NAG Library Function Document

nag_wfilt (c09aac)

1 Purpose

nag_wfilt (c09aac) returns the details of the chosen one-dimensional discrete wavelet filter. For a chosen mother wavelet, discrete wavelet transform type (single-level or multi-level DWT or MODWT) and end extension method, this function returns the maximum number of levels of resolution (appropriate to a multi-level transform), the filter length, and the number of approximation coefficients (equal to the number of detail coefficients) for a single-level DWT or MODWT or the total number of coefficients for a multi-level DWT or MODWT. This function must be called before any of the one-dimensional discrete transform functions in this chapter.

2 Specification

```
#include <nag.h>
#include <nagc09.h>
void nag_wfilt (Nag_Wavelet wavnam, Nag_WaveletTransform wtrans,
     Nag_WaveletMode mode, Integer n, Integer *nwlmax, Integer *nf,
     Integer *nwc, Integer icomm[], NagError *fail)
```

3 Description

One-dimensional discrete wavelet transforms (DWT) or maximum overlap wavelet transforms (MODWT) are characterised by the mother wavelet, the end extension method and whether multiresolution analysis is to be performed. For the selected combination of choices for these three characteristics, and for a given length, n, of the input data array, x, nag_wfilt (c09aac) returns the dimension details for the transform determined by this combination. The dimension details are: $l_{\max}$, the maximum number of levels of resolution that that could be computed were a multi-level DWT/MODWT applied; n_f, the filter length; n_c the number of approximation (or detail) coefficients for a single-level DWT/MODWT or the total number of coefficients generated by a multi-level DWT/MODWT over $l_{\max}$ levels. These values are also stored in the communication array **icomm**, as are the input choices, so that they may be conveniently communicated to the one-dimensional transform functions in this chapter.

4 References

None.

5 Arguments

1: **wavnam** – Nag_Wavelet *Input*

On entry: the name of the mother wavelet. See the c09 Chapter Introduction for details.

wavnam = Nag_Haar
Haar wavelet.

wavnam = Nag_Daubechies***n***, where $\boldsymbol{n} = 2, 3, \dots, 10$
Daubechies wavelet with $\boldsymbol{n}$ vanishing moments ($2\boldsymbol{n}$ coefficients). For example, **wavnam** = Nag_Daubechies4 is the name for the Daubechies wavelet with 4 vanishing moments (8 coefficients).

wavnam = Nag_Biorthogonal***x_y***, where ***x_y*** can be one of 1_1, 1_3, 1_5, 2_2, 2_4, 2_6, 2_8, 3_1, 3_3, 3_5 or 3_7

Biorthogonal wavelet of order ***x.y***. For example **wavnam** = Nag_Biorthogonal1_1 is the name for the Biorthogonal wavelet of order 1.1.

Constraint: **wavnam** = Nag_Haar, Nag_Daubechies2, Nag_Daubechies3, Nag_Daubechies4, Nag_Daubechies5, Nag_Daubechies6, Nag_Daubechies7, Nag_Daubechies8, Nag_Daubechies9, Nag_Daubechies10, Nag_Biorthogonal1_1, Nag_Biorthogonal1_3, Nag_Biorthogonal1_5, Nag_Biorthogonal2_2, Nag_Biorthogonal2_4, Nag_Biorthogonal2_6, Nag_Biorthogonal2_8, Nag_Biorthogonal3_1, Nag_Biorthogonal3_3, Nag_Biorthogonal3_5 or Nag_Biorthogonal3_7.

2: **wtrans** – Nag_WaveletTransform *Input*

On entry: the type of discrete wavelet transform that is to be applied.

wtrans = Nag_SingleLevel

Single-level decomposition or reconstruction by discrete wavelet transform.

wtrans = Nag_MultiLevel

Multiresolution, by a multi-level DWT or its inverse.

wtrans = Nag_MODWTSingle

Single-level decomposition or reconstruction by maximal overlap discrete wavelet transform.

wtrans = Nag_MODWTMulti

Multi-level resolution by a maximal overlap discrete wavelet transform or its inverse.

Constraint: **wtrans** = Nag_SingleLevel, Nag_MultiLevel, Nag_MODWTSingle or Nag_MODWTMulti.

3: **mode** – Nag_WaveletMode *Input*

On entry: the end extension method. Note that only periodic end extension is currently available for the MODWT.

mode = Nag_Periodic

Periodic end extension.

mode = Nag_HalfPointSymmetric

Half-point symmetric end extension.

mode = Nag_WholePointSymmetric

Whole-point symmetric end extension.

mode = Nag_ZeroPadded

Zero end extension.

Constraints:

mode = Nag_Periodic, Nag_HalfPointSymmetric, Nag_WholePointSymmetric or Nag_ZeroPadded for DWT;
mode = Nag_Periodic for MODWT.

4: **n** – Integer *Input*

On entry: the number of elements, n, in the input data array, x.

Constraint: **n** ≥ 2.

5: **nwlmax** – Integer * *Output*

On exit: the maximum number of levels of resolution, $l_{\max}$, that can be computed when a multi-level discrete wavelet transform is applied. It is such that $2^{l_{\max}} \leq n < 2^{l_{\max}+1}$, for $l_{\max}$ an integer.

6: **nf** – Integer * *Output*

On exit: the filter length, n_f, for the supplied mother wavelet. This is used to determine the number of coefficients to be generated by the chosen transform.

7: **nwc** – Integer * *Output*

On exit: for a single-level transform (**wtrans** = Nag_SingleLevel or Nag_MODWTSingle), the number of approximation coefficients that would be generated for the given problem size, mother wavelet, extension method and type of transform; this is also the corresponding number of detail coefficients. For a multi-level transform (**wtrans** = Nag_MultiLevel or Nag_MODWTMulti) the total number of coefficients that would be generated over $l_{\max}$ levels and with **keepa** = Nag_StoreAll for MODWT.

8: **icomm**[**100**] – Integer *Communication Array*

On exit: contains details of the wavelet transform and the problem dimension which is to be communicated to the one-dimensional discrete discrete transform functions in this chapter.

9: **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_BAD_PARAM

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

On entry, **wtrans** = Nag_MODWTSingle or Nag_MODWTMulti and **mode** $\neq$ Nag_Periodic.
Constraint: **mode** = Nag_Periodic when **wtrans** = Nag_MODWTSingle or Nag_MODWTMulti.

NE_INT

On entry, **n** = ⟨*value*⟩.
Constraint: **n** ≥ 2.

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

Not applicable.

8 Parallelism and Performance

Not applicable.

9 Further Comments

None.

10 Example

This example computes the one-dimensional multi-level resolution for 8 values by a discrete wavelet transform using the Haar wavelet with zero end extensions. The length of the wavelet filter, the number of levels of resolution, the number of approximation coefficients at each level and the total number of wavelet coefficients are printed.

10.1 Program Text

```
/* nag_wfilt (c09aac) 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, nnz, nwc, nwlmax, ny;
  Integer         *dwtlev = 0, *icomm = 0;
  NagError        fail;
  Nag_Wavelet     wavnamenum;
  Nag_WaveletMode modenum;
  /*Double scalar and array declarations */
  double          *c = 0, *x = 0, *y = 0;
  /*Character scalar and array declarations */
  char            mode[24], wavnam[20];

  INIT_FAIL(fail);

  printf("nag_wfilt (c09aac) Example Program Results\n\n");
  fflush(stdout);

  /*     Skip heading in data file*/
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif
  /*     Read n - length of input data sequence*/
#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 Wavelet name (wavnam) and end mode (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("   Parameters read from file :: \n");
      printf("     Wavelet  :%15s\n", wavnam);
      printf("     End mode :%15s\n", mode);
      printf("     N        :%15"NAG_IFMT"\n\n", n);
      /*        Read data array and write it out*/
      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("%8.3f%s", x[i], (i+1)%8?" ":"\n");
        }
#ifdef _WIN32
      scanf_s("%*[^\n] ");
#else
      scanf("%*[^\n] ");
#endif
      printf("\n");
      /*
       * nag_wfilt (c09aac)
       * Wavelet filter query
       */
      nag_wfilt(wavnamenum, Nag_MultiLevel, modenum, n, &nwlmax, &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 (!(c = NAG_ALLOC(nwc, double)) ||
          !(dwtlev = NAG_ALLOC(nwlmax+1, Integer)))
        {
          printf("Allocation failure\n");
          exit_status = -1;
          goto END;
        }
      /*        Perform Discrete Wavelet transform*/
      /*
       * nag_mldwt (c09ccc)
       * one-dimensional multi-level discrete wavelet transform (mldwt)
       */
      nag_mldwt(n, x, nwc, c, nwlmax, dwtlev, icomm, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf("Error from nag_mldwt (c09ccc).\n%s\n", fail.message);
          exit_status = 1;
          goto END;
        }
      printf("   Length of wavelet filter :            %10"NAG_IFMT"\n", nf);
      printf("   Number of Levels :                    %10"NAG_IFMT"\n",
             nwlmax);
      printf("   Number of coefficients in each level : \n");
      for (i = 0; i < nwlmax+1; i++)
```

```
        printf("    %8"NAG_IFMT"%s", dwtlev[i], (i+1)%8?" ":"\n");
      printf("\n");
      printf("   Total number of wavelet coefficients :%10"NAG_IFMT"\n", nwc);
      nnz = 0;
      for (i = 0; i < nwlmax+1; i++)
        nnz = nnz+dwtlev[i];
      printf("\n");
      printf("   Wavelet coefficients C : \n");
      for (i = 0; i < nnz; i++)
        printf("%8.3f%s", c[i], (i+1)%8?" ":"\n");
      printf("\n");
      /*          Reconstruct original data*/
      ny = n;
      /*
       * nag_imldwt (c09cdc)
       * one-dimensional inverse multi-level discrete wavelet transform
       * (imldwt)
       */
      nag_imldwt(nwlmax, nwc, c, n, y, icomm, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf("Error from nag_imldwt (c09cdc).\n%s\n", fail.message);
          exit_status = 1;
          goto END;
        }
      printf("\n");
      printf("   Reconstruction       Y : \n");
      for (i = 0; i < ny; i++)
        printf("%8.3f%s", y[i], (i+1)%8?" ":"\n");
    }

 END:
  NAG_FREE(c);
  NAG_FREE(dwtlev);
  NAG_FREE(x);
  NAG_FREE(y);
  NAG_FREE(icomm);

  return exit_status;
}
```

10.2 Program Data

```
nag_wfilt (c09aac) Example Program Data
8                      : n
Nag_Haar  Nag_ZeroPadded  : wavnam, mode
 2.0
 5.0
 8.0
 9.0
 7.0
 4.0
-1.0
 1.0                   : X(1:n)
```

10.3 Program Results

```
nag_wfilt (c09aac) Example Program Results

   Parameters read from file ::
     Wavelet  :       Nag_Haar
     End mode : Nag_ZeroPadded
     N        :              8

   Input Data          X :
   2.000    5.000    8.000    9.000    7.000    4.000   -1.000    1.000

   Length of wavelet filter :                     2
   Number of Levels :                             3
   Number of coefficients in each level :
```

```
         1            1            2            4
 Total number of wavelet coefficients :         8

 Wavelet coefficients C :
12.374    4.596   -5.000    5.500   -2.121   -0.707    2.121   -1.414


 Reconstruction       Y :
 2.000    5.000    8.000    9.000    7.000    4.000   -1.000    1.000
```