NAG Library Function Document
nag_sum_fft_hermitian_2d (c06pwc)

1 Purpose
nag_sum_fft_hermitian_2d (c06pwc) computes the two-dimensional inverse discrete Fourier transform of a bivariate Hermitian sequence of complex data values.

2 Specification

```c
#include <nag.h>
#include <nagc06.h>
void nag_sum_fft_hermitian_2d (Integer m, Integer n, const Complex y[],
  double x[], NagError *fail)
```

3 Description
nag_sum_fft_hermitian_2d (c06pwc) computes the two-dimensional inverse discrete Fourier transform of a bivariate Hermitian sequence of complex data values \( z_{j_1j_2} \), for \( j_1 = 0,1,\ldots,m-1 \) and \( j_2 = 0,1,\ldots,n-1 \).

The discrete Fourier transform is here defined by

\[
\hat{x}_{k_1k_2} = \frac{1}{\sqrt{mn}} \sum_{j_1=0}^{m-1} \sum_{j_2=0}^{n-1} z_{j_1j_2} \exp \left( 2\pi i \left( \frac{j_1k_1}{m} + \frac{j_2k_2}{n} \right) \right),
\]

where \( k_1 = 0,1,\ldots,m-1 \) and \( k_2 = 0,1,\ldots,n-1 \). (Note the scale factor of \( \frac{1}{\sqrt{mn}} \) in this definition.)

Because the input data satisfies conjugate symmetry (i.e., \( z_{k_1,k_2} \) is the complex conjugate of \( z_{(m-k_1),k_2} \), the transformed values \( \hat{x}_{k_1k_2} \) are real.

A call of nag_sum_fft_real_2d (c06pvc) followed by a call of nag_sum_fft_hermitian_2d (c06pwc) will restore the original data.

This function performs multiple one-dimensional discrete Fourier transforms by the fast Fourier transform (FFT) algorithm in Brigham (1974) and Temperton (1983).

4 References


5 Arguments

1: \( m \) – Integer

*Input*

*On entry:* \( m \), the first dimension of the transform.

*Constraint:* \( m \geq 1 \).

2: \( n \) – Integer

*Input*

*On entry:* \( n \), the second dimension of the transform.

*Constraint:* \( n \geq 1 \).
3: \[ y[(m/2 + 1) \times n] \] \text{ – const Complex} \quad \text{Input} \\
\text{On entry: the Hermitian sequence of complex input dataset } z, \text{ where } z_{j_1 j_2} \text{ is stored in } y[j_2 \times (m/2 + 1) + j_1], \text{ for } j_1 = 0, 1, \ldots, m/2 \text{ and } j_2 = 0, 1, \ldots, n - 1. \\

4: \[ x[m \times n] \] \text{ – double} \quad \text{Output} \\
\text{On exit: the real output dataset } \hat{x}, \text{ where } \hat{x}_{k_1 k_2} \text{ is stored in } x[k_2 \times m + k_1], \text{ for } k_1 = 0, 1, \ldots, m - 1 \text{ and } k_2 = 0, 1, \ldots, n - 1. \\

5: \text{ fail – NagError *} \quad \text{Input/Output} \\
\text{The NAG error argument (see Section 3.6 in the Essential Introduction).} \\

6 \quad \text{Error Indicators and Warnings} \\

\textbf{NE_ALLOC_FAIL} \\
\text{Dynamic memory allocation failed.} \\
\text{See Section 3.2.1.2 in the Essential Introduction for further information.} \\

\textbf{NE_BAD_PARAM} \\
\text{On entry, argument } \langle\text{value}\rangle \text{ had an illegal value.} \\

\textbf{NE_INT} \\
\text{On entry, } m = \langle\text{value}\rangle. \\
\text{Constraint: } m \geq 1. \\
\text{On entry, } n = \langle\text{value}\rangle. \\
\text{Constraint: } n \geq 1. \\

\textbf{NE_INTERNAL_ERROR} \\
\text{An internal error has occurred in this function. Check the function call and any array sizes. If the } \text{call is correct then please contact NAG for assistance.} \\
\text{An unexpected error has been triggered by this function. Please contact NAG.} \\
\text{See Section 3.6.6 in the Essential Introduction for further information.} \\

\textbf{NE_NO_LICENCE} \\
\text{Your licence key may have expired or may not have been installed correctly.} \\
\text{See Section 3.6.5 in the Essential Introduction for further information.} \\

7 \quad \text{Accuracy} \\
\text{Some indication of accuracy can be obtained by performing a forward transform using} \\
\text{nag_sum_fft_real_2d (c06pvc)} \text{ and a backward transform using } \text{nag_sum_fft_hermitian_2d (c06pwc)}, \text{ and comparing the results with the original sequence (in exact arithmetic they would be identical).} \\

8 \quad \text{Parallelism and Performance} \\
\text{nag_sum_fft_hermitian_2d (c06pwc)} \text{ is threaded by NAG for parallel execution in multithreaded} \\
\text{implementations of the NAG Library.} \\
\text{nag_sum_fft_hermitian_2d (c06pwc)} \text{ makes calls to BLAS and/or LAPACK routines, which may be} \\
\text{threaded within the vendor library used by this implementation. Consult the documentation for the} \\
\text{vendor library for further information.}
Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users’ Note for your implementation for any additional implementation-specific information.

9 Further Comments

The time taken by nag_sum_fft_hermitian_2d (c06pwc) is approximately proportional to $mn\log(mn)$, but also depends on the factors of $m$ and $n$. nag_sum_fft_hermitian_2d (c06pwc) is fastest if the only prime factors of $m$ and $n$ are 2, 3 and 5, and is particularly slow if $m$ or $n$ is a large prime, or has large prime factors.

Workspace is internally allocated by nag_sum_fft_hermitian_2d (c06pwc). The total size of these arrays is approximately proportional to $mn$.

10 Example

See Section 10 in nag_sum_fft_real_2d (c06pvc).