NAG Library Function Document

nag_dwt_3d (c09fac)

1 Purpose

nag_dwt_3d (c09fac) computes the three-dimensional discrete wavelet transform (DWT) at a single level. The initialization function nag_wfilt_3d (c09acc) must be called first to set up the DWT options.

2 Specification

```
#include <nag.h>
#include <nagc09.h>
void nag_dwt_3d (Integer m, Integer n, Integer fr, const double a[],
     Integer lda, Integer sda, Integer lenc, double c[], Integer icomm[],
     NagError *fail)
```

3 Description

nag_dwt_3d (c09fac) computes the three-dimensional DWT of some given three-dimensional input data, considered as a number of two-dimensional frames, 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 data, A, first over columns, next over rows and finally across frames. The three-dimensional approximation coefficients are produced by the low pass filter over columns, rows and frames. In addition there are 7 sets of three-dimensional detail coefficients, each corresponding to a different order of low pass and high pass filters (see the c09 Chapter Introduction). All coefficients are packed into a single array. 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 and zero end extension. The total number, n_{ct}, of coefficients computed is returned by the initialization function nag_wfilt_3d (c09acc).

4 References

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

5 Arguments

1: **m** – Integer *Input*

On entry: the number of rows of each two-dimensional frame.

Constraint: this must be the same as the value **m** passed to the initialization function nag_wfilt_3d (c09acc).

2: **n** – Integer *Input*

On entry: the number of columns of each two-dimensional frame.

Constraint: this must be the same as the value **n** passed to the initialization function nag_wfilt_3d (c09acc).

3: **fr** – Integer *Input*

On entry: the number of two-dimensional frames.

Constraint: this must be the same as the value **fr** passed to the initialization function nag_wfilt_3d (c09acc).

4: **a**[dim] – const double *Input*

Note: the dimension, *dim*, of the array **a** must be at least **lda** × **sda** × **fr**.

On entry: the m by n by fr three-dimensional input data A, where A_{ijk} is stored in **a**[$(k-1) \times$ **lda** × **sda** $+ (j-1) \times$ **lda** $+ i - 1$].

5: **lda** – Integer *Input*

On entry: the stride separating row elements of each of the sets of frame coefficients in the three-dimensional data stored in **a**.

Constraint: **lda** $\geq$ **m**.

6: **sda** – Integer *Input*

On entry: the stride separating corresponding coefficients of consecutive frames in the three-dimensional data stored in **a**.

Constraint: **sda** $\geq$ **n**.

7: **lenc** – Integer *Input*

On entry: the dimension of the array **c**.

Constraint: **lenc** $\geq n_{\mathrm{ct}}$, where n_{ct} is the total number of wavelet coefficients, as returned by nag_wfilt_3d (c09acc).

8: **c**[**lenc**] – double *Output*

On exit: the coefficients of the discrete wavelet transform. If you need to access or modify the approximation coefficients or any specific set of detail coefficients then the use of nag_wav_3d_coeff_ext (c09fyc) or nag_wav_3d_coeff_ins (c09fzc) is recommended. For completeness the following description provides details of precisely how the coefficients are stored in **c** but this information should only be required in rare cases.

The 8 sets of coefficients are stored in the following order: approximation coefficients (LLL) first, followed by 7 sets of detail coefficients: LLH, LHL, LHH, HLL, HLH, HHL, HHH, where L indicates the low pass filter, and H the high pass filter being applied to, respectively, the columns of length **m**, the rows of length **n** and then the frames of length **fr**. Note that for computational efficiency reasons each set of coefficients is stored in the order $n_{\mathrm{cfr}} \times n_{\mathrm{cm}} \times n_{\mathrm{cn}}$ (see output arguments **nwcfr**, **nwct** and **nwcn** in nag_wfilt_3d (c09acc)). See Section 10 for details of how to access each set of coefficients in order to perform extraction from **c** following a call to this function, or insertion into **c** before a call to the three-dimensional inverse function nag_idwt_3d (c09fbc).

9: **icomm**[**260**] – 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_3d (c09acc).

On exit: contains additional information on the computed transform.

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

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_INITIALIZATION

Either the communication array **icomm** has been corrupted or there has not been a prior call to the initialization function nag_wfilt_3d (c09acc).

The initialization function was called with **wtrans** = Nag_MultiLevel.

NE_INT

On entry, **fr** = ⟨*value*⟩.
Constraint: **fr** = ⟨*value*⟩, the value of **fr** on initialization (see nag_wfilt_3d (c09acc)).

On entry, **m** = ⟨*value*⟩.
Constraint: **m** = ⟨*value*⟩, the value of **m** on initialization (see nag_wfilt_3d (c09acc)).

On entry, **n** = ⟨*value*⟩.
Constraint: **n** = ⟨*value*⟩, the value of **n** on initialization (see nag_wfilt_3d (c09acc)).

NE_INT_2

On entry, **lda** = ⟨*value*⟩ and **m** = ⟨*value*⟩.
Constraint: **lda** $\geq$ **m**.

On entry, **lenc** = ⟨*value*⟩ and n_{ct} = ⟨*value*⟩.
Constraint: **lenc** $\geq n_{\mathrm{ct}}$, where n_{ct} is the number of DWT coefficients returned by nag_wfilt_3d (c09acc) in argument **nwct**.

On entry, **sda** = ⟨*value*⟩ and **n** = ⟨*value*⟩.
Constraint: **sda** $\geq$ **n**.

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 2.7.6 in How to Use the NAG Library and its Documentation for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 2.7.5 in How to Use the NAG Library and its Documentation 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

nag_dwt_3d (c09fac) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

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

None.

10 Example

This example computes the three-dimensional discrete wavelet decomposition for $5 \times 4 \times 3$ input data using the Haar wavelet, **wavnam** = Nag_Haar, with half point end extension, prints the wavelet coefficients and then reconstructs the original data using nag_idwt_3d (c09fbc). This example also demonstrates in general how to access any set of coefficients following a single level transform.

10.1 Program Text

```
/* nag_dwt_3d (c09fac) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */
#include <stdio.h>
#include <math.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagc09.h>

#define A(I,J,K) a[I-1 + (J-1)* lda + (K-1)* lda * sda]
#define B(I,J,K) b[I-1 + (J-1)* ldb + (K-1)* ldb * sdb]
#define D(I,J,K) d[I-1 + (J-1)* nwcm + (K-1)* nwcm * nwcn]

int main(void)
{
  /* Scalars */
  Integer exit_status = 0, zero = 0;
  Integer cindex, i, j, k, lda, ldb, lenc;
  Integer m, n, fr, nf, nwcfr, nwcm, nwcn, nwct, nwl, sda, sdb;
  /* Arrays */
  char mode[25], wavnam[25];
  double *a = 0, *b = 0, *c = 0, *d = 0;
  Integer icomm[260];
  /* Nag Types */
  Nag_Wavelet wavnamenum;
  Nag_WaveletMode modenum;
  Nag_MatrixType matrix = Nag_GeneralMatrix;
  Nag_OrderType order = Nag_ColMajor;
  Nag_DiagType diag = Nag_NonUnitDiag;
  NagError fail;

  INIT_FAIL(fail);

  printf("nag_dwt_3d (c09fac) Example Program Results\n\n");
  fflush(stdout);

  /* Skip heading in data file and read problem parameters */
#ifdef _WIN32
  scanf_s("%*[^\n] %" NAG_IFMT "%" NAG_IFMT "%" NAG_IFMT "%*[^\n]", &m, &n,
          &fr);
#else
  scanf("%*[^\n] %" NAG_IFMT "%" NAG_IFMT "%" NAG_IFMT "%*[^\n]", &m, &n,
        &fr);
#endif
  lda = m;
```

```
  ldb = m;
  sda = n;
  sdb = n;
#ifdef _WIN32
  scanf_s("%24s%24s%*[^\n]\n", wavnam, (unsigned)_countof(wavnam), mode,
          (unsigned)_countof(mode));
#else
  scanf("%24s%24s%*[^\n]\n", wavnam, mode);
#endif

  if (!(a = NAG_ALLOC((lda) * (sda) * (fr), double)) ||
      !(b = NAG_ALLOC((ldb) * (sdb) * (fr), double)))
  {
    printf("Allocation failure\n");
    exit_status = 1;
    goto END;
  }

  printf("Parameters read from file :: \n");
  printf("DWT :: Wavelet  : %s\n", wavnam);
  printf("       End mode : %s\n", mode);
  printf("       m  : %4" NAG_IFMT "\n", m);
  printf("       n  : %4" NAG_IFMT "\n", n);
  printf("       fr : %4" NAG_IFMT "\n\n", fr);

  /* 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);

  /* Read data array */
  for (k = 1; k <= fr; k++) {
    for (i = 1; i <= m; i++) {
#ifdef _WIN32
      for (j = 1; j <= n; j++)
        scanf_s("%lf", &A(i, j, k));
#else
      for (j = 1; j <= n; j++)
        scanf("%lf", &A(i, j, k));
#endif
    }
#ifdef _WIN32
    scanf_s("%*[^\n] ");
#else
    scanf("%*[^\n] ");
#endif
  }

  /* Print out the input data */
  printf("Input Data :\n");
  fflush(stdout);
  for (k = 1; k <= fr; k++) {
    /* nag_gen_real_mat_print_comp (x04cbc).
     * Prints out a matrix.
     */
    nag_gen_real_mat_print_comp(order, matrix, diag, m, n, &A(1, 1, k), lda,
                                "%8.4f", " ", Nag_NoLabels, 0, Nag_NoLabels,
                                0, 80, 0, 0, &fail);
    if (fail.code != NE_NOERROR) {
      printf("Error from nag_gen_real_mat_print_comp (x04cbc).\n%s\n",
             fail.message);
      exit_status = 2;
      goto END;
    }
    printf("\n");
    fflush(stdout);
  }

  /* nag_wfilt_3d (c09acc).
   * Three-dimensional wavelet filter initialization
```

```
 */
nag_wfilt_3d(wavnamenum, Nag_SingleLevel, modenum, m, n, fr, &nwl, &nf,
             &nwct, &nwcn, &nwcfr, icomm, &fail);
if (fail.code != NE_NOERROR) {
  printf("Error from nag_wfilt_3d (c09acc).\n%s\n", fail.message);
  exit_status = 3;
  goto END;
}

/* Calculate the number of wavelet coefficients in
 * the first dimension, nwcm.
 */
nwcm = nwct / (8 * nwcn * nwcfr);
lenc = nwct;

/* Allocate space for the coefficients array, C */
if (!(c = NAG_ALLOC((lenc), double)))
{
  printf("Allocation failure\n");
  exit_status = 4;
  goto END;
}

/* nag_dwt_3d (c09fac).
 * Three-dimensional discrete wavelet transform
 */
nag_dwt_3d(m, n, fr, a, lda, sda, lenc, c, icomm, &fail);
if (fail.code != NE_NOERROR) {
  printf("Error from nag_dwt_3d (c09fac).\n%s\n", fail.message);
  exit_status = 5;
  goto END;
}

/* Allocate space for extraction of coefficients of a single type */
if (!(d = NAG_ALLOC((nwcm) * (nwcn) * (nwcfr), double)))
{
  printf("Allocation failure\n");
  exit_status = 6;
  goto END;
}

for (cindex = 0; cindex <= 7; cindex++) {
  /* Use the extraction routine c09fyc to retrieve the required
   * coefficients.
   */

  /* nag_wav_3d_coeff_ext (c09fyc).
   * Extract the nominated coefficients.
   */
  nag_wav_3d_coeff_ext(zero, cindex, lenc, c, d, nwcm, nwcn, icomm, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_wav_3d_coeff_ext (c09fyc).\n%s\n", fail.message);
    exit_status = 7;
    goto END;
  }

  /* Print out the extracted coefficients */
  switch (cindex) {
  case 0:
    printf("Approximation coefficients (LLL)\n");
    break;
  case 1:
    printf("Detail coefficients (LLH)\n");
    break;
  case 2:
    printf("Detail coefficients (LHL)\n");
    break;
  case 3:
    printf("Detail coefficients (LHH)\n");
    break;
  case 4:
```

```
      printf("Detail coefficients (HLL)\n");
      break;
    case 5:
      printf("Detail coefficients (HLH)\n");
      break;
    case 6:
      printf("Detail coefficients (HHL)\n");
      break;
    case 7:
      printf("Detail coefficients (HHH)\n");
      break;
    }

    for (i = 1; i <= nwcm; i++) {
      if (i == 1) {
        printf("Coefficients   ");
        for (k = 1; k <= nwcfr; k++) {
          printf("Frame %4" NAG_IFMT, k);
          for (j = 1; j <= 9 * nwcn - 8; j++)
            printf(" ");
        }
        printf("\n");
      }

      for (k = 1; k <= nwcfr; k++) {
        if (k == 1 && i == 1)
          printf("%5" NAG_IFMT "%8s", cindex, " ");
        else if (k == 1)
          printf("%13s", " ");
        else
          printf("%2s", " ");
        for (j = 1; j <= nwcn; j++) {
          printf("%8.4f ", D(i, j, k));
        }
      }
      printf("\n");
    }
    printf("\n");
  }
  fflush(stdout);

  /* nag_idwt_3d (c09fbc).
   * Three-dimensional inverse discrete wavelet transform
   */
  nag_idwt_3d(m, n, fr, lenc, c, b, ldb, sdb, icomm, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_idwt_3d (c09fbc).\n%s\n", fail.message);
    exit_status = 8;
    goto END;
  }

  printf("Output Data :\n");
  fflush(stdout);
  for (k = 1; k <= fr; k++) {
    /* nag_gen_real_mat_print_comp (x04cbc).
     * Prints out a matrix.
     */
    nag_gen_real_mat_print_comp(order, matrix, diag, m, n, &B(1, 1, k), ldb,
                                "%8.4f", " ", Nag_NoLabels, 0, Nag_NoLabels,
                                0, 80, 0, 0, &fail);
    if (fail.code != NE_NOERROR) {
      printf("Error from nag_gen_real_mat_print_comp (x04cbc).\n%s\n",
              fail.message);
      exit_status = 9;
      goto END;
    }
    printf("\n");
    fflush(stdout);
  }

END:
```

```
  NAG_FREE(a);
  NAG_FREE(b);
  NAG_FREE(c);
  NAG_FREE(d);

  return exit_status;
}
```

10.2 Program Data

```
nag_dwt_3d (c09fac) Example Program Data

  5      4       3              : m, n, fr

 Nag_Haar
 Nag_HalfPointSymmetric         : wavnam, mode

 3.0000  2.0000  2.0000  2.0000
 2.0000  9.0000  1.0000  2.0000
 2.0000  5.0000  1.0000  2.0000
 1.0000  6.0000  2.0000  2.0000
 5.0000  3.0000  2.0000  2.0000 : frame 1

 2.0000  1.0000  5.0000  1.0000
 2.0000  9.0000  5.0000  2.0000
 2.0000  3.0000  2.0000  7.0000
 2.0000  1.0000  1.0000  2.0000
 2.0000  1.0000  2.0000  8.0000 : frame 2

 3.0000  1.0000  4.0000  1.0000
 1.0000  1.0000  2.0000  1.0000
 4.0000  1.0000  7.0000  2.0000
 3.0000  2.0000  1.0000  5.0000
 1.0000  1.0000  2.0000  2.0000 : frame 3
```

10.3 Program Results

```
nag_dwt_3d (c09fac) Example Program Results

Parameters read from file ::
DWT :: Wavelet  : Nag_Haar
       End mode : Nag_HalfPointSymmetric
       m  :    5
       n  :    4
       fr :    3

Input Data :
    3.0000   2.0000   2.0000   2.0000
    2.0000   9.0000   1.0000   2.0000
    2.0000   5.0000   1.0000   2.0000
    1.0000   6.0000   2.0000   2.0000
    5.0000   3.0000   2.0000   2.0000

    2.0000   1.0000   5.0000   1.0000
    2.0000   9.0000   5.0000   2.0000
    2.0000   3.0000   2.0000   7.0000
    2.0000   1.0000   1.0000   2.0000
    2.0000   1.0000   2.0000   8.0000

    3.0000   1.0000   4.0000   1.0000
    1.0000   1.0000   2.0000   1.0000
    4.0000   1.0000   7.0000   2.0000
    3.0000   2.0000   1.0000   5.0000
    1.0000   1.0000   2.0000   2.0000

Approximation coefficients (LLL)
Coefficients   Frame    1          Frame    2
    0         10.6066   7.0711     4.2426   5.6569
               7.7782   6.7175     7.0711  10.6066
               7.7782   9.8995     2.8284   5.6569
```

```
Detail coefficients (LLH)
Coefficients   Frame    1          Frame    2
    1          0.7071  -2.1213     0.0000   0.0000
               2.1213  -1.7678     0.0000   0.0000
               3.5355  -4.2426     0.0000   0.0000

Detail coefficients (LHL)
Coefficients   Frame    1          Frame    2
    2         -4.2426   2.1213     1.4142   2.8284
              -2.8284  -2.4749     2.8284   0.7071
               2.1213  -4.2426     0.0000   0.0000

Detail coefficients (LHH)
Coefficients   Frame    1          Frame    2
    3          0.0000  -2.8284     0.0000   0.0000
              -2.8284   1.7678     0.0000   0.0000
               0.7071   4.2426     0.0000   0.0000

Detail coefficients (HLL)
Coefficients   Frame    1          Frame    2
    4         -4.9497   0.0000     1.4142   1.4142
               0.7071   1.7678    -0.0000   2.1213
               0.0000   0.0000     0.0000   0.0000

Detail coefficients (HLH)
Coefficients   Frame    1          Frame    2
    5          0.7071   0.7071     0.0000   0.0000
              -0.7071  -2.4749     0.0000   0.0000
               0.0000   0.0000     0.0000   0.0000

Detail coefficients (HHL)
Coefficients   Frame    1          Frame    2
    6          5.6569   0.7071     1.4142   1.4142
               0.0000  -1.7678     1.4142   6.3640
               0.0000   0.0000     0.0000   0.0000

Detail coefficients (HHH)
Coefficients   Frame    1          Frame    2
    7          0.0000   0.0000     0.0000   0.0000
               1.4142   1.0607     0.0000   0.0000
               0.0000   0.0000     0.0000   0.0000

Output Data :
    3.0000   2.0000   2.0000   2.0000
    2.0000   9.0000   1.0000   2.0000
    2.0000   5.0000   1.0000   2.0000
    1.0000   6.0000   2.0000   2.0000
    5.0000   3.0000   2.0000   2.0000

    2.0000   1.0000   5.0000   1.0000
    2.0000   9.0000   5.0000   2.0000
    2.0000   3.0000   2.0000   7.0000
    2.0000   1.0000   1.0000   2.0000
    2.0000   1.0000   2.0000   8.0000

    3.0000   1.0000   4.0000   1.0000
    1.0000   1.0000   2.0000   1.0000
    4.0000   1.0000   7.0000   2.0000
    3.0000   2.0000   1.0000   5.0000
    1.0000   1.0000   2.0000   2.0000
```