

# Wavelet toolbox

Version 3.0

Copyright (c) D. E. Newland 1993, 1995

---

This toolbox includes the Matlab<sup>(R)</sup>\* M-files listed below in two directories. Those in the directory `wavebox3` have plotting instructions in Matlab version 3; those in the directory `wavebox4` have plotting instructions in Matlab version 4. All the files appear in both directories.

## *Dilation wavelet transforms*

function `a = wavedn(f,N)`

to compute the dilation wavelet transform `a` of a sequence `f` of  $2^n$  elements ( $n$  an integer) using a wavelet with `N` coefficients

function `f = iwavedn(a,N)`

to compute the inverse dilation wavelet transform `f` of a sequence `a` of  $2^n$  elements ( $n$  an integer) using a wavelet with `N` coefficients

function `A = displayn(f,N)`

to compute and display the one-dimensional dilation wavelet transform of an array `f` of  $2^n$  elements ( $n$  an integer); each row of `A` (of order  $n+1$  rows and  $2^n$  columns) gives the contribution to the reconstructed signal of each level of the wavelet transform

function `A = mapdn(f,N)`

to compute the two-dimensional array `A` which defines the mean-square dilation wavelet map of function `f`; the function `f` has length  $2^n$ , the wavelet has `N` coefficients and the order of `A` is  $n+1$  rows and  $2^{(n-1)}$  columns

function `A = drawmap(f,N)`

to draw the mean-square dilation wavelet map of function `f`, using `mapdn(f,N)`; the map is drawn with 8 contours logarithmically spaced to cover the full range of the magnitude of elements in the array `A`

---

\* MATLAB is a registered trademark of The MathWorks, Inc.

**function A = drawmesh(f,N)**

to draw a mesh diagram of the mean-square surface of function f, using mapdn(f,N)

**function A = wave2dn(F,N)**

to compute the dilation wavelet transform A of a two-dimensional array F using a wavelet with N coefficients; the orders of A and F are both  $2^{n1} \times 2^{n2}$  ( $n1$  and  $n2$  integers)

**function F = iwave2dn(A,N)**

to compute the inverse dilation wavelet transform F of a two-dimensional array A using a wavelet with N coefficients; the orders of A and F are both  $2^{n1} \times 2^{n2}$  ( $n1$  and  $n2$  integers)

**function c = dcoeffs(N)**

to compute the array of N coefficients (N even) required by the dilation wavelet transform; called by the above functions

### ***Harmonic wavelet transforms***

**function a = hwtdn(f)**

to compute the harmonic wavelet transform of the (complex) sequence f whose length is  $2^n$ , ( $n$  an integer); wavelet bands are in octaves

**function f = ihwtdn(a)**

to compute the inverse harmonic wavelet transform of the complex sequence a whose length is  $2^n$  ( $n$  an integer); wavelet bands are in octaves

**function A = hmapdn(f)**

calculates an array A of order  $(n+1)$  rows and  $2^{(n-2)}$  columns to describe the mean-square harmonic wavelet map of the real sequence f with length  $2^n$ ; wavelet bands are in octaves

**function m = musicdn(f)**

calculates the musical wavelet transform of the real sequence f whose length is  $2^n$ ; imports the matrix c from mwcoeffs(N) to define the partitioning

function f = imusicdn(m)

calculates the inverse musical wavelet transform of the complex sequence  $m$  of length  $2^n$ ;  $m$  must be the transform of a real sequence

function [c,C] = mwcoeffs(N)

calculates the array  $c$  which defines how a sequence of length  $2^n$  is partitioned for the musical wavelet transform and the array  $C$  which lays out the corresponding mean-square wavelet map

function A = cleffdn(f)

forms the two-dimensional array  $A$  which defines the mean-square musical wavelet map of the real array  $f$  of length  $2^n$ ; imports the matrix  $C$  from `mwcoeffs(N)` to define the partitioning

function A = musicmap(f)

plots the mean-square musical wavelet map of the real sequence  $f$  of length  $2^n$ ; imports the display matrix from `cleffdn(f)`

function drawmus(A)

draws the musical wavelet map of a two-dimensional array  $A$  derived from `cleffdn(f)`; an alternative to `musicmap(f)` to save recomputing  $A$  when it is desired to examine different contour definitions

### ***Demonstration program***

wavedem3 in directory wavebox 3

and

wavedem4 in directory wavebox4

is the same short demonstration program.

A description of each M-file, giving its purpose, calling statement, algorithm, source reference, and limitations is given below. Help statements are incorporated in each file. Application of the demonstration program is self-explanatory.

### ***Theoretical Background***

The discrete wavelet transform and its application is explained in detail in the book *Random Vibrations, Spectral & Wavelet Analysis* by D. E. Newland, 3rd edition, Longman and John Wiley, chapter 17, 1993. The book contains a detailed list of references to the key literature. A selection of some of the programs in this toolbox is also included in Appendix 7 of the book. Further detail on the harmonic wavelet

transform is given in *Harmonic wavelet analysis*, D. E. Newland, Proc. R. Soc. Lond. A, 443, 203-225, 1993. The theory of the musical wavelet transform is described in *Harmonic and musical wavelets*, D. E. Newland, Proc. R. Soc. Lond. A, 444, 605-620, 1994.

### *Description of M-files*

The references to chapter and figures are to the book *Random Vibrations, Spectral and Wavelet Analysis* (see above). The reference numbers are those listed at the back of the book.

## **wavedn, iwavedn**

### **Purpose:**

Dilation wavelet transform and its inverse

### **Synopsis:**

```
a = wavedn(f,N)
f = iwavedn(a,N)
```

### **Description:**

**wavedn(f,N)** returns an array whose elements are the dilation wavelet transform of the sequence of elements in **f**. The analysing wavelet has **N** coefficients where **N** is even. **iwavedn(a,N)** is the inverse transform.

### **Algorithm:**

Mallat's pyramid algorithm [110, 114] adapted for a circular transform, see Chapter 17.

### **Limitations:**

It is necessary for the sequence **f** to have  $2^n$  elements ( $n$  an integer); its transform **a** also has  $2^n$  elements. The dilation wavelet's coefficients are imported from the M-file **dcoeffs(N)** which is restricted to even values of **N** in the range 2 to 20.

If **f** is complex so that  $f = f_r + i * f_i$ , the wavelet transform **a** is also complex so that  $a = a_r + i * a_i$  where **a<sub>r</sub>** is the transform of **f<sub>r</sub>** and **a<sub>i</sub>** is the transform of **f<sub>i</sub>**, and vice versa.

## **displayn**

### **Purpose:**

To display the reconstruction of a (real) signal from its dilation wavelet transform.

**Synopsis:**

```
A = displayn(f,N)
```

**Description:**

Uses `a = wavedn(f,N)` to compute the dilation wavelet transform of `f` which is a sequence of length  $2^n$ ; then `iwavedn(b,N)` is used repeatedly to generate successive rows of the matrix `A` which has  $n+1$  rows and  $2^n$  columns. Row 1 is the zero-order reconstruction (always a constant), row 2 is the first-order reconstruction, row 3 the second-order reconstruction, and so on. All the rows of `A` added together regenerate the original signal `f`. The results are plotted for examination.

**Algorithm:**

The dilation wavelet transform `a` is partitioned into  $n+1$  adjacent sub-arrays of length 1, 1, 2, 4, 8, 16, ...,  $2^{(n-1)}$  elements. Each array `b` of length  $2^n$  consists of a null matrix except for an embedded sub-array from `a`. The corresponding level of the reconstruction of `f` is obtained by computing `iwavedn(b,N)`. A detailed description is given in Chapter 17.

**Limitations:**

Restricted to even values of `N` in the range 2 to 20. The signal `f` must have only real elements.

**Example:**

```
f = [-9 -7 4 4 4 12 2 6];
A = displayn(f,2)
gives
A =   2   2   2   2   2   2   2   2
      -4  -4  -4  -4   4   4   4   4
        -6  -6   6   6   2   2  -2  -2
          -1   1   0   0  -4   4  -2   2
```

## mapdn

**Purpose:**

Computes a two-dimensional array `A` which defines the mean-square dilation wavelet map for a one-dimensional function `f`.

**Synopsis:**

```
A = mapdn(f,N)
```

**Description:**

Uses `a = wavedn(f,N)` to compute the dilation wavelet transform of `f` which is a sequence of length  $2^n$ ; then reorders and arranges the square of the elements `a(1:2^n)` to form the two-dimensional array `A` which has  $(n+1)$

rows and  $2^{(n-1)}$  columns; the volume under this surface is equal to the mean-square value of  $f$ .

**Algorithm:**

The elements of the dilation wavelet transform  $a$  are reordered according to the strategy described in Chapter 17 so that when the matrix  $A$  is formed each element (squared) is located at the centre of the wavelet it represents. The location of the wavelets (after reordering) is as shown in Fig. 17.18.

**Limitations:**

Restricted to even values of  $N$  in the range 2 to 20;  $f$  may be real or complex.

**Example:**

```
f = [-9 -7 4 4 4 12 2 6];
A = mapdn(f,2)
gives
A =  4  4  4  4
     16 16 16 16
     36 36 4  4
     1  0 16  4
```

## **drawmap**

**Purpose:**

Draws a contour plot of the two-dimensional array  $A$  which defines the mean-square dilation wavelet map for a one-dimensional function  $f$ .

**Synopsis:**

```
A = drawmap(f,N)
```

**Description:**

Uses  $A = \text{mapdn}(f,N)$  to compute the array  $A$  and then the contour instruction to draw a contour plot of array  $A$  with 8 contours logarithmically spaced to cover the whole range of magnitude of the elements of  $A$ .

**Algorithm:**

Sets the contour levels to range from close to  $\min(\min(A))$  to about 80% of  $\max(\max(A))$ .

**Limitations:**

Restricted to even values of  $N$  in the range 2 to 20;  $f$  may be real or complex.

## **drawmesh**

### **Purpose:**

Draws a mesh plot of the two-dimensional array  $A$  which defines the mean-square dilation wavelet map for a one-dimensional function  $f$ .

### **Synopsis:**

$A = \text{drawmesh}(f,N)$

### **Description:**

Uses  $A = \text{mapdn}(f,N)$  to compute the array  $A$  and then the mesh instruction to draw a mesh plot of array  $A$ .

### **Algorithm:**

Applies the mesh instruction directly to  $A$  so that the diagram is drawn to an arithmetic scale. The viewing point is  $M$  where  $M = [-37.5,75]$ .

### **Limitations:**

Restricted to even values of  $N$  in the range 2 to 20;  $f$  may be real or complex.

## **wave2dn, iwave2dn**

### **Purpose:**

Two-dimensional dilation wavelet transform and its inverse

### **Synopsis:**

$A = \text{wave2dn}(F,N)$   
 $F = \text{iwave2dn}(A,N)$

### **Description:**

$\text{wave2dn}(F,N)$  returns a matrix whose elements are the dilation wavelet transform of the elements in  $F$ . The analysing wavelet has  $N$  coefficients where  $N$  is even.  $\text{iwave2dn}(A,N)$  is the inverse transform.

### **Algorithm:**

Mallat's pyramid algorithm [110, 114] adapted for a circular transform as described in Chapter 17.

### Limitations:

It is necessary for the matrix  $F$  to have  $2^{n1} \times 2^{n2}$  elements ( $n1$  and  $n2$  integers); its transform  $A$  also has  $2^{n1} \times 2^{n2}$  elements. The wavelet's coefficients are imported from the M-file `dcoeffs(N)` which is restricted to even values of  $N$  in the range 2 to 20.

If  $F$  is complex so that  $F = F_r + i * F_i$ , the wavelet transform  $A$  is also complex so that  $A = A_r + i * A_i$  where  $A_r$  is the transform of  $F_r$  and  $A_i$  is the transform of  $F_i$ , and vice versa.

## **dcoeffs**

### Purpose:

To generate the  $N$  coefficients required by the dilation wavelet transform.

### Synopsis:

`c = dcoeffs(N)`

### Description:

Generates an array with the  $N$  coefficients required by the dilation wavelet transform.

### Algorithm:

Numerical data is derived from results published by Daubechies [104].

### Limitations:

Restricted to even values of  $N$  in the range 2 to 20.

## **hwtdn, ihwtdn**

### Purpose:

Harmonic wavelet transform and its inverse.

### Synopsis:

`a = hwtdn(f)`  
`f = ihwtdn(a)`

### Description:

Computes the harmonic wavelet transform of the sequence  $f$ , and its inverse.

### Algorithm:

Newland's fft sandwich algorithm described in chapter 17.

**Limitations:**

The sequence length must be  $N=2^n$  where  $n$  is a positive integer. The elements of  $f$  may be complex.

## **hmapdn**

**Purpose:**

To compute the two-dimensional array  $A$  which defines the mean-square harmonic wavelet map.

**Synopsis:**

$A = \text{hmapdn}(f)$

**Description:**

Generates an array of order  $n+1$  rows and  $2^{(n-2)}$  columns to define the mean-square harmonic wavelet map of a sequence  $f$  of length  $2^n$ .

**Algorithm:**

Uses  $\text{hwtdn}(f)$  to compute the elements of the harmonic wavelet transform of  $f$ ; for a real sequence  $f$ , terms occur in complex conjugate pairs. Each element of  $A$  is obtained from complex conjugate pairs by adding their magnitudes squared.

**Limitations:**

The sequence length must be a power of 2 so that  $N=2^n$  ( $n$  an integer) and it is assumed that all the elements of  $f$  are real.

## **musicdn, imusicdn**

**Purpose:**

Musical wavelet transform and its inverse.

**Synopsis:**

$m = \text{musicdn}(f)$   
 $f = \text{imusicdn}(m)$

**Description:**

Computes the musical wavelet transform of the real sequence  $f$  and its inverse.

**Algorithm:**

Newland's fft sandwich algorithm, as for the harmonic wavelet transform except that Fourier coefficients are partitioned into semitone blocks rather than into octave blocks (see chapter 17 and the paper *Harmonic and musical*

*wavelets*, Proc. R. Soc. Lond. A, 1994). Partitioning is done by importing data from `mwcoeffs(N)`.

**Limitations:**

The input sequence `f` can have only real elements and must have  $2^n$  elements ( $n$  an integer).

## **mwcoeffs**

**Purpose:**

To partition the elements of the musical wavelet transform.

**Synopsis:**

`[c,C] = mwcoeffs(N)`

**Description:**

Generates two arrays of integer coefficients needed for the musical wavelet transform and its mean-square map. Array `c` has  $n-2$  rows and 13 columns with the coefficients needed to partition each of octave levels 1 to  $n-2$  into semitones (12 semitone intervals per octave). Array `C` is used for laying out the musical wavelet map; it has  $12*(n-2)$  rows and 2 columns of integer coefficients which define the beginning and end of each of the  $12*(n-2)$  semitones.

**Algorithm:**

The semitone intervals are computed exactly and then replaced by integer approximations.

**Limitations:**

The program calculates results for a sequence length of  $N=4$  and upwards; however results are only used in `musicdn(f)` for levels 5 and upwards ( $N \geq 128$ ).

## **cleffdn**

**Purpose:**

To determine the array `A` which defines the mean-square musical wavelet map.

**Synopsis:**

`A = cleffdn(f)`

**Description:**

Forms the two-dimensional array `A` which defines the mean-square musical wavelet map of the (real) sequence `f`.

**Algorithm:**

Uses  $m = \text{musicdn}(f)$  to compute the musical wavelet transform of  $f$ . Then forms  $A$  by putting each element of  $A$  equal to the square of the magnitudes of the elements in  $m$ .

**Limitations:**

For a sequence length  $N=2^n$  ( $n$  an integer), the array  $A$  covers only octaves 5 to  $n-2$ . Octave levels less than 5 are too short to be subdivided into semitones on an integer scale. It is necessary that  $n \geq 7$  ( $N \geq 128$ ). The input sequence  $f$  is assumed to be real. Because the elements of the musical wavelet transform then occur in complex conjugate pairs, only the lower half of the elements in  $m$  are used when forming  $A$ .

## **musicmap**

**Purpose:**

To draw the musical wavelet map.

**Synopsis:**

$A = \text{musicmap}(f)$

**Description:**

Uses the array  $A$  computed by  $A = \text{cleffdn}(f)$  to draw the mean-square musical wavelet map of a real sequence  $f$ .

**Algorithm:**

Uses the contour instruction to draw a contour map of the two-dimensional array  $A$ . Each octave is plotted as 12 semitones. The contour levels are set as fixed proportions of  $\max(\max(A))$ . Octave and semitone markers are drawn on the graph as dashed and dotted lines respectively.

**Limitations:**

The same as for `cleffdn`.

## **drawmus**

**Purpose:**

To draw the musical wavelet map of the array  $A$ .

**Synopsis:**

`drawmus(A)`

**Description:**

Draws the musical wavelet map from the array **A** which has been computed previously from  $A = \text{cleffdn}(f)$ .

**Algorithm:**

The same as in `musicmap` except that the array **A** is assumed to have been computed previously.

**Limitations:**

The same as for `cleffdn`.

***Errors***

These programs have been written in the course of research into the effectiveness of the wavelet method in signal analysis. They are not known to be error free and users are asked to check their results carefully for possible errors. The author would be extremely grateful for the notification of errors which may be sent by mail to him at the following address:

Professor D. E. Newland  
Cambridge University Engineering Department  
Trumpington Street  
Cambridge, CB2 1PZ, UK

or by e-mail to

`den@eng.cam.ac.uk`

---

***Legal disclaimer***

These M-files have been tested on a range of problems for which they are believed to have worked correctly. However, their accuracy is not guaranteed and no responsibility can be accepted by the author or the distributor in the event of errors being discovered in the programs. We make no warranties, express or implied, that the programs are free from error, or are consistent with any particular standard of merchantability, or that they meet any particular requirements. They should not be relied upon for solving a problem whose incorrect solution could result in injury

to a person or loss of property. The application of these programs is at the user's own risk and the author, publisher and distributor disclaim all liability for damages, whether direct, incidental or consequential, arising from such application or other use of the material in this book.

*Copyright*

These program listings are protected by copyright.

© D E Newland 1993, 1995