NAG Library Routine Document

C06FUF

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

C06FUF computes the two-dimensional discrete Fourier transform of a bivariate sequence of complex data values. This routine is designed to be particularly efficient on vector processors.

2 Specification

```
SUBROUTINE C06FUF (M, N, X, Y, INIT, TRIGM, TRIGN, WORK, IFAIL)

INTEGER             M, N, IFAIL
REAL (KIND=nag_wp) X(M*N), Y(M*N), TRIGM(2*M), TRIGN(2*N), WORK(2*M*N)
CHARACTER(1)        INIT
```

3 Description

C06FUF computes the two-dimensional discrete Fourier transform of a bivariate sequence of complex data values $z_{j_1 j_2}$, for $j_1 = 0, 1, \ldots, m-1$ and $j_2 = 0, 1, \ldots, n-1$.

The discrete Fourier transform is here defined by

$$\hat{z}_{k_1 k_2} = \frac{1}{\sqrt{mn}} \sum_{j_1=0}^{m-1} \sum_{j_2=0}^{n-1} z_{j_1 j_2} \times \exp\left(-2\pi i \left(\frac{j_1 k_1}{m} + \frac{j_2 k_2}{n}\right)\right),$$

where $k_1 = 0, 1, \ldots, m-1$, $k_2 = 0, 1, \ldots, n-1$.

(Note the scale factor of $\frac{1}{\sqrt{mn}}$ 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 in 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: M – INTEGER *Input*

On entry: m, the length of the first dimension of Z. Consider the matrix Z with elements $Z_{ij} = z_{i+1\,j+1}$, where z is the bivariate sequence defined in Section 3, then m is the number of rows of Z.

Constraint: M ≥ 1.

2: N – INTEGER *Input*

On entry: n, the length of the second dimension of Z. Consider the matrix Z with elements $Z_{ij} = z_{i+1j+1}$, where z is the bivariate sequence defined in Section 3, then n is the number of columns of Z.

Constraint: N ≥ 1.

3: X(M × N) – REAL (KIND=nag_wp) array *Input/Output*
4: Y(M × N) – REAL (KIND=nag_wp) 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 two-dimensional arrays of dimension $(0 : \mathrm{M}-1, 0 : \mathrm{N}-1)$, then $\mathrm{X}(j_1, j_2)$ and $\mathrm{Y}(j_1, j_2)$ must contain the real and imaginary parts of $z_{j_1 j_2}$.

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

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 values of m and n, and store in the corresponding arrays TRIGM and TRIGN.

INIT = 'S' or 'R'
The required trigonometric coefficients are assumed to have been calculated and stored in the arrays TRIGM and TRIGN in a prior call to C06FUF. The routine performs a simple check that the current values of m and n are consistent with the corresponding values stored in TRIGM and TRIGN.

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

6: TRIGM(2 × M) – REAL (KIND=nag_wp) array *Input/Output*
7: TRIGN(2 × N) – REAL (KIND=nag_wp) array *Input/Output*

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

If $m = n$ the same array may be supplied for TRIGM and TRIGN.

On exit: TRIGM and TRIGN contain the required coefficients (computed by the routine if INIT = 'I').

8: WORK(2 × M × N) – REAL (KIND=nag_wp) array *Workspace*

9: 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 at least one of the arrays TRIGM and TRIGN is inconsistent with the current value of M or N.

IFAIL $= 6$

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 $mn \times \log(mn)$, but also depends on the factorization of the individual dimensions m and n. C06FUF 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 bivariate sequence of complex data values and prints the two-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

```
!   C06FUF Example Program Text
!   Mark 24 Release. NAG Copyright 2012.

    Module c06fufe_mod

!     C06FUF Example Program Module:
!            Parameters and User-defined Routines

!     .. Use Statements ..
      Use nag_library, Only: nag_wp
!     .. Implicit None Statement ..
      Implicit None
```

```
!     .. Parameters ..
      Integer, Parameter                   :: nin = 5, nout = 6
    Contains
      Subroutine readxy(nin,x,y,n1,n2)
!       Read 2-dimensional complex data

!       .. Scalar Arguments ..
        Integer, Intent (In)                 :: n1, n2, nin
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Out)     :: x(n1,n2), y(n1,n2)
!       .. Local Scalars ..
        Integer                              :: i, j
!       .. Executable Statements ..
        Do i = 1, n1
          Read (nin,*)(x(i,j),j=1,n2)
          Read (nin,*)(y(i,j),j=1,n2)
        End Do
        Return
      End Subroutine readxy

      Subroutine writxy(nout,x,y,n1,n2)
!       Print 2-dimensional complex data

!       .. Scalar Arguments ..
        Integer, Intent (In)                 :: n1, n2, nout
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (In)      :: x(n1,n2), y(n1,n2)
!       .. Local Scalars ..
        Integer                              :: i, j
!       .. Executable Statements ..
        Do i = 1, n1
          Write (nout,*)
          Write (nout,99999) 'Real ', (x(i,j),j=1,n2)
          Write (nout,99999) 'Imag ', (y(i,j),j=1,n2)
        End Do
        Return

99999   Format (1X,A,7F10.3/(6X,7F10.3))
      End Subroutine writxy
    End Module c06fufe_mod

    Program c06fufe

!     C06FUF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: c06fuf, c06gcf, nag_wp
      Use c06fufe_mod, Only: nin, nout, readxy, writxy
!     .. Implicit None Statement ..
      Implicit None
!     .. Local Scalars ..
      Integer                              :: ieof, ifail, m, n, n_bi
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable      :: trigm(:), trign(:), work(:),     &
                                              x(:), y(:)
!     .. Executable Statements ..
      Write (nout,*) 'C06FUF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
loop: Do
        Read (nin,*,Iostat=ieof) m, n
        If (ieof<0) Exit loop

        Allocate (trigm(2*m),trign(2*n),work(2*m*n),x(m*n),y(m*n))
        Call readxy(nin,x,y,m,n)
        Write (nout,*)
        Write (nout,*) 'Original data values'
        Call writxy(nout,x,y,m,n)

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
```

```
        ifail = 0
!       Compute transform
        Call c06fuf(m,n,x,y,'Initial',trigm,trign,work,ifail)

        Write (nout,*)
        Write (nout,*) 'Components of discrete Fourier transform'
        Call writxy(nout,x,y,m,n)

!       Compute inverse transform
        n_bi = m*n
        Call c06gcf(y,n_bi,ifail)
        Call c06fuf(m,n,x,y,'Subsequent',trigm,trign,work,ifail)
        Call c06gcf(y,n_bi,ifail)

        Write (nout,*)
        Write (nout,*) 'Original sequence as restored by inverse transform'
        Call writxy(nout,x,y,m,n)
        Deallocate (trigm,trign,work,x,y)
      End Do loop

    End Program c06fufe
```

9.2 Program Data

```
C06FUF Example Program Data
3  5                                                  : m, n
     1.000     0.999     0.987     0.936     0.802
     0.000    -0.040    -0.159    -0.352    -0.597
     0.994     0.989     0.963     0.891     0.731
    -0.111    -0.151    -0.268    -0.454    -0.682
     0.903     0.885     0.823     0.694     0.467
    -0.430    -0.466    -0.568    -0.720    -0.884  : x, y
```

9.3 Program Results

```
 C06FUF Example Program Results

 Original data values

 Real      1.000     0.999     0.987     0.936     0.802
 Imag      0.000    -0.040    -0.159    -0.352    -0.597

 Real      0.994     0.989     0.963     0.891     0.731
 Imag     -0.111    -0.151    -0.268    -0.454    -0.682

 Real      0.903     0.885     0.823     0.694     0.467
 Imag     -0.430    -0.466    -0.568    -0.720    -0.884

 Components of discrete Fourier transform

 Real      3.373     0.481     0.251     0.054    -0.419
 Imag     -1.519    -0.091     0.178     0.319     0.415

 Real      0.457     0.055     0.009    -0.022    -0.076
 Imag      0.137     0.032     0.039     0.036     0.004

 Real     -0.170    -0.037    -0.042    -0.038    -0.002
 Imag      0.493     0.058     0.008    -0.025    -0.083

 Original sequence as restored by inverse transform

 Real      1.000     0.999     0.987     0.936     0.802
 Imag     -0.000    -0.040    -0.159    -0.352    -0.597

 Real      0.994     0.989     0.963     0.891     0.731
 Imag     -0.111    -0.151    -0.268    -0.454    -0.682

 Real      0.903     0.885     0.823     0.694     0.467
 Imag     -0.430    -0.466    -0.568    -0.720    -0.884
```