NAG Library Function Document

nag_imlmodwt (c09ddc)

1 Purpose

nag_imlmodwt (c09ddc) computes the inverse one-dimensional multi-level maximal overlap discrete wavelet transform (MODWT). This function reconstructs data from (possibly filtered or otherwise manipulated) wavelet transform coefficients calculated by nag_mlmodwt (c09dcc) from an original set of data. The initialization function nag_wfilt (c09aac) must be called first to set up the MODWT options.

2 Specification

```c
#include <nag.h>
#include <nagc09.h>
void nag_imlmodwt (Integer nwlinv, Nag_WaveletCoefficients keepa,
                   Integer lenc, const double c[], Integer n, double y[],
                   const Integer icomm[], NagError *fail)
```

3 Description

nag_imlmodwt (c09ddc) performs the inverse operation of nag_mlmodwt (c09dcc). That is, given a set of wavelet coefficients computed by nag_mlmodwt (c09dcc) using a MODWT as set up by the initialization function nag_wfilt (c09aac) on a real array of length \( n \), nag_imlmodwt (c09ddc) will reconstruct the data array \( y_i \) for \( i = 1, 2, \ldots, n \), from which the coefficients were derived.

4 References


5 Arguments

1: nwlinv – Integer

*Input*

On entry: the number of levels to be used in the inverse multi-level transform. The number of levels must be less than or equal to \( n_{fwd} \), which has the value of argument nwfl as used in the computation of the wavelet coefficients using nag_mlmodwt (c09dcc). The data will be reconstructed to level \( (\text{nwl} - \text{nwlinv}) \), where level 0 is the original input dataset provided to nag_mlmodwt (c09dcc).

Constraint: \( 1 \leq \text{nwlinv} \leq n_{fwd} \), where \( n_{fwd} \) is the value used in a preceding call to nag_mlmodwt (c09dcc).

2: keepa – Nag_WaveletCoefficients

*Input*

On entry: determines whether the approximation coefficients are stored in array c for every level of the computed transform or else only for the final level. In both cases, the detail coefficients are stored in c for every level computed.

keepa = Nag_StoreAll

Retain approximation coefficients for all levels computed.

keepa = Nag_StoreFinal

Retain approximation coefficients for only the final level computed.

Constraint: keepa = Nag_StoreAll or Nag_StoreFinal.
3: \texttt{lenc} – Integer \textit{Input}

\textit{On entry}: the dimension of the array \texttt{c}.

\textit{Constraints}:

- if \texttt{keepa} = \texttt{Nag\_StoreFinal}, \texttt{lenc} $\geq (n_l + 1) \times n_a$;
- if \texttt{keepa} = \texttt{Nag\_StoreAll}, \texttt{lenc} $\geq 2 \times n_l \times n_a$, where \texttt{n_a} is the number of approximation or detail coefficients at each level and is unchanged from the preceding call to nag_mlmmodwt (c09dcd).

4: \texttt{c[lenc]} – const double \textit{Input}

\textit{On entry}: the coefficients of a multi-level wavelet transform of the dataset.

The coefficients are stored in \texttt{c} as follows:

\textbf{If keepa} = \texttt{Nag\_StoreFinal},

\begin{align*}
\text{C}(1 : n_a) & \text{ Contains the level } n_l \text{ approximation coefficients; } \\
\text{C}(n_a + (i - 1) \times n_d + 1 : n_a + i \times n_d) & \text{ Contains the level } (n_l - i + 1) \text{ detail coefficients, for } i = 1, 2, \ldots, n_l;
\end{align*}

\textbf{If keepa} = \texttt{Nag\_StoreAll},

\begin{align*}
\text{C}((i - 1) \times n_a + 1 : i \times n_a) & \text{ Contains the level } (n_l - i + 1) \text{ approximation coefficients, for } i = 1, 2, \ldots, n_l; \\
\text{C}(n_l \times n_a + (i - 1) \times n_d + 1 : n_l \times n_a + i \times n_d) & \text{ Contains the level } i \text{ detail coefficients, for } i = 1, 2, \ldots, n_l.
\end{align*}

The values \texttt{n_a} and \texttt{n_d} denote the numbers of approximation and detail coefficients respectively, which are equal. This number is returned as output in \texttt{na} from a preceding call to nag_mlmmodwt (c09dcd). See nag_mlmmodwt (c09dcd) for details.

5: \texttt{n} – Integer \textit{Input}

\textit{On entry}: \texttt{n}, the length of the data array, \texttt{y}, to be reconstructed.

\textit{Constraint}: This must be the same as the value \texttt{n} passed to the initialization function nag_wfilt (c09aac).

6: \texttt{y[n]} – double \textit{Output}

\textit{On exit}: the dataset reconstructed from the multi-level wavelet transform coefficients and the transformation options supplied to the initialization function nag_wfilt (c09aac).

7: \texttt{icomm[100]} – const Integer \textit{Communication Array}

\textit{On entry}: contains details of the discrete wavelet transform and the problem dimension for the forward transform previously computed by nag_mlmmodwt (c09dcd).

8: \texttt{fail} – NagError * \textit{Input/Output}

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

6 \textbf{Error Indicators and Warnings}

\textbf{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, lenc is set too small: lenc = ⟨value⟩.
Constraint: lenc ≥ ⟨value⟩.

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

NE_INITIALIZATION
On entry, n is inconsistent with the value passed to the initialization function: n = ⟨value⟩, n should be ⟨value⟩.
On entry, the initialization function nag_wfilt (c09aac) has not been called first or it has not been called with wtrans = Nag_MODWTMulti, or the communication array icomm has become corrupted.

NE_INT
On entry, nwlinv = ⟨value⟩.
Constraint: nwlinv ≥ 1.

NE_INT_2
On entry, nwlinv is larger than the number of levels computed by the preceding call to nag_mlmodwt (c09dcc): nwlinv = ⟨value⟩, expected ⟨value⟩.

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
See Section 10 in nag_mlmodwt (c09dcc).