UCRL-TR-207189

LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

The Discrete Wavelet Transform with Lifting : A Step by Step Introduction

Chris Elofson

October 12, 2004

Disclaimer

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

The Discrete Wavelet Transform with Lifting : A Step by Step Introduction

Christopher M Elofson, University of Arizona

There is a great deal of information pertaining to wavelets readily available from various sources; several of the more recent sources describe the lifting technique for constructing wavelets. The tutorial paper by Sweldens and Schröder [1] gives a thorough explanation of the lifting approach for Haar bases. While it provides an excellent introduction to the topic, it is not immediately obvious how this approach is extended to nonuniformly spaced data on finite intervals. The present paper provides intermediate steps that supplement the material in [1]. After working through the following discussion, the reader should have no problem deriving the relevant equations presented in Sweldens and Schröder's article. Because of the abundance of information on the Haar basis, this discussion will instead work through the steps using a linear basis set.

I. Introduction

The technique of lifting is a radically simpler alternative to traditional methods for the construction of certain classes of wavelets. It also allows the building of "second generation" wavelets which are more general than "first generation" wavelets in that they can be used in situations where dilation and translation may not be readily usable [1]. The paper by Daubechies and Sweldens [2] provides an illuminating comparison between the construction of wavelets by lifting and by traditional approaches. In this paper, we confine our discussions to the method of lifting.

Lifting works entirely in the spatial domain and can be considered as made up of two steps – predict and update. These are explained below for uniformly spaced data sets and in the following section, we explain the adjustments necessary for non-uniform data sets.

II. Uniformly Spaced Data Sets

II.1 Predict Step

The predict step is in fact simply using some form of interpolation to estimate the values at odd indices (assuming 0-based indexing) by using the surrounding points, then placing at the odd index the difference between the original value at this index and the predicted value. Let (2i+1) be the odd index to be calculated in the predict step. Then we must use the even values at indices (2i) and (2i+2) during interpolation. For now, let us consider only regularly spaced data sets. Then the distance d_{2i+1}, defined as the distance between y_{2i+1} and the approximation is simply:

$$d_{2i+1} = y_{2i+1} - \frac{y_{2i} + y_{2i+2}}{2}$$

In the following figure, the predict process is shown graphically. Note that for this figure i = 0.

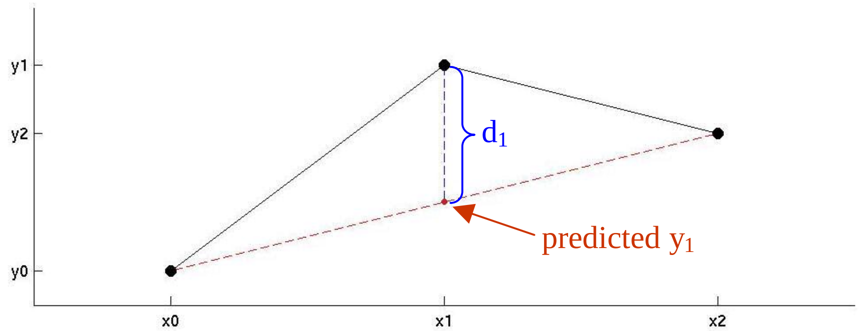

Figure 1. Original data points in black; d_1 is the difference between actual and predicted values at x_1.

As evident in Figure 1, restricting our example to regularly spaced points simplifies things, as the predicted odd value is simply the average of the two neighboring even points. The predict step for higher order polynomials differs only in that a different type of interpolation is used to calculate the predicted value (more points will need to be used).

Throughout this discussion we will use the following example data set so that the reader can easily understand the purpose and effects of each step.

$$x = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix} \qquad y = \begin{bmatrix} 0 \\ 7 \\ 10 \\ 13 \\ 2 \\ 4 \\ 8 \end{bmatrix}$$

Figure 2.1 Sample coordinates.

In Figure 2.2 on the following page, we see the effect of the predict step applied at all the odd points.

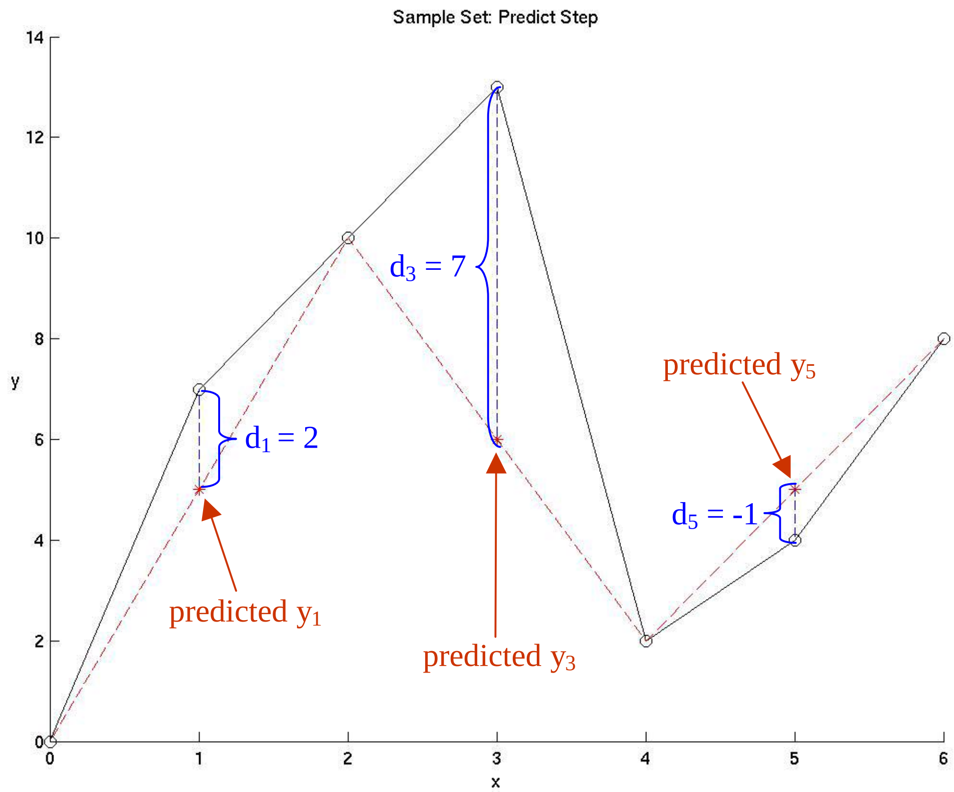

Figure 2.2 Predict Step on Sample Data Set

II.2 Update Step

The update step attempts to create an accurate representation of the original data at a coarser level. If the even values are used as they are, the representation at the coarser level would not be a satisfactory representation of the data at the finer level.

The idea used in the update step to maintain a faithful representation of the data at the finer level is to preserve certain quantities, the standard ones being moments of increasing order. In the case of the Haar basis, only the 0^{th} moment is conserved, but for a linear basis, we need to preserve the 0^{th} moment and the 1^{st} moment. Note that for an update step of degree n, (n+1) conditions are needed.

Let's take a simple example of the update step, using a slightly modified version of the sample data set on the previous page.   We are going to alter the data set so that the first 3 points and the last 3 points lie on a straight line.

$$x = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix} \qquad y = \begin{bmatrix} 0 \\ 5 \\ 10 \\ 13 \\ 2 \\ 4 \\ 6 \end{bmatrix}$$

Figure 3. Simplified Sample Data Set

As shown in Fig. 4, the only region that is changed by using the predicted values at the odd points is that between x_2 and x_4, as $d_1 = d_5 = 0$.

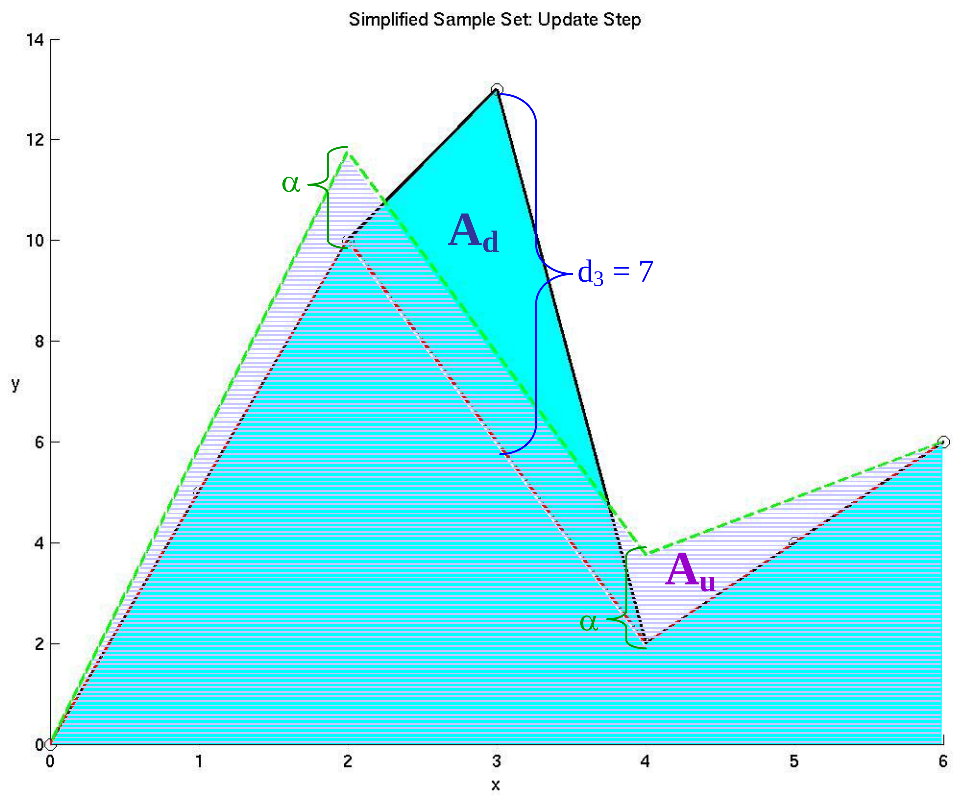

Figure 4. Update step on the Simplified Sample Set – original data shown in black, predict step in red, and updated data in green. The area under the original data is shown in cyan and that under the updated data, in purple.

Unlike the Haar basis, we cannot simply average the data, as this would cause the graph to become discontinuous (stepped, not piecewise-linear). We need to instead update the y-values at indices 2 and 4 in order to account for this change. This update will also alter future predictions over the intervals [x_0, x_2] and [x_4, x_6], effectively distributing the 'extra' area along the entire interval. The next task is to find these update values.

Remember that in the linear update step we want to conserve the 0^{th} and 1^{st} moments. It turns out that with regularly spaced points the changes in y_2 and y_4 are the same – let us call this quantity α – so in the following derivation we will simply work with the 0^{th} moment, or area under the data points. The explanation here is very closely patterned after that in [2].

Figure 4 shows the area under the original data, A_d (cyan), and the area under the updated data, A_u (purple). We require that these two quantities be equal:

$$A_d = \int_{x_0}^{x_6} y dx = \int_{x_0}^{x_6} u dx = A_u$$

Using the trapezoidal rule to calculate each of the integrals, and letting α be the difference between the updated data and the original data at indices 2 and 4 shown in Figure 4, we get the following equations:

$$\int_{x_0}^{x_6} y dx = \frac{1}{2}(x_1 - x_0)(y_1 + y_0) + \frac{1}{2}(x_2 - x_1)(y_2 + y_1) + ... + \frac{1}{2}(x_6 - x_5)(y_6 + y_5)$$

$$\int_{x_0}^{x_6} u dx = \frac{1}{2}(x_2 - x_0)((y_2 + \alpha) + y_0) + \frac{1}{2}(x_4 - x_2)((y_4 + \alpha) + (y_2 + \alpha)) + \frac{1}{2}(x_6 - x_4)(y_6 + (y_4 + \alpha))$$

Simplifying the area equations on the previous page and rewriting the differences in x-values in terms of Δx produces

$$A_s = A_d$$

$$\Delta x(4\alpha) = \frac{\Delta x}{2}(-y_0 + 2y_1 - 2y_2 + 2y_3 - 2y_4 + 2y_5 - y_6)$$

$$\Delta x(4\alpha) = \frac{\Delta x}{2}(-y_0 + 2y_1 - 2y_2 + 2y_3 - 2y_4 + 2y_5 - y_6)$$

$$4\alpha = \frac{-y_0 - y_2}{2} + \frac{-y_4 - y_6}{2} + \frac{1}{2}(2y_1 - y_2 + 2y_3 - y_4 + 2y_5)$$

Since $d_1 = d_5 = 0$, we know that y_1 equals the average of y_0 and y_2, and similarly, that y_5 equals the average of y_4 and y_6. Thus,

$$4\alpha = -y_1 - y_5 + \frac{1}{2}(2y_1 - y_2 + 2y_3 - y_4 + 2y_5)$$

and

$$\alpha = \frac{-y_2}{8} + \frac{y_3}{4} + \frac{-y_4}{8}$$

We also know

$$y_3 = d_3 + \frac{y_2 + y_4}{2} \quad \text{, and thus} \quad \alpha = \frac{d_3}{4}$$

In our example, evaluating this expression yields $\alpha = 1.75$. The area over the interval [0,6] has been conserved when going from the original data set to the updated data set. However, not all data sets are going to have only one imperfect prediction (i.e. with only one nonzero value of d). In the next section we demonstrate an approach to deal with more realistic data sets.

III. Non-Uniformly Spaced Points and Boundary Conditions

In the previous section we looked at the best case scenario – uniformly spaced points and only one nonzero d. While these assumptions ease the explanation of the predict and update steps of the discrete wavelet transform, it is highly unlikely that these conditions – especially the latter – would be present in practice. From this point forward, all derivations will be based on non-uniformly spaced data. The resulting expressions can easily be simplified to those for uniformly spaced data.

III.1 Predict

The linear predict step requires little change, as it is a weighted average of the neighboring even values:

$$d_{2i+1} = y_{2i+1} - \left[\left(y_{2i+2} - y_{2i}\right)\left(\frac{x_{2i+1} - x_{2i}}{x_{2i+2} - x_{2i}}\right) + y_{2i} \right]$$

Just as in the first section, the predict step is completed on each of the odd-indexed values in the data set. So far we have chosen data sets with an odd number of points. This is ideal for the predict step, as all odd indices have even neighbors to their right and left. It is also ideal for the update step, as adapting the update to an even number of points can cause conservation issues. When applying the discrete wavelet transform, it is likely that a set or subset (coarser iteration, as the complete transform is pyramidal in structure) will have an even number of points. Since we have a procedure that can accommodate non-uniform points, we will simply add a point between the last two points, with x and y values equal to the averages of those of the existing points. The transform can then continue as before.

III.2 Update

In Section II we looked at a special data set of size 7. We will soon look at a similar set, the only difference being that the points are non-uniformly spaced (d_1 and d_5 are still both 0). Let us first explain the role of this special set. The foundation of the linear update step is preserving the 0^{th} and 1^{st} moments of the data while going to a coarser level. If we were to use this directly on a set of size 7 with d_1, d_3, and d_5 all nonzero, we would have a system of six unknowns (many more in a large data set), and only two conditions to impose on them. If we try to impose more conditions by preserving the moments over each subinterval, we run into the problem of having many more conditions than we have variables. Instead, we will use the fact that we know how to find the update values for a set with only one imperfect update – we break up our data set into subintervals between even indices, keeping the above explanation in mind.

Figure 5 illustrates this idea using the same (uniformly spaced) sample data set shown in the second figure: x = {0,1,2,3,4,5,6} and y = {0,7,10,13,2,4,8}. The data is subdivided according to the 'region of influence' of each of the three differences in estimated values – d_1, d_2, and d_3. In the predict step, each odd prediction used the even values to either side of the odd value. Similarly, we must use the two neighboring odd values in each even update.

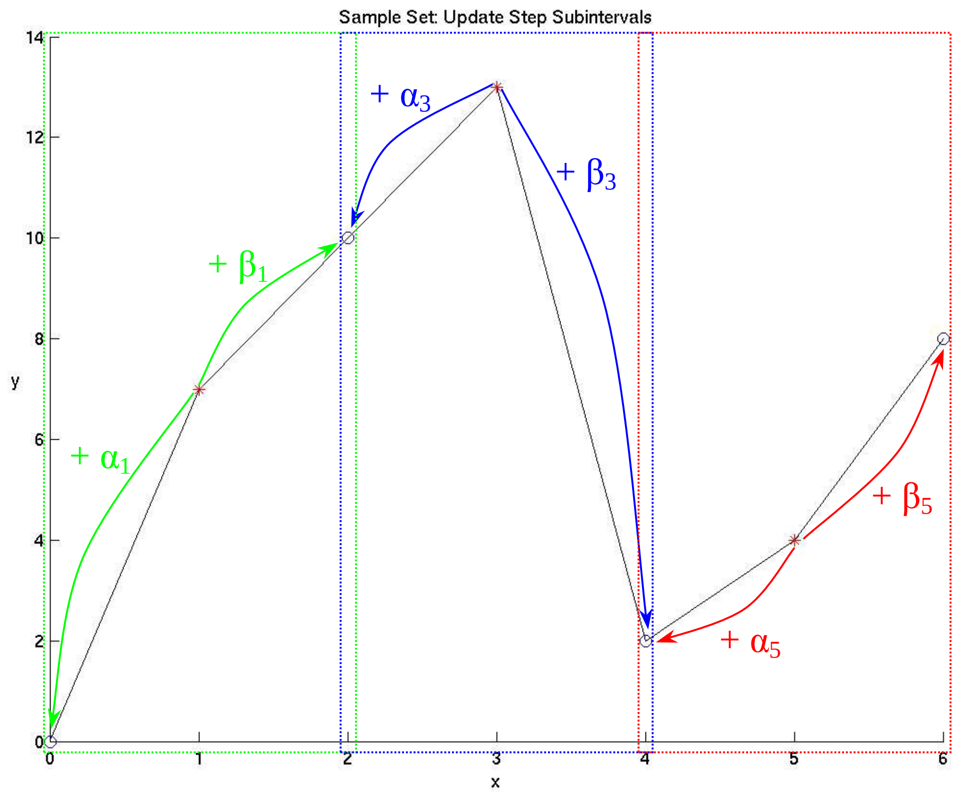

Figure 5. Illustration of Regions of Influence at the Update Step

Since y_0 and y_6 lie on boundaries, the effects of d_1 and d_5 are going to be slightly different. Let us first take indices 2 and 4 as examples in order to explore the effects of d_3. Since we are trying to average only the area on [2,4], we are going to make the assumption that $d_1 = d_5 = 0$. Using the equations of Section 2, adjusted to non-uniform data points, we get:

$$\int_{x_0}^{x_6} y dx = \frac{1}{2}(x_1 - x_0)(y_1 + y_0) + \frac{1}{2}(x_2 - x_1)(y_2 + y_1) + \ldots + \frac{1}{2}(x_6 - x_5)(y_6 + y_5)$$

$$\int_{x_0}^{x_6} u dx = \frac{1}{2}(x_2 - x_0)((y_2 + \alpha) + y_0) + \frac{1}{2}(x_4 - x_2)((y_4 + \beta) + (y_2 + \alpha)) + \frac{1}{2}(x_6 - x_4)(y_6 + (y_4 + \beta))$$

Since we now have two variables (due to non-uniform spacing), we must also use the equations for the 1st moment:

$$\int_{x_0}^{x_6} x\cdot y\,dx = \frac{(x_1-x_0)(x_0(2y_0+y_1)+x_1(y_0+2y_1))}{6} + \frac{(x_2-x_1)(x_1(2y_1+y_2)+x_2(y_1+2y_2))}{6} + \ldots + \frac{(x_6-x_5)(x_5(2y_5+y_6)+x_6(y_5+2y_6))}{6}$$

$$\int_{x_0}^{x_6} x\cdot u\,dx = \frac{(x_2-x_0)(x_0(2y_0+(y_2+\alpha))+x_2(y_0+2(y_2+\alpha)))}{6} + \frac{(x_4-x_2)(x_2(2(y_2+\alpha)+(y_4+\beta))+x_4((y_2+\alpha)+2(y_4+\beta)))}{6} + \frac{(x_6-x_4)(x_4(2(y_4+\beta)+y_6)+x_6((y_4+\beta)+2y_6))}{6}$$

Solving

$$\int_{x_0}^{x_6} y\,dx = \int_{x_0}^{x_6} u\,dx \qquad \text{and} \qquad \int_{x_0}^{x_6} x\cdot y\,dx = \int_{x_0}^{x_6} x\cdot u\,dx$$

for α, β gives

$$\alpha = \frac{d_3(x_4-x_2)(x_6-x_3)}{(x_4-x_0)(x_6-x_0)} \qquad \beta = \frac{d_3(x_3-x_0)(x_4-x_2)}{(x_6-x_0)(x_6-x_2)}$$

Notice that if we assume uniformly spaced points,

$$\alpha = \beta = \frac{d_3}{4},$$

just as we had in Section 2. We now have expressions for α and β –uniformly or non-uniformly spaced data – for the center case (no endpoint affected by the *d* value). We must now derive expressions for the update step that will distribute data properly when an endpoint is affected.

We can represent such a condition by taking a set of 5 points. We will assume that $d_3 = 0$, constructing a set in which the area change caused by d_1 can be distributed over only [0,4] – a left endpoint. The following figure depicts such a set.

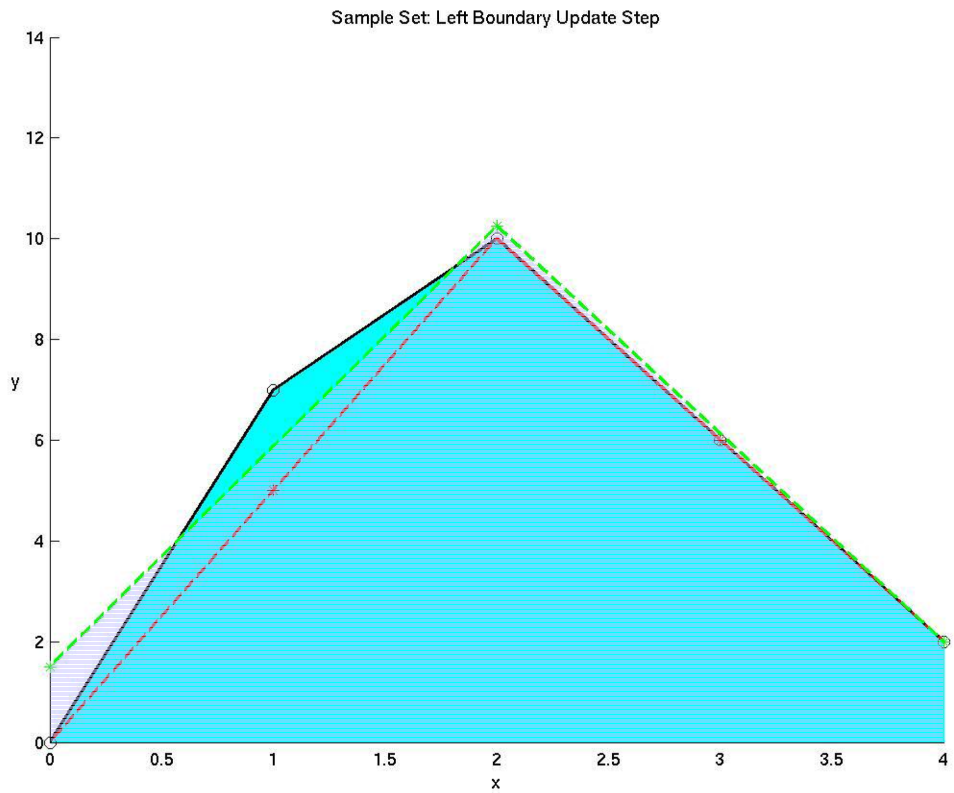

Figure 6. Left Endpoint Special Case Update

Similar to the center case, we will set up a system of equations that conserve the 0^{th} and 1^{st} moments over the update step.

$$\int_{x_0}^{x_4} y dx = \frac{1}{2}(x_1 - x_0)(y_1 + y_0) + \frac{1}{2}(x_2 - x_1)(y_2 + y_1) + ... + \frac{1}{2}(x_4 - x_3)(y_4 + y_3)$$

$$\int_{x_0}^{x_4} u dx = \frac{1}{2}(x_2 - x_0)((y_2 + \beta) + (y_0 + \alpha)) + \frac{1}{2}(x_4 - x_2)(y_4 + (y_2 + \beta))$$

$$\int_{x_0}^{x_4} x \cdot y dx = \frac{(x_1 - x_0)(x_0(2y_0 + y_1) + x_1(y_0 + 2y_1))}{6} + \frac{(x_2 - x_1)(x_1(2y_1 + y_2) + x_2(y_1 + 2y_2))}{6} + \ldots +$$

$$\frac{(x_4 - x_3)(x_3(2y_3 + y_4) + x_4(y_3 + 2y_4))}{6}$$

$$\int_{x_0}^{x_4} x \cdot u dx = \frac{(x_2 - x_0)(x_0(2(y_0 + \alpha) + (y_2 + \beta)) + x_2((y_0 + \alpha) + 2(y_2 + \beta)))}{6} +$$

$$\frac{(x_4 - x_2)(x_2(2(y_2 + \beta) + y_4) + x_4((y_2 + \beta) + 2y_4))}{6}$$

Solving

$$\int_{x_0}^{x_4} y dx = \int_{x_0}^{x_4} u dx \qquad \text{and} \qquad \int_{x_0}^{x_4} x \cdot y dx = \int_{x_0}^{x_4} x \cdot u dx$$

for α, β gives

$$\alpha = \frac{d_1(x_4 - x_1)}{(x_4 - x_0)} \qquad \beta = \frac{d_1(x_1 - x_0)(x_2 - x_0)}{(x_4 - x_0)^2}$$

which with uniform points simplifies to

$$\alpha = \frac{3d_1}{4} \qquad \beta = \frac{d_1}{8}$$

The right boundary case simply mirrors the left. The right boundary for uniform points, for example, results in the α value of (d/8) for the inner even point and the β value of (3d/4) for the outer even point (boundary point).

We now have expressions for non-uniform center points, left end points, and right end points. In the following general forms, for each d_i, α_i is the value of the change in the left even neighbor, and β_i is the change in the right even neighbor:

Center Points:

$$\alpha_i = \frac{d_i(x_{i+1} - x_{i-1})(x_{i+3} - x_i)}{(x_{i+1} - x_{i-3})(x_{i+3} - x_{i-3})} \qquad \beta_i = \frac{d_i(x_i - x_{i-3})(x_{i+1} - x_{i-1})}{(x_{i+3} - x_{i-3})(x_{i+3} - x_{i-1})}$$

Left End Points (i=1 for 0-based indexing):

$$\alpha_i = \frac{d_i\left(x_{i+3} - x_i\right)}{\left(x_{i+3} - x_{i-1}\right)} \qquad \beta_i = \frac{d_i\left(x_i - x_{i-1}\right)\left(x_{i+1} - x_{i-1}\right)}{\left(x_{i+3} - x_{i-1}\right)^2}$$

Right End Points:

$$\alpha_i = \frac{d_i\left(x_{i+1} - x_{i-1}\right)\left(x_{i+1} - x_i\right)}{\left(x_{i+1} - x_{i-3}\right)^2} \qquad \beta_i = \frac{d_i\left(x_i - x_{i-3}\right)}{\left(x_{i+1} - x_{i-3}\right)}$$

We now have a very good picture of how the linear update step is built. The same concept can be applied to any order polynomial, although depending on the application there may or may not be a benefit for using higher order bases. Recall our example set:

$$x = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix} \qquad y = \begin{bmatrix} 0 \\ 7 \\ 10 \\ 13 \\ 2 \\ 4 \\ 8 \end{bmatrix}$$

Figure 7. Sample Data Set

We know that for each *d* (difference of predicted odd) value, there are left and right even neighbors that are affected – the boundaries being the exception to the rule. Thus, for our 7 data point example we know that:

$$s_0 = y_0 + \alpha_1 = 0 + 2 \cdot \frac{3}{4} = 1.5 \qquad s_2 = y_2 + \beta_1 + \alpha_3 = 10 + \frac{1}{4} + \frac{7}{4} = 12$$

$$s_4 = y_4 + \beta_3 + \alpha_5 = 2 + \frac{7}{4} + \frac{-1}{8} = 3.625 \qquad s_6 = y_6 + \beta_5 = 8 + \frac{-3}{4} = 7.25$$

where s_i is the updated y-value at index *i*.* The following figure shows the completed update step on the sample set. Like Figure 4, the original data is plotted in black, the predict step in red, and the updated data in green. The area under the original data is shown in cyan, and that under the updated data is shown in purple.

* *s* refers to “smoothing”, where *d* originates from “detail”, the roles the two steps play in the wavelet transform.

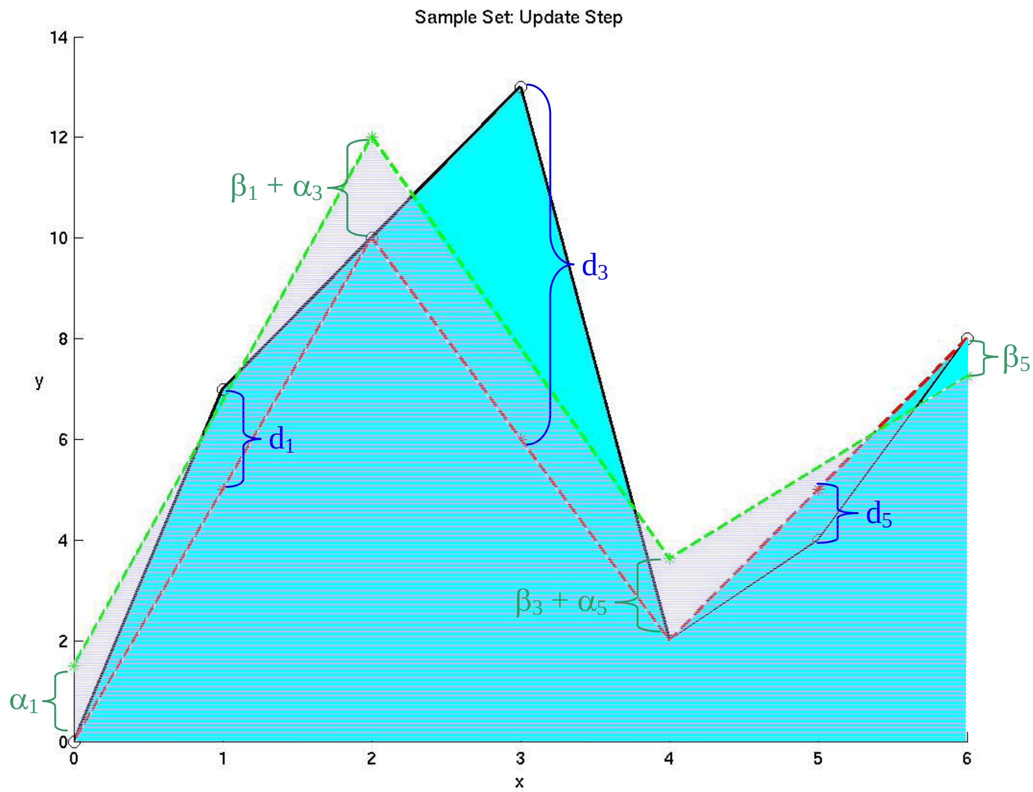

Figure 8. Completed Update Step on Sample Set

At this point we have completed one iteration of the linear discrete wavelet transform. In order to finish the transform, we must continue to perform iterations on coarser meshes over the domain until only three or four points remain. This is slightly different from the Haar construction, where only two points remain at completion. Each successive iteration is completed using a mesh constructed of only the updated points of the last iteration. As a result, the size of each successive mesh is equal to the ceiling of the size of the previous mesh divided by two.

IV. Application

While the main goal of this paper is to provide a simple explanation of the lifting approach for wavelet transformations for non-uniformly spaced data, the above approach has been tested on 6 astrophysical opacity data sets with a view to exploring the possibility of compressing the data. The behavior of the functions ranged from relatively smooth to strongly jagged. Each set was transformed with both Haar and linear basis functions and the change in the Rosseland mean was used as a criterion for dropping wavelet coefficients. On the smoother data sets, it was possible to get high levels of compression while preserving the Rosseland mean, (about 99.5% compression with only

10^{-5} difference in Rosseland means); however, on less smooth data sets, the compression was sharply reduced (about 51% compression with a 10^{-5} difference in Rosseland means). Linear basis functions performed better for the smoother data sets while preserving the Rosseland means to high accuracy, but for the less smooth data sets, Haar basis functions performed better. It remains to be seen as to whether entirely different basis functions specially tuned for opacity data sets can be used to obtain better results.

Acknowledgements

I would like to thank Vijay Sonnad, Carlos Iglesias and Brian Wilson for their assistance in clarifying several points at various stages of this work. Bill Isaacs provided some well chosen data sets that proved very useful in verifying the approach. Mark Duchaineau pointed out the many benefits of the lifting approach and provided important insights.

References

[1] W. Sweldens and P. Schröder. "Building Your Own Wavelets at Home." *Wavelets in Computer Graphics*. ACM SIGGRAPH course notes, 1996. pp. 15-87.

[2] I. Daubechies and W. Sweldens. "Factoring Wavelet Transforms into Lifting Steps." *Fourier Analysis and Applications*, 4(3), 1998. pp. 247-269.