NAG Library Routine Document

C06HCF

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

C06HCF computes the discrete quarter-wave Fourier sine transforms of m sequences of real data values. This routine is designed to be particularly efficient on vector processors.

2 Specification

```
SUBROUTINE C06HCF (DIRECT, M, N, X, INIT, TRIG, WORK, IFAIL)

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

3 Description

Given m sequences of n real data values x_j^p, for $j = 1, 2, \ldots, n$ and $p = 1, 2, \ldots, m$, C06HCF simultaneously calculates the quarter-wave Fourier sine transforms of all the sequences defined by

$$\hat{x}_k^p = \frac{1}{\sqrt{n}} \left\{ \sum_{j=1}^{n-1} x_j^p \times \sin\left(j(2k-1)\frac{\pi}{2n}\right) + \frac{1}{2} -1^{k-1} x_n^p \right\}, \qquad \text{if DIRECT} = \text{'F'},$$

or its inverse

$$x_k^p = \frac{2}{\sqrt{n}} \sum_{j=1}^{n} \hat{x}_j^p \times \sin\left((2j-1)k\frac{\pi}{2n}\right), \qquad \text{if DIRECT} = \text{'B'},$$

for $k = 1, 2, \ldots, n$ and $p = 1, 2, \ldots, m$.

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

A call of C06HCF with DIRECT = 'F' followed by a call with DIRECT = 'B' will restore the original data.

The transform calculated by this routine can be used to solve Poisson's equation when the solution is specified at the left boundary, and the derivative of the solution is specified at the right boundary (see Swarztrauber (1977)). (See the C06 Chapter Introduction.)

The routine uses a variant of the fast Fourier transform (FFT) algorithm (see Brigham (1974)) known as the Stockham self-sorting algorithm, described in Temperton (1983), together with pre- and post-processing stages described in Swarztrauber (1982). Special coding is provided for the factors 2, 3, 4, 5 and 6. This routine is designed to be particularly efficient on vector processors, and it becomes especially fast as m, the number of transforms to be computed in parallel, increases.

4 References

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

Swarztrauber P N (1977) The methods of cyclic reduction, Fourier analysis and the FACR algorithm for the discrete solution of Poisson's equation on a rectangle *SIAM Rev.* **19(3)** 490–501

Swarztrauber P N (1982) Vectorizing the FFT's *Parallel Computation* (ed G Rodrique) 51–83 Academic Press

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: M – INTEGER *Input*

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

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

3: N – INTEGER *Input*

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

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

4: X(M × N) – 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 $(1:\mathrm{M}, 1:\mathrm{N})$; each of the m sequences is stored in a **row** of the array. In other words, if the data values of the pth sequence to be transformed are denoted by x_j^p, for $j = 1, 2, \ldots, n$ and $p = 1, 2, \ldots, m$, then the mn elements of the array X must contain the values

$$x_1^1, x_1^2, \ldots, x_1^m, x_2^1, x_2^2, \ldots, x_2^m, \ldots, x_n^1, x_n^2, \ldots, x_n^m.$$

On exit: the m quarter-wave sine transforms stored as if in a two-dimensional array of dimension $(1:\mathrm{M}, 1:\mathrm{N})$. Each of the m transforms is stored in a **row** of the array, overwriting the corresponding original sequence. If the n components of the pth quarter-wave sine transform are denoted by $\hat{x}_k^p$, for $k = 1, 2, \ldots, n$ and $p = 1, 2, \ldots, m$, then the mn elements of the array X contain the values

$$\hat{x}_1^1, \hat{x}_1^2, \ldots, \hat{x}_1^m, \hat{x}_2^1, \hat{x}_2^2, \ldots, \hat{x}_2^m, \ldots, \hat{x}_n^1, \hat{x}_n^2, \ldots, \hat{x}_n^m.$$

5: INIT – CHARACTER(1) *Input*

On entry: indicates whether trigonometric coefficients are to be calculated.

INIT = 'I'
: Calculate the required trigonometric coefficients for the given value of n, and store in the array TRIG.

INIT = 'S' or 'R'
: The required trigonometric coefficients are assumed to have been calculated and stored in the array TRIG in a prior call to one of C06HAF, C06HBF, C06HCF or C06HDF. The routine performs a simple check that the current value of n is consistent with the values stored in TRIG.

Constraint: INIT = 'I', 'S' or 'R'.

6: TRIG(2 × N) – REAL (KIND=nag_wp) array *Input/Output*

On entry: if INIT = 'S' or 'R', TRIG must contain the required trigonometric coefficients calculated in a previous call of the routine. Otherwise TRIG need not be set.

On exit: contains the required coefficients (computed by the routine if INIT = 'I').

7: WORK(M × N) – REAL (KIND=nag_wp) array *Workspace*

8: 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, INIT $\neq$ 'I', 'S' or 'R'.

IFAIL $= 4$

Not used at this Mark.

IFAIL $= 5$

On entry, INIT $=$ 'S' or 'R', but the array TRIG and the current value of N are inconsistent.

IFAIL $= 6$

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

IFAIL $= 7$

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 C06HCF is approximately proportional to $nm \log n$, but also depends on the factors of n. C06HCF 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 quarter-wave sine transforms as computed by C06HCF with DIRECT = 'F'. It then calls the routine again with DIRECT = 'B' and prints the results which may be compared with the original data.

9.1 Program Text

```
    Program c06hcfe

!     C06HCF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: c06hcf, 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  :: trig(:), work(:), x(:)
!     .. Executable Statements ..
      Write (nout,*) 'C06HCF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
loop: Do
        Read (nin,*,Iostat=ieof) m, n
        If (ieof<0) Exit loop

        Allocate (trig(2*n),work(m*n),x(m*n))
        Do j = 1, m
          Read (nin,*)(x(i*m+j),i=0,n-1)
        End Do
        Write (nout,*)
        Write (nout,*) 'Original data values'
        Write (nout,*)
        Do j = 1, m
          Write (nout,99999)(x(i*m+j),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
!       -- Compute transform
        Call c06hcf('Forward',m,n,x,'Initial',trig,work,ifail)

        Write (nout,*)
        Write (nout,*) 'Discrete quarter-wave Fourier sine transforms'
        Write (nout,*)
        Do j = 1, m
          Write (nout,99999)(x(i*m+j),i=0,n-1)
        End Do

!       -- Compute inverse transform
        Call c06hcf('Backward',m,n,x,'Subsequent',trig,work,ifail)

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

```
        Deallocate (trig,work,x)
      End Do loop

99999 Format (6X,7F10.4)
    End Program c06hcfe
```

9.2 Program Data

```
C06HCF 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

```
 C06HCF 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 quarter-wave Fourier sine transforms

          0.7304    0.2078    0.1150    0.2577   -0.2869   -0.0815
          0.9274   -0.1152    0.2532    0.2883   -0.0026   -0.0635
          0.6268    0.3547    0.0760    0.3078    0.4987   -0.0507

 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
```