NAG Library Routine Document

C06PKF

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

C06PKF calculates the circular convolution or correlation of two complex vectors of period n.

2 Specification

```
SUBROUTINE C06PKF (JOB, X, Y, N, WORK, IFAIL)
INTEGER                 JOB, N, IFAIL
COMPLEX (KIND=nag_wp) X(N), Y(N), WORK(*)
```

3 Description

C06PKF 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

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

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

Note: this usage of the terms 'convolution' and 'correlation' is taken from Brigham (1974). 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, and $\tilde{x}$ is the inverse discrete Fourier transform of the sequence x_j, i.e.,

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

and

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

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).

4 References

Brigham E O (1974) *The Fast Fourier Transform* Prentice–Hall

5 Parameters

1: JOB – INTEGER *Input*

On entry: the computation to be performed:

JOB $= 1$

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

JOB $= 2$

$$w_k = \sum_{j=0}^{n-1} \bar{x}_j y_{k+j} \text{ (correlation).}$$

Constraint: JOB $= 1$ or 2.

2: X(N) – COMPLEX (KIND=nag_wp) array *Input/Output*

On entry: the elements of one period of the vector x. If X is declared with bounds $(0 : \mathrm{N} - 1)$ in the subroutine from which C06PKF is called, then $\mathrm{X}(j)$ must contain x_j, for $j = 0, 1, \ldots, n-1$.

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

3: Y(N) – COMPLEX (KIND=nag_wp) array *Input/Output*

On entry: the elements of one period of the vector y. If Y is declared with bounds $(0 : \mathrm{N} - 1)$ in the subroutine from which C06PKF is called, then $\mathrm{Y}(j)$ 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.

4: N – INTEGER *Input*

On entry: n, the number of values in one period of the vectors X and Y. The total number of prime factors of N, counting repetitions, must not exceed 30.

Constraint: N ≥ 1.

5: WORK$(*)$ – COMPLEX (KIND=nag_wp) array *Workspace*

Note: the dimension of the array WORK must be at least $2 \times \mathrm{N} + 15$.

The workspace requirements as documented for C06PKF may be an overestimate in some implementations.

On exit: the real part of WORK(1) contains the minimum workspace required for the current value of N with this implementation.

6: 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.

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.**

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, N < 1.

IFAIL $= 2$

On entry, JOB $\neq 1$ or 2.

IFAIL $= 3$

An unexpected error has occurred in an internal call. Check all subroutine calls and array dimensions. Seek expert help.

IFAIL $= 4$

On entry, N has more than 30 prime factors.

IFAIL $= -99$

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

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

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

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

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

8 Parallelism and Performance

C06PKF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

C06PKF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

The time taken is approximately proportional to $n \times \log(n)$, but also depends on the factorization of n. C06PKF is faster if the only prime factors of n are 2, 3 or 5; and fastest of all if n is a power of 2.

10 Example

This example reads in the elements of one period of two complex vectors x and y, and prints their discrete convolution and correlation (as computed by C06PKF). In realistic computations the number of data values would be much larger.

10.1 Program Text

```
    Program c06pkfe

!     C06PKF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: c06pkf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: ieof, ifail, j, n
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: work(:), xa(:), xb(:), ya(:), yb(:)
!     .. Executable Statements ..
      Write (nout,*) 'C06PKF Example Program Results'
      Write (nout,*)
!     Skip heading in data file
      Read (nin,*)
loop: Do
        Read (nin,*,Iostat=ieof) n
        If (ieof<0) Exit loop

        Allocate (work(2*n+15),xa(n),xb(n),ya(n),yb(n))
        Read (nin,*)(xa(j),ya(j),j=1,n)
        xb(1:n) = xa(1:n)
        yb(1:n) = ya(1:n)

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call c06pkf(1,xa,ya,n,work,ifail)

        Call c06pkf(2,xb,yb,n,work,ifail)

        Write (nout,*) '        Convolution            Correlation'
        Write (nout,*)
        Do j = 0, n - 1
          Write (nout,99999) j, xa(j+1), xb(j+1)
        End Do
        Deallocate (work,xa,xb,ya,yb)
      End Do loop

99999 Format (1X,I5,2(1X,'(',F9.5,',',F9.5,')'))
    End Program c06pkfe
```

10.2 Program Data

```
C06PKF Example Program Data
    9                                          : n
      (1.0E0,-0.5E0)      (0.5E0,-0.25E0)
      (1.0E0,-0.5E0)      (0.5E0,-0.25E0)
      (1.0E0,-0.5E0)      (0.5E0,-0.25E0)
      (1.0E0,-0.5E0)      (0.5E0,-0.25E0)
      (1.0E0,-0.5E0)      (0.0E0,-0.25E0)
      (0.0E0,-0.5E0)      (0.0E0,-0.25E0)
      (0.0E0,-0.5E0)      (0.0E0,-0.25E0)
      (0.0E0,-0.5E0)      (0.0E0,-0.25E0)
      (0.0E0,-0.5E0)      (0.0E0,-0.25E0)       : xa, ya
```

10.3 Program Results

```
 C06PKF Example Program Results

         Convolution            Correlation

    0 ( -0.62500, -2.25000) (  3.12500, -0.25000)
    1 ( -0.12500, -2.25000) (  2.62500, -0.25000)
    2 (  0.37500, -2.25000) (  2.12500, -0.25000)
    3 (  0.87500, -2.25000) (  1.62500, -0.25000)
    4 (  0.87500, -2.25000) (  1.12500, -0.25000)
    5 (  0.37500, -2.25000) (  1.62500, -0.25000)
    6 ( -0.12500, -2.25000) (  2.12500, -0.25000)
    7 ( -0.62500, -2.25000) (  2.62500, -0.25000)
    8 ( -1.12500, -2.25000) (  3.12500, -0.25000)
```