NAG Toolbox for MATLAB

c06pa

1 Purpose

c06pa calculates the discrete Fourier transform of a sequence of n real data values or of a Hermitian sequence of n complex data values.

2 Syntax

```
[x, ifail] = c06pa(direct, x, n)
```

3 Description

Given a sequence of n real data values x_j, for $j = 0, 1, \ldots, n-1$, c06pa calculates their discrete Fourier transform (in the **Forward** direction) defined by

$$\hat{z}_k = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} x_j \times \exp\left(-i\frac{2\pi jk}{n}\right), \qquad k = 0, 1, \ldots, n-1.$$

The transformed values $\hat{z}_k$ are complex, but they form a Hermitian sequence (i.e., $\hat{z}_{n-k}$ is the complex conjugate of $\hat{z}_k$), so they are completely determined by n real numbers (since $\hat{z}_0$ is real, as is $\hat{z}_{n/2}$ for n even).

Alternatively, given a Hermitian sequence of n complex data values z_j, this function calculates their inverse (**backward**) discrete Fourier transform defined by

$$\hat{x}_k = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} z_j \times \exp\left(i\frac{2\pi jk}{n}\right), \qquad k = 0, 1, \ldots, n-1.$$

The transformed values $\hat{x}_k$ are real.

(Note the scale factor of $\frac{1}{\sqrt{n}}$ in the above definitions.)

A call of c06pa with **direct** = 'F' followed by a call with **direct** = 'B' will restore the original data.

c06pa 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: **x**$(\mathbf{n}+\mathbf{2})$ – **double array**

If **x** is declared with bounds $(0:\mathbf{n}+1)$ in the (sub)program from which c06pa is called, then:

if **direct** is set to 'F', $\mathbf{x}(j)$ must contain x_j, for $j=0,1,\ldots,n-1$;
if **direct** is set to 'B', $\mathbf{x}(2\times k)$ and $\mathbf{x}(2\times k+1)$ must contain the real and imaginary parts respectively of $\hat{z}_k$, for $k=0,1,\ldots,n/2$. (Note that for the sequence $\hat{z}_k$ to be Hermitian, the imaginary part of $\hat{z}_0$, and of $\hat{z}_{n/2}$ for n even, must be zero.)

3: **n** – **int32 scalar**

n, the number of data values. The total number of prime factors of **n**, counting repetitions, must not exceed 30.

5.2 Optional Input Parameters

None.

5.3 Input Parameters Omitted from the MATLAB Interface

work

5.4 Output Parameters

1: **x**$(\mathbf{n}+\mathbf{2})$ – **double array**

if **direct** is set to 'F' and **x** is declared with bounds $(0:\mathbf{n}+1)$ then $\mathbf{x}(2\times k)$ and $\mathbf{x}(2\times k+1)$ will contain the real and imaginary parts respectively of $\hat{z}_k$, for $k=0,1,\ldots,n/2$;
if **direct** is set to 'B' and **x** is declared with bounds $(0:\mathbf{n}+1)$ then $\mathbf{x}(j)$ will contain x_j, for $j=0,1,\ldots,n-1$.

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, $\mathbf{n}\leq 1$.

ifail $=2$

On entry, **direct** $\neq$ 'F' or 'B'.

ifail $=3$

At least one of the prime factors of **n** is greater than 19.

ifail $=4$

On entry, **n** has more than 30 prime factors.

ifail $=5$

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 $n \times \log n$, but also depends on the factorization of n. c06pa is faster if the only prime factors of n are 2, 3 or 5; and fastest of all if n is a power of 2.

9 Example

```
direct = 'F';
x = [0.34907;
     0.5489000000000001;
     0.74776;
     0.94459;
     1.1385;
     1.3285;
     1.5137;
     0;
     0];
n = int32(7);
[xOut, ifail] = c06pa(direct, x, n)
```

```
xOut =
    2.4836
         0
   -0.2660
    0.5309
   -0.2577
    0.2030
   -0.2564
    0.0581
         0
ifail =
           0
```