NAG Library Routine Document

C06PCF

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

C06PCF calculates the discrete Fourier transform of a sequence of n complex data values (using complex data type).

2 Specification

```
SUBROUTINE C06PCF (DIRECT, X, N, WORK, IFAIL)

INTEGER               N, IFAIL
COMPLEX (KIND=nag_wp) X(N), WORK(*)
CHARACTER(1)          DIRECT
```

3 Description

Given a sequence of n complex data values z_j, for $j = 0, 1, \ldots, n - 1$, C06PCF calculates their (**forward** or **backward**) discrete Fourier transform (DFT) defined by

$$\hat{z}_k = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} z_j \times \exp\left(\pm i \frac{2\pi jk}{n}\right), \qquad k = 0, 1, \ldots, n - 1.$$

(Note the scale factor of $\frac{1}{\sqrt{n}}$ in this definition.) The minus sign is taken in the argument of the exponential within the summation when the forward transform is required, and the plus sign is taken when the backward transform is required.

A call of C06PCF with DIRECT = 'F' followed by a call with DIRECT = 'B' will restore the original data.

C06PCF uses a variant of the fast Fourier transform (FFT) algorithm (see Brigham (1974)) known as the Stockham self-sorting algorithm, which is described in Temperton (1983). If n is a large prime number or if n contains large prime factors, then the Fourier transform is performed using Bluestein's algorithm (see Bluestein (1968)), which expresses the DFT as a convolution that in turn can be efficiently computed using FFTs of highly composite sizes.

4 References

Bluestein L I (1968) A linear filtering approach to the computation of the discrete Fourier transform *Northeast Electronics Research and Engineering Meeting Record 10* 218–219

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

Temperton C (1983) Self-sorting mixed-radix fast Fourier transforms *J. Comput. Phys.* **52** 1–23

5 Parameters

1: DIRECT – CHARACTER(1) *Input*

On entry: if the forward transform as defined in Section 3 is to be computed, then DIRECT must be set equal to 'F'.

If the backward transform is to be computed then DIRECT must be set equal to 'B'.

Constraint: DIRECT = 'F' or 'B'.

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

On entry: if X is declared with bounds $(0 : \mathrm{N} - 1)$ in the subroutine from which C06PCF is called, then $\mathrm{X}(j)$ must contain z_j, for $j = 0, 1, \ldots, n - 1$.

On exit: the components of the discrete Fourier transform. If X is declared with bounds $(0 : \mathrm{N} - 1)$ in the subroutine from which C06PCF is called, then for $0 \leq k \leq n - 1$, $\hat{z}_k$ is contained in $\mathrm{X}(k)$.

3: N – INTEGER *Input*

On entry: n, the number of data values. The total number of prime factors of N, counting repetitions, must not exceed 30.

Constraint: $\mathrm{N} \geq 1$.

4: 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 C06PCF may be an overestimate in some implementations.

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

5: 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: $\mathrm{IFAIL} = 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry $\mathrm{IFAIL} = 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

$\mathrm{IFAIL} = 1$

On entry, $\mathrm{N} < 1$.

$\mathrm{IFAIL} = 2$

On entry, $\mathrm{DIRECT} \neq$ 'F' or 'B'.

$\mathrm{IFAIL} = 3$

On entry, N has more than 30 prime factors.

$\mathrm{IFAIL} = 4$

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

7 Accuracy

Some indication of accuracy can be obtained by performing a subsequent inverse transform and comparing the results with the original sequence (in exact arithmetic they would be identical).

8 Further Comments

The time taken is approximately proportional to $n \times \log n$, but also depends on the factorization of n. C06PCF 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. When the Bluestein's FFT algorithm is in use, an addtional workspace of size approximately $8n$ is allocated internally.

9 Example

This example reads in a sequence of complex data values and prints their discrete Fourier transform (as computed by C06PCF with DIRECT = 'F'). It then performs an inverse transform using C06PCF with DIRECT = 'B', and prints the sequence so obtained alongside the original data values.

9.1 Program Text

```
    Program c06pcfe

!     C06PCF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: c06pcf, 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(:), x(:), xx(:)
!     .. Executable Statements ..
      Write (nout,*) 'C06PCF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
loop: Do
        Read (nin,*,Iostat=ieof) n
        If (ieof<0) Exit loop

        Allocate (work(2*n+15),x(0:n-1),xx(0:n-1))
        Read (nin,*) x(0:n-1)
        xx(0:n-1) = x(0:n-1)

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call c06pcf('F',x,n,work,ifail)

        Write (nout,*)
        Write (nout,*) 'Components of discrete Fourier transform'
        Write (nout,*)
        Write (nout,*) '               Real      Imag'
        Write (nout,*)
        Do j = 0, n - 1
          Write (nout,99999) j, x(j)
        End Do

        Call c06pcf('B',x,n,work,ifail)

        Write (nout,*)
        Write (nout,*) 'Original sequence as restored by inverse transform'
```

```
        Write (nout,*)
        Write (nout,*) '                Original                 Restored'
        Write (nout,*) '             Real      Imag             Real      Imag'
        Write (nout,*)
        Do j = 0, n - 1
          Write (nout,99999) j, xx(j), x(j)
        End Do
        Deallocate (work,x,xx)
      End Do loop

99999 Format (1X,I5,2(:5X,'(',F8.5,',',F8.5,')'))
    End Program c06pcfe
```

9.2 Program Data

```
C06PCF Example Program Data
    7                  : n
  (0.34907,  -0.37168)
  (0.54890,  -0.35669)
  (0.74776,  -0.31175)
  (0.94459,  -0.23702)
  (1.13850,  -0.13274)
  (1.32850,   0.00074)
  (1.51370,   0.16298)  : x
```

9.3 Program Results

```
 C06PCF Example Program Results

 Components of discrete Fourier transform

              Real      Imag

     0     ( 2.48361,-0.47100)
     1     (-0.55180, 0.49684)
     2     (-0.36711, 0.09756)
     3     (-0.28767,-0.05865)
     4     (-0.22506,-0.17477)
     5     (-0.14825,-0.30840)
     6     ( 0.01983,-0.56496)

 Original sequence as restored by inverse transform

                 Original                 Restored
              Real      Imag             Real      Imag

     0     ( 0.34907,-0.37168)     ( 0.34907,-0.37168)
     1     ( 0.54890,-0.35669)     ( 0.54890,-0.35669)
     2     ( 0.74776,-0.31175)     ( 0.74776,-0.31175)
     3     ( 0.94459,-0.23702)     ( 0.94459,-0.23702)
     4     ( 1.13850,-0.13274)     ( 1.13850,-0.13274)
     5     ( 1.32850, 0.00074)     ( 1.32850, 0.00074)
     6     ( 1.51370, 0.16298)     ( 1.51370, 0.16298)
```