NAG Library Function Document

nag_conjugate_hermitian (c06gbc)

1 Purpose

nag_conjugate_hermitian (c06gbc) forms the complex conjugate of a Hermitian sequence of n data values.

2 Specification

```
#include <nag.h>
#include <nagc06.h>
void nag_conjugate_hermitian (Integer n, double x[], NagError *fail)
```

3 Description

This is a utility function for use in conjunction with nag_fft_real (c06eac) and nag_fft_hermitian (c06ebc), to calculate inverse discrete Fourier transforms.

4 References

None.

5 Arguments

1: **n** – Integer *Input*

On entry: the number of data values, n.

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

2: **x**[**n**] – double *Input/Output*

On entry: if the data values z_j are written as $x_j + iy_j$, then for $0 \leq j \leq n/2$, $\mathbf{x}[j]$ must contain $x_j (= x_{n-j})$, while for $n/2 < j \leq n-1$, $\mathbf{x}[j]$ must contain $-y_j (= y_{n-j})$. In other words, **x** must contain the Hermitian sequence in Hermitian form.

On exit: the imaginary parts y_j are negated. The real parts x_j are not referenced.

3: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_INT_ARG_LT

On entry, $\mathbf{n} = \langle$*value*$\rangle$.
Constraint: $\mathbf{n} \geq 1$.

7 Accuracy

Exact.

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken is negligible.

10 Example

This program reads in a sequence of real data values, calls nag_fft_real (c06eac) followed by nag_conjugate_hermitian (c06gbc) to compute their inverse discrete Fourier transform, and prints this after expanding it from Hermitian form into a full complex sequence.

10.1 Program Text

```
/* nag_conjugate_hermitian (c06gbc) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 1, 1990.
 * Mark 8 revised, 2004.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagc06.h>

int main(void)
{
  Integer  exit_status = 0, j, n, n2, nj;
  NagError fail;
  double   *a = 0, *b = 0, *x = 0;

  INIT_FAIL(fail);

  printf("nag_conjugate_hermitian (c06gbc) Example Program Results\n");
  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif
#ifdef _WIN32
  while (scanf_s("%"NAG_IFMT"", &n) != EOF)
#else
  while (scanf("%"NAG_IFMT"", &n) != EOF)
#endif
    {
      if (n > 1)
        {
          if (!(a = NAG_ALLOC(n, double)) ||
              !(b = NAG_ALLOC(n, double)) ||
              !(x = NAG_ALLOC(n, double)))
            {
              printf("Allocation failure\n");
              exit_status = -1;
              goto END;
            }
        }
      else
        {
          printf("Invalid n.n");
          exit_status = 1;
          return exit_status;
        }
      for (j = 0; j < n; j++)
#ifdef _WIN32
        scanf_s("%lf", &x[j]);
#else
        scanf("%lf", &x[j]);
#endif
      /* Calculate the Fourier transform of data */
      /* nag_fft_real (c06eac).
       * Single one-dimensional real discrete Fourier transform
       */
      nag_fft_real(n, x, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf("Error from nag_fft_real (c06eac).\n%s\n",
                  fail.message);
          exit_status = 1;
          goto END;
        }

      /* Calculate conjugates of Hermitian result to */
      /* give inverse tranform */
      /* nag_conjugate_hermitian (c06gbc).
       * Complex conjugate of Hermitian sequence
       */
      nag_conjugate_hermitian(n, x, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf("Error from nag_conjugate_hermitian (c06gbc).\n%s\n",
                  fail.message);
          exit_status = 1;
          goto END;
        }

      /* Expand conjugated Hermitian sequence to full complex */
      a[0] = x[0];
      b[0] = 0.0;
      n2 = (n-1)/2;
      for (j = 1; j <= n2; j++)
        {
          nj = n - j;
          a[j] = x[j];
          a[nj] = x[j];
          b[j] = x[nj];
          b[nj] = -x[nj];
        }
      if (n % 2 == 0)
        {
          a[n2+1] = x[n2+1];
          b[n2+1] = 0.0;
        }
      printf("\nComponents of inverse discrete Fourier transform\n");
      printf("\n          Real       Imag \n\n");
      for (j = 0; j < n; j++)
        printf("%3"NAG_IFMT" %10.5f %10.5f\n", j, a[j], b[j]);
 END:
      NAG_FREE(a);
      NAG_FREE(b);
      NAG_FREE(x);
    }
  return exit_status;
}
```

10.2 Program Data

```
nag_conjugate_hermitian (c06gbc) Example Program Data
    7
  0.34907
  0.54890
  0.74776
  0.94459
  1.13850
  1.32850
  1.51370
```

10.3 Program Results

```
nag_conjugate_hermitian (c06gbc) Example Program Results

Components of inverse discrete Fourier transform

         Real       Imag

  0    2.48361    0.00000
  1   -0.26599   -0.53090
  2   -0.25768   -0.20298
  3   -0.25636   -0.05806
  4   -0.25636    0.05806
  5   -0.25768    0.20298
  6   -0.26599    0.53090
```