C06FKF – NAG Fortran Library Routine Document

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

C06FKF calculates the circular convolution or correlation of two real vectors of period n (using a work array for extra speed).

2 Specification

```
SUBROUTINE C06FKF(JOB, X, Y, N, WORK, IFAIL)
INTEGER           JOB, N, IFAIL
real              X(N), Y(N), WORK(N)
```

3 Description

This routine computes:

if JOB = 1, the discrete **convolution** of x and y, defined by:

$$z_k = \sum_{j=0}^{n-1} x_j y_{k-j} = \sum_{j=0}^{n-1} x_{k-j} y_j;$$

if JOB = 2, the discrete **correlation** of x and y defined by:

$$w_k = \sum_{j=0}^{n-1} x_j y_{k+j}.$$

Here x and y are real vectors, assumed to be periodic, with period n, i.e., $x_j = x_{j\pm n} = x_{j\pm 2n} = \cdots$; z and w are then also periodic with period n.

Note. This usage of the terms 'convolution' and 'correlation' is taken from Brigham [1]. The term 'convolution' is sometimes used to denote both these computations.

If $\hat{x}$, $\hat{y}$, $\hat{z}$ and $\hat{w}$ are the discrete Fourier transforms of these sequences, i.e.,

$$\hat{x}_k = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} x_j \times \exp\left(-i\frac{2\pi jk}{n}\right), \text{etc.},$$

then $\hat{z}_k = \sqrt{n}.\hat{x}_k \hat{y}_k$

and $\hat{w}_k = \sqrt{n}.\bar{\hat{x}}_k \hat{y}_k$

(the bar denoting complex conjugate).

This routine calls the same auxiliary routines as C06FAF and C06FBF to compute discrete Fourier transforms, and there are some restrictions on the value of n.

4 References

[1] Brigham E O (1973) *The Fast Fourier Transform* Prentice–Hall

5 Parameters

1: JOB — INTEGER *Input*

On entry: the computation to be performed:

if JOB = 1, $z_k = \sum_{j=0}^{n-1} x_j y_{k-j}$ (convolution);

if JOB = 2, $w_k = \sum_{j=0}^{n-1} x_j y_{k+j}$ (correlation).

Constraint: JOB = 1 or 2.

2: X(N) — ***real*** array *Input/Output*

On entry: the elements of one period of the vector x. If X is declared with bounds (0:N−1) in the (sub)program from which C06FKF is called, then X(j) must contains x_j, for $j = 0, 1, \ldots, n-1$.

On exit: the corresponding elements of the discrete convolution or correlation.

3: Y(N) — ***real*** array *Input/Output*

On entry: the elements of one period of the vector y. If Y is declared with bounds (0:N−1) in the (sub)program from which C06FKF is called, then Y(j) must contain y_j, for $j = 0, 1, \ldots, n-1$.

On exit: the discrete Fourier transform of the convolution or correlation returned in the array X; the transform is stored in Hermitian form, exactly as described in the document for C06FAF.

4: N — INTEGER *Input*

On entry: the number of values, n, in one period of the vectors X and Y. The largest prime factor of N must not exceed 19 and the total number of prime factors of N, counting repetitions, must not exceed 20.

Constraint: N > 1.

5: WORK(N) — ***real*** array *Workspace*

6: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

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

6 Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

At least one of the prime factors of N is greater than 19.

IFAIL = 2

N has more than 20 prime factors.

IFAIL = 3

N $\leq$ 1.

IFAIL = 4

JOB $\neq$ 1 or 2.

7 Accuracy

The results should be accurate to within a small multiple of the ***machine precision***.

8 Further Comments

The time taken by the routine is approximately proportional to $n \times \log n$, but also depends on the factorization of n. The routine is somewhat faster than average if the only prime factors of n are 2, 3 or 5; and fastest of all if n is a power of 2.

9 Example

This program reads in the elements of one period of two real vectors x and y, and prints their discrete convolution and correlation (as computed by C06FKF). In realistic computations the number of data values would be much larger.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     C06FKF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NMAX
      PARAMETER        (NMAX=64)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      INTEGER          IFAIL, J, N
*     .. Local Arrays ..
      real             WORK(NMAX), XA(NMAX), XB(NMAX), YA(NMAX),
     +                 YB(NMAX)
*     .. External Subroutines ..
      EXTERNAL         C06FKF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'C06FKF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
   20 READ (NIN,*,END=80) N
      WRITE (NOUT,*)
      IF (N.GT.1 .AND. N.LE.NMAX) THEN
         DO 40 J = 1, N
            READ (NIN,*) XA(J), YA(J)
            XB(J) = XA(J)
            YB(J) = YA(J)
   40    CONTINUE
         IFAIL = 0
*
         CALL C06FKF(1,XA,YA,N,WORK,IFAIL)
         CALL C06FKF(2,XB,YB,N,WORK,IFAIL)
*
         WRITE (NOUT,*) '        Convolution  Correlation'
         WRITE (NOUT,*)
         DO 60 J = 1, N
            WRITE (NOUT,99999) J - 1, XA(J), XB(J)
   60    CONTINUE
         GO TO 20
      ELSE
         WRITE (NOUT,*) 'Invalid value of N'
      END IF
   80 STOP
```

```
*
99999 FORMAT (1X,I5,2F13.5)
      END
```

9.2 Program Data

```
C06FKF Example Program Data
    9
      1.00      0.50
      1.00      0.50
      1.00      0.50
      1.00      0.50
      1.00      0.00
      0.00      0.00
      0.00      0.00
      0.00      0.00
      0.00      0.00
```

9.3 Program Results

```
 C06FKF Example Program Results

         Convolution  Correlation

     0      0.50000      2.00000
     1      1.00000      1.50000
     2      1.50000      1.00000
     3      2.00000      0.50000
     4      2.00000      0.00000
     5      1.50000      0.50000
     6      1.00000      1.00000
     7      0.50000      1.50000
     8      0.00000      2.00000
```