NAG Library Routine Document

C06PQF

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

C06PQF computes the discrete Fourier transforms of m sequences, each containing n real data values or a Hermitian complex sequence stored column-wise in a complex storage format.

2 Specification

```
SUBROUTINE C06PQF (DIRECT, N, M, X, WORK, IFAIL)

INTEGER              N, M, IFAIL
REAL (KIND=nag_wp) X((N+2)*M), WORK(*)
CHARACTER(1)         DIRECT
```

3 Description

Given m sequences of n real data values x_j^p, for $j = 0, 1, \ldots, n-1$ and $p = 1, 2, \ldots, m$, C06PQF simultaneously calculates the Fourier transforms of all the sequences defined by

$$\hat{z}_k^p = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} x_j^p \times \exp\left(-i\frac{2\pi jk}{n}\right), \qquad k0, 1, \ldots, n-1 \text{ and } p = 1, 2, \ldots, m.$$

The transformed values $\hat{z}_k^p$ are complex, but for each value of p the $\hat{z}_k^p$ form a Hermitian sequence (i.e., $\hat{z}_{n-k}^p$ is the complex conjugate of $\hat{z}_k^p$), so they are completely determined by mn real numbers (since $\hat{z}_0^p$ is real, as is $\hat{z}_{n/2}^p$ for n even).

Alternatively, given m Hermitian sequences of n complex data values z_j^p, this routine simultaneously calculates their inverse (**backward**) discrete Fourier transforms defined by

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

The transformed values $\hat{x}_k^p$ are real.

(Note the scale factor $\frac{1}{\sqrt{n}}$ in the above definition.)

A call of C06PQF with DIRECT = 'F' followed by a call with DIRECT = 'B' will restore the original data.

The routine 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). Special coding is provided for the factors 2, 3, 4 and 5.

4 References

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

Temperton C (1983) Fast mixed-radix real Fourier transforms *J. Comput. Phys.* **52** 340–350

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: N – INTEGER *Input*

On entry: n, the number of real or complex values in each sequence.

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

3: M – INTEGER *Input*

On entry: m, the number of sequences to be transformed.

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

4: $\mathrm{X}((\mathrm{N}+2) \times \mathrm{M})$ – REAL (KIND=nag_wp) array *Input/Output*

On entry: the data must be stored in X as if in a two-dimensional array of dimension $(0 : \mathrm{N}+1, 1 : \mathrm{M})$; each of the m sequences is stored in a **column** of the array. In other words, if the data values of the pth sequence to be transformed are denoted by x_j^p, for $j = 0, 1, \ldots, n-1$, then:

if DIRECT = 'F', $\mathrm{X}((p-1) \times (\mathrm{N}+2) + j)$ must contain x_j^p, for $j = 0, 1, \ldots, n-1$ and $p = 1, 2, \ldots, m$;

if DIRECT = 'B', $\mathrm{X}((p-1) \times (\mathrm{N}+2) + 2 \times k)$ and $\mathrm{X}((p-1) \times (\mathrm{N}+2) + 2 \times k + 1)$ must contain the real and imaginary parts respectively of $\hat{z}_k^p$, for $k = 0, 1, \ldots, n/2$ and $p = 1, 2, \ldots, m$. (Note that for the sequence $\hat{z}_k^p$ to be Hermitian, the imaginary part of $\hat{z}_0^p$, and of $\hat{z}_{n/2}^p$ for n even, must be zero.)

On exit:

if DIRECT = 'F' and X is declared with bounds $(0 : \mathrm{N}+1, 1 : \mathrm{M})$ then $\mathrm{X}(2 \times k, p)$ and $\mathrm{X}(2 \times k + 1, p)$ will contain the real and imaginary parts respectively of $\hat{z}_k^p$, for $k = 0, 1, \ldots, n/2$ and $p = 1, 2, \ldots, m$;

if DIRECT = 'B' and X is declared with bounds $(0 : \mathrm{N}+1, 1 : \mathrm{M})$ then $\mathrm{X}(j, p)$ will contain x_j^p, for $j = 0, 1, \ldots, n-1$ and $p = 1, 2, \ldots, m$.

5: $\mathrm{WORK}(*)$ – REAL (KIND=nag_wp) array *Workspace*

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

The workspace requirements as documented for C06PQF may be an overestimate in some implementations.

On exit: $\mathrm{WORK}(1)$ contains the minimum workspace required for the current values of M and 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, M < 1.

IFAIL = 2

On entry, N < 1.

IFAIL = 3

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

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 by C06PQF is approximately proportional to $nm \log n$, but also depends on the factors of n. C06PQF is fastest if the only prime factors of n are 2, 3 and 5, and is particularly slow if n is a large prime, or has large prime factors.

9 Example

This example reads in sequences of real data values and prints their discrete Fourier transforms (as computed by C06PQF with DIRECT = 'F'), after expanding them from complex Hermitian form into a full complex sequences.

Inverse transforms are then calculated by calling C06PQF with DIRECT = 'B' showing that the original sequences are restored.

9.1 Program Text

```
    Program c06pqfe

!     C06PQF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: c06pqf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ieof, ifail, j, m, n
```

```
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: work(:), x(:)
!     .. Executable Statements ..
      Write (nout,*) 'C06PQF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
loop: Do
        Read (nin,*,Iostat=ieof) m, n
        If (ieof<0) Exit loop

        Allocate (work((m+2)*n+15),x(m*(n+2)))
        Do j = 1, m*(n+2), n + 2
          Read (nin,*)(x(j+i),i=0,n-1)
        End Do
        Write (nout,*)
        Write (nout,*) 'Original data values'
        Write (nout,*)
        Do j = 1, m*(n+2), n + 2
          Write (nout,99999) '      ', (x(j+i),i=0,n-1)
        End Do

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call c06pqf('F',n,m,x,work,ifail)

        Write (nout,*)
        Write (nout,*) &
          'Discrete Fourier transforms in complex Hermitian format'
        Do j = 1, m*(n+2), n + 2
          Write (nout,*)
          Write (nout,99999) 'Real ', (x(j+2*i),i=0,n/2)
          Write (nout,99999) 'Imag ', (x(j+2*i+1),i=0,n/2)
        End Do
        Write (nout,*)
        Write (nout,*) 'Fourier transforms in full complex form'

        Do j = 1, m*(n+2), n + 2
          Write (nout,*)
          Write (nout,99999) 'Real ', (x(j+2*i),i=0,n/2), &
            (x(j+2*(n-i)),i=n/2+1,n-1)
          Write (nout,99999) 'Imag ', (x(j+2*i+1),i=0,n/2), &
            (-x(j+2*(n-i)+1),i=n/2+1,n-1)
        End Do

        Call c06pqf('B',n,m,x,work,ifail)

        Write (nout,*)
        Write (nout,*) 'Original data as restored by inverse transform'
        Write (nout,*)
        Do j = 1, m*(n+2), n + 2
          Write (nout,99999) '      ', (x(j+i),i=0,n-1)
        End Do
        Deallocate (x,work)
      End Do loop

99999 Format (1X,A,9(:1X,F10.4))
    End Program c06pqfe
```

9.2 Program Data

```
C06PQF Example Program Data
     3    6                                                 : m, n
     0.3854    0.6772    0.1138    0.6751    0.6362    0.1424
     0.5417    0.2983    0.1181    0.7255    0.8638    0.8723
     0.9172    0.0644    0.6037    0.6430    0.0428    0.4815   : x
```

9.3 Program Results

```
C06PQF Example Program Results

Original data values

          0.3854     0.6772     0.1138     0.6751     0.6362     0.1424
          0.5417     0.2983     0.1181     0.7255     0.8638     0.8723
          0.9172     0.0644     0.6037     0.6430     0.0428     0.4815

Discrete Fourier transforms in complex Hermitian format

Real      1.0737    -0.1041     0.1126    -0.1467
Imag      0.0000    -0.0044    -0.3738     0.0000

Real      1.3961    -0.0365     0.0780    -0.1521
Imag      0.0000     0.4666    -0.0607     0.0000

Real      1.1237     0.0914     0.3936     0.1530
Imag      0.0000    -0.0508     0.3458     0.0000

Fourier transforms in full complex form

Real      1.0737    -0.1041     0.1126    -0.1467     0.1126    -0.1041
Imag      0.0000    -0.0044    -0.3738     0.0000     0.3738     0.0044

Real      1.3961    -0.0365     0.0780    -0.1521     0.0780    -0.0365
Imag      0.0000     0.4666    -0.0607     0.0000     0.0607    -0.4666

Real      1.1237     0.0914     0.3936     0.1530     0.3936     0.0914
Imag      0.0000    -0.0508     0.3458     0.0000    -0.3458     0.0508

Original data as restored by inverse transform

          0.3854     0.6772     0.1138     0.6751     0.6362     0.1424
          0.5417     0.2983     0.1181     0.7255     0.8638     0.8723
          0.9172     0.0644     0.6037     0.6430     0.0428     0.4815
```