

THE LIFTING METHOD

Patrick J. Van Fleet

Center for Applied Mathematics
University of St. Thomas
St. Paul, MN USA

PREP - Wavelet Workshop, 2007



UNIVERSITY of ST. THOMAS

OUTLINE

TODAY'S SCHEDULE

The LeGall Transform

Mapping Integers to Integers

LIFTING

Basic Construction

Inversion

Integers To Integers



TODAY'S SCHEDULE

9:00-10:15 ⇒ **Lecture Nine:** The Lifting Method

10:15-10:30 **Coffee Break** (OSS 235)

10:30-11:45 **Computer Session Five:** Computer Session:
JPEG2000

11:45-12:00 Evaluations/Wrap Up

12:00-1:00 **Lunch** (Cafeteria)



- ▶ The (5, 3) biorthogonal spline filter is

$$\mathbf{h} = \left(-\frac{\sqrt{2}}{8}, \frac{\sqrt{2}}{4}, \frac{3\sqrt{2}}{4}, \frac{\sqrt{2}}{4}, -\frac{\sqrt{2}}{8}\right) \quad \text{and} \quad \tilde{\mathbf{h}} = \left(\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{4}\right)$$

- ▶ For use with JPEG2000, the 5-term filter is multiplied by $\sqrt{2}/2$ and the 3-term filter is multiplied by $\sqrt{2}$. Moreover, the filters are “reversed” - the 5-term filter is used to build \tilde{W}_N .
- ▶ Thus we define the **LeGall filter** (due to Didier LeGall) by

$$\tilde{\mathbf{h}} = \left(-\frac{1}{8}, \frac{1}{4}, \frac{3}{4}, \frac{1}{4}, -\frac{1}{8}\right) \quad \text{and} \quad \mathbf{h} = \left(\frac{1}{2}, 1, \frac{1}{2}\right)$$



- ▶ The (5, 3) biorthogonal spline filter is

$$\mathbf{h} = \left(-\frac{\sqrt{2}}{8}, \frac{\sqrt{2}}{4}, \frac{3\sqrt{2}}{4}, \frac{\sqrt{2}}{4}, -\frac{\sqrt{2}}{8}\right) \quad \text{and} \quad \tilde{\mathbf{h}} = \left(\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{4}\right)$$

- ▶ For use with JPEG2000, the 5-term filter is multiplied by $\sqrt{2}/2$ and the 3-term filter is multiplied by $\sqrt{2}$. Moreover, the filters are “reversed” - the 5-term filter is used to build \tilde{W}_N .
- ▶ Thus we define the **LeGall filter** (due to Didier LeGall) by

$$\tilde{\mathbf{h}} = \left(-\frac{1}{8}, \frac{1}{4}, \frac{3}{4}, \frac{1}{4}, -\frac{1}{8}\right) \quad \text{and} \quad \mathbf{h} = \left(\frac{1}{2}, 1, \frac{1}{2}\right)$$



- ▶ The (5, 3) biorthogonal spline filter is

$$\mathbf{h} = \left(-\frac{\sqrt{2}}{8}, \frac{\sqrt{2}}{4}, \frac{3\sqrt{2}}{4}, \frac{\sqrt{2}}{4}, -\frac{\sqrt{2}}{8}\right) \quad \text{and} \quad \tilde{\mathbf{h}} = \left(\frac{\sqrt{2}}{4}, \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{4}\right)$$

- ▶ For use with JPEG2000, the 5-term filter is multiplied by $\sqrt{2}/2$ and the 3-term filter is multiplied by $\sqrt{2}$. Moreover, the filters are “reversed” - the 5-term filter is used to build \tilde{W}_N .
- ▶ Thus we define the **LeGall filter** (due to Didier LeGall) by

$$\tilde{\mathbf{h}} = \left(-\frac{1}{8}, \frac{1}{4}, \frac{3}{4}, \frac{1}{4}, -\frac{1}{8}\right) \quad \text{and} \quad \mathbf{h} = \left(\frac{1}{2}, 1, \frac{1}{2}\right)$$



The LeGall wavelet transform for a 10-vector is:

$$\tilde{W}_{10} = \begin{pmatrix} \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{8} & \frac{1}{4} \\ -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 \\ -\frac{1}{8} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} \\ \hline -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 \\ -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 1 \end{pmatrix}$$



The transpose of the inverse is

$$W_{10}^T = \begin{bmatrix} 1 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 \\ \hline -\frac{1}{4} & \frac{3}{4} & -\frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{8} \\ 0 & -\frac{1}{8} & -\frac{1}{4} & \frac{3}{4} & -\frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{8} & -\frac{1}{4} & \frac{3}{4} & -\frac{1}{4} & -\frac{1}{8} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{8} & -\frac{1}{4} & \frac{3}{4} & -\frac{1}{4} & -\frac{1}{8} \\ -\frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{8} & -\frac{1}{4} & \frac{3}{4} \end{bmatrix}$$



- ▶ This transform still has “wrapping” problems.
- ▶ We can exploit symmetry to modify the transform.



- ▶ This transform still has “wrapping” problems.
- ▶ We can exploit symmetry to modify the transform.



We modify the blue values:

$$\tilde{W}_{10} = \begin{pmatrix} \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{8} & \frac{1}{4} \\ -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 \\ -\frac{1}{8} & 0 & 0 & 0 & 0 & 0 & -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} \\ \hline -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 \\ -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 1 \end{pmatrix}$$



We modify the blue values:

$$\tilde{W}_{10} = \begin{pmatrix} \frac{3}{4} & \frac{1}{2} & -\frac{1}{4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{8} & \frac{1}{4} & \frac{5}{8} & \frac{1}{4} \\ \hline -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$



- ▶ Encoding methods work best if the transform maps integers to integers.
- ▶ The matrix is comprised of rational numbers - not likely that they will map integers to integers.
- ▶ The transform does do a good job handling edge effects - if we compute $W_{10}\mathbf{v}$ where $\mathbf{v} = (v_1, v_2, \dots, v_{10})$ we obtain



- ▶ Encoding methods work best if the transform maps integers to integers.
- ▶ The matrix is comprised of rational numbers - not likely that they will map integers to integers.
- ▶ The transform does do a good job handling edge effects - if we compute $W_{10}\mathbf{v}$ where $\mathbf{v} = (v_1, v_2, \dots, v_{10})$ we obtain



- ▶ Encoding methods work best if the transform maps integers to integers.
- ▶ The matrix is comprised of rational numbers - not likely that they will map integers to integers.
- ▶ The transform does do a good job handling edge effects - if we compute $W_{10}\mathbf{v}$ where $\mathbf{v} = (v_1, v_2, \dots, v_{10})$ we obtain



$$W_{10}\mathbf{v} = \begin{pmatrix} \frac{3}{4}v_1 + \frac{1}{2}v_2 - \frac{1}{4}v_3 \\ -\frac{1}{8}v_1 + \frac{1}{4}v_2 + \frac{3}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5 \\ -\frac{1}{8}v_3 + \frac{1}{4}v_4 + \frac{3}{4}v_5 + \frac{1}{4}v_6 - \frac{1}{8}v_7 \\ -\frac{1}{8}v_5 + \frac{1}{4}v_6 + \frac{3}{4}v_7 + \frac{1}{4}v_8 - \frac{1}{8}v_9 \\ -\frac{1}{8}v_7 + \frac{1}{4}v_8 + \frac{5}{8}v_9 + \frac{1}{4}v_{10} \\ \hline -\frac{1}{2}v_1 + v_2 - \frac{1}{2}v_3 \\ -\frac{1}{2}v_3 + v_4 - \frac{1}{2}v_5 \\ -\frac{1}{2}v_5 + v_6 - \frac{1}{2}v_7 \\ -\frac{1}{2}v_7 + v_8 - \frac{1}{2}v_9 \\ -v_9 + v_{10} \end{pmatrix}$$

There is some hidden symmetry here:



$$W_{10}\mathbf{v} = \begin{pmatrix} -\frac{1}{8}v_3 + \frac{1}{4}v_2 + \frac{3}{4}v_1 + \frac{1}{4}v_2 - \frac{1}{8}v_3 \\ -\frac{1}{8}v_1 + \frac{1}{4}v_2 + \frac{3}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5 \\ -\frac{1}{8}v_3 + \frac{1}{4}v_4 + \frac{3}{4}v_5 + \frac{1}{4}v_6 - \frac{1}{8}v_7 \\ -\frac{1}{8}v_5 + \frac{1}{4}v_6 + \frac{3}{4}v_7 + \frac{1}{4}v_8 - \frac{1}{8}v_9 \\ -\frac{1}{8}v_7 + \frac{1}{4}v_8 + \frac{3}{4}v_9 + \frac{1}{4}v_{10} - \frac{1}{8}v_9 \\ \hline -\frac{1}{2}v_1 + v_2 - \frac{1}{2}v_3 \\ -\frac{1}{2}v_3 + v_4 - \frac{1}{2}v_5 \\ -\frac{1}{2}v_5 + v_6 - \frac{1}{2}v_7 \\ -\frac{1}{2}v_7 + v_8 - \frac{1}{2}v_9 \\ -\frac{1}{2}v_9 + v_{10} - \frac{1}{2}v_9 \end{pmatrix}$$

Note the top is a weighted moving average, the bottom computes differences.



- ▶ How to modify the Le Gall Wavelet Transform so that it maps integers to integers?
- ▶ We could multiply the transform by 8, but that increases the range of the output and that's not always best for encoders.
- ▶ We will see that by constructing a fast algorithm for implementing W_N we can find a way to modify the transform so that it maps integers to integers **AND** is invertible!
- ▶ This algorithm is the one used by JPEG2000 to perform lossless image compression.



- ▶ How to modify the Le Gall Wavelet Transform so that it maps integers to integers?
- ▶ We could multiply the transform by 8, but that increases the range of the output and that's not always best for encoders.
- ▶ We will see that by constructing a fast algorithm for implementing W_N we can find a way to modify the transform so that it maps integers to integers **AND** is invertible!
- ▶ This algorithm is the one used by JPEG2000 to perform lossless image compression.



- ▶ How to modify the Le Gall Wavelet Transform so that it maps integers to integers?
- ▶ We could multiply the transform by 8, but that increases the range of the output and that's not always best for encoders.
- ▶ We will see that by constructing a fast algorithm for implementing W_N we can find a way to modify the transform so that it maps integers to integers **AND** is invertible!
- ▶ This algorithm is the one used by JPEG2000 to perform lossless image compression.



- ▶ How to modify the Le Gall Wavelet Transform so that it maps integers to integers?
- ▶ We could multiply the transform by 8, but that increases the range of the output and that's not always best for encoders.
- ▶ We will see that by constructing a fast algorithm for implementing W_N we can find a way to modify the transform so that it maps integers to integers **AND** is invertible!
- ▶ This algorithm is the one used by JPEG2000 to perform lossless image compression.



Let's return to the computation $W_{10}\mathbf{v}$. The output is

$$W_{10}\mathbf{v} = \begin{pmatrix} -\frac{1}{8}v_3 + \frac{1}{4}v_2 + \frac{3}{4}v_1 + \frac{1}{4}v_2 - \frac{1}{8}v_3 \\ -\frac{1}{8}v_1 + \frac{1}{4}v_2 + \frac{3}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5 \\ -\frac{1}{8}v_3 + \frac{1}{4}v_4 + \frac{3}{4}v_5 + \frac{1}{4}v_6 - \frac{1}{8}v_7 \\ -\frac{1}{8}v_5 + \frac{1}{4}v_6 + \frac{3}{4}v_7 + \frac{1}{4}v_8 - \frac{1}{8}v_9 \\ -\frac{1}{8}v_7 + \frac{1}{4}v_8 + \frac{3}{4}v_9 + \frac{1}{4}v_{10} - \frac{1}{8}v_9 \\ \hline -\frac{1}{2}v_1 + v_2 - \frac{1}{2}v_3 \\ -\frac{1}{2}v_3 + v_4 - \frac{1}{2}v_5 \\ -\frac{1}{2}v_5 + v_6 - \frac{1}{2}v_7 \\ -\frac{1}{2}v_7 + v_8 - \frac{1}{2}v_9 \\ -\frac{1}{2}v_9 + v_{10} - \frac{1}{2}v_9 \end{pmatrix}$$

Let's look at the bottom half of the output:



$$d_1 = -\frac{1}{2} v_1 + \mathbf{v}_2 - \frac{1}{2} v_3$$

$$d_2 = -\frac{1}{2} v_3 + \mathbf{v}_4 - \frac{1}{2} v_5$$

$$d_3 = -\frac{1}{2} v_5 + \mathbf{v}_6 - \frac{1}{2} v_7$$

$$d_4 = -\frac{1}{2} v_7 + \mathbf{v}_8 - \frac{1}{2} v_9$$

$$d_5 = -\frac{1}{2} v_9 + \mathbf{v}_{10} - \frac{1}{2} v_9$$

- ▶ If we form “even” and “odd” vectors,

$$\mathbf{e} = (e_1, e_2, e_3, e_4, e_5)^T = (v_2, v_4, v_6, v_8, v_{10})^T$$

and

$$\mathbf{o} = (o_1, o_2, o_3, o_4, o_5)^T = (v_1, v_3, v_5, v_7, v_9)^T$$

we can write

- ▶ $d_k = e_k - \frac{1}{2}(o_{k+1} + o_k)$, where $o_6 := o_5$.



$$d_1 = -\frac{1}{2} v_1 + \mathbf{v}_2 - \frac{1}{2} v_3$$

$$d_2 = -\frac{1}{2} v_3 + \mathbf{v}_4 - \frac{1}{2} v_5$$

$$d_3 = -\frac{1}{2} v_5 + \mathbf{v}_6 - \frac{1}{2} v_7$$

$$d_4 = -\frac{1}{2} v_7 + \mathbf{v}_8 - \frac{1}{2} v_9$$

$$d_5 = -\frac{1}{2} v_9 + \mathbf{v}_{10} - \frac{1}{2} v_9$$

- ▶ If we form “even” and “odd” vectors,

$$\mathbf{e} = (e_1, e_2, e_3, e_4, e_5)^T = (v_2, v_4, v_6, v_8, v_{10})^T$$

and

$$\mathbf{o} = (o_1, o_2, o_3, o_4, o_5)^T = (v_1, v_3, v_5, v_7, v_9)^T$$

we can write

- ▶ $d_k = e_k - \frac{1}{2}(o_{k+1} + o_k)$, where $o_6 := o_5$.



$$d_1 = -\frac{1}{2} v_1 + \mathbf{v}_2 - \frac{1}{2} v_3$$

$$d_2 = -\frac{1}{2} v_3 + \mathbf{v}_4 - \frac{1}{2} v_5$$

$$d_3 = -\frac{1}{2} v_5 + \mathbf{v}_6 - \frac{1}{2} v_7$$

$$d_4 = -\frac{1}{2} v_7 + \mathbf{v}_8 - \frac{1}{2} v_9$$

$$d_5 = -\frac{1}{2} v_9 + \mathbf{v}_{10} - \frac{1}{2} v_9$$

- ▶ If we form “even” and “odd” vectors,

$$\mathbf{e} = (e_1, e_2, e_3, e_4, e_5)^T = (v_2, v_4, v_6, v_8, v_{10})^T$$

and

$$\mathbf{o} = (o_1, o_2, o_3, o_4, o_5)^T = (v_1, v_3, v_5, v_7, v_9)^T$$

we can write

- ▶ $d_k = e_k - \frac{1}{2}(o_{k+1} + o_k)$, where $o_6 := o_5$.



- ▶ The so-called *lifting step* (due to Wim Sweldens) is evident when we make the observation that the sum of d_1 and d_2 is related to s_2 . If we compute $d_1 + d_2$, we have



$$\begin{aligned}
 d_1 + d_2 &= e_1 - \frac{1}{2}(o_1 + o_2) + e_2 - \frac{1}{2}(o_2 + o_3) \\
 &= v_2 - \frac{1}{2}(v_1 + v_3) + v_4 - \frac{1}{2}(v_3 + v_5) \\
 &= -\frac{1}{2}v_1 + v_2 - v_3 + v_4 - \frac{1}{2}v_5
 \end{aligned}$$

- ▶ The result is “close” to

$$s_2 = -\frac{1}{8}v_1 + \frac{1}{4}v_2 + \frac{3}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5$$



- ▶ The so-called *lifting step* (due to Wim Sweldens) is evident when we make the observation that the sum of d_1 and d_2 is related to s_2 . If we compute $d_1 + d_2$, we have



$$\begin{aligned}
 d_1 + d_2 &= e_1 - \frac{1}{2}(o_1 + o_2) + e_2 - \frac{1}{2}(o_2 + o_3) \\
 &= v_2 - \frac{1}{2}(v_1 + v_3) + v_4 - \frac{1}{2}(v_3 + v_5) \\
 &= -\frac{1}{2}v_1 + v_2 - v_3 + v_4 - \frac{1}{2}v_5
 \end{aligned}$$

- ▶ The result is “close” to

$$s_2 = -\frac{1}{8}v_1 + \frac{1}{4}v_2 + \frac{3}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5$$



- ▶ The so-called *lifting step* (due to Wim Sweldens) is evident when we make the observation that the sum of d_1 and d_2 is related to s_2 . If we compute $d_1 + d_2$, we have



$$\begin{aligned}
 d_1 + d_2 &= e_1 - \frac{1}{2}(o_1 + o_2) + e_2 - \frac{1}{2}(o_2 + o_3) \\
 &= v_2 - \frac{1}{2}(v_1 + v_3) + v_4 - \frac{1}{2}(v_3 + v_5) \\
 &= -\frac{1}{2}v_1 + v_2 - v_3 + v_4 - \frac{1}{2}v_5
 \end{aligned}$$

- ▶ The result is “close” to

$$s_2 = -\frac{1}{8}v_1 + \frac{1}{4}v_2 + \frac{3}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5$$



- ▶ Indeed if we divide the sum by 4 and manipulate the term containing v_3 , we see that



$$\begin{aligned}
 \frac{d_1 + d_2}{4} &= -\frac{1}{8}v_1 + \frac{1}{4}v_2 - \frac{1}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5 \\
 &= \left(-\frac{1}{8}v_1 + \frac{1}{4}v_2 + \frac{3}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5\right) - v_3 \\
 &= s_2 - v_3 \\
 &= s_2 - o_2
 \end{aligned}$$

- ▶ Solving for s_2 gives

$$s_2 = o_2 + \frac{1}{4}(d_1 + d_2)$$

- ▶ In a similar manner, we can show that

$$s_k = o_k + \frac{1}{4}(d_k + d_{k-1}), \quad k = 2, 3, 4, 5$$



- ▶ Indeed if we divide the sum by 4 and manipulate the term containing v_3 , we see that



$$\begin{aligned}
 \frac{d_1 + d_2}{4} &= -\frac{1}{8}v_1 + \frac{1}{4}v_2 - \frac{1}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5 \\
 &= \left(-\frac{1}{8}v_1 + \frac{1}{4}v_2 + \frac{3}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5\right) - v_3 \\
 &= s_2 - v_3 \\
 &= s_2 - o_2
 \end{aligned}$$

- ▶ Solving for s_2 gives

$$s_2 = o_2 + \frac{1}{4}(d_1 + d_2)$$

- ▶ In a similar manner, we can show that

$$s_k = o_k + \frac{1}{4}(d_k + d_{k-1}), \quad k = 2, 3, 4, 5$$



- ▶ Indeed if we divide the sum by 4 and manipulate the term containing v_3 , we see that



$$\begin{aligned}
 \frac{d_1 + d_2}{4} &= -\frac{1}{8}v_1 + \frac{1}{4}v_2 - \frac{1}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5 \\
 &= \left(-\frac{1}{8}v_1 + \frac{1}{4}v_2 + \frac{3}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5\right) - v_3 \\
 &= s_2 - v_3 \\
 &= s_2 - o_2
 \end{aligned}$$

- ▶ Solving for s_2 gives

$$s_2 = o_2 + \frac{1}{4}(d_1 + d_2)$$

- ▶ In a similar manner, we can show that

$$s_k = o_k + \frac{1}{4}(d_k + d_{k-1}), \quad k = 2, 3, 4, 5$$



- ▶ Indeed if we divide the sum by 4 and manipulate the term containing v_3 , we see that



$$\begin{aligned}
 \frac{d_1 + d_2}{4} &= -\frac{1}{8}v_1 + \frac{1}{4}v_2 - \frac{1}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5 \\
 &= \left(-\frac{1}{8}v_1 + \frac{1}{4}v_2 + \frac{3}{4}v_3 + \frac{1}{4}v_4 - \frac{1}{8}v_5\right) - v_3 \\
 &= s_2 - v_3 \\
 &= s_2 - o_2
 \end{aligned}$$

- ▶ Solving for s_2 gives

$$s_2 = o_2 + \frac{1}{4}(d_1 + d_2)$$

- ▶ In a similar manner, we can show that

$$s_k = o_k + \frac{1}{4}(d_k + d_{k-1}), \quad k = 2, 3, 4, 5$$



- ▶ Some special consideration is needed for s_1 . We can write

$$\begin{aligned} s_1 &= -\frac{1}{8}v_3 + \frac{1}{4}v_2 + \frac{3}{4}v_1 + \frac{1}{4}v_2 - \frac{1}{8}v_3 \\ &= -\frac{1}{4}v_3 + \frac{1}{2}v_2 + \frac{3}{4}v_1 \\ &= \left(-\frac{1}{4}v_3 + \frac{1}{2}v_2 - \frac{1}{4}v_1\right) + v_1 \\ &= \frac{1}{4}(-v_3 + 2v_2 - v_1) + v_1 \\ &= \frac{1}{4}(d_1 + d_1) + v_1 \end{aligned}$$



- ▶ Some special consideration is needed for s_1 . We can write

$$\begin{aligned} s_1 &= -\frac{1}{8}v_3 + \frac{1}{4}v_2 + \frac{3}{4}v_1 + \frac{1}{4}v_2 - \frac{1}{8}v_3 \\ &= -\frac{1}{4}v_3 + \frac{1}{2}v_2 + \frac{3}{4}v_1 \\ &= \left(-\frac{1}{4}v_3 + \frac{1}{2}v_2 - \frac{1}{4}v_1\right) + v_1 \\ &= \frac{1}{4}(-v_3 + 2v_2 - v_1) + v_1 \\ &= \frac{1}{4}(d_1 + d_1) + v_1 \end{aligned}$$



- ▶ In general, we first compute

$$d_k = e_k - \frac{1}{2}(o_k + o_{k+1}), \quad k = 1, \dots, N/2$$

where $o_{\frac{N}{2}+1} := o_{N/2}$, and use this result to compute



$$s_k = o_k + \frac{1}{4}(d_k + d_{k-1}), \quad k = 1, \dots, N/2$$

where $d_0 := d_1$.



- ▶ In general, we first compute

$$d_k = e_k - \frac{1}{2}(o_k + o_{k+1}), \quad k = 1, \dots, N/2$$

where $o_{\frac{N}{2}+1} := o_{N/2}$, and use this result to compute

- ▶
$$s_k = o_k + \frac{1}{4}(d_k + d_{k-1}), \quad k = 1, \dots, N/2$$

where $d_0 := d_1$.



- ▶ Inverting is trivial. Given \mathbf{s} and \mathbf{d} , we first solve for \mathbf{o} :

$$o_k = s_k - \frac{1}{4}(d_k + d_{k-1}), \quad k = 1, \dots, N/2$$

- ▶ Once we have \mathbf{s} , we can easily solve for \mathbf{d} to get \mathbf{e} :

$$e_k = d_k + \frac{1}{2}(o_k + o_{k+1}), \quad k = 1, \dots, N/2$$



- ▶ Inverting is trivial. Given \mathbf{s} and \mathbf{d} , we first solve for \mathbf{o} :

$$o_k = s_k - \frac{1}{4}(d_k + d_{k-1}), \quad k = 1, \dots, N/2$$

- ▶ Once we have \mathbf{s} , we can easily solve for \mathbf{d} to get \mathbf{e} :

$$e_k = d_k + \frac{1}{2}(o_k + o_{k+1}), \quad k = 1, \dots, N/2$$



To map integers to integers we change

$$d_k = e_k - \frac{1}{2}(o_k + o_{k+1}), \quad k = 1, \dots, N/2$$

and

$$s_k = o_k + \frac{1}{4}(d_k + d_{k-1}), \quad k = 1, \dots, N/2$$

to

$$d_k^* = e_k - \lfloor \frac{1}{2}(o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2$$

and

$$s_k^* = o_k + \lfloor \frac{1}{4}(d_k^* + d_{k-1}^*) + \frac{1}{2} \rfloor, \quad k = 1, \dots, N/2$$

- ▶ The $+\frac{1}{2}$ prevents bias.



To map integers to integers we change

$$d_k = e_k - \frac{1}{2}(o_k + o_{k+1}), \quad k = 1, \dots, N/2$$

and

$$s_k = o_k + \frac{1}{4}(d_k + d_{k-1}), \quad k = 1, \dots, N/2$$

to

$$d_k^* = e_k - \lfloor \frac{1}{2}(o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2$$

and

$$s_k^* = o_k + \lfloor \frac{1}{4}(d_k^* + d_{k-1}^*) + \frac{1}{2} \rfloor, \quad k = 1, \dots, N/2$$

► The $+\frac{1}{2}$ prevents bias.



To map integers to integers we change

$$d_k = e_k - \frac{1}{2}(o_k + o_{k+1}), \quad k = 1, \dots, N/2$$

and

$$s_k = o_k + \frac{1}{4}(d_k + d_{k-1}), \quad k = 1, \dots, N/2$$

to

$$d_k^* = e_k - \lfloor \frac{1}{2}(o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2$$

and

$$s_k^* = o_k + \lfloor \frac{1}{4}(d_k^* + d_{k-1}^*) + \frac{1}{2} \rfloor, \quad k = 1, \dots, N/2$$

- ▶ The $+\frac{1}{2}$ prevents bias.



- ▶ Certainly d_k^* and s_k^* are integers and we haven't altered the range of the output by much.
- ▶ We can also exactly recover o_k since

$$o_k = s_k^* - \lfloor \frac{1}{4} (d_k^* + d_{k-1}^*) + \frac{1}{2} \rfloor, \quad k = 1, \dots, N/2$$

- ▶ Can we exactly recover the e_k from

$$d_k^* = e_k - \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2?$$

- ▶ Let's use e_k^* until we are sure it is e_k . We have

$$e_k^* = d_k^* + \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2$$



- ▶ Certainly d_k^* and s_k^* are integers and we haven't altered the range of the output by much.
- ▶ We can also exactly recover o_k since

$$o_k = s_k^* - \lfloor \frac{1}{4} (d_k^* + d_{k-1}^*) + \frac{1}{2} \rfloor, \quad k = 1, \dots, N/2$$

- ▶ Can we exactly recover the e_k from

$$d_k^* = e_k - \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2?$$

- ▶ Let's use e_k^* until we are sure it is e_k . We have

$$e_k^* = d_k^* + \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2$$



- ▶ Certainly d_k^* and s_k^* are integers and we haven't altered the range of the output by much.
- ▶ We can also exactly recover o_k since

$$o_k = s_k^* - \lfloor \frac{1}{4} (d_k^* + d_{k-1}^*) + \frac{1}{2} \rfloor, \quad k = 1, \dots, N/2$$

- ▶ Can we exactly recover the e_k from

$$d_k^* = e_k - \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2?$$

- ▶ Let's use e_k^* until we are sure it is e_k . We have

$$e_k^* = d_k^* + \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2$$



- ▶ Certainly d_k^* and s_k^* are integers and we haven't altered the range of the output by much.
- ▶ We can also exactly recover o_k since

$$o_k = s_k^* - \lfloor \frac{1}{4} (d_k^* + d_{k-1}^*) + \frac{1}{2} \rfloor, \quad k = 1, \dots, N/2$$

- ▶ Can we exactly recover the e_k from

$$d_k^* = e_k - \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2?$$

- ▶ Let's use e_k^* until we are sure it is e_k . We have

$$e_k^* = d_k^* + \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor, \quad k = 1, \dots, N/2$$



- ▶ The key observation is that

$$\lfloor \frac{1}{2}(o_k + o_{k+1}) \rfloor = \begin{cases} \frac{1}{2}(o_k + o_{k+1}), & o_k + o_{k+1} \text{ even} \\ \frac{1}{2}(o_k + o_{k+1}) - \frac{1}{2}, & o_k + o_{k+1} \text{ odd} \end{cases}$$



► So

$$d_k^* = e_k - \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor = \begin{cases} d_k, & o_k + o_{k+1} \text{ even} \\ d_k + \frac{1}{2}, & o_k + o_{k+1} \text{ odd} \end{cases}$$

and

►

$$\begin{aligned} e_k^* &= d_k^* + \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor \\ &= \begin{cases} d_k + \frac{1}{2} (o_k + o_{k+1}) & o_k + o_{k+1} \text{ even} \\ d_k + \frac{1}{2} + \frac{1}{2} (o_k + o_{k+1}) - \frac{1}{2}, & o_k + o_{k+1} \text{ odd} \end{cases} \\ &= e_k \end{aligned}$$



► So

$$d_k^* = e_k - \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor = \begin{cases} d_k, & o_k + o_{k+1} \text{ even} \\ d_k + \frac{1}{2}, & o_k + o_{k+1} \text{ odd} \end{cases}$$

and

►

$$\begin{aligned} e_k^* &= d_k^* + \lfloor \frac{1}{2} (o_k + o_{k+1}) \rfloor \\ &= \begin{cases} d_k + \frac{1}{2} (o_k + o_{k+1}) & o_k + o_{k+1} \text{ even} \\ d_k + \frac{1}{2} + \frac{1}{2} (o_k + o_{k+1}) - \frac{1}{2}, & o_k + o_{k+1} \text{ odd} \end{cases} \\ &= e_k \end{aligned}$$



These formulas are easy to code in *Mathematica*. Let's have a look at the notebook

```
LiftingJPEG2000.nb
```

