NAG Library Function Document

nag_convolution_real (c06ekc)

1 Purpose

nag_convolution_real (c06ekc) calculates the circular convolution or correlation of two real vectors of period n.

2 Specification

```
#include <nag.h>
#include <nagc06.h>
```

```
void nag_convolution_real (Nag_VectorOp operation, Integer n, double x[],
     double y[], NagError *fail)
```

3 Description

nag_convolution_real (c06ekc) computes:

if **operation** $=$ Nag_Convolution, the discrete convolution of x and y, defined by

$$z_k = \sum_{j=0}^{n-1} x_j y_{k-j} = \sum_{j=0}^{n-1} x_{k-j} y_j;$$

if **operation** $=$ Nag_Correlation, the discrete correlation of x and y defined by

$$w_k = \sum_{j=0}^{n-1} x_j y_{k+j}.$$

Here x and y are real vectors, assumed to be periodic, with period n, i.e., $x_j = x_{j\pm n} = x_{j\pm 2n} = \ldots$; z and w are then also periodic with period n.

Note: this usage of the terms 'convolution' and 'correlation' is taken from Brigham (1974). The term 'convolution' is sometimes used to denote both these computations.

If $\hat{x}$, $\hat{y}$, $\hat{z}$ and $\hat{w}$ are the discrete Fourier transforms of these sequences, i.e.,

$$\hat{x}_k = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} x_j \exp\left(-i\frac{2\pi jk}{n}\right), \text{ etc.,}$$

then $\hat{z}_k = \sqrt{n}\hat{x}_k\hat{y}_k$ and $\hat{w}_k = \sqrt{n}\bar{\hat{x}}_k\hat{y}_k$ (the bar denoting complex conjugate).

This function calls the same auxiliary functions as nag_fft_real (c06eac) and nag_fft_hermitian (c06ebc) to compute discrete Fourier transforms, and there are some restrictions on the value of n.

4 References

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

5 Arguments

1: **operation** – Nag_VectorOp *Input*

On entry: the computation to be performed.

operation $=$ Nag_Convolution

$z_k = \sum_{j=0}^{n-1} x_j y_{k-j}$.

operation = Nag_Correlation

$w_k = \sum_{j=0}^{n-1} x_j y_{k+j}$.

Constraint: **operation** = Nag_Convolution or Nag_Correlation.

2: **n** – Integer *Input*

On entry: n, the number of values, in one period of the vectors **x** and **y**.

Constraints:

n > 1;
The largest prime factor of **n** must not exceed 19, and the total number of prime factors of **n**, counting repetitions, must not exceed 20.

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

On entry: the elements of one period of the vector x. **x**[j] must contain x_j, for $j = 0, 1, \ldots, n-1$.

On exit: the corresponding elements of the discrete convolution or correlation.

4: **y**[**n**] – double *Input/Output*

On entry: the elements of one period of the vector y. **y**[j] must contain y_j, for $j = 0, 1, \ldots, n-1$.

On exit: the discrete Fourier transform of the convolution or correlation returned in the array **x**; the transform is stored in Hermitian form, exactly as described in the document nag_fft_real (c06eac).

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

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

6 Error Indicators and Warnings

NE_BAD_PARAM

On entry, argument **operation** had an illegal value.

NE_C06_FACTOR_GT

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

NE_C06_TOO_MANY_FACTORS

n has more than 20 prime factors.

NE_INT_ARG_LE

On entry, **n** = $\langle$*value*$\rangle$.
Constraint: **n** > 1.

7 Accuracy

The results should be accurate to within a small multiple of the ***machine precision***.

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken is approximately proportional to $n\log(n)$, but also depends on the factorization of n. nag_convolution_real (c06ekc) is somewhat faster than average if the only prime factors of n are 2, 3 or 5; and fastest of all if n is a power of 2.

On the other hand, nag_convolution_real (c06ekc) is particularly slow if n has several unpaired prime factors, i.e., if the 'square-free' part of n has several factors.

10 Example

This example reads in the elements of one period of two real vectors x and y and prints their discrete convolution and correlation (as computed by nag_convolution_real (c06ekc)). In realistic computations the number of data values would be much larger.

10.1 Program Text

```
/* nag_convolution_real (c06ekc) 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;
  NagError fail;
  double   *xa = 0, *xb = 0, *ya = 0, *yb = 0;

  INIT_FAIL(fail);

  printf("nag_convolution_real (c06ekc) 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 (!(xa = NAG_ALLOC(n, double)) ||
              !(xb = NAG_ALLOC(n, double)) ||
              !(ya = NAG_ALLOC(n, double)) ||
              !(yb = 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%lf", &xa[j], &ya[j]);
#else
          scanf("%lf%lf", &xa[j], &ya[j]);
#endif
          xb[j] = xa[j];
          yb[j] = ya[j];
        }
      /* nag_convolution_real (c06ekc).
       * Circular convolution or correlation of two real vectors
       */
      nag_convolution_real(Nag_Convolution, n, xa, ya, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf("Error from nag_convolution_real (c06ekc).\n%s\n",
                  fail.message);
          exit_status = 1;
          goto END;
        }
      /* nag_convolution_real (c06ekc), see above. */
      nag_convolution_real(Nag_Correlation, n, xb, yb, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf("Error from nag_convolution_real (c06ekc).\n%s\n",
                  fail.message);
          exit_status = 1;
          goto END;
        }
      printf("\n          Convolution   Correlation\n\n");
      for (j = 0; j < n; ++j)
        printf("%5"NAG_IFMT" %13.5f %13.5f\n", j, xa[j], xb[j]);
 END:
      NAG_FREE(xa);
      NAG_FREE(xb);
      NAG_FREE(ya);
      NAG_FREE(yb);
    }
  return exit_status;
}
```

10.2 Program Data

```
nag_convolution_real (c06ekc) Example Program Data
    9
      1.00      0.50
      1.00      0.50
      1.00      0.50
      1.00      0.50
      1.00      0.00
      0.00      0.00
      0.00      0.00
      0.00      0.00
      0.00      0.00
```

10.3 Program Results

```
nag_convolution_real (c06ekc) Example Program Results

          Convolution   Correlation

    0       0.50000       2.00000
    1       1.00000       1.50000
    2       1.50000       1.00000
    3       2.00000       0.50000
```

```
    4       2.00000       0.00000
    5       1.50000       0.50000
    6       1.00000       1.00000
    7       0.50000       1.50000
    8       0.00000       2.00000
```