Wavelet Algorithms

Assignment 1

Vladimir Kulyukin
Department of Computer Science
Utah State University

- This is a coding assignment. Zip up your Java project (Eclipse or NetBeans) and submit it via Canvas.
- This assignment is worth 10 points.
- This assignment is due by 11:59pm May 21, 2016.
- Readings: Ch. 1 in "Wavelets Made Easy" by Y. Nievergelt.
- You may not use any online coding materials, including any publicly available source code, other than the class lecture notes and the readings. Work it out, do your own coding, and do not plagiarize.

Problem 1 (5 points)

Implement a Java class **OneDHaar** with two methods:

```
public static void orderedFastHaarWaveletTransformForNumIters(double[] sample, int num_iters) {}
public static void inPlaceFastHaarWaveletTransformForNumIters(double[] sample, int num_iters) {}
```

The first method implements the ordered inverse FHWT whereas the second method implements the in-place inverse FHWT, as discussed in class. Then implement the following examples from Chapter 1 of "Wavelets Made Easy": example 1.11, p. 16, example 1.12, p. 16, example 1.17, p. 25, and example 1.18, p. 25. You can use **WaveletsMadeEasyCh01.java** as your test template. Below is the output generated by my code.

```
---- Example 1.11, p. 17
1ordered sweep: 3.0 5.0 2.0 -3.0
2ordered sweep: 4.0 -1.0 2.0 -3.0

---- Example 1.17, p. 25
1 inplace sweep: 3.0 2.0 5.0 -3.0
2 inplace sweep: 4.0 2.0 -1.0 -3.0

---- Example 1.12, p. 19
1 ordered sweep: 2.0 2.0 7.0 9.0 1.0 -2.0 1.0 0.0
2 ordered sweep: 2.0 8.0 0.0 -1.0 1.0 -2.0 1.0 0.0
3 ordered sweep: 5.0 -3.0 0.0 -1.0 1.0 -2.0 1.0 0.0

---- Example 1.18, p. 25
1 inplace sweep: 2.0 1.0 2.0 -2.0 7.0 1.0 9.0 0.0
2 inplace sweep: 2.0 1.0 0.0 -2.0 8.0 1.0 -1.0 0.0
3 inplace sweep: 5.0 1.0 0.0 -2.0 -3.0 1.0 -1.0 0.0
```

Problem 2 (5 points)

Apply your implementation of either ordered transform to the file **beehive_temperatures.txt**. The file contains temperature sensor readings taken in a beehive shown in **Figure 1**.

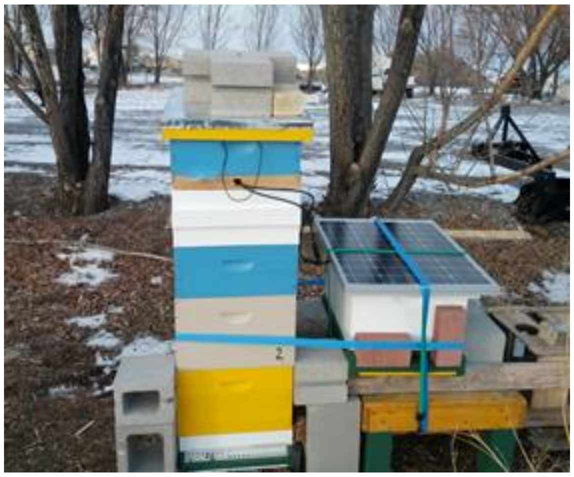

Figure 1. Solar-Powered Beehive Monitor

The file has the following format:

```
2015-05-11_11-45-38 17.812
2015-05-11_11-55-38 18.75
2015-05-11_12-05-38 19.125
2015-05-11_12-15-38 19.625
2015-05-11_12-25-38 20.0
2015-05-11_12-35-38 20.0
2015-05-11_12-45-38 20.062
2015-05-11_12-55-38 20.375
```

The part before the underscore is the date **2015-05-11**.The part after the underscore is the time in 24hr format. The float after the space, e.g., **17.812**, is an actual temperature reading.

Create a Java class **TemperatureAnalysis.java** and do a programmatic data analysis of this file to discover any interesting patterns in the data. Here are a few interesting questions for you to consider. Can you tell apart morning, daytime, evening, night? What was the period of time when the temperature oscillated the greatest? What was the period of time when the temperature oscillated the smallest?

Since this is a research problem, there are no right or wrong answers to these questions. Simply state your observations/discoveries in a short README or in the comments to your source code.