CS 6810/7810: Wavelets and Wavelet Algorithms
Assignment 2
Ordered and In-Place Fast Haar Wavelet Transforms

Vladimir Kulyukin
Department of Computer Science
Utah State University

January 20, 2017

Learning Objectives

1. Ordered Fast Haar Wavelet Transform
2. In-Place Fast Haar Wavelet Transform

Introduction

In this assignment, you will implement two types of the Haar Wavelet Transform covered in Lecture 2: Ordered Fast Haar Wavelet Transform (ordered FHWT) and In-Place Fast Haar Wavelet Transform (in-place FHWT). Although the problems below are formulated in terms of JAVA, you can code up your solution in Python. Please use JAVA 7 and Python 2.7. The reason that I would recommend that you use Python 2.7 is that later in the course, when we do image processing with OpenCV, Python 2.7 will integrate with OpenCV 3.0 much more nicely than Python 3.X. I am sure this will change in time as Python 3 matures, but this appears to be the status quo for now.

Problem 1 (5 pts)

Implement a JAVA class OneDHWT with the following method:

```
public static void ordFHWTForNumIters(double[] signal,
      int num_iters)
{
  // your code here
}
```

This method computes the ordered FHWT applied to the signal for a given number of iterations given as the integer value of the second parameter. I have included the JAVA class $WaveletAlgos\_S17\_HW2.java$ with a few test cases from Ch. 1 of Professor Nievergelt's text book. For example, consider the method below that implements example 1.12 on p. 19.

```
public static void example_1_12_p19(int num_iters)
{
   final double[] signal = {3, 1, 0, 4, 8, 6, 9, 9};
   System.out.print("Original signal: ");
   displayArray(signal);
   OneDHWT.ordFHWTForNumIters(signal, num_iters);
   if ( num_iters == 1 )
      System.out.print("Signal after " + num_iters
                       + " ordered iter: ");
   else
      System.out.print("Signal after " + num_iters
                       + " ordered iters: ");
   displayArray(signal);
   System.out.println();
}
```

Consider this code segment in the main method that calls the above method with different numbers of iterations.

```
System.out.println("---- Example 1.12, p. 19");
example_1_12_p19(1);
example_1_12_p19(2);
example_1_12_p19(3);
System.out.println();
```

The above code segment generates the following output:

```
---- Example 1.12, p. 19
Original signal: 3.0 1.0 0.0 4.0 8.0 6.0 9.0 9.0
```

```
Signal after 1 ordered iter: 2.0 2.0 7.0 9.0 1.0 -2.0 1.0 0.0

Original signal: 3.0 1.0 0.0 4.0 8.0 6.0 9.0 9.0
Signal after 2 ordered iters: 2.0 8.0 0.0 -1.0 1.0 -2.0 1.0 0.0

Original signal: 3.0 1.0 0.0 4.0 8.0 6.0 9.0 9.0
Signal after 3 ordered iters: 5.0 -3.0 0.0 -1.0 1.0 -2.0 1.0 0.0
```

Problem 2 (5 pts)

Add the following method to your OneDHWT class:

```
public static void ordFHWTForNumIters(double[] signal,
      int num_iters)
{
  // your code here
}
```

This method computes the in-place FHWT applied to the signal for a given number of iterations given as the integer value of the second parameter. The class $WaveletAlgos\_S17\_HW2.java$ has several test cases for this method as well.

Output of All Test Cases

To help you debug your code, the file *console_output.txt* includes the output generated in my NetBeans console by the main method in $WaveletAlgos\_S17\_HW2.java$

```
public static void main(String[] args)
{
   System.out.println("---- Example 1.11, p. 17");
   example_1_11_p16(1);
   example_1_11_p16(2);
   System.out.println();

   System.out.println("---- Example 1.12, p. 19");
   example_1_12_p19(1);
   example_1_12_p19(2);
   example_1_12_p19(3);
   System.out.println();
```

```
   System.out.println("---- Example 1.17, p. 25");
   example_1_17_p25(1);
   example_1_17_p25(2);
   System.out.println();

   System.out.println("---- Example 1.18, p. 25");
   example_1_18_p25(1);
   example_1_18_p25(2);
   example_1_18_p25(3);
   System.out.println();

   ordFHWTTest01();
   inpFHWTTest01();
}
```

What To Submit

Submit your OneDHWT.java or OneDHWT.py via Canvas.

Happy Hacking!