NAG Library Function Document

nag_dwt_2d (c09eac)

1 Purpose
nag_dwt_2d (c09eac) computes the two-dimensional discrete wavelet transform (DWT) at a single level. The initialization function nag_wfilt_2d (c09abc) must be called first to set up the DWT options.

2 Specification
#include <nag.h>
#include <nagc09.h>

void nag_dwt_2d (Integer m, Integer n, const double a[], Integer lda,
double ca[], Integer ldca, double ch[], Integer ldch, double cv[],
Integer ldcv, double cd[], Integer ldcd, Integer icomm[],
NagError *fail)

3 Description
nag_dwt_2d (c09eac) computes the two-dimensional DWT of a given input data array, considered as a matrix $A$, 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, $A$, first over columns and then to the result over rows. The matrix of approximation (or smooth) coefficients, $Ca$, is produced by the low pass filter over columns and rows; the matrix of horizontal coefficients, $Ch$, is produced by the high pass filter over columns and the low pass filter over rows; the matrix of vertical coefficients, $Cv$, is produced by the low pass filter over columns and the high pass filter over rows; and the matrix of diagonal coefficients, $Cd$, is produced by the high pass filter over columns and rows. 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, $nct$, of coefficients computed for $Ca$, $Ch$, $Cv$, and $Cd$ together and the number of columns of each coefficients matrix, $ncm$, are returned by the initialization function nag_wfilt_2d (c09abc). These values can be used to calculate the number of rows of each coefficients matrix, $nem$, using the formula $nem = nct/(4ncm)$.

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

5 Arguments

1:  m – Integer

    Input

    On entry: number of rows, $m$, of data matrix $A$.

    Constraint: this must be the same as the value m passed to the initialization function nag_wfilt_2d (c09abc).

2:  n – Integer

    Input

    On entry: number of columns, $n$, of data matrix $A$.

    Constraint: this must be the same as the value n passed to the initialization function nag_wfilt_2d (c09abc).
3: \(a[lda \times n] - \text{const double} \) \(\text{Input}\)  
\text{Note: the \((i,j)\)th element of the matrix } A \text{ is stored in } a[(j-1) \times lda + i - 1].  
\text{On entry: the } m \text{ by } n \text{ data matrix } A.

4: \(lda - \text{Integer} \) \(\text{Input}\)  
\text{On entry: the stride separating matrix row elements in the array } a.  
\text{Constraint: } lda \geq m.

5: \(ca[dim] - \text{double} \) \(\text{Output}\)  
\text{Note: the dimension, } dim, \text{ of the array } ca \text{ must be at least } ldca \times n_{cn} \text{ where } n_{cn} \text{ is the argument } nwcn \text{ returned by function nag_wfilt_2d (c09abc).}  
\text{The \((i,j)\)th element of the matrix is stored in } ca[(j-1) \times ldca + i - 1].  
\text{On exit: contains the } n_{cm} \text{ by } n_{cn} \text{ matrix of approximation coefficients, } Ca. 

6: \(ldca - \text{Integer} \) \(\text{Input}\)  
\text{On entry: the stride separating matrix row elements in the array } ca.  
\text{Constraint: } ldca \geq n_{cm} \text{ where } n_{cm} = n_{ct}/(4n_{cn}) \text{ and } n_{cn}, n_{ct} \text{ are returned by the initialization function nag_wfilt_2d (c09abc).}

7: \(ch[dim] - \text{double} \) \(\text{Output}\)  
\text{Note: the dimension, } dim, \text{ of the array } ch \text{ must be at least } ldch \times n_{cn} \text{ where } n_{cn} \text{ is the argument } nwcn \text{ returned by function nag_wfilt_2d (c09abc).}  
\text{The \((i,j)\)th element of the matrix is stored in } ch[(j-1) \times ldch + i - 1].  
\text{On exit: contains the } n_{cm} \text{ by } n_{cn} \text{ matrix of horizontal coefficients, } Ch. 

8: \(ldch - \text{Integer} \) \(\text{Input}\)  
\text{On entry: the stride separating matrix row elements in the array } ch.  
\text{Constraint: } ldch \geq n_{cm} \text{ where } n_{cm} = n_{ct}/(4n_{cn}) \text{ and } n_{cn}, n_{ct} \text{ are returned by the initialization function nag_wfilt_2d (c09abc).}

9: \(cv[dim] - \text{double} \) \(\text{Output}\)  
\text{Note: the dimension, } dim, \text{ of the array } cv \text{ must be at least } ldcv \times n_{cn} \text{ where } n_{cn} \text{ is the argument } nwcn \text{ returned by function nag_wfilt_2d (c09abc).}  
\text{The \((i,j)\)th element of the matrix is stored in } cv[(j-1) \times ldcv + i - 1].  
\text{On exit: contains the } n_{cm} \text{ by } n_{cn} \text{ matrix of vertical coefficients, } Cv. 

10: \(ldcv - \text{Integer} \) \(\text{Input}\)  
\text{On entry: the stride separating matrix row elements in the array } cv.  
\text{Constraint: } ldcv \geq n_{cm} \text{ where } n_{cm} = n_{ct}/(4n_{cn}) \text{ and } n_{cn}, n_{ct} \text{ are returned by the initialization function nag_wfilt_2d (c09abc).}

11: \(cd[dim] - \text{double} \) \(\text{Output}\)  
\text{Note: the dimension, } dim, \text{ of the array } cd \text{ must be at least } ldcd \times n_{cn} \text{ where } n_{cn} \text{ is the argument } nwcn \text{ returned by function nag_wfilt_2d (c09abc).}  
\text{The \((i,j)\)th element of the matrix is stored in } cd[(j-1) \times ldcd + i - 1].  
\text{On exit: contains the } n_{cm} \text{ by } n_{cn} \text{ matrix of diagonal coefficients, } Cd.
ldcd – Integer

Input

On entry: the stride separating matrix row elements in the array cd.

Constraint: \( \text{ldcd} \geq n_{cm} \) where \( n_{cm} = n_{ct}/(4n_{cn}) \) and \( n_{cm}, n_{ct} \) are returned by the initialization function nag_wfilt_2d (c09abc).

icom[180] – 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_2d (c09abc).

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 \( \langle \text{value} \rangle \) had an illegal value.

NE_INITIALIZATION

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

Either the initialization function was called with wtrans = Nag_MultiLevel or icomm has been corrupted.

NE_INT

On entry, \( \text{ldca} = \langle \text{value} \rangle \).

Constraint: \( \text{ldca} \geq \langle \text{value} \rangle \), the number of wavelet coefficients in the first dimension.

On entry, \( \text{ldcd} = \langle \text{value} \rangle \).

Constraint: \( \text{ldcd} \geq \langle \text{value} \rangle \), the number of wavelet coefficients in the first dimension.

On entry, \( \text{ldch} = \langle \text{value} \rangle \).

Constraint: \( \text{ldch} \geq \langle \text{value} \rangle \), the number of wavelet coefficients in the first dimension.

On entry, \( \text{ldev} = \langle \text{value} \rangle \).

Constraint: \( \text{ldev} \geq \langle \text{value} \rangle \), the number of wavelet coefficients in the first dimension.

On entry, \( \text{m} = \langle \text{value} \rangle \).

Constraint: \( \text{m} = \langle \text{value} \rangle \), the value of \( \text{m} \) on initialization (see nag_wfilt_2d (c09abc)).

On entry, \( \text{n} = \langle \text{value} \rangle \).

Constraint: \( \text{n} = \langle \text{value} \rangle \), the value of \( \text{n} \) on initialization (see nag_wfilt_2d (c09abc)).

NE_INT_2

On entry, \( \text{lda} = \langle \text{value} \rangle \) and \( \text{m} = \langle \text{value} \rangle \).

Constraint: \( \text{lda} \geq \text{m} \).

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.
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_2d (c09eac) 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 two-dimensional discrete wavelet decomposition for a $6 \times 6$ input matrix using the Daubechies wavelet, wavnam = Nag_Daubechies4, with half point symmetric end extension.

10.1 Program Text

```c
#include <string.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagc09.h>
#include <nagx04.h>

int main(void)
{
    /* Scalars */
    Integer exit_status = 0;
    Integer i, j, m, n, nf, nwcm, nwcn, nwct, nwl, pda, pdb, pdc;
    /* Arrays */
    char mode[24], wavnam[20], title[50];
    double *a = 0, *b = 0, *ca = 0, *cd = 0, *ch = 0, *cv = 0;
    Integer icomm[(180)];
    /* 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_2d (c09eac) Example Program Results\n\n");
    /* Skip heading in data file and read problem parameters */
```
```c
#ifdef _WIN32
    scanf_s("%*[\n] %"NAG_IFMT"%"NAG_IFMT"%[\n]", &m, &n);
#else
    scanf("%*[\n] %"NAG_IFMT"%"NAG_IFMT"%[\n]", &m, &n);
#endif
pda = m;
pdb = m;
#ifdef _WIN32
    scanf_s("%19s%23s%*[\n] 
", wavnam, _countof(wavnam), mode, _countof(mode));
#else
    scanf("%19s%23s%*[\n] 
", wavnam, mode);
#endif
if (!(a = NAG_ALLOC((pda)*(n), double)) ||
    !(b = NAG_ALLOC((pdb)*(n), double)))
{
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
}
printf(" Parameters read from file ::
");
printf("  DWT :: Wavelet : %s\n", wavnam);
printf(  End mode: %s\n", mode);
fflush(stdout);

wavnamenum = (Nag_Wavelet) nag_enum_name_to_value(wavnam);
modenum = (Nag_WaveletMode) nag_enum_name_to_value(mode);

/* Read data array*/
#define A(I, J) a[(J-1)*pda + I-1]
for (i = 1; i <= m; i++)
#ifdef _WIN32
    for (j = 1; j <= n; j++) scanf_s("%lf", &A(i, j));
#else
    for (j = 1; j <= n; j++) scanf("%lf", &A(i, j));
#endif
#ifdef _WIN32
    scanf_s("%*[\n] 
");
#else
    scanf("%*[\n] 
");
#endif
printf("\n");
fflush(stdout);
nag_gen_real_mat_print_comp(order, matrix, diag, m, n, a, pda, "%8.4f",
"Input Data A :", 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 = 1;
    goto END;
}
printf("\n");

/* nag_wfilt_2d (c09abc).  
* Two-dimensional wavelet filter initialization 
*/
nag_wfilt_2d(wavnamenum, Nag_SingleLevel, modenum, m, n, &nwl, &nf, &nwct, 
&nwcn, icomm, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_wfilt_2d (c09abc).\n%s\n", fail.message);
    exit_status = 2;
    goto END;
}
nwcm = nwct/(4 * nwcn);
```
if ( |
!(ca = NAG_ALLOC((nwcm)*(nwcn), double)) || |
!(cd = NAG_ALLOC((nwcm)*(nwcn), double)) || |
!(cv = NAG_ALLOC((nwcm)*(nwcn), double)) || |
!(ch = NAG_ALLOC((nwcm)*(nwcn), double)) |
)
{
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
}

pdc = nwcm;
/* nag_dwt_2d (c09eac). */
* Two-dimensional discrete wavelet transform */

nag_dwt_2d(m, n, a, pda, ca, pdc, ch, pdc, cv, pdc, cd, pdc, icomm, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_dwt_2d (c09eac).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

fflush(stdout);

/* Print decomposition */
#ifdef _WIN32
    strcpy_s(title, _countof(title), "Approximation coefficients CA :");
#else
    strcpy(title, "Approximation coefficients CA :");
#endif

nag_gen_real_mat_print_comp(order, matrix, diag, nwcm, nwcn, ca, pdc,
"%8.4f",
    title, Nag_NoLabels, 0, Nag_NoLabels, 0, 80, 0,
    &fail);

printf("\n");
fflush(stdout);
#ifdef _WIN32
    strcpy_s(title, _countof(title), "Diagonal coefficients CD :");
#else
    strcpy(title, "Diagonal coefficients CD :");
#endif

nag_gen_real_mat_print_comp(order, matrix, diag, nwcm, nwcn, cd, pdc,
"%8.4f",
    title, Nag_NoLabels, 0, Nag_NoLabels, 0, 80, 0,
    &fail);

printf("\n");
fflush(stdout);
#ifdef _WIN32
    strcpy_s(title, _countof(title), "Horizontal coefficients CH :");
#else
    strcpy(title, "Horizontal coefficients CH :");
#endif

nag_gen_real_mat_print_comp(order, matrix, diag, nwcm, nwcn, ch, pdc,
"%8.4f",
    title, Nag_NoLabels, 0, Nag_NoLabels, 0, 80, 0,
    &fail);

printf("\n");
fflush(stdout);
#ifdef _WIN32
    strcpy_s(title, _countof(title), "Vertical coefficients CV :");
#else
    strcpy(title, "Vertical coefficients CV :");
#endif

nag_gen_real_mat_print_comp(order, matrix, diag, nwcm, nwcn, cv, pdc,
"%8.4f",
    title, Nag_NoLabels, 0, Nag_NoLabels, 0, 80, 0,
    &fail);
&fail); printf("\n");

/* nag_idwt_2d (c09ebc).
* Two-dimensional inverse discrete wavelet transform
*/
nag_idwt_2d(m, n, ca, pdc, ch, pdc, cv, pdc, cd, pdc, b, pdb, icomm, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_idwt_2d (c09ebc).\n%s\n", fail.message);
    exit_status = 3;
goto END;
}
fflush(stdout);
/* Print reconstruction */
#ifdef _WIN32
    strcpy_s(title, _countof(title), "Reconstruction B :");
#else
    strcpy(title, "Reconstruction B :");
#endif
    nag_gen_real_mat_print_comp(order, matrix, diag, m, n, b, pdb, "%8.4f",
    title,
    Nag_NoLabels, 0, Nag_NoLabels, 0, 80, 0, 0, &fail);
END:
NAG_FREE(a);
NAG_FREE(b);
NAG_FREE(ca);
NAG_FREE(cd);
NAG_FREE(ch);
NAG_FREE(cv);
return exit_status;
}

10.2 Program Data

nag_dwt_2d (c09eac) Example Program Data

6 6 : m,n
Nag_Daubechies4 Nag_HalfPointSymmetric : wavnam, mode
8.0000 7.0000 3.0000 3.0000 1.0000 1.0000
4.0000 6.0000 1.0000 5.0000 2.0000 9.0000
8.0000 1.0000 4.0000 9.0000 3.0000 7.0000
9.0000 3.0000 8.0000 2.0000 4.0000 3.0000
1.0000 3.0000 7.0000 1.0000 5.0000 2.0000
4.0000 3.0000 7.0000 7.0000 6.0000 1.0000 :a

10.3 Program Results

nag_dwt_2d (c09eac) Example Program Results

Parameters read from file ::
DWT :: Wavelet : Nag_Daubechies4
End mode: Nag_HalfPointSymmetric

Input Data A :
8.0000 7.0000 3.0000 3.0000 1.0000 1.0000
4.0000 6.0000 1.0000 5.0000 2.0000 9.0000
8.0000 1.0000 4.0000 9.0000 3.0000 7.0000
9.0000 3.0000 8.0000 2.0000 4.0000 3.0000
1.0000 3.0000 7.0000 1.0000 5.0000 2.0000
4.0000 3.0000 7.0000 7.0000 6.0000 1.0000 :a

Approximation coefficients CA :
2.0630 8.4499 15.4726 12.1764 3.8920 2.7112
11.7141 11.1018 5.2923 8.1272 14.5540 2.5729
### Diagonal coefficients CD:

<table>
<thead>
<tr>
<th></th>
<th>0.4777</th>
<th>1.0230</th>
<th>-0.3147</th>
<th>0.0625</th>
<th>0.0831</th>
<th>-1.3316</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.0689</td>
<td>1.5671</td>
<td>-2.1422</td>
<td>0.5565</td>
<td>1.7593</td>
<td>-2.8097</td>
</tr>
<tr>
<td></td>
<td>-0.9555</td>
<td>-1.9276</td>
<td>0.9195</td>
<td>-0.2228</td>
<td>-0.5125</td>
<td>2.6989</td>
</tr>
<tr>
<td></td>
<td>0.2899</td>
<td>0.4453</td>
<td>-0.5695</td>
<td>0.1541</td>
<td>0.4749</td>
<td>-0.7946</td>
</tr>
<tr>
<td></td>
<td>0.4944</td>
<td>1.4145</td>
<td>0.3488</td>
<td>-0.1187</td>
<td>-0.6212</td>
<td>-1.5177</td>
</tr>
<tr>
<td></td>
<td>-1.3753</td>
<td>-2.5224</td>
<td>1.7581</td>
<td>-0.4316</td>
<td>-1.1835</td>
<td>3.7547</td>
</tr>
</tbody>
</table>

### Horizontal coefficients CH:

<table>
<thead>
<tr>
<th></th>
<th>0.4100</th>
<th>-0.1827</th>
<th>1.5354</th>
<th>0.0784</th>
<th>0.8101</th>
<th>-1.3594</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2.3496</td>
<td>-0.9422</td>
<td>2.3780</td>
<td>-1.0540</td>
<td>2.7743</td>
<td>-2.2648</td>
</tr>
<tr>
<td></td>
<td>-1.2690</td>
<td>0.0152</td>
<td>-6.9338</td>
<td>-1.7435</td>
<td>-1.6917</td>
<td>1.2388</td>
</tr>
<tr>
<td></td>
<td>0.6317</td>
<td>-0.0969</td>
<td>2.3300</td>
<td>0.4637</td>
<td>0.6365</td>
<td>-0.1162</td>
</tr>
<tr>
<td></td>
<td>-0.2343</td>
<td>0.3923</td>
<td>5.5457</td>
<td>2.1818</td>
<td>0.2103</td>
<td>-0.8573</td>
</tr>
<tr>
<td></td>
<td>-1.8880</td>
<td>0.8142</td>
<td>-4.8552</td>
<td>0.0736</td>
<td>-2.7395</td>
<td>3.3590</td>
</tr>
</tbody>
</table>

### Vertical coefficients CV:

<table>
<thead>
<tr>
<th></th>
<th>1.5365</th>
<th>5.9678</th>
<th>3.4309</th>
<th>-1.0585</th>
<th>-5.0275</th>
<th>-4.8492</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.6779</td>
<td>-0.0294</td>
<td>-5.3274</td>
<td>1.6483</td>
<td>4.8689</td>
<td>-1.8383</td>
</tr>
<tr>
<td></td>
<td>-1.1065</td>
<td>-2.8791</td>
<td>0.1535</td>
<td>0.0982</td>
<td>0.8417</td>
<td>2.8923</td>
</tr>
<tr>
<td></td>
<td>-0.1359</td>
<td>-2.6633</td>
<td>-5.8549</td>
<td>1.8440</td>
<td>6.2403</td>
<td>0.5697</td>
</tr>
<tr>
<td></td>
<td>1.4244</td>
<td>5.2140</td>
<td>1.6410</td>
<td>-0.4669</td>
<td>-3.2369</td>
<td>-4.5757</td>
</tr>
<tr>
<td></td>
<td>1.0288</td>
<td>2.2521</td>
<td>0.0574</td>
<td>-0.1359</td>
<td>-0.5170</td>
<td>-2.6854</td>
</tr>
</tbody>
</table>

### Reconstruction B:

<table>
<thead>
<tr>
<th></th>
<th>8.0000</th>
<th>7.0000</th>
<th>3.0000</th>
<th>1.0000</th>
<th>1.0000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4.0000</td>
<td>6.0000</td>
<td>1.0000</td>
<td>5.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td></td>
<td>8.0000</td>
<td>1.0000</td>
<td>4.0000</td>
<td>9.0000</td>
<td>3.0000</td>
</tr>
<tr>
<td></td>
<td>9.0000</td>
<td>3.0000</td>
<td>8.0000</td>
<td>2.0000</td>
<td>4.0000</td>
</tr>
<tr>
<td></td>
<td>1.0000</td>
<td>3.0000</td>
<td>7.0000</td>
<td>1.0000</td>
<td>5.0000</td>
</tr>
<tr>
<td></td>
<td>4.0000</td>
<td>3.0000</td>
<td>7.0000</td>
<td>7.0000</td>
<td>6.0000</td>
</tr>
</tbody>
</table>