NAG Toolbox for MATLAB

c06fa

1   Purpose

c06fa calculates the discrete Fourier transform of a sequence of n real data values (using a work array for extra speed).

2   Syntax

```
[x, ifail] = c06fa(x, 'n', n)
```

3   Description

Given a sequence of n real data values x_j, for $j = 0, 1, \ldots, n - 1$, c06fa calculates their discrete Fourier transform defined by

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

(Note the scale factor of $\frac{1}{\sqrt{n}}$ in this definition.)   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 (see also the C06 Chapter Introduction).

To compute the inverse discrete Fourier transform defined by

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

this function should be followed by a call of c06gb to form the complex conjugates of the $\hat{z}_k$.

c06fa uses the fast Fourier transform (FFT) algorithm (see Brigham 1974). There are some restrictions on the value of n (see Section 5).

4   References

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

5   Parameters

5.1   Compulsory Input Parameters

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

   $\mathbf{x}(j + 1)$ must contain x_j, for $j = 0, 1, \ldots, n - 1$.

5.2   Optional Input Parameters

1:    **n – int32 scalar**

   Default: The dimension of the array $\mathbf{x}$.

   n, the number of data values. The largest prime factor of $\mathbf{n}$ must not exceed 19, and the total number of prime factors of $\mathbf{n}$, counting repetitions, must not exceed 20.

   Constraint: $\mathbf{n} > 1$.

5.3   Input Parameters Omitted from the MATLAB Interface

work

5.4   Output Parameters

1:     **x(n) – double array**

The discrete Fourier transform stored in Hermitian form. If the components of the transform $\hat{z}_k$ are written as $a_k + ib_k$, and if **x** is declared with bounds $(0 : \mathbf{n} - 1)$ in the (sub)program from which c06fa is called, then for $0 \le k \le n/2$, a_k is contained in $\mathbf{x}(k)$, and for $1 \le k \le (n - 1)/2$, b_k is contained in $\mathbf{x}(n - k)$. (See also Section missing_entity_c06background12 in the C06 Chapter Introduction and Section 9.)

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 $= 1$

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

ifail $= 2$

n has more than 20 prime factors.

ifail $= 3$

On entry, $\mathbf{n} \le 1$.

ifail $= 4$

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. c06fa 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

```
x = [0.34907;
     0.5489000000000001;
     0.74776;
     0.94459;
     1.1385;
     1.3285;
     1.5137];
[xOut, ifail] = c06fa(x)
```

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