NAG Library Routine Document

C06FXF

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

C06FXF computes the three-dimensional discrete Fourier transform of a trivariate sequence of complex data values. This routine is designed to be particularly efficient on vector processors.

2 Specification

```
 SUBROUTINE C06FXF(N1, N2, N3, X, Y, INIT, TRIGN1, TRIGN2, TRIGN3, WORK,
1                  IFAIL)
 INTEGER          N1, N2, N3, IFAIL
 double precision X(N1*N2*N3), Y(N1*N2*N3), TRIGN1(2*N1), TRIGN2(2*N2),
1                 TRIGN3(2*N3), WORK(2*N1*N2*N3)
 CHARACTER*1      INIT
```

3 Description

C06FXF computes the three-dimensional discrete Fourier transform of a trivariate sequence of complex data values $z_{j_1 j_2 j_3}$, where $j_1 = 0, 1, \ldots, n_1 - 1$, $j_2 = 0, 1, \ldots, n_2 - 1$, $j_3 = 0, 1, \ldots, n_3 - 1$.

The discrete Fourier transform is here defined by

$$\hat{z}_{k_1 k_2 k_3} = \frac{1}{\sqrt{n_1 n_2 n_3}} \sum_{j_1=0}^{n_1-1} \sum_{j_2=0}^{n_2-1} \sum_{j_3=0}^{n_3-1} z_{j_1 j_2 j_3} \times \exp\left(-2\pi i \left(\frac{j_1 k_1}{n_1} + \frac{j_2 k_2}{n_2} + \frac{j_3 k_3}{n_3}\right)\right),$$

where $k_1 = 0, 1, \ldots, n_1 - 1$, $k_2 = 0, 1, \ldots, n_2 - 1$, $k_3 = 0, 1, \ldots, n_3 - 1$.

(Note the scale factor of $\frac{1}{\sqrt{n_1 n_2 n_3}}$ in this definition.)

To compute the inverse discrete Fourier transform, defined with $\exp(+2\pi i(\ldots))$ in the above formula instead of $\exp(-2\pi i(\ldots))$, this routine should be preceded and followed by calls of C06GCF to form the complex conjugates of the data values and the transform.

This routine calls C06FRF to perform multiple one-dimensional discrete Fourier transforms by the fast Fourier transform (FFT) algorithm (see Brigham (1974)). It is designed to be particularly efficient on vector processors.

4 References

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: N1 – INTEGER *Input*

On entry: n_1, the first dimension of the transform.

Constraint: N1 ≥ 1.

2: N2 – INTEGER *Input*

On entry: n_2, the second dimension of the transform.

Constraint: $\mathrm{N2} \geq 1$.

3: N3 – INTEGER *Input*

On entry: n_3, the third dimension of the transform.

Constraint: $\mathrm{N3} \geq 1$.

4: $\mathrm{X}(\mathrm{N1} \times \mathrm{N2} \times \mathrm{N3})$ – ***double precision*** array *Input/Output*
5: $\mathrm{Y}(\mathrm{N1} \times \mathrm{N2} \times \mathrm{N3})$ – ***double precision*** array *Input/Output*

On entry: the real and imaginary parts of the complex data values must be stored in arrays X and Y respectively. If X and Y are regarded as three-dimensional arrays of dimension $(0:\mathrm{N1}-1, 0:\mathrm{N2}-1, 0:\mathrm{N3}-1)$, then $\mathrm{X}(j_1, j_2, j_3)$ and $\mathrm{Y}(j_1, j_2, j_3)$ must contain the real and imaginary parts of $z_{j_1 j_2 j_3}$.

On exit: the real and imaginary parts respectively of the corresponding elements of the computed transform.

6: INIT – CHARACTER*1 *Input*

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

INIT = 'I'
: Calculate the required trigonometric coefficients for the given values of n_1, n_2 and n_3, and store in the corresponding arrays TRIGN1, TRIGN2 and TRIGN3.

INIT = 'S' or 'R'
: The required trigonometric coefficients are assumed to have been calculated and stored in the arrays TRIGN1, TRIGN2 and TRIGN3 in a prior call to C06FXF. The routine performs a simple check that the current values of n_1, n_2 and n_3 are consistent with the corresponding values stored in TRIGN1, TRIGN2 and TRIGN3.

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

7: $\mathrm{TRIGN1}(2 \times \mathrm{N1})$ – ***double precision*** array *Input/Output*
8: $\mathrm{TRIGN2}(2 \times \mathrm{N2})$ – ***double precision*** array *Input/Output*
9: $\mathrm{TRIGN3}(2 \times \mathrm{N3})$ – ***double precision*** array *Input/Output*

On entry: if INIT = 'S' or 'R', TRIGN1, TRIGN2 and TRIGN3 must contain the required coefficients calculated in a previous call of the routine. Otherwise TRIGN1, TRIGN2 and TRIGN3 need not be set. If $\mathrm{N1} = \mathrm{N2}$, the same array may be supplied for TRIGN1 and TRIGN2. Similar considerations apply if $\mathrm{N2} = \mathrm{N3}$ or $\mathrm{N1} = \mathrm{N3}$.

On exit: TRIGN1, TRIGN2 and TRIGN3 contain the required coefficients (computed by the routine if INIT = 'I').

10: $\mathrm{WORK}(2 \times \mathrm{N1} \times \mathrm{N2} \times \mathrm{N3})$ – ***double precision*** array *Workspace*

11: 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.

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

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.**

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, N1 < 1.

IFAIL = 2

On entry, N2 < 1.

IFAIL = 3

On entry, N3 < 1.

IFAIL = 4

On entry, INIT $\neq$ 'I', 'S' or 'R'.

IFAIL = 5

Not used at this Mark.

IFAIL = 6

On entry, INIT = 'S' or 'R', but at least one of the arrays TRIGN1, TRIGN2 and TRIGN3 is inconsistent with the current value of N1, N2 or N3.

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 is approximately proportional to $n_1n_2n_3 \times \log(n_1n_2n_3)$, but also depends on the factorization of the individual dimensions n_1, n_2 and n_3. C06FXF is faster if the only prime factors are 2, 3 or 5; and fastest of all if they are powers of 2.

9 Example

This example reads in a trivariate sequence of complex data values and prints the three-dimensional Fourier transform. It then performs an inverse transform and prints the sequence so obtained, which may be compared to the original data values.

9.1 Program Text

```
*     C06FXF Example Program Text
*     Mark 17 Release. NAG Copyright 1995.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          N1MAX, N2MAX, N3MAX, NMAX
      PARAMETER        (N1MAX=16,N2MAX=16,N3MAX=16,
     +                 NMAX=N1MAX*N2MAX*N3MAX)
```

```
*     .. Local Scalars ..
      INTEGER          IFAIL, N, N1, N2, N3
*     .. Local Arrays ..
      DOUBLE PRECISION TRIGN1(2*N1MAX), TRIGN2(2*N2MAX),
     +                 TRIGN3(2*N3MAX), WORK(2*NMAX), X(NMAX), Y(NMAX)
*     .. External Subroutines ..
      EXTERNAL         C06FXF, C06GCF, READXY, WRITXY
*     .. Executable Statements ..
      WRITE (NOUT,*) 'C06FXF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
   20 READ (NIN,*,END=40) N1, N2, N3
      N = N1*N2*N3
      IF (N.GE.1 .AND. N.LE.NMAX) THEN
         CALL READXY(NIN,X,Y,N1,N2,N3)
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Original data values'
         CALL WRITXY(NOUT,X,Y,N1,N2,N3)
         IFAIL = 1
*
*        -- Compute transform
         CALL C06FXF(N1,N2,N3,X,Y,'Initial',TRIGN1,TRIGN2,TRIGN3,WORK,
     +               IFAIL)
*
         IF (IFAIL.EQ.0) THEN
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Components of discrete Fourier transform'
            CALL WRITXY(NOUT,X,Y,N1,N2,N3)
*
*           -- Compute inverse transform
            CALL C06GCF(Y,N,IFAIL)
            CALL C06FXF(N1,N2,N3,X,Y,'Subsequent',TRIGN1,TRIGN2,TRIGN3,
     +                  WORK,IFAIL)
            CALL C06GCF(Y,N,IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     +        'Original sequence as restored by inverse transform'
            CALL WRITXY(NOUT,X,Y,N1,N2,N3)
            GO TO 20
         ELSE
            WRITE (NOUT,*)
            WRITE (NOUT,99999) ' ** C06FXF returned with IFAIL = ',
     +        IFAIL
         END IF
      ELSE
         WRITE (NOUT,*) ' ** Invalid value of n1, n2 or n3'
      END IF
   40 CONTINUE
*
99999 FORMAT (1X,A,I5)
      END
*
      SUBROUTINE READXY(NIN,X,Y,N1,N2,N3)
*     Read 3-dimensional complex data
*     .. Scalar Arguments ..
      INTEGER          N1, N2, N3, NIN
*     .. Array Arguments ..
      DOUBLE PRECISION X(N1,N2,N3), Y(N1,N2,N3)
*     .. Local Scalars ..
      INTEGER          I, J, K
*     .. Executable Statements ..
      DO 40 I = 1, N1
         DO 20 J = 1, N2
            READ (NIN,*) (X(I,J,K),K=1,N3)
            READ (NIN,*) (Y(I,J,K),K=1,N3)
   20    CONTINUE
   40 CONTINUE
      RETURN
      END
*
```

```
      SUBROUTINE WRITXY(NOUT,X,Y,N1,N2,N3)
*     Print 3-dimensional complex data
*     .. Scalar Arguments ..
      INTEGER          N1, N2, N3, NOUT
*     .. Array Arguments ..
      DOUBLE PRECISION X(N1,N2,N3), Y(N1,N2,N3)
*     .. Local Scalars ..
      INTEGER          I, J, K
*     .. Executable Statements ..
      DO 40 I = 1, N1
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'z(i,j,k) for i =', I
         DO 20 J = 1, N2
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Real ', (X(I,J,K),K=1,N3)
            WRITE (NOUT,99999) 'Imag ', (Y(I,J,K),K=1,N3)
   20    CONTINUE
   40 CONTINUE
      RETURN
*
99999 FORMAT (1X,A,7F10.3,/(6X,7F10.3))
99998 FORMAT (1X,A,I6)
      END
```

9.2 Program Data

```
C06FXF Example Program Data
2 3 4  : values of N1, N2, N3
     1.000     0.999     0.987     0.936   :  X(0,0,J), J=0,...,N3-1
     0.000    -0.040    -0.159    -0.352   :  Y(0,0,J), J=0,...,N3-1
     0.994     0.989     0.963     0.891   :  X(0,1,J), J=0,...,N3-1
    -0.111    -0.151    -0.268    -0.454   :  Y(0,1,J), J=0,...,N3-1
     0.903     0.885     0.823     0.694   :  X(0,2,J), J=0,...,N3-1
    -0.430    -0.466    -0.568    -0.720   :  Y(0,2,J), J=0,...,N3-1
     0.500     0.499     0.487     0.436   :  X(1,0,J), J=0,...,N3-1
     0.500     0.040     0.159     0.352   :  Y(1,0,J), J=0,...,N3-1
     0.494     0.489     0.463     0.391   :  X(1,1,J), J=0,...,N3-1
     0.111     0.151     0.268     0.454   :  Y(1,1,J), J=0,...,N3-1
     0.403     0.385     0.323     0.194   :  X(1,2,J), J=0,...,N3-1
     0.430     0.466     0.568     0.720   :  Y(1,2,J), J=0,...,N3-1
```

9.3 Program Results

```
 C06FXF Example Program Results

 Original data values

 z(i,j,k) for i =     1

 Real      1.000     0.999     0.987     0.936
 Imag      0.000    -0.040    -0.159    -0.352

 Real      0.994     0.989     0.963     0.891
 Imag     -0.111    -0.151    -0.268    -0.454

 Real      0.903     0.885     0.823     0.694
 Imag     -0.430    -0.466    -0.568    -0.720

 z(i,j,k) for i =     2

 Real      0.500     0.499     0.487     0.436
 Imag      0.500     0.040     0.159     0.352

 Real      0.494     0.489     0.463     0.391
 Imag      0.111     0.151     0.268     0.454

 Real      0.403     0.385     0.323     0.194
 Imag      0.430     0.466     0.568     0.720

 Components of discrete Fourier transform
```

```
z(i,j,k) for i =     1

Real      3.292     0.051     0.113     0.051
Imag      0.102    -0.042     0.102     0.246

Real      0.143     0.016    -0.024    -0.050
Imag     -0.086     0.153     0.127     0.086

Real      0.143    -0.050    -0.024     0.016
Imag      0.290     0.118     0.077     0.051

z(i,j,k) for i =     2

Real      1.225     0.355     0.000    -0.355
Imag     -1.620     0.083     0.162     0.083

Real      0.424     0.020     0.013    -0.007
Imag      0.320    -0.115    -0.091    -0.080

Real     -0.424     0.007    -0.013    -0.020
Imag      0.320    -0.080    -0.091    -0.115

Original sequence as restored by inverse transform

z(i,j,k) for i =     1

Real      1.000     0.999     0.987     0.936
Imag      0.000    -0.040    -0.159    -0.352

Real      0.994     0.989     0.963     0.891
Imag     -0.111    -0.151    -0.268    -0.454

Real      0.903     0.885     0.823     0.694
Imag     -0.430    -0.466    -0.568    -0.720

z(i,j,k) for i =     2

Real      0.500     0.499     0.487     0.436
Imag      0.500     0.040     0.159     0.352

Real      0.494     0.489     0.463     0.391
Imag      0.111     0.151     0.268     0.454

Real      0.403     0.385     0.323     0.194
Imag      0.430     0.466     0.568     0.720
```