NAG Toolbox

nag_sum_fft_realherm_1d (c06pa)

1 Purpose

nag_sum_fft_realherm_1d (c06pa) calculates the discrete Fourier transform of a sequence of n real data values or of a Hermitian sequence of n complex data values stored in compact form in a double array.

2 Syntax

```
[x, ifail] = nag_sum_fft_realherm_1d(direct, x, n)
[x, ifail] = c06pa(direct, x, n)
```

3 Description

Given a sequence of n real data values x_j, for $j = 0, 1, \ldots, n-1$, nag_sum_fft_realherm_1d (c06pa) calculates their discrete Fourier transform (in the **forward** direction) defined by

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

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

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

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

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

A call of nag_sum_fft_realherm_1d (c06pa) with **direct** $=$ 'F' followed by a call with **direct** $=$ 'B' will restore the original data.

nag_sum_fft_realherm_1d (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 (1983).

The same functionality is available using the forward and backward transform function pair: nag_sum_fft_real_2d (c06pv) and nag_sum_fft_hermitian_2d (c06pw) on setting $\mathbf{n} = 1$. This pair use a different storage solution; real data is stored in a double array, while Hermitian data (the first unconjugated half) is stored in a complex array.

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: $\mathbf{x}(\mathbf{n}+\mathbf{2})$ – REAL (KIND=nag_wp) array

If **x** is declared with bounds $(0:\mathbf{n}+1)$ in the function from which nag_sum_fft_realherm_1d (c06pa) is called, then:

if **direct** = 'F', $\mathbf{x}(j)$ must contain x_j, for $j=0,1,\ldots,n-1$;

if **direct** = 'B', $\mathbf{x}(2\times k)$ and $\mathbf{x}(2\times k+1)$ must contain the real and imaginary parts respectively of z_k, for $k=0,1,\ldots,n/2$. (Note that for the sequence z_k to be Hermitian, the imaginary part of z_0, and of $z_{n/2}$ for n even, must be zero.)

3: **n** – INTEGER

n, the number of data values.

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

5.2 Optional Input Parameters

None.

5.3 Output Parameters

1: $\mathbf{x}(\mathbf{n}+\mathbf{2})$ – REAL (KIND=nag_wp) array

if **direct** = 'F' and **x** is declared with bounds $(0:\mathbf{n}+1)$, $\mathbf{x}(2\times k)$ 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** = 'B' and **x** is declared with bounds $(0:\mathbf{n}+1)$, $\mathbf{x}(j)$ will contain $\hat{x}_j$, for $j=0,1,\ldots,n-1$.

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 $=1$

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

ifail $=2$

$\langle\mathit{value}\rangle$ is an invalid value of **direct**.

ifail $=3$

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

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 $n \times \log(n)$, but also depends on the factorization of n. nag_sum_fft_realherm_1d (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

This example reads in a sequence of real data values and prints their discrete Fourier transform (as computed by nag_sum_fft_realherm_1d (c06pa) with **direct** = 'F'), after expanding it from complex Hermitian form into a full complex sequence. It then performs an inverse transform using nag_sum_fft_realherm_1d (c06pa) with **direct** = 'B', and prints the sequence so obtained alongside the original data values.

9.1 Program Text

```
     function c06pa_example

fprintf('c06pa example results\n\n');

% Real data x
n = nag_int(7);
x = zeros(n+2,1);
x(1:n) = [0.34907;  0.5489;   0.74776;   0.94459;
          1.13850;  1.3285;   1.51370];

% Transform x to get Hermitian data in compact form
direct = 'F';
[xt, ifail] = c06pa(direct, x, n);
zt = nag_herm2complex(n,xt);
disp('Discrete Fourier Transform of x:');
disp(transpose(zt));

% Restore x by inverse transform
direct = 'B';
[xr, ifail] = c06pa(direct, xt, n);

fprintf('Original sequence as restored by inverse transform\n\n');
fprintf('       Original   Restored\n');
for j = 1:n
  fprintf('%3d   %7.4f    %7.4f\n',j, x(j),xr(j));
end

function [z] = nag_herm2complex(n,x);
```

```
  z(1) = complex(x(1));
  for j = 1:floor(double(n)/2) + 1
    z(j) = x(2*j-1) + i*x(2*j);
    z(n-j+2) = x(2*j-1) - i*x(2*j);
  end
```

9.2 Program Results

```
     c06pa example results

Discrete Fourier Transform of x:
   2.4836 + 0.0000i
  -0.2660 + 0.5309i
  -0.2577 + 0.2030i
  -0.2564 + 0.0581i
  -0.2564 - 0.0581i
  -0.2577 - 0.2030i
  -0.2660 - 0.5309i
   2.4836 + 0.0000i

Original sequence as restored by inverse transform

       Original   Restored
  1    0.3491     0.3491
  2    0.5489     0.5489
  3    0.7478     0.7478
  4    0.9446     0.9446
  5    1.1385     1.1385
  6    1.3285     1.3285
  7    1.5137     1.5137
```