

NAG Toolbox for MATLAB

c06fu

1 Purpose

c06fu computes the two-dimensional discrete Fourier transform of a bivariate sequence of complex data values. This function is designed to be particularly efficient on vector processors.

2 Syntax

```
[x, y, trigm, trign, ifail] = c06fu(m, n, x, y, init, trigm, trign)
```

3 Description

c06fu computes the two-dimensional discrete Fourier transform of a bivariate sequence of complex data values $z_{j_1 j_2}$, where $j_1 = 0, 1, \dots, m-1$, $j_2 = 0, 1, \dots, n-1$.

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

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

To compute the inverse discrete Fourier transform, defined with $\exp(+2\pi i(\dots))$ in the above formula instead of $\exp(-2\pi i(\dots))$, this function should be preceded and followed by calls of c06gc to form the complex conjugates of the data values and the transform.

This function calls c06fr 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 1983b Self-sorting mixed-radix fast Fourier transforms *J. Comput. Phys.* **52** 1–23

5 Parameters

5.1 Compulsory Input Parameters

1: **m – int32 scalar**

m , the number of rows of the arrays **x** and **y**.

Constraint: $m \geq 1$.

2: **n – int32 scalar**

n , the number of columns of the arrays **x** and **y**.

Constraint: $n \geq 1$.

3: **x(m × n) – double array**

4: **y(m × n) – double array**

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 $(0 : m - 1, 0 : n - 1)$, then $x(j_1, j_2)$ and $y(j_1, j_2)$ must contain the real and imaginary parts of $z_{j_1 j_2}$.

5: **init – string**

If the trigonometric coefficients required to compute the transforms are to be calculated by the function and stored in the arrays **trigm** and **trign**, then **init** must be set equal to 'I', (Initial call).

If **init** = 'S' (Subsequent call), the function assumes that trigonometric coefficients for the specified values of m and n are supplied in the arrays **trigm** and **trign**, having been calculated in a previous call to the function.

If **init** = 'R' (Restart), the function assumes that trigonometric coefficients for the specified values of m and n are supplied in the arrays **trigm** and **trign**, but does not check that the function has previously been called. This option allows the **trigm** and **trign** arrays to be stored in an external file, read in and re-used without the need for a call with **init** equal to 'I'. The function carries out a simple test to check that the current values of m and n are compatible with the arrays **trigm** and **trign**.

Constraint: **init** = 'I', 'S' or 'R'.

6: **trigm(2 × m) – double array**

7: **trign(2 × n) – double array**

If **init** = 'S' or 'R', **trigm** and **trign** must contain the required coefficients calculated in a previous call of the function. Otherwise **trigm** and **trign** need not be set.

If $m = n$ the same array may be supplied for **trigm** and **trign**.

5.2 Optional Input Parameters

None.

5.3 Input Parameters Omitted from the MATLAB Interface

work

5.4 Output Parameters

1: **x(m × n) – double array**

2: **y(m × n) – double array**

The real and imaginary parts respectively of the corresponding elements of the computed transform.

3: **trigm(2 × m) – double array**

4: **trign(2 × n) – double array**

trigm and **trign** contain the required coefficients (computed by the function if **init** = 'I').

5: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail = 1

On entry, **m** < 1.

ifail = 2

On entry, **n** < 1.

ifail = 3

On entry, **init** $\neq$ 'T', '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 (sub)program 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 . c06fu is somewhat faster than average if their only prime factors are 2, 3 or 5; and fastest of all if they are powers of 2.

9 Example

```
m = int32(3);
n = int32(5);
x = [1;
     0.994;
     0.903;
     0.999;
     0.989;
     0.885;
     0.987;
     0.963;
     0.823;
     0.9360000000000001;
     0.891;
     0.694;
     0.802;
     0.731;
     0.467];
y = [0;
     -0.111;
```

```

-0.43;
-0.04;
-0.151;
-0.466;
-0.159;
-0.268;
-0.5679999999999999;
-0.352;
-0.454;
-0.72;
-0.597;
-0.6820000000000001;
-0.884];
init = 'Initial';
trigm = zeros(6,1);
trign = zeros(10,1);
[xOut, yOut, trigmOut, trignOut, ifail] = c06fu(m, n, x, y, init, trigm,
trign)

xOut =
    3.3731
    0.4565
   -0.1705
    0.4814
    0.0549
   -0.0375
    0.2507
    0.0093
   -0.0423
    0.0543
   -0.0217
   -0.0377
   -0.4194
   -0.0759
   -0.0022
yOut =
   -1.5187
    0.1368
    0.4927
   -0.0907
    0.0317
    0.0584
    0.1776
    0.0389
    0.0082
    0.3188
    0.0356
   -0.0255
    0.4145
    0.0045
   -0.0829
trigmOut =
     1
     1
     3
     0
     0
     3
trignOut =
     1
     1
     1
     1
     5
     0
     0
     0
     0
     5
ifail =

```

0
