NAG Library Routine Document

F01CWF

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

F01CWF adds two complex matrices, each one optionally transposed and multiplied by a scalar.

2 Specification

```
SUBROUTINE F01CWF(TRANSA, TRANSB, M, N, ALPHA, A, LDA, BETA, B, LDB, C,
1                 LDC, IFAIL)
INTEGER           M, N, LDA, LDB, LDC, IFAIL
complex*16        ALPHA, A(LDA,*), BETA, B(LDB,*), C(LDC,*)
CHARACTER*1       TRANSA, TRANSB
```

3 Description

F01CWF performs one of the operations

$C := \alpha A + \beta B$,

$C := \alpha A^{\mathrm{T}} + \beta B$,

$C := \alpha A^{\mathrm{H}} + \beta B$,

$C := \alpha A + \beta B^{\mathrm{T}}$,

$C := \alpha A^{\mathrm{T}} + \beta B^{\mathrm{T}}$,

$C := \alpha A^{\mathrm{H}} + \beta B^{\mathrm{T}}$,

$C := \alpha A + \beta B^{\mathrm{H}}$,

$C := \alpha A^{\mathrm{T}} + \beta B^{\mathrm{H}}$ or

$C := \alpha A^{\mathrm{H}} + \beta B^{\mathrm{H}}$,

where A, B and C are matrices, α and β are scalars, T denotes transposition and H denotes conjugate transposition. For efficiency, the routine contains special code for the cases when one or both of α, β is equal to zero, unity or minus unity. The matrices, or their transposes, must be compatible for addition. A and B are either m by n or n by m matrices, depending on whether they are to be transposed before addition. C is an m by n matrix.

4 References

None.

5 Parameters

1: TRANSA – CHARACTER*1 *Input*

2: TRANSB – CHARACTER*1 *Input*

On entry: TRANSA and TRANSB must specify whether or not the matrix A and the matrix B, respectively, are to be transposed before addition.

TRANSA or TRANSB = 'N'
The matrix will not be transposed.

TRANSA or TRANSB = 'T'
The matrix will be transposed.

TRANSA or TRANSB = 'C'
The matrix will be transposed and conjugated.

Constraint: TRANSA and TRANSB must be one of 'N', 'T' or 'C'.

3: M – INTEGER *Input*

On entry: m, the number of rows of the matrices A and B or their transposes. Also the number of rows of the matrix C.

Constraint: M ≥ 0.

4: N – INTEGER *Input*

On entry: n, the number of columns of the matrices A and B or their transposes. Also the number of columns of the matrix C.

Constraint: N ≥ 0.

5: ALPHA – ***complex*16*** *Input*

On entry: the scalar α, by which matrix A is multiplied before addition.

6: A(LDA,$*$) – ***complex*16*** array *Input*

Note: the second dimension of the array A must be at least $\max(1, \mathrm{N})$ if ALPHA $\neq 0$ and TRANSA = 'N', $\max(1, \mathrm{M})$ if ALPHA $\neq 0$ and TRANSA = 'T' or 'C' and at least 1 if ALPHA = 0.

On entry: the matrix A. If $\alpha = 0$, the array A is not referenced.

7: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F01CWF is called.

Constraints:

if TRANSA = 'N', LDA $\geq \max(1, \mathrm{M})$;
otherwise LDA $\geq \max(1, \mathrm{N})$.

8: BETA – ***complex*16*** *Input*

On entry: the scalar β, by which matrix B is multiplied before addition.

9: B(LDB,$*$) – ***complex*16*** array *Input*

Note: the second dimension of the array B must be at least $\max(1, \mathrm{N})$ if BETA $\neq 0$ and TRANSB = 'N', $\max(1, \mathrm{M})$ if BETA $\neq 0$ and TRANSB = 'T' or 'C' and at least 1 if ALPHA = 0.

On entry: the matrix B. If $\beta = 0$, the array B is not referenced.

10: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F01CWF is called.

Constraints:

if TRANSB = 'N', LDB $\geq \max(1, \mathrm{M})$;
otherwise LDB $\geq \max(1, \mathrm{N})$.

11: C(LDC,$*$) – ***complex*16*** array *Output*

Note: the second dimension of the array C must be at least $\max(1, \mathrm{N})$.

On exit: the elements of the m by n matrix C.

12: LDC – INTEGER *Input*

On entry: the first dimension of the array C as declared in the (sub)program from which F01CWF is called.

Constraint: LDC $\geq \max(1, \mathrm{M})$.

13: 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 Section 3.3 in the Essential Introduction 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, one or both of TRANSA or TRANSB is not equal to 'N', 'T' or 'C'.

IFAIL $= 2$

On entry, one or both of M or N is less than 0.

IFAIL $= 3$

On entry, LDA $< \max(1, P)$, where P $=$ M if TRANSA $=$ 'N', and P $=$ N otherwise.

IFAIL $= 4$

On entry, LDB $< \max(1, P)$, where P $=$ M if TRANSB $=$ 'N', and P $=$ N otherwise.

IFAIL $= 5$

On entry, LDC $< \max(1, \mathrm{M})$.

7 Accuracy

The results returned by F01CWF are accurate to ***machine precision***.

8 Further Comments

The time taken for a call of F01CWF varies with M, N and the values of α and β. The routine is quickest if either or both of α and β are equal to zero, or plus or minus unity.

9 Example

The following program reads in a pair of matrices A and B, along with values for TRANSA, TRANSB, ALPHA and BETA, and adds them together, printing the result matrix C. The process is continued until the end of the input stream is reached.

9.1 Program Text

```
*     F01CWF Example Program Text
*     Mark 18 Revised. NAG Copyright 1997.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LDA, LDB, LDC
      PARAMETER        (NMAX=6,LDA=NMAX,LDB=LDA,LDC=LDA)
*     .. Local Scalars ..
      COMPLEX *16      ALPHA, BETA
      INTEGER          I, IFAIL, J, M, N, NCOLA, NCOLB, NROWA, NROWB
      CHARACTER        TRANSA, TRANSB
*     .. Local Arrays ..
      COMPLEX *16      A(LDA,NMAX), B(LDB,NMAX), C(LDC,NMAX)
*     .. External Subroutines ..
      EXTERNAL         F01CWF, X04DAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F01CWF Example Program Results'
      WRITE (NOUT,*)
*     Skip heading in data file
      READ (NIN,*)
   20 CONTINUE
*     Skip Subexample heading
      READ (NIN,*,END=80)
*     Read matrices A and B.
      READ (NIN,*) NROWA, NCOLA, TRANSA, ALPHA
*     Check that the arrays are large enough to hold the matrices.
      IF (NROWA.LE.LDA .AND. NCOLA.LE.NMAX) THEN
         DO 40 I = 1, NROWA
            READ (NIN,*) (A(I,J),J=1,NCOLA)
   40    CONTINUE
         READ (NIN,*) NROWB, NCOLB, TRANSB, BETA
         IF (NROWB.LE.LDB .AND. NCOLB.LE.NMAX) THEN
            DO 60 I = 1, NROWB
               READ (NIN,*) (B(I,J),J=1,NCOLB)
   60       CONTINUE
            IF (TRANSA.EQ.'N' .OR. TRANSA.EQ.'n') THEN
               M = NROWA
               N = NCOLA
            ELSE
               M = NCOLA
               N = NROWA
            END IF
            IFAIL = 1
*
*           Add the two matrices A and B.
            CALL F01CWF(TRANSA,TRANSB,M,N,ALPHA,A,LDA,BETA,B,LDB,C,LDC,
     +                  IFAIL)
*
            IF (IFAIL.EQ.0) THEN
*              Print the result matrix C.
               WRITE (NOUT,99999) TRANSA, TRANSB
               WRITE (NOUT,99998) ALPHA, BETA
               CALL X04DAF('G','X',M,N,C,LDC,'Matrix C:',IFAIL)
               WRITE (NOUT,*)
               GO TO 20
            ELSE
               WRITE (NOUT,99997) IFAIL
            END IF
         END IF
      END IF
   80 CONTINUE
*
99999 FORMAT (1X,'TRANSA = ''',A,''', TRANSB = ''',A,''',')
99998 FORMAT (1X,'ALPHA = (',1P,E11.4,',',E11.4,'), BETA = (',E11.4,',',
     +       E11.4,')')
99997 FORMAT (1X,/1X,' ** F01CWF returned with IFAIL = ',I5)
      END
```

9.2 Program Data

```
F01CWF Example Program Data
Example 1:
4 3 'N'   (1.0, 0.0)
  ( 1.0, 2.0)  ( 2.5,-1.5)  ( 2.5,-1.0)
  (-2.0,-2.0)  ( 2.0,-1.0)  (-1.5,-1.0)
  ( 3.5,-1.5)  ( 2.0, 1.5)  ( 2.0, 3.0)
  (-2.5, 0.0)  (-3.0, 2.5)  (-2.0, 2.0)
4 3 'N'   (1.0, 0.0)
  ( 2.0, 1.0)  (-2.5, 3.0)  (-0.5, 0.0)
  ( 1.0, 0.0)  ( 1.0,-1.5)  ( 1.5,-1.5)
  (-1.5,-0.5)  ( 2.5,-2.0)  (-0.5, 1.0)
  ( 2.5,-1.5)  (-1.0, 1.5)  ( 2.0, 3.0)
Example 2:
2 3 'N'   (1.0, 0.0)
  ( 1.0, 1.0)  ( 2.5,-1.5)  ( 3.0, 1.5)
  (-2.0,-0.5)  ( 2.0, 1.5)  (-1.5,-2.5)
3 2 'T'  (-1.0, 0.0)
  ( 2.0, 1.0)  (-2.5, 2.0)
  ( 1.0, 0.0)  ( 1.0, 1.5)
  (-1.5,-0.5)  ( 2.5,-1.0)
```

9.3 Program Results

```
 F01CWF Example Program Results

 TRANSA = 'N', TRANSB = 'N',
 ALPHA = ( 1.0000E+00, 0.0000E+00), BETA = ( 1.0000E+00, 0.0000E+00)
 Matrix C:
             1          2          3
 1      3.0000     0.0000     2.0000
        3.0000     1.5000    -1.0000

 2     -1.0000     3.0000     0.0000
       -2.0000    -2.5000    -2.5000

 3      2.0000     4.5000     1.5000
       -2.0000    -0.5000     4.0000

 4      0.0000    -4.0000     0.0000
       -1.5000     4.0000     5.0000

 TRANSA = 'N', TRANSB = 'T',
 ALPHA = ( 1.0000E+00, 0.0000E+00), BETA = (-1.0000E+00, 0.0000E+00)
 Matrix C:
             1          2          3
 1     -1.0000     1.5000     4.5000
        0.0000    -1.5000     2.0000

 2      0.5000     1.0000    -4.0000
       -2.5000     0.0000    -1.5000
```