NAG Toolbox

nag_wav_3d_sngl_fwd (c09fa)

1 Purpose

nag_wav_3d_sngl_fwd (c09fa) computes the three-dimensional discrete wavelet transform (DWT) at a single level. The initialization function nag_wav_3d_init (c09ac) must be called first to set up the DWT options.

2 Syntax

```
[c, icomm, ifail] = nag_wav_3d_sngl_fwd(n, fr, a, lenc, icomm, 'm', m)
[c, icomm, ifail] = c09fa(n, fr, a, lenc, icomm, 'm', m)
```

3 Description

nag_wav_3d_sngl_fwd (c09fa) 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_wav_3d_init (c09ac).

4 References

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

5 Parameters

5.1 Compulsory Input Parameters

1: **n** – INTEGER

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_wav_3d_init (c09ac).

2: **fr** – INTEGER

The number of two-dimensional frames.

Constraint: this must be the same as the value **fr** passed to the initialization function nag_wav_3d_init (c09ac).

3: **a**$(lda, sda, \mathbf{fr})$ – REAL (KIND=nag_wp) array

lda, the first dimension of the array, must satisfy the constraint $lda \geq \mathbf{m}$.

The m by n by fr three-dimensional input data A, where A_{ijk} is stored in **a**(i, j, k).

4: **lenc** – INTEGER

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_wav_3d_init (c09ac).

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: **m** – INTEGER

Default: the first dimension of the array **a**.

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_wav_3d_init (c09ac).

5.3 Output Parameters

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

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 (c09fy) or nag_wav_3d_coeff_ins (c09fz) 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_wav_3d_init (c09ac)). 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_wav_3d_sngl_inv (c09fb).

2: **icomm**(**260**) – INTEGER array

Contains additional information on the computed transform.

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: **fr** $= \langle value \rangle$, the value of **fr** on initialization (see nag_wav_3d_init (c09ac)).

Constraint: **m** $= \langle value \rangle$, the value of **m** on initialization (see nag_wav_3d_init (c09ac)).

Constraint: **n** $= \langle value \rangle$, the value of **n** on initialization (see nag_wav_3d_init (c09ac)).

ifail $= 2$

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

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

ifail $= 3$

Constraint: **lenc** $\geq n_{\mathrm{ct}}$, where n_{ct} is the number of DWT coefficients returned by nag_wav_3d_init (c09ac) in argument **nwct**.

ifail $= 6$

Either the communication array **icomm** has been corrupted or there has not been a prior call to the initialization function nag_wav_3d_init (c09ac).

The initialization function was called with **wtrans** = 'M'.

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

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 Further Comments

None.

9 Example

This example computes the three-dimensional discrete wavelet decomposition for $5 \times 4 \times 3$ input data using the Haar wavelet, **wavnam** = 'HAAR', with half point end extension, prints the wavelet coefficients and then reconstructs the original data using nag_wav_3d_sngl_inv (c09fb). This example also demonstrates in general how to access any set of coefficients following a single level transform.

9.1 Program Text

```
function c09fa_example

fprintf('c09fa example results\n\n');

% Data
m  = nag_int(5);
n  = nag_int(4);
fr = nag_int(3);
a = zeros(m, n, fr);
a(:, :, 1) = [3, 2, 2, 2;
              2, 9, 1, 2;
              2, 5, 1, 2;
              1, 6, 2, 2;
              5, 3, 2, 2];
a(:, :, 2) = [2, 1, 5, 1;
              2, 9, 5, 2;
              2, 3, 2, 7;
```

```
              2, 1, 1, 2;
              2, 1, 2, 8];
a(:, :, 3) = [3, 1, 4, 1;
              1, 1, 2, 1;
              4, 1, 7, 2;
              3, 2, 1, 5;
              1, 1, 2, 2];

% Query wavelet filter dimensions
wavnam = 'Haar';
mode   = 'half';
wtrans = 'Single Level';
[lmax, nf, nwct, nwcn, nwcfr, icomm, ifail] = ...
  c09ac(...
         wavnam, wtrans, mode, m, n, fr);

% 3D DWT decomposition
[c, icomm, ifail] = c09fa(...
                           n, fr, a, nwct, icomm);

cpass = char('LLH','LHL','LHH','HLL','HLH','HHL','HHH');

% Loop over Low/High passes
want_level = nag_int(0);
matrix = 'General'; diag   = 'Non-unit'; fmt = 'F9.4';
labrow = 'Integer'; labcol = labrow;
rlabs  = {' '};     clabs  = rlabs;
ncols  = nag_int(80); indent = nag_int(0);

for cindex = nag_int(1:7)

  if cindex==0
    fprintf('Approximation coefficients (LLL)\n');
  else
    fprintf('Detail coefficients (%s)\n',cpass(cindex,:));
  end

  % Extract coeficients
  [d, icomm, ifail] = c09fy(...
                             want_level, cindex, c, icomm);
  for k = 1:nwcfr
    title = sprintf('Frame: %3d',k);
    [ifail] =  x04cb(...
                   matrix, diag, d(:,:,k), fmt, title, labrow, ...
                   rlabs, labcol, clabs, ncols, indent);
  end
  fprintf('\n');
end

% 3D DWT reconstruction
[b, ifail] = c09fb(m, n, fr, c, icomm);

fprintf('\nReconstructed Data          b : \n');
% More compact output for expected values
fmt = 'F6.1';
for k=1:fr
  fprintf('\n');
  title = sprintf('Frame: %3d',k);
  [ifail] =  x04cb(...
                   matrix, diag, b(:,:,k), fmt, title, labrow, ...
                   rlabs, labcol, clabs, ncols, indent);
end
```

9.2 Program Results

```
     c09fa example results

Detail coefficients (LLH)
 Frame:   1
           1        2
 1    0.7071  -2.1213
 2    2.1213  -1.7678
 3    3.5355  -4.2426
 Frame:   2
           1        2
 1    0.0000   0.0000
 2    0.0000   0.0000
 3    0.0000   0.0000

Detail coefficients (LHL)
 Frame:   1
           1        2
 1   -4.2426   2.1213
 2   -2.8284  -2.4749
 3    2.1213  -4.2426
 Frame:   2
           1        2
 1    1.4142   2.8284
 2    2.8284   0.7071
 3    0.0000   0.0000

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

Detail coefficients (HLL)
 Frame:   1
           1        2
 1   -4.9497   0.0000
 2    0.7071   1.7678
 3    0.0000   0.0000
 Frame:   2
           1        2
 1    1.4142   1.4142
 2   -0.0000   2.1213
 3    0.0000   0.0000

Detail coefficients (HLH)
 Frame:   1
           1        2
 1    0.7071   0.7071
 2   -0.7071  -2.4749
 3    0.0000   0.0000
 Frame:   2
           1        2
 1    0.0000   0.0000
 2    0.0000   0.0000
 3    0.0000   0.0000

Detail coefficients (HHL)
 Frame:   1
           1        2
 1    5.6569   0.7071
 2    0.0000  -1.7678
 3    0.0000   0.0000
 Frame:   2
```

```
           1        2
 1    1.4142   1.4142
 2    1.4142   6.3640
 3    0.0000   0.0000

Detail coefficients (HHH)
 Frame:   1
           1        2
 1    0.0000   0.0000
 2    1.4142   1.0607
 3    0.0000   0.0000
 Frame:   2
           1        2
 1    0.0000   0.0000
 2    0.0000   0.0000
 3    0.0000   0.0000

Reconstructed Data          b :

 Frame:   1
        1     2     3     4
 1    3.0   2.0   2.0   2.0
 2    2.0   9.0   1.0   2.0
 3    2.0   5.0   1.0   2.0
 4    1.0   6.0   2.0   2.0
 5    5.0   3.0   2.0   2.0

 Frame:   2
        1     2     3     4
 1    2.0   1.0   5.0   1.0
 2    2.0   9.0   5.0   2.0
 3    2.0   3.0   2.0   7.0
 4    2.0   1.0   1.0   2.0
 5    2.0   1.0   2.0   8.0

 Frame:   3
        1     2     3     4
 1    3.0   1.0   4.0   1.0
 2    1.0   1.0   2.0   1.0
 3    4.0   1.0   7.0   2.0
 4    3.0   2.0   1.0   5.0
 5    1.0   1.0   2.0   2.0
```