NAG Toolbox

nag_sum_fft_complex_multid_1d (c06pf)

1 Purpose

nag_sum_fft_complex_multid_1d (c06pf) computes the discrete Fourier transform of one variable in a multivariate sequence of complex data values.

2 Syntax

```
[x, ifail] = nag_sum_fft_complex_multid_1d(direct, l, nd, x, 'ndim', ndim, 'n', n)
[x, ifail] = c06pf(direct, l, nd, x, 'ndim', ndim, 'n', n)
```

3 Description

nag_sum_fft_complex_multid_1d (c06pf) computes the discrete Fourier transform of one variable (the lth say) in a multivariate sequence of complex data values $z_{j_1 j_2 \cdots j_m}$, 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 $n = n_1 \times n_2 \times \cdots \times n_m$.

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 nag_sum_fft_complex_multid_1d (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 nag_sum_fft_complex_1d_multi_row (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 (1983).

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

5.1 Compulsory Input Parameters

1: **direct** – CHARACTER(1)

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

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

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

2: **l** – INTEGER

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

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

3: **nd**(**ndim**) – INTEGER 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.

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

4: **x**(**n**) – COMPLEX (KIND=nag_wp) array

The complex data values. Data values are stored in **x** using column-major ordering for storing multidimensional 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** – INTEGER

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

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

Constraint: **ndim** ≥ 1.

2: **n** – INTEGER

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 Output Parameters

1: **x**(**n**) – COMPLEX (KIND=nag_wp) array

The corresponding elements of the computed transform.

2: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, **ndim** < 1.

ifail $= 2$

On entry, **l** < 1 or **l** $>$ **ndim**.

ifail $= 3$

On entry, **direct** $\neq$ '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 $= 8$

An unexpected error has occurred in an internal call. Check all function calls and array dimensions. Seek expert help.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

Your licence key may have expired or may not have been installed correctly.

ifail $= -999$

Dynamic memory allocation failed.

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. nag_sum_fft_complex_multid_1d (c06pf) is faster 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

This example reads in a bivariate sequence of complex data values and prints the discrete Fourier transform of the second variable. It then performs an inverse transform and prints the sequence so obtained, which may be compared with the original data values.

9.1 Program Text

```
function c06pf_example

fprintf('c06pf example results\n\n');

% 2D Sequence
nd = [nag_int(3) 5];
x = [1.000+0.000i  0.999-0.040i  0.987-0.159i  0.936-0.352i  0.802-0.597i;
     0.994-0.111i  0.989-0.151i  0.963-0.268i  0.891-0.454i  0.731-0.682i;
     0.903-0.430i  0.885-0.466i  0.823-0.568i  0.694-0.720i  0.467-0.884i];

% Transform along second dimension
direct = 'F';
l = nag_int(2);
[xt, ifail] = c06pf(direct, l, nd, x);
```

```
% Restore x by inverse transform
direct = 'B';
[xr, ifail] = c06pf(direct, l, nd, xt);

disp('Original data:');
disp(x);
disp('Discrete Fourier Transform along second dimension:');
disp(reshape(xt,nd));
fprintf('Original sequence as restored by inverse transform:\n');
disp(reshape(xr,nd));
```

9.2 Program Results

```
     c06pf example results

Original data:
    1.0000 + 0.0000i    0.9990 - 0.0400i    0.9870 - 0.1590i    0.9360 - 0.3520i
0.8020 - 0.5970i
    0.9940 - 0.1110i    0.9890 - 0.1510i    0.9630 - 0.2680i    0.8910 - 0.4540i
0.7310 - 0.6820i
    0.9030 - 0.4300i    0.8850 - 0.4660i    0.8230 - 0.5680i    0.6940 - 0.7200i
0.4670 - 0.8840i

Discrete Fourier Transform along second dimension:
    2.1126 - 0.5134i    0.2880 - 0.0003i    0.1257 + 0.1298i   -0.0030 + 0.1899i
-0.2873 + 0.1940i
    2.0429 - 0.7451i    0.2862 - 0.0322i    0.1389 + 0.1148i    0.0180 + 0.1892i
-0.2633 + 0.2251i
    1.6869 - 1.3721i    0.2596 - 0.1246i    0.1695 + 0.0631i    0.0791 + 0.1731i
-0.1759 + 0.2988i

Original sequence as restored by inverse transform:
    1.0000 + 0.0000i    0.9990 - 0.0400i    0.9870 - 0.1590i    0.9360 - 0.3520i
0.8020 - 0.5970i
    0.9940 - 0.1110i    0.9890 - 0.1510i    0.9630 - 0.2680i    0.8910 - 0.4540i
0.7310 - 0.6820i
    0.9030 - 0.4300i    0.8850 - 0.4660i    0.8230 - 0.5680i    0.6940 - 0.7200i
0.4670 - 0.8840i
```