# The Daubechies wavelet transform

**Kristian Sandberg**
**Dept. of Applied Mathematics**
**University of Colorado at Boulder**

## 1 Goal

The goal with this lab is to design a Daubechies wavelet transform and use it to compress and de-noise one dimensional signals and images.

## 2 Introduction

This lab can be thought of as an extension to the second lab The Haar wavelet transform. This lab does not introduce any new mathematical concepts compared to the multiresolution analysis you studied in the last lab for the Haar basis. The multiresolution structure is true for the Daubechies basis you will work with in this lab, and just as the Haar transform is a unitary transform, so is the Daubechies transform you will use for this lab.

The main difference in this lab will be that your wavelet transformation code now has to take into account that the filter sometimes will ``run outside the boundary" of the signal.

Keep in mind that the tools you develop in this lab might be helpful for your final project and any future work in signal processing.

## 3 The computational cost of the wavelet transform

Let us compute the computational cost for performing a wavelet transform for a filter of length $L$ and a signal of length $N=2^k$. (Here we do not consider boundary effects but when the length of our signal is long, boundary consideration is not expensive relative to the interior of the signal.)

At the first level we need to move our low pass filter along the whole signal. Since we down sample by a factor of two, this means that the first low pass filtering costs $\frac{LN}{2}$ operations. At the second level we need to move our low pass filter along half of the signal which costs $\frac{LN}{4}$ operations. Continuing in this manner we find that the cost for low pass filtering is given by

$$cost_{low-pass} = L\left(\frac{N}{2}+\frac{N}{4}+\ldots+\frac{N}{N^{\log_2 N}}\right) = NL\left(\frac{1}{2}+\frac{1}{4}+\ldots+\frac{1}{N}\right) \ .$$

The cost for high pass filtering is the same. Hence,

$$\begin{aligned} cost &= cost_{low-pass} + cost_{high-pass} \\ &= NL\left(1+\frac{1}{2}+\frac{1}{4}+\ldots+\frac{1}{2^{k-1}}\right) \\ &= NL\sum_{m=0}^{k-1}\frac{1}{2^m} = NL\left(\frac{\left(\frac{1}{2}\right)^k-1}{\frac{1}{2}-1}\right) \\ &= NL\frac{\left(1-\left(\frac{1}{2}\right)^k\right)}{1-\frac{1}{2}} = 2NL\left(1-\left(\frac{1}{2}\right)^k\right) < 2LN\ . \end{aligned}$$

Now the punch line: *the wavelet transform is an O*(*N*) *operation!* Note that this is not only for performing a one level wavelet decomposition, but for performing the full wavelet transformation. This is somewhat counter intuitive, we may expect a cost of $O(Nlog_2N)$, but the wavelet transform is even fast than that.

## 4 Example: A Daubechies wavelet applied to a finite signal

In this section we will use the (normalized) Daubechies filter

$$\mathbf{c} = (c(0), c(1), c(2), c(3)) = (\frac{1+\sqrt{3}}{4\sqrt{2}}, \frac{3+\sqrt{3}}{4\sqrt{2}}, \frac{3-\sqrt{3}}{4\sqrt{2}}, \frac{1-\sqrt{3}}{4\sqrt{2}})$$

as a low pass filter and

$$\mathbf{d} = (d(0), d(1), d(2), d(3)) = (c(3), -c(2), c(1), -c(0))$$

as a high pass filter to wavelet decompose a signal. The factor $1/\sqrt{2}$ is a normalization factor so that $||\mathbf{c}||_2 = ||\mathbf{d}||_2 = 1$. Before using a filter that is given to you, make sure to find out whether it is normalized or not.

The two filters combined form a so-called *filter bank*. The idea is to use the following example as a test example for the code you will be asked to write and use later on in this lab. Since our filter now has length 4, we need to decide what to do when our filter encounters a boundary point. This example will periodically extend the signal.

Consider $f = (f(0), f(1), \ldots, f(7)) = (2, 5, 8, 9, 7, 4, -1, 1)$. We wish to expand this signal in the basis defined by the filter bank above. In practice, we do this by performing the following steps:

**Step 1:**
Extend the signal periodically. We could also do an even extension of the signal or pad with some constant. What we choose to do at the boundary depends on the application and the signal. If we expect our signal to come from some ``periodic process", a periodic extension makes sense.

We denote the extended signal with a tilde ( $\tilde{f}$ ).

$$\tilde{f} = (f(6), f(7), f(0), f(1), f(2), f(3), f(4), f(5), f(6), f(7)) = (-1, 1, 2, 5, 8, 9, 7, 4,$$

Now move the low and high pass filters along this vector, two steps at a time (shifting by two steps comes from the fact that we are *down sampling* the signal):

$$f_1 = \begin{pmatrix} c(0)f(6) + c(1)f(7) + c(2)f(0) + c(3)f(1) \\ c(0)f(0) + c(1)f(1) + c(2)f(2) + c(3)f(3) \\ c(0)f(2) + c(1)f(3) + c(2)f(4) + c(3)f(5) \\ c(0)f(4) + c(1)f(5) + c(2)f(6) + c(3)f(7) \\ d(0)f(6) + d(1)f(7) + d(2)f(0) + d(3)f(1) \\ d(0)f(0) + d(1)f(1) + d(2)f(2) + d(3)f(3) \\ d(0)f(2) + d(1)f(3) + d(2)f(4) + d(3)f(5) \\ d(0)f(4) + d(1)f(5) + d(2)f(6) + d(3)f(7) \end{pmatrix}$$

$$= (0.155, 5.78, 12.4, 6.37, -0.837, 0.966, 0.871, -3.12)\,.$$

(For clarity, only three significant digits will be printed out in this example but the actual computations made used more digits.)

**Step 2:**
For the next step, we keep the last half of the vector $f_1$ fixed while we low- and high pass filter the first half of the vector. In order to do this, it is necessary to periodically extend the first half of the vector $f_1$.

$$\tilde{f}_1 = (f_1(2), f_1(3), f_1(0), f_1(1), f_1(2), f_1(3), f_1(4), f_1(5), f_1(6), f_1(7))$$

=(12.4,6.37,0.155,5.78,12.4,6.37,-0.837,0.966,0.871,-3.12).

Now move the low and high pass filters along the first six elements of this vector, two steps at a time:

$$f_2 = \begin{pmatrix} c(0)f_1(2) + c(1)f_1(3) + c(2)f_1(0) + c(3)f_1(1) \\ c(0)f_1(0) + c(1)f_1(1) + c(2)f_1(2) + c(3)f_1(3) \\ d(0)f_1(2) + d(1)f_1(3) + d(2)f_1(0) + d(3)f_1(1) \\ d(0)f_1(0) + d(1)f_1(1) + d(2)f_1(2) + d(3)f_1(3) \\ f_1(4) \\ f_1(5) \\ f_1(6) \\ f_1(7) \end{pmatrix}$$
$$= (10.6, 6.87, -5.70, 6.02, -0.837, 0.966, 0.871, -3.12) .$$

Note that we didn't touch the last four elements!

**Step 3:**
In the last step we only low- and high pass filter the first quarter of the vector $f_2$. First we extend the first two elements periodically:

$$\tilde{f}_2 = (f_2(0), f_2(1), f_2(0), f_2(1), f_2(2), f_2(3), f_2(4), f_2(5), f_2(6), f_2(7))$$

=(10.6,6.87,10.6,6.87,-5.70,6.02,-0.837,0.966,0.871,-3.12).

As the final step we act with the filter on the first four elements of this vector:

$$f3 = \begin{pmatrix} c(0)f_2(0) + c(1)f_2(1) + c(2)f_2(0) + c(3)f_2(1) \\ d(0)f_2(0) + d(1)f_2(1) + d(2)f_2(0) + d(3)f_2(1) \\ f_2(2) \\ f_2(3) \\ f_2(4) \\ f_2(5) \\ f_2(6) \\ f_2(7) \end{pmatrix}$$

$$= (12.4, 2.66, -5.70, 6.02, -0.837, 0.966, 0.871, -3.12) .$$

Note that we didn't touch the last six elements!

## 5 Exercises

## Exercise 1: A one dimensional wavelet transform with filter length greater than 2

Construct a code that performs a wavelet decomposition of a signal of length $2^k$ where $k$ is a positive integer. Your code should be able to handle the case when your filter is at least four elements long such as the filter given in the example above.

It is up to you to decide what to do at the boundaries. Among the possibilities are zero padding, even extension or periodic extension. The example above used periodic extension. (Your code only need to handle one kind of extensions.)

Program a function with the following in-/output:
**Input:**

- A vector of length $2^k$ where $k$ is a positive integer.
- The ``level'' of decomposition. (E.g., the first step in the example in section 4 corresponds to a first level decomposition, the second step corresponds to a second level decomposition and the third step corresponds to a third level decomposition.)

**Output:**

- A vector which is the wavelet transformation of the vector you gave as input corresponding to the level you gave as an input.

**a)** Verify that your transform can reproduce the example in section 4.

**b)** Verify that the $l^2$-norm is preserved in each of the three steps in the example.

**c)** Build an inverse transform, i.e., a code that takes a transformed vector and returns a non-transformed vector. (A ``reverse'' of the code in **a)**.)

## Exercise 2: Compression of a one dimensional signal

Generate a signal that samples the function

$$f(t) = \begin{cases} e^{-t}\cos(32\pi t) & t \in [0, 1/2) \\ 0 & t \in [1/2, 3/4) \\ \cos(96\pi t) & t \in [3/4, 7/8) \\ 0 & t \in [7/8, 1) \end{cases} \qquad (1)$$

at at least 1024 points. Plot the wavelet decomposition of the signal at a few different levels.

By using thresholding (like in the previous lab) on the transformed signal, investigate the compression performance of the transform. Compare the compression performance to a compression using the FFT and the Haar transform of the same signal.

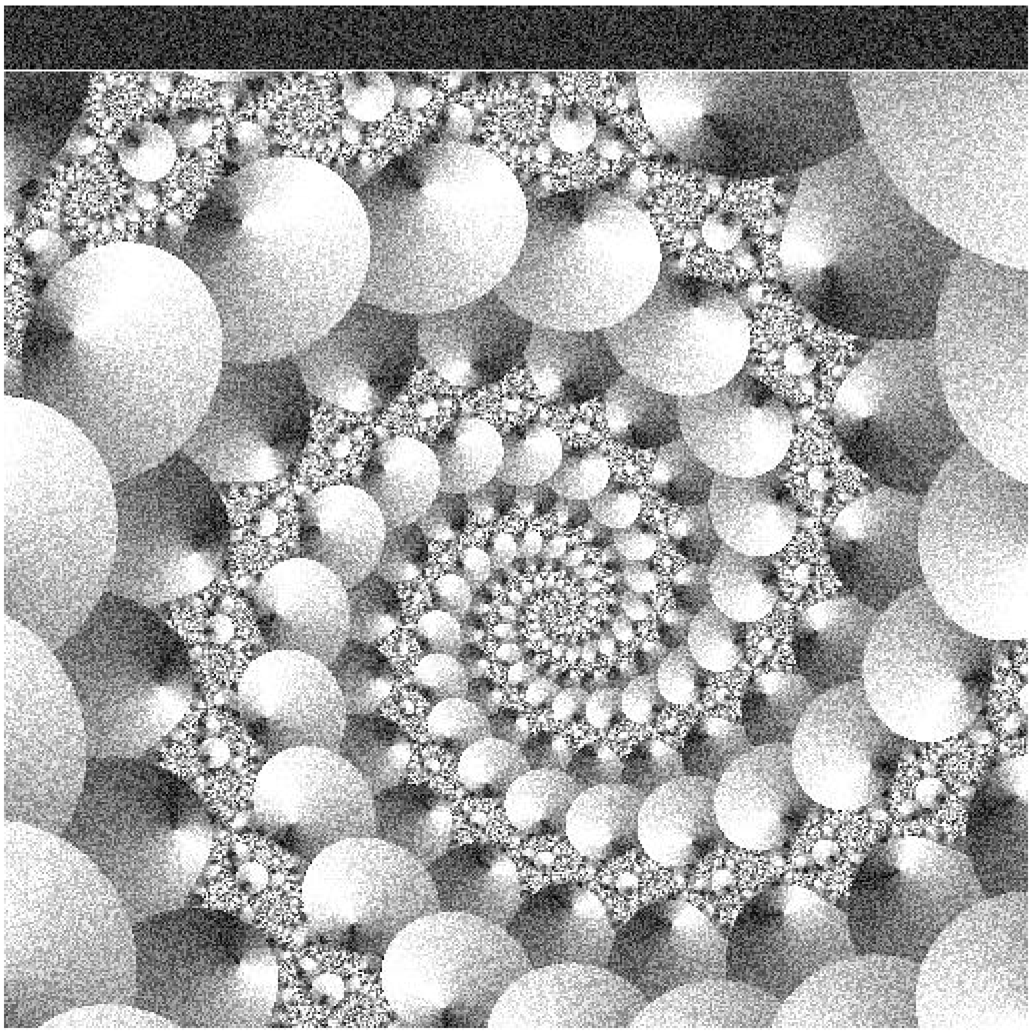
## Exercise 3: The two dimensional wavelet transform

A standard decomposition of a two dimensional signal (image) is easily done by first performing a one dimensional transformation on each row followed by a one dimensional transformation of each column.

Construct a function that wavelet transforms an image for different levels. Test your function on the same images as used for the previous labs (the basket and the image with campus and the Rockies). Compress the image and investigate the compression performance of the transform compared to the other compression techniques you learned in the course.

## Exercise 4: De-noising images

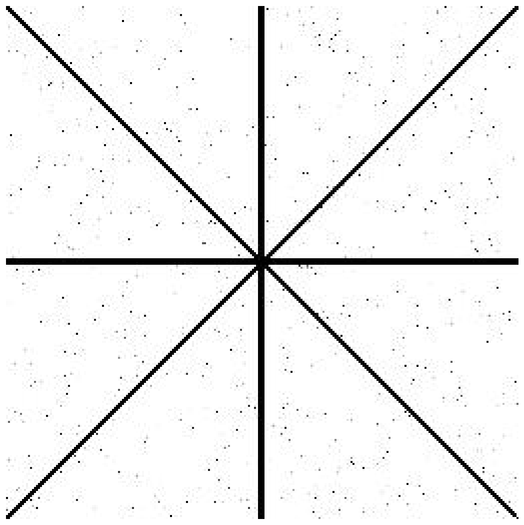
Using the transform constructed in Exercise 1, try to de-noise the following image:



Fractal from Carlson's Fractal Gallery with random noise added.

## Exercise 5: Horizontal, vertical and diagonal features

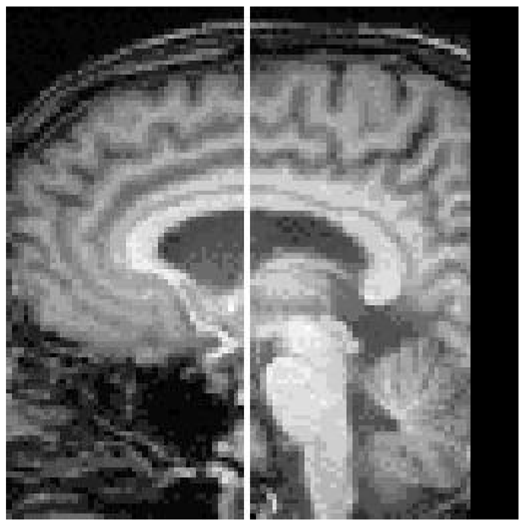
Down load the following image:



Do a wavelet decomposition of this image at a few different levels. What are your observations? (You may have to adjust the brightness of the wavelet decomposition using `brighten()` in Matlab.)

### Exercise 6: ``De-scratching''

Remove (or reduce) the ``scratch'' from the following image!



An MRI (Magnetic Resonance Imaging) image from First Radiology Clinic with "a scratch" added. Medical imaging is a very important application of digital image processing.

## 6 Format of your lab report

Write down your comments to all of the exercises above (word processed). Include a representative collection of the code(s) that you used as an appendix. Make sure to include comments in your code explaining what it does. Include any plots or images you need to justify your conclusions.

## 7 Questions?

In case you have questions regarding the material in this lab, do not hesitate to contact me at `kristian.sandberg@colorado.edu` or visit me during my lab hours Mondays 4:30-6 pm and Thursdays 4-5:30 pm in ECCR 143.

*Lycka till!*

---

## About this document ...

**The Daubechies wavelet transform**

---

*Kristian Sandberg*
*2000-04-14*