      Flush (nout)

!     Convert the whole matrix a to packed storage.

      ku = ku_a

      job = 'P'

      ifail = 0
      Call f01zcf(job,m,n,kl,ku,a,lda,ab,ldab,ifail)

!     Print m by n band matrix with kl subdiagonals, ku superdiagonals,
!     user-supplied format and row and column labels

      ifail = 0
      Call x04cff(m,n,kl,ku,ab,ldab,'F8.2','Example 2:','Character',rlabs,     &
        'Character',clabs,ncols,indent,ifail)

      Write (nout,*)
      Flush (nout)

!     Print m by n band matrix with kl subdiagonals, ku superdiagonals,
!     in MATLAB format
!     Row and column labelling is ignored

      ifail = 0
      Call x04cff(m,n,kl,ku,ab,ldab,'MATLABF8.2','A',' ',rlabs,' ',clabs,      &
        ncols,indent,ifail)

    End Program x04cffe
```

10.2 Program Data

None.

10.3 Program Results

```
 X04CFF Example Program Results

 Example 1:
            1          2          3          4          5
1     11.0000    12.0000
2     21.0000    22.0000    23.0000
3                32.0000    33.0000    34.0000
4                           43.0000    44.0000    45.0000
5                                      54.0000    55.0000

 Example 2:
              Un    Deux   Trois  Quatre    Cinq
    Uno    11.00   12.00   13.00
    Due    21.00   22.00   23.00   24.00
    Tre            32.00   33.00   34.00   35.00
Quattro                    43.00   44.00   45.00
 Cinque                            54.00   55.00

 A = [
    11.00   12.00   13.00    0.00    0.00;
```

```
     21.00   22.00   23.00   24.00    0.00;
      0.00   32.00   33.00   34.00   35.00;
      0.00    0.00   43.00   44.00   45.00;
      0.00    0.00    0.00   54.00   55.00;
     ];
```