NAG Toolbox

nag_wav_3d_coeff_ins (c09fz)

1 Purpose

nag_wav_3d_coeff_ins (c09fz) inserts a selected set of three-dimensional discrete wavelet transform (DWT) coefficients into the full set of coefficients stored in compact form, which may be later used as input to the reconstruction functions nag_wav_3d_sngl_inv (c09fb) or nag_wav_3d_mxolap_multi_inv (c09fd).

2 Syntax

```
[c, icomm, ifail] = nag_wav_3d_coeff_ins(ilev, cindex, c, d, icomm, 'lenc', lenc)
[c, icomm, ifail] = c09fz(ilev, cindex, c, d, icomm, 'lenc', lenc)
```

3 Description

nag_wav_3d_coeff_ins (c09fz) inserts a selected set of three-dimensional DWT coefficients into the full set of coefficients stored in compact form in a one-dimensional array **c**. It is required that nag_wav_3d_coeff_ins (c09fz) is preceded by a call to the initialization function nag_wav_3d_init (c09ac) and either the forwards transform function nag_wav_3d_sngl_fwd (c09fa) or multi-level forwards transform function nag_wav_3d_multi_fwd (c09fc).

Given an initial three-dimensional data set A, a prior call to nag_wav_3d_sngl_fwd (c09fa) or nag_wav_3d_multi_fwd (c09fc) computes the approximation coefficients (at the highest requested level in the case of nag_wav_3d_multi_fwd (c09fc)) and, seven sets of detail coefficients (at all levels in the case of nag_wav_3d_multi_fwd (c09fc)) and stores these in compact form in a one-dimensional array **c**. nag_wav_3d_coeff_ext (c09fy) can then extract either the approximation coefficients or one of the sets of detail coefficients (at one of the levels following nag_wav_3d_multi_fwd (c09fc)) into a three-dimensional array, **d**. Following some calculation on this set of coefficients (for example, denoising), the updated coefficients in **d** are inserted back into the full set **c** using nag_wav_3d_coeff_ins (c09fz). Several extractions and insertions may be performed. nag_wav_3d_sngl_inv (c09fb) or nag_wav_3d_mxolap_multi_inv (c09fd) can then be used to reconstruct a manipulated data set $\tilde{A}$. The dimensions of **d** depend on the level extracted and are available from either: the arrays **dwtlvm**, **dwtlvn** and **dwtlvfr** as returned by nag_wav_3d_multi_fwd (c09fc) if this was called first; or, otherwise from **nwct**, **nwcn** and **nwcfr** as returned by nag_wav_3d_init (c09ac). See Section 2.1 in the C09 Chapter Introduction for a discussion of the three-dimensional DWT.

4 References

None.

5 Parameters

Note: the following notation is used in this section:

n_{cm} is the number of wavelet coefficients in the first dimension. Following a call to nag_wav_3d_sngl_fwd (c09fa) (i.e., when **ilev** $= 0$) this is equal to **nwct**$/(8 \times$ **nwcn** $\times$ **nwcfr**$)$ as returned by nag_wav_3d_init (c09ac). Following a call to nag_wav_3d_multi_fwd (c09fc) transforming **nwl** levels, and when inserting at level **ilev** > 0, this is equal to **dwtlvm**$($**nwl** $-$ **ilev** $+ 1)$.

n_{cn} is the number of wavelet coefficients in the second dimension. Following a call to nag_wav_3d_sngl_fwd (c09fa) (i.e., when **ilev** $= 0$) this is equal to **nwcn** as returned by

nag_wav_3d_init (c09ac). Following a call to nag_wav_3d_multi_fwd (c09fc) transforming **nwl** levels, and when inserting at level **ilev** > 0, this is equal to **dwtlvn**(**nwl** $-$ **ilev** $+ 1$).

n_{cfr} is the number of wavelet coefficients in the third dimension. Following a call to nag_wav_3d_sngl_fwd (c09fa) (i.e., when **ilev** $= 0$) this is equal to **nwcfr** as returned by nag_wav_3d_init (c09ac). Following a call to nag_wav_3d_multi_fwd (c09fc) transforming **nwl** levels, and when inserting at level **ilev** > 0, this is equal to **dwtlvfr**(**nwl** $-$ **ilev** $+ 1$).

5.1 Compulsory Input Parameters

1: **ilev** – INTEGER

The level at which coefficients are to be inserted.

If **ilev** $= 0$, it is assumed that the coefficient array **c** was produced by a preceding call to the single level function nag_wav_3d_sngl_fwd (c09fa).

If **ilev** > 0, it is assumed that the coefficient array **c** was produced by a preceding call to the multi-level function nag_wav_3d_multi_fwd (c09fc).

Constraints:

ilev $= 0$ (following a call to nag_wav_3d_sngl_fwd (c09fa));
$0 \leq$ **ilev** $\leq$ **nwl**, where **nwl** is as used in a preceding call to nag_wav_3d_multi_fwd (c09fc);
if **cindex** $= 0$, **ilev** $=$ **nwl** (following a call to nag_wav_3d_multi_fwd (c09fc)).

2: **cindex** – INTEGER

Identifies which coefficients to insert. The coefficients are identified as follows:

cindex $= 0$
The approximation coefficients, produced by application of the low pass filter over columns, rows and frames of A (LLL). After a call to the multi-level transform function nag_wav_3d_multi_fwd (c09fc) (which implies that **ilev** > 0) the approximation coefficients are present only for **ilev** $=$ **nwl**, where **nwl** is the value used in a preceding call to nag_wav_3d_multi_fwd (c09fc).

cindex $= 1$
The detail coefficients produced by applying the low pass filter over columns and rows of A and the high pass filter over frames (LLH).

cindex $= 2$
The detail coefficients produced by applying the low pass filter over columns, high pass filter over rows and low pass filter over frames of A (LHL).

cindex $= 3$
The detail coefficients produced by applying the low pass filter over columns of A and high pass filter over rows and frames (LHH).

cindex $= 4$
The detail coefficients produced by applying the high pass filter over columns of A and low pass filter over rows and frames (HLL).

cindex $= 5$
The detail coefficients produced by applying the high pass filter over columns, low pass filter over rows and high pass filter over frames of A (HLH).

cindex $= 6$
The detail coefficients produced by applying the high pass filter over columns and rows of A and the low pass filter over frames (HHL).

cindex $= 7$
The detail coefficients produced by applying the high pass filter over columns, rows and frames of A (HHH).

Constraints:

if **ilev** $= 0$, $0 \leq$ **cindex** ≤ 7;
if **ilev** $=$ **nwl**, following a call to nag_wav_3d_multi_fwd (c09fc) transforming **nwl** levels, $0 \leq$ **cindex** ≤ 7;
otherwise $1 \leq$ **cindex** ≤ 7.

3: **c**(**lenc**) – REAL (KIND=nag_wp) array

Contains the DWT coefficients inserted by previous calls to nag_wav_3d_coeff_ins (c09fz), or computed by a previous call to either nag_wav_3d_sngl_fwd (c09fa) or nag_wav_3d_multi_fwd (c09fc).

4: **d**$(ldd, sdd, :)$ – REAL (KIND=nag_wp) array

The last dimension of the array **d** must be at least n_{cfr}

The coefficients to be inserted.

If the DWT coefficients were computed by nag_wav_3d_sngl_fwd (c09fa) then

if **cindex** $= 0$, the approximation coefficients must be stored in $\mathbf{d}(i,j,k)$, for $i = 1,2,\ldots,n_{\text{cm}}$, $j = 1,2,\ldots,n_{\text{cn}}$ and $k = 1,2,\ldots,n_{\text{cfr}}$;

if $1 \leq$ **cindex** ≤ 7, the detail coefficients, as indicated by **cindex**, must be stored in $\mathbf{d}(i,j,k)$, for $i = 1,2,\ldots,n_{\text{cm}}$, $j = 1,2,\ldots,n_{\text{cn}}$ and $k = 1,2,\ldots,n_{\text{cfr}}$.

If the DWT coefficients were computed by nag_wav_3d_multi_fwd (c09fc) then

if **cindex** $= 0$ and **ilev** $=$ **nwl**, the approximation coefficients must be stored in $\mathbf{d}(i,j,k)$, for $i = 1,2,\ldots,n_{\text{cm}}$, $j = 1,2,\ldots,n_{\text{cn}}$ and $k = 1,2,\ldots,n_{\text{cfr}}$;

if $1 \leq$ **cindex** ≤ 7, the detail coefficients, as indicated by **cindex**, for level **ilev** must be stored in $\mathbf{d}(i,j,k)$, for $i = 1,2,\ldots,n_{\text{cm}}$, $j = 1,2,\ldots,n_{\text{cn}}$ and $k = 1,2,\ldots,n_{\text{cfr}}$.

5: **icomm**(**260**) – INTEGER array

Contains details of the discrete wavelet transform and the problem dimension as setup in the call to the initialization function nag_wav_3d_init (c09ac).

5.2 Optional Input Parameters

1: **lenc** – INTEGER

Default: the dimension of the array **c**.

The dimension of the array **c**.

Constraint: **lenc** must be unchanged from the value used in the preceding call to either nag_wav_3d_sngl_fwd (c09fa) or nag_wav_3d_multi_fwd (c09fc)..

5.3 Output Parameters

1: **c**(**lenc**) – REAL (KIND=nag_wp) array

Contains the same DWT coefficients provided on entry except for those identified by **ilev** and **cindex**, which are updated with the values supplied in **d**, inserted into the correct locations as expected by one of the reconstruction functions nag_wav_3d_sngl_inv (c09fb) (if nag_wav_3d_sngl_fwd (c09fa) was called previously) or nag_wav_3d_mxolap_multi_inv (c09fd) (if nag_wav_3d_multi_fwd (c09fc) was called previously).

2: **icomm**(**260**) – INTEGER array

Communication array, used to store information between calls to nag_wav_3d_coeff_ins (c09fz).

3: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

Constraint: **ilev** $= 0$ following a call to the single level function nag_wav_3d_sngl_fwd (c09fa).

Constraint: **ilev** > 0 following a call to the multi-level function nag_wav_3d_multi_fwd (c09fc).

Constraint: **ilev** $\leq$ **nwl**, where **nwl** is the number of levels used in the call to nag_wav_3d_multi_fwd (c09fc).

ifail $= 2$

Constraint: **cindex** ≤ 7.

Constraint: **cindex** ≥ 0.

ifail $= 3$

Constraint: **lenc** $\geq n_{\mathrm{ct}}$, where n_{ct} is the number of DWT coefficients computed in a previous call to nag_wav_3d_sngl_fwd (c09fa).

Constraint: **lenc** $\geq n_{\mathrm{ct}}$, where n_{ct} is the number of DWT coefficients computed in a previous call to nag_wav_3d_multi_fwd (c09fc).

ifail $= 4$

Constraint: $ldd \geq n_{\mathrm{cm}}$, where n_{cm} is the number of DWT coefficients in the first dimension at the selected level **ilev**.

Constraint: $ldd \geq n_{\mathrm{cm}}$, where n_{cm} is the number of DWT coefficients in the first dimension following the single level transform.

Constraint: $sdd \geq n_{\mathrm{cn}}$, where n_{cn} is the number of DWT coefficients in the second dimension at the selected level **ilev**.

Constraint: $sdd \geq n_{\mathrm{cn}}$, where n_{cn} is the number of DWT coefficients in the second dimension following the single level transform.

ifail $= 5$

Constraint: **cindex** > 0 when **ilev** $<$ **nwl** in the preceding call to nag_wav_3d_multi_fwd (c09fc).

ifail $= 6$

Either the initialization function has not been called first or **icomm** has been corrupted.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

Your licence key may have expired or may not have been installed correctly.

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

The following example demonstrates using the coefficient extraction and insertion functions in order to apply denoising using a thresholding operation. The original input data has artificial noise introduced to it, taken from a normal random number distribution. Reconstruction then takes place on both the noisy data and denoised data. The Mean Square Errors (MSE) of the two reconstructions are printed along with the reconstruction of the denoised data. The MSE of the denoised reconstruction is less than that of the noisy reconstruction.

9.1 Program Text

```
      function c09fz_example

fprintf('c09fz example results\n\n');

% 3D Data
m = nag_int(4);
n = nag_int(4);
f = nag_int(4);
a = zeros(m,n,f);
a(1:2:m,:,1:2:f) = 0.01;
a(2:2:m,:,2:2:f) = 0.01;
a(2:2:m,:,1:2:f) = 1;
a(1:2:m,:,2:2:f) = 1;

% Add Noise
genid = nag_int(3);
subid = nag_int(0);
seed(1) = nag_int(642521);
[state, ifail] = g05kf(genid, subid, seed);
[state, x, ifail] = g05sk(m*n*f, 0, 1.0e-4, state);
an = a + reshape(x,[m,n,f]);

fprintf('\nInput data a:\n');
disp(a);
fprintf('\nNoisy data an:\n');
disp(an);

% Wavelet setup
wavnam = 'Haar';
mode = 'Period';
wtrans = 'Multilevel';
[nwl, nf, nwct, nwcn, nwcf, icomm, ifail] = ...
c09ac(...
      wavnam, wtrans, mode, m, n, f);

% Multi-level wavelet transform on noisy data
[c, dwtlvm, dwtlvn, dwtlvfr, icomm, ifail] = ...
c09fc(...
       n, f, an, nwct, nwl, icomm);

% Reconstruct without thresholding of detail coefficients
[b, ifail] = c09fd(nwl, c, m, n, f, icomm);

% Mean square error of noisy reconstruction
mse = (norm(reshape(a-b,[m*n*f,1]))^2)/(double(m*n*f));
fprintf('Without denoising Mean Square Error is %9.6f\n',mse);
```

```
% De-noise by applying hard threshold to detail coefficients
thresh = 0.01*sqrt(2*log(double(m*n*f)));
nt = 0;
nnt = 0;
for ilev = 1:nwl
  level = nag_int(nwl - ilev + 1);

  for detail = nag_int(1:7)
    % Extract the selected set of coefficients.
    [d, icomm, ifail] = c09fy(...
                              level, detail, c, icomm);

    % Threshold
    d1 = dwtlvm(ilev);
    d2 = dwtlvn(ilev);
    d3 = dwtlvfr(ilev);
    for i = 1:d1
      for j = 1:d2
        for k = 1:d3
          if abs(d(i,j,k))<thresh
            d(i,j,k) = 0;
            nt = nt + 1;
          end
          nnt = nnt + 1;
        end
      end
    end
    % Insert de-noised coefficients back into c
    [c, icomm, ifail] = c09fz(...
                               level, detail, c, d, icomm);
  end
end

fprintf('\nNumber of coefficients denoised is %3d out of %3d\n',nt,nnt);

% Reconstruct data after threholding
[b, ifail] = c09fd(nwl, c, m, n, f, icomm);

% Mean square error of de-noised reconstruction
mse = (norm(reshape(a-b,[m*n*f,1]))^2)/(double(m*n*f));
fprintf('With denoising Mean Square Error is %9.6f\n\n',mse);

disp('Reconstruction of denoised input: ');
disp(b);
```

9.2 Program Results

```
     c09fz example results

Input data a:

(:,:,1) =

    0.0100    0.0100    0.0100    0.0100
    1.0000    1.0000    1.0000    1.0000
    0.0100    0.0100    0.0100    0.0100
    1.0000    1.0000    1.0000    1.0000

(:,:,2) =

    1.0000    1.0000    1.0000    1.0000
    0.0100    0.0100    0.0100    0.0100
    1.0000    1.0000    1.0000    1.0000
    0.0100    0.0100    0.0100    0.0100

(:,:,3) =

    0.0100    0.0100    0.0100    0.0100
    1.0000    1.0000    1.0000    1.0000
    0.0100    0.0100    0.0100    0.0100
```

```
    1.0000    1.0000    1.0000    1.0000

(:,:,4) =

    1.0000    1.0000    1.0000    1.0000
    0.0100    0.0100    0.0100    0.0100
    1.0000    1.0000    1.0000    1.0000
    0.0100    0.0100    0.0100    0.0100

Noisy data an:

(:,:,1) =

    0.0135   -0.0093   -0.0004    0.0378
    1.0015    0.9842    1.0007    0.9889
   -0.0017    0.0139    0.0138   -0.0049
    0.9899    1.0070    1.0049    0.9983

(:,:,2) =

    1.0094    1.0080    0.9921    0.9902
    0.0105   -0.0009    0.0160    0.0197
    0.9994    1.0044    0.9956    1.0014
    0.0091   -0.0084    0.0187    0.0023

(:,:,3) =

    0.0058   -0.0053    0.0011    0.0159
    1.0113    0.9894    1.0018    0.9992
    0.0106    0.0082    0.0093    0.0153
    1.0023    1.0157    1.0084    0.9834

(:,:,4) =

    0.9969    1.0010    0.9904    0.9968
    0.0227    0.0022    0.0062    0.0214
    0.9948    0.9981    0.9951    0.9968
    0.0121    0.0103    0.0114    0.0206

Without denoising Mean Square Error is  0.000081

Number of coefficients denoised is  55 out of  63
With denoising Mean Square Error is  0.000015

Reconstruction of denoised input:

(:,:,1) =

    0.0053    0.0053    0.0166    0.0166
    1.0026    1.0026    0.9913    0.9913
    0.0055    0.0055    0.0077    0.0077
    1.0025    1.0025    1.0003    1.0003

(:,:,2) =

    1.0026    1.0026    0.9913    0.9913
    0.0053    0.0053    0.0166    0.0166
    1.0025    1.0025    1.0003    1.0003
    0.0055    0.0055    0.0077    0.0077

(:,:,3) =

    0.0073    0.0073    0.0110    0.0110
    1.0006    1.0006    0.9969    0.9969
    0.0078    0.0078    0.0131    0.0131
    1.0002    1.0002    0.9949    0.9949

(:,:,4) =
```

```
    1.0006    1.0006    0.9969    0.9969
    0.0073    0.0073    0.0110    0.0110
    1.0002    1.0002    0.9949    0.9949
    0.0078    0.0078    0.0131    0.0131
```