NAG Toolbox for MATLAB

c06pf

1 Purpose

c06pf computes the discrete Fourier transform of one variable in a multivariate sequence of complex data values.

2 Syntax

```
[x, ifail] = c06pf(direct, l, nd, x, 'ndim', ndim, 'n', n)
```

3 Description

c06pf computes the discrete Fourier transform of one variable (the lth say) in a multivariate sequence of complex data values $z_{j_1 j_2 \ldots j_m}$, where $j_1 = 0, 1, \ldots, n_1 - 1$, $j_2 = 0, 1, \ldots, n_2 - 1$, 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 function computes n/n_l one-dimensional transforms defined by

$$\hat{z}_{j_1 \ldots k_l \ldots j_m} = \frac{1}{\sqrt{n_l}} \sum_{j_l=0}^{n_l-1} z_{j_1 \ldots j_l \ldots j_m} \times \exp\left(\pm \frac{2\pi i j_l k_l}{n_l}\right),$$

where $k_l = 0, 1, \ldots, n_l - 1$. The plus or minus sign in the argument of the exponential terms in the above definition determine the direction of the transform: a minus sign defines the **forward** direction and a plus sign defines the **backward** direction.

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

A call of c06pf with **direct** = 'F' followed by a call with **direct** = 'B' will restore the original data.

The data values must be supplied in a one-dimensional complex array using column-major storage ordering of multidimensional data (i.e., with the first subscript j_1 varying most rapidly).

This function calls c06pr to perform one-dimensional discrete Fourier transforms. Hence, the function uses a variant of the fast Fourier transform (FFT) algorithm (see Brigham 1974) known as the Stockham self-sorting algorithm, which is described in Temperton 1983b.

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: **direct – string**

If the **F**orward transform as defined in Section 3 is to be computed, then **direct** must be set equal to 'F'.

If the **B**ackward transform is to be computed then **direct** must be set equal to 'B'.

Constraint: **direct** = 'F' or 'B'.

2: **l – int32 scalar**

l, the index of the variable (or dimension) on which the discrete Fourier transform is to be performed.

Constraint: $1 \leq \mathbf{l} \leq$ **ndim**.

3: **nd(ndim) – int32 array**

The elements of **nd** must contain the dimensions of the **ndim** variables; that is, **nd**(i) must contain the dimension of the ith variable.

Constraints:

nd$(i) \geq 1$, for $i = 1, 2, \ldots,$ **ndim**;
nd(**l**) must have less than 31 prime factors (counting repetitions).

4: **x(n) – complex array**

The complex data values. Data values are stored in **x** using column-major ordering for storing multi-dimensional arrays; that is, $z_{j_1 j_2 \cdots j_m}$ is stored in $\mathbf{x}(1 + j_1 + n_1 j_2 + n_1 n_2 j_3 + \cdots)$.

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 array **x**.

n, the total number of data values.

Constraint: **n** must equal the product of the first **ndim** elements of the array **nd**

5.3 Input Parameters Omitted from the MATLAB Interface

work, lwork

5.4 Output Parameters

1: **x(n) – complex array**

The corresponding elements of the computed transform.

2: **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, $\mathbf{l} < 1$ or $\mathbf{l} >$ **ndim**.

ifail = 3

On entry, **direct** ≠ 'F' or 'B'.

ifail = 4

On entry, at least one of the first **ndim** elements of **nd** is less than 1.

ifail = 5

On entry, **n** does not equal the product of the first **ndim** elements of **nd**.

ifail = 6

On entry, **lwork** is too small. The minimum amount of workspace required is returned in **work**(1).

ifail = 7

On entry, **nd**(**l**) has more than 30 prime factors.

ifail = 8

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 $n \times \log n_l$, but also depends on the factorization of n_l. c06pf is somewhat faster than average if the only prime factors of n_l are 2, 3 or 5; and fastest of all if n_l is a power of 2.

9 Example

```
direct = ’F’;
l = int32(2);
nd = [int32(3);
      int32(5)];
x = [complex(1, +0);
      complex(0.994, -0.111);
      complex(0.903, -0.43);
      complex(0.999, -0.04);
      complex(0.989, -0.151);
      complex(0.885, -0.466);
      complex(0.987, -0.159);
      complex(0.963, -0.268);
      complex(0.823, -0.5679999999999999);
      complex(0.9360000000000001, -0.352);
      complex(0.891, -0.454);
      complex(0.694, -0.72);
      complex(0.802, -0.597);
      complex(0.731, -0.6820000000000001);
      complex(0.467, -0.884)];
[xOut, ifail] = c06pf(direct, l, nd, x)
```

```
xOut =
   2.1126 - 0.5134i
   2.0429 - 0.7451i
   1.6869 - 1.3721i
   0.2880 - 0.0003i
   0.2862 - 0.0322i
   0.2596 - 0.1246i
   0.1257 + 0.1298i
   0.1389 + 0.1148i
   0.1695 + 0.0631i
  -0.0030 + 0.1899i
   0.0180 + 0.1892i
   0.0791 + 0.1731i
  -0.2873 + 0.1940i
  -0.2633 + 0.2251i
  -0.1759 + 0.2988i
ifail =
           0
```