

# NAG Library Function Document

## nag\_dwt (c09cac)

### 1 Purpose

nag\_dwt (c09cac) computes the one-dimensional discrete wavelet transform (DWT) at a single level. The initialization function nag\_wfilt (c09aac) must be called first to set up the DWT options.

### 2 Specification

```
#include <nag.h>
#include <nagc09.h>

void nag_dwt (Integer n, const double x[], Integer lenc, double ca[],
             double cd[], Integer icomm[], NagError *fail)
```

### 3 Description

nag\_dwt (c09cac) computes the one-dimensional DWT of a given input data array,  $x_i$ , for  $i = 1, 2, \dots, n$ , at a single level. For a chosen wavelet filter pair, the output coefficients are obtained by applying convolution and downsampling by two to the input,  $x$ . The approximation (or smooth) coefficients,  $C_a$ , are produced by the low pass filter and the detail coefficients,  $C_d$ , by the high pass filter. To reduce distortion effects at the ends of the data array, several end extension methods are commonly used. Those provided are: periodic or circular convolution end extension, half-point symmetric end extension, whole-point symmetric end extension or zero end extension. The number  $n_c$ , of coefficients  $C_a$  or  $C_d$  is returned by the initialization function nag\_wfilt (c09aac).

### 4 References

Daubechies I (1992) *Ten Lectures on Wavelets* SIAM, Philadelphia

### 5 Arguments

- 1: **n** – Integer *Input*  
*On entry:* the number of elements,  $n$ , in the data array  $x$ .  
*Constraint:* this must be the same as the value **n** passed to the initialization function nag\_wfilt (c09aac).
- 2: **x[n]** – const double *Input*  
*On entry:* **x** contains the input dataset  $x_i$ , for  $i = 1, 2, \dots, n$ .
- 3: **lenc** – Integer *Input*  
*On entry:* the dimension of the arrays **ca** and **cd**. This must be at least the number,  $n_c$ , of approximation coefficients,  $C_a$ , and detail coefficients,  $C_d$ , of the discrete wavelet transform as returned in **nwc** by the call to the initialization function nag\_wfilt (c09aac).  
*Constraint:* **lenc**  $\geq n_c$ , where  $n_c$  is the value returned in **nwc** by the call to the initialization function nag\_wfilt (c09aac).
- 4: **ca[lenc]** – double *Output*  
*On exit:* **ca**[ $i - 1$ ] contains the  $i$ th approximation coefficient,  $C_a(i)$ , for  $i = 1, 2, \dots, n_c$ .

- 5: **cd[lenc]** – double *Output*  
*On exit:* **cd**[ $i - 1$ ] contains the  $i$ th detail coefficient,  $C_d(i)$ , for  $i = 1, 2, \dots, n_c$ .
- 6: **icomm[100]** – Integer *Communication Array*  
*On entry:* contains details of the discrete wavelet transform and the problem dimension as setup in the call to the initialization function nag\_wfilt (c09aac).  
*On exit:* contains additional information on the computed transform.
- 7: **fail** – NagError \* *Input/Output*  
The NAG error argument (see Section 3.6 in the Essential Introduction).

## 6 Error Indicators and Warnings

### NE\_ALLOC\_FAIL

Dynamic memory allocation failed.  
See Section 3.2.1.2 in the Essential Introduction for further information.

### NE\_ARRAY\_DIM\_LEN

On entry, array dimension **lenc** not large enough: **lenc** =  $\langle value \rangle$  but must be at least  $\langle value \rangle$ .

### NE\_BAD\_PARAM

On entry, argument  $\langle value \rangle$  had an illegal value.

### NE\_INITIALIZATION

Either the initialization function has not been called first or array **icomm** has been corrupted.  
Either the initialization function was called with **wtrans** = Nag\_MultiLevel or array **icomm** has been corrupted.  
On entry, **n** is inconsistent with the value passed to the initialization function: **n** =  $\langle value \rangle$ , **n** should be  $\langle value \rangle$ .

### NE\_INTERNAL\_ERROR

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.  
An unexpected error has been triggered by this function. Please contact NAG.  
See Section 3.6.6 in the Essential Introduction for further information.

### NE\_NO\_LICENCE

Your licence key may have expired or may not have been installed correctly.  
See Section 3.6.5 in the Essential Introduction for further information.

## 7 Accuracy

The accuracy of the wavelet transform depends only on the floating-point operations used in the convolution and downsampling and should thus be close to *machine precision*.

## 8 Parallelism and Performance

Not applicable.

## 9 Further Comments

None.

## 10 Example

This example computes the one-dimensional discrete wavelet decomposition for 8 values using the Daubechies wavelet, `wavnam = Nag_Daubechies4`, with zero end extension.

### 10.1 Program Text

```

/* nag_dwt (c09cac) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 9, 2009.
 */
/* Pre-processor includes */
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagc09.h>

int main(void)
{
    /* Constants */
    Integer      licomm = 100;
    /*Integer scalar and array declarations */
    Integer      exit_status = 0;
    Integer      i, n, nf, nwc, nwl, ny;
    Integer      *icomm = 0;
    NagError     fail;
    Nag_Wavelet  wavnamenum;
    Nag_WaveletMode modenum;
    /*Double scalar and array declarations */
    double       *ca = 0, *cd = 0, *x = 0, *y = 0;
    /*Character scalar and array declarations */
    char         mode[24], wavnam[20];

    INIT_FAIL(fail);

    printf("nag_dwt (c09cac) Example Program Results\n\n");
    fflush(stdout);

    /*      Skip heading in data file*/
#ifdef _WIN32
    scanf_s("%*[\n] ");
#else
    scanf("%*[\n] ");
#endif
    /*      Read n*/
#ifdef _WIN32
    scanf_s("%"NAG_IFMT"%*[\n] ", &n);
#else
    scanf("%"NAG_IFMT"%*[\n] ", &n);
#endif
    if (!(x = NAG_ALLOC(n, double)) ||
        !(y = NAG_ALLOC(n, double)) ||
        !(icomm = NAG_ALLOC(licomm, Integer)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }
    /*      Read wavnam, mode*/
#ifdef _WIN32

```

```

scanf_s("%19s%23s%[^\\n] ", wavnam, _countof(wavnam), mode, _countof(mode));
#else
scanf("%19s%23s%[^\\n] ", wavnam, mode);
#endif
/*
 * nag_enum_name_to_value (x04nac).
 * Converts NAG enum member name to value
 */
wavnamenum = (Nag_Wavelet) nag_enum_name_to_value(wavnam);
modenum = (Nag_WaveletMode) nag_enum_name_to_value(mode);
if (n >= 2)
{
printf("DWT :: \\n");
printf("      Wavelet   :%16s\\n", wavnam);
printf("      End mode  :%16s\\n", mode);
printf("      N          :%16"NAG_IFMT"\\n\\n", n);
/*      Read array*/
printf("%s\\n", "Input Data          X :");
for (i = 0; i < n; i++)
{
#ifdef _WIN32
scanf_s("%lf ", &x[i]);
#else
scanf("%lf ", &x[i]);
#endif
printf("%8.4f%s", x[i], (i+1)%8?" ":"\\n");
}
printf("\\n");
/*
 * nag_wfilt (c09aac)
 * Wavelet filter query
 */
nag_wfilt(wavnamenum, Nag_SingleLevel, modenum, n, &nwl, &nf, &nwc,
          icomm, &fail);
if (fail.code != NE_NOERROR)
{
printf("Error from nag_wfilt (c09aac).\\n%s\\n", fail.message);
exit_status = 1;
goto END;
}
if (!(ca = NAG_ALLOC(nwc, double)) ||
    !(cd = NAG_ALLOC(nwc, double)))
{
printf("Allocation failure\\n");
exit_status = -1;
goto END;
}
/*
 * nag_dwt (c09cac)
 * one-dimensional discrete wavelet transform (dwt)
 */
nag_dwt(n, x, nwc, ca, cd, icomm, &fail);
if (fail.code != NE_NOERROR)
{
printf("Error from nag_dwt (c09cac).\\n%s\\n", fail.message);
exit_status = 1;
goto END;
}
printf("Approximation coefficients CA : \\n");
for (i = 0; i < nwc; i++)
printf("%8.4f%s", ca[i]*sqrt(2.00e0), (i+1)%8?" ":"\\n");
printf("\\n");
printf("Detail coefficients          CD : \\n");
for (i = 0; i < nwc; i++)
printf("%8.4f%s", cd[i]*sqrt(2.00e0), (i+1)%8?" ":"\\n");
printf("\\n\\n");
if (modenum == Nag_Periodic)
{
ny = 2*nwc;
}
else

```

```

    {
        ny = n;
    }
    /*
    * nag_idwt (c09cac)
    * one-dimensional inverse discrete wavelet transform (IDWT)
    */
    nag_idwt(nwc, ca, cd, n, y, icomm, &fail);
    if (fail.code != NE_NOERROR)
    {
        printf("Error from nag_idwt (c09cac).\n%s\n", fail.message);
        exit_status = 1;
        goto END;
    }
    printf("Reconstruction          Y : \n");
    for (i = 0; i < ny; i++)
        printf("%8.4f%s", y[i], (i+1)%8?" ":"\n");
}

END:
    NAG_FREE(ca);
    NAG_FREE(cd);
    NAG_FREE(x);
    NAG_FREE(y);
    NAG_FREE(icomm);

    return exit_status;
}

```

## 10.2 Program Data

```

nag_dwt (c09cac) Example Program Data
8                               : n
Nag_Daubechies4 Nag_ZeroPadded : wavnam, mode
1.0
3.0
5.0
7.0
6.0
4.0
5.0
2.0                               : X(1:n)

```

## 10.3 Program Results

```

nag_dwt (c09cac) Example Program Results

DWT ::
    Wavelet : Nag_Daubechies4
    End mode : Nag_ZeroPadded
    N       : 8

Input Data          X :
    1.0000   3.0000   5.0000   7.0000   6.0000   4.0000   5.0000   2.0000

Approximation coefficients CA :
    0.0015  -0.0060  -0.0247   6.3326  12.6652  10.3805   3.6509
Detail coefficients   CD :
    0.0335   0.0579  -0.8437   2.5120  -1.0630   0.4712  -0.1679

Reconstruction          Y :
    1.0000   3.0000   5.0000   7.0000   6.0000   4.0000   5.0000   2.0000

```

---