NAG Toolbox for MATLAB

c06fj

1 Purpose

c06fj computes the multi-dimensional discrete Fourier transform of a multivariate sequence of complex data values.

2 Syntax

```
[x, y, ifail] = c06fj(nd, x, y, 'ndim', ndim, 'n', n)
```

3 Description

c06fj computes the multi-dimensional discrete Fourier transform of a multi-dimensional sequence of complex data values $z_{j_1 j_2 \ldots j_m}$, where $j_1 = 0, 1, \ldots, n_1 - 1, \quad j_2 = 0, 1, \ldots, n_2 - 1$, and so on. Thus the individual dimensions are $n_1, n_2, \ldots, n_m$, and the total number of data values is $n = n_1 \times n_2 \times \ldots \times n_m$.

The discrete Fourier transform is here defined (e.g., for $m = 2$) by:

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

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

The extension to higher dimensions is obvious. (Note the scale factor of $\frac{1}{\sqrt{n}}$ 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 function should be preceded and followed by calls of c06gc to form the complex conjugates of the data values and the transform.

The data values must be supplied in a pair of one-dimensional arrays (real and imaginary parts separately), in accordance with the Fortran convention for storing multi-dimensional data (i.e., with the first subscript j_1 varying most rapidly).

This function calls c06fc to perform one-dimensional discrete Fourier transforms by the fast Fourier transform (FFT) algorithm in Brigham 1974, and hence there are some restrictions on the values of the n_i (see Section 5).

4 References

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

5 Parameters

5.1 Compulsory Input Parameters

1: **nd**(**ndim**) – **int32 array**

nd(i) must contain n_i (the dimension of the ith variable), for $i = 1, 2, \ldots, m$. The largest prime factor of each **nd**(i) must not exceed 19, and the total number of prime factors of **nd**(i), counting repetitions, must not exceed 20.

Constraint: **nd**(i) ≥ 1, for $i = 1, 2, \ldots,$ **ndim**.

2: **x(n) – double array**

$\mathbf{x}(1 + j_1 + n_1 j_2 + n_1 n_2 j_3 + \ldots)$ must contain the real part of the complex data value $z_{j_1 j_2 \ldots j_m}$, for $0 \leq j_1 \leq n_1 - 1, 0 \leq j_2 \leq n_2 - 1, \ldots$; i.e., the values are stored in consecutive elements of the array according to the Fortran convention for storing multi-dimensional arrays.

3: **y(n) – double array**

The imaginary parts of the complex data values, stored in the same way as the real parts in the array **x**.

5.2 Optional Input Parameters

1: **ndim – int32 scalar**

Default: The dimension of the array **nd**.

m, the number of dimensions (or variables) in the multivariate data.

Constraint: **ndim** ≥ 1.

2: **n – int32 scalar**

Default: The dimension of the arrays **x**, **y**. (An error is raised if these dimensions are not equal.)

n, the total number of data values.

Constraint: $\mathbf{n} = \mathbf{nd}(1) \times \mathbf{nd}(2) \times \ldots \times \mathbf{nd}(\mathbf{ndim})$.

5.3 Input Parameters Omitted from the MATLAB Interface

work, lwork

5.4 Output Parameters

1: **x(n) – double array**

The real parts of the corresponding elements of the computed transform.

2: **y(n) – double array**

The imaginary parts of the corresponding elements of the computed transform.

3: **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, **ndim** < 1.

ifail $= 2$

On entry, $\mathbf{n} \neq \mathbf{nd}(1) \times \mathbf{nd}(2) \times \ldots \times \mathbf{nd}(\mathbf{ndim})$.

ifail $= 10 \times l + 1$

At least one of the prime factors of $\mathbf{nd}(l)$ is greater than 19.

ifail $= 10 \times l + 2$

$\mathbf{nd}(l)$ has more than 20 prime factors.

ifail $= 10 \times l + 3$

On entry, **nd**$(l) < 1$.

ifail $= 10 \times l + 4$

On entry, **lwork** $< 3 \times$ **nd**(l).

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 $n \times \log n$, but also depends on the factorization of the individual dimensions **nd**(i). c06fj 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

```
nd = [int32(3);
      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];
[xOut, yOut, ifail] = c06fj(nd, x, y)
```

```
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
ifail =
           0
```